Исследование возможностей перенаправления пакетов в протоколах ARP and ICMP

(c) Yuri Volobuyev 1997

Это баг и это особенность
(There're bugs and there're features - оригинал)

Как это часто бывает, различия неотличимы. Я рассматриваю 2 официальных протокола модели TCP/IP-ARP и ICMP, которые будучи внимательно рассмотрены, могут использоваться для достижения определённых целей.

На данный момент (1997 г. - прим. переводчика) большее распространение получили пассивные типы атак (сетевой сниффинг) для получения root-доступа в локальной сети, активные атаки приобрели меньшее распространены. Однако, разновидности активных атак в LAN могуть представлять большую опасность для системных администраторов. Технические аспекты атаки могут быть не вполне очевидны, поэтому давайте остановимся на них.

Варианты атаки, рассматриваемые в данном документе, включают спуффинг (spoofing-подмена одного из участников сетевого соединения, прим. переводчика) и DoS (Denial Of Service - отказ в обслуживании). IP-blind spoofing (я бы перевёл как слепая подмена IP-адреса) - наиболее общий и очень мощный способ атаки, требующий больших усилий и являющийся достаточно сложным в части реализации. В отличие от него ARP-спуффинг достаточно прост.

Поскольку ARP-spoofing возможен только в локальной сети, он является очень серьёзным способом для расширения границ сетевой атаки в случае получения доступа к одной из машин сети.

[--Основы протокола ARP--]

Автор пишет, что он мог бы углубиться в описание протокола ARP, но вместо этого рекомендует книгу W.Richard Stevens "TCP/IP Illustrated". (Прим. переводчика - ZDNC довольно подробно описал нюансы этого протокола в своей статье). Протокол описан в RFC826.

[--Описание реализации атаки--]

Давайте представим гипотетическую сеть.

IP	10.0.0.1	10.0.0.2	10.0.0.3	10.0.04
hostname	CAT	RAT	DOG	BAT
hwaddr	AA:AA	BB:BB	CC:CC	DD:DD (для упрощения)

Все машины соединены простейшим способом (коаксиал, нет коммутаторов и интеллектуальных концентраторов). Вы находитесь на машине CAT рутом. Ваша задача - стать для остальных клиентов сети машиной DOG. Вы знаете, что DOG доверяет хосту RAT, так что Ваша задача - подставить в сетевом протоколе свою машину вместо RAT.

Первая мысль, которая приходит в голову - "почему я не могу установить IP-адрес другой машины и..." Но это не будет работать, по крайней мере это не будет работать надёжно. Если Вы скажете Ethernet-драйверу на CAT, что его IP-адрес 10.0.0.2, он станет отвечать ARP-ответами на этот IP. Но таким же образом будет себя вести и RAT. В данной ситуации не выиграет никто (в том числе и Вы). Подобная ситуация случается иногда в локальной сети при неправильной сетевой конфигурации машины (если ей назначается работающий IP-адрес). Многие сетевые программы (в том числе анализаторы сетевого траффика) моментально обнаруживают подобные вещи. Кроме того на консоли сетевого администратора может появиться сообщение, что такой-то MAC использует такой-то IP. То есть Вы не достигли того, чего хотели.

Программа send_arp.c (прилагаемая к данному тексту) может быть

использована для решения поставленной задачи. Как видно из её названия, она посылает ARP-пакет (ARP-ответ, если быть точным: как следует из определения протокола, ответ будет принят успешно даже если его никто не запрашивал) в сеть, и Вы можете делать с данным пакетом всё, что Вы хотите. Посылая пакет при помощи данной программы, необходимо указать IP-адрес источника посылки и цели атаки и их hw-адреса (MAC).

Прежде всего необходимо "заглушить" свой Ethernet-драйвер. Это можно сделать при помощи команды ifconfig -arp. Разумеется, драйвер всё равно нуждается в ARP-информации (обмен пакетами ведётся на канальном уровне OSI-модели посредством включения в заголовки пакетов MAC-адресов), но этого можно достичь, указав ядру ARP-информацию вручную при помощи команды arp (8). Критическая часть нашего замысла - убедить соседей (по сети) в достоверности передаваемой информации. В случае, описанном здесь, Вы хотите, чтобы DOG верил, что MAC-адрес RAT на самом деле равен MAC-адресу CAT (AA:AA). Поэтому вы посылаете ARP-ответ с источником IP-адреса 10.0.0.2, hw-адресом источника AA:AA, IP-адресом цели 10.0.0.3 и hw-адресом CC:CC. Сейчас DOG знает, что RAT-это AA:AA. ARP-кэш, конечно, необходимо периодически обновлять (поскольку в противном случае ARP-запросы начнёт посылать сама машина, что не входит в наши планы). Периодичность часто зависит от конкретной ОС, но как выяснилось, посылки пакета один раз в 40 сек в большинстве случаев бывает достаточно. Вы можете посылать пакет и чаще - это не повредит.

Сложность здесь связана с особенностью механизма ARP-кэширования. Некоторые ОС (например, LINUX) будут пытаться обновить кэш-таблицу, посылая unicast (однонаправленные) пакеты к кэшируемым адресам. Так как такой запрос, направленный к ИСТИННОМУ RAT может помешать нам в наших планах, его необходимо предупредить. Этого можно достичь, зафлудив систему, которая могла послать ARP-запрос, ложными ARP-ответами таким образом, что она никогда не пошлёт ARP-запрос. Предупреждение, как всегда, является лучшим лекарством. В то время, как настоящий пакет от DOG к RAT должен быть послан, посылается ложный пакет с ARP-ответом от CAT. Как говорилось выше, периодичности в 40 сек. бывает достаточно.

Итак, процедура довольно простая. "Поднимаем" (bring up) алиас-интерфейс, т.е. eth0:1 (или используем текущий, не имеет значения) с IP-адресом RAT и включённым ARP-поскольку необходимо заполнить кэш-таблицу сначала, а это невозможно с выключенным ARP. Устанавливаем маршрут для DOG через правильный интерфейс. Устанавливаем в кэш-таблице запись для DOG и затем отключаем ARP. Теперь у нас всё установлено.

После этого запускаем прилагаемую программу send_arp для DOG и для RAT и теперь DOG уверен, что на самом деле Вы RAT. В дальнейшем не забываем периодически посылать ARP-пакеты для DOG и RAT.

Описанная атака работает, конечно, только в локальной сети (вернее, граница атаки зависит от дальности прохождения ARP-пакетов, которые почти никогда не маршрутизируются). Но интересное применение возможно, если в приведённом выше примере заменить DOG hw-адрес на hw-адрес маршрутизатора. Если это сработает (я не уверен, что это всегда будет работать, ARP-реализация на маршрутизаторах может быть достаточно глупой), Вы можете легко подменить любую машину в локальной сети для всего остального мира (Интернета, имеется в виду - прим. переводчика). Таким образом целевая машина мрожет быть где угодно, но машина, которую Вы хотите подментить должна быть в одном с Вами сегменте сети.

Исследования различных сетевых ОС (источник - книга "Атака на Интернет") выявили, что в ОС Linux при адресации к хосту, находящемуся в одной подсети с данным хостом, ARP-запрос передаётся, если в ARP-таблице отсутствует соответствующая запись о Ethernet-адресе, и при последующих обращениях к данному хосту ARP-запрос не посылается. В SunOS при каждом новом обращении к хосту (в том случае, если в течение некоторого времени обращения не было) происходит передача ARP-запроса и, следовательно, ARP-таблица динамически обновляется. ОС Windows 95 ведёт себя почти как Linux, за исключением того, что периодически (каждую минуту) посылает ARP-запрос о MAC-адресе маршрутизатора; в результате в течение нескольких минут вся LAN с Win95 без труда берётся под контроль ложным ARP-сервером. ОС WinNT 4.0 также использует динамически изменяемую ARP-таблицу, и

ARP-запросы о MAC-адресах маршрутизатора передаются каждые 5 минут.

[--Что ещё может быть сделано--]

Кроме спуффинга, есть ещё большой диапазон действий, которые могут быть совершены при помощи манипуляций с ARP-протоколом. DoS (Denial Of Service - отказ в обслуживании - прим. переводчика) - наиболее очевидное применение. Наводняя жертву неправильными hw-адресами, можно сделать её полностью "немой". Возможно даже полностью предотвратить связь этой машины с любой (или определённой) машиной сети (и размер ARP-кэша обычно достаточен для заполнения ARP-информацией всех машин LAN). Также очевидной целью атаки могут быть маршрутизаторы. Удар по ARP-кэшу в этом случае также должен быть двухсторонним: обе системы - жертва и машина, с которой Вы хотите отключить связь данной машины, должны быть зафлужены ложными пакетами. В простейшем случае в пакете должен быть установлен несуществующий адрес. Но это не очень эффективно, поскольку машина быстро поймёт, что она говорит с несуществующей машиной и пошлёт обратный ARP-запрос. Конечно, послав со своей стороны следующий ARP-ответ, вы опять "обнулите" результат, но необходимо это делать достаточно часто. Более эффективный путь - наводнить жертву пакетами с hw-адресами машины, которая "жива" и работает в сети. Опять же, это зависит от определённой ситуации, но очень часто случается, что жертва посылает обратно пакеты различных типов, которые достигают неверного места назначения - тогда система, принимающая пакеты, будет генерировать ICMP-пакеты типа 3 "ICMP Xxxx Unreachable" для машины-жертвы, таким образом эмулируя соединения несколько неправильным способом. Подобного рода псевдосоединения могут легко отсрочить время истечения ARP-кэша. В Linux, например, псевдосоединения увеличивают время истечения срока кэша (то есть время, когда будет послан пакет на обновление кэш-таблицы) с обычных 1 мин. до 10 мин. На это время большинство (или все) TCP-соединения оказываются заблокированными (screw up).

Особый интерес представляет так называемый "даровой (gratuitous) ARP". Подобная ситуация возникает, если IP-адреса источника и цели в ARP-запросе равны. Это обычно происходит в случае широковещательного Ethernet-запроса. Некоторые реализации TCP/IP-стека определяют подобную ситуацию, как специальный случай, и система посылает запросы об обновлении информации ко всем сетевым клиентам, обновляя собственный кэш ARP-ответами. В этом случае один пакет может привести к компроментации всей сети. Хотелось бы отметить, что "даровой (gratuitous) ARP" не определён, как часть стандрата ARP, поэтому большинство производителей ОС не применяют его, что придаёт ему меньшую популярность.

ARP - достаточно серьёзный инструмент для всякого рода сетевых атак. Давайте представим, что кто-то установил ретранслятор (релэй), или тоннель на своей машине и убедил две соседние машины посылать пакеты друг другу через Ethernet-карту ретранслятора. Если этот ретранслятор просто пересылает пакеты к их настоящему месту назначения, никто ничего не заметит. Однако, в данной ситуации машина-ретранслятор способна модифицировать передаваемые данные, что представляет собой довольно большую угрозу конфиденциальности (безопасности) передаваемых данных. Простой программный фильтр может подставлять 2 случайных байта в произвольные моменты времени. Контрольная сумма большинства пакетов при этом не будет изменяться, но суммарный приходящий пакет будет искажён.

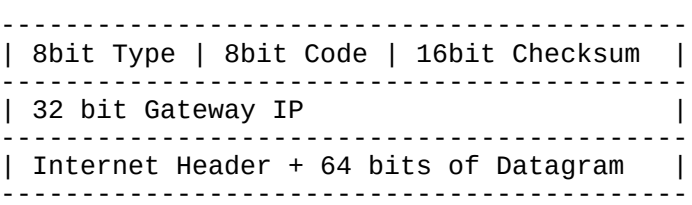
[--ICMP redirects--]

Эффект, подобный описанному выше при работе с протоколом ARP, может быть также достигнут другим путём, используя законную особенность протокола ICMP, а именно возможность посылки сообщения, требующего перенаправления пакетов при маршрутизации. Протокол ICMP (Internet Control Message Protocol - RFC792) предназначен для управления процессом передачи IP-пакетов и формирования сообщений при возникновении ошибок при передаче отправителю датаграммы. Одной из функций этого протокола является управление таблицами маршрутизации на хостах внутри сегмента сети. Сетевая подсистема предназначена для работы в коммуникационной среде, представляющей собой набор сегментов, связанных между собой. связь между отдельными сегментами достигается путём подключения их к хостам, имеющим несколько различных сетевых интерфейсов. Такие хосты (называемые

маршрутизаторами) при необходимости выполняют передачу данных от одного сегмента к другому (forwarding). Для сетей пакетной коммутации выполнение этой задачи непосредственно связано с выбором маршрута прохождения пакетов данных (routing). Для этого система хранит таблицы маршрутизации, используемые протоколами сетевого уровня для выбора требуемого интерфейса для передачи пакета адресату. Информация о маршрутах хранится в виде 2-х таблиц, одна из которых предназначена для маршрутов к хостам, а вторая - для маршрутов к сетям. Посмотреть таблицы маршрутизации можно при помощи команд route (route -n) и netstat -rn.

При определении маршрута модель сетевого протокола (IP) сначала просматривает элементы таблицы для хостов, а затем для сетей. Если оба поиска не дают результата, используется маршрут по умолчанию (обозначенный в таблице в поле Destination как 0.0.0.0). Обычно используется первый найденный маршрут. Таким образом, порядок поиска обеспечивает приоритетность маршрутов к хостам по отношению к маршрутам к сетям, что естественно, поскольку первые представлены более конкретными адресами. Также маршруты подразделяются на прямые (direct) и косвенные (indirect). Маршрут в сеть, непосредственно подключенную к сетевому интерфейсу, является прямым. Маршрут по умолчанию - косвенный, так как адресует получателя, расположенного вне вне непосредственно доступных сетевых сегментов. Время жизни маршрута зависит от протокола верхнего уровня. Например модуль TCP хранит маршрут на протяжении жизни виртуального канала (образуемого при установлении TCP-сеанса).

Перенаправление (изменение маршрута) осуществляется функцией rtredirect(), вызываемой модулем протокола в ответ на получение от соседних шлюзов управляющих сообщений о перенаправлении маршрута. Динамическое управление маршрутизацией изначально задумывалось для предотвращения возможной передачи сообщений по неоптимальному маршруту, а также для повышения отказоустойчивости Сети в целом. Предполагалось, что сетевой сегмент может быть подключён не через один (как обычно), а через несколько маршрутизаторов. В этом случае адресоваться во внешнюю сеть можно через любой из ближайших узлов. При изменении оптимального маршрута или отказе одного из маршрутизаторов необходимо изменение таблицы маршрутизации в памяти операционной системы. Такое динамическое изменение возложено на протокол ICMP. Протокол в этом случае посылает сообщение ICMP Redirect Message следующего формата:



Поля имеют значения: Type-5 (ICMP-тип-Redirect), Code -
0=Redirect datagrams for the Network
1=Redirect datagrams for the Host
2=Redirect datagrams for the Type of Service and Network
3=Redirect datagrams for Type of Service and Host.

Сообщения с кодами 0 и 2 не используются современными ОС.
Сообщение с кодом 1 информирует хост о том, что следует создать новый маршрут к отмеченному в сообщении объекту и внести его в таблицу маршрутизации, указывая IP-адрес хоста, для которого нужна смена маршрута (адрес будет занесён в поле Destination в пристыкованном IP-заголовке), и новый IP маршрутизатора, куда необходимо направлять пакеты для данного хоста (этот адрес заносится в поле Gateway). Что касается поля Type of Service (TOS) в заголовке IP-пакета, оно, по замыслу разработчиков должно было определять приоритет при маршрутизации. На это поле отводится 8 бит. Различные биты представлют собой значимость, задержку, скорость передачи, надёжность. TOS определяет обработку датаграммы при передаче через различные сети от источника к получателю. В большинстве случаев может оказаться невозможным удовлетворить сразу нескольким требованиям при обработке, предусмотренным полем TOS.

Сообщения ICMP Redirect Message в нормальной ситуации посылаются

маршрутизатором по умолчанию к машине для указания более короткого машрута к месту назначения. В исходном виде (RFC) оба переназначения маршрутов (для сети и для хоста) были предложены, но позже перенаправление для сети не нашло применения и сейчас считается равным перенаправлением маршрута для хоста. Правильно сконструированный ICMP-пакет, который прошёл все проверки на достоверность (он должен прийти с маршрутизатора по умолчанию для точки назначения редиректа, новый маршрутизатор должен быть в напрямую соединённой сети и т.д.) вызовет добавление в таблицу машрутизации данного хоста.

---------- Добавление, включённое переводчиком. ----------
Согласно книге "TCP/IP Illusrated" хост с реализацией сокетов 4.4 BSD, который принял ICMP-redirect применяет некоторые правила до модификации таблицы маршрутизации. Это предотвращает неправильное (и неправомерное) изменение таблиц маршрутизации. Такие правила включают следующие условия:
1. Новый маршрутизатор должен находиться на непосредственно подсоединённой сети.
2. Перенаправление должно быть с маршрутизатора по умолчанию для данной точки назначения.
3. Переназначение не может быть назначено с самого хоста как маршрутизатора.
4. Вновь назначаемый маршрут должен быть непрямым маршрутом.

Таким образом, если Вы имеете 2 машины в одной подсети и они обе имеют сетевой маршрут для этой подсети, вы не сможете при помощи ICMP перенаправить маршрут одной через другую. Однако вы будете способны использовать перенаправление для передачи траффика, предназначенного для машин в другой подсети или снаружи сети.

Безопасность ICMP как протокола в данной ситуации равна нулю. Подменить IP-адрес в пакете на IP-адрес маршрутизатора достаточно просто и прилагаемая программа icmp_redir.c делает это. RFC, в которм предъявляются требования к хостам, определяет, что система ДОЛЖНА следовать указаниям ICMP redirect, если пакет исходит от роутера. Многие системы именно так и поступают (за исключением непатченного (vanilla) Linux 2.0.30, где этого не произошло и тестовых 2.0.29 и 2.0.31pre9 Алана Кокса).

ICMP redirect представляет большую потенциальную возможность для DoS. В отличии от ARP-кэша, маршруты хоста не требуют обновления со временем. И конечно, в данном случае не требуется доступа к локальной сети, атака может быть запущена с любого места. Таким образом, если машина-жертва принимает ICMP redirects (и пакеты могут действительно достичь её), то система может прекратить поддерживать связь с любым определённым адресом в сети (не со всеми, но с теми, которые расположены в других физических сетях с жертвой). ДНС-сервера в данном случае могут быть потенциальными жертвами.

Для выполнения межсегментной атаки атакующему необходимо, во-первых, знать внутренний IP маршрутизатора, к которому подключён хост, и, во-вторых, выбрать имя сервера, которому требуется изменить маршрутизацию.

Первая проблема - внутренний IP роутера - решается простым перебором, так как узнать его из внешней сети не представляется возможным (traceroute дает только IP роутера во внешней сети). Так как большинство хостов в сети находится в подсетях класса C, то для осуществления этой атаки достаточно будет послать 254 пакета (0-й и 255-й адрес отпадают по вполне понятным причинам) ICMP Redirect с различными IP отправителя.

Вторая проблема - выбор имени (или IP) сервера, к которому будет изменена маршрутизация. Тут можно просто посылать пакеты с IP известных и часто используемых серверов Internet (поисковые сервера, серверы известных фирм, IRC-сервера и т.д.)

Эксперимент, проведённый авторами книги "Атака на Интернет" показал, что реализовать вариант удалённой атаки типа ICMP Redirect удалось осуществить как межсегментно, так и внутрисегментно на ОС Linux 1.2.8, ОС Win95 и ОС WinNT 4.0. Остальные сетевые ОС (испытывались Linux версии выше 2.0.0 и CX/LAN/SX, защищённая по классу B1 UNIX), проигнорировали ICMP Redirect.

---------- Добавление, включённое переводчиком. ----------
Здесь хотелось бы добавить, что сам Алан Кокс в дискуссии на сервере BUGTRAQ по данной статье указывает, что Юрий неправ. ICMP redirect маршруты для хостов всё-таки не постоянно держатся в таблице маршрутизации, а убираются через некоторе время. Однако если находиться на коммутируемой сети (switched в оригинале) с большой сетевой маской (например сети класса B), можно использовать переназначение ICMP против многих хостов для добавления больше 65000 маршрутов в их таблицы. Таким образом *nix-машины просто израсходуют всё доступоное RAM. Многие "десктоповые" ОС используют линейный алгоритм для поиска маршрутов.

[--Как защититься--]

ARP, как протокол низкого уровня обычно скрыт от большинства людей (работа протокола - прим. переводчика). LAN-администраторы, конечно, представляют себе механизм работы протокола, но обычно уделяют ему мало внимания. Вы всегда можете исследовать содержимое ARP-таблицы используя команду arp(8) в случае возникновения каких-нибудь сетевых проблемм, но это не самое очевидное, что обычно приходит в голову. Даже M$-системы имеют arp-команду и необходимо помнить, что изучение содержимого ARP-таблицы может помочь в некоторых ситуациях. Однако, если Вы цель атаки, источник которой из другой подсети, проникший через подмену ARP маршрутизатора, Вам это ни о чём не скажет. Просто, таблица маршрутизации хоста может быть исследована на предмет опознавания входов, сгенерированных ICMP (в большинстве версий route(1)они маркируются флагом D в поле флагов).

Напомним, что просмотреть ARP-кэш можно командой arp -a (или -an, чтобы предотвратить обращение к ДНС-серверу).

Приведённая выше схема ARP-атаки работает в сетях 10Base2 (коаксиал). Однако, если машины соединены более "продвинутым" способом, используя некоторые "разумные" концентраторы или коммутаторы, атака может быть обнаружена и даже невозможна (то же самое касается и пассивных атак).

Почему я не остановился более подробно на ICMP-атаках? Дело в том, что многие сети имеют очень простую структуру и поэтому нет необходимости для добавления чего-нибудь в таблицы маршрутизации. Во-вторых, во многих сетях обычно устанавливают статические таблицы маршрутизации вручную, поэтому зачем доверять подобное измененение ICMP? И окончательно, ввиду опасности вышеописанной атаки, я даже запретил подобную ситуацию в моей ОС, даже ценой несоблюдения RFC1122. Но это может быть довольно непросто. На Linux или любой другой ОС с открытыми кодами, это может быть сделано на уровне "хака" ядра. На IRIX 6.2 и возможно других версий это достигается выставкой icmp_dropdirects=1 при использовании systune. Также можно достичь хорошего эффекта запретив прохождение пакетов типа ICMP redirect через маршрутизаторы (файрволлы). Как это сделать для Linux описывается во многих примерах настройки пакета ipchains (ipfw, iptables), также настройке фильтрации ICMP-пакетов уделено много места в книге "Брандмауэры в Linux". В качестве примера хотелось бы сказать что в Linux можно вообще очень просто запретить приём ICMP Redirect пакетов включением в какой-нибудь из стартовых скриптов следующей последовательности:

```
for f in /proc/sys/net/ipv4/conf/*/accept_redirects; do
    echo 0 > &f
done;
```

Что касается операционной системы FreeBSD, то реагирование данной ОС на пакеты ICMP redirect перекрывается следующим образом:

```
/etc/rc.conf icmp_drop_redirect="YES"
```

В ARP мы сталкиваемся с ситуацией динамического разрешения IP-адресов без применения централизованного сервера. В протоколе DNS, например, при преобразовании hostname-адресов в IP-адреса сначала проверяется файл /etc/hosts и в случае отсутствия там IP-адреса посылается запрос на ДНС-сервер (хотя порядок опредения можно и изменить). Я не вижу причины, почему бы также не поступить в случае с ARP. Ethernet-адреса не изменяются

очень часто. Кроме того отказ от динамического обновления ARP-таблицы значительно снизит нагрузку сети. Ethernet-карта может быть запущена без включения ARP-режима, а необходимые записи в ARP-таблицу могут быть включены вручную. Кроме того, это уменьшит количество пакетов, передаваемых по сети. Существует также способ брать MAC-адреса из файла /etc/ethers. Еще существет способ статически прописывать ARP-записи в ARP-кэш при помощи опций -f -s команды ARP. Можно, для счастливых пользователей ОС Linux привести пример скрипта на Perl, который статически прописывает ARP-таблицу из предварительно составленного файла. Данный скрипт должен запускаться ПОСЛЕ "поднятия" сетевых интерфейсов.

```
#!/usr/bin/perl
# by John Goerzen
#Program: forcehwaddr
#Program to run ARP to force certain tables.

#Specify filename to read from on command line, or read from stdin.

foreach(<>){              # For each input line...
    chomp;                # Strip if CR/LF
    if(/^#/) { next; }    # If it's a comment, skip it.
    if((($host, $hw)=/\s*(.+?)\s+(\S+)\s*/)==2)&&
        !(/^#/)) {
        # The text between the slashes parses the input line as follows:
        # Ignore leading whitespace. (\s*)
        # Then, start matching and put it into $host($host,(.+?))
        # Skip over the whitespace after that (\s+)
        # Start matching. Continue matching until end of line or optional
        # trailing whitespace.

        # Then, the if checks to see that both a
        # host and a hardware address were matched.
        # (2 matches). If not, we skip the
        # line (assuming it is blank or invalid or something).
        # a poung sign; if so, ignore it (as a comment).

        # Otherwise, run the appropriate command:
        printf("Setting IP %-15s to hardware address %s\n",$host,$hw);
        system "/usr/sbin/arp -s $host $hw\n";
        }
}
```

В состав операционной системы LINUX входит утилита arpwatch, при помощи которой ARP-кэш держит таблицу соответствия MAC и IP-адресов. Из мануала:
Arpwatch keeps track for ethernet/ip address pairings. It syslogs activity and reports certain changes via email. Arpwatch uses pcap(3) to listen for arp packets on a local ethernet interface.

Также, для UNIX есть утилита ARP Wrap, которая предотвращает атаки типа АРП-спуффинга до выполнения программ (telnet, SSH). Из оригинального описания:
Arpwrap is a tool which attempts to detect ARP spoofing attacks before executing a unix command (such as SSH or Telnet).
Скачать Linux-версию этой программы можно здесь:
http://packetstormsecurity.org/linux/security/arpwrap.linux.180701.tgz
Скачать Solaris-версию можно здесь:
http://packetstormsecurity.org/UNIX/security/arpwrap.solaris.190701.tar.gz

==========

Приложение:
Исходные тексты программ
Скачать send_arp.c (скачано 58 раз).
Скачать icmp_redir.c (скачано 46 раз).

Полная версия этой статьи в PDF формате: ARP-ICMP.pdf (скачано 64 раз).

Special Thanks:
1) Господам Медведовскому И.Д., Семьянову П.В., Леонову Д.Г. ака dl за

издание книги "Атака на Интернет", выдержки из которой использовались в качестве многих дополнений.
2) А.Робачевскому за прекрасную книгу "Операционная система UNIX"
3) XR за просмотр первоначального варианта статьи и выдачу пожеланий.
4) KMINT21 за дополнение по защите FreeBSD.

Использованная литература:
1. Статья Ю.Волобуева:
http://www.securityfocus.com/data/library/arp_fun.txt
2. Медведовский И.Д., Семьянов П.В., Леонов Д.Г. - "Атака на Интернет", изд. второе, изд. ДМК, 1999 г.
3. Робачевский А. - "Операционная система UNIX", изд BHV, 1997 г.
4. Роберт Л. Зиглер - "Брандмауэры в Linux", 2000 г.

[c]2001 SOLDIER (перевод и дополнения)
uinC Member
soldier7@mail.ru
[c]uinC

Написано специально для uinC