Нестандартное использование ARP

Обновлена 16.08.01

В данной статье описывается для
чего нужен и как можно использовать
ARP протокол

*Автор не несет никакой ответственности за ущерб, который может быть нанесен после прочтения данной статьи. Материал предоставляется **ИСКЛЮЧИТЕЛЬНО** в учебных целях.*

Прежде всего мне хотелось бы немного рассказать о том, как в локальной сети пакеты передаются от одного интерфейса к другому. В соответствии с семиуровневой моделью OSI уровень связи данных (**Data link layer**) организует данные в кадры (*frame*). Иногда ее называют канальным уровнем. Каждый кадр имеет заголовок (*header*), содержащий адрес и управляющую информацию, а завершающая секция кадра (*trailer*) используется для исправления ошибок (иногда ее называют хвостом кадра). Кадр доставляется на сетевой адаптер, физический адрес которого совпадает с физическим адресом назначения из заголовка кадра. Заголовок кадра содержит физический адрес интерфейса назначения, часто называемый MAC-адресом (от **Media Access Control** - управление доступом к носителю). Система с указанным адресом принимает посланное сообщение и обрабатывает его.

Структура MAC кадра для DIX

MAC - кадр	IP датагарамма	FCS (контрольная сумма MAC кадра)

Поля MAC кадра для DIX Ethernet

00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D
Dst MAC							Src MAC					Tp	

Обозначения:
Dst MAC - Destination Hardware Address
Src MAC - Source Hardware Address
Tp - Protocol Type

Таким образом, для доставки датаграммы в локальной сети нужно определить физический адрес узла назначения. Именно для этого существует процедура автоматического определения физических адресов. Протокол разрешения адресов (**Address Resolution Protocol** - ARP) обеспечивает метод динамической трансляции между IP-адресом и соответствующим физическим адресом на основе широковещательных рассылок.
Система локальной сети самостоятельно использует ARP для исследования информации о физических адресах (хотя есть возможность при необходимости вручную ввести ARP-таблицу). Когда хосту нужно начать коммуникацию с другим хостом в локальной сети, он ищет IP-адрес получателя в ARP-таблице, которая обычно располагается в оперативной памяти. Если для нужного IP-адреса не находится требуемого элемента таблицы, хост посылает широковещательный запрос ARP, содержащий искомый IP-адрес назначения. Отметим что MAC адрес для широковещательной рассылки **FF:FF:FF:FF:FF:FF**.
Целевой хост узнает свой IP-адрес и обрабатывает запрос. После этого он изменяет собственную таблицу трансляции соответствия IP и MAC адресов, включая в нее IP-адрес и физический адрес отправителя широковещательного ARP-пакета и посылает ответ, содержащий аппаратный адрес своего интерфейса. Когда система, пославшая запрос, получает такой ответ, она обновляет свою таблицу ARP и становится готовой к пересылке данных по локальной сети.

Формат сообщений ARP	
Количество октетов	Поле
2	Тип аппаратного адреса
2	Протокол адресации высокого уровня
1	Длина аппаратного адреса
1	Длина адреса высокого уровня
2	Тип сообщения: 00 01 - запрос, 00 02 - ответ
*	Аппаратный адрес источника
*	Адрес высокого уровня источника
*	Аппаратный адрес приемника
*	Адрес высокого уровня приемника

Пример кадра с ARP запросом																
Offset	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
0000:	FF	FF	FF	FF	FF	FF	00	40	95	01	59	01	08	06	00	01
0010:	08	00	06	04	00	01	00	40	95	01	59	01	0A	00	00	06
0020:	00	00	00	00	00	00	0A	00	00	01						

Расшифровка пакета полученная с помощью сниффера NetXRay		
Ethernet Version II		
Address	00-40-95-01-59-01	FF-FF-FF-FF-FF-FF
Ethernet II Protocol Type	ARP (08 06)	
Address Resolution Protocol		
Hardware Type	Ethernet (00 01)	
Protocol Type	IP Protocol (80 00)	
Hardware Address Lenght	6 bytes (06)	
Protocol Address Lenght	4 bytes (04)	
Operations	ARP Request (00 01)	
Source Hardware Address	00-40-95-01-59-01	
IP Source Address	10.0.0.6 (0A 00 00 06)	
Destination Hardware Address	00-00-00-00-00-00	
IP Destination Address	10.0.0.1 (0A 00 00 01)	
Calculate CRC: 0x6ffd5fa9		

Данный ARP-запрос был "пойман" после того как была запущена программа *ping 10.0.0.1*. В ARP-таблице не было данного IP-адреса - поэтому был послан запрос. Отметим что поле **Destination Hardware Address** неизвестно - поэтому заполнено нулями (**00:00:00:00:00:00**).
Как уже говорилось выше, данные о соответствии IP адреса и физического адреса хранятся в ARP-таблице. Просмотреть ее содержимое можно командой *ARP*.

> *ARP -a* выводит список всех закешированных элементов
> *ARP -a <IP> [IA]* выведет соответствие MAC для указанного *<IP>* для интерфейса *IA*
> *ARP -d <IP> [IA]* удаляет существующий элемент с *<IP>* из таблицы для интерфейса *IA*
> *ARP -s <IP> <MAC> [IA]* добавляет элемент для интерфейса *IA*

Как можно нестандартно использовать ARP протокол? Ответ прост - дело в том, что большинство операционных систем ARP-ответ заносят сразу же без проверки (посылала ли система запрос) в ARP таблицу (исключением является Solaris, который игнорирует ARP ответы, если не посылался ARP запрос). Почему в остальных ОС не применен такой же метод, остается загадкой.
Чем грозит такая ситуация? Если интерфейс получит данные о том, что какому-то IP соответствует MAC, который на самом деле не существует, то IP-датаграммы к данному хосту будут вкладываться в кадр с фальшивым MAC. Это приведет к тому, что ни один сетевой интерфейс в локальной сети не будет воспринимать этот пакет. Таким образом, все данные уйдут в "никуда".

Для реализации подобной атаки необходимо периодически (интервал зависит от операционной системы) посылать ложные ARP ответы. В результате атакуемый хост не сможет установить соединение с хостом, IP адрес которого указан в ложном ARP ответе. Данная атака применима даже если уже установлено соединение между двумя хостами. После посылки даже одного ложного ARP ответа соединение будет разорвано по таймауту. Для того чтобы два хоста не смогли обмениваться пакетами друг с другом, необходимо направить ложные ARP ответы на один из хостов. Если же ставить целью полностью отключить хост от сети, то необходимо периодически посылать ложные ARP ответы от всех хостов в сети. Тогда на атакуемом хосте сложится впечатление, что поврежден кабель или вышла из строя сетевая карта. Однако это впечатление легко рассеять, достаточно лишь запустить сниффер и убедиться что интерфейс работает, и пакеты отправляются и получаются.

Демострация атаки:

на атакаемом хосте запускаем *ping* с ключом *-t* (до прерывания пользователем). Через некоторое время посылается ложный ARP-ответ, а затем ARP ответ с правильным MAC-адресом. Посылать ARP-ответ можно либо с помощью сниффера **NetXRay**, либо специально написанными для этого программами. Вот что будет результатом:

>ping -t 10.0.0.1

Обмен пакетами с 10.0.0.1 по 32 байт:

Ответ от 10.0.0.1: число байт=32 время<10мс TTL=128
Ответ от 10.0.0.1: число байт=32 время<10мс TTL=128
Ответ от 10.0.0.1: число байт=32 время<10мс TTL=128
Ответ от 10.0.0.1: число байт=32 время<10мс TTL=128
Превышен интервал ожидания для запроса.
Превышен интервал ожидания для запроса.
Превышен интервал ожидания для запроса.
Превышен интервал ожидания для запроса.
Ответ от 10.0.0.1: число байт=32 время<10мс TTL=128
Ответ от 10.0.0.1: число байт=32 время<10мс TTL=128
Ответ от 10.0.0.1: число байт=32 время<10мс TTL=128
Ответ от 10.0.0.1: число байт=32 время<10мс TTL=128

Как видим, возникает таймаут посланных пакетов. А какая будет ситуация, если послать только один ARP-ответ с несуществующим MAC в данном случае? В этом случае хост не сможет установить соединение с хостом с IP-адресом 10.0.0.1 до тех пор пока не будет прервано выполнение команды *ping*. Дело в том, что при попытке послать данные другим приложением по этому же IP-адресу, в кеше ARP-таблицы будет найдено соответствие MAC и IP (повторяющаяся команда *ping* не дает этим данным устареть). Если же мы прекратить пинговать хост, то через некоторое время (для многих систем 35-40 секунд) элемент ARP-таблицы будет удален из нее, и при последующих попытках какого-нибудь приложения установить соединение или послать датаграмму к хосту, будет послан ARP-ответ, и в таблицу соответствия MAC и IP адреса занесутся верные данные.
Как защитить себя от данной атаки? Необходимо статически прописать элементы в ARP таблице. Это не совсем удобно (учитывая то, что в локальной сети может быть много хостов), но тогда можно быть уверенным, что данная атака на ваш хост не принесет желаемого результата для атакующего. Однако определить какой именно хост является атакующим не представляется возможным.
Есть одна особенность, с которой столкнулся автор статьи в процессе экспериментов со статическими записями в ARP-таблице на Windows NT 4.0 (Service Pack 5). Попробуем добавить статическую запись, а потом попробовать снова послать ложный ARP-ответ.
Добавляем элемент:

arp -s 10.0.0.1 00-95-40-20-40-01

Проверяем список элементов:

arp -a

Интерфейс: 10.0.0.6 on Interface 2
Адрес IP Физический адрес Тип
10.0.0.1 00-95-40-20-40-01 статический

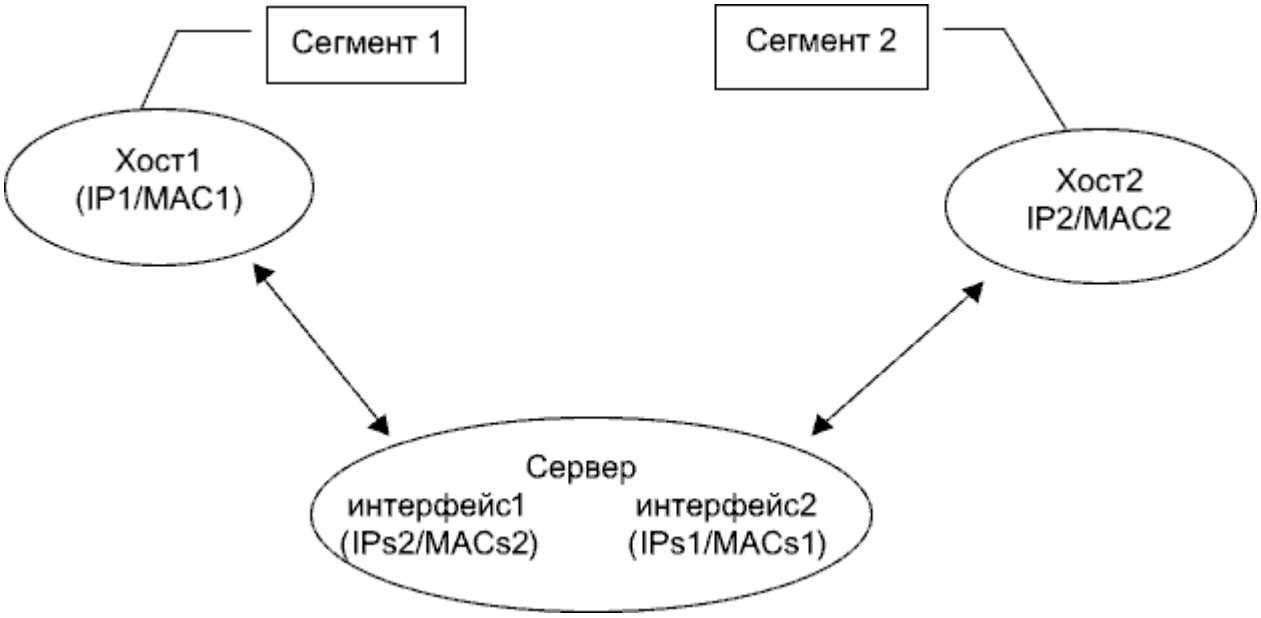
По идее, все в порядке - элемент добавлен, и он является статическим. То есть, он не должен изменяться ни при каких обстоятельствах. Проверим это на деле - посылаем ложный ARP-ответ (*Src IP: 10.0.0.1, Src MAC: 00-11-22-33-44-55, Dst IP: 10.0.0.6, Dst MAC: 00-40-95-01-59-01*). Затем проверяем таблицу записей:

arp -a

Интерфейс: 10.0.0.6 on Interface 2
Адрес IP Физический адрес Тип
10.0.0.1 00-11-22-33-44-55 статический

Что получается? Элемент является статическим, т.е. он не будет удален из ARP-таблицы через некоторое время, а MAC-адрес поменялся после получения ложного ARP-ответа! Аналогичная особенность присутствует и на Windows 95/98/Me, и даже на Windows 2000 (Professional, Advanced Server). Из этого можно сделать вывод, что защитить себя, прописав статические элементы на данных ОС, к сожалению, нельзя.
Использование персональных файрволов, таких как Tiny Personal FireWall, AtGuard, OutPost - тоже не сможет защитить от данной атаки. В этих брендмауэрах анализируются IP-датаграммы (TCP, UDP, ICMP протоколы). Очень жаль, что нет мониторинга ARP-пакетов. Будем надеяться, что в последующих версиях появятся эти функции.
Однако существует такой персональный брендмауэр как **Conseal**, который позволяет устанавливать правила для протоколов IPX, ARP, ну и конечно TCP/IP. Используя данный файрвол можно оградить себя от ARP-атак. Для этого необходимо занести в таблицу соответствия MAC и IP адресов статические записи для всех хостов в сети. Дальше в Conseal установить правила, запрещающие прием ARP-ответов. Теперь можно будет с уверенностью сказать, что хост неподвержен ARP-атаке.
Еще одной атакой, в основу которой положены слабые места ARP-протокола, является создание "ложного ARP сервера". Представим себе сеть, разделенную на сегменты, которые соединяются сервером, имеющим два интерфейса. Пакеты из одного сегмента, предназначенные для хостов из другого сегмента пересылаются через сервер.

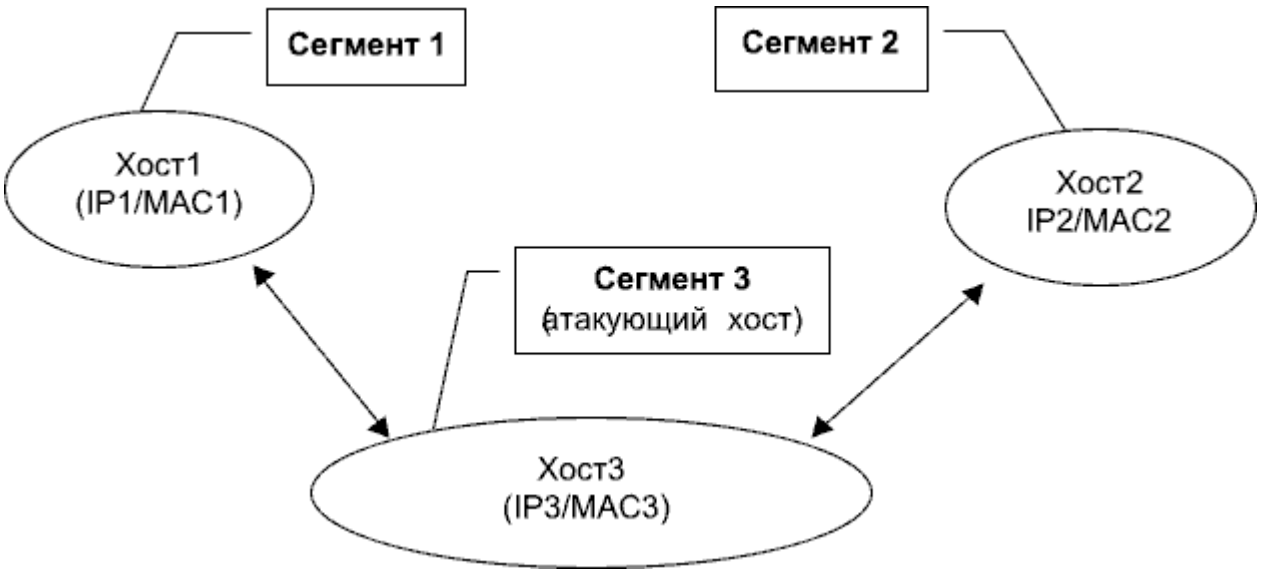
Схема 1



Для того чтобы *хост1* отправил датаграмму IP *хосту2*, необходимо переслать сначала ее на *интефейс1* сервера, а затем датаграмма будет отправлена с *интерфейса2 хосту2*. Как это происходит: IP-датаграмма укладывается в MAC-кадр, в котором *Src MAC=MAC1, Dst MAC=MACs1*.

При получении *интерфейсом1* на сервере, определяется что датаграмма предназначена для *хоста2*. IP-датаграмма укладывается в MAC-кадр, в котором *Src MAC=MACs2, Dst MAC=MAC2* и отправляется. Данный кадр, отправленный с *интерфейса2* сервера будет получен *хостом2*. Этот пример приведен для того чтобы яснее представить атаку "ложный ARP-сервер". Используя данную атаку возможно сниффить траффик между хостами, находящимися в разных сегментах. Данный вид атаки позволяет сниффить траффик даже если сеть разделена на сегменты свитчами. Итак, имеем сеть, поделенную на сегменты свитчами или хабами.

Схема 2



При обычных условиях *хост3* (атакующий) не может прослушивать траффик между хостами *хост1* и *хост2*. Для создания ложного ARP-сервера сначала необходимо *хосту3* послать два ложных ARP-ответа *хост1* и *хосту2*:

1. ARP ответ направленный хосту1	
MAC-кадр	**Src MAC:MAC3**
	Dst MAC:MAC1
ARP-Ответ	**Src IP :IP2**
	Src MAC:MAC3
	Dst IP :IP1
	Dst MAC:MAC1
2. ARP ответ направленный хосту2	
MAC-кадр	**Src MAC:MAC3**
	Dst MAC:MAC2
ARP-Ответ	**Src IP :IP1**
	Src MAC:MAC3
	Dst IP :IP2
	Dst MAC:MAC2

Таким образом после получения хостами указанных пакетов, в ARP-таблицах появятся следующие элементы:

хост1: IP2:MAC3
хост2: IP1:MAC3

Вследствие этого *хост1* все пакеты, предназначенные для *хост2* будет направлять по MAC-адресу *MAC3*, тоже самое будет делать и *хост2* с пакетами для *хост1*.
Теперь *хосту3* необходимо "слушать" весь траффик и следующим образом обрабатывать такие

пакеты

- получив (Ethernet) пакет с *MAC1->MAC3(IP1->IP2)* сменить в этом пакете *MAC3->MAC2(IP1->IP2)* и отправить пакет;
- получив (Ethernet) пакет с *MAC2->MAC3(IP2->IP1)* сменить в этом пакете *MAC3->MAC1(IP2->IP1)* и отправить пакет.

Кроме того, необходимо периодически посылать ложные ARP-ответы хостам *xocm1* и *xocm2* - для чтобы созданные записи в ARP-таблице не "устаревали".
При выполнении указанных действий через *xocm3* будут проходить все пакеты между *xocm1* и *xocm2*. Достаточно будет запустить сниффер и поставить его в режим записи всего траффика, с целью последующего анализа соединений между хостами, для выявления нужной информации.
Хочется добавить, что каждый пакет, проходящий через *xocm3*, будет повторяться дважды - один раз - когда он приходит от хоста-отправителя, и второй - когда он будет отправлен по месту назначения.
Как защитить себя и свою сеть, а также вовремя обнаружить данную атаку? Необходимо статически прописать элементы ARP-таблицы к тем хостам, куда передаются важные данные (POP3-пароли, telnet сессия и т.д.), чаще всего в локальной сети это default gateway, а также хосты, куда вы заходите telnet'ом (опять же это не спасет положения на Windows-платформах). Также можно жестко прописать соответствия ARP-IP в таблице на коммутаторе и запретить ее обновление во время работы. Обнаружить атаку можно просмотрев закешированные данные ARP-таблицы и сравнить их с реальными данными. При обнаружении несовпадения с реальными данными, к примеру, у вас в таблице хост имеет MACx, а на самом деле у него MACs. Это значит, что вас атакуют с хоста, имеющего MACx. Обнаружить атакующего не составляет труда.
Очень часто в локальных сетях при необходимости ограничить доступ к определенному хосту или для идентификации хоста администраторы использует именно MAC-адрес сетевого адаптера. IP-адрес изменить не составляет труда (при наличии прав администратора на Windows NT), а вот решение проблемы свободного изменения MAC адреса для Windows NT (универсального метода, рабатающего на абсолютно всех адаптерах) автор еще не встречал. Естественно, вы можете возразить, что поменять MAC-адрес можно непосредственно в настройках сетевой карты (или ключа в реестре). Да, действительно, существуют подобные сетевые карты, к которым прилагаются такие драйвера, но это возможно не на всех сетевых адаптерах. К примеру, на сетевых картах Realtek8029, WinBond, 3Com можно изменять MAC-адрес, устанавливая соответствующие ключи в реестре:
Для win9x:

[HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Class\Net\0000] - здесь необходимо создать 2 строковые переменные (string):

"NetworkAddress"="xx-xx-xx-xx-xx-xx"
"SelectedID"="xx-xx-xx-xx-xx-xx"

где xx-xx-xx-xx-xx-xx - любой MAC-адрес, который будет душе угоден.
Для winNT:

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\\Parameters]

"NetworkAddress"="xx-xx-xx-xx-xx-xx"
"SelectedID"="xx-xx-xx-xx-xx-xx"

Во всех остальных случаях, когда имеется сетевая карта, на которой таким образом не удается изменить MAC-адрес - как поменять на ней "жестко" зашитые данные? Оказывается и на ней подменить собственный MAC-адрес все же можно.
Автору известны два метода. Начнем с наиболее трудного.
Идея первого очень похожа на создание ложного ARP-сервера. Только на этот раз мы будем обманывать собственный интерфейс. Для реализации необходимо иметь два интерфейса. Пакеты, приходящие на первый интерфейс мы будем "укладывать" в желаемый MAC-адрес и отправлять его с другого интерфейса.
Дальше у нас есть выбор как это реализовать. Во-первых, мы можем установить дополнительный интерфейс, установить маршрутизацию пакетов и использовать этот интерфейс для подключения нескольких или одного хостов. Все пакеты, приходящие от них, будут "перекладываться" на уже

имеющийся интерфейс и отправляться с него с желаемым MAC-адресом.
Во втором варианте мы можем создать виртуальный интерфейс или выделив реальный интерфейс в отдельную подсеть и установив маршрутизацию пакетов, либо оставить все как есть и настроить только роутинговую таблицу. Этот метод проще, поэтому на нем и остановимся. Поняв его, можно будет с легкостью подменить MAC-адрес и с помощью двух реальных интерфейсов.
Установить виртуальный интерфейс в Windows NT не составляет труда. Он называется **Адаптер MS loopback** и устанавливается как обычный адаптер.
При его конфигурации необходимо настроить его также как имеющийся интерфейс, только надо присвоить ему любой неиспользуемый IP-адрес.
Теперь необходимо настроить таблицу роутинга. Смотрим имеющуюся схему маршрутизации пакетов:

>route print

```
===========================================================================

Список
интерфейсов
0x1                         MS TCP Loopback interface
0x2              20 4c 4f 4f 50 20   MS LoopBack Driver
0x3              00 40 95 01 59 01   Winbond W89C940 PCI Network Adapter

===========================================================================
===========================================================================

Активные маршруты:
Сетевой адрес         Маска          Шлюз       Интерфейс  Метрика
          0.0.0.0          0.0.0.0      10.0.0.7      10.0.0.6        1
          0.0.0.0          0.0.0.0    10.0.0.250      10.0.0.7        1
         10.0.0.0    255.255.255.0      10.0.0.6      10.0.0.6        1
         10.0.0.0    255.255.255.0      10.0.0.7      10.0.0.7        1
         10.0.0.6    255.255.255.0     127.0.0.1     127.0.0.1        1
         10.0.0.7    255.255.255.0     127.0.0.1     127.0.0.1        1
   10.255.255.255  255.255.255.255      10.0.0.7      10.0.0.7        1
        127.0.0.0        255.0.0.0     127.0.0.1     127.0.0.1        1
        224.0.0.0        224.0.0.0      10.0.0.6      10.0.0.6        1
        224.0.0.0        224.0.0.0      10.0.0.7      10.0.0.7        1

===========================================================================
```

где:
0x1:MS TCP Loopback interface
0x2:20 4c 4f 4f 50 20 (10.0.0.7):MS LoopBack Driver
0x3:00 40 95 01 59 01 (10.0.0.6):Winbond W89C940 PCI Network Adapter

Что имеем? Расшифровка построчно
default gateway для интерфейса 3 - *10.0.0.7*
default gateway для интерфейса 2 - *10.0.0.250* <-- это default gateway
пакеты к адресам *10.*.*.** отправляются с интефейса 3
пакеты к адресам *10.*.*.** отправляются с интефейса 2
пакеты отправляемые на самих себя отправляются на *127.0.0.1*

пакеты отправляемые на самих себя отправляются на *127.0.0.1*

все остальное не представляет интереса и не рассматривается. Для реализации нашей идеи необходимо удалить строку 3, таким образом, все пакеты с локальными адресами (10.0.0.*) будут отправляться с виртуального интерфейса. Удаляем маршрут:

>route delete 10.0.0.0 if 3

Что необходимо для подмены MAC-адреса? Обозначим для удобства:

10.0.0.6	*IPr*
00 40 95 01 59 01	*MACr*
10.0.0.7	*IPv*
20 4c 4f 4f 50 20	*MACv*

MACn/IPn - MAC/IP хоста с которым будет устанавливаться соединение
MACw - желаемый MAC

пакеты отправленные с интерфейса 3 будут иметь следующий вид

MAC-кадр	**Src MAC: MACr**
	Dst MAC: MACv
IP-датаграмма	**Src IP: IPr**
	Dst IP: IPn
	Data

Необходимо получив виртуальным интерфейсом пакет с таким кадром переложить IP-датаграму в MAC-кадр и отправить с интерфейса 3:

MAC-кадр	**Src MAC : MACw**
	Dst MAC : MACn
IP-датаграмма	**Src IP : IPr**
	Dst IP : IPn
	Data

а получив на интерфейс 3 такой MAC-кадр:

MAC-кадр	**Src MAC : MACw**
	Dst MAC : MACn
IP-датаграмма:	**Src IP : IPn**
	Dst IP : IPr
	Data

переложить в следующий MAC-кадр и отправить с виртуального интерфейса:

MAC-кадр	**Src MAC : MACv**
	Dst MAC : MACr
IP-датаграмма	**Src IP : IPn**
	Dst IP : IPr
	Data

Отправлять такие пакеты можно при условии что хост, от "имени" (MAC) которого будут отправляться пакеты должен быть неактивным. Для того, чтобы имитация другого хоста была полной, необходимо также сменить имя компьютера.
Все выглядит сложным, но, разобравшись написать такую программу, не составило труда.
Второй метод смены MAC-адреса намного проще. Первый способ требует написания программы работающей на низком уровне, в то время как сменить MAC-адрес можно используя лишь отладчик **SoftIce**. Дело в том, что драйвер сетевой карты считывает MAC-адрес через стандартные порты ввода-вывода только на этапе инициализации. Перехватив этот момент и подменив реальные данные на желаемые, можно добиться желаемого результата. Использовать данный метод, во-первых, удобнее, а во вторых это не требует затрат времени на подмену (драйвер будет просто работать как и обычно). В первом способе каждый принятый/отправленный пакет, должен обработаться еще программой, что не дает возможным использование его в 100 мега-битной сети (пакеты не будут успевать обрабатываваться).
Что нужно для этого метода? Как уже говорилось необходимо иметь отладчик **SoftIce** или любой другой, который будет загружаться до загрузки системы и позволяющий отлаживать драйвера устройств. В **SoftIce** это режим *"Boot"*. Теперь необходимо определить диапазон ввода-вывода сетевой карты. К примеру, на Winbond W89C940 это E400-E41F. Чтение MAC-адреса происходит через порт E410h (начало диапазона плюс 10h). Именно этот порт и нужно контролировать. Запоминаем собственный MAC-адрес (так как через порт может считываться не только он - передается также и служебная информация, к примеру название чипсета).
Устанавливаем режим отладчика *"Boot"* и перезагружаемся. После загрузки SoftIce, входим в его консоль (по умолчанию комбинация ctrl-D) и устанавливаем точку останова при попытке чтения из порта E410h:

:bpio e410 r

затем выходим:

:x

и даем системе грузиться дальше. После того как происходит попытка чтения из указанного порта (при инициализации драйвера сетевой карты) попадаем в консоль **SoftIce**. Теперь необходимо определить что считывается - MAC-адрес или это другая информация. Сравниваем первый байт MAC-адреса с содержимым регистра AL и при совпадениии меняем его на нужное значение. Если это не он, то пропускаем и ждем следующего чтения (F5). 6 байт MAC-адреса будут считываться последовательно, но нужно учесть, что считывание происходит дважды. Т.е. заменив один раз все 6 байт, трассируем до тех пор, пока MAC-адрес не будет считываться повторно. Подменив его и во второй раз, можно очистить точку останова и дать системе загрузиться полностью. Для этого в консоли **SoftIce** пишем:

*:bc **
:x

и выходим с консоли. После загрузки системы драйвер сетевой карты будет работать с новым MAC-адресом. Убедиться в этом можно выполнив команду:

nbtstat -A <host_IP>

Данная операция занимает не более 5 минут. Сложно будет только в первый раз, когда неизвестно в какой именно момент считывается MAC-адрес.
Как обнаружить, что кто-то одним из указанных методов подменил MAC-адрес? Сложно сказать, потому как, анализируя сетевой траффик, никаких аномалий не наблюдается. Можно предложить следующие дополнительные проверки, которые можно сделать, к сожалению, только вручную (проверку MAC-адреса может контролировать программа) - проверка общих ресурсов (если у настоящего хоста их достаточно много, то имитировать их будет довольно сложно), а также определение удаленной ОС (хотя можно проводить атаку с хоста имеющего идентичную ОС). Из всего этого следует сделать вывод, что полагаться только на идентификацию истинности хоста только на основании соответствия MAC-адреса - нельзя.

Хочется отметить, что все вышеописанные атаки имеют воплощение в виде программ, которые по

понятным причинам не выставляются для всеобщего тестирования. Большая просьба отнестись к этому с пониманием и не присылать письма с просьбой отправить эти программы. При большом желании вы можете написать их сами - пусть это будет еще один повод заняться написанием программ.
Для написания программ был использован драйвер Packet capture driver by **Netgroup of Politecnico di Torino PCCap**, доступный здесь:
http://netgroup-serv.polito.it/winpcap/

Литература:
TCP/IP "Architecture, protocols, and implementation with IPv6 and IPv4 security"
Dr. Sidnie Feit

PDF-версия (скачано 279 раз).

u$e youR PoWeR iN GooD, NoT eViL
Be$T ReGaRD$ FRoM
-=<ZaDNiCa>=-

$PeCiaL THaNK$ 2:
Nick, LeD0RuB, NiFi [uinC], ERRor, KMiNT21
Написано специально для uinC