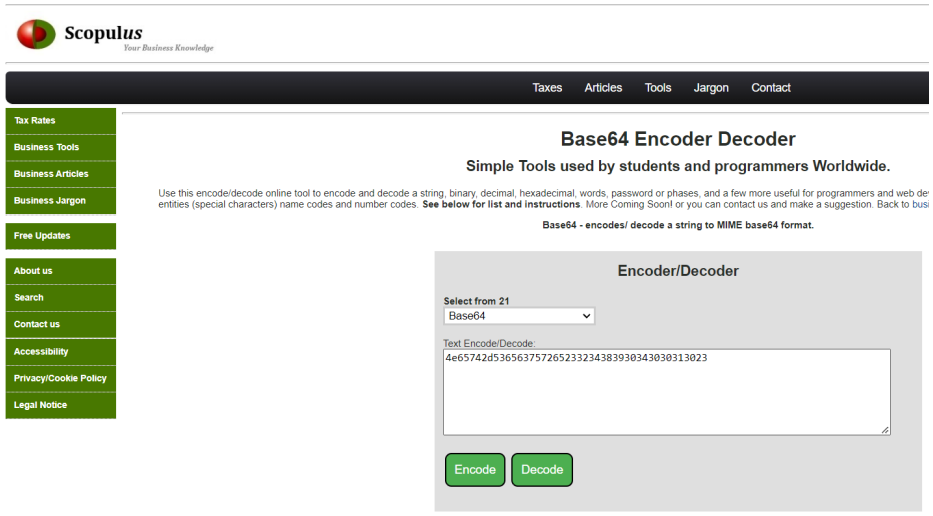
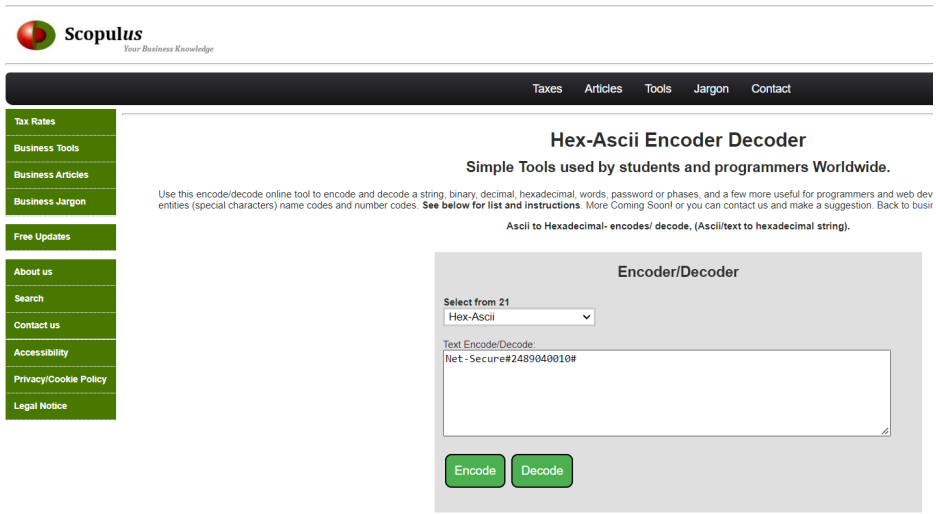
NET SQUARE MACHINE TEST

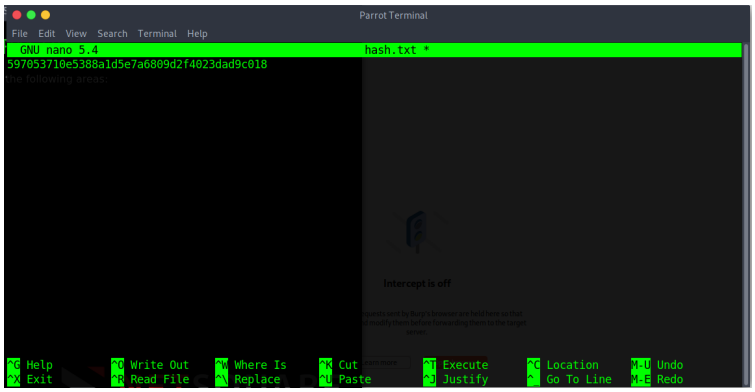
1.encrypt/hash/encode 17/A3

a. Decode as base 62 (4times) + Hex




b. Use john and rockyou.txt to crack the hash



```
┌──[vaishakh@KKz ]─[~]
└─$ john --wordlist=rockyou.txt hash.txt
Warning: detected hash type "Raw-SHA1", but the string is also recognized as "Raw-SHA1-AxCrypt"
Use the "--format=Raw-SHA1-AxCrypt" option to force loading these as that type instead
Warning: detected hash type "Raw-SHA1", but the string is also recognized as "Raw-SHA1-Linkedin"
Use the "--format=Raw-SHA1-Linkedin" option to force loading these as that type instead
Warning: detected hash type "Raw-SHA1", but the string is also recognized as "ripemd-160"
Use the "--format=ripemd-160" option to force loading these as that type instead
Warning: detected hash type "Raw-SHA1", but the string is also recognized as "has-160"
Use the "--format=has-160" option to force loading these as that type instead
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-SHA1 [SHA1 128/128 SSE2 4x])
Warning: no OpenMP support for this hash type, consider --fork=5
Press 'q' or Ctrl-C to abort, almost any other key for status
cassidy          (?)
1g 0:00:00:00 DONE (2022-10-22 15:25) 50.00g/s 77200p/s 77200c/s 77200C/s chopper..134679
Use the "--show --format=Raw-SHA1" options to display all of the cracked passwords reliably
Session completed
```

c. Base 64 + ECB mod and 128 bit key to decrypt

## AES Online Decryption

Enter text to be Decrypted

umFoNEptlUGGXR8MZBNWnT8+GYxpHKwgQUnNQ5CPDB4=

Input Text Format: ◉Base64 ○Hex

Select Cipher Mode of Decryption

ECB

Key Size in Bits

128

Enter Secret Key used for Encryption

0fffff713370ffff

Decrypt

AES Decrypted Output **(Base64)**:

TmV0LVNlY3VyZSM3NjEyMjI3NDQxIw==

Decode to Plain Text

Net-Secure#7612227441#

Challenge List

## OK, let us begin with observation skill

**A3:2017-Sensitive Data Exposure**

Over the last few years, this has been the most common impactful attack. The most common flaw is simply not encrypting sensitive data. When crypto is employed, weak key generation and management, and weak algorithm, protocol and cipher usage is common, particularly for weak password hashing storage techniques. For data in transit, server-side weaknesses are mainly easy to detect, but hard for data at rest.

Use your powers of observation and identify algorithm for decode/dehashing/decrypt the following data.
In encryption section, we are using Symmetric-key algorithm with **ECB** mode and **128** sized key and key is **"0fffff713370ffff"**.

Encoded DATA :

```
Vkd0a1ZrMXJOVlZaZWtKT1lsWkZlRlJZY0ZwTlZUVnhWRlJPVDFa
SFRqVlViWEJXWlZVeE5sUlliRTVsYkVZMlZEQlNUazVWTVRaUldI
QlBVa1V4TTFSWWNFSmxhekZWVkZoa1RtRnJNRGs9
```

Hashed DATA : 597053710e5388a1d5e7a6809d2f4023dad9c018

Encrypted DATA : umFoNEptlUGGXR8MZBNWnT8+GYxpHKwgQUnNQ5CPDB4=

Encoded Value : Net-Secure#2489040010#
Hashing Value : cassidy
Encrypted Value : Net-Secure#7612227441#

submit

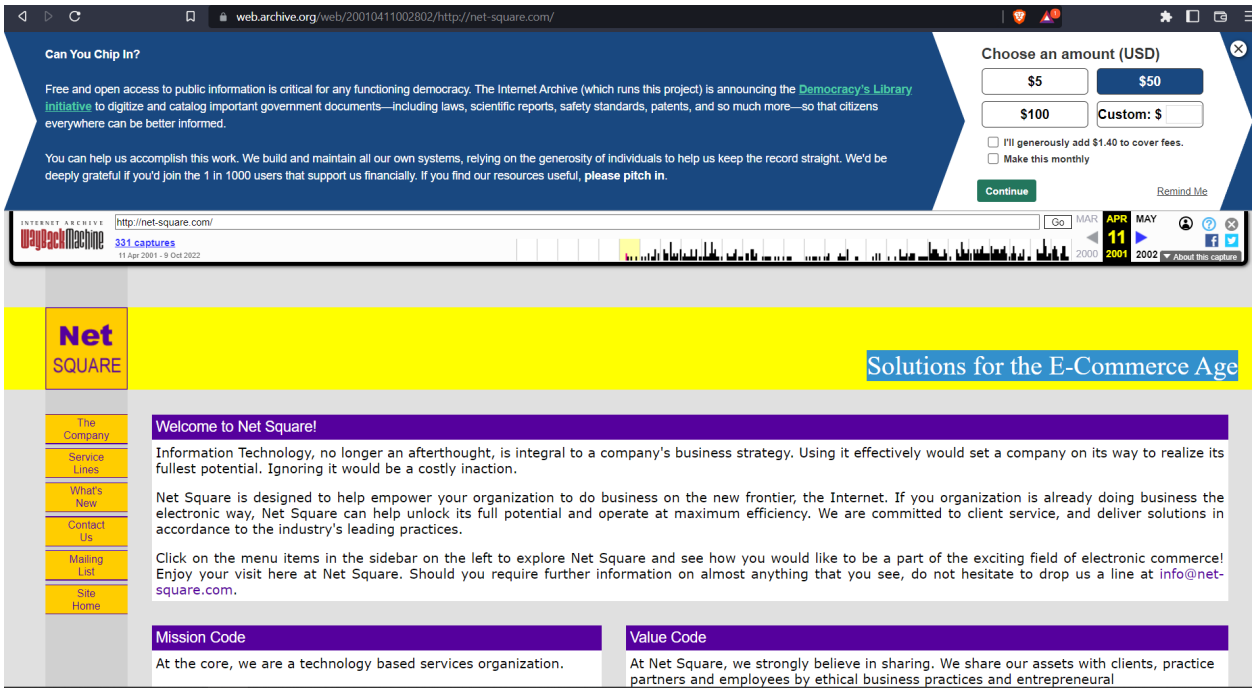
| Challanges | Value | Result |
|---|---|---|
| Encoding | Net-Secure#2489040010# | passed |
| Hashing | cassidy | passed |
| Encryption | Net-Secure#7612227441# | passed |

Yes you did! Continue on to the next challenge

NETSQUARE
secure • innovate • automate

2. Time traveller

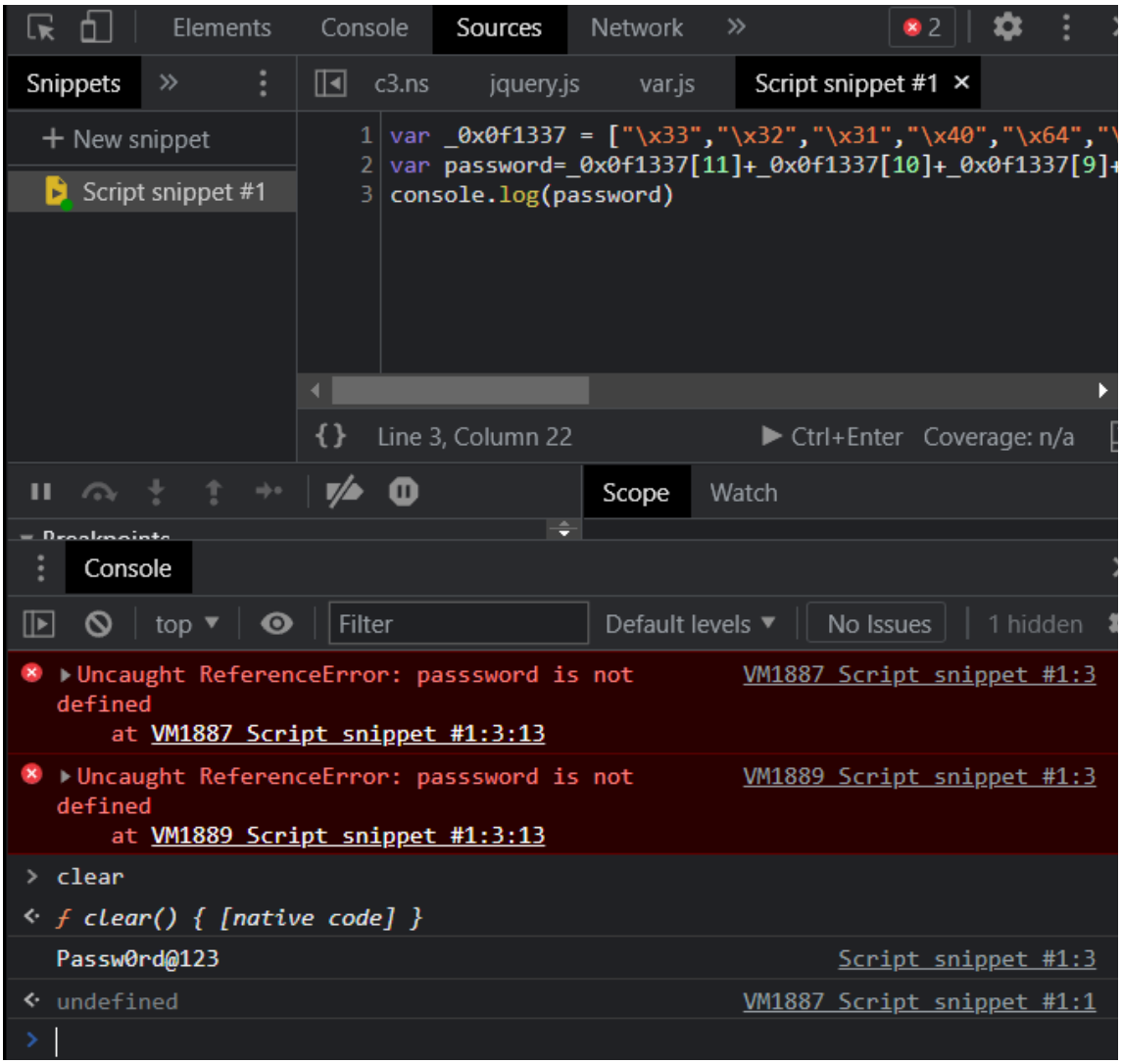
Wayback machine to achieve old version

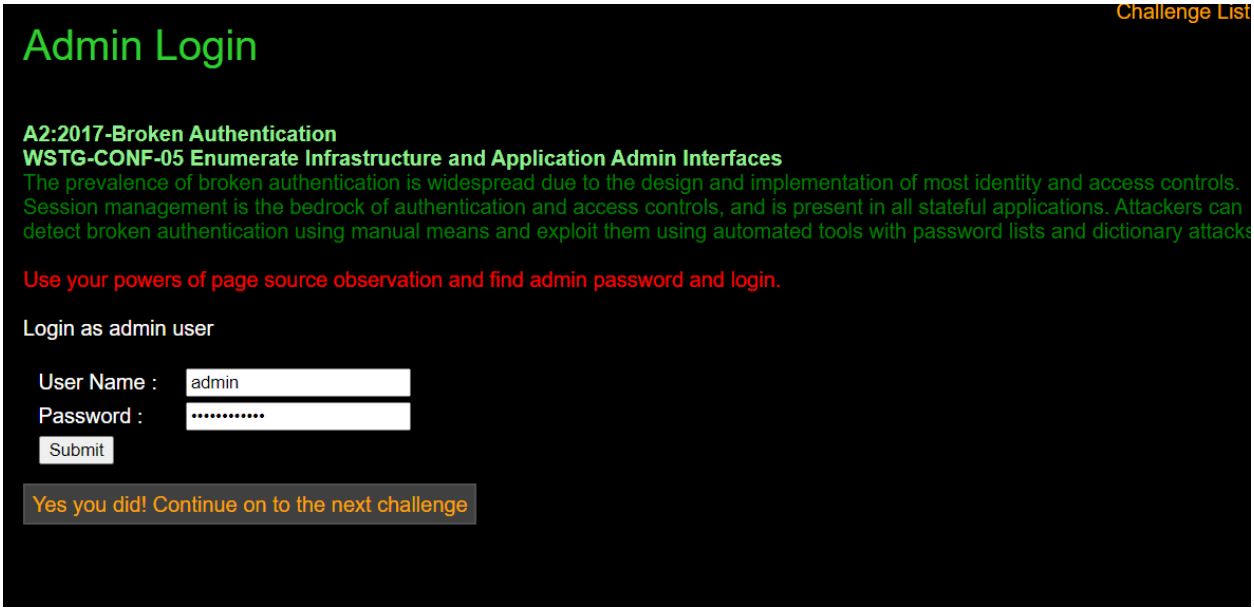



3. User Login Bypass

Insepcted code and find a js file var.js in /admin/ path

Run the code with a console.log(password)



4. Fingerprinting

Intercepted response in burpsuit to find details

Response from http://enigma.ctf.ns.exploitlab.net:80/web/c4.ns [162.252.242.82]

Forward | Drop | Intercept is on | Action | Open Browser

Pretty | Raw | Hex | Render

```
1  HTTP/1.1 200 OK
2  Date: Sat, 22 Oct 2022 10:13:43 GMT
3  Server: Microsoft-IIS/10.0
4  X-Frame-Options: SAMEORIGIN
5  X-Content-Type-Options: nosniff
6  Application-Engine: NS-ENG
7  Application-Server: Microsoft-IIS/10.0
8  Expires: Thu, 19 Nov 1981 08:52:00 GMT
9  Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
10 Pragma: no-cache
11 Vary: Accept-Encoding
12 Content-Length: 1312
13 Connection: close
14 Content-Type: text/html; charset=UTF-8
15
```

Challenge List

# Fingerprinting

**WSTG-INFO-02 Fingerprint Web Server**

Web server fingerprinting is the task of identifying the type and version of web server that a target is running on. While web server fingerprinting is often encapsulated in automated testing tools, it is important for researchers to understand the fundamentals of how these tools attempt to identify software, and why this is useful.

Accurately discovering the type of web server that an application runs on can enable security testers to determine if the application is vulnerable to attack. In particular, servers running older versions of software without up-to-date security patches can be susceptible to known version-specific exploits.

What web server is this site running on? Microsoft-IIS/10.0

What is the back-end application engine? NS-ENG

Did I get those right?

Yes you did! Continue on to the next challenge

5. HTTP Form Manipulation

Inspected the page and changed values in form

```
<td>
  <input type="text" name="my_textfield" value="more than 15 chars" minlength=
  "15">
  <span style="color: red;">make this value more than 15 characters</span>
</td>
</tr>
<tr>
  <td>Dropdown</td>
  <td>
    <select name="my_dropdown">
      <option value="redhat" selected>redhat</option> slot
      <option value="whitehat">White Hat</option> slot
      <option value="greyhat">Grey Hat</option> slot
    </select>
    <span style="color: red;">make this "redhat"</span>
  </td>
</tr>
<tr>
  <td>Radio Button</td>
  <td>
    <input type="radio" name="my_radio" value="symbian" checked>
    "symbian"
    <input type="radio" name="my_radio" value="ios">
    "iOS "
    <input type="radio" name="my_radio" value="windows8">
    "Windows 8 "
    <input type="radio" name="my_radio" value="blackberry">
    "BlackBerry "
    <span style="color: red;">make this "symbian"</span>
  </td>
</tr>
<tr>
... <td> == $0
    <input type="hidden" value="Hacked!" name="my_hidden">
  </td>
</tr>
<tr>…</tr>
```

Challenge List

# Play with HTTP Forms

**A6:2017-Security Misconfiguration**

Security misconfiguration can happen at any level of an application stack. Some time developer puts validation only on client side and missed validating parameter on server side.

Text Field max 15 chars make this value more than 15 characters

Dropdown Black Hat make this "redhat"

Radio Button Android iOS Windows 8 BlackBerry make this "symbian"

Submit

| Name | Value | Result |
|---|---|---|
| my_textfield | more than 15 chars | passed |
| my_dropdown | redhat | passed |
| my_radio | symbian | passed |
| Who am I? | Hacked! | passed |

Level cleared! Next challenge

6. Client side input validation bypass

Intercepted the response in burpsuit and changed form validation script

Response from http://enigma.ctf.ns.exploitlab.net:80/web/c6.ns [162.252.242.82]

Forward | Drop | Intercept is on | Action | Open Browser

Pretty | Raw | Hex | Render

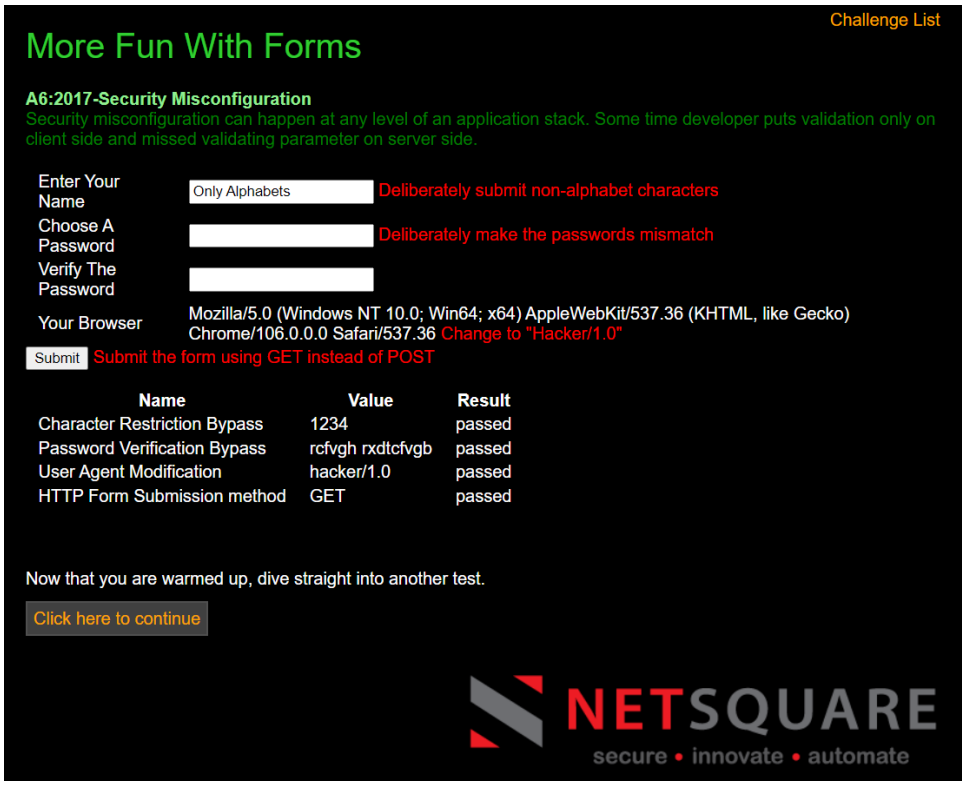
```
      function validateForm() {
        var name = document.forms["moreformplay"]["my_name"].value;
        var pass1 = document.forms["moreformplay"]["my_pass1"].value;
        var pass2 = document.forms["moreformplay"]["my_pass2"].value;
        var out_name = document.getElementById('out_name');
        var out_pass1 = document.getElementById('out_pass1');
        var out_pass2 = document.getElementById('out_pass2');
        out_name.innerHTML = "";
        out_pass1.innerHTML = "";
        out_pass2.innerHTML = "";
        if(!name.match("^[a-zA-Z ]+$")) {
          out_name.innerHTML = "Alphabets only please!";
          return true;
        }
        if(pass1 != pass2 || pass1 == "" || pass2 == "") {
          out_pass1.innerHTML = "Passwords don't match";
          return true;
        }
        else {
          return false;
        }
      }
    </script>
  </head>
  <body>
    <h1>
      More Fun With Forms
    </h1>
    <div style="color: green">
      <b style="color: #90ee90">
        A6:2017-Security Misconfiguration
      </b>
      <br>
      Security misconfiguration can happen at any level of an application stack. Some time developer puts validation only on client side
      and missed validating parameter on server side.<br>
      <br>
    </div>
    <div id="score">
      <a href="../score.ns">
        Challenge List
      </a>
    </div>
    <form name="moreformplay" method="get" onsubmit="return validateForm()">
```

Request to http://enigma.ctf.ns.exploitlab.net:80 [162.252.242.82]

Forward | Drop | Intercept is on | Action | Open Browser

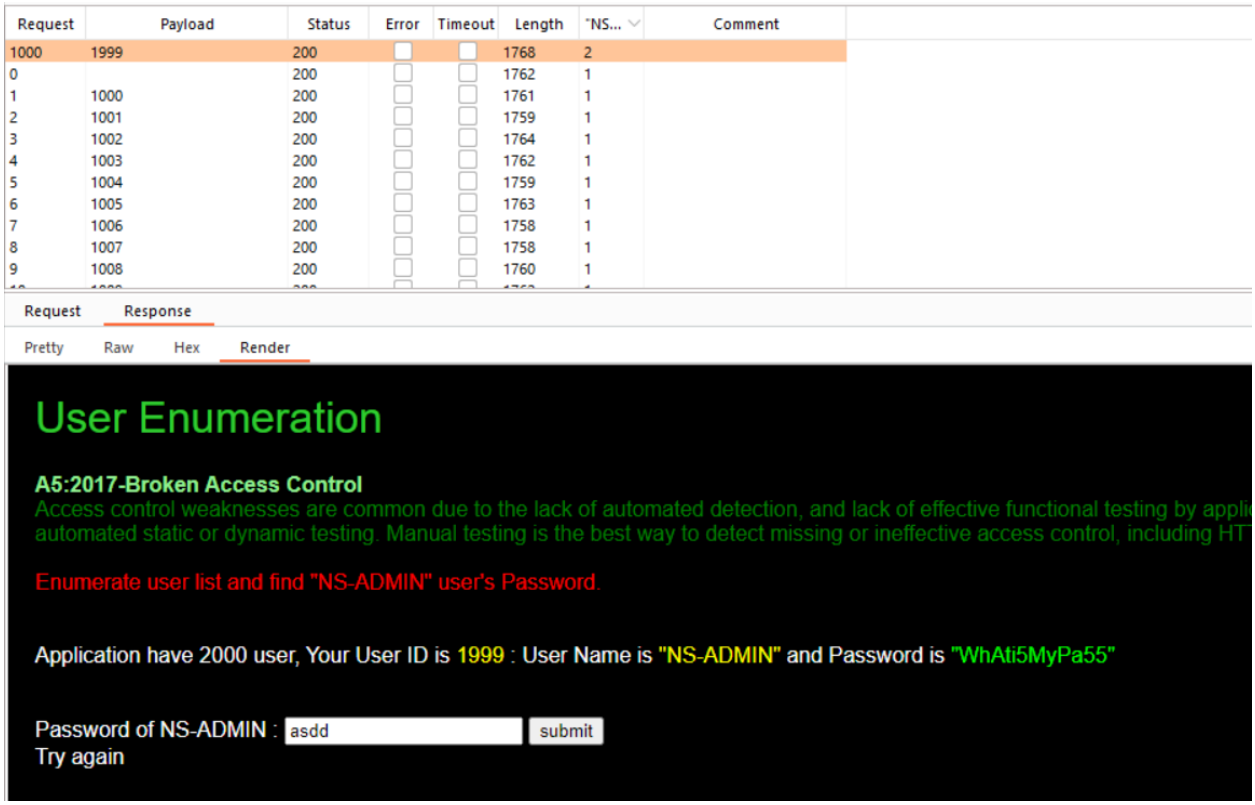
Pretty | Raw | Hex

```
GET /web/c6.ns?my_name=1234&my_pass1=rcfvgh&my_pass2=rxdtcfvgb&my_submit=Submit HTTP/1.1
Host: enigma.ctf.ns.exploitlab.net
Upgrade-Insecure-Requests: 1
User-Agent: Hacker/1.0
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=0.9
Referer: http://enigma.ctf.ns.exploitlab.net/web/c6.ns
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: PHPSESSID=14p6bf04v8anu8jk215pd39gr7; __utmc=3bab5c84e0b9c3018301b37a0a31363c; __utmp=3bab5c84e0b9c3018301b37a0a31363c; __utme
=3bab5c84e0b9c3018301b37a0a31363c; __utma=3bab5c84e0b9c3018301b37a0a31363c
Connection: close
```

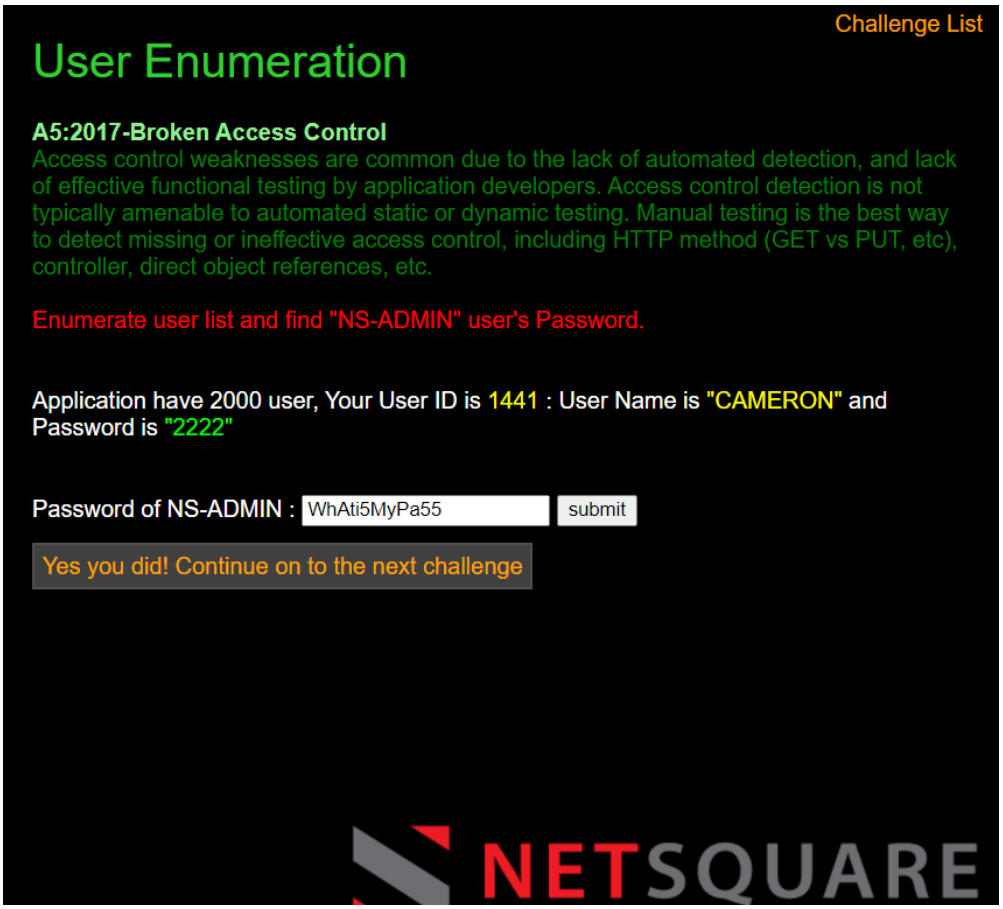



7. User ID enumeration

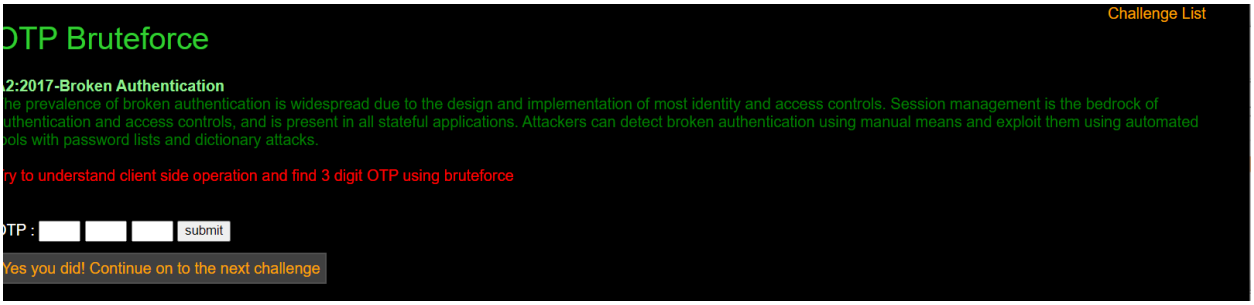
Used burpsuit intruder and bruteforced UID field



| Request | Payload | Status | Error | Timeout | Length | *NS... | Comment |
|---|---|---|---|---|---|---|---|
| 1000 | 1999 | 200 | | | 1768 | 2 | |
| 0 | | 200 | | | 1762 | 1 | |
| 1 | 1000 | 200 | | | 1761 | 1 | |
| 2 | 1001 | 200 | | | 1759 | 1 | |
| 3 | 1002 | 200 | | | 1764 | 1 | |
| 4 | 1003 | 200 | | | 1762 | 1 | |
| 5 | 1004 | 200 | | | 1759 | 1 | |
| 6 | 1005 | 200 | | | 1763 | 1 | |
| 7 | 1006 | 200 | | | 1758 | 1 | |
| 8 | 1007 | 200 | | | 1758 | 1 | |
| 9 | 1008 | 200 | | | 1760 | 1 | |




8. 2FA bypass




9.HTML5 Storage

Elements Console Sources Network Performance Application

Application
- Manifest
- Service Workers
- Storage

Storage
- Local Storage
  - http://enigma.ctf.ns.exploitl
- Session Storage
- IndexedDB
- Web SQL
- Cookies
  - http://enigma.ctf.ns.exploitl
- Trust Tokens
- Interest Groups

Cache
- Cache Storage
- Back/forward cache

Background Services

Filter

| Key | Value |
|---|---|
| __utma | 3bab5c84e0b9c3018301b37a0a31363c |
| NS_KEY | Y93xibcFwYWHXr7N2J5Atxo4iEWiU35n |

1 Y93xibcFwYWHXr7N2J5Atxo4iEWiU35n

Console Issues

Elements Console Sources Network Performance Application

Application
- Manifest
- Service Workers
- Storage

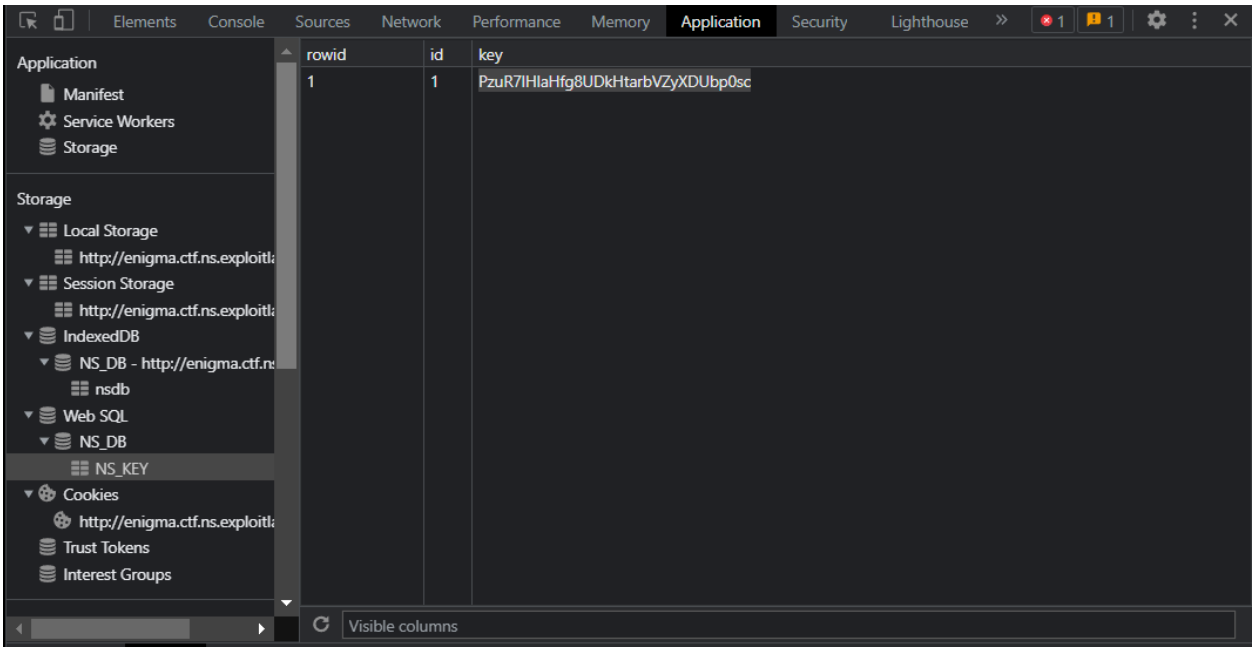
Storage
- Local Storage
  - http://enigma.ctf.ns.exploitl
- Session Storage
  - http://enigma.ctf.ns.exploitl
- IndexedDB
- Web SQL
- Cookies
  - http://enigma.ctf.ns.exploitl
- Trust Tokens
- Interest Groups

Cache
- Cache Storage
- Back/forward cache

Filter

| Key | Value |
|---|---|
| __utma | 3bab5c84e0b9c3018301b37a0a31363c |
| NS_KEY | tvBpfjMLXFFUnMToKxD4BAhdG0tSrFVN |

1 tvBpfjMLXFFUnMToKxD4BAhdG0tSrFVN


Elements
Console
Sources
Network
Performance
Memory
Application
Security
Lighthouse
Application
Manifest
Service Workers
Storage
Storage
Local Storage
http://enigma.ctf.ns.exploitl
Session Storage
http://enigma.ctf.ns.exploitl
IndexedDB
NS_DB - http://enigma.ctf.n
nsdb
Web SQL
NS_DB
NS_KEY
Cookies
http://enigma.ctf.ns.exploitl
Trust Tokens
Interest Groups
rowid
id
key
1
1
PzuR7lHlaHfg8UDkHtarbVZyXDUbp0sc
Visible columns

Challenge List

# HTML5 Storage

**A3:2017-Sensitive Data Exposure**

Over the last few years, this has been the most common impactful attack. The most common flaw is simply not encrypting sensitive data. When crypto is employed, weak key generation and management, and weak algorithm, protocol and cipher usage is common, particularly for weak password hashing storage techniques. For data in transit, server-side weaknesses are mainly easy to detect, but hard for data at rest.

Retrieve the HTML5 Storage/Communication and get the keys, which is stored in client side.

| | |
|---|---|
| Local Storage NS_KEY: | Y93xibcFwYWHXr7N2J5Atxo4iEWiU35n |
| Session Storage NS_KEY: | tvBpfjMLXFFUnMToKxD4BAhdG0tSrFVN |
| IndexedDB NS_KEY: | 7GhHBQpSDdyxp5XRASosupITGsuyiHT |
| Web SQL NS_KEY: | PzuR7IHlaHfg8UDkHtarbVZyXDUbp0sc |

submit

| Name | Value | Result |
|---|---|---|
| Local Storage NS_KEY | Y93xibcFwYWHXr7N2J5Atxo4iEWiU35n | passed |
| Session Storage NS_KEY | tvBpfjMLXFFUnMToKxD4BAhdG0tSrFVN | passed |
| IndexedDB NS_KEY | 7GhHBQpSDdyxp5XRASosupITGsuyiHTL | passed |
| Web SQL NS_KEY | PzuR7IHlaHfg8UDkHtarbVZyXDUbp0sc | passed |

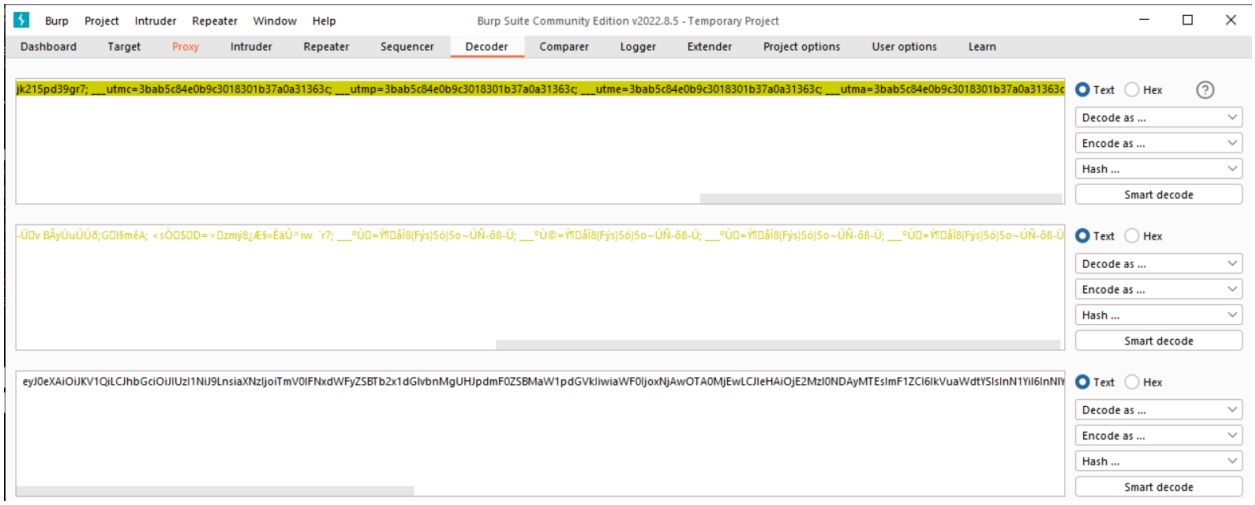
Yes you did!

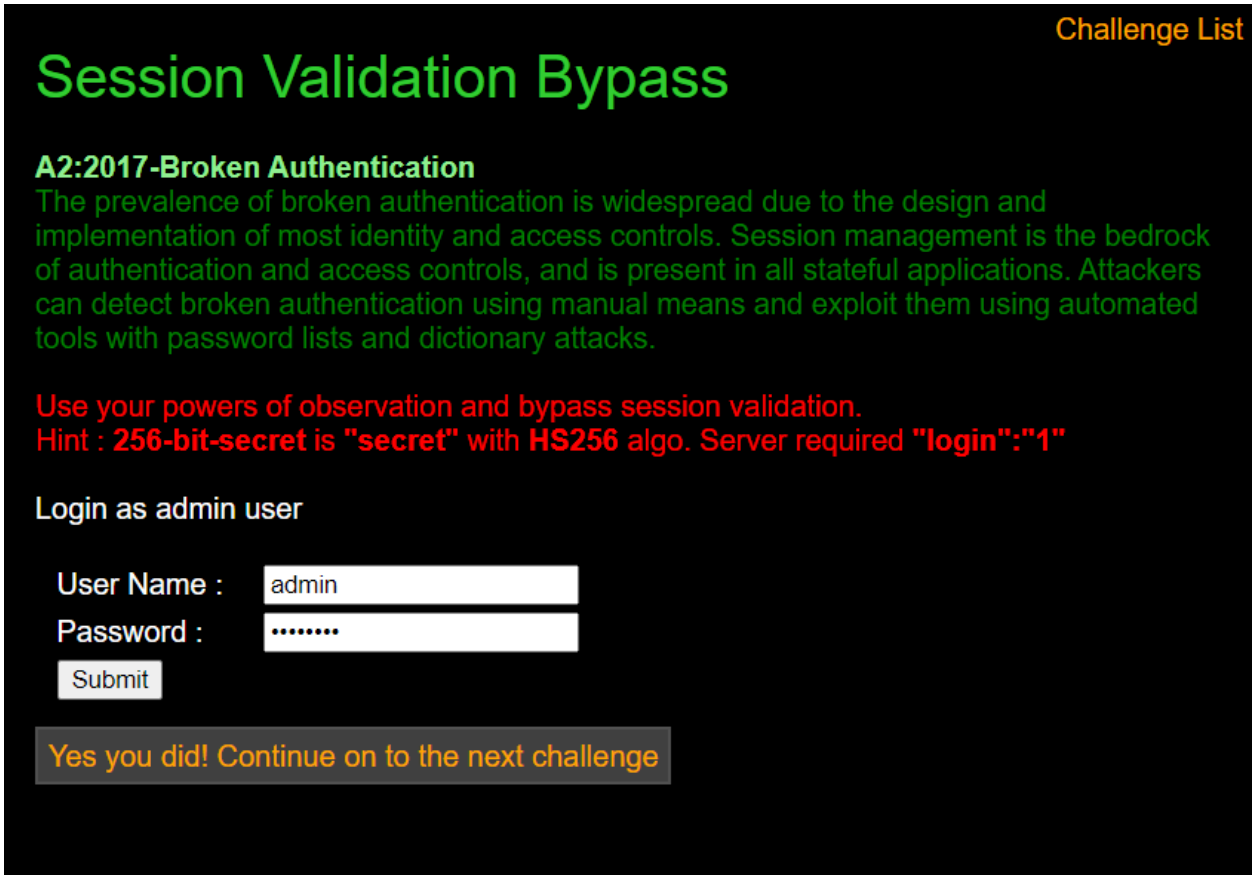
NETSQUARE

secure • innovate • automate

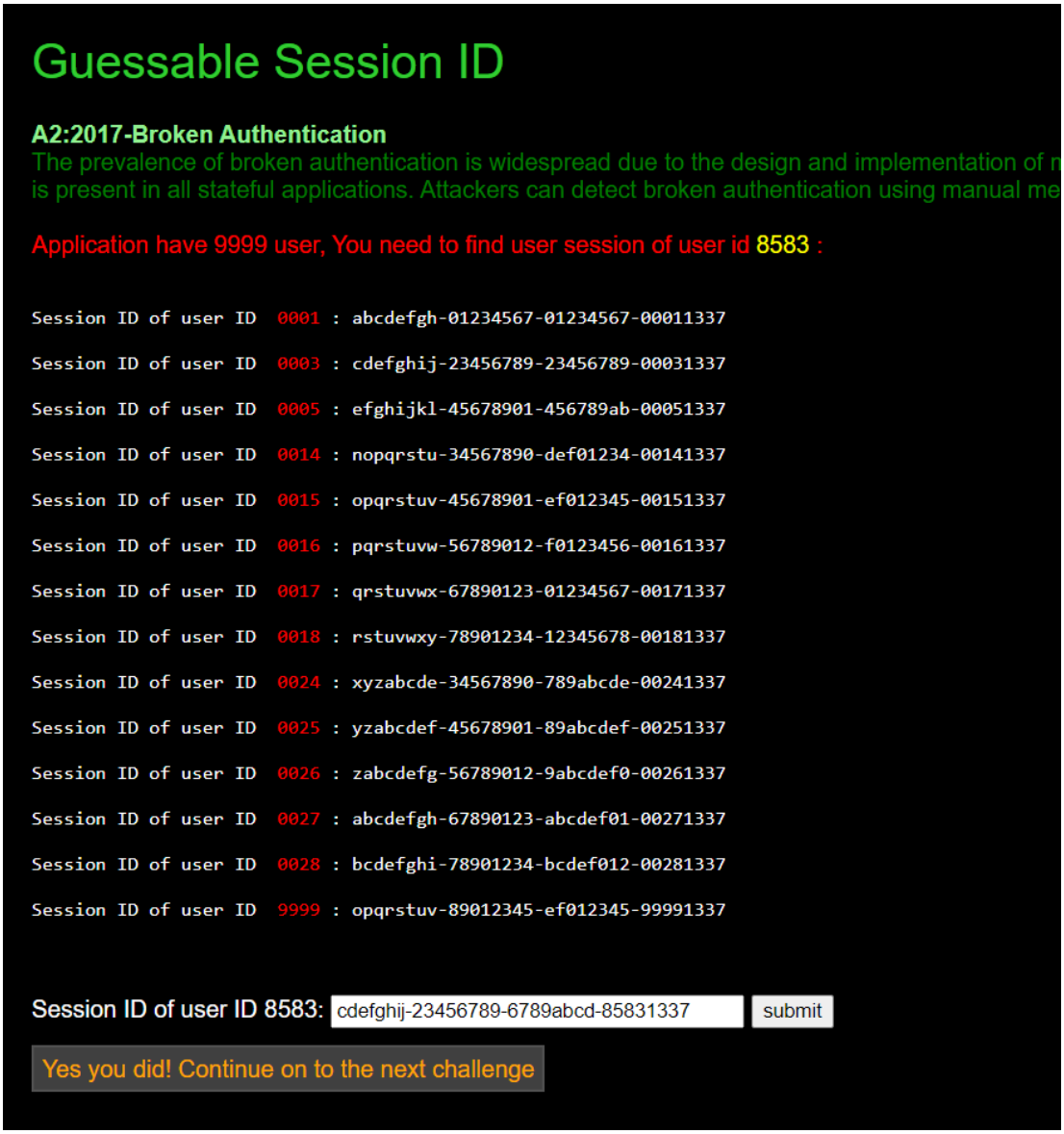
10. Session Validation Bypass

Decoded the jwt token



13. Weak Session ID




14. Insecure Direct Object Reference

```
1860                                                                                                c14.ns:31
1861                                                                                                c14.ns:31
1862                                                                                                c14.ns:31
1863                                                                                                c14.ns:31
1864                                                                                                c14.ns:31
1865                                                                                                c14.ns:31
1866                                                                                                c14.ns:31
1867                                                                                                c14.ns:31
1868                                                                                                c14.ns:31
1869                                                                                                c14.ns:31
1870                                                                                                c14.ns:31
1871                                                                                                c14.ns:31
1872                                                                                                c14.ns:31
1873                                                                                                c14.ns:31
1874                                                                                                c14.ns:31
1875                                                                                                c14.ns:31
1876                                                                                                c14.ns:31
1877                                                                                                c14.ns:31
1878                                                                                                c14.ns:31
1879                                                                                                c14.ns:31
1880                                                                                                c14.ns:31
1881                                                                                                c14.ns:31
<table border='1'><tr><td>User ID : 1866 </td></tr><tr><td>User Name : JOSEF</td></tr><tr>          c14.ns:42
<td>User Password: <input type=password value='Pussy1'></td></tr></table>
1882                                                                                                c14.ns:31
1883                                                                                                c14.ns:31
1884                                                                                                c14.ns:31
1885                                                                                                c14.ns:31
1886                                                                                                c14.ns:31
1887                                                                                                c14.ns:31
1888                                                                                                c14.ns:31
1889                                                                                                c14.ns:31
1890                                                                                                c14.ns:31
1891                                                                                                c14.ns:31
1892                                                                                                c14.ns:31
1893                                                                                                c14.ns:31
1894                                                                                                c14.ns:31
1895                                                                                                c14.ns:31
1896                                                                                                c14.ns:31
1897                                                                                                c14.ns:31
1898                                                                                                c14.ns:31
1899                                                                                                c14.ns:31
1900                                                                                                c14.ns:31
```

Challenge List

# Insecure Direct Object Reference in User Information.

**A5:2017-Broken Access Control**
**WSTG-ATHZ-04 Testing for Insecure Direct Object References**

Insecure Direct Object References occur when an application provides direct access to objects based on user-supplied input. As a result of this vulnerability attackers can bypass authorization and access resources in the system directly, for example database records or files. Insecure Direct Object References allow attackers to bypass authorization and access resources directly by modifying the value of a parameter used to directly point to an object. Such resources can be database entries belonging to other users, files in the system, and more. This is caused by the fact that the application takes user supplied input and uses it to retrieve an object without performing sufficient authorization checks.

Click to get user information. Application have 1999 user, Click to following button to get your user information, your task to find to get **JOSEF** user's Password.

Get current user information

---

User ID of **JOSEF** : 1866

Password of **JOSEF** : Pussy1

submit

Yes you did! Continue on to the next challenge

15. HTTP method

Request to http://enigma.ctf.ns.exploitlab.net:80 [162.252.242.82]

Forward | Drop | Intercept is on | Action | Open Browser

Pretty | Raw | Hex

```
1  DELETE /web/api.ns HTTP/1.1
2  Host: enigma.ctf.ns.exploitlab.net
3  Content-Length: 42
4  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/106.0.0.0 Safari/537.36
5  Content-Type: application/json
6  Accept: */*
7  Origin: http://enigma.ctf.ns.exploitlab.net
8  Referer: http://enigma.ctf.ns.exploitlab.net/web/cl5.ns
9  Accept-Encoding: gzip, deflate
10 Accept-Language: en-US,en;q=0.9
11 Cookie: jwt_access=
   eyJ0eXAi0iJKV1QiLCJhbGci0iJIUzI1NiJ9.eyJpc3Mi0iJ0ZXQgU3F1YXJ1IFNvbHV0aW9ucyBQcml2YXR1IExpbW10ZWQiLCJpYXQi0jE2MDA5MDQyMTAsIm
   MDIxMSwiYXVkIjoiRW5pZ2lhIiwic3ViIjoiMjU2LWJpdC1zZWNyZXQgaXMgc2VjcmV0IHdpdGggSFMyNTYiLCJlc2VyIjoiYWRtaW4iLCJsb2dpbiI6IjAifQ.
   LdybdqBCw3ncddra8DtHhmynbQDqA; PHPSESSID=14p6bf04v8anu8jk215pd39gr7; ___utma=3bab5c84e0b9c3018301b37a0a31363c; ___utmc=
   3bab5c84e0b9c3018301b37a0a31363c; ___utmp=3bab5c84e0b9c3018301b37a0a31363c; ___utme=3bab5c84e0b9c3018301b37a0a31363c
12 Connection: close
13
14 {
     "uid":"0",
     "uname":"admin",
     "empid":"1"
   }
```

Challenge List

# REST API HTTP Methods

**A5:2017-Broken Access Control**

Access control weaknesses are common due to the lack of automated detection, and lack of effective functional testing by application developers. Access control detection is not typically amenable to automated static or dynamic testing. Manual testing is the best way to detect missing or ineffective access control, including HTTP method (GET vs PUT, etc), controller, direct object references, etc.

REST APIs enable you to develop any kind of web application having all possible CRUD (create, retrieve, update, delete) operations.
We have provide insert data forms to the user, but some how attacker is able to delete another user data. your task is to **delete admin user** with **emp id is 1 and uid is 0**.

**Insert Your Data.**

Enter Name : admin
Enter EID : 1
submit

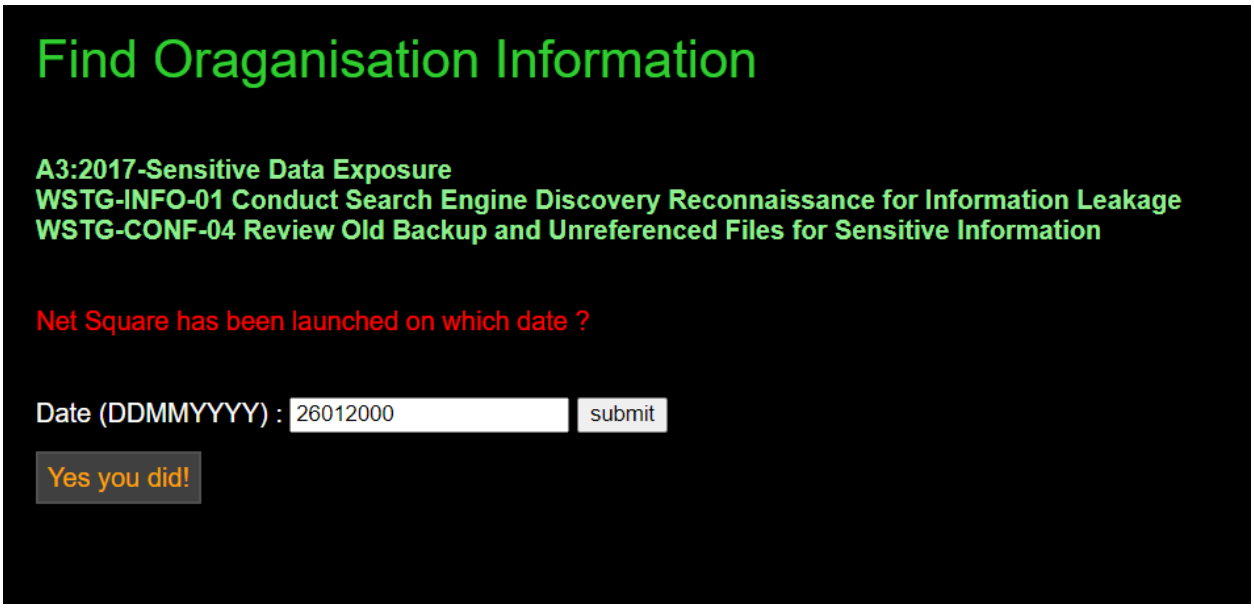
Yes you did!

**Update Your Data.**

Enter Name :
Enter EID :
submit

NETSQUARE
secure • innovate • automate

26. Passive Informtion Gathering




17. SSRF

Challenge List

# Finance Management System

**A6:2017-Security Misconfiguration**
Security misconfiguration can happen at any level of an application stack, including the network services, platform, web server, application server, database, frameworks, custom code, and pre-installed virtual machines, containers, or storage. Automated scanners are useful for detecting misconfigurations, use of default accounts or configurations, unnecessary services, legacy options, etc.

We are using multiple internal servers to comunicate internally application service.
*our internal IP range is **192.168.0.1/22** and internal service port number is **8080** which contains **secret.txt** file.*

Connect to Server

Server URL: http://www.example.com/

Submit

Token: NS-6DC792B2900F5312A76B

Submit Secret

Secret Token: NS-XXXXXXXXXXXXXXXXXXXX

Submit

Challenge List

# Finance Management System

**A6:2017-Security Misconfiguration**
Security misconfiguration can happen at any level of an application stack, including the network services, platform, web server, application server, database, frameworks, custom code, and pre-installed virtual machines, containers, or storage. Automated scanners are useful for detecting misconfigurations, use of default accounts or configurations, unnecessary services, legacy options, etc.

We are using multiple internal servers to comunicate internally application service.
*our internal IP range is **192.168.0.1/22** and internal service port number is **8080** which contains **secret.txt** file.*

Connect to Server

Server URL: http://www.example.com/

Submit

Submit Secret

Secret Token: NS-XXXXXXXXXXXXXXXXXXXX

Submit

Yes you did!

40. IP white listing bypass

# Code Review : IP whitelisting Bypass

**A2:2017-Broken Authentication**

The prevalence of broken authentication is widespread due to the design and implementation of most identity and access controls.

Understand the code and bypass the local IP verification to complete the task.

Code snippet

```
function getIpAddress()
{
        return isset($_SERVER["HTTP_X_FORWARDED_FOR"]) ? $_SERVER["HTTP_X_FORWARDED_FOR"] : $_SERVER["REMOTE_ADDR"] ;
}

function isIpLocal()
{
        if(substr(getIpAddress(),0,4) == "127.")
        {
                return true;
        }
}

if(isset($_POST['value']))
{
        if(isIpLocal())
        {
                echo "Yes you did!";
        }
        else
        {
                echo "Try again. Your IP address is " . getIpAddress() ;
        }
}
```

Click on following button to send request to the server.

NETSQUARE
secure • innovate • automate

103.153.104.155 submit

Yes you did! Continue on to the next challenge

Code:


```
gdnayoMs/abbBdZw0qdecsLJDIvMaj9O5/5S56DNP/scZnY9OO/yDeroehh5sHbpUOYSYj7EE
NECA5vtPV4iTGsDXXO8sGc0nM9EZQiduW6pvCtXjMCOyxqAHBpxr18MxWgJoIWpPCloyKs1vl
IhbQdPHu1xntvi2qura7fj+shWmICGwtgg+UJRXmb3Mq1cnmg9qSbG9mEfpuEqK7drkiQ40ES
5uLnQUYNOMzOK9sV0ya3hpABq/sRLiz8pJ1cgL6ayZe6mR/U3ZAn1dNrx6C/B5tesKK2h/lbG
D7bQTbGJoRp/6nScZhMU6RImmhifzYk7NRLO2nzqrWSIIX1pI7XPzTXV5vTmt9f8y49/LFaZz
KGvUVBwUWcQe4/g795sEvDI5ePcCoEkWk47Lk+wbf/ZWPNw8Iu7mFx6jVA8vQhHrSchhqXAgU
WSDzk322bz/Ce6aQ4tBpGgeH9Ak3BwmQwmYAcEEmQn1MTClzgpkHnaB5eaWIEK2eiDeeasq9z
LUG4MyBMYQhZI6cGd7wa0lgZgGKqNWyD4WjovLD/wkYmAun/mTFESMXYKXQrue3yhs8VZKBAi
XXwaDG6NzF51J6yidcR+FyJ+4Dia+iwCWZVjNmnMtCbjVHjG/DK9EsG4lDDuJRmli16PUo34m
9rKvruj4aCwe3UupfEuXCGvDQL/uwZsXWfrq+ef52qWNFj23ATDNTlcFYOhxgauqRePg/ZQHQ
H5i1pc6Bo7/m4LlazkV8MvPxkfrcnU2HFqQM3LUyBHPitpL2kjHhPrbTflVzHCFwgmqBat+34
EKoN+m3KLjYDDUFOHU5oMICoiHLuosN7/fgajzOdtqYB1W2qJgR9nG+u+jRcmVJHE/QxRWI5C
8Ac43SN24Ynrle2uA4c4PX0xrX5fmCdviLqBwRFPpg15SmuElvq6h9iHZV5AX78scQWO4gnu2
VyAX7J4nov/wqzrMUrK93Cp3KdksvjIW1kCwlLWntOk0ed8Vi9Bypy1/Utji+GLxPGwsClwkt
0v6UbeiesOiCn5F0bDIZCDICceFWSYYAyQF7OubHmszCoOlCY9gN1td4u7IpwBkIK8xYW3UXT
zNHQyjMeycmguExfnesXRgwaX47+eLlEeDQZRA8zYOiZpsSiTOtP/7Qsgmat+Xs5xhQhJmM2R
b6CW/5fM58td2G+J8qgalokg6F4DVpFsaU92qGpR+FF5Pj7wD4bRYOTVkpSkNiNfzdZnPxPIg
c25xZ/HKX/T3ffh1Zbi7zW2YEJXVXcJUoAhdXZP/m685PjgnPHMgxf+V6YUFegWNuqQ9ezNUW
hjhTRvtvFbGD8TZnNdGEdoapExbJztZDYqoDoTD8W9luUX9whixc9VLisuydf0cK1+fVfxV6R
YLtziLJmlilKo4yAv4CmNr8XWFvutQfOfADQ0GCMhmeXwb+WD3eI36mFBLjP7086MyDwN208r
hyFq+Ps88LQUu6eZLKYrjxyj3t6n42oGysLiEP2UE3X6fRLoZ1Bm4HDUMmaRxv8rumGJPXNLE
GzFMEUcR7Y0hCuMdmVuVsOPYg==
```