

# Ambassador write-up

## 1. Description

This article is the write-up for the Ambassador box of HackTheBox.

To begin, you need to be connected to the HackTheBox vpn.

## 2. Port Enumeration

First, let's make a map of the host.

What ports are open ?

What services are running ?

For this, we will use a combinaison of RustScan and Nmap.

```
rustscan IP -- -sC -sV -oN nmap_initial.txt
```

This is the result of the scan :

PORT	STATE SERVICE	REASON	VERSION
22/tcp	open	ssh	syn-ack OpenSSH 8.2p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux; protocol 2.0)
80/tcp	open	http	syn-ack Apache httpd 2.4.41 ((Ubuntu))
3000/tcp	open	ppp?	syn-ack
3306/tcp	open	nagios- nsca	syn-ack Nagios NSCA

On the port 22, an SSH server is running, but at this point, we don't have any credentials.

On the port 80, an Apache web server is running, it must be interesting to start here.

On the port 3000, the scan don't tell us much about it, but if you try to connect to <http://IP:3000>, we are greeted by a **Grafana** login page.

On the port 3306, the scan output tell us that it is a Nagis NSCA service, but 3306 is a common port for a mysql server, and trying to connect to the server using :

```
mysql IP -u root -p
```

indicate that a mysql server is running and accessible.

### 3. Web exploration

---

Let's begin with the web server running on port 80.

There is only one thing on this webserver, a single post in which we can have a usefull information :

*Use the **developer** account to SSH, DevOps will give you the password.*

We now have a username ! but not a password.. so let's continue to explore others services since running a directory scanner will not find us anything new.

### 4. Grafana

---

Grafana is a web application which allow us to monitor and analyse data from differents sources like databases.

At the bottom of the login page, we can discover the version of the application :

**v8.2.0 (d7f71e9eae)**

It is always good to search online for public exploits of popular applications on site like google, or [exploit-db](#)

The first links tell us about a **directory traversal** that allow us to read sensitive file on the server

This endpoint is vulnerable : <http://IP/public/plugins/../../../../../../../../fileToRead>  
where pluginName is a plugin of Grafana like :

- alertlist
- annolist

You can view a more complete list [here](#)

So, now, we can read sensitive data with :

```
curl --path-as-is http://IP:3000/public/plugins/alertlist/../../../../../../../../fi
```

**--path-as-is** is needed, because other wise, curl will normalise the url as : <http://IP:3000/public/plugins/alertlist/filename> and this will not work.

We need to find usefull informations by reading important files like **/etc/passwd**:

```
curl --path-as-is http://10.10.11.183:3000/public/plugins/alertlist/../../../../../../../../etc/passwd
```

Which leak one user and confirm the one discovered previously :

```
developer:x:1000:1000:developer:/home/developer:/bin/bash
```

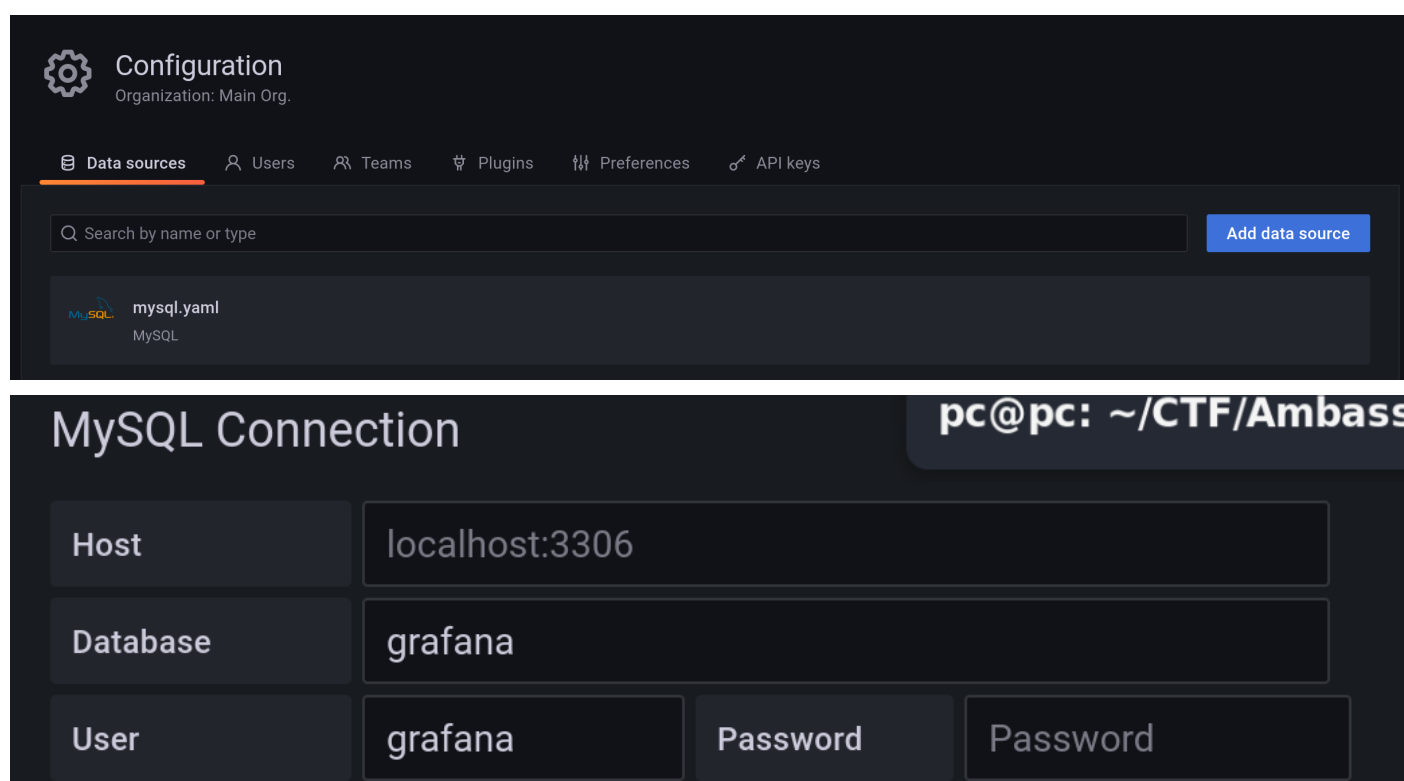
```
consul:x:997:997::/home/consul:/bin/false
```

Grafana store credentials in configurations like grafana.ini or defaults.ini which can be found at :  
 /etc/grafana/grafana.ini `curl --path-as-is http://10.10.11.183:3000/public/plugins/alertlist/../../../../../../../../etc/grafana/grafana.ini --output grafana.ini`

in which we can retrieve the admin password of Grafana : [REDACTED]

We can also dump the whole Grafana sqlite db at /var/lib/grafana/grafana.db `curl --path-as-is http://10.10.11.183:3000/public/plugins/alertlist/../../../../../../../../var/lib/grafana/grafana.db --output grafana.db`

After login on the Grafana webapp, we can see in datasources that a mysql server is connected on a **grafana** with the **grafana** user, but the password remain empty.



The screenshot shows the Grafana Configuration page for the 'Main Org.' with the 'Data sources' tab selected. A search bar is present with a blue 'Add data source' button. Below the search bar, a 'mysql.yaml' data source is listed. The 'MySQL Connection' configuration form is displayed with the following fields:

- Host: localhost:3306
- Database: grafana
- User: grafana
- Password: (empty)

The top right of the configuration form shows the user 'pc@pc' and the path '~ /CTF/Ambass'.

This is because Grafana don't show the password in the webapp, but the password is somewhere, in the **grafana database** that we retrieve earlier.

We can explore the retrieved database using **DB Browser for Sqlite**

We know that we need to find the **datasource**, and there is a datasource table in the database

id	org_id	version	type	name	access	url	password	user	database
	Fil...	Filtre	Fil...	Filtre	Filtre		Filtre	Filtre	Filtre
1	2	1	mysql	mysql.yaml	proxy		dontStandSoCloseT...	grafana	grafana

[Here](#) is an automated script that allow to retrieve usefull files from Grafana, search for Grafana tokens and decrypt Grafana passwords.

## 5. Mysql

With the credentials we found, we can connect to the mysql database.

```
mysql -u grafana -p -h IP
```

We can find an odd databased called **whackywidget** This database contain only one table **users**

user	pass
developer	[REDACTED]

We have found the password of the developer account ! It is in base64 encoding, we need to decode it and then we can connect with ssh to the server.

## 6. Privilege Escalation

On the server, we can directly find the **user.txt** on the developer home directory.

When listing all files in the home directory, **including hidden file**, one particular file is present **.gitconfig**

This file leak us an important information : **directory = /opt/my-app**

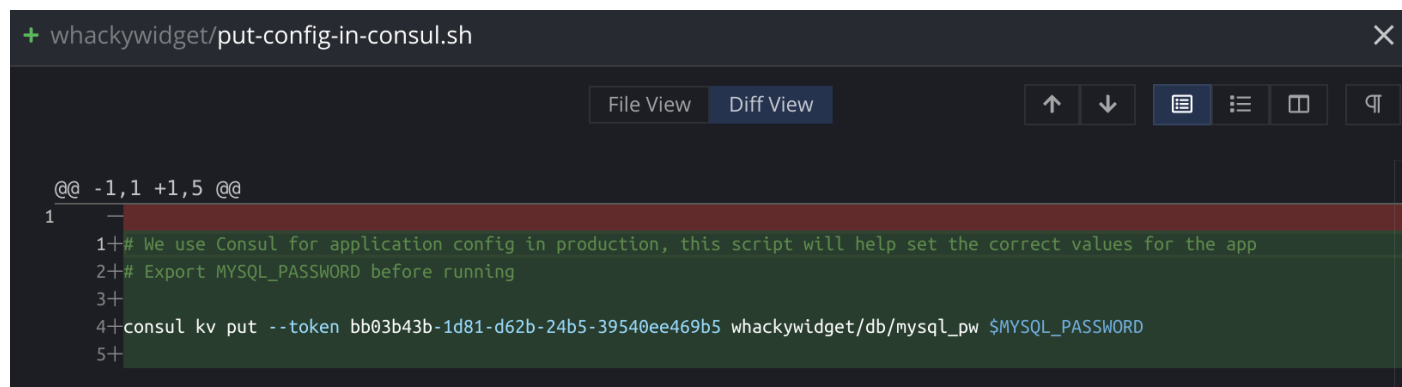
In this directory, we can find an application named **whackywidget** and a **.git** directory, which contain **all changes made to the application**

Using a program like [GitKraken](#), we can see the changes made to the application.

For this, you need to transfer the **.git** directory to your attacker machine, and open it with GitKraken

the actual version of the script **put-config-in-consul.sh** can't be run because it need a token that we don't have.

But reviewing ancient version of the script using GitKraken, we can find a token that was used before.



```
+ whackywidget/put-config-in-consul.sh

File View  Diff View

@@ -1,1 +1,5 @@
1-
1+# We use Consul for application config in production, this script will help set the correct values for the app
2+# Export MYSQL_PASSWORD before running
3+
4+consul kv put --token bb03b43b-1d81-d62b-24b5-39540ee469b5 whackywidget/db/mysql_pw $MYSQL_PASSWORD
5+
```

You can then use a RCE like [this](#) on to get root using consul.

Then get the root flag in the /root folder