**Спуфинг**

Спуфинг (от английского слова spoofing) - это кибер-атака, в рамках которой мошенник выдает себя за какой-либо надежный источник, чтобы получить доступ к важным данным или информации. Такая подмена (спуфинг-атака) может происходить через веб-сайты, электронную почту, телефонные звонки, текстовые сообщения, IP-адреса и серверы.

Основная цель спуфинга – получить доступ к личной информации, украсть деньги, обойти контроль доступа к сети или распространить вредоносное ПО через ссылки на зараженные веб-страницы или зараженные файлы, вложенные в электронное письмо / сообщение.

**Спуфинг с подменой номера вызывающего абонента (Caller ID Spoofing)**

Идентификатор вызывающего абонента (Caller ID) позволяет получателю телефонного звонка определить личность того, кто звонит. Такой вид спуфинга происходит в тех случаях, когда мошенник использует ложную информацию для изменения идентификатора вызывающего абонента (т.е. мошенник звонит якобы с другого телефона – например, телефон вашего друга). Поскольку спуфинг с подменой идентификатора вызывающего абонента делает невозможной блокировку номера, многие телефонные мошенники используют такой вид спуфинга, чтобы скрыть свой реальный номер телефона, с которого осуществляется данный звонок, чтобы в конечном итоге скрыть свою личность. Иногда мошенники используют ваш код города, чтобы создать впечатление, что звонок местный.

**Спуфинг с подменой сайта (Website Spoofing)**

Спуфинг с подменой сайта – это тип спуфинг-атаки, в рамках которой мошенник пытается создать опасный (вредоносный) сайт похожим на надежный безопасный сайт (например, известного банка), используя его шрифты, цвета и логотипы. Такой спуфинг проводится путем репликации оригинального надежного сайта с целью привлечения пользователей на специально созданный поддельный фишинговый или вредоносный сайт. Такие «скопированные» сайты, как правило, имеют примерно такой же адрес сайта, как и исходный оригинальный сайт, а потому, на первый взгляд, кажутся обычным пользователям реальными (оригинальными) сайтами. Однако такие «скопированные» сайты обычно создаются для незаконного получения личной информации посетителя сайта.

**Спуфинг с подменой адреса электронной почты (Email Spoofing)**

Спуфинг с подменой электронной почты – это тип спуфинг-атаки, в рамках которой мошенник рассылает электронные письма с поддельными адресами отправителей с намерением заразить ваш компьютер вредоносными программами, заполучить деньги или украсть информацию.

**Спуфинг с подменой IP-адреса (IP Spoofing)**

Когда мошенник стремится скрыть реальное местоположение в Интернете того места, откуда запрашиваются или куда отправляются данные пользователя/жертвы, то обычно используется спуфинг с подменой IP-адреса. Цель IP-спуфинга состоит в том, чтобы заставить компьютер жертвы думать, что информация, отправляемая мошенником пользователю, исходит из надежного источника, что позволяет вредоносному контенту доходить до пользователя.

**Спуфинг с подменой DNS-сервера (DNS Server Spoofing)**

Спуфинг с подменой Domain Name System (DNS), также известный как «отравление кэша», используется для перенаправления трафика пользователя на поддельные IP-адреса. Такой метод позволяет перенаправлять пользователей на вредоносные сайты. В рамках такой атаки мошенник меняет IP-адреса DNS-серверов, указанных на компьютере жертвы, на поддельные IP-адреса, который мошенник хочет использовать для обмана жертвы.

**ARP-спуфинг (ARP Spoofing)**

ARP-спуфинг (Address Resolution Protocol) часто используется для изменения или кражи данных, а также для взлома компьютера жертвы внутри его сессии (подключения). Для этого злоумышленник свяжет себя с IP-адресом жертвы, чтобы иметь возможность получить доступ к тем данным, которые изначально предназначались для владельца этого IP-адреса (т.е. жертвы).

**SMS-спуфинг (Text Message Spoofing)**

Спуфинг с подменой текстовых сообщений (известен также как SMS-спуфинг) – это тип спуфинг-атаки, в рамках которой мошенник отправляет текстовое или SMS-сообщение, используя номер телефона другого человека. Мошенники делают это, скрывая свою личность за буквенно-цифровым идентификатором отправителя, и обычно в свои сообщения включают ссылки для загрузки вредоносных программ или для перехода на фишинговые сайты.

**GPS-спуфинг (GPS Spoofing)**

Атака типа GPS-спуфинг происходит для «обмана» GPS-приемника, когда передаются поддельные сигналы, которые напоминают настоящие. Другими словами, мошенник притворяется, что находится в одном месте, а на самом деле находится в другом. Мошенники могут использовать такой прием, чтобы, например, взломать GPS в автомобиле и отправить вас по ложному адресу или даже вмешаться в GPS-сигналы кораблей, самолетов и т.д.

**Атака посредника или «человека посередине» (Man-in-the-middle Attack, MitM)**

Атаки типа Man-in-the-middle (MitM) происходят в тех случаях, когда мошенник взламывает Wi-Fi сеть или создает дублирующую поддельную Wi-Fi сеть в том же месте для перехвата веб-трафика между двумя сторонами подключения (отправитель и получатель трафика). С помощью такой атаки мошенники могут перенаправлять к себе используемую жертвой конфиденциальную информацию, такую как логины, пароли или номера банковских карт.

**Спуфинг с подменой расширений файлов (Extension Spoofing)**

Чтобы замаскировать вредоносные программы, мошенники часто используют спуфинг с подменой расширений. Например, они могут переименовать файл в «filename.txt.exe», спрятав вредоносную программу внутри расширения. Таким образом, файл, который кажется текстовым документом, на самом деле при его открытии запускает вредоносную программу.

**Формат пакета IPV4 и IPV6, отличие v4 от v6, формат адресов**

IPv4 – это протокол для использования в сетях Link Layer (например, Ethernet). IPv4 использует 32-битные поля источника и адреса назначения, которые ограничивают адресное пространство до 4,3 млрд. адреса.

IPv6 более совершенен и имеет лучшие функции по сравнению с IPv4. Он имеет возможность предоставлять бесконечное количество адресов.

Он заменяет IPv4 для удовлетворения растущего числа сетей по всему миру и помогает решить проблему исчерпания IP-адреса.

IPv6 был разработан Целевой группой Internet Engineering Task Force (IETF).

Одним из основных отличий между IPv4 и IPv6 является их адресное пространство.

Размер адреса в IPv4 составляет 32 бита, тогда как поля адреса IPv6 равны 128 бит.

Из-за их разницы в адресном пространстве – появление IP-адресов в IPv4 и IPv6 также выглядит по-другому.

В IPv4 IP-адреса отображаются как четыре десятичных числа в 1 байт, разделенные точкой (например: 192.168.1.1), а в IPv6 IP-адресах отображаются шестнадцатеричные числа, разделенные двоеточиями (например: fe80 :: d4a8: 6435: d2d8: d9f3b11).

IPv4

Размер пакета: требуется 576 байт, фрагментация необязательна.

Фрагментация пакетов: маршрутизаторы и отправители хосты.

У IPv4 отсутствует безопасность.

IPv4 никогда не был разработан для обеспечения безопасности

– Первоначально разработанный для изолированной военной сети

–Затем адаптирована для общественной образовательной и исследовательской сети.

Заголовок IPv4 имеет 20 байтов. Заголовок IPv4 имеет много полей (13 полей)

У многих интернет-провайдеров есть подключение к IPv4 или есть как IPv4, так и IPv6

Неравномерное географическое распределение (> 50% США)

IPv6

Размер пакета: 1280 байтов без фрагментации.

Фрагментация пакетов: хосты.

IPv6 имеет встроенную надежную защиту

– Шифрование;

– Аутентификация.

Заголовок IPv6 является двойным, он имеет 40 байт. В заголовке IPv6 меньше полей, у него есть 8 полей.

У многих интернет-провайдеров нет подключения к IPv6.

Отсутствие географического ограничения.

**Протокол ARP, RARP**

Протокол ARP (Address Resolution Protocol, RFC-826, std-38, Протокол распознавания адреса) предназначен для преобразования IP-адресов в MAC-адреса, часто называемые также физическими адресами.

В сетях нет однозначного соответствия между физическим адресом сетевого интерфейса (MAC адресом сетевой карты) и его IP-адресом. Поиск по IP-адресу соответствующего Ethernet-адреса производится протоколом ARP, функционирующим на уровне доступа к среде передачи. Протокол поддерживает в оперативной памяти динамическую arp-таблицу в целях кэширования полученной информации.

Порядок функционирования протокола следующий.

С межсетевого уровня поступает IP-дейтаграмма для передачи в физический канал (Ethernet), вместе с дейтаграммой передается, среди прочих параметров, IP-адрес узла назначения. Если в arp-таблице не содержится записи об Ethernet-адресе, соответствующем нужному IP-адресу, модуль arp ставит дейтаграмму в очередь и формирует широковещательный запрос. Запрос получают все узлы, подключенные к данной сети; узел, опознавший свой IP-адрес, отправляет arp-ответ (arp-response) со значением своего адреса Ethernet.

Полученные данные заносятся в таблицу, ждущая дейтаграмма извлекается из очереди и передается на инкапсуляцию в кадр Ethernet для последующей отправки по физическому каналу. Протокол ARP может поддерживать не только Ethernet, но и другие типы физических сред.

Reverse ARP, обратный ARP протокол служит для того, чтобы по имеющемуся MAC адресу узнать IP адрес. Этот протокол используется в бездисковых машинах, загружающихся по сети. Первым делом такая машина должна узнать свой IP адрес, и параметры сети, чтобы она могла обратиться по сети, допустим к TFTP серверу, с которого она будет скачивать загрузочную запись. Единтсвенное, что знает о себе эта машина — её MAC адрес.

Она посылает в сеть широковещательный запрос с поиском RARP сервера и спрашивает у него, какой IP адрес будет ей соответствовать, если у неё вот такой MAC адрес. Это не тоже самое, что DHCP, хотя смысл похожий.

**Связь адреса IPv6 c MAC**

При помощи использования MAC-адреса интерфейса данного узла. Для генерации идентификатора интерфейса существует определённый алгоритм. Для устройства с MAC-адресом, например, f8:ac65:2b:ba:11 это выглядит следующим образом:

* В середину MAC-адреса нужно вставить ff:fe – и привести запись к формату IPv6 – f8ac:65ff:fe2b:ba11.
* Первый октет необходимо перевести из шестнадцатеричной системы в двоичную: f8 -> 11111000.
* Шестой бит (начиная с нулевого) нужно инвертировать: 11111000 – > 11111010.
* Полученное число необходимо снова перевести в шестнадцатеричную систему: 11111010 -> fa.
* Первый октет нужно заменить на полученное значение: faac:65ff:fe2b:ba11.

Таким образом вместо того, чтобы получать адрес по DHCP, как это происходит в IPv4, сетевой адаптер самостоятельно назначает себе адрес IPv6, используя для этого свой же MAC-адрес.

**Контрольная сумма в 4 и 6 версии**

Если нет повреждений, результат суммирования всего IP-заголовка, включая контрольную сумму, должен быть равен нулю. На каждом переходе проверяется контрольная сумма. Пакеты с несоответствием контрольной суммы отбрасываются. Маршрутизатор должен скорректировать контрольную сумму, если он изменяет заголовок IP (например, при уменьшении TTL).

**Расчет контрольной суммы заголовка IPv4**

Возьмите следующий усеченный фрагмент пакета IPv4. Заголовок выделен жирным шрифтом, а контрольная сумма подчеркнута.

4500 0073 0000 4000 4011 b861 c0a8 0001

c0a8 00c7 0035 e97c 005f 279f 1e4b 8180

При добавлении дополнения к единице каждый раз, когда происходит перенос, мы должны прибавлять 1 к сумме. Проверка переноса и коррекция могут выполняться при каждом добавлении или в качестве пост-обработки после всех добавлений. Если корректировкой генерируется еще один перенос, к сумме добавляется еще 1.

Чтобы вычислить контрольную сумму, мы можем сначала вычислить сумму каждого 16-битного значения в заголовке, пропуская только само поле контрольной суммы. Обратите внимание, что эти значения представлены в шестнадцатеричной системе счисления.

4500 + 0073 + 0000 + 4000 + 4011 + c0a8 + 0001 + c0a8 + 00c7 = 2479c

Первая цифра – это счетчик переноса, который добавляется к сумме:

2 + 479c = 479e (если этим сложением генерируется еще один перенос, еще 1 должен быть добавлен к сумме)

Чтобы получить контрольную сумму, мы берем дополнение до единиц этого результата: b861 (как показано подчеркнутым в исходном заголовке IP-пакета).

**Проверка контрольной суммы заголовка IPv4**

При проверке контрольной суммы используется та же процедура, что и выше, за исключением того, что исходная контрольная сумма заголовка не опускается.

4500 + 0073 + 0000 + 4000 + 4011 + b861 + c0a8 + 0001 + c0a8 + 00c7 = 2fffd

Добавьте биты переноса:

fffd + 2 = ffff

Взяв дополнение до единиц (перевернув каждый бит), получится 0000, что означает отсутствие ошибки. обнаружен. Контрольная сумма IP-заголовка не проверяет правильный порядок 16-битных значений в заголовке.

**В отличие от IPv4-протокола, когда IPv6-узлом/отправителем** передаются UDP-дейтаграммы, вычисление проверочных сумм этих дейтаграмм является обязательным. Другими словами, если, когда бы то ни было, передается UDP-пакет (то есть IPv6-пакет, содержащий UDP-дейтаграмму), то тогда IPv6-узел/отправитель должен вычислить проверочную сумму данного пакета, используя для этого последовательность октетов, в которую входят сам пакет и псевдозаголовок. Если результат вычисления оказался равным нулю, то тогда 16-битовое поле «Контрольная сумма» в заголовке UDP-дейтаграммы должно заполняться единицами. IPv6-узлы/получатели должны уничтожать UDP-пакет, в котором поле «Контрольная сумма» заголовка UDP-дейтаграммы содержит только одни нули, и после этого они должны регистрировать возникновение нештатной ситуации (ошибки).

Шестая версия ICMP-протокола (ICMPv6) включает рассмотренный выше псевдозаголовок в последовательность октетов, по которой вычисляется проверочная сумма (в этом заключается одно из отличий от ICMPv4, которая не предусматривает включение псевдозаголовка при расчёте проверочной суммы). Причина этого заключается в том, что ICMPv6-сообщения необходимо защитить от ошибочной доставки и искажения тех полей IPv6-заголовка, на которые оно ссылается, так как (в отличие от IPv4-протокола) IPv6-заголовок не входит в последовательность октетов, по которой вычисляется проверочная сумма сетевого уровня Internet-архитектуры. Если значение в поле «Идентификатор следующего заголовка» псевдозаголовка для ICMP-протокола равно «58», то это указывает на наличие ICMPv6-сообщения.

Для примера рассчитаем контрольную сумму нескольких 16-битных слов:

0х398а, 0xf802, 0x14b2, 0xc281

Для этого можно сначала сложить попарно числа, рассматривая их как 16-разрядные беззнаковые числа с последующим приведением к дополнительному коду путём прибавления единицы к результату, если при сложении произошёл перенос в старший (17-й) разряд. Или, что равноценно, можно считать, что перенос прибавляется к младшему разряду числа.

0x398a + 0xf802 = 0x1318c → 0x318d (перенос в старший разряд)

0x318d + 0x14b2 = 0x0463f → 0x463f (число положительное)

0x463f + 0xc281 = 0x108c0 → 0x08c1

В конце выполняется инверсия всех битов получившегося числа

0x08c1 = 0000 1000 1100 0001 → 1111 0111 0011 1110 = 0xf73e или, иначе — 0xffff − 0x08c1 = 0xf73e.

Это и есть искомая контрольная сумма.

**Формат адреса 6**

В протоколе IPv6 размер адреса составляет 128 бит. Предпочтительным является следующее представление адреса IPv6: x:x:x:x:x:x:x:x, где каждая буква x — это шестнадцатеричные значения шести 16-битных элементов адреса. Диапазон адресов IPv6 составляет от 0000:0000:0000:0000:0000:0000:0000:0000 до ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff.

Помимо обычного формата, адреса IPv6 могут быть представлены в двух других форматах:

- С пропуском начальных нулей. Адрес IPv6 записывается с пропуском начальных нулей. Например, адрес IPv6 1050:0000:0000:0000:0005:0600:300c:326b можно записать как 1050:0:0:0:5:600:300c:326b.

- Двойное двоеточие. В адресе IPv6 на месте нескольких нулей ставится двойное двоеточие (::). Например, адрес IPv6 ff06:0:0:0:0:0:0:c3 можно записать как ff06::c3. В одном IP-адресе двойное двоеточие может использоваться только один раз.

В альтернативном формате адреса IPv6 совмещаются формат с двоеточиями и формат с точками, поэтому адреса IPv4 можно вставлять в адреса IPv6. В первых 96 битах указываются шестнадцатеричные значения, а в последних 32 битах указываются десятичные значения, задающие адрес IPv4. Такой формат обеспечивает совместимость между узлами IPv6 и IPv4.

Адрес IPv6, преобразованный для IPv4, использует альтернативный формат. Такие адреса представляют узлы IPv4 в сети IPv6. С их помощью приложение IPv6 может напрямую взаимодействовать с приложением IPv4. Например, 0:0:0:0:0:ffff:192.1.56.10 и ::ffff:192.1.56.10/96 (сокращенный формат).

**Формат сегмента TCP, SCTP и UDP.**

**TCP**

TCP-сегмент состоит из поля данных и нескольких полей заголовка. Поле данных содержит фрагмент данных, передаваемых между процессами. Размер поля данных ограничивается величиной MSS (максимальный размер сегмента). Когда протокол осуществляет передачу большого файла, он, как правило, разбивает данные на фрагменты размером MSS (кроме последнего фрагмента, который обычно имеет меньший размер). Интерактивные приложения, напротив, часто обмениваются данными, объем которых значительно меньше MSS. Поскольку обычно длина заголовка ТСР-сегмента составляет 20 байт (что на 12 байт больше, чем в UDP), полный размер сегмента в этом случае равен 21 байт.

Как и в протоколе UDP, заголовок включает номера портов отправителя и получателя, предназначенные для процедур мультиплексирования и демультиплексирования данных, а также поле контрольной суммы. Кроме того, в состав TCP-сегмента входят еще некоторые поля.

- 32-разрядные поля порядкового номера и номера подтверждения;

- 4-разрядное поле длины заголовка определяет длину TCP-заголовка в 32-разрядных словах;

- Поле флагов состоит из 6 бит;

- 16-разрядное окно приема используется для управления потоком данных. Оно содержит количество байтов, которое способна принять принимающая сторона;

- Указатель важности — 16-битовое значение положительного смещения от порядкового номера в данном сегменте. Это поле указывает порядковый номер октета, которым заканчиваются важные (urgent) данные;

- Необязательное поле параметров используется в случаях, когда передающая и принимающая стороны «договариваются» о максимальном размере сегмента, либо для масштабирования окна в высокоскоростных сетях.

**SCTP**

Для всех пакетов SCTP требуется раздел общего заголовка.

- Исходный порт. Это поле определяет порт отправки;

- Порт назначения. Это поле определяет порт приема, который хосты используют для маршрутизации пакета к соответствующей конечной точке /приложению;

- Верификационный тег. 32-битное случайное значение, созданное во время инициализации, чтобы отличить устаревшие пакеты от предыдущего соединения;

- Контрольная сумма.

Каждый пакет SCTP состоит, помимо общего заголовка, из кусков. Каждый блок имеет общий формат, но содержимое может отличаться.

- Тип блока. 8-битное значение, заданное IETF для идентификации содержимого поля chunk value;

- Флаги блоков. Восемь флаг-битов, определение которых зависит от типа блока. Значение по умолчанию равно нулю;

Длина блока. 16-битное значение без знака, определяющее общую длину блока в байтах (исключая любое заполнение), которое включает в себя тип блока, флаги, длину и поля значений;

Фрагментные данные. Поле данных общего назначения, определение которого зависит от типа блока.

**UDP**

Состоит из 4-х полей (все по 16 бит):

- Порт отправителя;

- Порт получателя;

- Длина UDP, в которой содержится длина всей дейтаграммы UDP;

- Контрольная сумма UDP, которая используется для корректной доставки дейтаграммы.

Длина:

- Минимум 8 байт (только заголовок);

- Максимум 65 515 байт (максимальная длина данных IP-пакета).

**Команды ICMP.**

ICMP (англ. Internet Control Message Protocol — протокол межсетевых управляющих сообщений) — сетевой протокол, входящий в стек протоколов TCP/IP. В основном ICMP используется для передачи сообщений об ошибках и других исключительных ситуациях, возникших при передаче данных, например, запрашиваемая услуга недоступна или хост, или маршрутизатор не отвечают. Также на ICMP возлагаются некоторые сервисные функции.

**ICMP флуд атака.**

Очень примитивный метод забивания полосы пропускания и создания нагрузок на сетевой стек через монотонную посылку запросов ICMP ECHO (пинг). Легко обнаруживается с помощью анализа потоков трафика в обе стороны: во время атаки типа ICMP-флуд они практически идентичны. Почти безболезненный способ абсолютной защиты основан на отключении ответов на запросы ICMP ECHO или с помощью брандмауэра.

**TCP флуд атака.**

Один из распространенных способов не только забить канал связи, но и ввестисетевой стек операционной системы в такое состояние, когда он уже не сможетпринимать новые запросы на подключение. Основан на попытке инициализациибольшого числа одновременных TCP-соединений через посылку SYN-пакета снесуществующим обратным адресом. После нескольких попыток отослать ответныйACK-пакет на недоступный адрес большинство операционок ставят неустановленноесоединение в очередь. И только после n-ой попытки закрывают соединение. Так какпоток ACK-пакетов очень велик, вскоре очередь оказывается заполненной, и ядродает отказ на попытки открыть новое соединение. Наиболее умные DoS-боты еще ианализируют систему перед началом атаки, чтобы слать запросы только на открытыежизненно важные порты. Идентифицировать такую атаку просто: достаточнопопробовать подключиться к одному из сервисов. Оборонительные мероприятия обычновключают в себя:

* Увеличение очереди "полуоткрытых" TCP-соединений;
* Уменьшение времени удержания "полуоткрытых" соединений;
* Включение механизма TCP syncookies;
* Ограничение максимального числа "полуоткрытых" соединений с одного IP к конкретному порту.

**Перенаправление шлюза.**

Когда маршрутизатор должен послать пакет, предназначенный для другой сети, он должен знать IP-адрес следующего соответствующего маршрутизатора. То же самое верно, если передатчик — хост. И маршрутизаторы, и хосты должны иметь таблицу маршрутизации, чтобы найти адрес маршрутизатора или следующего маршрутизатора. Маршрутизаторы принимают участие в процессе модификации таблиц маршрутизации, и процесс модификации производится периодически или постоянно. Маршрутизация является динамической.

Однако хосты не принимают участие в процессе модификации таблиц маршрутизации, потому что хостов в Интернете намного больше, чем маршрутизаторов. Обновление таблиц маршрутизации хостов динамически создает недопустимый трафик. Хосты обычно используют статическую маршрутизацию. Когда хост возникает, его таблица маршрутизации имеет ограниченное число входов. Он обычно знает только адрес IP одного маршрутизатора, заданного по умолчанию. По этой причине хост может послать дейтаграмму, которая предназначена для другой сети, неправильному маршрутизатору. В этом случае маршрутизатор, который получает дейтаграмму, отправит дейтаграмму правильному маршрутизатору. Однако, чтобы обновить таблицу маршрутизации хоста, он посылает сообщение переназначения хосту. Хост A хочет послать дейтаграмму хосту Б. Маршрутизатор R2 — очевидно самый эффективный выбор направления, но хост не выбрал маршрутизатор R2. Дейтаграмма вместо этого идет в R1. R1, после сравнения со своей таблицей, находит, что пакет должен был уйти в R2. Он посылает пакет R2 и, в то же самое время, посылает сообщение переназначения хосту А. Таблица маршрутизации хоста А может теперь быть модифицирована.

**Эхо запрос, эхо ответ.**

Эхо-запрос и сообщения эхо-ответа разработаны для целей диагностики. Сетевые менеджеры и пользователи применяют эту пару сообщений, чтобы идентифицировать сетевые проблемы.

Комбинация сообщений запроса эха и ответа эха определяет, могут ли две системы (хосты или маршрутизаторы) связаться друг с другом.

Хост или маршрутизатор могут послать сообщение запроса эха другому хосту или маршрутизатору. Хост или маршрутизатор, который получает сообщение запроса эха, создает сообщение ответа эха и возвращает его первоначальному передатчику.

Запрос эха и сообщения ответа эха могут использоваться, чтобы определить, есть ли связь на уровне IP. Поскольку ICMP-сообщения инкапсулированы в дейтаграммах IP, получение сообщения ответа эха устройством, которое послало запрос эха, – доказательство, что протоколы IP в передатчике и приемнике связаны друг с другом с использованием дейтаграмм IP. Также это доказательство, что промежуточные маршрутизаторы получают, обрабатывают и устанавливают соединения дейтаграмм IP.

Запрос эха и сообщения ответа эха могут также использоваться хостом, чтобы видеть, достижим ли другой хост. На пользовательском уровне это делается вызовом программы тестирования каналов сетей пакетной коммутации (утилита PING – Packet Internet Groper).

**Подавление трафика.**

Выбираются точки (участки сети), трафик в которых анализируется на предмет выявления аномалий. В зависимости от того, что мы защищаем, такими точками могут являться пиринг-соединения оператора связи с вышестоящими операторами, точки подключения нижестоящих операторов или абонентов, каналы подключения центров обработки данных к сети.

Специальные детекторы анализируют трафик в этих точках, строят (изучают) профиль трафика в его нормальном состоянии, при появлении DDoS-атаки или аномалии – обнаруживают её, изучают и динамически формируют её характеристики. Далее, информация анализируется оператором системы, и в полуавтоматическом или автоматическом режиме запускается процесс подавления атаки. Подавление заключается в том, что трафик, предназначенный "жертве", динамически перенаправляется через устройство фильтрации, на котором к этому трафику применяются фильтры, сформированные детектором и отражающие индивидуальный характер этой атаки. Очищенный трафик вводится в сеть и отправляется получателю (потому и возникло название Clean Pipes – абонент получает "чистый канал", не содержащий атаку).

Таким образом, весь цикл защиты от DDoS-атак включает следующие основные стадии:

- Обучение контрольным характеристикам трафика;

- Обнаружение атак и аномалий;

- Перенаправление трафика с целью его пропуска через устройство очистки;

- Фильтрация трафика с целью подавления атак;

- Ввод трафика обратно в сеть и отправка адресату.