Факультет компьютерных технологий и прикладной математики

Кафедра вычислительных технологий

02.03.02

Информационная безопасность

Лабораторная работа № 8

Тема: Distributed Denial of Service (DDoS). Основные особенности их организации и защиты от них. Iptables, WEB APLICATION FIREWALL.

Цель работы

Получить теоретические и практические навыки о принципах осуществления DDoS атак и защиты от них. Изучение межсетевых экранов. Приобретение навыков работы с Iptables и WAF.

Указания к работе

Вначале студенты изучают теоретическую часть. Далее каждый студент должен выполнить задания, а также ответить на вопросы к лабораторной работе. За проделанную работу студент может получить оценку от «неудовлетворительно» до «отлично». Для получения оценки «удовлетворительно» студент должен выполнить ВСЕ задания к лабораторной работе. Оценка «хорошо» ставится, если студент ответил на ВСЕ вопросы к лабораторной работе. Оценку «отлично» студент получает, если подготовлен отчёт по лабораторной работе.

ОЦЕНКУ ЗА ПРОДЕЛАННУЮ РАБОТУ МОЖНО ПОВЫСИТЬ ДО СДАЧИ СЛЕДУЮЩЕЙ ЛАБОРАТОРНОЙ РАБОТЫ.

Теоретическая часть

DoS/DDoS

DoS (от англ. Denial of Service – отказ в обслуживании) – хакерская атака на вычислительную систему с целью довести её до отказа, то есть создание таких условий, при которых легальные пользователи системы не могут получить доступ к предоставляемым системным ресурсам (серверам), либо этот доступ затруднён. Отказ «вражеской» системы может быть и

шагом к овладению системой (если в нештатной ситуации ПО выдаёт какую-либо критическую информацию – например, версию, часть программного кода и т.д.). Но чаще это мера экономического давления: простой службы, приносящей доход, счета от провайдера и меры по уходу от атаки ощутимо бьют «цель» по карману. В настоящее время DoS и DDoS-атаки наиболее популярны, так как позволяют довести до отказа практически любую систему, не оставляя юридически значимых улик.

Если атака выполняется одновременно с большого числа компьютеров, говорят о DDoS-атаке (от англ. Distributed Denial of Service, распределённая атака типа «отказ в обслуживании»).

Для обнаружения распределённых сетевых атак типа «отказ в обслуживании» и защите от них необходимо классифицировать их и знать принципы их работы.

В данной работе критерием для классификации рассмотрим объект, на который нацелена атака. В таком случае, получается четыре основных класса атак, соответствующих уровням модели ISO OSI.

Канальный уровень (L2) – атаки направлены на исчерпание ёмкости сетевого канала. В следствие этого лишается доступ сервера к внешней сети. Для реализации используются объёмные потоки трафика. На данным момент измеряются в Гб/с. Во время этой атаки обрабатывать трафик необходимо на стороне провайдера, дата-центра. С помощью BGP Flow Spec фильтруется часть атак по сигнатурам пакета. Amplification атаки отсекаются по порту.

Сетевой уровень (L3) – атаки направлены на нарушение работы элементов сетевой инфраструктуры. Необходим ручной анализ сетевой инфраструктуры. Если своей автономной системы нет, то борьба с атаками данного класса ведется провайдером или дата-центром. Желательно сотрудничество с ними.

Транспортный уровень (L4) – атаки направлены на эксплуатацию слабых мест TCP-стека. В TCP протоколе используется таблица открытых соединений. Атаки именно на неё составляют этот класс. Необходим

постоянный анализ поведения TCP-стека, TCP-клиентов, TCP-пакетов. Эвристический анализ.

Прикладной уровень (L7) – атаки направленны на нарушение работы Web-приложения. Атаки этого класса характеризуются большим разнообразием. Исчерпывают ресурсы сервера. Необходим поведенческий и корреляционный анализ, мониторинг ресурсов сервера. Необходима оптимальная настройка сервера под решаемые им задачи. Полностью автоматизировать борьбу с данным классом атак почти невозможно.

Медленная атака. SlowLoris.

Уровень атаки: транспортный уровень (L4).

Описание и принцип работы:

Атака Slowloris устанавливает много открытых соединений на сервере с помощью постоянной отправки незавершенных HTTP-запросов. В определенные моменты времени Slowloris отправляет следующие HTTP заголовки для каждого запроса, но не завершает соединение. Если запросы посылаются с оптимальной периодичностью, сервер начинает ожидать завершения открытых соединений. В данном случае ресурсы сервера остаются относительно свободными, но сам сервер перестаёт обслуживать новые подключения.

Дело в том, что веб-серверы Apache 1.x, Apache 2.x, dhttpd, GoAhead WebServer и Squid поддерживают ограниченное число одновременно открытых подключений. Но Slowloris не представляет угрозы для серверов IIS, lighttpd, NGINX. Они имеют эффективные механизмы распределения нагрузки и используют worker pool – «пулы рабочих потоков», которые позволяют удерживать любое количество открытых соединений при наличии свободных ресурсов.

Медленная атака. Slow HTTP POST/GET.

Уровень атаки: транспортный уровень (L4).

Описание и принцип работы:

Атака основана на уязвимости в протоколе HTTP. Slow HTTP POST атака отправляет POST заголовок с полем «Content-Length». Веб-сервер понимает, какой объём данных он должен получить. После этого с очень низкой скоростью передаётся тело POST сообщения. Это позволяет задействовать ресурсы сервера длительное время, и в последствии помешать обработке других запросов. Атака опасна для веб-серверов Microsoft IIS и Apache и NGINX со стандартными настройками в рамках протоколов HTTP, HTTPS, подключений SSL, VPN. Также атака может быть настроена для работы с SMTP и DNS-серверами.

Данный тип атаки на отказ в обслуживании, можно организовать через proxy. Трафик данной атаки схож с легитимным трафиком.

Медленная атака. Sockstress.

Уровень атаки: транспортный уровень (L4).

Описание и принцип работы:

Атака заключается в следующем. Если на веб-сервере есть объект, размер которого больше send buffer, выделенного ядром для соединения. То можно заставить ядро не принимать данные, а сервер будет пробовать отправить кусок данных, занимая стек соединений, ресурсы процессора и память. При большом количество подобных соединений TCP-стек заполниться и не будет открывать новые соединения.

Примеры:

Отправить в пакете размер окна равный нулю, то есть нет места для получения данных. Скрипт Sockstress отправляет такие пакеты и считает время, когда их отправлять, чтобы не загрузить persist timer.

Создать сокет с малым объёмом receive buffer на клиенте. Отправлять HTTP-запрос на объект сайта больший по размеру, чем буфер. И

периодически считываем пару байт из receive buffer. Сервер будет пытаться отправлять данные и занимать ресурсы.

Атака произвольными пакетами. HTTP-Flood.

Уровень атаки: канальный уровень (L2), прикладной уровень (L7).

Описание и принцип работы:

В основе этой атаки лежит механизм отправления максимального числа HTTP запросов на 80-й порт веб-сервера. Целью атаки может быть корень сервера или ресурсоёмкий элемент. В результате данной атаки возможно прекращение предоставления услуг по HTTP, и затруднен доступ легитимных пользователей к сайту. Распознать атаку можно с помощью выявления быстрого роста количества запросов к некоторым элементам веб-сервера и логов сервера.

Атака произвольными пакетами. UDP-Flood.

Уровень атаки: канальный уровень (L2), прикладной уровень (L7).

Описание и принцип работы:

UDP flood атака основана на отправке большого количества UDP-пакетов на некоторые порты сервера. Он должен определить приложение для каждого полученного пакета, удостоверится в его неактивности и отправить в ответ ICPM-сообщение «недоступен». В итоге вырастут затрачиваемые ресурсы атакуемого сервера и полоса пропускания заполнится UDP-пакетами. В UDP протоколе нет механизма проверки отправителя пакетов, тем самым злоумышленник может подменить IP-адреса и обеспечить анонимность.

Атака произвольными пакетами. SYN-Flood.

Уровень атаки: канальный уровень (L2), транспортный уровень (L4).

Описание и принцип работы:

SYN Flood атака использует механизм рукопожатия в протоколе TCP. Работает следующим образом. Посылается пакет с флагом SYN на атакуемый сервер. Он вынужден отправить в ответ пакеты с флагами SYN+ACK.

Злоумышленник игнорирует SYN+ACK пакеты сервера и не высылает в ответ пакет ACK. Либо подделывает IP-адрес SYN пакета, чтобы ответный SYN+ACK отправляется на некорректный адрес. Цель данной атаки – заполнение TCP стека множеством полуоткрытых соединений, в следствие чего сервер перестают устанавливать соединения с новыми клиентами. Запросы на соединение полученные сервером хранятся в стеке с определенным размером, который зависит от операционной системы. Они находятся в стеке, пока сервер не получит информацию об установленном соединении от клиента.

Атака произвольными пакетами. ICMP-Flood.

Уровень атаки: канальный уровень (L2).

Описание и принцип работы:

Данный тип флуда направлен на сетевое оборудование. Принцип данной атаки заключается в том, что ICMP-пакет при небольшом размере самого запроса требует от устройств значительно большего объёма работы. То есть, при отправлении сравнительно небольшого объёма ICMP запросов возникает перегрузка сетевого оборудования и значительная часть легитимных запросов теряется. Злоумышленник, меняет IP-адрес источника, отправляет ICMP Echo Request пакет к определённым компьютерам, входящим в бот-нет. Они отвечают ICMP Echo Reply пакетом, посылая его на изменённый IP-адрес. Для увеличения мощности атаки используют локальные сети (LAN) с включенной опцией направленной широковещательной рассылки (directed broadcast).

Атака с помощью SSL

Уровень атаки: прикладной уровень (L7).

Описание и принцип работы:

Secure Sockets Layer (SSL) – это протокол безопасности для защиты целостности сети связи и передачи данных. SSL может зашифровать подключение к сети на транспортном уровне. Процессы шифрования протокола SSL, расшифровки и обмена ключами потребляют огромное

количество системных ресурсов. Существует два типа атака основанных на протоколе SSL.

Первый тип эксплуатирует механизм рукопожатия, который исчерпывает ресурсы атакуемого сервера. Злоумышленник посылает некорректные SSL данные на сервер, на обработку которых затрачивается большое количество вычислительной мощности.

Второй тип использует функцию повторного подтверждения соединения – SSL Renegotiation.

Установка безопасного соединения и повторное подтверждение SSL затрачивают в разы больше вычислительной мощности на сервере, чем на стороне клиента. Благодаря этому возможно осуществление атаки и истощение ресурсов атакуемого сервера. Как правило, HTTPS расшифровывается глубоко внутри организационной сети, где серверы и модули более уязвимы к вредоносному трафику. Также злоумышленники используют этот протокол для обхода механизмов безопасности. Таким образом, возможно туннелирование других атак.

Атака почтового сервера. SMTP-Flood.

Уровень атаки: прикладной уровень (L7).

Описание и принцип работы:

В атаках этого типа злоумышленник пытается установить соединение с почтовым сервером и отправляет произвольные письма на сгенерированные случайным образом адреса, либо бездействует до истечения тайм-аута, удерживая соединение открытым. Каждое SMTP соединение утилизирует часть ресурсов сервера, тем самым атакующий пытается вызвать отказ в обслуживании.

Технология заключается в том, что когда мы пишем e-mail получателя несуществующий, то SMTP server оповещает нас с помощью e-mail (отправителя), что письмо не дойдёт. Мы посылаем на SMTP server очень много запросов (пишем письма) и указываем все e-mail не валидные, а отправителя(объект атаки). И к нему будут приходить уведомления, что

письмо не дошло. С нескольких SMTP server будет отправляться трафик на объект атаки.

Некорректные пакеты/фрагменты. UDP fragment flood.

Уровень атаки: прикладной уровень (L7).

Описание и принцип работы:

Данный тип атаки основан на отправки UDP датаграмм, которые случайным образом ссылаются на датаграммы отсутствующие в потоке. Это приводит к увеличению потребления памяти на атакуемом сервере. При атаке UDP Fragment Flood, злоумышленники посылают UDP пакеты большого размера, для истощения пропускной способности канала.

Некорректные IP-фрагменты

Уровень атаки: прикладной уровень (L7).

Описание и принцип работы:

Данный тип атак эксплуатирует уязвимости в поддержке фрагментации пакетов протокола IP. Одна из атак этого типа – это пересечение IP-фрагментов. Она реализуется с помощью уязвимости операционной системы, которая заключается в сборке фрагментированных IP- пакетов. В процессе сборки образуется цикл по принятым фрагментам. Затем из них копируется информативная часть и передаётся на IP уровень. Разработчики предусмотрели проверку на чрезмерный объем копируемой информации, но не ввели проверку на копирование фрагмента отрицательной длины. Копирование блока информации отрицательной длины равносильно копированию очень большого блока информации. Это приводит к затиранию большого участка памяти и к нарушению работы ЭВМ (электронная вычислительная машина). Существует две программы с небольшими отличиями в константах механизма, который осуществляет пересечение IP-фрагментов: newtear и teardrop. Они отправляют пакеты с заданного IP-адреса на любой порт, независимо, открыт он или закрыт. Еще один вариант данной атаки – bonk. После сборки фрагментов в пакете остаются пустые места. Это приводит к сбою ядра операционной системы и нарушению

работы электронных вычислительных средств. Данные уязвимости присутствуют в старых версиях ОС(операционных систем) Windows и Linux. На сегодняшний день большинство сетевых ОС защищены от сбоев в работе, вызванных данной атакой.

Неверные значения в заголовках пакетов

Уровень атаки: прикладной уровень (L7).

Описание и принцип работы:

Данные атаки нацелены на определённые приложения и операционные системы, которые неправильно обрабатывают некорректные значения в заголовках пакетов. Пример такой атаки – Land attack. В ходе неё злоумышленник устанавливает в пакете одинаковый IP-адрес для источника и приёмника. Это приводит к зацикливанию установки соединения сервера с самим собой.

Amplification атаки

Уровень атаки: канальный уровень (L2).

Описание и принцип работы:

В основе данных атак лежит отсутствие проверки отправителя в UDP протоколе. Ответ посылается адресату, указанному в заголовках пакета. Злоумышленник может подменить свой IP-адрес на IP-адрес атакуемого сервера в заголовках отправляемых пакетов. Также суть атаки заключается в многократном превышении объёма ответа по сравнению с запросом. Таким образом, злоумышленник может анонимно организовывать атаки с огромным объёмом трафика. Службы, работающие по UDP протоколу: DNS, NTP, SNMP, rsyslog и многие другие, могут использоваться для реализации атаки. Дело в том, что сетевые устройства с этими службами встречаются в сети повсеместно. Службы включены по умолчанию и часто некорректно настроены.

В таблице 1 представлены типы amplification атак проведённых в ходе исследования. В ней отображается по какому протоколу осуществляются

атаки, коэффициент их усиления и уязвимая команда, используемая для реализации атаки.

Таблица 1 – Amplification атаки

Протокол	**Коэффициент усиления**	**Уязвимая команда**
DNS	x28-x92	DNS server request
NTP	x994	Monlist request
SNMPv2	x29	Get Bulk request
CharGEN	x350	Character generation request
BitTorrent	x4	File search
RIPv1	x131	Malformed request
SSDP	x31	SEARCH request
NetBIOS	x4	Name resolution
Quake Network Protocol	x64	Server info exchange
Steam Protocol	x5.5	Server info exchange

DNS, NTP, SNMPv2 – протоколы для получения информации о доменах, синхронизации времени и сетевого управления. Часто встречаются в сети. NetBIOS и SSDP – протоколы в Windows. CharGEN – старый тестовый сервис, но его до сих пор можно встретить на различных системах. BitTorrent – протокол для обмена файлами. Quake Network Protocol и Steam Protocol – протоколы компьютерных игр.

NTP amplification атака

Уровень атаки: канальный уровень (L2).

Описание и принцип работы:

Злоумышленник отправляет запрос monlist с IP-адресом атакуемого сервера к NTP-серверу. Ответ monlist включает в себя список 600 последних клиентов ntpd. Сущность амплификации заключается в том, что нарушитель отправляет небольшой запрос к уязвимому серверу и с него на атакуемый

сервер отправляется большой поток UDP трафика. Уязвимый NTP-сервер является невольным промежуточным звеном атаки. Ntpd до версии 4.2.7p26 подвержены атаке.

DNS Amplification атака

Уровень атаки: канальный уровень (L2).

Описание и принцип работы:

Атака основана на том, что нарушитель отправляет запрос уязвимому DNS-серверу с IP-адресом атакуемого сервера. DNS-сервер отправляет ответ, размер которого во много раз превышает запрос, жертве. Таким образом, исчерпывается канальная ёмкость атакуемого сервера.

Можно выделить ключевые моменты атаки:

Эффект отражения: подмена IP-адреса позволяет перенаправить ответы от всех DNS-серверов на атакуемый сервер.

Коэффициент усиления атаки: (amplification factor): он может принимать значения от 28 до 92. То есть на 1 байт запроса – совокупность DNS-серверов отправит 28-92 байт ответа. Это обеспечивает кратное увеличение объёма трафика.

Проблема «open resolver»: это неправильно настроенный или старой версии DNS-сервер. Он разрешает получать запросы из сторонних сетей, выполняя рекурсивные запросы для них, и отправлять ответы без необходимых предварительных проверок.

HTTP flood с помощью сервисов

Уровень атаки: прикладной уровень (L7).

Описание и принцип работы:

WordPress сайт с включенным Pingback, можно использовать для проведения HTTP flood атаки на другие сайты. Они отправляют множество запросов к атакуемому сайту со случайными параметрами («?a=a» и др.), с помощью которых обходится кэширование страницы. Эта операция быстро расходует ресурсы атакуемого сервера и нарушает его работу. Злоумышленник может использовать большое количество обычных

WordPress сайтов для DDoS атаки и не бояться быть обнаруженным с помощью Pingback запросу к файлу XML-RPC. Google использует FeedFetcher для кэширования любого контента в Google Spreadsheet, вставленного через формулу =image(«link»). Если в клетку таблицы вставить формулу =image(«http://target/file.pdf»), то Google отправит FeedFetcher скачать этот PDF файл и закэшировать для дальнейшего отображения в таблице. Но если добавлять случайный параметр к URL картинке (от «?r=1» до «?r=1000»), FeedFetcher будет скачивать её каждый раз заново. Это приведёт к исчерпанию лимита трафика атакуемого сервера. Злоумышленник может запустить массированную HTTP GET flood атаку на веб-сервер, используя браузер с одной открытой вкладкой.

Практическая часть

Для эффективной защиты от распределённых сетевых атак типа «отказ в обслуживании» требуется своевременное обнаружение начала атаки. Чтобы обнаружить DDoS-атаку необходимо накапливать статистические данные о трафике сервера в сети, работающем в штатном режиме. При знании среднестатистических значений характеристик сервера, можно отследить появление аномалии трафика в сети.

Для выполнения данной лабораторной работы необходимо создать две виртуальные машины или объединить в локальную сеть два компьютера. Одна машина будет жертвой, другая злоумышленником.

На машине жертве необходимо установить:

1. Web-сервер (Apache, NGINX);

2. Средства мониторинга и анализа ресурсов сервера и сетевого трафика (iptraf, ksysguard или другие).

Дополнительно ознакомьтесь со следующими ресурсами 1, 2, 3, 4.

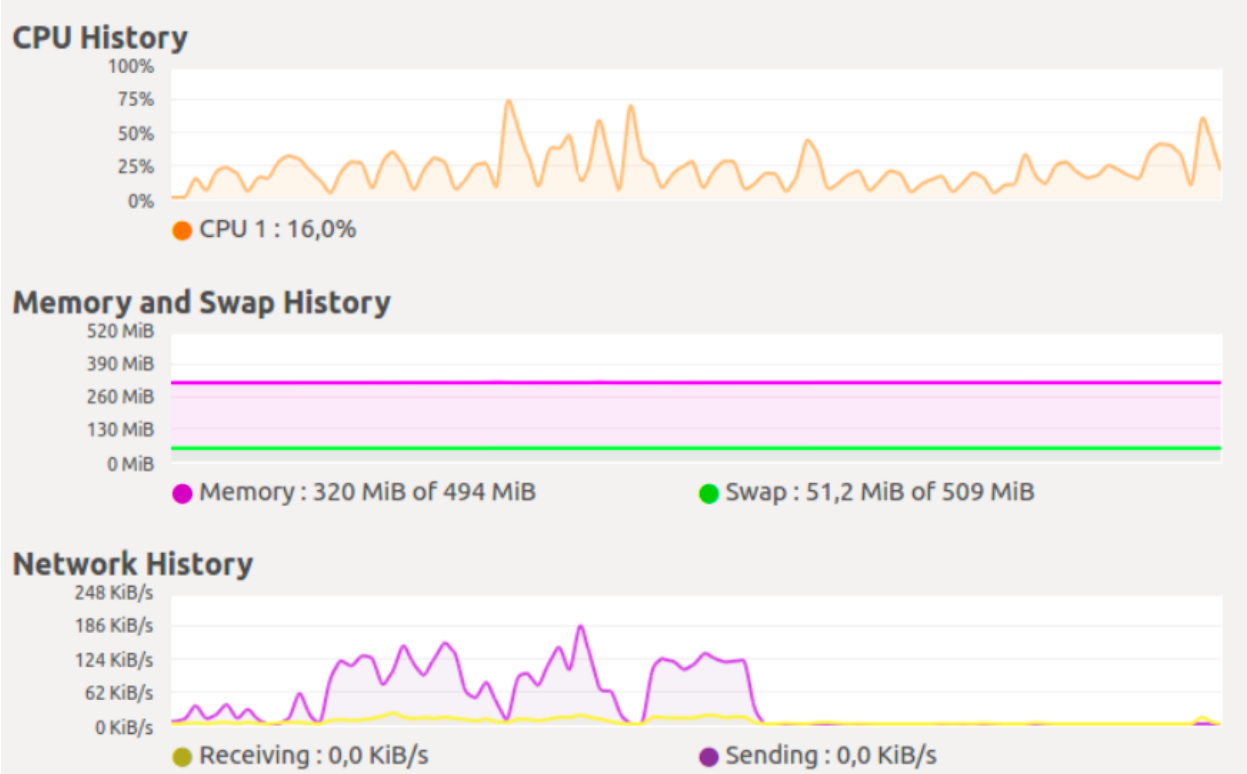
На машине злоумышленника необходимо установить:

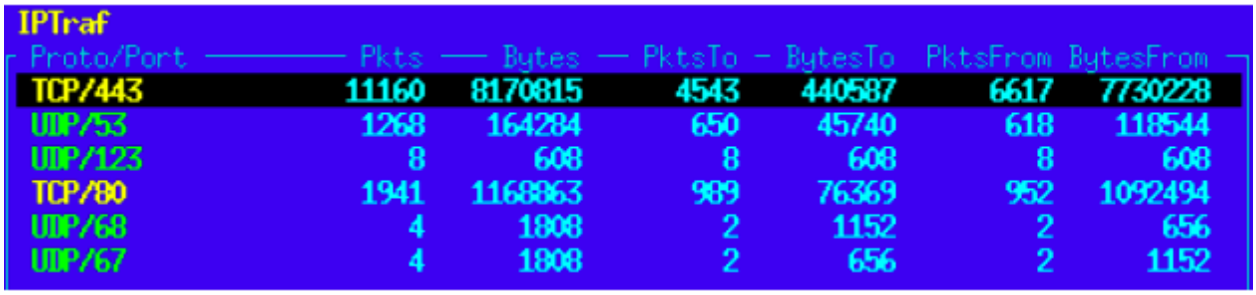
1. Программное обеспечение и скрипты для проведения DDoS атак;

2. Уязвимые сервисы для усиления DDoS атак (dns, ntp, chargen).

Чтобы сделать сервис уязвимым, ознакомьтесь со следующим ресурсом 1.

На машине-жертве фиксируются системные характеристики, такие как: загрузки процессора, объем оперативной памяти и характеристики сетевого трафика, такие как: количество пакетов, открытые соединения.





Скриншоты мониторинга и трафика

Изменение указанных характеристик не обязательно обозначает атаку на сервер, но показывает отклонение от среднестатистических показателей работы в штатном режиме. Для большинства DDoS-атак можно выделить аномалии и признаки, соответствующие только им.

В ходе данной лабораторной работы необходимо провести каждый из представленных типов DDoS атак, отследить аномалии в использовании ресурсов сервера и сетевом трафике, проверить признаки

DDoS атаки и, если они подтвердились, защитить сервер от атаки с помощью настройки сервера, добавление правил для межсетевого экрана, фильтрации пакетов по сигнатурам.

Пошаговый алгоритм методики:

1. Обнаружение аномалии в наблюдаемых системных характеристиках и характеристиках сетевого трафика.

2. Проверить присутствие признаков атаки в трафике.

3. Внести изменения в настройки сервера под конкретную атаку и внести правила для фильтрации.

Для каждого типа атак в сети интернет необходимо найти по несколько программ и скриптов, которые реализуют данный вид атаки.

1. Для атаки HTTP flood: goldeneye, ddosim, DAVOSET, HULK, LOIC, BBHH.

2. Для медленных атак: r-u-dead-yet, slowhttptest, pyloris, sockstress, torshammer. Для UDP flood: LOIC, fudp, hping3.

3. Для SYN flood и атак на TCP-стек: sprut, sitekiller, mummy, hping3.

4. Для ICMP flood: hping3.

5. Для land атаки: fudp, hping3.

6. Для атак с использованием SSL: thc-ssl-dos.

7. Для amplification атак: saddam, chargen_amp.

Скачать набор программного обеспечения можно по этой ссылке.

Ознакомьтесь со справочной документацией используемых Вами программ и скриптов.

Дополнительная информация

Медленные атаки

Аномалия: Кратковременный скачок нагрузки процессора, повышение исходящего трафика и используемой оперативной памяти.

Признак: В TCP-стеке большое количество соединений со статусом ESTABLISHED.

Защита: Если установлен apache, то поставить перед ним кэширующий сервер NGINX. Установить и настроить балансировщик нагрузки. На apache можно установить mod_security – firewall с готовыми правилами от OWASP и mod_reqtimeout – установка таймаутов и минимальной скорости передачи данных для получения запросов.

Добавить правила межсетевого экрана:

1. Ограничение количества соединений с одного IP-адреса.

```
# iptables -A INPUT -p tcp --syn --dport 80 -m connlimit --connlimit-above 30 -j DROP
```

2. Заблокировать IP после 10 подключений к порту 80 в течение 30 секунд.

```
# iptables -I INPUT -p tcp --dport 80 -m state --state NEW -m recent --set
# iptables -I INPUT -p tcp --dport 80 -m state --state NEW -m recent --update --seconds 30 --hitcount 10 -j DROP
```

Также NGINX имеет функцию кэширования клиентского запроса перед отправкой на бэкэнд – client_body_buffer_size. Бэкэнд получит запрос, только когда он полностью загрузится.

Атака произвольными пакетами.

Аномалия: Резкое увеличение нагрузки процессора, входящего и исходящего-HTTP трафика.

Признак: На сервер приходит много однотипных пакетов, резко выросло количество обращений к ресурсоёмкому элементу сайта.

Защита: Борьба с HTTP flood ведётся с помощью усовершенствования работы веб-сервера и баз данных. Также меры включают в себя использование обработки подключений методом epoll, увеличение количества соединений, отключение тайм-аута на закрытие подключений keep-alive.

```
worker_processes 2; worker_rlimit_nofile 8000; events {
worker_connections 4000;
use epoll; }
```

Установить на NGINX модуль ngx_http_limit_req_module и блокировать IP-адреса, которые стали получать ответ «Service unavailable».

```
http {
limit_req_zone $binary_remote_addr zone=zne:15m rate=3r/s; server {
location / {
limit_req zone=zne burst=5;
}
```

Атака произвольными пакетами. UDP-Flood

Аномалия: Резкое увеличение нагрузки процессора, входящего UDP-трафика.

Признак: Со всех открытых UDP-портов идёт исходящий трафик.

Защита: Необходимо выставить ограничение на количество подключений к открытым портам и закрыть неиспользуемые порты с помощью межсетевых экранов.

Добавить правила межсетевого экрана:

1. Ограничить количество подключений.

```
# iptables -I INPUT -p udp --dport 53 -j DROP -m iplimit --iplimit-above 1
```

2. Разрешить подключение только доверенным IP-адресам.

```
# iptables -A OUTPUT -p udp --dport 53 -d 8.8.4.4 -j ACCEPT
```

3. Блокировать все остальные порты.

```
# iptables -A OUTPUT -p udp -j DROP
```

Атака произвольными пакетами. SYN-Flood

Аномалия: Кратковременный скачок используемых ресурсов процессора и многократное увеличение входящего и исходящего трафика, и их количество равно.

Признак: В TCP-стеке появляется большое количество полуоткрытых соединений со статусом SYN_RECV.

Защита: Современные реализации TCP протокола и некоторые межсетевые экраны имеют механизм защиты от SYN flood атак.

Принцип его действия заключается в следующем:

1. Серверу отправляется запрос на установление соединения, механизм регистрирует его в таблице.

2. После ответа сервера о подтверждении запроса на соединение механизм отправляет пакет серверу с подтверждением соединения и отправляет ответный пакет клиенту. В этот момент в механизме запускается таймер, отсчитывающий время на ответ от клиента.

3. При получении пакета с подтверждением соединения механизм считает запрос валидным, остановит таймер и отправит его на сервер.

4. Если пакет не пришёл в установленное время, механизм отправляет серверу запрос на удаление информации о соединении.

Принцип действие механизма основан на контроле времени задержки ответа клиента. При слишком коротком тайм-ауте будут обрываться легитимные пользователи, при длинном будут накапливаться соединения, что может привести к серьёзным последствиям.

Также можно использовать SYN cookie, или ограничить количество запросов на новые подключения от конкретного пользователя за конкретный период времени. На Рисунке 1 представлен пример настройки ядра операционной системы для защиты от SYN-Flood с помощью команды sysctl.

```
test@test:~$ sudo sysctl -w net.ipv4.tcp_syncookies=1
net.ipv4.tcp_syncookies = 1
test@test:~$ sudo sysctl -w net.ipv4.tcp_max_syn_backlog=4096
net.ipv4.tcp_max_syn_backlog = 4096
test@test:~$ sudo sysctl -w net.ipv4.tcp_keepalive_intvl=10
net.ipv4.tcp_keepalive_intvl = 10
test@test:~$ sudo sysctl -w net.ipv4.tcp_keepalive_probes=5
net.ipv4.tcp_keepalive_probes = 5
test@test:~$ sudo sysctl -w net.ipv4.conf.default.rp_filter=1
net.ipv4.conf.default.rp_filter = 1
test@test:~$ sudo sysctl -w net.ipv4.tcp_synack_retries=1
net.ipv4.tcp_synack_retries = 1
test@test:~$ sudo sysctl -w net.ipv4.tcp_fin_timeout=10
net.ipv4.tcp_fin_timeout = 10
```

Атака произвольными пакетами. ICMP-Flood

Аномалия: Очень сильная загрузка процессора и сильное увеличение входящего и исходящего ICMP трафика, равного по значениям.

Признак: Множество пакетов передаётся по протоколу ICMP.

Защита: Для противодействия атаке возможны следующие меры:

1. Отключить ответы на ICMP-запросы.

```
#sysctl net.ipv4.icmp_echo_ignore_all=1
```

2. Понизить приоритет обработки ICMP-сообщений.

3. Отбросить или фильтровать ICMP-трафик межсетевым экраном.

```
# iptables -A INPUT -p icmp -j DROP --icmp-type 8
```

4. Увеличить очередь обрабатываемых подключений.

Атака с помощью SSL

Аномалия: Резкое повышение используемых системных ресурсов.

Признак: Появляется HTTPS трафик. Большое количество обращений к SSL серверу.

Защита: Можно установить правила разрыва соединения с пользователем, выполняющего функцию повторного подтверждения больше заданного количества раз в определённый период времени. Можно использовать контроллер доставки приложений (ADC), чтобы выгрузить SSL с сервера и использовать Web Application Firewall (WAF), для просмотра трафика на наличие атак. Но грамотно спланированные атаки могут превысить количество подключений к этим устройствам и нарушить их работу. Необходимо проводить поведенческий анализ. Целью этих методов является снижение объема трафика атаки пока ресурсы сервера не смогут эффективно бороться с ним. Данный тип атак не отличается поведением от легитимного трафика на сетевом уровне. Из-за этого борьба не всегда эффективна и механизмы ослабления склонны к ложным срабатываниям.

Amplification атаки

Аномалия: Огромное увеличение входящего трафика, которое может достигать до 600 Гб/с.

Признак: Увеличение входящих UDP-пакетов на порты, используемые уязвимыми сервисами. Например, NTP 123 порт, DNS 53 порт, CharGEN 19 порт.

Защита: Такие атаки фильтруются по порту источника и сигнатурам пакетов, так как для каждой amplification атаки используется определённая уязвимая команда сервиса, сигнатура которой известна.

В межсетевом экране необходимо добавить правила:

1. Закрыть неиспользуемые UDP порты.

```
# iptables -A INPUT -p udp -j DROP
```

2. Разрешить подключение к X UDP порту только с IP-адреса используемого сервиса.

```
#iptables -A INPUT -p udp --dport X -d x.x.x.x -j ACCEPT
```

3. Пропускать только пакеты, которые имеют определённую сигнатуру 0x00=0x00000000.

```
# iptables -A INPUT -p udp --dport X -m u32 --u32 «0x00=0x00000000» -j ACCEPT
```

Межсетевой экран

Скорее всего, ранее вы уже сталкивались с таким понятием как межсетевой экран. В ядро Linux встроен свой межсетевой экран, называемый Netfilter. Управление им осуществляется с помощью утилиты Iptables.

Межсетевой экран, сетевой экран, файервол, брандмауэр – комплекс аппаратных или программных средств, осуществляющий контроль и фильтрацию проходящих через него сетевых пакетов в соответствии с заданными правилами. Основной задачей сетевого экрана является защита компьютерных сетей или отдельных узлов от несанкционированного доступа. Также сетевые экраны часто называют фильтрами, так как их основная задача – не пропускать (фильтровать) пакеты, не подходящие под критерии, определённые в конфигурации.

Рассмотрим принцип работы Netfilter. Когда сетевые пакеты попадают в сетевой интерфейс, они после ряда проверок ядром проходят последовательность так называемых цепочек. Пакет обязательно проходит через цепочку PREROUTING, после чего определяется, кому он, собственно, был адресован. Если пакет не адресован локальной системе (в нашем случае

серверу), он попадает в цепочка FORWARD, а иначе – в цепочку INPUT, после прохождения которой отдается локальным демонам или процессам. После этого при необходимости формируется ответ, который направляется в цепочку OUTPUT. После цепочек OUTPUT или FORWARD пакет в очередной раз встречается с правилами маршрутизации и направляется в цепочку POSTROUTING. В результате прохождения пакетом цепочек фильтрации несколько раз, проверка его принадлежности определенным критериям осуществляется несколько раз. В соответствии с этими проверками к пакету применяется определенное действие:

ACCEPT – пакет «принимается» и передается в следующую цепочку.

DROP – удовлетворяющий условию пакет отбрасывается и не передается в другие таблицы или цепочки.

REJECT – пакет отбрасывается, но при этом отправителю отправляется ICMP-сообщение, сообщающее об отказе.

RETURN – пакет возвращается в предыдущую цепочку и продолжает её прохождение начиная со следующего правила

SNAT – применить трансляцию источника в пакете. Используется только в цепочках POSTROUTING и OUTPUT таблицы nat.

DNAT – применить трансляцию адреса назначения в пакете. Используется в цепочках PREROUTING и (очень редко) OUTPUT в таблице nat.

Параметр	**Описание**	**Пример**
--append (-A)	Позволяет добавить в указанную цепочку и таблицу заданное правило, помещаемое в КОНЕЦ списка.	iptables -A FORWARD критерии -j действие
--delete (-D)	Позволяет удалить заданное номером или каким-либо правилом правило. В первом примере удаляются все	iptables -D 10,12 iptables -t mangle -D PREROUTING критерии -j действие

	правила с номерами 10,12 во всех цепочках, в таблицах filter.	
--rename-chain (-E)	Изменить имя цепочки.	iptables -E OLD_CHAIN NEW_CHAIN
--flush (-F)	Очищает все правила текущей таблицы. Ко всем пакетам, относящимся к уже установленным соединениям, применяется терминальное действие ACCEPT – пропустить.	iptables -F
--insert (-I)	Добавляет заданное правило в соответствии с номером.	iptables -I FORWARD 5 критерии -j действие
--list (-L)	Позволяет просматривать существующие правила (без явного указания таблицы – отображается таблица filter всех цепочек).	iptables -L
--policy (-P)	Позволяет устанавливать стандартную политику для заданной цепочки.	iptables -t mangle -P PREROUTING DROP
--replace (-R)	Заменяет заданное номером правило на заданное в критериях.	iptables -R POSROUTING 7 | критерии -j действие
--delete-chain (-X)	Удалить ВСЕ созданные вручную цепочки (оставить только стандартные INPUT, OUTPUT…)	iptables -X
--zero (-Z)	Обнуляет счетчики переданных данных в цепочке.	iptables -Z INPUT

--line-numbers	Указывать номера правил при выводе (может использоваться совместно с -L).	iptables -L --line-numbers
--help (-h)	Помощь/	iptables --help
-t таблица	Задает название таблицы, над которой необходимо совершить действие. В примере сбрасывается таблица nat во всех цепочках.	iptables -t nat -F
--verbose (-v)	Детальный вывод.	iptables -L -v
Основные правила отбора пакетов		
--protocol (-p)	Определяет протокол транспортного уровня. Опции tcp, udp, icmp, all или любой другой протокол определенный в /etc/protocols.	iptables -A INPUT -p tcp
--source(-s, --src)	IP адрес источника пакета. Может быть определен несколькими путями: Одиночный хост: host.domain.tld, или IP адрес: 10.10.10.3 Пул-адресов (подсеть): 10.10.10.3/24 или 10.10.10.3/255.255.255.0.	iptables -A INPUT -s 10.10.10.3
--destination(-d)	IP адрес назначения пакета. Может быть определен несколькими путями (см. --source).	iptables -A INPUT --destination 192.168.1.0/24
--in-interface (-i)	Определяет интерфейс, на который прибыл пакет. Полезно для NAT и машин с	iptables -t nat -A PREROUTING --in-interface eth0

	несколькими сетевыми интерфейсами. Применяется в цепочках INPUT, FORWARD и PREROUTING. Возможно использование знака +, тогда подразумевается использование всех интерфейсов, начинающихся на имя+ (например eth+ - все интерфейсы eth).	
--out-interface(-o)	Определяет интерфейс, с которого уйдет пакет. Полезно для NAT и машин с несколькими сетевыми интерфейсами. Применяется в цепочках OUTPUT, FORWARD и POSTROUTING. Возможно использование знака +.	iptables -t nat -A POSTROUTING --in-interface eth1
Неявные (необщие) параметры		
-p proto -h	Вывод справки по неявным параметрам протокола proto.	iptables -p icmp -h
--source-port (--sport)	Порт источник, возможно только для протоколов --protocol tcp, или --protocol udp.	iptables -A INPUT --protocol tcp --source-port 25
--destination-port (--dport)	Порт назначения, возможно только для протоколов --protocol tcp, или --protemocol udp.	iptables -A INPUT --protocol udp --destination-port 67

Явные параметры		
-m state --state (устарел) он же -m conntrack --ctstate	Состояние соединения. Доступные опции: NEW (Все пакеты устанавливающие новое соединение) ESTABLISHED (Все пакеты, принадлежащие установленному соединению) RELATED (Пакеты, не принадлежащие установленному соединению, но связанные с ним. Например - FTP в активном режиме использует разные соединения для передачи данных. Эти соединения связаны.) INVALID (Пакеты, которые не могут быть по тем или иным причинам идентифицированы).	iptables -A INPUT -m state --state NEW, ESTABLISHED iptables -A INPUT -m conntrack --ctstate NEW, ESTABLISHED
-m mac --mac-source	Задает MAC адрес сетевого узла, передавшего пакет. MAC адрес должен указываться в форме XX:XX:XX:XX:XX:XX.	-m mac --mac-source 00:00:00:00:00:0
Дополнительные параметры		
--to-destination	Указывает, какой IP адрес должен быть подставлен в качестве адреса места назначения. В примере во всех пакетах протокола tcp, пришедших на адрес 1.2.3.4,	iptables -t nat -A PREROUTING -p tcp -d 1.2.3.4 -j DNAT --to-destination 4.3.2.1

	данный адрес будет заменен на 4.3.2.1.	
LOG		
--log-level	Используется для задания уровня журналирования (log level). В примере установлен максимальный уровень логирования для всех tcp пакетов в таблице filter цепочки FORWARD.	iptables -A FORWARD -p tcp -j LOG --log-level debug
--log-prefix	Задает текст (префикс), которым будут предваряться все сообщения iptables. Префикс может содержать до 29 символов, включая и пробелы. В примере отправляются в syslog все tcp пакеты в таблице filter цепочки INPUT с префиксом INRUT-filter.	iptables -A INPUT -p tcp -j LOG --log-prefix INRUT-filter
--log-ip-options	Позволяет заносить в системный журнал различные сведения из заголовка IP пакета.	iptables -A FORWARD -p tcp -j LOG --log-ipoptions

Основные цепочки межсетевого экрана Netfilter:

1. PREROUTING – изначальная обработка входящих пакетов.

2. INPUT – для входящих пакетов, адресованных непосредственно локальному компьютеру.

3. FORWARD – для маршрутизируемых пакетов.

4. OUTPUT – для пакетов, исходящих с локального компьютера.

5. POSTROUTING – для окончательной обработки исходящих пакетов.

Таблицы межсетевого экрана Netfilter:

1. raw – используется для маркировки пакетов, которые не должны обрабатываться системой определения состояний. Содержится в цепочках PREROUTING и OUTPUT.

2. mangle – содержит правила модификации IP-пакетов.

3. nat – предназначена для подмены адреса отправителя или получателя. Данную таблицу проходят только первые пакеты из потока – трансляция адресов или маскировка (подмена адреса отправителя или получателя) применяются ко всем последующим пакетам в потоке автоматически. Поддерживает действия DNAT, SNAT, MASQUERADE, REDIRECT. Содержится в цепочках PREROUTING, OUTPUT, и POSTROUTING.

4. filter – основная таблица, используется по умолчанию если название таблицы не указано. Используется для фильтрации пакетов. Содержится в цепочках INPUT, FORWARD, и OUTPUT.

Пример создания правила для межсетевого экрана

Рассмотрим две цепочки, задающие два основных правила Iptables – PREROUTING и FORWARD.

```
iptables -t nat -A PREROUTING -i eth0 -j DNAT --to-destination 192.168.57.102
iptables -A FORWARD -d 192.168.57.102 -j ACCEPT
```

Первая из них определяет первоначальную обработку всех пакетов, приходящих на адаптер eth0:

1. -t определяет подключаемую таблицу, в данном случае – nat – для подмены адреса отправителя или получателя.

2. -A – выбор цепочки.

3. -i – входящий интерфейс.

4. -j – действие с пакетами, удовлетворяющими условию – в данном случае DNAT – подмена адреса получателя.

5. --to-destination – выбор адреса, на который перенаправляются пакеты.

Вторая определяет проброс пакетов через сервер:

6. -A – выбор цепочки.

7. -d – выбор адресата.

8. -j – выбор действия.

Web Application Firewall

WAF (Web Application Firewall) – это межсетевые экраны, работающие на прикладном уровне и осуществляющие фильтрацию трафика Web-приложений. Эти средства не требуют изменений в исходном коде Web-приложения и, как правило, защищают Web-сервисы гораздо лучше обычных межсетевых экранов и средств обнаружения вторжений.

Основные преимущества:

1. Анализ поведения пользователя в используемом приложении.

2. Позволяет осуществлять мониторинг HTTP трафика и проводить анализ событий в реальном режиме времени.

3. Предотвращение вредоносных запросов.

4. Распознавание большинства опасных угроз.

5. Дополнение сетевых средств безопасности.

6. Просматривать детальные отчеты об атаках и попытках взлома.

Задания к лабораторной работе

Часть 1

1. Установить необходимое ПО и настроить сервера.

2. Сделать скриншоты мониторинга ресурсов сервера и сетевого трафика в штатном режиме.

3. Провести хотя бы две атаки из списка.

4. Сделать скриншоты мониторинга для каждой проведённой атаки.

5. Написать какие аномалии вы наблюдали в используемых ресурсах сервера, характеристиках сервера, и что является признаком атаки.

6. Защитить сервер от атак и сделать скриншоты результатов.

Часть 2

1. Установите web-сервер <sudo apt-get install apache2>

2. Просмотрите список текущих правил iptables таблицы filter

sudo iptables -L

3. Вы увидите, что список содержит три цепочки по умолчанию (INPUT, OUTPUT и FORWARD), в каждой из которых установлена политика по умолчанию (на данный момент это ACCEPT).

4. С помощью команды <sudo iptables -S> данный список можно просмотреть в другом формате, который отражает команды, необходимые для активации правил и политик.

5. Чтобы сбросить текущие правила (если таковые есть), наберите:

sudo iptables -F

6. Цепочка INPUT отвечает за входящий траффик.

7. Чтобы внести локальный интерфейс выполните:

sudo iptables -A INPUT -i lo -j ACCEPT

8. Чтобы заблокировать весь исходящий трафик, кроме портов для SSH и веб-сервера, нужно сначала разрешить подключения к этим портам. В цепочку ACCEPT добавьте два порта (порт SSH 22 и порт http 80), что разрешит трафик на эти порты.

sudo iptables -A INPUT -p tcp -m tcp --dport 22 -j ACCEPT

sudo iptables -A INPUT -p tcp -m tcp --dport 80 -j ACCEPT

9. В данной работе мы не используем SSH. Так что удалим ненужное правило. Для этого:

sudo iptables -D INPUT -p tcp -m tcp --dport 22 -j ACCEPT

10. Нужно добавить еще одно правило, которое позволит устанавливать исходящие соединения (т.е. использовать ping или запускать обновления программного обеспечения):

sudo iptables -I INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

11. Создав все эти правила, можно заблокировать все остальное и разрешить все исходящие соединения.

sudo iptables -P OUTPUT ACCEPT

sudo iptables -P INPUT DROP

12. Просмотрите список правил

sudo iptables -L

13. Добавим еще несколько правил для блокировки наиболее распространенных атак. Для начала нужно заблокировать нулевые пакеты <sudo iptables -A INPUT -p tcp --tcp-flags ALL NONE -j DROP>.

14. Следующее правило отражает атаки syn-flood <sudo iptables -A INPUT -p tcp ! --syn -m state --state NEW -j DROP>. Теперь фаервол не будет принимать входящих пакетов с tcp-флагами. Нулевые пакеты, по сути, разведывательные. они используются, чтобы выяснить настройки сервера и определить его слабые места.

15. Далее нужно защитить сервер от разведывательных пакетов XMAS <sudo iptables -A INPUT -p tcp --tcp-flags ALL ALL -j DROP>. Теперь сервер защищен от некоторых общих атак, которые ищут его уязвимости.

16. Со второй виртуальной машиины, на которую установите nmap, проведите XMAS сканирование <sudo nmap -sX>.

17. По умолчанию все не сохраненные правила действуют до следующей перезагрузки сервера; сразу же после перезагрузки не сохраненные правила будут потеряны. Самый простой способ загрузить пакет iptables-persistent <sudo apt-get install iptables-persistent>. Во время инсталляции пакет уточнит, нужно ли сохранить текущие правила для дальнейшей автоматической загрузки, если текущие правила были протестированы и соответствуют всем требованиям, их можно сохранить.

Часть 3

1. На виртуальную машину Server необходимо установить набор программного обеспечения LAMP (Linux Apache MySQL PHP)

sudo apt-get update

sudo apt-get install tasksel

sudo tasksel

Выберите LAMP. Чтобы отметить выделите и нажмите пробел. И нажмите Enter.

2. Установите mod_security <sudo apt-get install libapache2-mod-security2>

3. Выполните команду <sudo apachectl -M | grep --color security2>. Если на экране появился модуль по имени security2_module (shared), значит, все прошло успешно.

4. В каталоге логов Apache можно найти новый лог-файл для mod_security. /var/log/apache2/modsec_audit.log

5. Установка ModSecurity включает в себя конфигурационный файл, который нужно переименовать: <sudo mv /etc/modsecurity/modsecurity.conf-recommended /etc/modsecurity/modsecurity.conf>.

6. Затем перезапустите Apache <sudo service apache2 reload>.

7. Стандартный конфигурационный файл настроен на DetectionOnly, то есть, фаервол только отслеживает логи, при этом ничего не блокируя. Чтобы изменить это поведение, отредактируйте файл modsecurity.conf: <sudo nano /etc/modsecurity/modsecurity.conf>

8. Найдите в файле строку: «SecRuleEngine DetectionOnly». И измените ее так: «SecRuleEngine On».

9. Найдите «SecResponseBodyAccess On» и замените на «SecResponseBodyAccess Off». Эта директива отвечает за буферизацию тела ответа; ее рекомендуется включать, только если требуется обнаружение и предохранение от утечки данных. Включенная директива (SecResponseBodyAccess On) не только будет использовать больше ресурсов сервера, но и увеличит размер лог-файла, следовательно, ее желательно отключить.

10. По умолчанию mod_security поставляется с базовым набором правил CRS (Core Rule Set), которые находятся в /usr/share/modsecurity-crs/

11. Чтобы подгрузить эти готовые правила, нужно, чтобы веб-сервер Apache читал указанные выше каталоги. Для этого отредактируйте файл mod-security.conf:

sudo nano /etc/apache2/mods-enabled/mod-security.conf

12. Между <IfModule security2_module> </IfModule> внесите следующие параметры:

Include "/usr/share/modsecurity-crs/*.conf"

Include "/usr/share/modsecurity-crs/activated_rules/*.conf"

13. Директория activated_rules аналогична директории Apache mods-enabled. Правила доступны в каталогах: /usr/share/modsecurity-crs/base_rules ; /usr/share/modsecurity-crs/optional_rules ; /usr/share/modsecurity-crs/experimental_rules

14. Чтобы активировать правила, нужно создавать символические ссылки в каталоге activated_rules. <cd /usr/share/modsecurity-crs/activated_rules/>

15. Добавьте несколько правил, например <sudo ln -s /usr/share/modsecurity-crs/base_rules/modsecurity_crs_30_http_policy.conf> ; <sudo ln -s /usr/share/modsecurity-crs/base_rules/modsecurity_crs_49_generic_attacks.conf>

16. Чтобы новые правила вступили в исполнение, нужно перезапустить Apache <sudo service apache2 reload>

Вопросы к лабораторной работе

Часть 1

1. Принцип работы каждой из сетевых атак типа «отказ в обслуживании»?

2. Какие аномалии в сетевом трафике и ресурсах сервера вы заметили в каждой атаке?

3. Что является признаком конкретной атаки?

4. В чем отличие DDoS от DoS?

5. Что такое UDP-флуд?

6. Как защититься от различных атак DoS/DDoS?

7. Что такое SYN-флуд? Укажите решение защиты от SYN-флуда с помощью iptables.

8. Как определить, что осуществляется атака SYN-флуд?

9. Как осуществить защиту от HTTP-флуда?

10. Как работают атаки типа SlowLoris?

11. Как совершить DDoS-атаку на почтовый сервер и в дальнейшем реализовать его защиту?

12. Как работают атаки типа SSL? Как определить, что осуществляется атака?

13. Перечислите Amplification атаки с коэффициентами их усиления.

14. Опишите теоретически возможность расследования DoS/DDoS атак, установление личности атакующего, определение источников атак.

Часть 2

1. Что такое межсетевой экран?

2. Для чего используется межсетвой экран?

3. Принцип работы Netfilter.

4. Таблицы межсетевого экрана Netfilter. Для чего они используются?

5. Что такое правила межсетевого экрана?

6. Как создавать правила для межсетевого экрана утилитой Iptables?

7. Как сохранить правила для последующей автозагрузки?

8. Что такое Web Application Firewall?

9. Как настроить правила в WAF mod_security?