Проект OWASP WebGoat

Виктор Борисов

May 29, 2016

Contents

1 Цель работы

Изучить описания 10 самых распространенных веб-уязвимостей согласно рейтингу OWASP.

2 Ход работы

2.1 Изучение

2.1.1 Описания десяти самых распространенных веб-уязвимости согласно рейтингу OWASP

- **Injection** Атака на интепритатор машины-цели, позволяя выполнять произвольный код от ее имени. Чаще всего встрачаются в SQL, LDAP, Xpath, или NoSQL запросах, парсерах xml, аргументах программ и т.д.
- **Broken Authentication and Session Management** Атака на уязвимости систем авторизации и управления сессиями с целью кражи и/или выполнения каких либо действий от чужого имени.
- **Cross-Site Scripting** Атака на браузер путем подмены загружаемых скриптов. В результате злоумышлиниками может быть получена почти любая информация.
- **Insecure Direct Object References** Суть атаки - изменение некого объекта, используемого в авторизированной сессии.

  Изменение параметра позволит отправлять измененные запросы от имени авторизированного пользователя.
- **Security Misconfiguration** Ошибки в конфигурации. Атакующий может получить доступ к файлам, акаунтам, системе и т.д.
- **Sensitive Data Exposure** Кража ценной/личной информации. Атака сложна если используется шифрование. В таком случае данные крадутся косвенными методами: на стороне клиента, когда данные уже рашифрованы, man-in-the-middle атака и другими способами.
- **Missing Function Level Access Control** Доступ неавторизированного пользователя к привелегированным функциям.
- **Cross-Site Request Forgery** Атака путем выполнения запросов к некоторому защищенному ресурсу от его имени авторизованного пользователя. Недостаток - атакующий не может перехватить ответ от ресурса. В этом случае вводят так называемые CSRF-токены: каждый последующий пакет от клиента содержит токен, полученный в пердыдущем ответе сервера.
- **Using Components with Known Vulnerabilities** Атака на уязвимый компонент системы, выявленный в результате сканирования.
- **Unvalidated Redirects and Forwards** Скрытые ссылки в картинках, фреймах и т.д., ведущих на доверенный сайт. Позволяет произвести любой запрос.

2.2 Подготовка рабочей среды

Необходимо заупстить и настроить WebGoat, ZAP, Mantra. WebGoat запущен на порту по умолчанию - 8080. ZAP настраивается на работу на 8081 порт. И такой же порт прописывается в настройках прокси в Mantra.

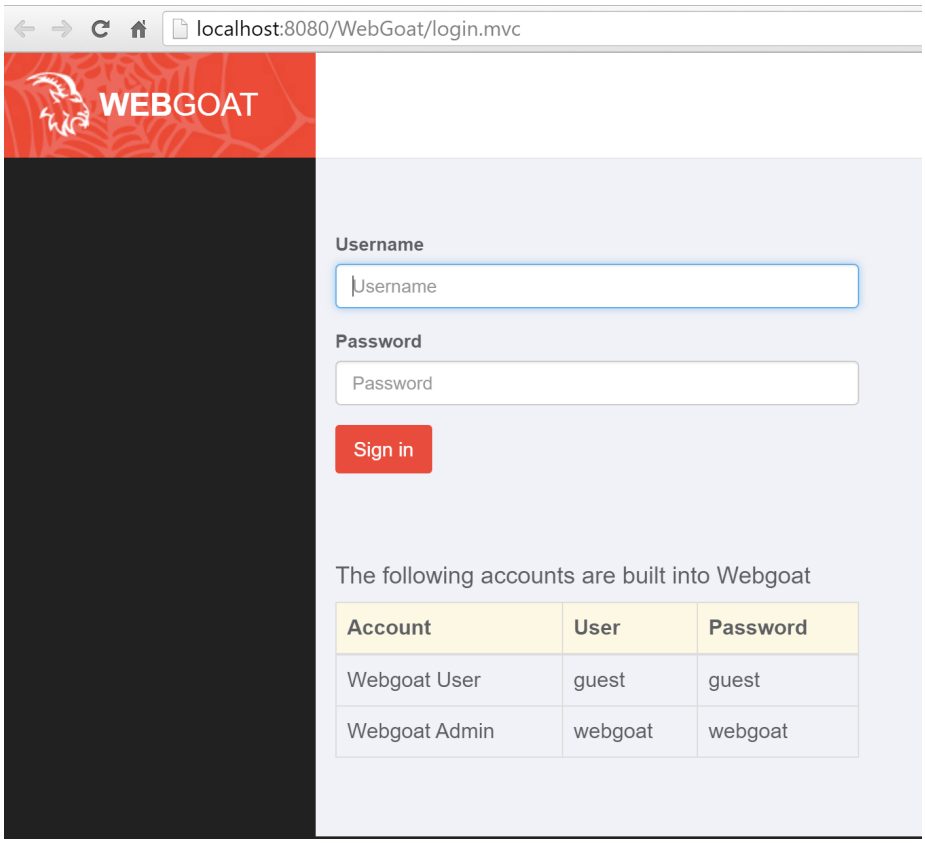

Figure 1: Запуск WebGoat

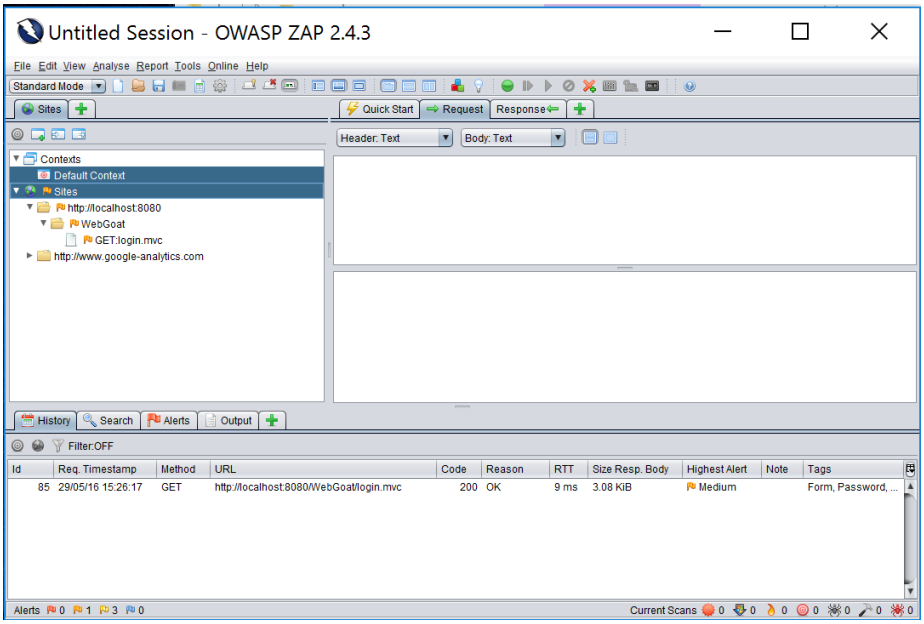

Figure 2: Запуск ZAP.

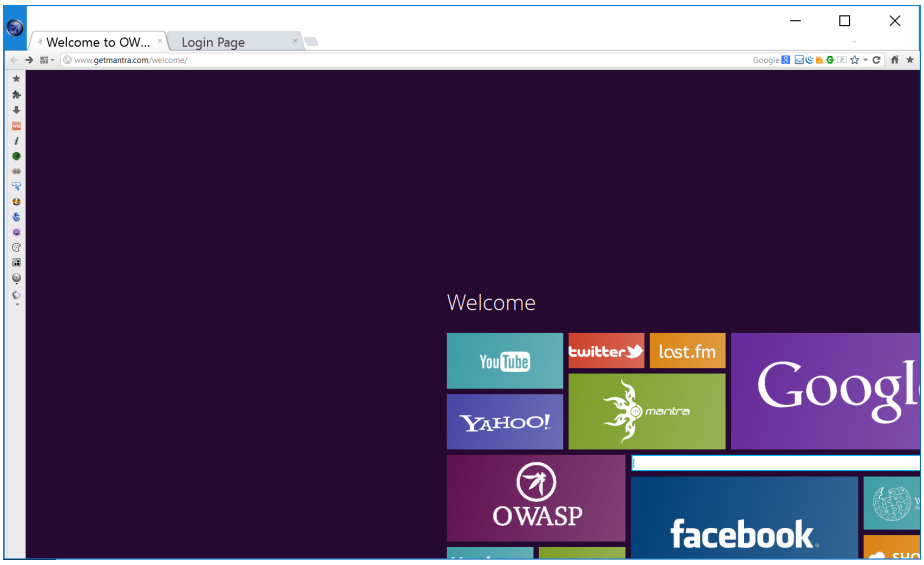

Figure 3: Запуск Mantra.

2.3 Практическое задание

2.3.1 Недостатки контроля доступа

Bypass a Path Based Access Control Scheme
Получение доступа к файлу, находящемуся вне нашей директории. Для

этого перехватим запрос и заменим имя файла на "../../../../WEB-INF/spring-security.xml". Доступ к файлу получен.

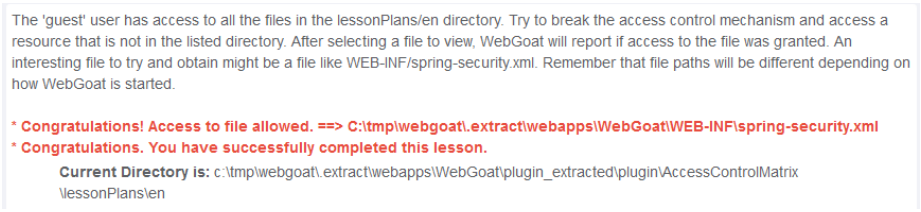

Figure 4: Доступ к файлу вне директории.

Bypass Bussines Layer Access
Заходим под аккаунтом, не имеющем права удаления, например, Tom (пароль tom), и, перехватывая запрос View Profile, подменим вызываемый метод на DeleteProfile. Профиль другого сотрудника удален.

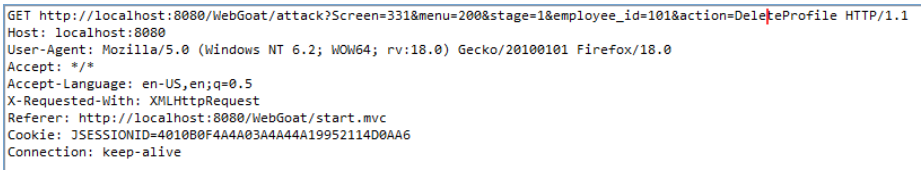

Figure 5: Удаление.

Bypass Data Layer Access
В данном задании необходимо подменить ID пользователя (аналогично предыдущему заданию), тем самым получив данные другого пользователя.

2.3.2 Безопасность AJAX

Client Side Filtering
Показывает недостатки фильтрации полученных данных на клиентской стороне. Так как можно получить информацию, непредназначенную для данного пользователя.
Dom based cros-site scripring
Показывает необходимость экранирования входных данных. Например, выполнить произвольный html код.
DOM injection
Показывает принцип работы DOM injection. Используя инъекцию можно изменить свойства элементов страницы. Например, изменить submit.disabled на false при помощи перехвата запроса и инъекции следующей строки:

```
document.forms[0].SUBMIT.disabled=false;
```

Кнопка станет активной.
XML Injection
Показывает пример XML инъекции. В перехваченный запрос нужно вставить

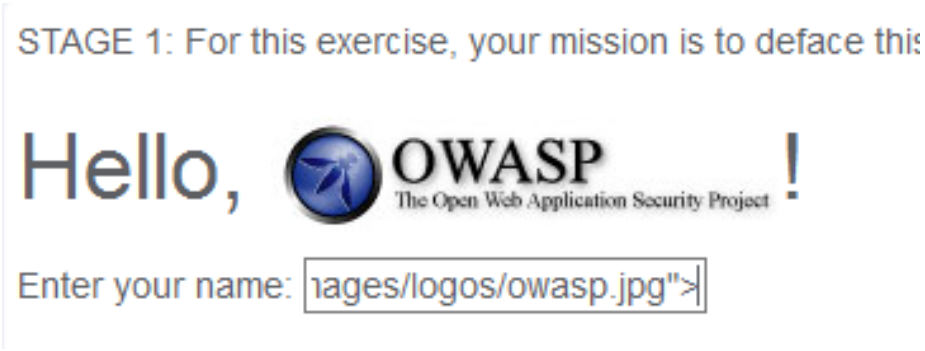

Figure 6: Результат подмены.

```
<root>
<reward>WebGoat Mug 20 Pts</reward>
<reward>WebGoat t-shirt 50 Pts</reward>
<reward>WebGoat Secure Kettle 30 Pts</reward>
<reward>WebGoat Core Duo Laptop 2000 Pts</reward>
<reward>WebGoat Hawaii Cruise 3000 Pts</reward>
</root>
```

JSON Injection
Аналогично предыдущему заданию, но теперь JSON инъекция

```
{
"From": "Boston",
"To": "Seattle",
"flights": [
{"stops": "0", "transit" : "N/A", "price": "$20"},
{"stops": "2", "transit" : "Newark,Chicago", "price": "$300"}
]
}
```

Третья часть урока показывает, что необходимо делать валидацию данных не только на клиенте, но и на сервере, чтобы исключить подмену ajax-запросов. Изменив скрипт обработки запроса мы исключили возможность выдачи лишних данных клиенту.
Седьмая часть - не стоит делать проверку на клиентской стороне. Необходимо найти клиентскую функцию для отправки и вызвать ее.

```
submitData(555, 1000000)
```

Insecure client storage
Показывает, что не стоит использовать вводимые поля помеченные атрибутом disabled или readonly, так, например, можно изменить стоимость товара при оплате.
Dangerous use of eval
Опасность функции eval

```
‘)%3Balert(document.cookie
```

2.3.3 Недостатки аутентификации

Password strength
Показывает зависимость сложности подбора пароля от его длины и набора символов.
Forgot password
Показывает, что механизм восстановления пароля не должен быть простым, иначе нет смысла в сложном пароле.
Multi level login 1
При перехвате пакета выставляем hiddentan=1 и получаем список TAN.
Multi level login 2
Авторизуемся под пользователем Joe и используем его TAN, перехватыем пакет и изменяет пользователя на Jane.

2.3.4 Переполнение буфера

Перехватываем пакет, добавляем в значение аргумента roomno >4096 символов.

2.3.5 Качество кода

В коде страницы можно найти логин и пароль администратора.

2.3.6 Многопоточность

Thread safety problem
При одновременной работе нескольких пользователей возможно перемешивание получаемых данных. Открываем два окна и вводим имена пользователей.
В некоторых случаях можно получить чужие данные.
Shopping cart concurrency flew
Открываем два окна, в одном делаем большую покупку, в другом - маленькую. Продолжаем маленькую покупку и обновляем большую. При подтверждении платим за маленькую, а получаем большую покупку.

2.3.7 Межсайтовое выполнение сценариев

С помощью XSS и HTML можно заменять элементы страницы на фиктивные, при этом пользователь не заметит изменений.

2.3.8 Неправильная обработка ошибок

В перехваченном пакете достаточно удалить поле пароля и авторизация пройдет успешно.

2.3.9 Недостатки приводящие к осуществлению инъекций (SQL и прочее)

- Command injection Перехватываем запрос, добавляем к имени файла строку:

  `%22%3B%20netstat%20-a`

- Numeric SQL injection Перехватываем запрос. Модифицируем:

```
station=101or%201%3D1&SUBMIT=Go!
```

- Log spoofing Перехватываем запрос, меняем имя на следующее:

  ```
  somename
  Admin succefully entered!
  ```

  В результате, в логе создается видимость того, что админ авторизовался.

- SQL Injection Перехватываем сообщение. В качестве пароля вводим:

  ```
  123123%27%20OR%20%271%27%3D%271
  ```

  При попытке получить данные на втором шаге получаем сообщение:

  ```
  THIS LESSON ONLY WORKS WITH THE DEVELOPER VERSION OF WEBGOAT
  ```

- String sql injection Аналогично. Вместо имени вводим 12345' OR 'a' = 'a. Получаем все возможные значения.

- Modify Data with SQL INJECTION Вместо имени вводим:

  ```
  UPDATE salaries SET salary=5 WHERE userid='jsmith
  ```

- Database backdoors По такой же схеме можно добавлять и триггеры:

  ```
  101; CREATE TRIGGER bckDoor BEFORE INSERT ON employee FOR EACH ROW BEGIN UPDATE emplo
  ```

2.3.10 Отказ в обслуживании

ZipBomb
Создается файл zip, содержащий большое количество одинаковых символов. Такой тип файлов хорошо поддается сжатию, поэтому при распаковке займет очень много дискового пространства.
DoS from Multiple logins
Используем инъекцию: вместо пароля вводим

```
12345' or '1' or '1
```

Возвращается

```
101 jsnow passwd1
102 jdoe passwd2
103 jplane passwd3
104 jeff jeff
105 dave dave
```

Авторизуемся этими данными. Получаем отказ в обслуживании из-за большого количества сессий.

2.3.11 Небезопасное сетевое взаимодействие

При передаче пароля вне защищенного соединения его можно легко перехватить, для избежания этого необходимо использовать https + TLS.

2.3.12 Небезопасная конфигурация

Показывает необходимость сокрытия расположения панели администрирования, чтобы предотвраить нежелательный доступ к ней.

2.3.13 Небезопасное хранилище

Показываются различные алгоритмы кодирования текста.

2.3.14 Исполнение злонамеренного кода

ВОзможность загрузки собственного скрипта и выполнение его.

```
<html>
<%
java.io.File fil = new java.io.File("c:\tmp\webgoat\.extract\webapps\WebGoat\mfe_target\gu
file.createNewFile();
%>
</html>
```

2.3.15 Подделка параметров

Bypas HTML Field Restrictions
При перехвате пакета изменить значение полей и добавить disabledinput.
Exploit Hidden Fields
Аналогично.
Exploit unchecked email
В поле собщения вводим тело скрипта

```
<script>alert("hi")</script>
```

Для отправки скрипта на другой адрес меняем в перехваченном пакете значение аргумента "to":

```
gId=GMail+id&gPass=password&subject=Comment+for+WebGoat&
to=friend%40owasp.org&msg=%3Cscript%3Ealert%28%22hi%22
%29%3C.script%3E&SUBMIT=Send%21
```

Bypass Client Side JavaScript Validation
Аналогично.

2.3.16 Недостатки управление сессией

Hijack a Session
Задание показывает необходимость использования сложных механизмов генерации пользовательских идентификаторов, чтобы нельзя былл с легкостью перехватить чужую сессию.
Session fixation
Отправка электронного письма, содержащего в себе ссылку с добавленным SID (Session ID). После того как жертва осуществит вход по данной ссылке нам будет доступна сессия с переданным нами ID.

2.3.17 Безопасность веб-сервисов

3 Выводы

В ходе работы было с помощью средства WebGoat были изучены и отработаны наиболее распространненые уязвимости веб-приложений, а так же способы их устранения.