IT Forensics Workshop 9

Topics:

- Network Forensics Tools:
  - `tcpflow`, `tshark`, and Wireshark
- Linux command line tools:
  - `grep`, `awk`, `sort`
- Network Forensics Exercises
- Password Attack Methods

Covered Learning Outcomes:

- Explain the motivations and landscape of network forensic investigations in an IT context
- Become familiar with some of the network forensic tools
- Deduce information and explain the events from captured traffic in a network forensic investigation

Instructions:

- Individual and group activities.

Files and access required:

For this week's activity you need:

- John the Ripper password recovery tool
- an account on https://www.hackthebox.eu.
- the virtual machine: lu16d-coremu-v1.3.ova (username: `muni` and password: `muni`)
- the packet capture file `nitroba.pcap` available on:

  https://digitalcorpora.org/corpora/scenarios/nitroba- university-harassment-scenario
  - You may get a warning when using Google Chrome to download from this website which may be due to the fact that some of the content is related to malware forensics, you can select Keep rather than Discard after the file is downloaded.

Activity A: John the Ripper

Installing John the Ripper Jumbo

Download the john-1.9.0-jumbo-1.tar.xz. Extract the source using the following command:

```
tar -xvf john-1.9.0-jumbo-1.tar.xz
```

Change the current directory to john-1.9.0-jumbo-1.tar.xz/src, configure, and make the tool using the following commands:

```
cd john-1.9.0-jumbo-1.tar.xz/src ./configure
make -s clean && make -sj4
```

Writing Basic Rules for Wordlist Password Recovery

Change the directory to `../run`. The `john.conf` contains the configuration of the JtR tool. Open `john.conf` in a text editor. Go to the end of the file and add the following lines before the last three commented lines that indicate the end of the configuration:

```
[List.Rules:Forensics]
.include <rules/forensics.rule>
```

Open the page JtR Wordlist Rules Syntax and have a look at the wordlist rule syntax. Open a text editor to create a new file named test-pass.lst which we will use to test out how the wordlist rules generate password combinations using a wordlist file. Enter the following passwords in the test-pass.lst file.

```
pass
123
Password
forensicS
pass123
123pass
1234
12345
123456
1234567
cat
dog
case
home
clear
switch
```

Open (create) the file rules/forensics.rule and complete the following tasks. For all the activities use the sample password list file created above. After adding/editing each rule run the following command to see the effect.

```
./john -w=pass-test.lst --stdout --rule=Forensics
```

1. **Write a rule that only accepts five or more character passwords from the wordlist file. Check "Length control commands".**
2. **Write a rule that only accepts exactly five-character passwords from the wordlist file.**
3. **Write a rule that rejects passwords that contain any digits from the wordlist file. Check the "Character class commands" section.**
4. **Write a rule that produces passwords that are all lowercase, are four or more characters long, and do not contain any digits from the wordlist file.**
5. **Now edit the rule from the previous step to append two digits to the end of each word to generate a password. Check the "Simple commands" section.**
6. **Use the following command to generate the MD5 hash of the password `clear123` and store it in a file. Then edit the rule in the previous step and the sample wordlist file to recover the password from the hash.**

```
echo -n clear123 | md5sum | awk '{ print $1 }' > hash.txt
```

Activity B

Scenario[1]

A teacher in the Chemistry Department of the (fictional) Nitroba State University has made a complaint about receiving harassing emails. Lily Tuckridge, the teacher, suspects that these emails are being sent by a student in her Chemistry 109 class. The emails are sent to her personal email "`lilytuckridge@yahoo.com`". She has provided a screenshot of the email in a web browser shown in Figure 1.

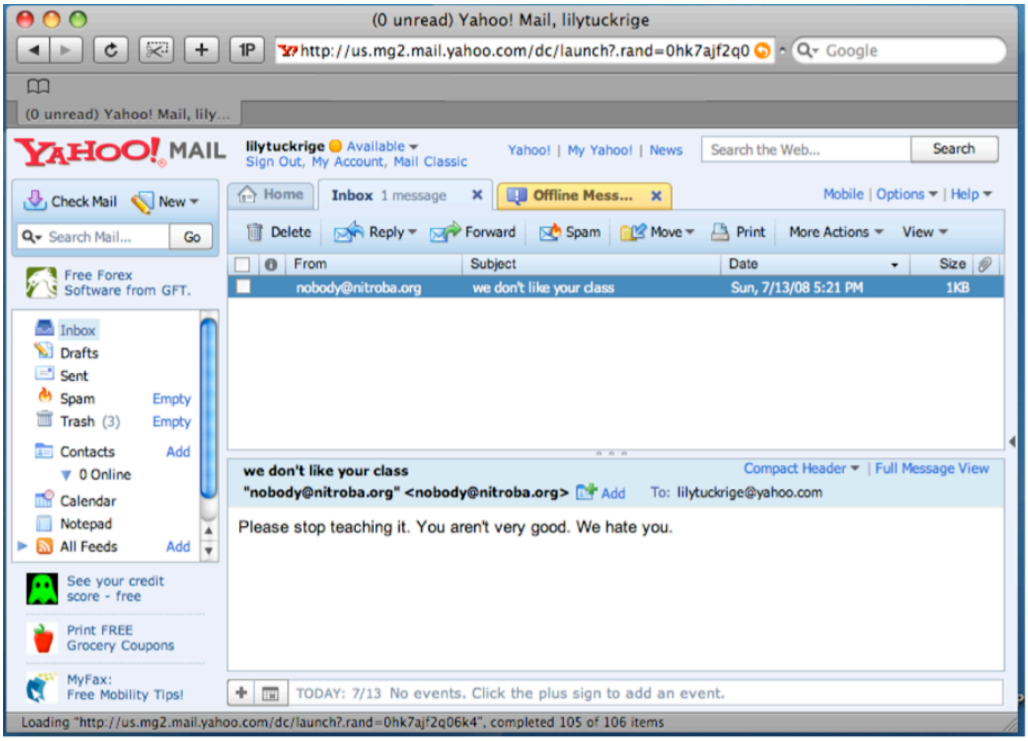

Figure 1: Email received by Lily Tuckridge

Upon request she has also provided the headers of the email shown in Figure 2.

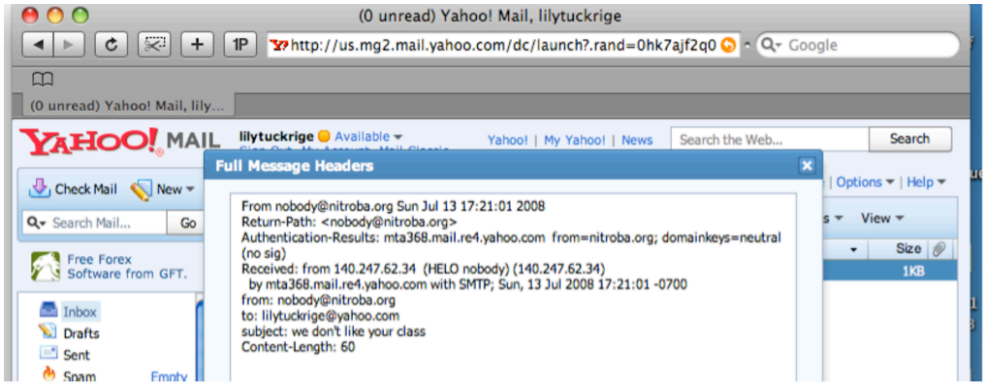

Figure 2: Email headers

1. **What is the IP address of the origin of the email?**

[1]Retrieved from https://digitalcorpora.org/corpora/scenarios/nitroba-university-harassment-scenario

The IT department identifies the IP address as one of the addresses used for students' dorm rooms. The dorm room is shared by three students. The university provides 10Mbps Ethernet access for students and no wireless access. One of the students has installed a wireless router with no password. The IT department sets up a packet sniffer on the Ethernet port as shown in Figure 3.

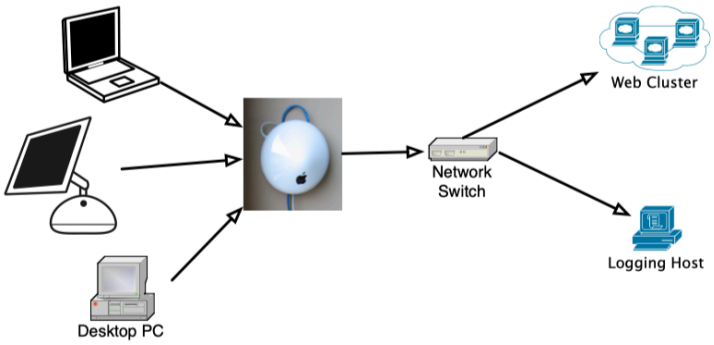

Figure 3: Packet sniffer setup on Ethernet switch

Lily receives another email however this time the email is sent using an online self-destructing message board shown in Figure 4 and Figure 5.

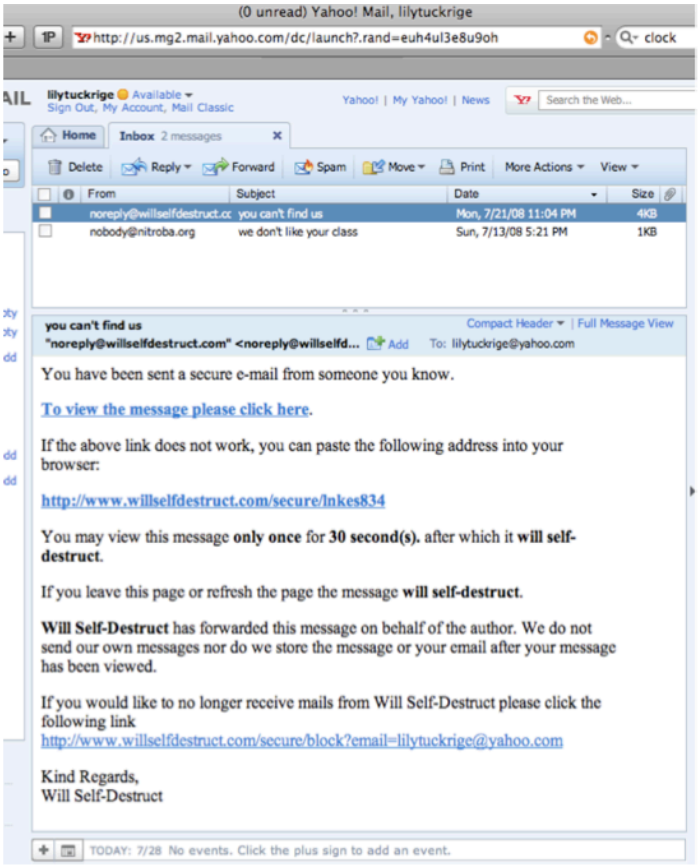

Figure 4: Self-destruct email

Figure 5: Self-destruct email

The list of students in Chemistry 109 taught by Lily is as follows:

- Amy Smith
- Burt Greedom
- Tuck Gorge
- Ava Book
- Johnny Coach
- Jeremy Ledvkin
- Nancy Colburne
- Tamara Perkins
- Esther Pringle
- Asar Misrad
- Jenny Kant

2. **Use the tool `tcpflow` (`sudo apt install tcpflow` to install and `man tcpflow` for more information) to store each TCP flow in the pcap file and inspect the files to identify the hostile message.**
   a) **How many flows are identified?**
   b) **How are the filenames chosen? What information is used in the filenames?**
   c) **Use `grep` to search and identify the flow that contains the hostile message.**
   d) **Use information gathered from previous tasks to identify the user who has sent the emails. Remember that the IP address is shared by multiple users (Hint: How many different browsers have used the shared IP address?). Useful tools besides `grep`:`awk` and `sort`.**

Activity C: Hack the Box

Log in to your account on https://www.hackthebox.eu inside the VM and navigate to `Labs → Challenges → Forensics` and open Obscure challenge.

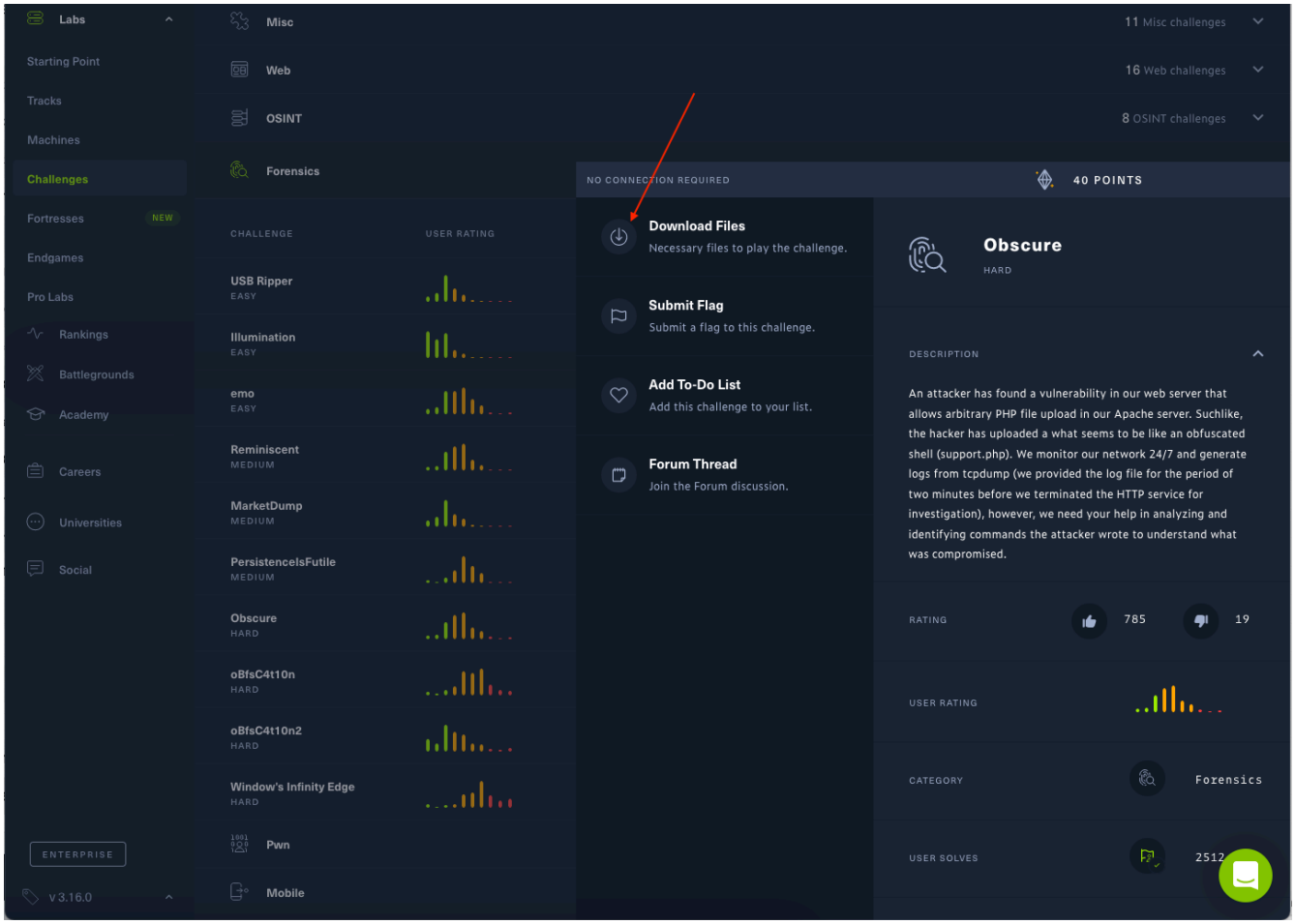

Figure 6: Log in to `hackthebox` and navigate to Labs → Challenges → Forensics → Obscure

Read the description of the challenge and then download the zip file and then extract. The password for the zip file is `hackthebox`. You would find two files extracted: `support.php` and `190521_22532255.pcap`. Open the `pcap` file in Wireshark and `support.php` in a text editor (such as LeafPad, or from terminal `nano` or `vi`).

Tasks

1. **Is the content of `support.php` readable? Find out how to decode the content to PHP.**
2. **Analyse the decoded `support.php` file and explain what it does?**
3. **From what you have learned from the previous step, analyse the pcap file using wireshark and decode all of the attackers commands and the responses. You can also use `tshark` to automate the extraction of the artefacts. You also need to write some code to decode the extracted commands. Work in groups and ask for clues from your tutors if you get stuck.**

Appendix: Activity B

Some explanation related to command:

For searching based on a parameter. For example, `grep -e "key value"` filename (for example: `grep -e "1556" list1.txt`) [It will search for the pattern `"1556"` in the file `list1.txt` and output all the lines that contain this pattern.]

Another way to do the search over all files listed under ls

```
grep -e "1556" $(ls)
```

Now if we want to filter one item from the search result, we have to combine the two commands. (grep and awk)

If we are interested about first item before ":" from search result, then command

```
grep -e "1556" $(ls) | awk -F : '{ print $1}'
```

[prints the first field (the part before the colon) from each line. In this case, it will print the **filenames** (since grep output typically includes filename:line format).]

Activity B: Part 2.

1. Turn on the virtual machine, download "Nitroba" from Moodle. Open LXTerminal. Change your current location to the directory of nitroba and run the following `tcpflow` command.

   Use the tool `tcpflow` (`sudo apt install tcpflow` to install and `man tcpflow` for more information) to store each TCP flow in the pcap file and inspect the files to identify the hostile message.

   Command: `tcpflow -r nitroba.pcap -o nitroba`

   Go to the directory where you downloaded/put Nitroba.pcap. You will see the nitroba directory. Inspect it and answer following questions

   a) How many flows are identified?

   b) How the filenames are chosen or what information is used in the filenames?

   c) Use grep to search and identify the flow that contains the hostile message. (do `man grep`)

   We can look for all flows that used the web site `willselfdestruct.com`

Change you current directory to nitroba (cd nitroba) and run following command

Command:

```
grep -e 'willselfdestruct' $(ls) | awk -F: '{ print $1 }' > ../interesting_flows.txt
```

Inspect `interesting_flows.txt`.

This identifies 108 flows, all from and to `192.168.015.004`.

We can search for the hostile content within the identified flows. Bear in mind that space may have been converted to +.

We will search/see the "`teaching`" keyword from the `interesting_flows.txt`

Command: `grep e 'teaching' $(cat ../interesting_flows.txt)`

This identifies the file `192.168.015.004.36044 - 069.025.094.022.00080`, which we can inspect from the nitroba directory to see the content.

```
POST /secure/submit HTTP/1.1
Accept: image/gif, image/xxbitmap,
image/jpeg, image/pjpeg, application/xshockwaveflash,
*/* Referer: http://www.willselfdestruct.com/secure/submit
AcceptLanguage: enus
ContentType: application/xwwwformurlencoded
AcceptEncoding: gzip, deflate
UserAgent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)
Host: www.willselfdestruct.com
ContentLength: 188
Connection: KeepAlive
CacheControl: nocache
to=lilytuckrige@yahoo.com&from=&subject=you+can%27t+find+us&message=a
nd+you+can%27t+hide+from+us.%
```

Alternatively, we could have searched for the hostile email content in all flows and identified three flows to inspect.

Command:

```
grep -e 'teaching' $(ls) | awk -F: '{ print $1 }' > ../interesting_flows.txt
```

```
018.007.022.069.00080192.168.015.004.32898
192.168.015.004.35876069.080.225.091.00080
192.168.015.004.36044069.025.094.022.00080
```

Inspect the above three files from the nitroba directory one by one.

This also identifies another email in the second flow above and the web site sendanonymousemail.net.

Giving us another keyword to search.

```
POST /send.php HTTP/1.1
Accept: image/gif, image/xxbitmap,
image/jpeg, image/pjpeg, application/xshockwaveflash,
*/*  Referer: http://www.sendanonymousemail.net/
AcceptLanguage: enus
ContentType: application/xwwwformurlencoded
AcceptEncoding: gzip, deflate
UserAgent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)
Host: www.sendanonymousemail.net
ContentLength: 275
Connection: KeepAlive
CacheControl: nocache
Cookie: PHPSESSID=762adba03236142ccec305f6a20aaffa
email=lilytuckrige@yahoo.com&sender=the_whole_world_is_watching@nitroba
.org &subject=Your+class+stinks&Accept: */*
Referer: http://www.sendanonymousemail.net/
AcceptLanguage: enus
AcceptEncoding: gzip, deflate
UserAgent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)
Host: www.sendanonymousemail.net
Connection: KeepAlive
Cookie: PHPSESSID=762adba03236142ccec305f6a20aaffa
```

d) Use information gathered from previous tasks to identify the user who has sent the emails. Remember that the IP address is shared by multiple users (Hint: How many different browsers have used the shared IP address?). Useful tools besides `grep`: `awk` and `sort`.

We can first see if the browser used in all flows from the IP address `192.168.015.004` are similar or different.

Enter the command in one line. (current directory is nitroba)

Command:

```
grep -e 'User-Agent' $(ls *192.168.015.004*) | awk -F": " '{ print $2 }' | sort --unique > ../browsers.txt
```

Inspect `browsers.txt`. We will see following information:

```
Adium/1.2.7 (Mac OS X) Sparkle/1.1
ApplePubSub/65.1.1
CFNetwork/330.4
ee://aol/http
iTunes/7.7 (Macintosh; N; Intel)
iTunes/7.7 (Macintosh; U; Intel Mac OS X 10.5.4)
Mozilla/4.0 (compatible; MSIE 5.5)
Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)
Mozilla/5.0 (Macintosh; U; Intel Mac OS X 10_5_4; enus)
AppleWebKit/525.18 (KHTML, like Gecko) Version/Mozilla/5.0
(Macintosh; U; Intel Mac OS X 10.5; enUS;
rv:1.9b5) Gecko/2008032619 Firefox/3.0b5
Mozilla/5.0 (Macintosh; U; Intel Mac OS X; enUS;
rv:1.8.1.16) Gecko/20080702 Firefox/2.0.0.16
Mozilla/5.0 (Macintosh; U; PPC Mac OS X MachO;
enUS;
rv:1.7.5)
Mozilla/5.0 (Windows; U; Windows NT 5.1; enUS;
rv:1.9.0.1) Gecko/2008070208 Firefox/3.0.1
WindowsUpdateAgent
```

Comparing the above list with the identified flows of hostile messages it seems that the attacker's browser could be used to identify the user. We now look for all flows that contain that particular User-Agent.

Command:

```
grep -e 'Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)' $(ls) | awk -F:  '{ print $1 }'  > ../all_hostile_flows.txt
```

We can then combine all the content to have a quick inspection:

Command:

```
cat $(cat ../all_hostile_flows.txt) > ../all_hostile_content.txt
```

We will get some errors. Just ignore it. Inspect `all_hostie_content.txt`.
Search/inspect the following lines "*how to annoy people*", "*sending anonymous mail*", "*i want to harass my teacher*".

We can then get all the unique cookies set/used in the identified flows.

Command:

```
grep -e 'Cookie' $(cat ../all_hostile_flows.txt) | awk -F": " '{print $2 }' | sort --unique  > ../hostile_cookies.txt
```

Inspect `hostile_flows.txt`

Now tell me the culprit name for whom you and I had to do a lot of work!!!!!! lol…..