Computer and Information Security
(ECE590-04, Fall 2018, Duke Univ., Prof. Tyler Bletsch)

Homework 4

Name: Rijish Ganguly

Duke NetID: rg239

Instructions - read all carefully:

- DON'T SCREW UP: Read each question carefully and be sure to answer all parts. Some questions are a mix of explanation and questions, so pay close attention to where you are being asked for something. It is recommended to answer the questions in the order they are asked, as they build on each other.
- COMPUTERS YOU WILL NEED:
  - The assignment will make use of the computers described below.
  - VMs you already have on the Duke VCM service:
    - The Ubuntu 18.04 VM; which we'll call your Linux VM.
    - The Windows 10 VM; which we'll call your Windows VM.
    - The Kali Linux VM; which we'll call your Kali VM.
  - A new throw-away VM with Ubuntu 18.04, which we'll call your backup server.
  - Your own machine on Duke wifi: your personal computer (any OS).
- WRITTEN PORTION DIRECTIONS:
  - This assignment is designed to be copied into a new document so you can answer questions inline (either as a Google doc or in a local word processor).
  - This assignment should be submitted as a PDF through Gradescope. Other formats or methods of submission will not be accepted.
  - When you submit, the tool will ask you to mark which pages contain which questions. This is easiest if you avoid having two questions on one page and keep the large question headers intact. Be sure to mark your answer pages appropriately.
- PROGRAMMING PORTION DIRECTIONS:
  - There is a buffer overflow programming project in this assignment; your code for this will be submitted as a separate file via the Sakai assignment facility. See the question itself for details.
- CITE YOUR SOURCES: Make sure you document any resources you may use when answering the questions, including classmates and the textbook. Please use authoritative sources like RFCs, ISOs, NIST SPs, man pages, etc. for your references.

This assignment is adapted from material by Samuel Carter (NCSU).

Question 0: Accessing the Homework (0 points, but necessary)

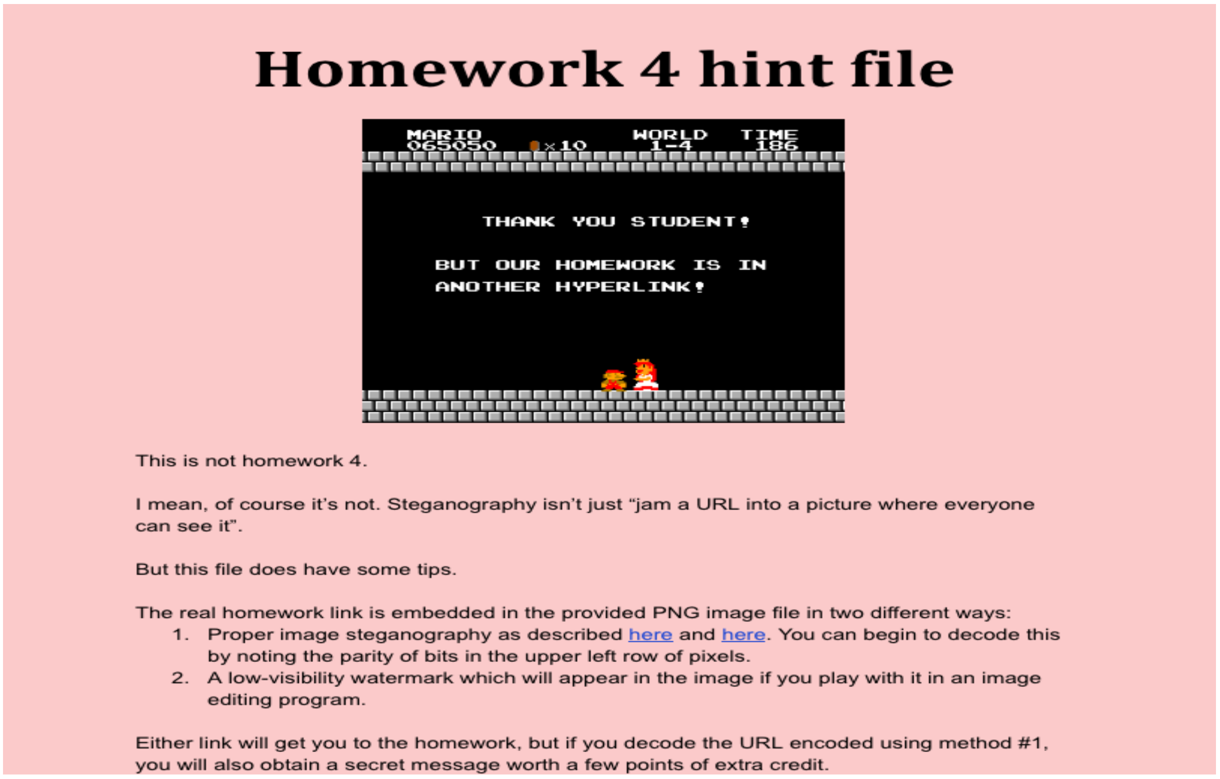
To get here, you found the URL via one of two steganography methods. Did you find this hint file along the way?

Yes, I did.

Question 1: Show your work from Question 0 (2 points)

Below, show the code and/or image work you did to get here.

At first I did cat homework4.png for the provided image which directed me to the link https://docs.google.com/document/d/16RgAXAox1entA3aORk2ADBhM_Ek60_SwGa1eqQoMmbc/edit which had the following clue.



Homework 4 hint file

This is not homework 4.

I mean, of course it's not. Steganography isn't just "jam a URL into a picture where everyone can see it".

But this file does have some tips.

The real homework link is embedded in the provided PNG image file in two different ways:

1. Proper image steganography as described here and here. You can begin to decode this by noting the parity of bits in the upper left row of pixels.
2. A low-visibility watermark which will appear in the image if you play with it in an image editing program.

Either link will get you to the homework, but if you decode the URL encoded using method #1, you will also obtain a secret message worth a few points of extra credit.

I used ColorSync Utility to play around with the provided picture. By increasing the exposure and reducing the saturation of the image significantly, I was able to obtain the link required to access the HW. The link was https://tinyurl.com/sec590h4

OPTIONAL: For 5 points of extra credit, give the secret message included with the URL in the bitwise-encoded message. See the hints file for details.

Question 2: Endpoint security (9 points)

Let’s explore how a defender could have hardened the web server from Question 1 of the previous homework (“Victimco”).

Set up vulnerable web server (1pt)

On your Linux VM, perform the following steps to recreate the Victimco web server setup.

Fully update the environment:

```
  sudo apt update && sudo apt dist-upgrade && sudo apt autoremove
```

Note: if asked to update or replace a file relating to grub or apt, choose “keep the local version”. Duke VCM has environment-specific settings in these files we’ll want to preserve.

Install Apache web server, PHP, and the whois tool[1]:

```
sudo apt install tasksel
sudo tasksel install lamp-server
sudo apt install whois
```

Navigate to your VCM node in a local web browser and confirm you see the "Apache2 Ubuntu Default Page".

Remove the "Apache2 Ubuntu Default Page" page by deleting /var/www/html/index.html.

As root, download vco-web.tgz and extract it to /var/www/html/.
Note: don't put the .tgz file itself into /var/www/html/!

Edit /etc/apache2/apache2.conf so that AllowOverride for /var/www is as follows. This allows our .htaccess file to specify simple password login to prevent being compromised from random internet people and/or bots.

```
<Directory /var/www/>
        Options Indexes FollowSymLinks
        AllowOverride AuthConfig
        Require all granted
</Directory>
```

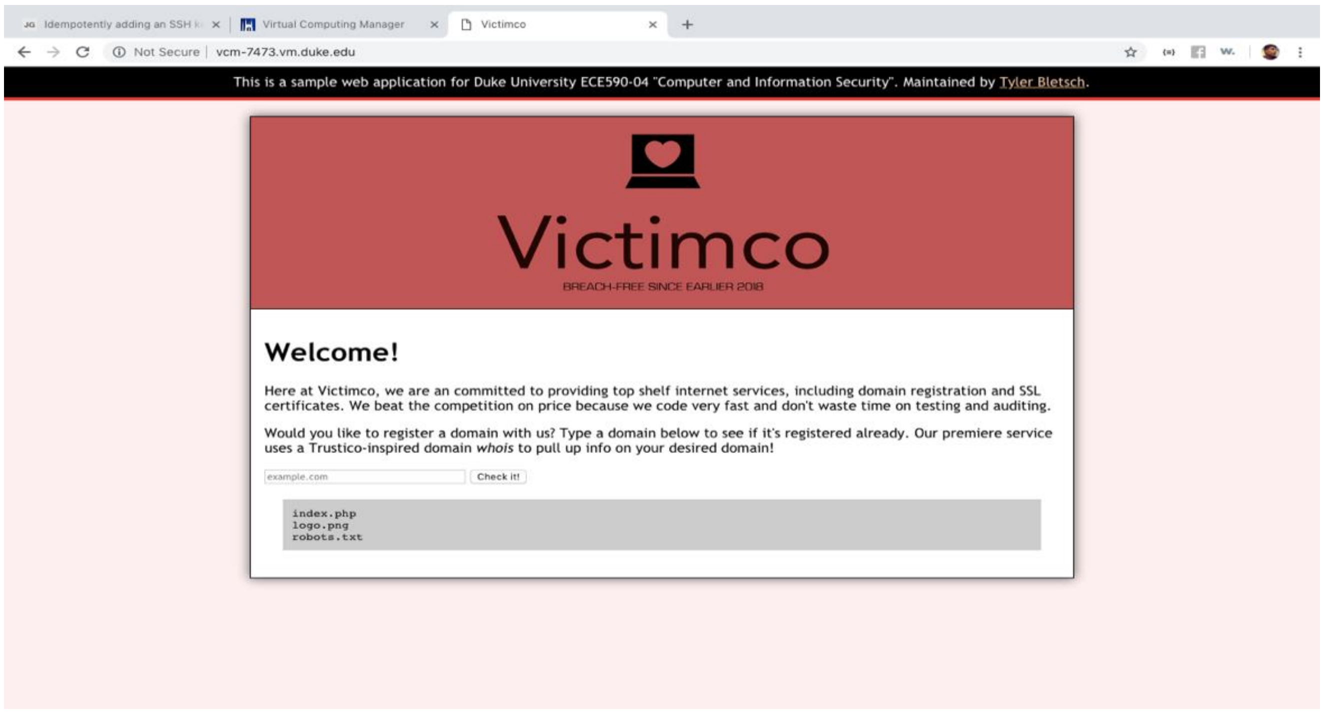
Restart Apache to make the setting change take effect.

```
sudo apachectl restart
```

Navigate to your Linux VM in a web browser and confirm that the Victimco page is working and vulnerable as before (including simple password authentication!).

For all of the above, you just need to post a screenshot showing the Victimco page up and running on your VM.

[1] You may wonder why we don't just `apt install` Apache and PHP directly using apt. It would indeed be wise to do so, but the `tasksel` command is a common recommendation if you google "install php ubuntu", so let's do it this naive way for now. We'll remove needless things later as we harden the server.



This is a sample web application for Duke University ECE590-04 "Computer and Information Security". Maintained by Tyler Bletsch.

Victimco

BREACH-FREE SINCE EARLIER 2018

Welcome!

Here at Victimco, we are an committed to providing top shelf internet services, including domain registration and SSL certificates. We beat the competition on price because we code very fast and don't waste time on testing and auditing.

Would you like to register a domain with us? Type a domain below to see if it's registered already. Our premiere service uses a Trustico-inspired domain *whois* to pull up info on your desired domain!

example.com [Check it!]

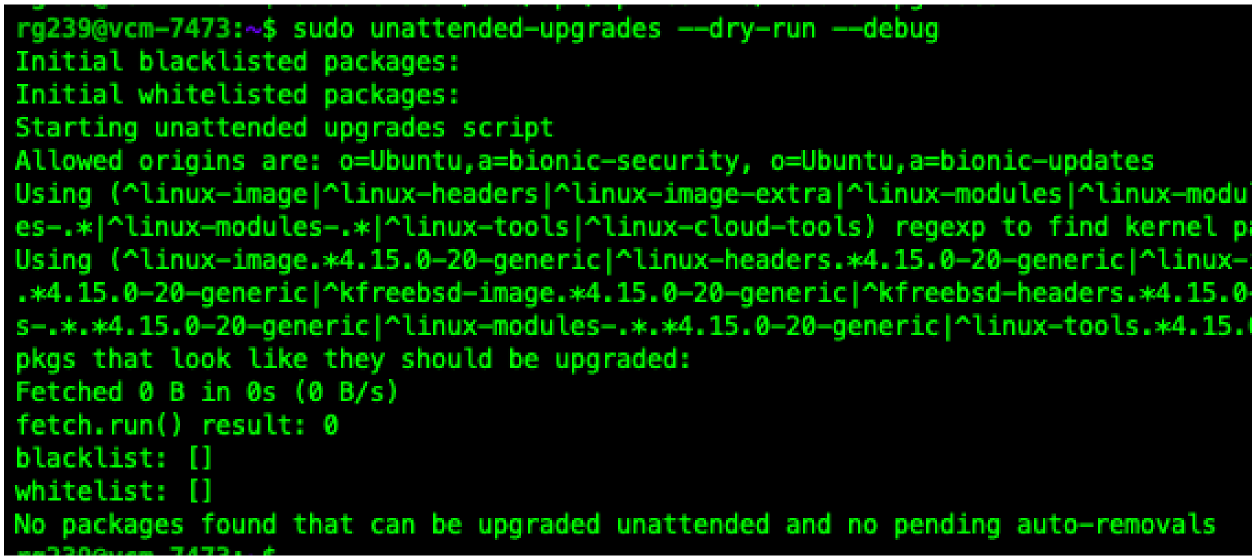
```
index.php
logo.png
robots.txt
```

Victimco is up and running, albeit vulnerable

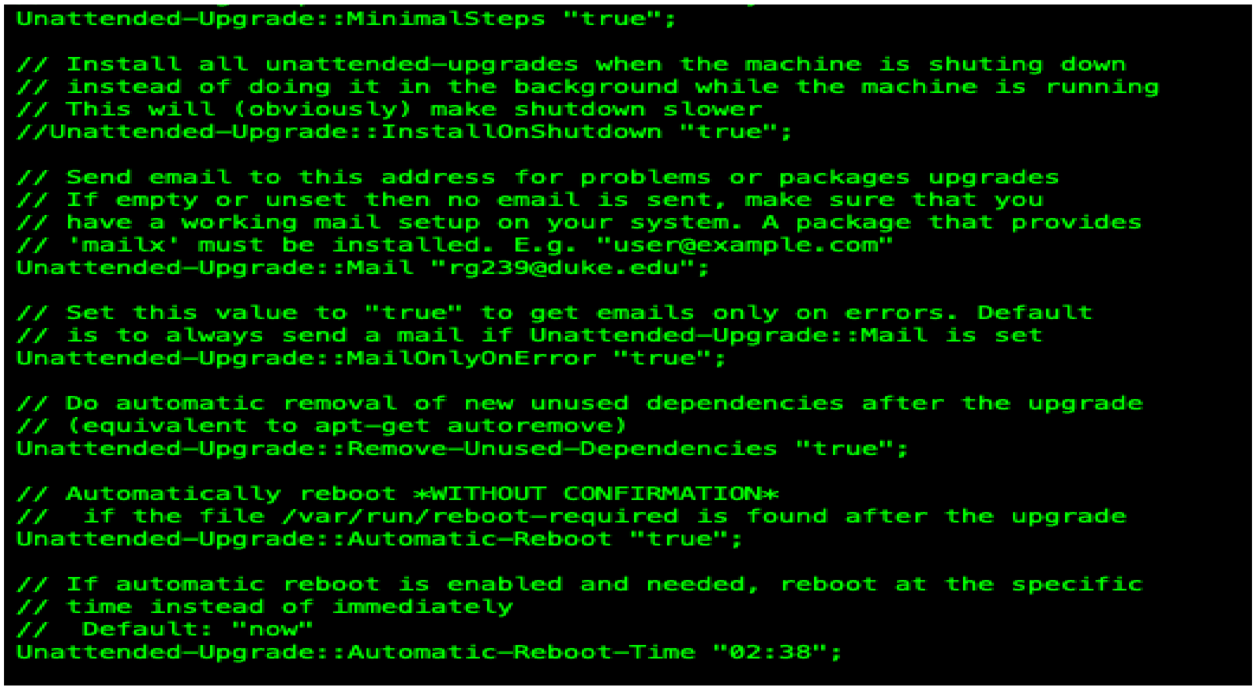
Enable automatic updates (1pt)

While automatic updates won't fix our particular web application flaw, it will close other holes at the OS and core application level. Duke VCM already enables automatic updates, but let's walk through the procedure to double-check.

Follow this procedure and succinctly document confirmation that the documented changes (or equivalent) are already present on your VM.

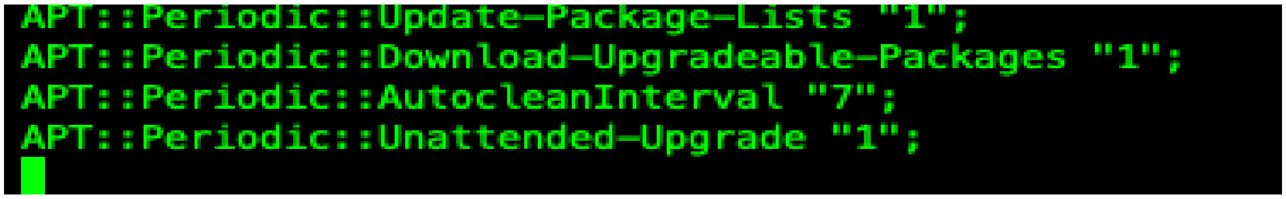


```
rg239@vcm-7473:~$ sudo unattended-upgrades --dry-run --debug
Initial blacklisted packages:
Initial whitelisted packages:
Starting unattended upgrades script
Allowed origins are: o=Ubuntu,a=bionic-security, o=Ubuntu,a=bionic-updates
Using (^linux-image|^linux-headers|^linux-image-extra|^linux-modules|^linux-modu
es-.*|^linux-modules-.*|^linux-tools|^linux-cloud-tools) regexp to find kernel p
Using (^linux-image.*4.15.0-20-generic|^linux-headers.*4.15.0-20-generic|^linux-
.*4.15.0-20-generic|^kfreebsd-image.*4.15.0-20-generic|^kfreebsd-headers.*4.15.0
s-.*.*4.15.0-20-generic|^linux-modules-.*.*4.15.0-20-generic|^linux-tools.*4.15.
pkgs that look like they should be upgraded:
Fetched 0 B in 0s (0 B/s)
fetch.run() result: 0
blacklist: []
whitelist: []
No packages found that can be upgraded unattended and no pending auto-removals
```



```
Unattended-Upgrade::MinimalSteps "true";

// Install all unattended-upgrades when the machine is shuting down
// instead of doing it in the background while the machine is running
// This will (obviously) make shutdown slower
//Unattended-Upgrade::InstallOnShutdown "true";

// Send email to this address for problems or packages upgrades
// If empty or unset then no email is sent, make sure that you
// have a working mail setup on your system. A package that provides
// 'mailx' must be installed. E.g. "user@example.com"
Unattended-Upgrade::Mail "rg239@duke.edu";

// Set this value to "true" to get emails only on errors. Default
// is to always send a mail if Unattended-Upgrade::Mail is set
Unattended-Upgrade::MailOnlyOnError "true";

// Do automatic removal of new unused dependencies after the upgrade
// (equivalent to apt-get autoremove)
Unattended-Upgrade::Remove-Unused-Dependencies "true";

// Automatically reboot *WITHOUT CONFIRMATION*
//  if the file /var/run/reboot-required is found after the upgrade
Unattended-Upgrade::Automatic-Reboot "true";

// If automatic reboot is enabled and needed, reboot at the specific
// time instead of immediately
//  Default: "now"
Unattended-Upgrade::Automatic-Reboot-Time "02:38";
```

All the required Unattended-Upgrade were already uncommented. The updates line was already uncommented.



```
APT::Periodic::Update-Package-Lists "1";
APT::Periodic::Download-Upgradeable-Packages "1";
APT::Periodic::AutocleanInterval "7";
APT::Periodic::Unattended-Upgrade "1";
```

Added the two lines:

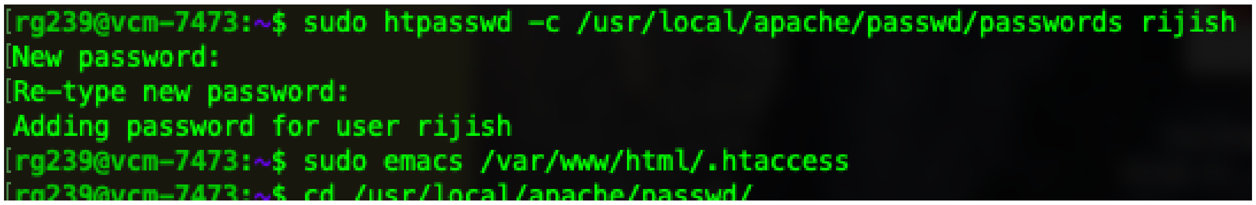
APT::Periodic::Download-Upgradeable-Packages "1";
APT::Periodic::AutocleanInterval "7";

Ensure correct settings (1pt)

Our vulnerable web application is a bit too small to have a large number of configuration options, but there is one thing you could consider changing: the HTTP authentication password. If you leave it with the provided default of username "student" and password "sec@590", other students could compromise your VM.

Research the htpasswd tool and Apache authentication in general and change your HTTP authentication password for your site. Document how you do this.

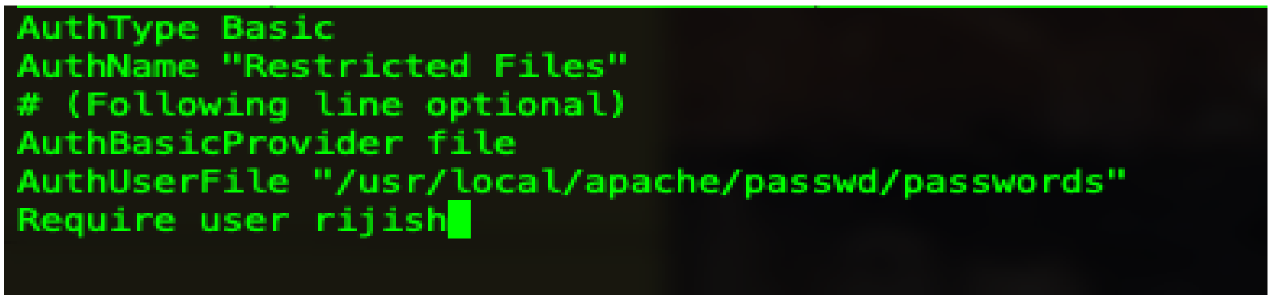
First, I created a password file which cannot be accessible from the web. To create the file, I used the command:



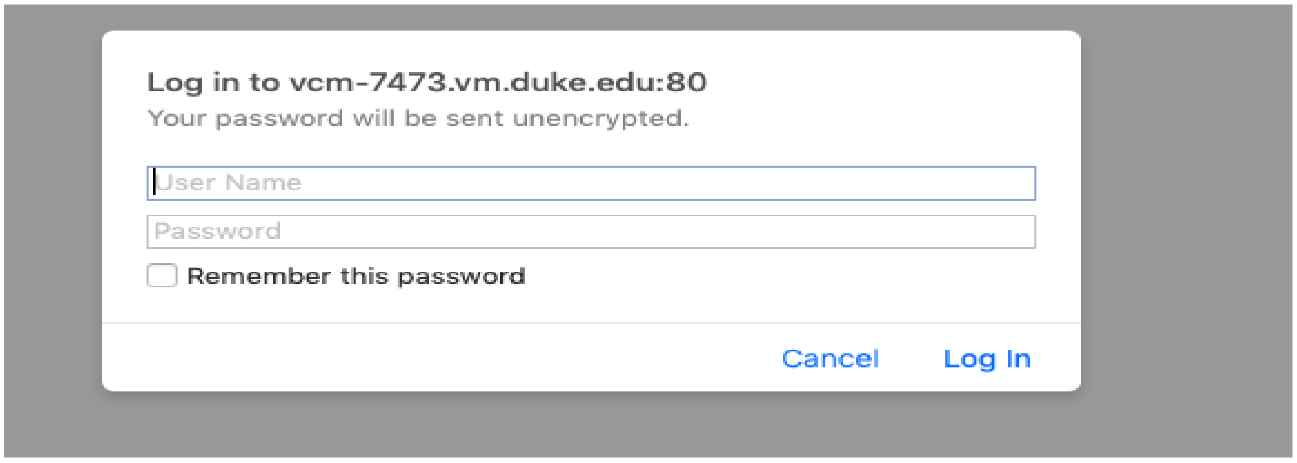
```
[rg239@vcm-7473:~$ sudo htpasswd -c /usr/local/apache/passwd/passwords rijish
[New password:
[Re-type new password:
 Adding password for user rijish
[rg239@vcm-7473:~$ sudo emacs /var/www/html/.htaccess
[rg239@vcm-7473:~$ cd /usr/local/apache/passwd/
```

Next, I had to configure the server to request a password and tell the server which users are allowed to access. I created a .htaccess file in the directory /var/www/html/

Contents of the .htaccess file:



```
AuthType Basic
AuthName "Restricted Files"
# (Following line optional)
AuthBasicProvider file
AuthUserFile "/usr/local/apache/passwd/passwords"
Require user rijish
```



Log in to vcm-7473.vm.duke.edu:80

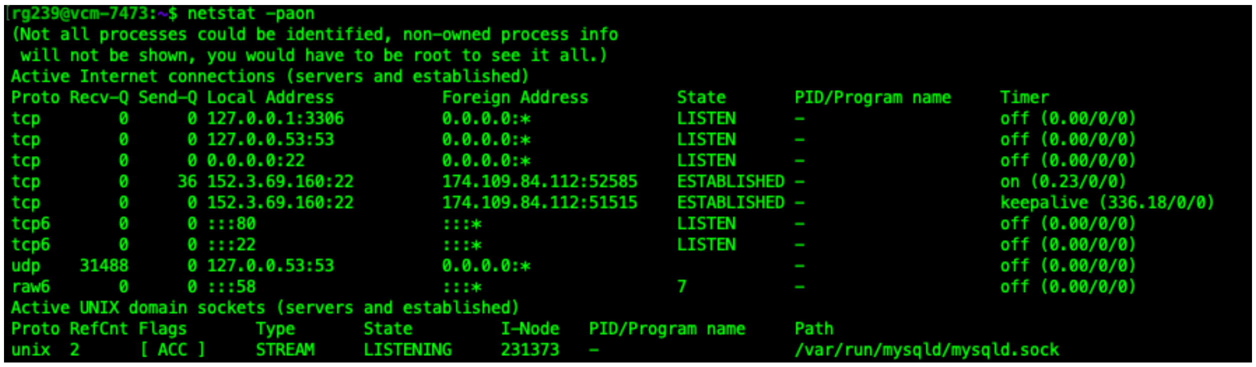
Your password will be sent unencrypted.

Authentication

Reduce attack surface: Software (2pt)

When we installed Apache and PHP, we did it by installing a “LAMP stack”, which stands for Linux, Apache, MySQL, and PHP. We aren’t using the MySQL part of the stack, so it’s a purely needless running service that brings its own set of issues.

Using netstat, show that a MySQL daemon is running and listening on a TCP port.



```
[rg239@vcm-7473:~$ netstat -paon
(Not all processes could be identified, non-owned process info
 will not be shown, you would have to be root to see it all.)
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name     Timer
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN      -                    off (0.00/0/0)
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN      -                    off (0.00/0/0)
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      -                    off (0.00/0/0)
tcp        0     36 152.3.69.160:22         174.109.84.112:52585    ESTABLISHED -                    on (0.23/0/0)
tcp        0      0 152.3.69.160:22         174.109.84.112:51515    ESTABLISHED -                    keepalive (336.18/0/0)
tcp6       0      0 :::80                   :::*                    LISTEN      -                    off (0.00/0/0)
tcp6       0      0 :::22                   :::*                    LISTEN      -                    off (0.00/0/0)
udp    31488      0 127.0.0.53:53           0.0.0.0:*                           -                    off (0.00/0/0)
raw6       0      0 :::58                   :::*                    7           -                    off (0.00/0/0)
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node   PID/Program name     Path
unix  2      [ ACC ]     STREAM     LISTENING     231373   -                    /var/run/mysqld/mysqld.sock
```

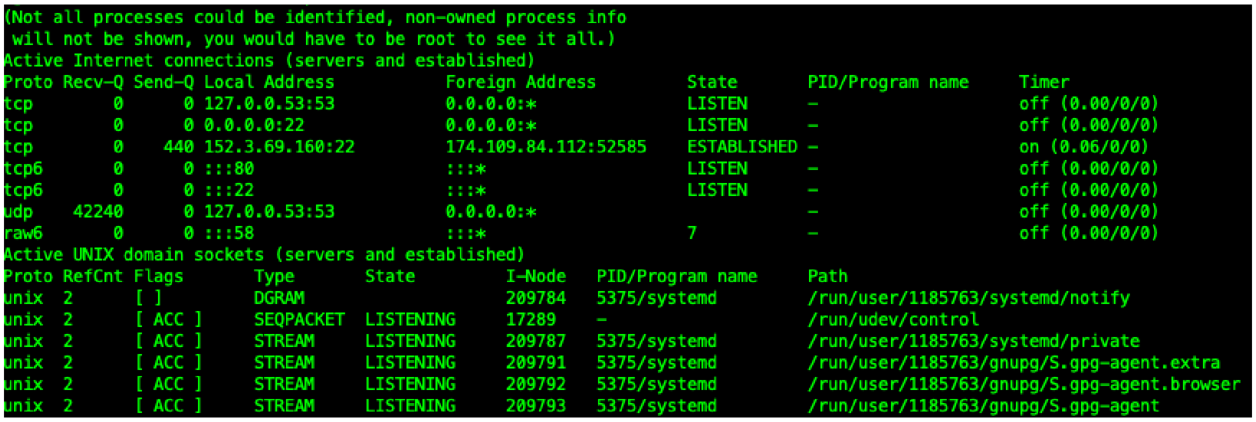
MySQL daemon is listening at port number 3306 which is the default TCP port for MySQL.

Completely remove MySQL from your Linux VM and document how you did so.

Show the same netstat output confirming that MySQL is no longer present.

To completely remove MySQL, I executed the following commands:

sudo apt-get remove --purge mysql*
sudo-apt get purge mysql*
sudo apt-get autoremove
sudo apt-get autoclean

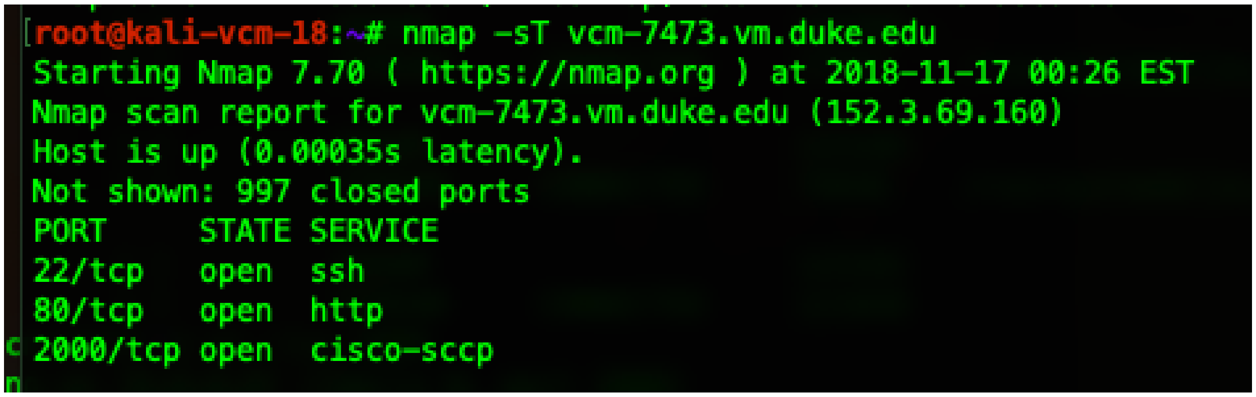


```
(Not all processes could be identified, non-owned process info
 will not be shown, you would have to be root to see it all.)
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name     Timer
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN      -                    off (0.00/0/0)
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      -                    off (0.00/0/0)
tcp        0    440 152.3.69.160:22         174.109.84.112:52585    ESTABLISHED -                    on (0.06/0/0)
tcp6       0      0 :::80                   :::*                    LISTEN      -                    off (0.00/0/0)
tcp6       0      0 :::22                   :::*                    LISTEN      -                    off (0.00/0/0)
udp    42240      0 127.0.0.53:53           0.0.0.0:*                           -                    off (0.00/0/0)
raw6       0      0 :::58                   :::*                    7           -                    off (0.00/0/0)
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node   PID/Program name     Path
unix  2      [ ]         DGRAM                    209784   5375/systemd         /run/user/1185763/systemd/notify
unix  2      [ ACC ]     SEQPACKET  LISTENING     17289    -                    /run/udev/control
unix  2      [ ACC ]     STREAM     LISTENING     209787   5375/systemd         /run/user/1185763/systemd/private
unix  2      [ ACC ]     STREAM     LISTENING     209791   5375/systemd         /run/user/1185763/gnupg/S.gpg-agent.extra
unix  2      [ ACC ]     STREAM     LISTENING     209792   5375/systemd         /run/user/1185763/gnupg/S.gpg-agent.browser
unix  2      [ ACC ]     STREAM     LISTENING     209793   5375/systemd         /run/user/1185763/gnupg/S.gpg-agent
```

MySQL daemon isn't running anymore

Reduce attack surface: Firewall (2pt)

The web server from Homework 3 was behind a NAT router with port forwarding, which made it necessary for attackers to use a reverse shell to gain persistent access. Our server, in contrast, has a public internet IP address, so any malware we happen to get infected with can simply open listening ports on the server to allow direct access for an attacker. Confirm this by using netcat to listen on port 2000, then from your Kali VM, show that port 2000 is open.



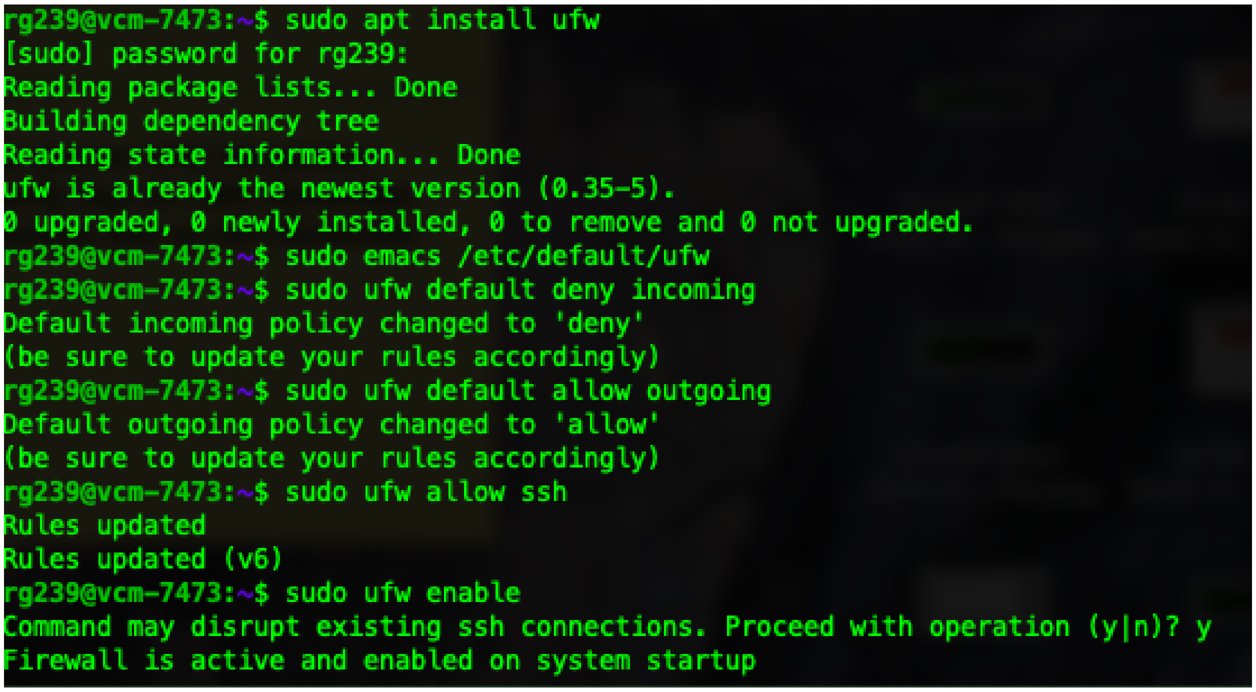
```
root@kali-vcm-18:~# nmap -sT vcm-7473.vm.duke.edu
Starting Nmap 7.70 ( https://nmap.org ) at 2018-11-17 00:26 EST
Nmap scan report for vcm-7473.vm.duke.edu (152.3.69.160)
Host is up (0.00035s latency).
Not shown: 997 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
2000/tcp open  cisco-sccp
```

Let's prevent this kind of thing by deploying a software firewall.

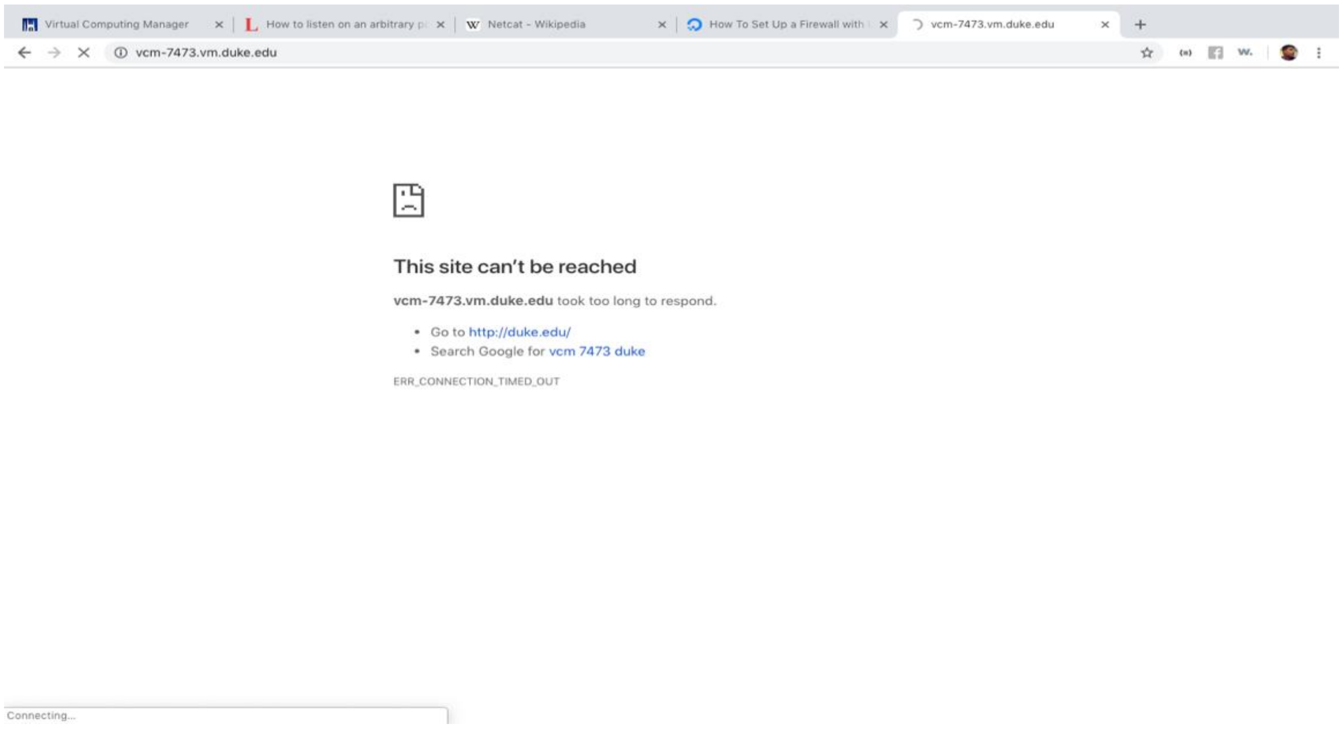
Refer to this introduction to setup "ufw" (the Ubuntu Firewall). Set the firewall to enable SSH (port 22) only and enable it[2], showing your work. Confirm that your web browser is now not able to access the Victimco web interface; show a screenshot.

Work done to enable SSH (port 22)

[2] NOTE: Be sure to allow access to port 22 (SSH) before you turn the firewall on, else you'll lose access to your VM! If this happens, you'll need to use the VCM interface to reload your VM from scratch and repeat the steps above, losing any data stored on the way.



```
rg239@vcm-7473:~$ sudo apt install ufw
[sudo] password for rg239:
Reading package lists... Done
Building dependency tree
Reading state information... Done
ufw is already the newest version (0.35-5).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
rg239@vcm-7473:~$ sudo emacs /etc/default/ufw
rg239@vcm-7473:~$ sudo ufw default deny incoming
Default incoming policy changed to 'deny'
(be sure to update your rules accordingly)
rg239@vcm-7473:~$ sudo ufw default allow outgoing
Default outgoing policy changed to 'allow'
(be sure to update your rules accordingly)
rg239@vcm-7473:~$ sudo ufw allow ssh
Rules updated
Rules updated (v6)
rg239@vcm-7473:~$ sudo ufw enable
Command may disrupt existing ssh connections. Proceed with operation (y|n)? y
Firewall is active and enabled on system startup
```



Virtual Computing Manager
How to listen on an arbitrary p
Netcat - Wikipedia
How To Set Up a Firewall with
vcm-7473.vm.duke.edu
vcm-7473.vm.duke.edu

This site can't be reached

vcm-7473.vm.duke.edu took too long to respond.

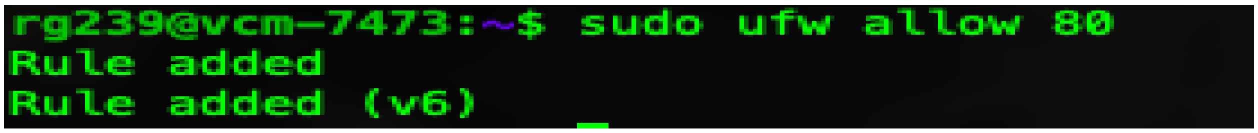
- Go to http://duke.edu/
- Search Google for vcm 7473 duke

ERR_CONNECTION_TIMED_OUT

Connecting...

The web interface cannot be accessed anymore

Now enable web access to port 80. Show how you did so. Confirm your browser is again able to access the VM.

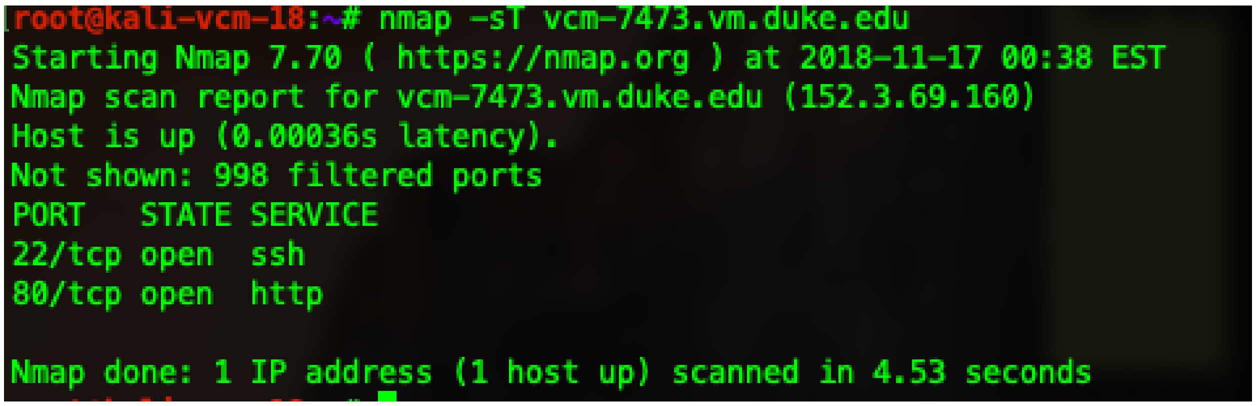


```
rg239@vcm-7473:~$ sudo ufw allow 80
Rule added
Rule added (v6)
```

The browser is able to access the VM again.

Now, arbitrary ports are no longer available for listening. Confirm this by using netcat to listen on port 2000, then from your Kali VM, show that port 2000 is <u>not</u> open.

We can observe that port 2000 is not open anymore.

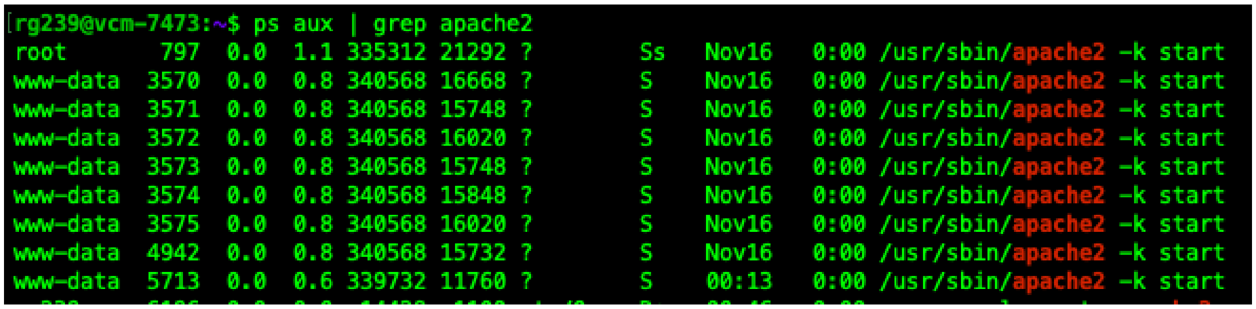


```
root@kali-vcm-18:~# nmap -sT vcm-7473.vm.duke.edu
Starting Nmap 7.70 ( https://nmap.org ) at 2018-11-17 00:38 EST
Nmap scan report for vcm-7473.vm.duke.edu (152.3.69.160)
Host is up (0.00036s latency).
Not shown: 998 filtered ports
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 4.53 seconds
```

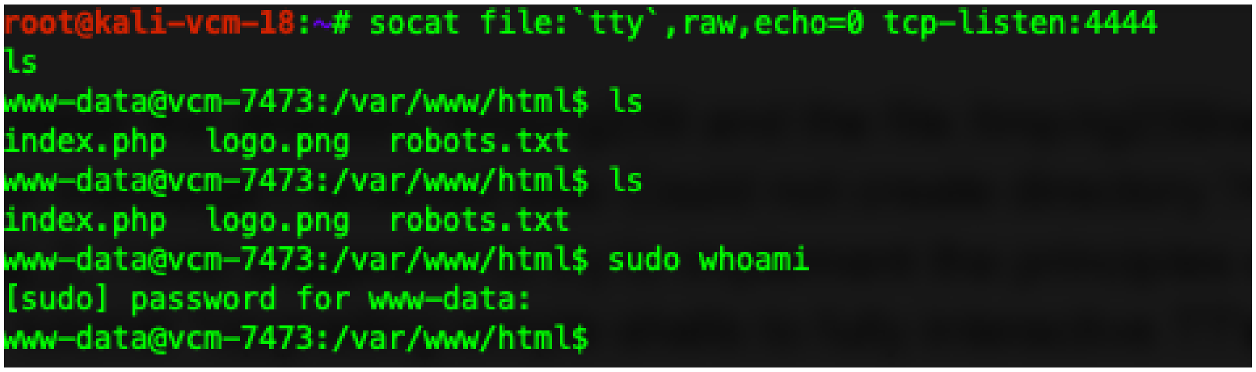
Limit privilege (1pt)

Linux by default already does user separation for daemons like the web server. Confirm this by identifying the username of the user running the apache2 processes in ps.

The username is www-data


```
[rg239@vcm-7473:~$ ps aux | grep apache2
 root       797  0.0  1.1 335312 21292 ?        Ss   Nov16   0:00 /usr/sbin/apache2 -k start
 www-data  3570  0.0  0.8 340568 16668 ?        S    Nov16   0:00 /usr/sbin/apache2 -k start
 www-data  3571  0.0  0.8 340568 15748 ?        S    Nov16   0:00 /usr/sbin/apache2 -k start
 www-data  3572  0.0  0.8 340568 16020 ?        S    Nov16   0:00 /usr/sbin/apache2 -k start
 www-data  3573  0.0  0.8 340568 15748 ?        S    Nov16   0:00 /usr/sbin/apache2 -k start
 www-data  3574  0.0  0.8 340568 15848 ?        S    Nov16   0:00 /usr/sbin/apache2 -k start
 www-data  3575  0.0  0.8 340568 16020 ?        S    Nov16   0:00 /usr/sbin/apache2 -k start
 www-data  4942  0.0  0.8 340568 15732 ?        S    Nov16   0:00 /usr/sbin/apache2 -k start
 www-data  5713  0.0  0.6 339732 11760 ?        S    00:13   0:00 /usr/sbin/apache2 -k start
```

Show that this separation is helpful by exploiting the web application to get a shell, then attempting and failing to use sudo to run a command as root.


```
root@kali-vcm-18:~# socat file:`tty`,raw,echo=0 tcp-listen:4444
ls
www-data@vcm-7473:/var/www/html$ ls
index.php  logo.png  robots.txt
www-data@vcm-7473:/var/www/html$ ls
index.php  logo.png  robots.txt
www-data@vcm-7473:/var/www/html$ sudo whoami
[sudo] password for www-data:
www-data@vcm-7473:/var/www/html$
```

Conclusion (1pt)

Did any of the steps above prevent the web-based vulnerability from working?

Nope, it didn't.

What processes above, if any, reduced the severity of impact of the web-based vulnerability?

The enabling of the firewall and the user privilege separation reduced the severity of impact of the web-based vulnerability to an extent.

A host-based intrusion detection system (HIDS) detects brute force attacks, changes to key system files, root-based installation of packages or kernel modules, the opening of newly listening ports, and more. Does triggering a reverse shell fall into any of these categories? As a result, would a HIDS have preventing an attacker from using the web server as a jumping-off point for an attack on the accounting server?

The reverse shell does not fall into any of these categories. Hence, a HIDS would not be able to prevent an attacker from using the web server as jumping-off point for an attack on accounting server.

Given the bleak answers you just gave, why were all the steps above still worth doing? What kinds of attacks *could* they mitigate?

The above countermeasures can protect the system against unauthorized users without proper credentials and prevent direct attacks by limiting access to arbitrary ports with the help of a firewall.

Question 3: Buffer Overflow (10 points)

This question is highly flexible, so read carefully!

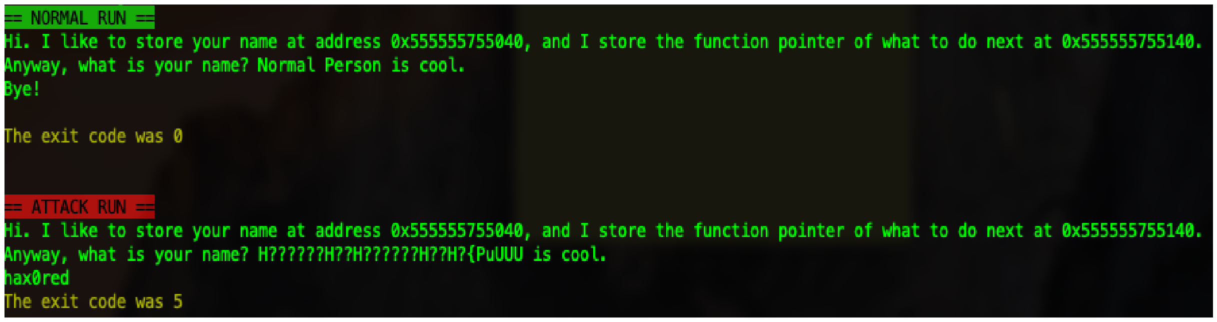
Here's how to disable ASLR. This article in the highly esteemed blog publication "Fr33k K0mPu73R" explains how to turn off W^X (also known as NX support) at compile time. Note that for the provided vulnerable program, you will probably not need to disable ASLR, as our particular ASLR implementation doesn't affect the global data region, which is what will be exploited.

Here's the base problem:

- Prepare VM: In your Linux VM, install nasm ("sudo apt install nasm").
  - Note: If you want to build an attack for another OS, contact the instructors for permission, which we'll likely give, but you'll be on your own.
- Get set up: Download, extract, and test the example exploit kit.
  - The Makefile is set up to build the vulnerable program with W^X disabled.
  - Read through the vulnerable program "vuln1.c" and the attack script "attack.asm" closely.
  - Watch this walkthrough video I recorded.
    NOTE: This video was recorded for an earlier version of the problem, so there are slight differences, especially:
    - It claims that we need to avoid null bytes in the attack buffer, this is actually not true for a gets() exploit.
    - It claims that we have to turn off ASLR for this attack to work -- it turns out that's not true, as the ASLR implementation on Linux doesn't move the global data region, which is what we're exploiting.
  - The included attack will make the program skip saying "Bye!" and exit with status code 5 instead of the usual 0. Recall that the exit status of a program can be checked by running "echo $?" after the program exits.
  - Assuming your VM's memory map looks like mine, you should be able to run the attack straight away by executing the including "demo" script. If it doesn't work,

you may need to update the memory location constants in the attack script, re-build, and try again.

- Make exploit: Develop an attack buffer that makes the program print “hax0red”.
    - Note: your *exploit* must cause it to say this -- merely having the word “hax0red” appear among the program’s usual “so-and-so is cool” output does not count.
    - You can use any tool you wish to develop the attack buffer, though using the included NASM attack.asm as a starting point is probably easiest.
    - gdb is your friend.
    - If pursuing the extra credit (see below), you may target a program *other than* the provided vulnerable program.
- Hack: Run your attack and paste a screenshot of it succeeding below.



```
== NORMAL RUN ==
Hi. I like to store your name at address 0x555555755040, and I store the function pointer of what to do next at 0x555555755140.
Anyway, what is your name? Normal Person is cool.
Bye!

The exit code was 0


== ATTACK RUN ==
Hi. I like to store your name at address 0x555555755040, and I store the function pointer of what to do next at 0x555555755140.
Anyway, what is your name? H??????H??H??????H??H?{PuUUU is cool.
hax0red
The exit code was 5
```

- Document: Make a file called “readme.txt” that lists all dependencies your attack has (packages needed to be installed, special steps to be taken, including the steps to disable ASLR and W^X)).
- Submit: Gather up *EVERYTHING* involved in the attack (readme.txt, the vulnerable program source and binary, the attack source code and attack binary file, and any Makefiles or helper scripts). Zip all of this up and submit it to Sakai as “<NetID>_attack.zip”.

Here’s some random tips to help you:

- GDB quick reference card.
- Syscall numbers.
- Intel x86 assembly reference:
    - The giant Intel big book: the authoritative source.
    - Some site.
    - NASM manual, including instruction list.
    - Low-level conversions between hex bytes and CPU instructions: Numeric order, mnemonic order.

- You can use “ndisasm -b64 <file>” to do a raw disassembly of your attack buffer, with output in Intel (NASM) syntax.
- You can use “hd <file>” to get a hexdump of it, which is useful when looking for nulls or whitespace bytes that snuck into your attack buffer.

Extra credit achievements

You may apply any combination of the following achievements to the problem for extra credit. Put an "X" in the brackets and color the whole line red for any of these variations that you're claiming. Then, in your readme, *explain in detail* how you achieved the goals claimed.

Note -- there are a lot of unknowns in this program, so the instructor reserves the right to adjust extra credit if there's some trivial way to get way too many points.

Extra credit for protections left enabled:

- [ ] (+5) Defeat ASLR: Leave ASLR on *and* have the vulnerable program store its data in the stack or heap instead of the global region.
- [ ] (+5) Defeat W^X: Leave W^X on.

Extra credit for choice of target:

- [ ] (+2) Custom target: Develop a significantly different sample vulnerable program from scratch. This program must use a different exploitable function instead of gets() (e.g. scanf()) and/or a different kind of control data instead of a function pointer (e.g. a return address).
- [ ] (+2) Server target: Your target program accepts the attack buffer over the network.
- [ ] (+10) Kernel target: Exploit code operating in kernel space instead of user space. In this case, your printing of the message can go to the kernel log or to another creative destination.
- [ ] (+15) Real target: Instead of a toy sample program, attack a real program. The program must have been in significant production at some point in the past 15 years. If you go with this option, your attack must still be completely from scratch -- you can't just use metasploit or a sample script or something. You must also cite sources in your readme and explain in detail how the vulnerability and your exploit work.
- [ ] (+30) Very real target: Base your attack on a brand-new never-before-published vulnerability in a real program in significant production use. If you go with this option, you must document in detail how the vulnerability was discovered, how your exploit works, and you must submit a report to the software vendor as well as the instructors. This achievement implies "Real target".
- [ ] (+60) Bragging rights: Awarded if you did the "Very real target" achievement and your discovery of the vulnerability results in the publication of an advisory from an industry-standard source (CVE, NVD, an article in a major vendor knowledge base such as the Microsoft KB, public bug report and patch, etc.). The instructor will also buy you dinner.

Extra credit for payload:

- [ ] (+2) Verbose hacker: If you attack prints more text than just "hax0red", including multiple newline characters. This can be a challenge since the newline character is the delimiter for gets(), which is used in the sample vulnerable program.
- [ ] (+5) Math hack: If your payload computes and prints the first 100 Fibonacci numbers in addition to the message. Note that the attack must *compute* the numbers, not just print them from memory.
- [ ] (+5) File IO: If your payload causes the program to appear to the user to function normally in every way, except the message you're printing is saved to a file called "0wned.txt". (Note the zero in the filename -- this is used to indicate that we are very cool, mature people.)
- [ ] (+10) Reverse shell: If your payload connects out to a TCP host and, upon successful connection, establishes remote control of a /bin/bash shell.
- [ ] (+10) Code reuse: If your payload operates *entirely* on code reuse and does no code injection.
- [ ] (+15) Dr. Bletsch's dissertation: If you payload operates on code reuse and does no code injection, *and* does not exploit a single ret instruction.

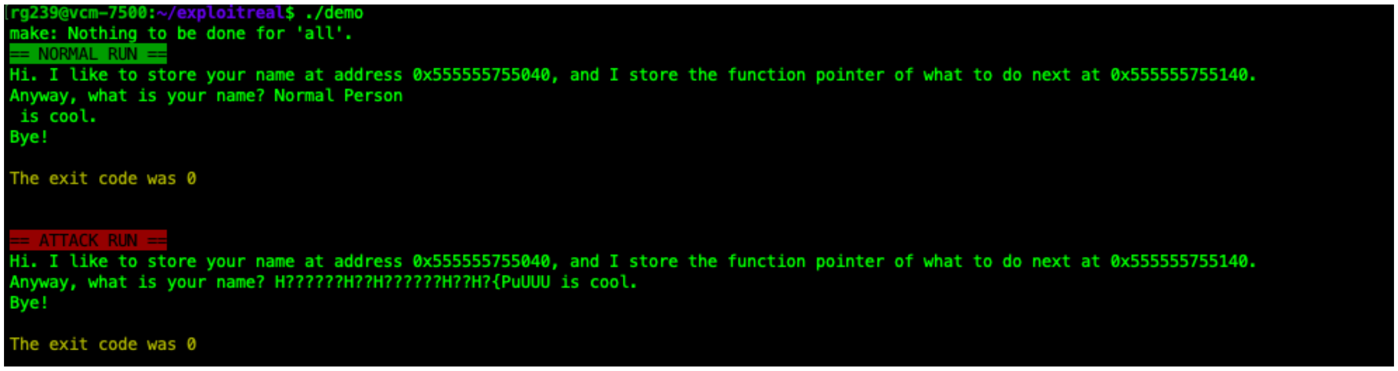
Extra credit wildcard:

- [ ] (+$1000) Overachiever: If you unlock every extra credit achievement above, I will hand you a thousand dollars.
- [ ] (+?) Wildcard: If you're doing something else fancy with your attack that you feel is worth more points, pitch it to the instructor. You will need to document this additional feature in your readme file.

Question 4: Buffer Overflow Prevention (5 points)

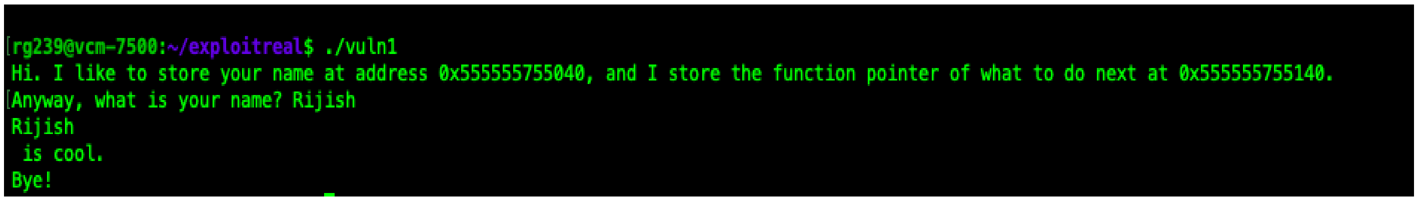
Fix it: Take whatever program you used as your target for the previous problem and fix the vulnerability. Submit the fixed code as “<NetID>_fixed.zip” to Sakai.

Prove you actually fixed it: Paste a screenshot of the program NOT being exploited by the attack input from the previous question. Note: If a screenshot cannot show the improvement for your particular program, contact the instructor to discuss an alternative form of response.



```
[rg239@vcm-7500:~/exploitreal$ ./demo
make: Nothing to be done for 'all'.
== NORMAL RUN ==
Hi. I like to store your name at address 0x555555755040, and I store the function pointer of what to do next at 0x555555755140.
Anyway, what is your name? Normal Person
 is cool.
Bye!

The exit code was 0


== ATTACK RUN ==
Hi. I like to store your name at address 0x555555755040, and I store the function pointer of what to do next at 0x555555755140.
Anyway, what is your name? H??????H??H??????H??H?{PuUUU is cool.
Bye!

The exit code was 0
```

Prove you didn’t make it worse: Paste a screenshot of the program continuing to function correctly under normal input.



```
[rg239@vcm-7500:~/exploitreal$ ./vuln1
Hi. I like to store your name at address 0x555555755040, and I store the function pointer of what to do next at 0x555555755140.
[Anyway, what is your name? Rijish
Rijish
 is cool.
Bye!
```

Question 5: SQL Injection (6 points)

Use a SQL injection attack to login to http://sqldemo.googz.us/. This is a simple login page written in PHP and modeled after the many intro-to-PHP guides available online.

Note: do not modify the database in any way; just use the attack to fool the script into logging you in. Also, this is running on a production web host, so do not use brute force techniques in your attack.

Paste the username and password you used below.

Username: ' or '1'='1'#,'81dc9bdb52d04dc20036dbd8313ed055'
Password: 1234

Upon successful login, the system will give you a secret code. Paste this code below.

The secret code is ‘Some Sharks Typing On A Computer'.

Extra credit +5: Find out jimmy’s full name.

Extra credit +5: Find out the actual admin password.

Note: If you wish to use a tool or script to make successful requests in pursuit of the extra credit, email me first. If such a request is granted, you will need to throttle your attack to no more than one request every <u>five (5) seconds</u>. No tool is need to get the main part of the question.

Question 6: Countermeasures (4 points)

Many times in security systems administration you will run into the situation of where there is a new vulnerability out with a known exploit yet the vender has not released a patch. However, your vulnerable systems must stay online despite the risks. You should be able to provide some countermeasures for a situation like this. Follow the 3 Layer Security (prevention, detection, response) model when developing your countermeasures.

Consider the following scenario: You are using a licensing service on your Windows server to limit the number of users that can run Matlab. When a user starts Matlab, the program checks with the license service and if less than 60 people are running Matlab, then it allows the application to start. You have just received a security alert stating there is a buffer overflow vulnerability in this license service. The vendor has not released a patch. The server has no firewall so it could be accessed by anyone. How would you provide the same access but prevent, detect, and/or respond to attacks based on this vulnerability?

Countermeasures to provide the same access but prevent, detect and respond to attacks based on this vulnerability:

a) Prevent code execution on the stack. We can modify the organization of stack allocated data to include a canary value, which when destroyed by a stack buffer overflow, would show that a buffer preceding it in memory has been overflowed. By verifying the canary value, execution of the affected program can be terminated.
b) We can enable some form of tagging, making sure that memory allotted for storing data wouldn't contain executable code. Certain areas of the main memory can be marked as non-allocated.
c) Implement a firewall which only allows trusted users and limit the number of connections to 59. After that, the firewall prevents additional users from connecting. We can check the sizes for incoming packets and reject anything which seem unusual.

Question 7: Backups with rsnapshot (6 points)

NOTE: This question is almost identical to one posed in my Enterprise Storage Architecture course. If you are currently or have been previously enrolled in that course, you may either (a) simply write "I did this in Enterprise Storage Architecture" for full credit, or (b) re-do the procedure from scratch on your Linux VM as described. In any case, you may not re-use material from the other course for this question.

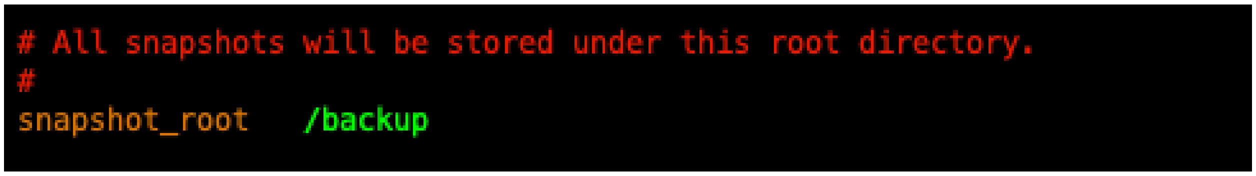
Let's build an automated backup system. Make an additional Linux VM in Duke VCM. (You should have a free slot to do this because you deleted throw-away Windows VM from the previous assignment.) This new Linux VM will refer to as the Backup Server. The thing we'll be backing up is your pre-existing Linux VM; this is the Backup Client, You can use this guide, and note the following:

- Backup source: We're going to back up your user home directory on the backup client machine. This is /home/<NETID>.
- Backup destination: As root, make a directory /backup.

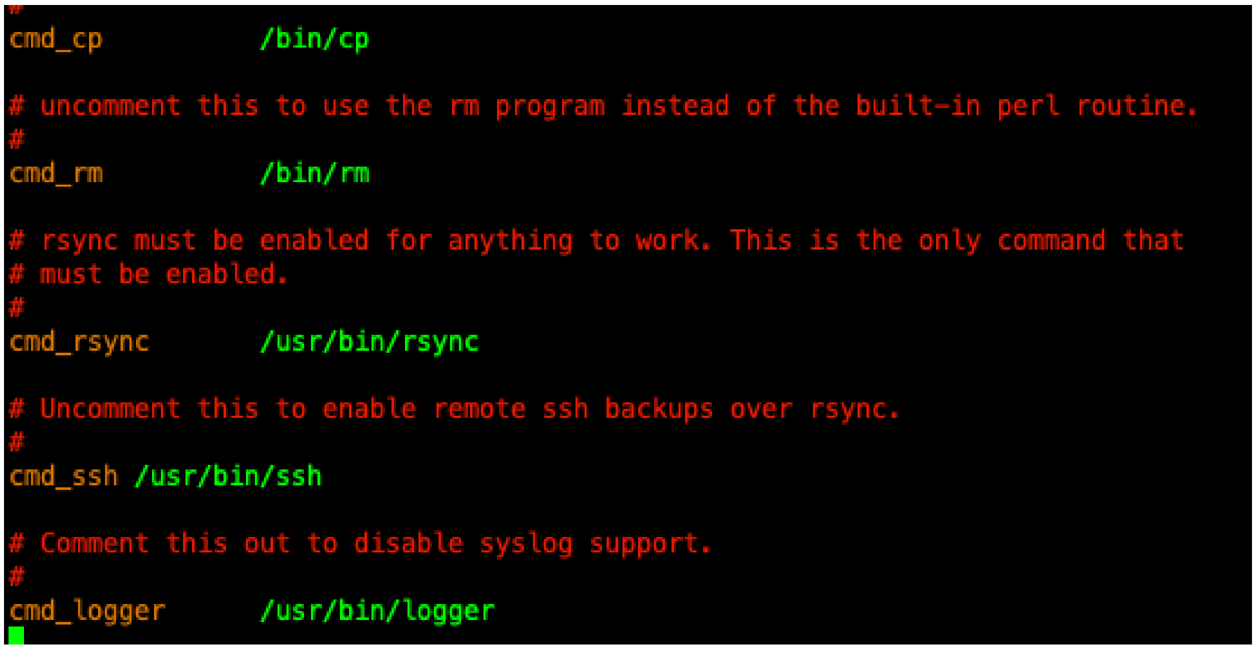
- SSH keys needed: Because the source and destination are different machines, you'll need to set up SSH keys so that the backup server can *pull* data as needed.
- Frequency: Set up nightly and weekly intervals.
- Automation: You do NOT need to do cron-based automation. Once you have rsnapshot configured, you can just run "rsnapshot nightly" to perform the backup that would happen nightly, and "rsnapshot weekly" to perform the backup that would happen weekly. You can run these commands as often as you wish; there's nothing in the software that actually needs the backups to be done nightly/weekly as opposed to every few dozen seconds (i.e., rsnapshot does not actually care that they're called "nightly" or "weekly").

Provide screenshots or a terminal log of these steps in your writeup.
Provide the relevant portions of rsnapshot.conf.

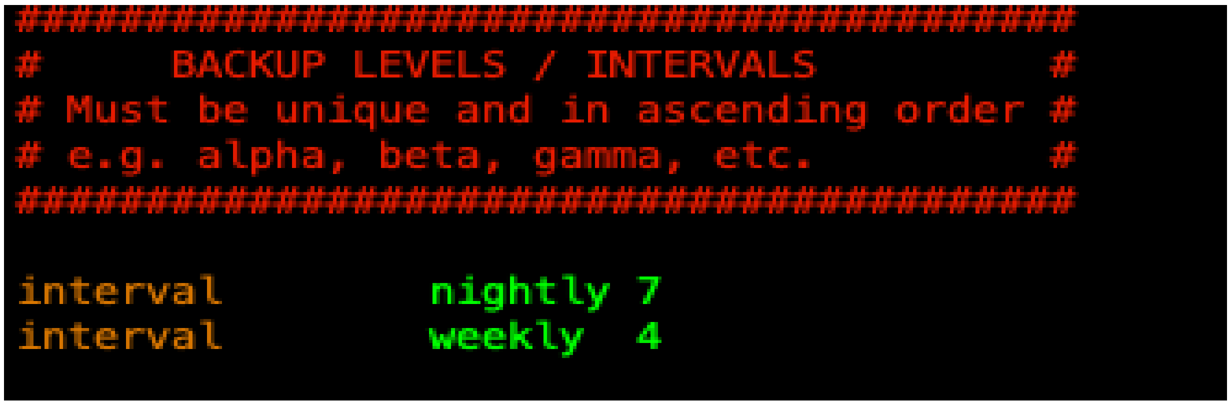
Relevant portions of **rsnapshot.conf**

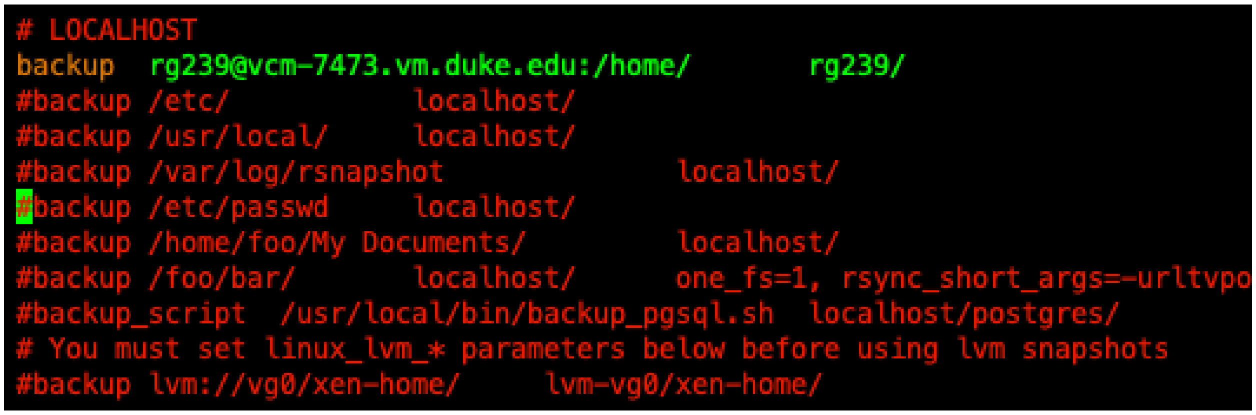


```
# All snapshots will be stored under this root directory.
#
snapshot_root   /backup
```



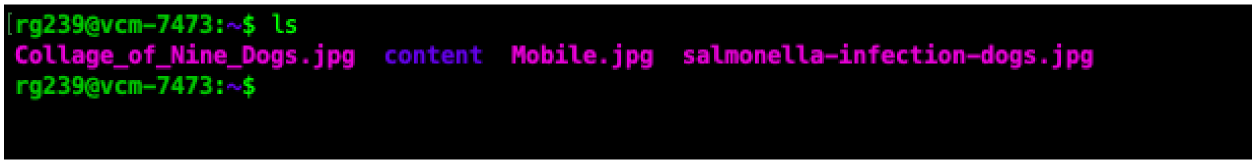
```
cmd_cp          /bin/cp

# uncomment this to use the rm program instead of the built-in perl routine.
#
cmd_rm          /bin/rm

# rsync must be enabled for anything to work. This is the only command that
# must be enabled.
#
cmd_rsync       /usr/bin/rsync

# Uncomment this to enable remote ssh backups over rsync.
#
cmd_ssh /usr/bin/ssh

# Comment this out to disable syslog support.
#
cmd_logger      /usr/bin/logger
```


```
###########################################
#      BACKUP LEVELS / INTERVALS          #
# Must be unique and in ascending order #
# e.g. alpha, beta, gamma, etc.           #
###########################################

interval        nightly 7
interval        weekly  4
```

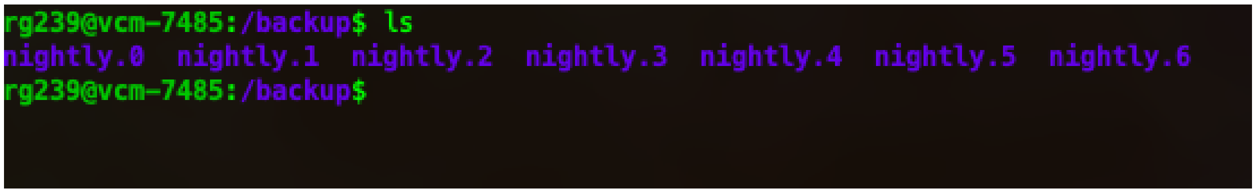

```
# LOCALHOST
backup  rg239@vcm-7473.vm.duke.edu:/home/       rg239/
#backup /etc/           localhost/
#backup /usr/local/     localhost/
#backup /var/log/rsnapshot              localhost/
#backup /etc/passwd     localhost/
#backup /home/foo/My Documents/         localhost/
#backup /foo/bar/       localhost/      one_fs=1, rsync_short_args=-urltvpo
#backup_script  /usr/local/bin/backup_pgsql.sh  localhost/postgres/
# You must set linux_lvm_* parameters below before using lvm snapshots
#backup lvm://vg0/xen-home/     lvm-vg0/xen-home/
```

Let's test it. Open one terminal window as root, and another terminal window as the non-root user. Do the following, and for each step, show the process via screenshots or terminal logs. Label or otherwise identify each step you're doing in your screenshot/log.

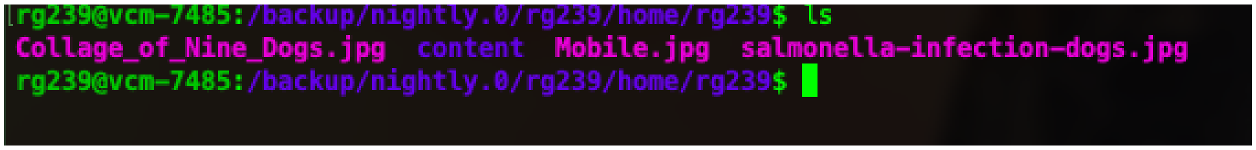
1. As the user on the backup client, add some content to the user home directory.


```
rg239@vcm-7473:~$ ls
Collage_of_Nine_Dogs.jpg  content  Mobile.jpg  salmonella-infection-dogs.jpg
rg239@vcm-7473:~$
```

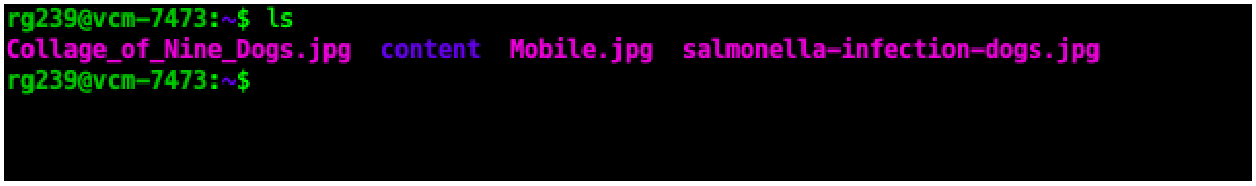
2. As root on the backup server, do several nightly and weekly backups to populate your backups.



```
rg239@vcm-7485:/backup$ ls
nightly.0  nightly.1  nightly.2  nightly.3  nightly.4  nightly.5  nightly.6
rg239@vcm-7485:/backup$
```

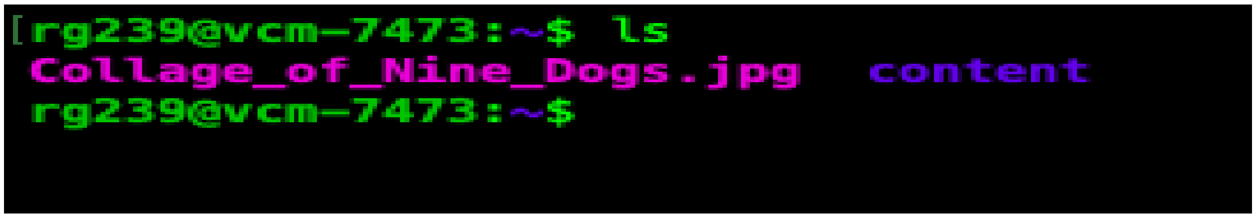


```
rg239@vcm-7485:/backup/nightly.0/rg239/home/rg239$ ls
Collage_of_Nine_Dogs.jpg  content  Mobile.jpg  salmonella-infection-dogs.jpg
rg239@vcm-7485:/backup/nightly.0/rg239/home/rg239$
```



```
rg239@vcm-7473:~$ ls
Collage_of_Nine_Dogs.jpg  content  Mobile.jpg  salmonella-infection-dogs.jpg
rg239@vcm-7473:~$
```

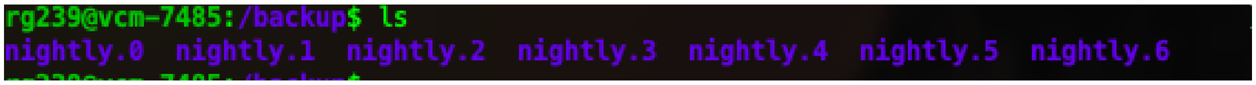
3. As the user on the backup client, “corrupt” an important file (overwrite or delete it).



```
rg239@vcm-7473:~$ ls
Collage_of_Nine_Dogs.jpg  content
rg239@vcm-7473:~$
```

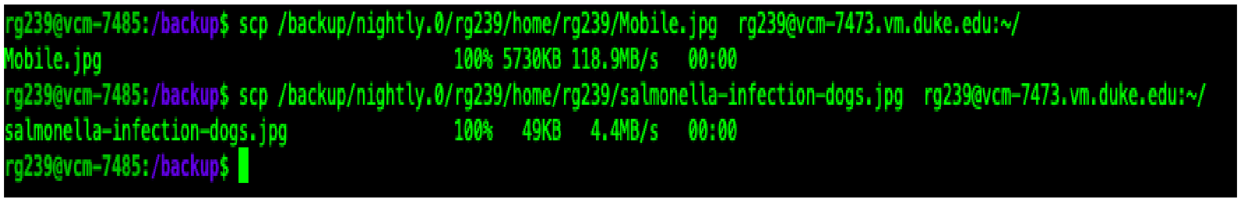
Deleted the two files – salmonella-infection-dogs.jpg
Mobile.jpg

4. As root on the backup server, take another few nightly backups to simulate time passing.

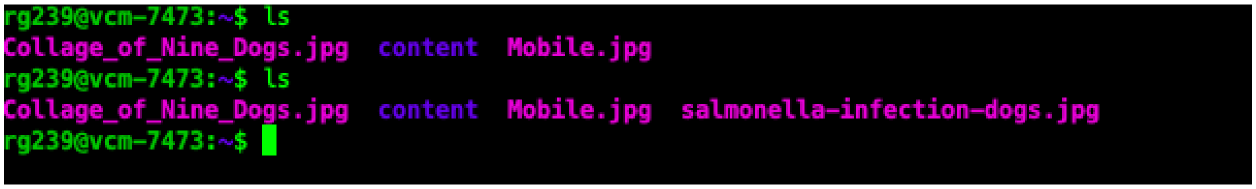


```
rg239@vcm-7485:/backup$ ls
nightly.0  nightly.1  nightly.2  nightly.3  nightly.4  nightly.5  nightly.6
```

5. Copy the appropriate backup from the backup server to the client’s home directory, thus restoring the damaged file.



```
rg239@vcm-7485:/backup$ scp /backup/nightly.0/rg239/home/rg239/Mobile.jpg  rg239@vcm-7473.vm.duke.edu:~/
Mobile.jpg                                    100% 5730KB 118.9MB/s   00:00
rg239@vcm-7485:/backup$ scp /backup/nightly.0/rg239/home/rg239/salmonella-infection-dogs.jpg  rg239@vcm-7473.vm.duke.edu:~/
salmonella-infection-dogs.jpg                 100%   49KB   4.4MB/s   00:00
rg239@vcm-7485:/backup$
```


```
rg239@vcm-7473:~$ ls
Collage_of_Nine_Dogs.jpg  content  Mobile.jpg
rg239@vcm-7473:~$ ls
Collage_of_Nine_Dogs.jpg  content  Mobile.jpg  salmonella-infection-dogs.jpg
rg239@vcm-7473:~$ 
```

Question 8: Run a honeypot and see what you get (8 points)

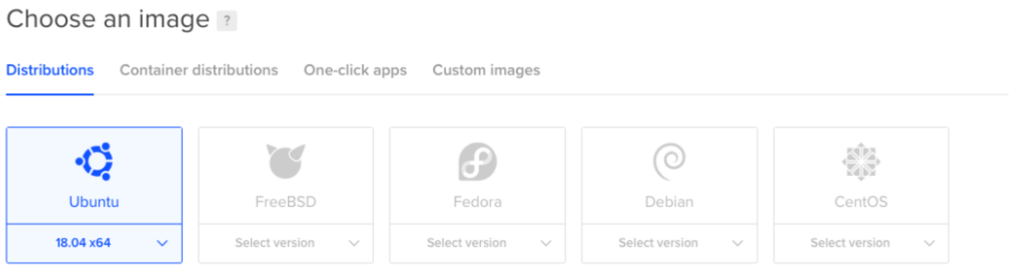
A honeypot is an intentionally-vulnerable system with monitoring. It is used to see what attackers do when they succeed in a given environment. There are many kinds of honeypots (Windows/RDP, Linux/SSH, etc.). They can be divided into "real environment" (i.e., the attacker is really breaking in, but the system they're gaining access to is one we don't care about) versus "simulated environment" (i.e., it looks like the real thing, but every operation is simulated and sandboxed).

Setup (4pts)

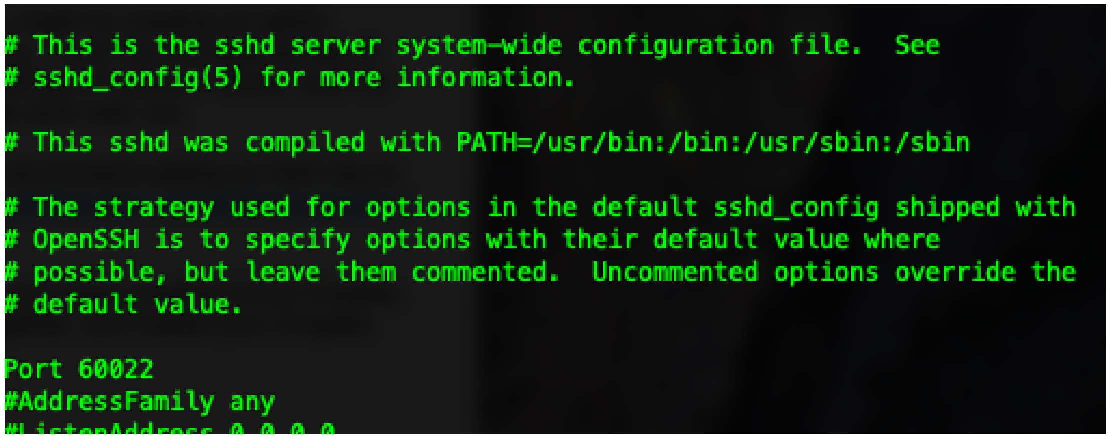
We'll be setting up a simulated-environment SSH sandbox called Cowrie, directions here.

Because Duke IT has a lot of existing automated defenses, for best results, you'll be installing to the Amazon cloud as an EC2 instance. As far as money goes, you can use the AWS Free Tier or the AWS Educate program, or just give them the ~$5 this experiment should cost. You may use another commercial cloud (Linode, Digital Ocean, Azure, etc.) if you wish. You may want to play with what geographic area you place your VM, as this can affect what kind of attacks you see.
Show the steps needed to set up this environment.

- I used Digital Ocean to set up my Ubuntu VM.



- I changed the SSH port to 60022.



```
# This is the sshd server system-wide configuration file.  See
# sshd_config(5) for more information.

# This sshd was compiled with PATH=/usr/bin:/bin:/usr/sbin:/sbin

# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented.  Uncommented options override the
# default value.

Port 60022
#AddressFamily any
#ListenAddress 0.0.0.0
```

- Run the commands:
  apt update
  sudo apt-get install git python-virtualenv libssl-dev libffi-dev build-essential libpython3-dev python3-minimal authbind [Download Python virtual environment and other dependencies]

- Add a non-root user
  sudo adduser --disabled-password cowrie
  su – cowrie

- Download the program:
  git clone http://github.com/cowrie/cowrie

- Create the virtual environment and install packages
  virtualenv --python=python3 cowrie-env
  pip install --upgrade pip
  pip install --upgrade -r requirements.txt
  Configure the Cowrie daemon:
  cp etc/cowrie.cfg.dist cowrie.cfg

- Change Hostname and enable telnet

```
# Hostname for the honeypot. Displayed by the shell prompt of the virtual
# environment
# (default: svr04)
hostname = testserver5

[telnet]
# Enable Telnet support, disabled by default
enabled = true
```

- Start the cowrie daemon by running the command:
  `bin/cowrie start`

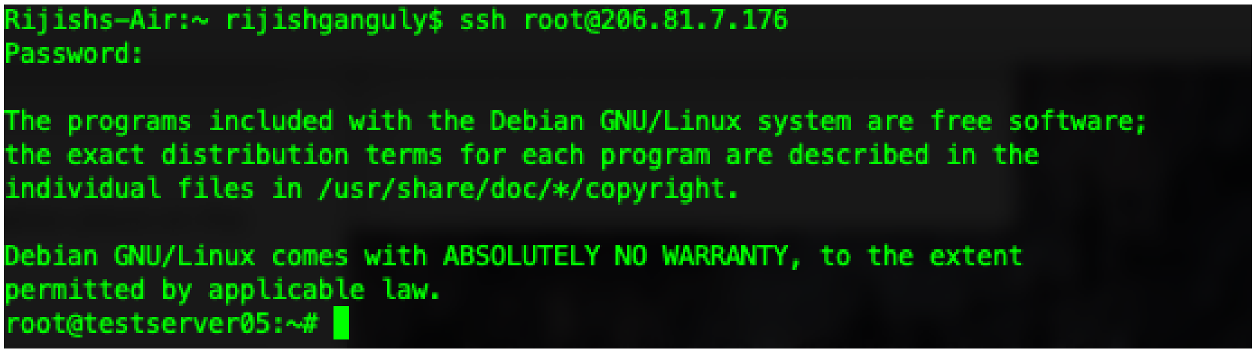
- Redirect traffic from 22 and 23 to 2222 and 2223 using iptables

  ```
  iptables -t nat -A PREROUTING -p tcp --dport 22 -j REDIRECT --to-port 2222
  iptables -t nat -A PREROUTING -p tcp --dport 23 -j REDIRECT --to-port 2223
  ```

Set up your VM with Cowrie listening on port 22 and the real SSH daemon listening on port 60022. Be sure your cloud's network settings allow access on both ports.

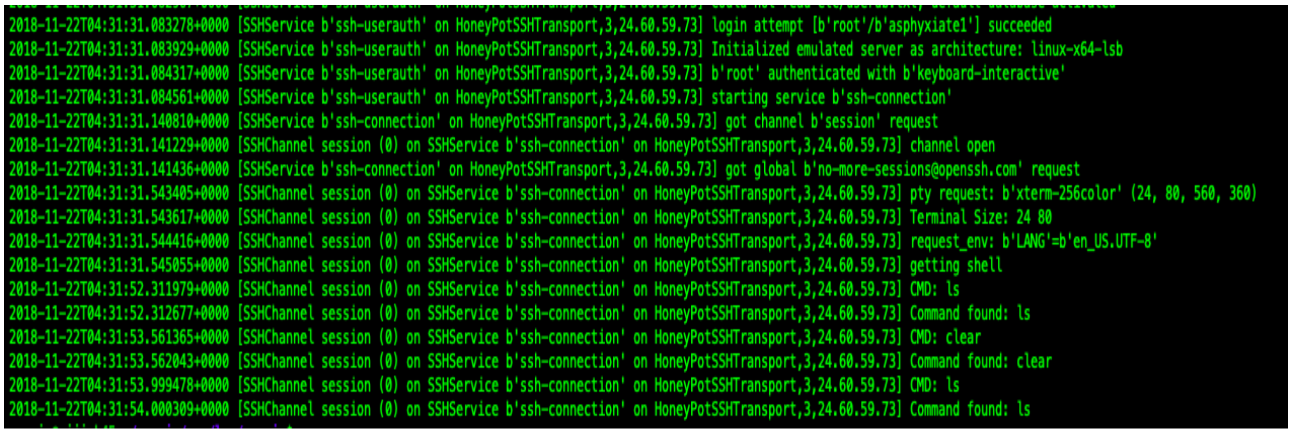
Login to your own honeypot and look around. Can you tell it's simulated?

No, I couldn't tell that it was simulated.



```
Rijishs-Air:~ rijishganguly$ ssh root@206.81.7.176
Password:

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
root@testserver05:~#
```

From outside the honeypot, show that you can find logs of your attempted “intrusion”.



```
2018-11-22T04:31:31.083278+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,3,24.60.59.73] login attempt [b'root'/b'asphyxiate1'] succeeded
2018-11-22T04:31:31.083929+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,3,24.60.59.73] Initialized emulated server as architecture: linux-x64-lsb
2018-11-22T04:31:31.084317+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,3,24.60.59.73] b'root' authenticated with b'keyboard-interactive'
2018-11-22T04:31:31.084561+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,3,24.60.59.73] starting service b'ssh-connection'
2018-11-22T04:31:31.140810+0000 [SSHService b'ssh-connection' on HoneyPotSSHTransport,3,24.60.59.73] got channel b'session' request
2018-11-22T04:31:31.141229+0000 [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,3,24.60.59.73] channel open
2018-11-22T04:31:31.141436+0000 [SSHService b'ssh-connection' on HoneyPotSSHTransport,3,24.60.59.73] got global b'no-more-sessions@openssh.com' request
2018-11-22T04:31:31.543405+0000 [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,3,24.60.59.73] pty request: b'xterm-256color' (24, 80, 560, 360)
2018-11-22T04:31:31.543617+0000 [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,3,24.60.59.73] Terminal Size: 24 80
2018-11-22T04:31:31.544416+0000 [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,3,24.60.59.73] request_env: b'LANG'=b'en_US.UTF-8'
2018-11-22T04:31:31.545055+0000 [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,3,24.60.59.73] getting shell
2018-11-22T04:31:52.311979+0000 [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,3,24.60.59.73] CMD: ls
2018-11-22T04:31:52.312677+0000 [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,3,24.60.59.73] Command found: ls
2018-11-22T04:31:53.561365+0000 [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,3,24.60.59.73] CMD: clear
2018-11-22T04:31:53.562043+0000 [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,3,24.60.59.73] Command found: clear
2018-11-22T04:31:53.999478+0000 [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,3,24.60.59.73] CMD: ls
2018-11-22T04:31:54.000309+0000 [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,3,24.60.59.73] Command found: ls
```

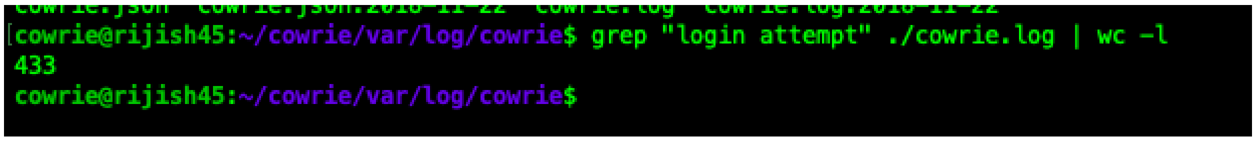
My public IP address – 24.60.59.73

Once you’re happy it’s set up well, let it run for at least ~3 days or until you get several attacks logged!

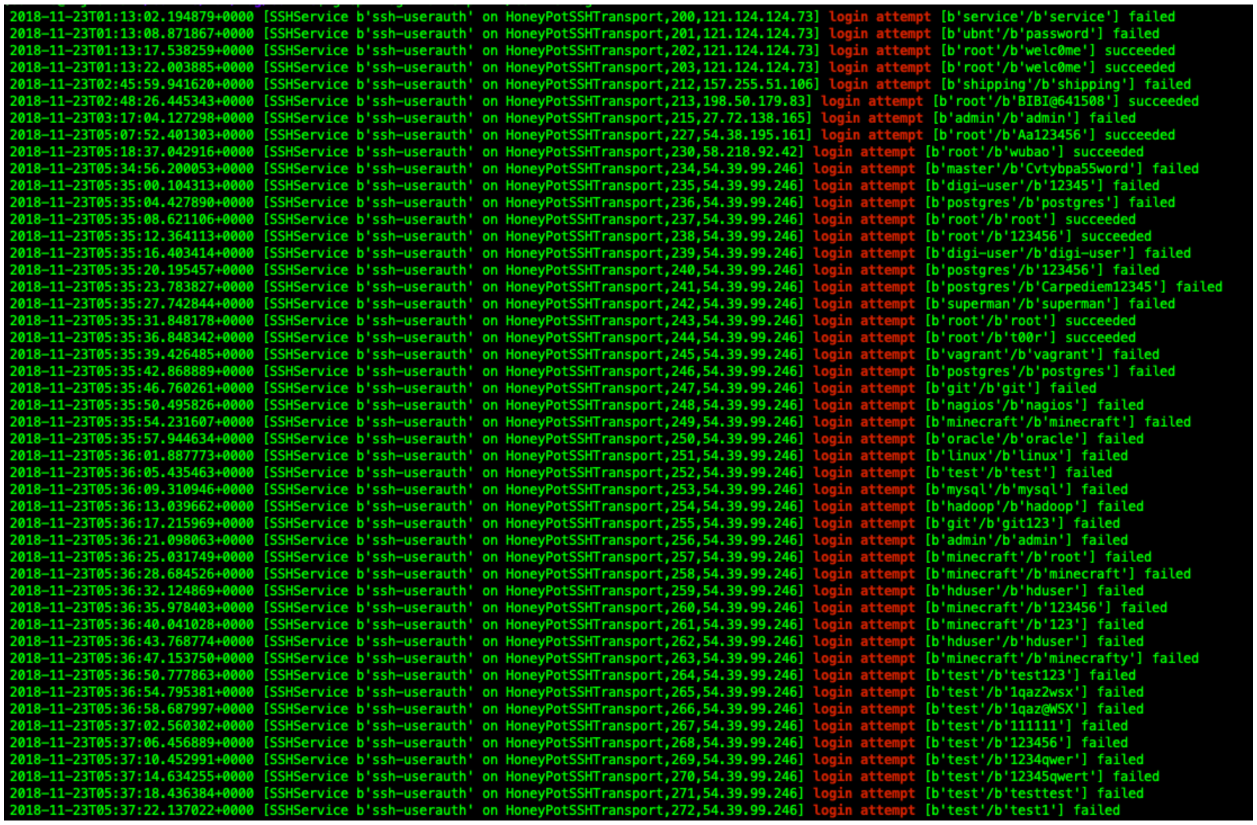
Results (4pts)

Take a look at what attackers did. How many attacks did you capture? What kinds of things did attackers tend to do?

I captured 433 login attempts.



```
cowrie.json  cowrie.json.2018-11-22  cowrie.log  cowrie.log.2018-11-22
cowrie@rijish45:~/cowrie/var/log/cowrie$ grep "login attempt" ./cowrie.log | wc -l
433
cowrie@rijish45:~/cowrie/var/log/cowrie$
```

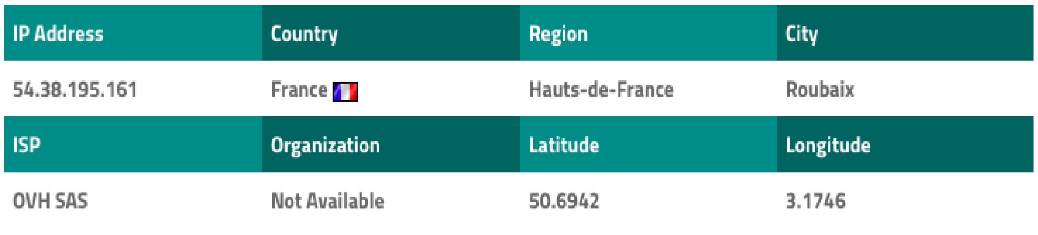


```
2018-11-23T01:13:02.194879+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,200,121.124.124.73] login attempt [b'service'/b'service'] failed
2018-11-23T01:13:08.871867+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,201,121.124.124.73] login attempt [b'ubnt'/b'password'] failed
2018-11-23T01:13:17.538259+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,202,121.124.124.73] login attempt [b'root'/b'welc0me'] succeeded
2018-11-23T01:13:22.003885+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,203,121.124.124.73] login attempt [b'root'/b'welc0me'] succeeded
2018-11-23T02:45:59.941620+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,212,157.255.51.106] login attempt [b'shipping'/b'shipping'] failed
2018-11-23T02:48:26.445343+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,213,198.50.179.83] login attempt [b'root'/b'BIBI@641508'] succeeded
2018-11-23T03:17:04.127298+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,215,27.72.138.165] login attempt [b'admin'/b'admin'] failed
2018-11-23T05:07:52.401303+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,227,54.38.195.161] login attempt [b'root'/b'Aa123456'] succeeded
2018-11-23T05:18:37.042916+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,230,58.218.92.42] login attempt [b'root'/b'wubao'] succeeded
2018-11-23T05:34:56.200053+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,234,54.39.99.246] login attempt [b'master'/b'Cvtybpa55word'] failed
2018-11-23T05:35:00.104313+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,235,54.39.99.246] login attempt [b'digi-user'/b'12345'] failed
2018-11-23T05:35:04.427890+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,236,54.39.99.246] login attempt [b'postgres'/b'postgres'] failed
2018-11-23T05:35:08.621106+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,237,54.39.99.246] login attempt [b'root'/b'root'] succeeded
2018-11-23T05:35:12.364113+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,238,54.39.99.246] login attempt [b'root'/b'123456'] succeeded
2018-11-23T05:35:16.403414+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,239,54.39.99.246] login attempt [b'digi-user'/b'digi-user'] failed
2018-11-23T05:35:20.195457+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,240,54.39.99.246] login attempt [b'postgres'/b'123456'] failed
2018-11-23T05:35:23.783827+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,241,54.39.99.246] login attempt [b'postgres'/b'Carpediem12345'] failed
2018-11-23T05:35:27.742844+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,242,54.39.99.246] login attempt [b'superman'/b'superman'] failed
2018-11-23T05:35:31.848178+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,243,54.39.99.246] login attempt [b'root'/b'root'] succeeded
2018-11-23T05:35:36.848342+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,244,54.39.99.246] login attempt [b'root'/b't00r'] succeeded
2018-11-23T05:35:39.426485+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,245,54.39.99.246] login attempt [b'vagrant'/b'vagrant'] failed
2018-11-23T05:35:42.868889+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,246,54.39.99.246] login attempt [b'postgres'/b'postgres'] failed
2018-11-23T05:35:46.760261+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,247,54.39.99.246] login attempt [b'git'/b'git'] failed
2018-11-23T05:35:50.495826+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,248,54.39.99.246] login attempt [b'nagios'/b'nagios'] failed
2018-11-23T05:35:54.231607+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,249,54.39.99.246] login attempt [b'minecraft'/b'minecraft'] failed
2018-11-23T05:35:57.944634+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,250,54.39.99.246] login attempt [b'oracle'/b'oracle'] failed
2018-11-23T05:36:01.887773+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,251,54.39.99.246] login attempt [b'linux'/b'linux'] failed
2018-11-23T05:36:05.435463+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,252,54.39.99.246] login attempt [b'test'/b'test'] failed
2018-11-23T05:36:09.310946+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,253,54.39.99.246] login attempt [b'mysql'/b'mysql'] failed
2018-11-23T05:36:13.039662+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,254,54.39.99.246] login attempt [b'hadoop'/b'hadoop'] failed
2018-11-23T05:36:17.215969+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,255,54.39.99.246] login attempt [b'git'/b'git123'] failed
2018-11-23T05:36:21.098063+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,256,54.39.99.246] login attempt [b'admin'/b'admin'] failed
2018-11-23T05:36:25.031749+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,257,54.39.99.246] login attempt [b'minecraft'/b'root'] failed
2018-11-23T05:36:28.684526+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,258,54.39.99.246] login attempt [b'minecraft'/b'minecraft'] failed
2018-11-23T05:36:32.124869+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,259,54.39.99.246] login attempt [b'hduser'/b'hduser'] failed
2018-11-23T05:36:35.978403+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,260,54.39.99.246] login attempt [b'minecraft'/b'123456'] failed
2018-11-23T05:36:40.041028+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,261,54.39.99.246] login attempt [b'minecraft'/b'123'] failed
2018-11-23T05:36:43.768774+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,262,54.39.99.246] login attempt [b'hduser'/b'hduser'] failed
2018-11-23T05:36:47.153750+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,263,54.39.99.246] login attempt [b'minecraft'/b'minecrafty'] failed
2018-11-23T05:36:50.777863+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,264,54.39.99.246] login attempt [b'test'/b'test123'] failed
2018-11-23T05:36:54.795381+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,265,54.39.99.246] login attempt [b'test'/b'1qaz2wsx'] failed
2018-11-23T05:36:58.687997+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,266,54.39.99.246] login attempt [b'test'/b'1qaz@WSX'] failed
2018-11-23T05:37:02.560302+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,267,54.39.99.246] login attempt [b'test'/b'111111'] failed
2018-11-23T05:37:06.456889+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,268,54.39.99.246] login attempt [b'test'/b'123456'] failed
2018-11-23T05:37:10.452991+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,269,54.39.99.246] login attempt [b'test'/b'1234qwer'] failed
2018-11-23T05:37:14.634255+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,270,54.39.99.246] login attempt [b'test'/b'12345qwert'] failed
2018-11-23T05:37:18.436384+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,271,54.39.99.246] login attempt [b'test'/b'testtest'] failed
2018-11-23T05:37:22.137022+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,272,54.39.99.246] login attempt [b'test'/b'test1'] failed
```

Pick a particular attack and describe:

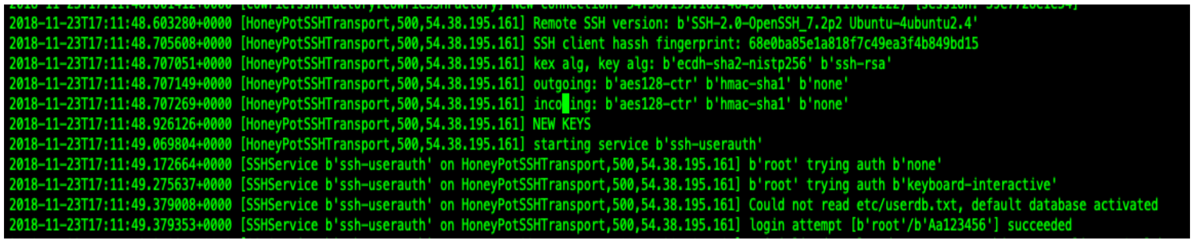
Particular successful attack from 54.38.195.161

- The network origin (IP address organization and geolocation)



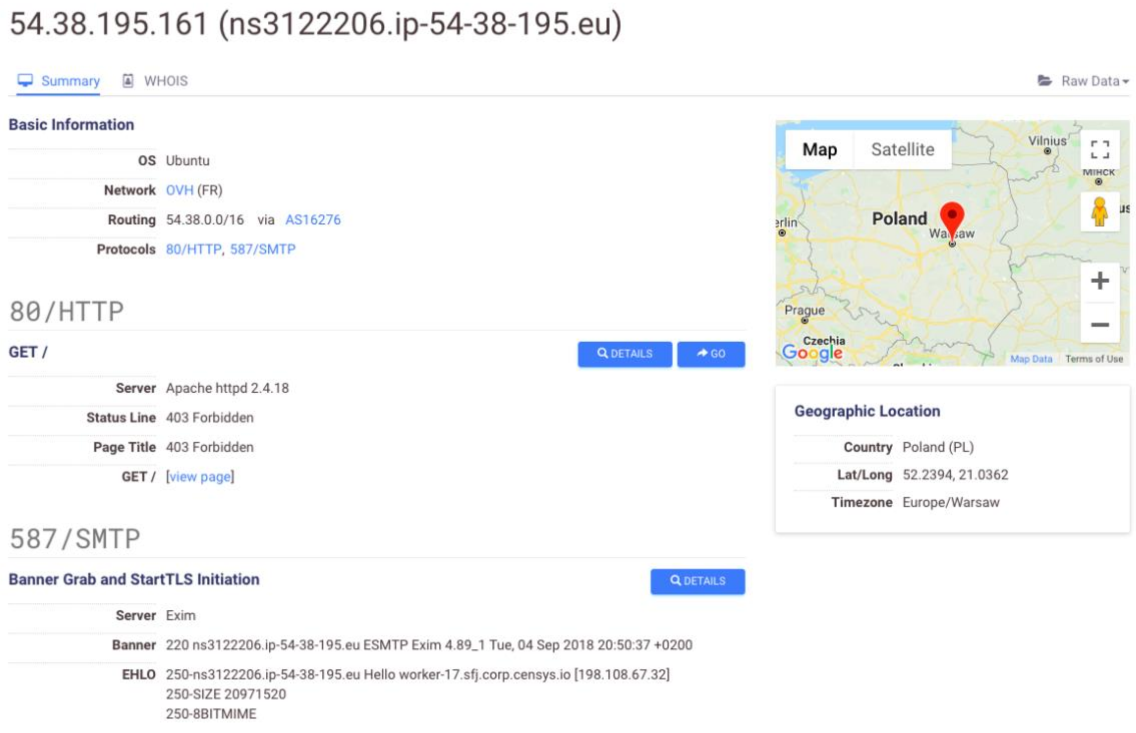
IP Address	Country	Region	City
54.38.195.161	France	Hauts-de-France	Roubaix
ISP	**Organization**	**Latitude**	**Longitude**
OVH SAS	Not Available	50.6942	3.1746

- The steps carried out



```
2018-11-23T17:11:48.603280+0000 [HoneyPotSSHTransport,500,54.38.195.161] Remote SSH version: b'SSH-2.0-OpenSSH_7.2p2 Ubuntu-4ubuntu2.4'
2018-11-23T17:11:48.705608+0000 [HoneyPotSSHTransport,500,54.38.195.161] SSH client hassh fingerprint: 68e0ba85e1a818f7c49ea3f4b849bd15
2018-11-23T17:11:48.707051+0000 [HoneyPotSSHTransport,500,54.38.195.161] kex alg, key alg: b'ecdh-sha2-nistp256' b'ssh-rsa'
2018-11-23T17:11:48.707149+0000 [HoneyPotSSHTransport,500,54.38.195.161] outgoing: b'aes128-ctr' b'hmac-sha1' b'none'
2018-11-23T17:11:48.707269+0000 [HoneyPotSSHTransport,500,54.38.195.161] incoming: b'aes128-ctr' b'hmac-sha1' b'none'
2018-11-23T17:11:48.926126+0000 [HoneyPotSSHTransport,500,54.38.195.161] NEW KEYS
2018-11-23T17:11:49.069804+0000 [HoneyPotSSHTransport,500,54.38.195.161] starting service b'ssh-userauth'
2018-11-23T17:11:49.172664+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,500,54.38.195.161] b'root' trying auth b'none'
2018-11-23T17:11:49.275637+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,500,54.38.195.161] b'root' trying auth b'keyboard-interactive'
2018-11-23T17:11:49.379008+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,500,54.38.195.161] Could not read etc/userdb.txt, default database activated
2018-11-23T17:11:49.379353+0000 [SSHService b'ssh-userauth' on HoneyPotSSHTransport,500,54.38.195.161] login attempt [b'root'/b'Aa123456'] succeeded
```

- Does it appear to be automated or manual?



54.38.195.161 (ns3122206.ip-54-38-195.eu)

Basic Information

OS	Ubuntu
Network	OVH (FR)
Routing	54.38.0.0/16 via AS16276
Protocols	80/HTTP, 587/SMTP

80/HTTP

GET /

Server	Apache httpd 2.4.18
Status Line	403 Forbidden
Page Title	403 Forbidden
GET /	[view page]

587/SMTP

Banner Grab and StartTLS Initiation

Server	Exim
Banner	220 ns3122206.ip-54-38-195.eu ESMTP Exim 4.89_1 Tue, 04 Sep 2018 20:50:37 +0200
EHLO	250-ns3122206.ip-54-38-195.eu Hello worker-17.sfj.corp.censys.io [198.108.67.32] 250-SIZE 20971520 250-8BITMIME

Geographic Location

Country	Poland (PL)
Lat/Long	52.2394, 21.0362
Timezone	Europe/Warsaw

The attack seemed to be manual

- What appears to have been the attacker's goal?

The intruder attempted to connect to the target IP on port 80 but the request was discarded several times.

- If they downloaded file(s), analyze these

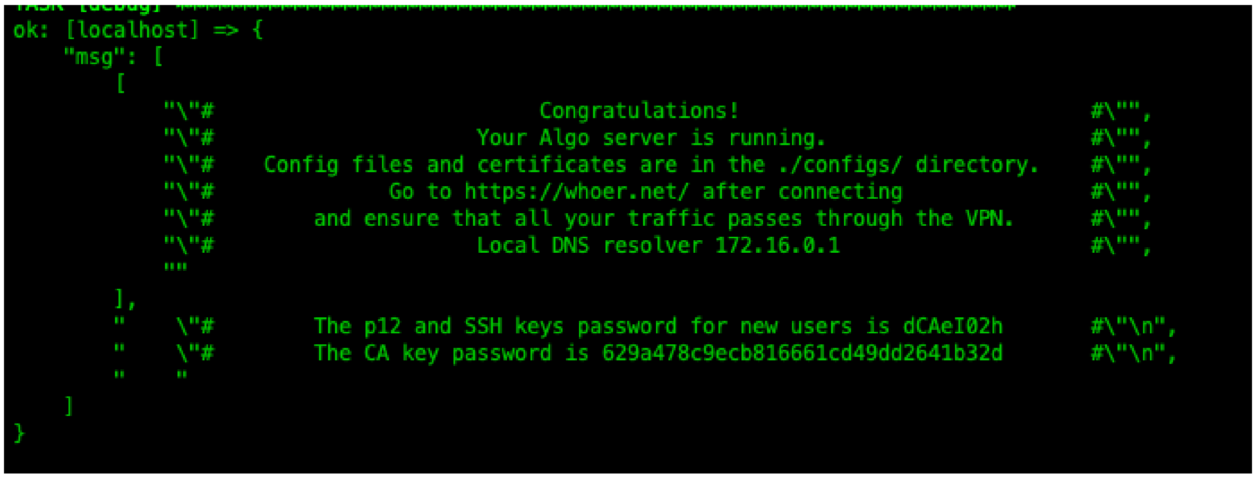
  No files were downloaded.

Question 9: VPN (6 points)

A Virtual Private Network (VPN) allows all network traffic to tunnel to an endpoint and be routed from there, and the tunnel usually encrypts the traffic. This allows a secure way to connect to otherwise private networks, and unlike SSH tunneling, it includes *all* network traffic generated by a client.

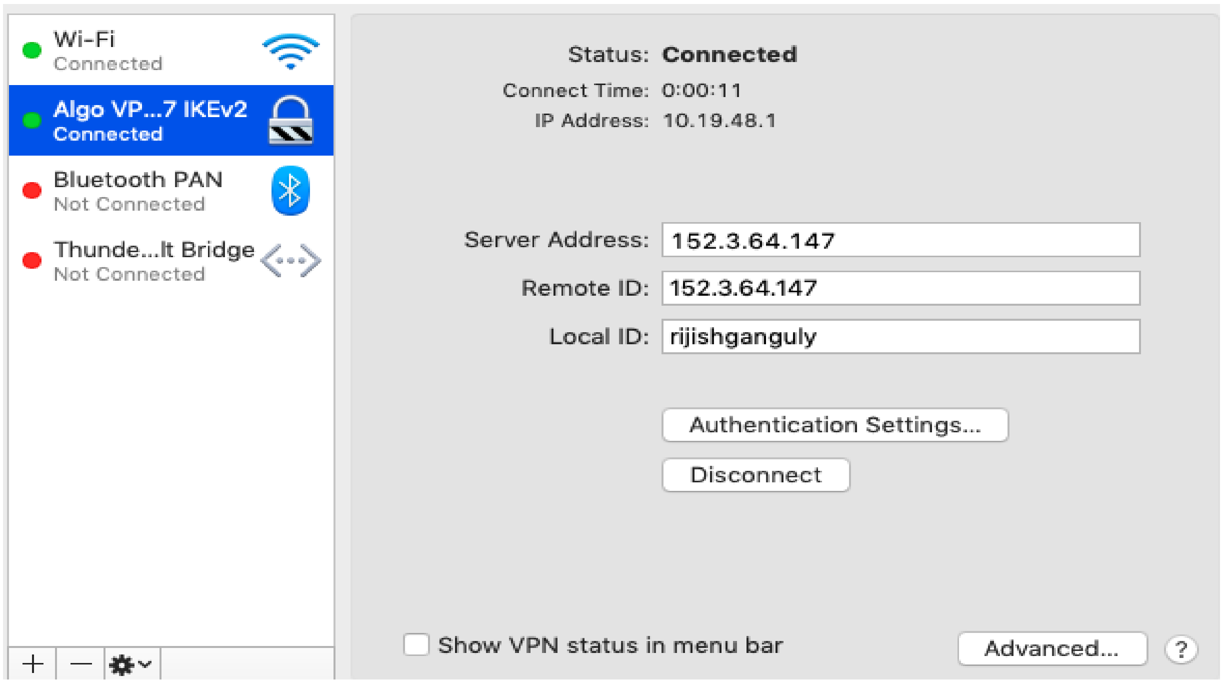
VPNs are very commonly deployed. However, there's a lot of cryptography setup and other configuration choices involved in setting up a VPN, and it's easy to do it in a way that subtly compromises security.

Algo is a set of scripts to automate deployment of a VPN target. Deploy Algo on your Linux VM (or to a cloud-based VM), then configure your personal computer to connect to it.



```
ok: [localhost] => {
    "msg": [
        [
            "\"#                          Congratulations!                          #\"",
            "\"#                     Your Algo server is running.                   #\"",
            "\"#    Config files and certificates are in the ./configs/ directory.  #\"",
            "\"#              Go to https://whoer.net/ after connecting             #\"",
            "\"#        and ensure that all your traffic passes through the VPN.    #\"",
            "\"#                     Local DNS resolver 172.16.0.1                  #\"",
            """
        ],
        "    \"#        The p12 and SSH keys password for new users is dCAeI02h       #\"\n",
        "    \"#        The CA key password is 629a478c9ecb816661cd49dd2641b32d       #\"\n",
        "    "
    ]
}
```

Show a screenshot of your personal computer's VPN client software showing the successful connection.



Wi-Fi
Connected

Algo VP...7 IKEv2
Connected

Bluetooth PAN
Not Connected

Thunde...lt Bridge
Not Connected

Status: **Connected**

Connect Time: 0:00:11

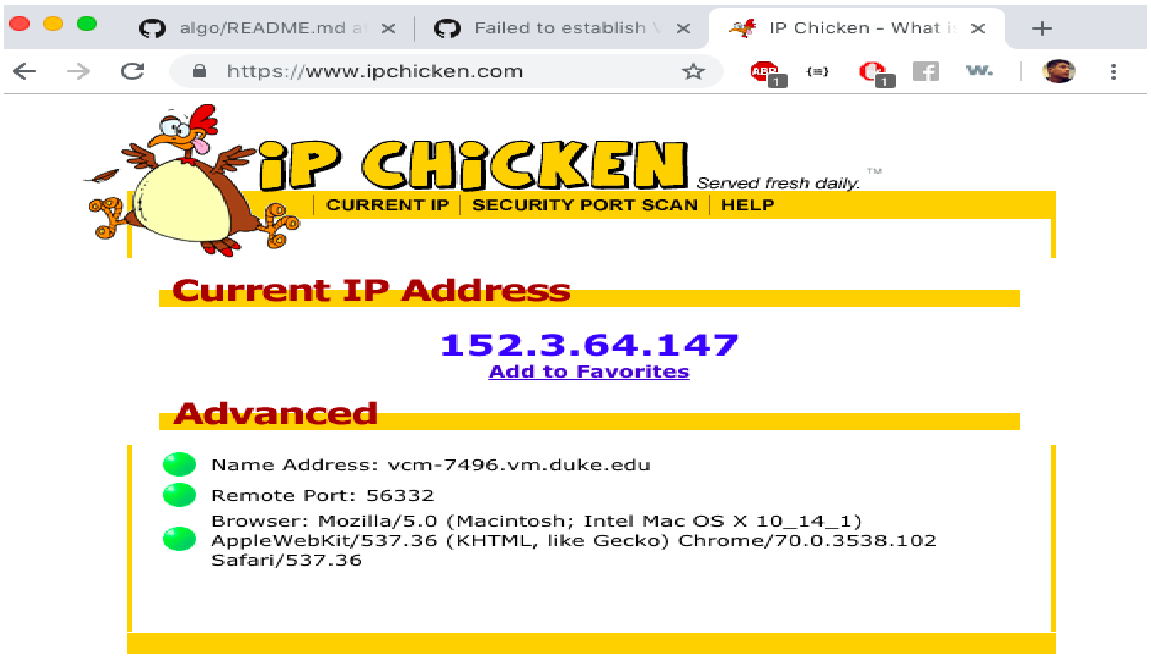
IP Address: 10.19.48.1

Server Address: 152.3.64.147

Remote ID: 152.3.64.147

Local ID: rijishganguly

Show a screenshot of a browser on your personal computer visiting an IP address identifier site like IPChicken.com.



Current IP Address

152.3.64.147
Add to Favorites

Advanced

Name Address: vcm-7496.vm.duke.edu
Remote Port: 56332
Browser: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.102 Safari/537.36

Question 10: Denial of Service attack using TorsHammer (6 points)

Perform the following steps on your Linux VM.

Download the torshammer.tgz from https://sourceforge.net/projects/torshammer/

To launch an attack, use the torshammer.py file and pass the necessary parameters to it.

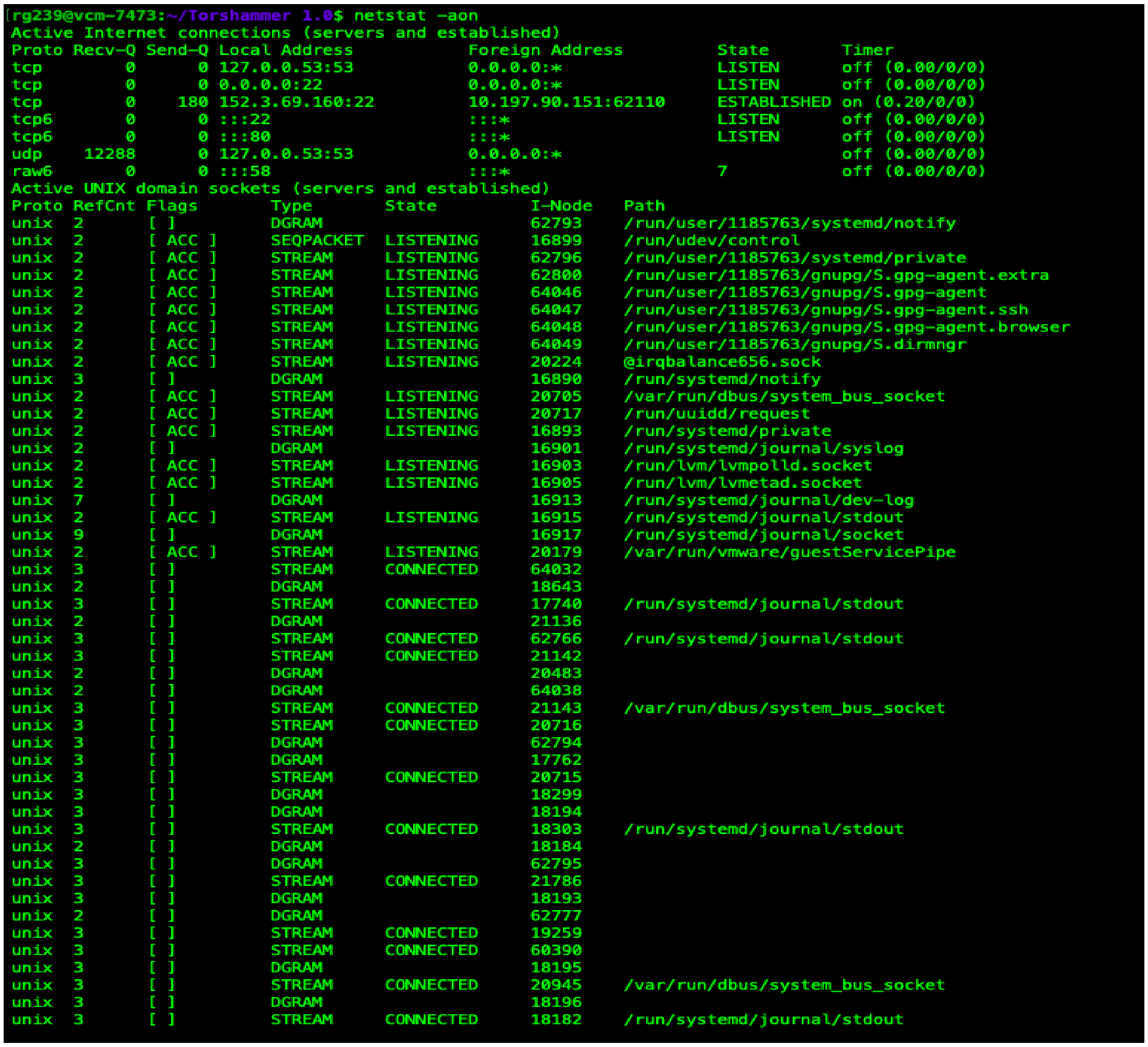
```
# ./torshammer.py
```

If you haven’t already, install the Apache web server on the VM. Attack yourself over the loopback adapter.

Show the output of netstat and ps aux before versus during an attack.

From another host, use the time and curl commands a few times roughly gauge the the latency of HTTP responses before, during, and after the attack.

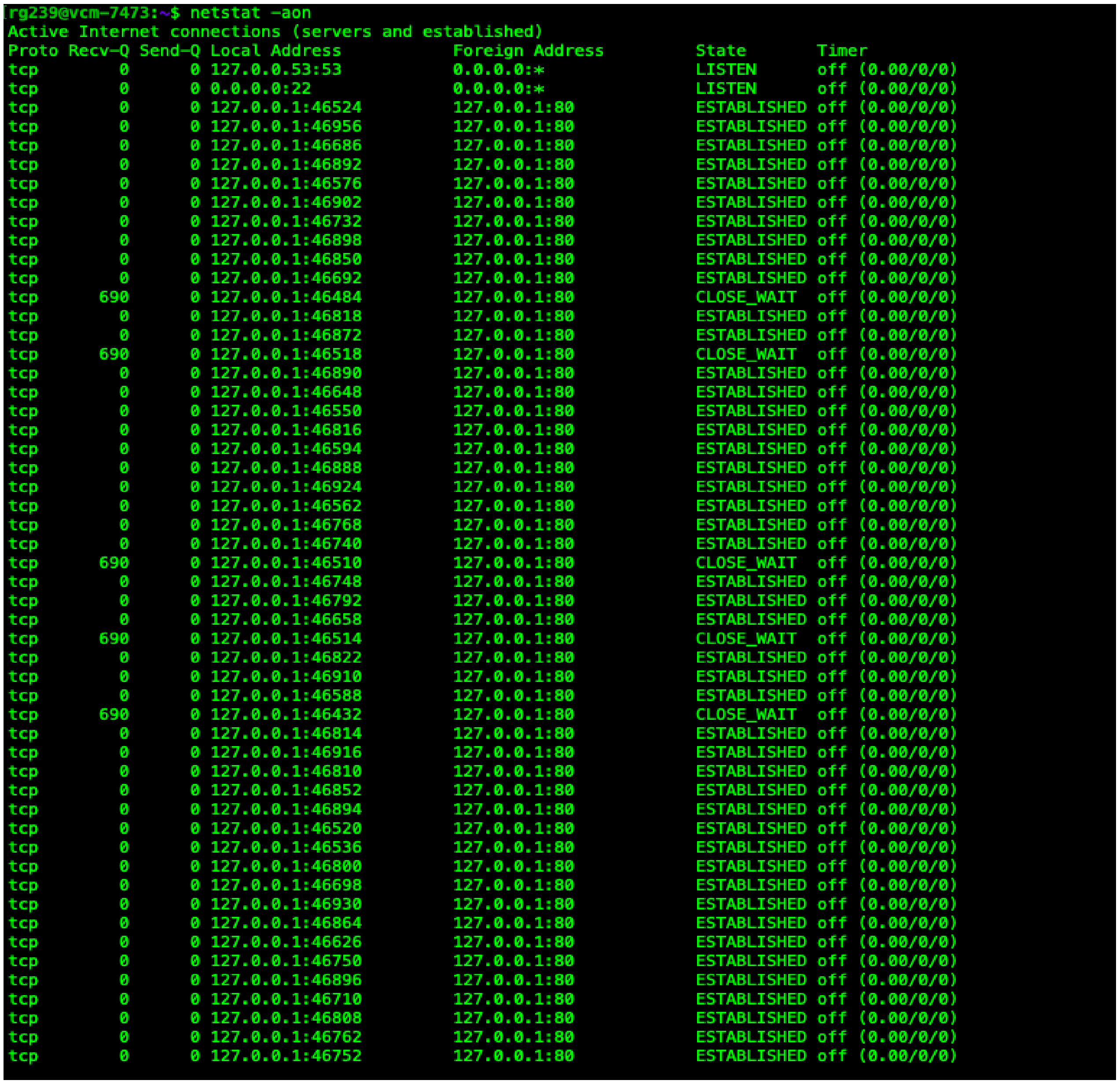
- Before attack:
  command**:**
  netstat -aon



```
rg239@vcm-7473:~/Torshammer 1.0$ netstat -aon
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       Timer
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN      off (0.00/0/0)
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      off (0.00/0/0)
tcp        0    180 152.3.69.160:22         10.197.90.151:62110     ESTABLISHED on (0.20/0/0)
tcp6       0      0 :::22                   :::*                    LISTEN      off (0.00/0/0)
tcp6       0      0 :::80                   :::*                    LISTEN      off (0.00/0/0)
udp    12288      0 127.0.0.53:53           0.0.0.0:*                           off (0.00/0/0)
raw6       0      0 :::58                   :::*                    7           off (0.00/0/0)
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node   Path
unix  2      [ ]         DGRAM                    62793    /run/user/1185763/systemd/notify
unix  2      [ ACC ]     SEQPACKET  LISTENING     16899    /run/udev/control
unix  2      [ ACC ]     STREAM     LISTENING     62796    /run/user/1185763/systemd/private
unix  2      [ ACC ]     STREAM     LISTENING     62800    /run/user/1185763/gnupg/S.gpg-agent.extra
unix  2      [ ACC ]     STREAM     LISTENING     64046    /run/user/1185763/gnupg/S.gpg-agent
unix  2      [ ACC ]     STREAM     LISTENING     64047    /run/user/1185763/gnupg/S.gpg-agent.ssh
unix  2      [ ACC ]     STREAM     LISTENING     64048    /run/user/1185763/gnupg/S.gpg-agent.browser
unix  2      [ ACC ]     STREAM     LISTENING     64049    /run/user/1185763/gnupg/S.dirmngr
unix  2      [ ACC ]     STREAM     LISTENING     20224    @irqbalance656.sock
unix  3      [ ]         DGRAM                    16890    /run/systemd/notify
unix  2      [ ACC ]     STREAM     LISTENING     20705    /var/run/dbus/system_bus_socket
unix  2      [ ACC ]     STREAM     LISTENING     20717    /run/uuidd/request
unix  2      [ ACC ]     STREAM     LISTENING     16893    /run/systemd/private
unix  2      [ ]         DGRAM                    16901    /run/systemd/journal/syslog
unix  2      [ ACC ]     STREAM     LISTENING     16903    /run/lvm/lvmpolld.socket
unix  2      [ ACC ]     STREAM     LISTENING     16905    /run/lvm/lvmetad.socket
unix  7      [ ]         DGRAM                    16913    /run/systemd/journal/dev-log
unix  2      [ ACC ]     STREAM     LISTENING     16915    /run/systemd/journal/stdout
unix  9      [ ]         DGRAM                    16917    /run/systemd/journal/socket
unix  2      [ ACC ]     STREAM     LISTENING     20179    /var/run/vmware/guestServicePipe
unix  3      [ ]         STREAM     CONNECTED     64032
unix  2      [ ]         DGRAM                    18643
unix  3      [ ]         STREAM     CONNECTED     17740    /run/systemd/journal/stdout
unix  2      [ ]         DGRAM                    21136
unix  3      [ ]         STREAM     CONNECTED     62766    /run/systemd/journal/stdout
unix  3      [ ]         STREAM     CONNECTED     21142
unix  2      [ ]         DGRAM                    20483
unix  2      [ ]         DGRAM                    64038
unix  3      [ ]         STREAM     CONNECTED     21143    /var/run/dbus/system_bus_socket
unix  3      [ ]         STREAM     CONNECTED     20716
unix  3      [ ]         DGRAM                    62794
unix  3      [ ]         DGRAM                    17762
unix  3      [ ]         STREAM     CONNECTED     20715
unix  3      [ ]         DGRAM                    18299
unix  3      [ ]         DGRAM                    18194
unix  3      [ ]         STREAM     CONNECTED     18303    /run/systemd/journal/stdout
unix  2      [ ]         DGRAM                    18184
unix  3      [ ]         DGRAM                    62795
unix  3      [ ]         STREAM     CONNECTED     21786
unix  3      [ ]         DGRAM                    18193
unix  2      [ ]         DGRAM                    62777
unix  3      [ ]         STREAM     CONNECTED     19259
unix  3      [ ]         STREAM     CONNECTED     60390
unix  3      [ ]         DGRAM                    18195
unix  3      [ ]         STREAM     CONNECTED     20945    /var/run/dbus/system_bus_socket
unix  3      [ ]         DGRAM                    18196
unix  3      [ ]         STREAM     CONNECTED     18182    /run/systemd/journal/stdout
```

- During the attack:

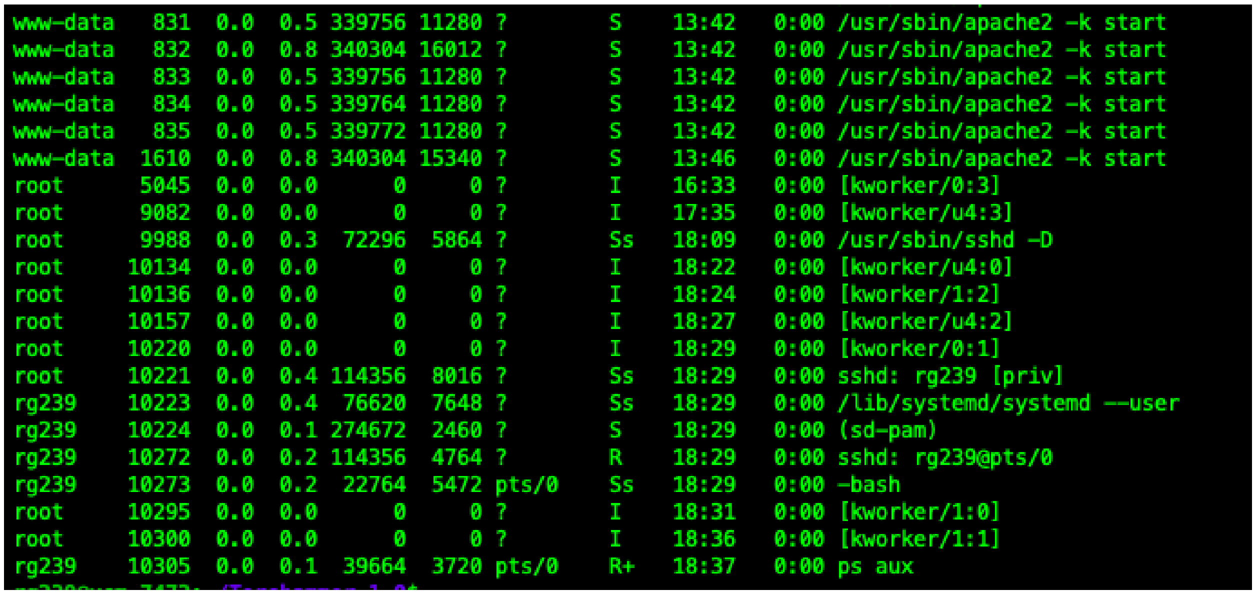
  command:
  netstat -aon



```
rg239@vcm-7473:~$ netstat -aon
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       Timer
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN      off (0.00/0/0)
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      off (0.00/0/0)
tcp        0      0 127.0.0.1:46524         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46956         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46686         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46892         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46576         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46902         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46732         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46898         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46850         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46692         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp      690      0 127.0.0.1:46484         127.0.0.1:80            CLOSE_WAIT  off (0.00/0/0)
tcp        0      0 127.0.0.1:46818         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46872         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp      690      0 127.0.0.1:46518         127.0.0.1:80            CLOSE_WAIT  off (0.00/0/0)
tcp        0      0 127.0.0.1:46890         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46648         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46550         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46816         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46594         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46888         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46924         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46562         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46768         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46740         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp      690      0 127.0.0.1:46510         127.0.0.1:80            CLOSE_WAIT  off (0.00/0/0)
tcp        0      0 127.0.0.1:46748         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46792         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46658         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp      690      0 127.0.0.1:46514         127.0.0.1:80            CLOSE_WAIT  off (0.00/0/0)
tcp        0      0 127.0.0.1:46822         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46910         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46588         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp      690      0 127.0.0.1:46432         127.0.0.1:80            CLOSE_WAIT  off (0.00/0/0)
tcp        0      0 127.0.0.1:46814         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46916         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46810         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46852         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46894         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46520         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46536         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46800         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46698         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46930         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46864         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46626         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46750         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46896         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46710         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46808         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46762         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
tcp        0      0 127.0.0.1:46752         127.0.0.1:80            ESTABLISHED off (0.00/0/0)
```

- Before attack:

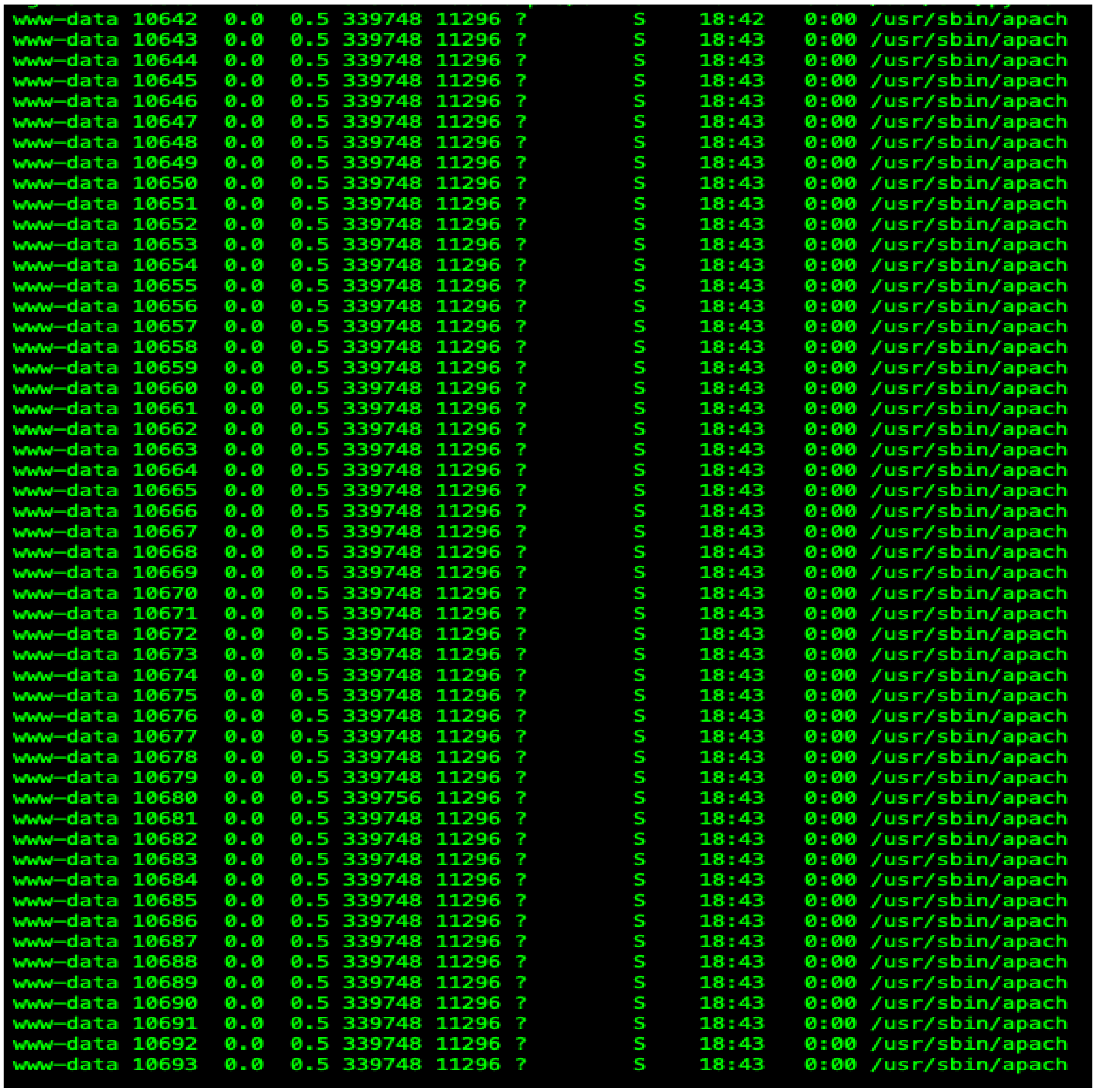
  command:
  ps aux



```
www-data   831  0.0  0.5 339756 11280 ?        S    13:42   0:00 /usr/sbin/apache2 -k start
www-data   832  0.0  0.8 340304 16012 ?        S    13:42   0:00 /usr/sbin/apache2 -k start
www-data   833  0.0  0.5 339756 11280 ?        S    13:42   0:00 /usr/sbin/apache2 -k start
www-data   834  0.0  0.5 339764 11280 ?        S    13:42   0:00 /usr/sbin/apache2 -k start
www-data   835  0.0  0.5 339772 11280 ?        S    13:42   0:00 /usr/sbin/apache2 -k start
www-data  1610  0.0  0.8 340304 15340 ?        S    13:46   0:00 /usr/sbin/apache2 -k start
root      5045  0.0  0.0      0     0 ?        I    16:33   0:00 [kworker/0:3]
root      9082  0.0  0.0      0     0 ?        I    17:35   0:00 [kworker/u4:3]
root      9988  0.0  0.3  72296  5864 ?        Ss   18:09   0:00 /usr/sbin/sshd -D
root     10134  0.0  0.0      0     0 ?        I    18:22   0:00 [kworker/u4:0]
root     10136  0.0  0.0      0     0 ?        I    18:24   0:00 [kworker/1:2]
root     10157  0.0  0.0      0     0 ?        I    18:27   0:00 [kworker/u4:2]
root     10220  0.0  0.0      0     0 ?        I    18:29   0:00 [kworker/0:1]
root     10221  0.0  0.4 114356  8016 ?        Ss   18:29   0:00 sshd: rg239 [priv]
rg239    10223  0.0  0.4  76620  7648 ?        Ss   18:29   0:00 /lib/systemd/systemd --user
rg239    10224  0.0  0.1 274672  2460 ?        S    18:29   0:00 (sd-pam)
rg239    10272  0.0  0.2 114356  4764 ?        R    18:29   0:00 sshd: rg239@pts/0
rg239    10273  0.0  0.2  22764  5472 pts/0    Ss   18:29   0:00 -bash
root     10295  0.0  0.0      0     0 ?        I    18:31   0:00 [kworker/1:0]
root     10300  0.0  0.0      0     0 ?        I    18:36   0:00 [kworker/1:1]
rg239    10305  0.0  0.1  39664  3720 pts/0    R+   18:37   0:00 ps aux
```

- After Attack:

    command:
    ps aux



```
www-data 10642  0.0  0.5 339748 11296 ?        S    18:42   0:00 /usr/sbin/apach
www-data 10643  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10644  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10645  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10646  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10647  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10648  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10649  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10650  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10651  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10652  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10653  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10654  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10655  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10656  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10657  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10658  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10659  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10660  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10661  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10662  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10663  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10664  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10665  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10666  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10667  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10668  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10669  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10670  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10671  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10672  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10673  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10674  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10675  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10676  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10677  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10678  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10679  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10680  0.0  0.5 339756 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10681  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10682  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10683  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10684  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10685  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10686  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10687  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10688  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10689  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10690  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10691  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10692  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
www-data 10693  0.0  0.5 339748 11296 ?        S    18:43   0:00 /usr/sbin/apach
```

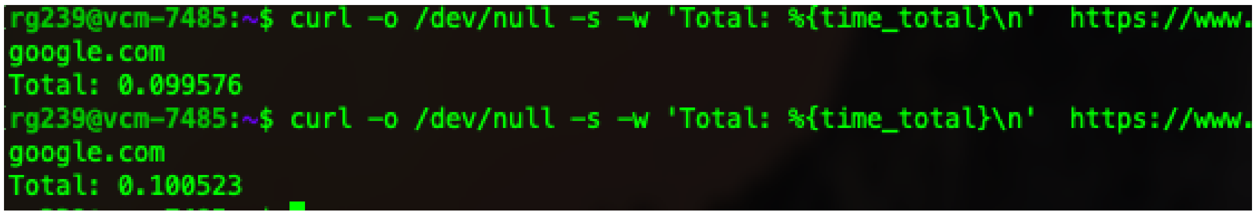
<u>Checking the latency:</u>

Using the command

curl -o /dev/null -s -w 'Total: %{time_total}\n' https://www.google.com

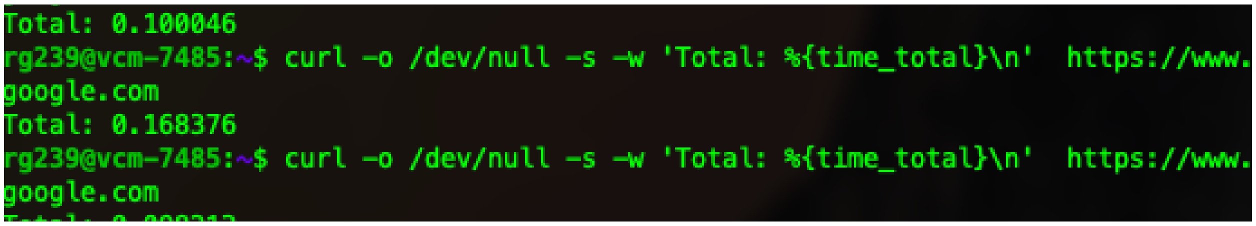
We can measure the response time of the curl request

Before attack:



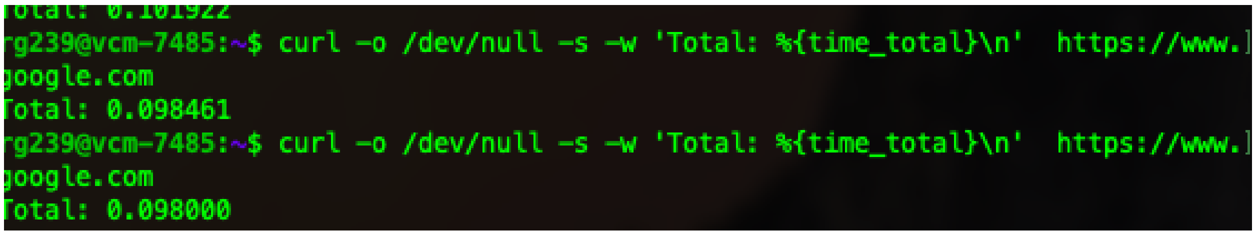
```
rg239@vcm-7485:~$ curl -o /dev/null -s -w 'Total: %{time_total}\n'  https://www.
google.com
Total: 0.099576
rg239@vcm-7485:~$ curl -o /dev/null -s -w 'Total: %{time_total}\n'  https://www.
google.com
Total: 0.100523
```

During attack:



```
Total: 0.100046
rg239@vcm-7485:~$ curl -o /dev/null -s -w 'Total: %{time_total}\n'  https://www.
google.com
Total: 0.168376
rg239@vcm-7485:~$ curl -o /dev/null -s -w 'Total: %{time_total}\n'  https://www.
google.com
```

After attack:



```
otal: 0.101922
g239@vcm-7485:~$ curl -o /dev/null -s -w 'Total: %{time_total}\n'  https://www.]
oogle.com
otal: 0.098461
g239@vcm-7485:~$ curl -o /dev/null -s -w 'Total: %{time_total}\n'  https://www.]
oogle.com
otal: 0.098000
```

Resource used: https://stackoverflow.com/questions/18215389/how-do-i-measure-request-and-response-times-at-once-using-curl

Question 11: Reading some security literature (5 points)

I've made a few references to the hacking journal PoC||GTFO. This journal does a fantastic job of presenting concrete, real-world feats of security engineering (and you are welcome to either embrace or ignore the church parody overtones). Check it out and pick a substantive article (i.e., not one of the sermons, one-pagers, historical ads, poems, etc.). Write a one paragraph summary and a one paragraph reflection on how the work relates to concepts we've learned in class (both foundational, e.g. the CIA model, and technical, e.g. cryptography).

Note: if you like this sort of thing, they sell a gilded, leather-bound print edition:
volume 1 and volume 2.

Proceedings of the Society of PoC||GTFO

Burning a Phone
Josh Thomas

In this article, the author delves into destroying phones beyond repair using software tricks. The attacks are focused on the arch/arm/mach-msm subsystem and breaking the commonplace NAND Flash and SD Card hardware components. The Linux power regulation framework defines voltage parameters for specific hardware connected to the PCB and the framework regulates PMIC and other control devices to ensure specific hardware is given the correct voltages. The framework directly interacts with both the kernel and the physical PCB. The PCB has some protections against voltage manipulations. Moreover, the kernel, has a robust framework to detect thermal issues ensuring device safety. On the Sony Xperia Z platform, all NAND Flash and all SDCard interactions are controlled by the Qualcomm MSM 7X00A SDCC hardware. By implementing a patch to the kernel, we can increase the voltage supply to the 7X00A from 2.95V to 5.9V. By doing this, we can ensure that voltage pushed to the NAND or SD-Card during read / write operations is well above the defined specification. The internal battery cannot actually deliver 5.9 V. However, the PMIC will try to and what we get as an end result is NAND Flash chip that corrupts nearly every block of storage it attempts to write or read.

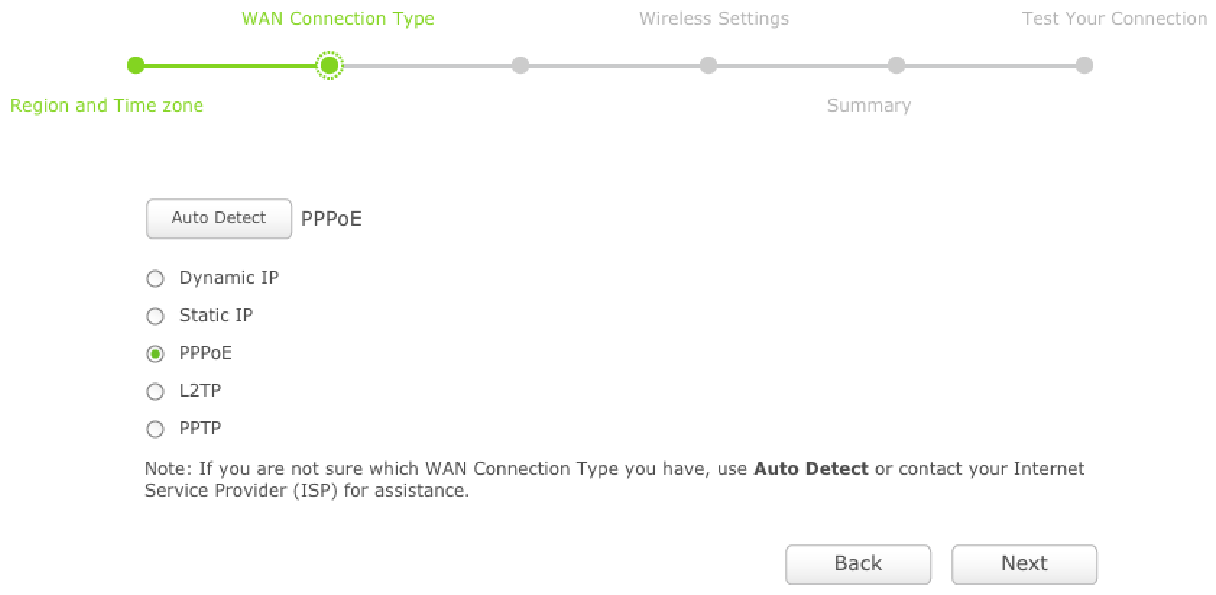
According to the CIA triad, System Integrity is defined as the quality that a system has when it performs its intended function in an unimpaired manner, free from unauthorized manipulation of the system and Availability is defined as ensuring timely and reliable access to and use of information. The above attack violates the CIA triad by compromising Availability and System Integrity. The asset here is the Sony Xperia Z. With two small values changed in the code base of the kernel this attack ensures that all persistent data is basically unusable and untrustworthy. There is no possible way to recover from this attack as rebooting makes the situation much worse by corrupting large swaths of the codebase on disk during the read operation.

Question 12: Wireless Security (5 points)

Router hardening

Router manufacturer TP-Link has web-based simulators for various model's web interfaces. This link simulates the "Archer C9" model wireless router. This is a real device and can be purchased here on Amazon.

Using the simulator, show and describe in detail 6 different configuration changes you would make to securely set up and harden this type of wireless router. Be sure to describe any relevant assumptions you may be making about the environment in which the device is being set up.



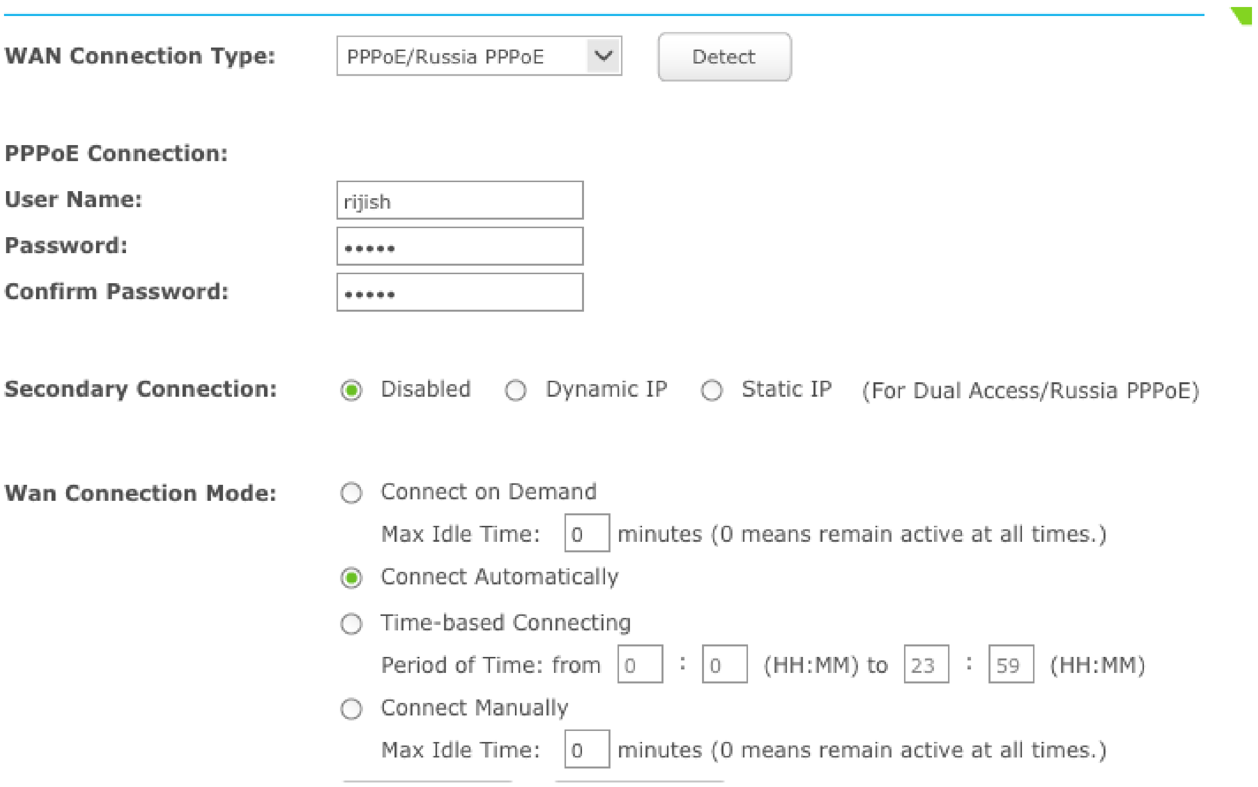
PPPoE

( ) Dynamic IP
( ) Static IP
[✓ PPPoE]
( ) L2TP
( ) PPTP

Note: If you are not sure which WAN Connection Type you have, use **Auto Detect** or contact your Internet Service Provider (ISP) for assistance.

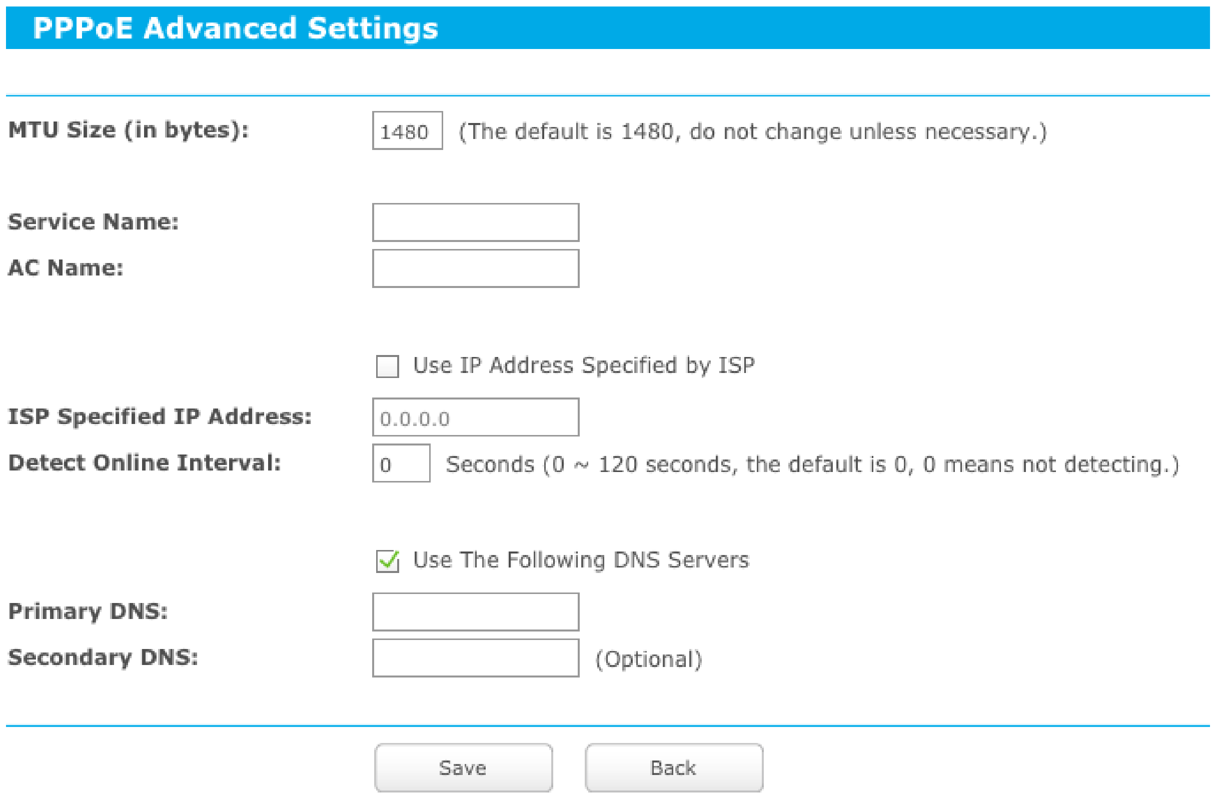
WAN Connection Type – PPPoE

Setting up the WAN connection type as PPPoE. Point-to-Point Protocol over Ethernet is a specification for connecting multiple computer users on an Ethernet local area network to a remote site through common customer premises equipment. The PPP protocol information is encapsulated within an Ethernet frame.



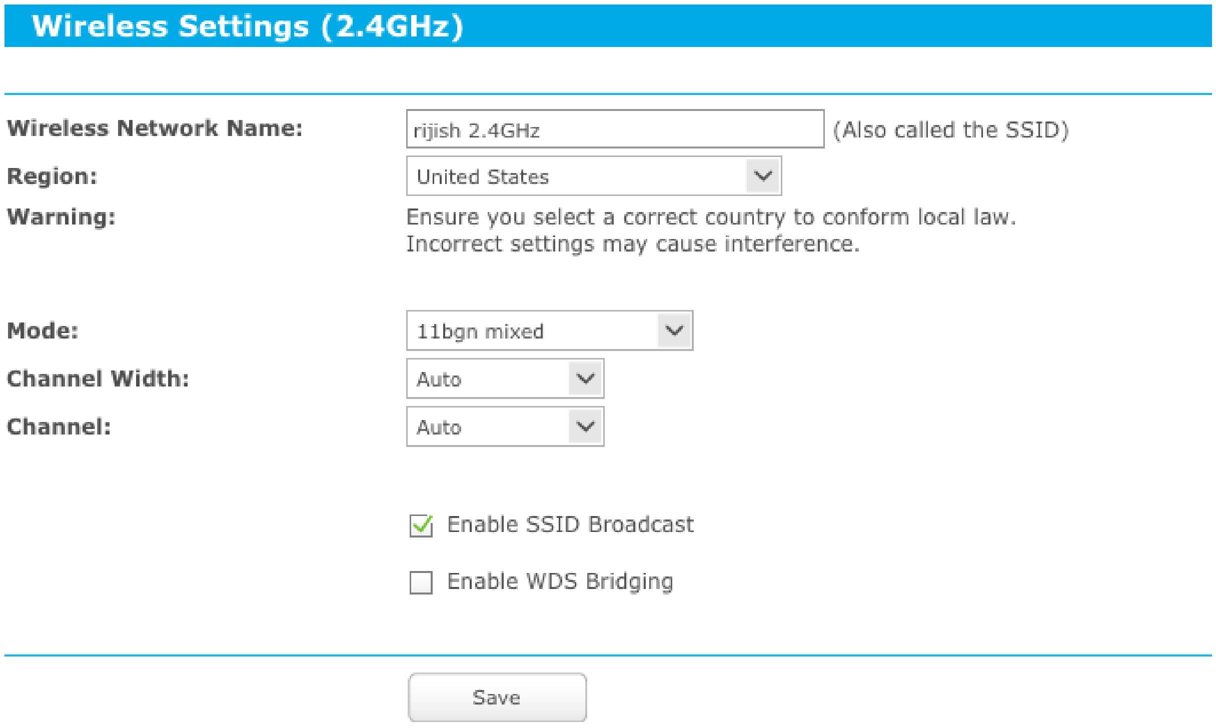
WAN Connection Type:	PPPoE/Russia PPPoE ▾ Detect
PPPoE Connection:	
User Name:	rijish
Password:	•••••
Confirm Password:	•••••
Secondary Connection:	[✓ Disabled] Dynamic IP Static IP (For Dual Access/Russia PPPoE)
Wan Connection Mode:	Connect on Demand
	Max Idle Time: 0 minutes (0 means remain active at all times.)
	[✓ Connect Automatically]
	Time-based Connecting
	Period of Time: from 0 : 0 (HH:MM) to 23 : 59 (HH:MM)
	Connect Manually
	Max Idle Time: 0 minutes (0 means remain active at all times.)

Setting up a password and WAN Connection Mode



MTU Size (in bytes):	1480 (The default is 1480, do not change unless necessary.)
Service Name:	
AC Name:	
	☐ Use IP Address Specified by ISP
ISP Specified IP Address:	0.0.0.0
Detect Online Interval:	0 Seconds (0 ~ 120 seconds, the default is 0, 0 means not detecting.)
	✓ Use The Following DNS Servers
Primary DNS:	
Secondary DNS:	(Optional)

PPPoE Advanced Settings

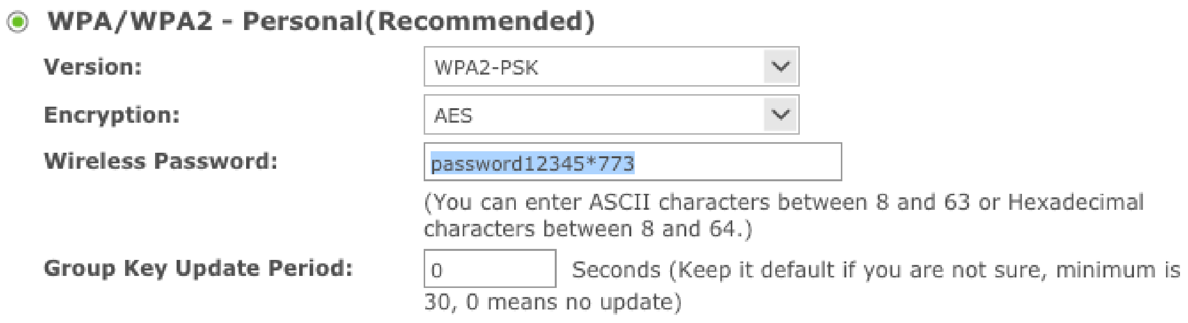

Wireless Settings (2.4GHz)

Wireless Network Name:	rijish 2.4GHz (Also called the SSID)
Region:	United States
Warning:	Ensure you select a correct country to conform local law. Incorrect settings may cause interference.
Mode:	11bgn mixed
Channel Width:	Auto
Channel:	Auto

✓ Enable SSID Broadcast

☐ Enable WDS Bridging

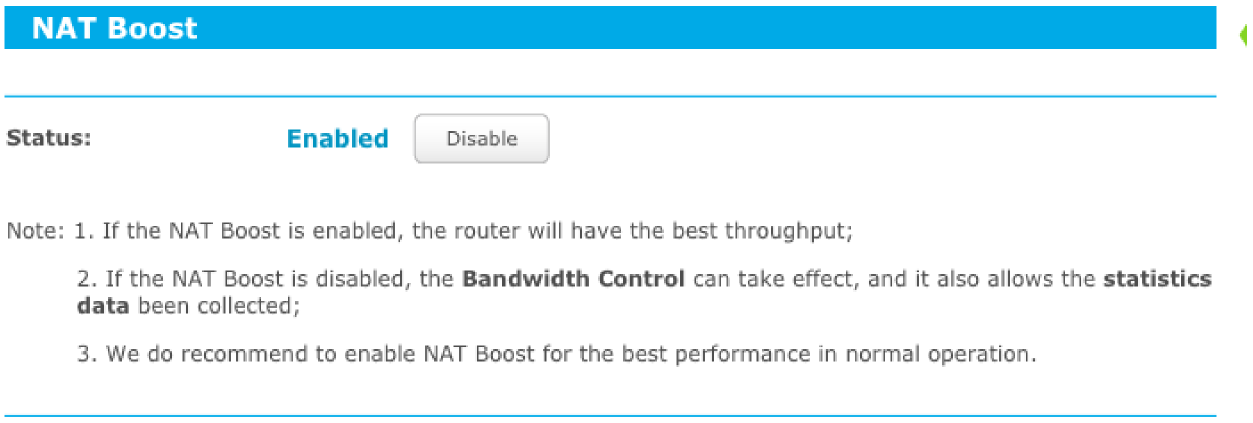
Wireless Settings 2.4Ghz


● WPA/WPA2 - Personal(Recommended)

Version:	WPA2-PSK
Encryption:	AES
Wireless Password:	password12345*773
	(You can enter ASCII characters between 8 and 63 or Hexadecimal characters between 8 and 64.)
Group Key Update Period:	0 Seconds (Keep it default if you are not sure, minimum is 30, 0 means no update)

WPA/WPA2 – Personal

WPA2 is currently the most secure choice. I chose AES encryption as it’s a worldwide encryption standard that’s even been adopted by the US government.


NAT Boost

Status: Enabled [Disable]

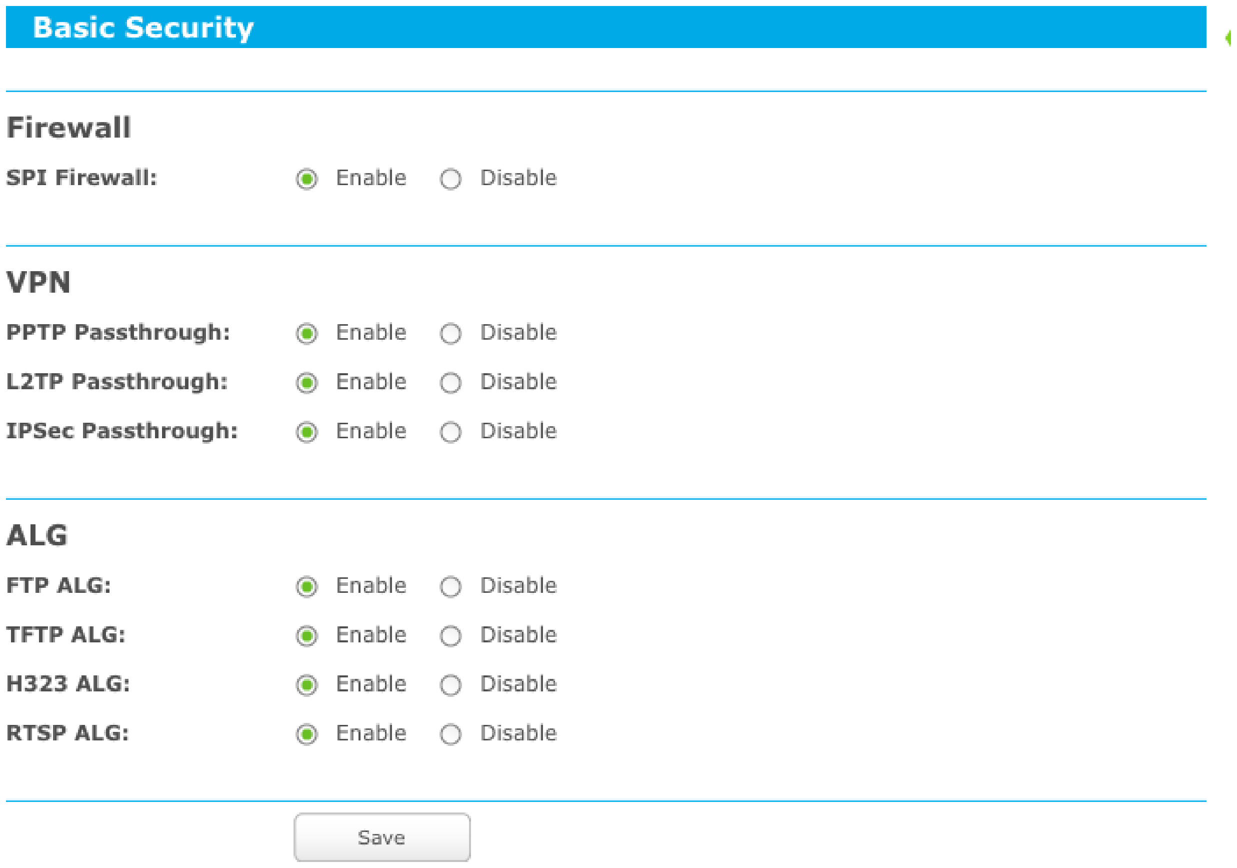
Note: 1. If the NAT Boost is enabled, the router will have the best throughput;

2. If the NAT Boost is disabled, the **Bandwidth Control** can take effect, and it also allows the **statistics data** been collected;

3. We do recommend to enable NAT Boost for the best performance in normal operation.

Enabling NAT boost

The NAT boost enables the router to have the best throughput.


Basic Security

Firewall

SPI Firewall:	[✓ Enable] Disable

VPN

PPTP Passthrough:	[✓ Enable] Disable
L2TP Passthrough:	[✓ Enable] Disable
IPSec Passthrough:	[✓ Enable] Disable

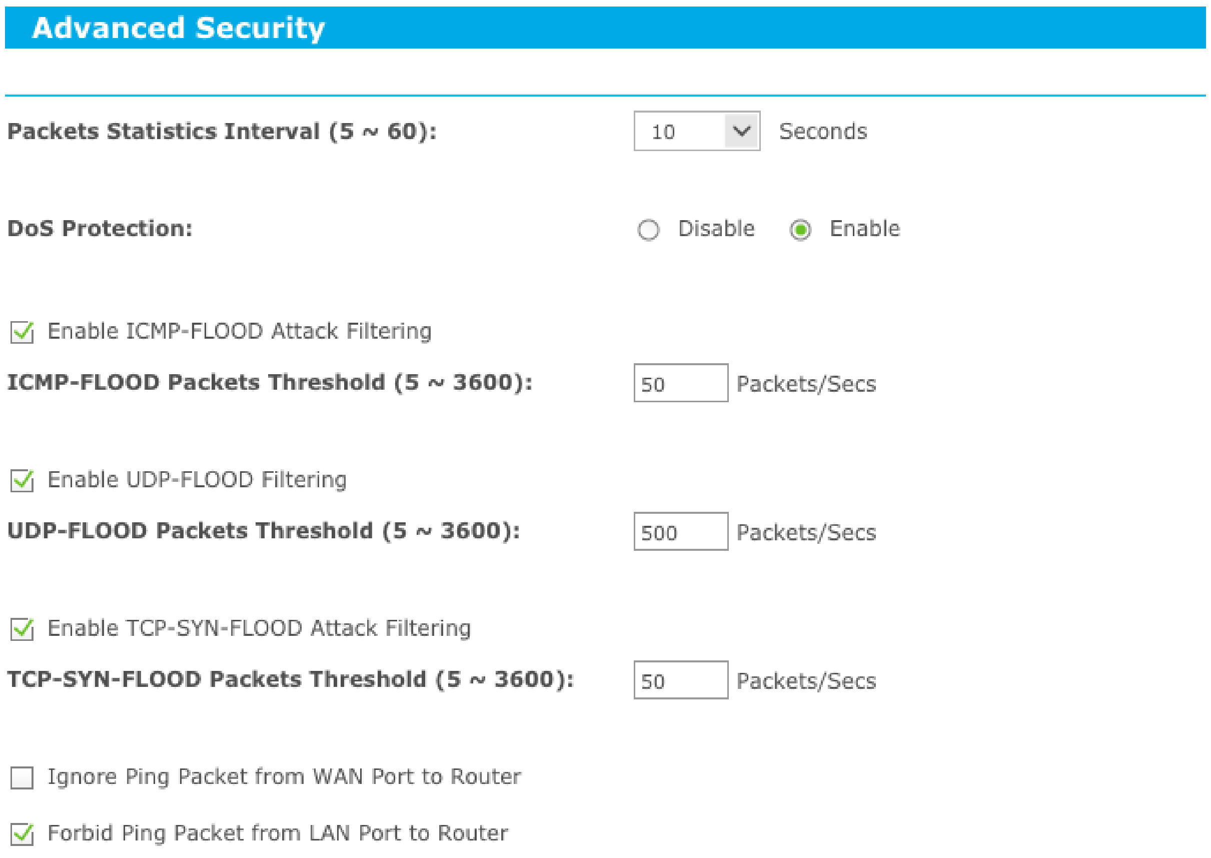
ALG

FTP ALG:	[✓ Enable] Disable
TFTP ALG:	[✓ Enable] Disable
H323 ALG:	[✓ Enable] Disable
RTSP ALG:	[✓ Enable] Disable

[Save]

Enabling Firewall

A stateful firewall keeps track of the state of network connections (such as TCP streams or UDP communication) and is able to hold significant attributes of each connection in memory. These attributes are collectively known as the state of the connection and may include such details as the IP addresses and ports involved in the connection and the sequence numbers of the packets traversing the connection.



Advanced Security

Packets Statistics Interval (5 ~ 60):	10 Seconds
DoS Protection:	Disable [✓ Enable]

✓ Enable ICMP-FLOOD Attack Filtering

ICMP-FLOOD Packets Threshold (5 ~ 3600): 50 Packets/Secs

✓ Enable UDP-FLOOD Filtering

UDP-FLOOD Packets Threshold (5 ~ 3600): 500 Packets/Secs

✓ Enable TCP-SYN-FLOOD Attack Filtering

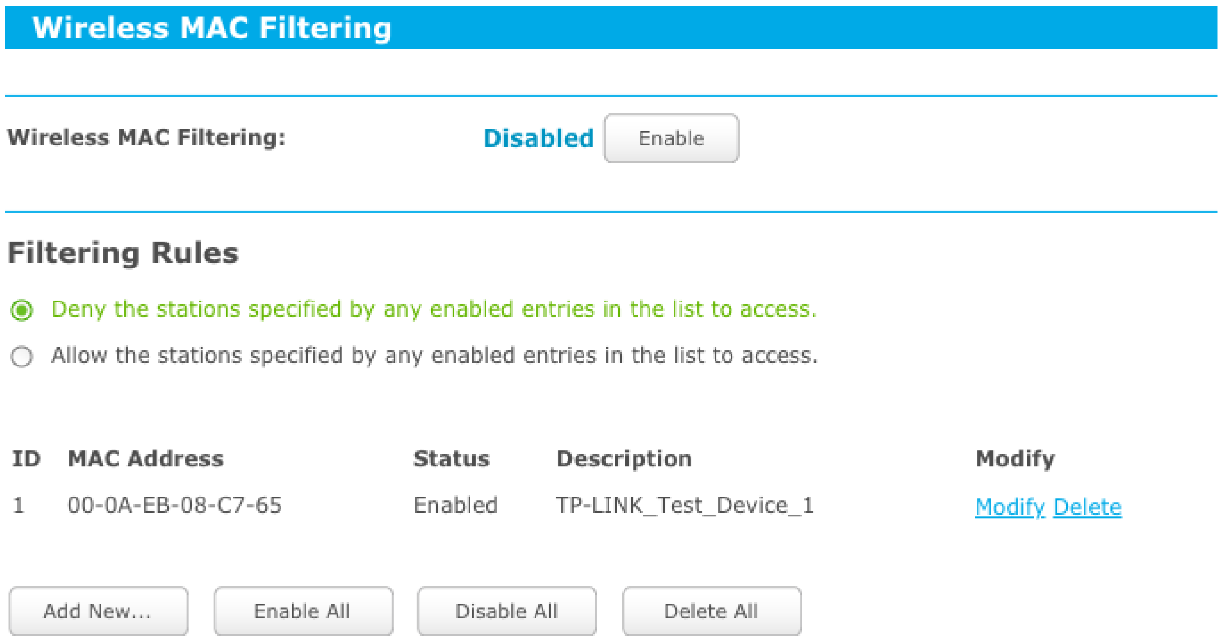
TCP-SYN-FLOOD Packets Threshold (5 ~ 3600): 50 Packets/Secs

☐ Ignore Ping Packet from WAN Port to Router

✓ Forbid Ping Packet from LAN Port to Router

Prevent **DoS** attacks by enabling:

1. ICMP-FLOOD Attack Filtering
2. UDO-FLOOD Filtering
3. TCP-SYN-FLOOD Attack Filtering


Wireless MAC Filtering

Wireless MAC Filtering: **Disabled** Enable

Filtering Rules

[✓ Deny the stations specified by any enabled entries in the list to access.]

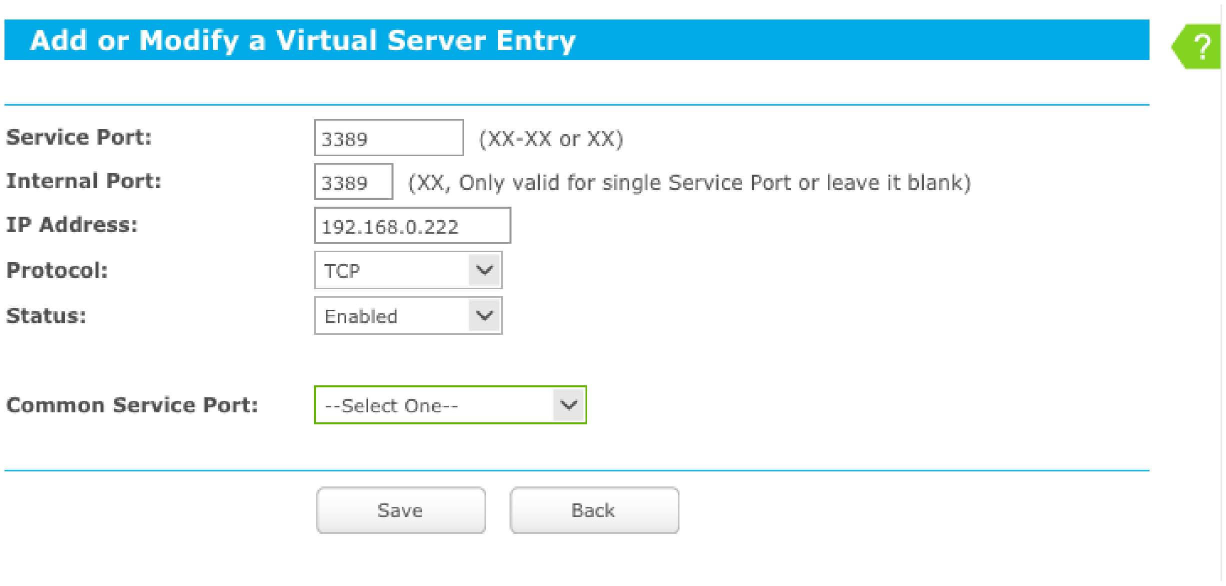
Allow the stations specified by any enabled entries in the list to access.

ID	MAC Address	Status	Description	Modify
1	00-0A-EB-08-C7-65	Enabled	TP-LINK_Test_Device_1	Modify Delete

Enabling Wireless MAC Filtering lets us define a list of devices and only these devices can connect on our Wi-Fi network. We configure a list of allowed MAC addresses.

Additional configuration options

Suppose someone outside the wireless network (WAN) needs to access to a Windows machine (192.168.0.222) inside the wireless network (WLAN) via RDP? What configuration could allow this? Show a screenshot of adding such a configuration. (Note: this simulator won't save settings, so just show the dialog where you've input the settings before clicking 'Save'.)



Add or Modify a Virtual Server Entry

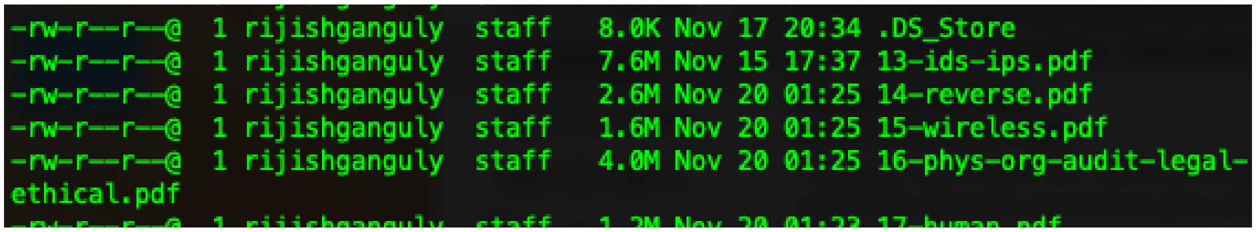
Service Port:	3389 (XX-XX or XX)
Internal Port:	3389 (XX, Only valid for single Service Port or leave it blank)
IP Address:	192.168.0.222
Protocol:	TCP
Status:	Enabled
Common Service Port:	--Select One--

RDP port: 3389

Question 13: File permissions (6 points)

From Problem 4.5: UNIX treats file directories in the same fashion as files; that is, both are defined by the same type of data structure, called an inode. As with files, directories include a nine-bit protection string. If care is not taken, this can create access control problems. For example, consider a file with the protection mode 644 (octal) contained in a directory with protection mode 773.

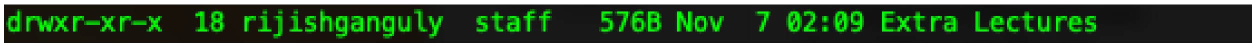
Explain the interpretation of the 9-bit protection string, for both files and directories



```
-rw-r--r--@  1 rijishganguly  staff   8.0K Nov 17 20:34 .DS_Store
-rw-r--r--@  1 rijishganguly  staff   7.6M Nov 15 17:37 13-ids-ips.pdf
-rw-r--r--@  1 rijishganguly  staff   2.6M Nov 20 01:25 14-reverse.pdf
-rw-r--r--@  1 rijishganguly  staff   1.6M Nov 20 01:25 15-wireless.pdf
-rw-r--r--@  1 rijishganguly  staff   4.0M Nov 20 01:25 16-phys-org-audit-legal-
ethical.pdf
-rw-r--r--@  1 rijishganguly  staff   1.2M Nov 20 01:23 17-human.pdf
```

The 9 protection characters can be broken up into 3 triplets ("rw-", "r--", and "r--" for the file reverse.pdf). Each triplet specifies the file protection for a different set of people. The first triplet ("rw-") specifies what the owner of the file may do with the file. The middle triplet ("r--") specifies what users in the same group as the file may do with the file. The last triplet ("r--") specifies what everyone else can do with the file.

Each character in the triplets can be either a letter or a dash ("-"). If the first character in a triplet is an "r", the set of users corresponding to the triplet (owner, group, or world) can read the file. If the second character in a triplet is a "w", the corresponding set of users can write to the file. If the third character in a triplet is an "x", the corresponding set of users can execute the file. The lack of one of these access privileges is designated by the dash ("-").



```
drwxr-xr-x  18 rijishganguly  staff   576B Nov  7 02:09 Extra Lectures
```

Directories are indicated by a "d" in the first character of the protection string. "r" means you can read the contents of the directory. "w" means you can write to the directory. You can create, rename, and remove files contained in the directory (regardless of their individual protections). "x" means you can access files in the directory, subject to their individual permissions.

Explain the meaning of the octal protection string of 644 for a file, and 773 for a directory.

644 for a file means rw-rw-r-- which implies that owner and group can read and write the file, whereas others can only read the file.
773 for a directory means rwxrwx-wx which implies that everyone can access the files in the directory, write to the directory but only the user and group can read the contents of the directory.

Do an experiment: Can non-owner non-group users list the contents of the directory? Can such users create new files in the directory?

The non-owner, non-group users cannot list the contents of the directory. However, they can create new files in the directory, since creating new files require permission to write to the directory (“w”).

Consider the ls output below.

```
 -rw-r--r--  1 root     root        1.6M Sep 25 16:58 gosh.tar.gz
```

Explain in detail what each element this line means.

The -rw-r--r—suggests that gosh.tar.gz is a file with the owner having read and write permissions. Group and other user have only reading permissions.

1 suggests that number of files within this file.

The user and owner of the file is root.
The group this file belongs to is root.

The file is 1.6 MB in size.
The file was last modified at 16:58 pm
The name of the file is gosh.tar.gz

What command would you use to change the access rights to allow any user to edit the file?

chmod u+w,g+w,o+w gosh.tar.gz
or
chmod a+w gosh.tar.gz

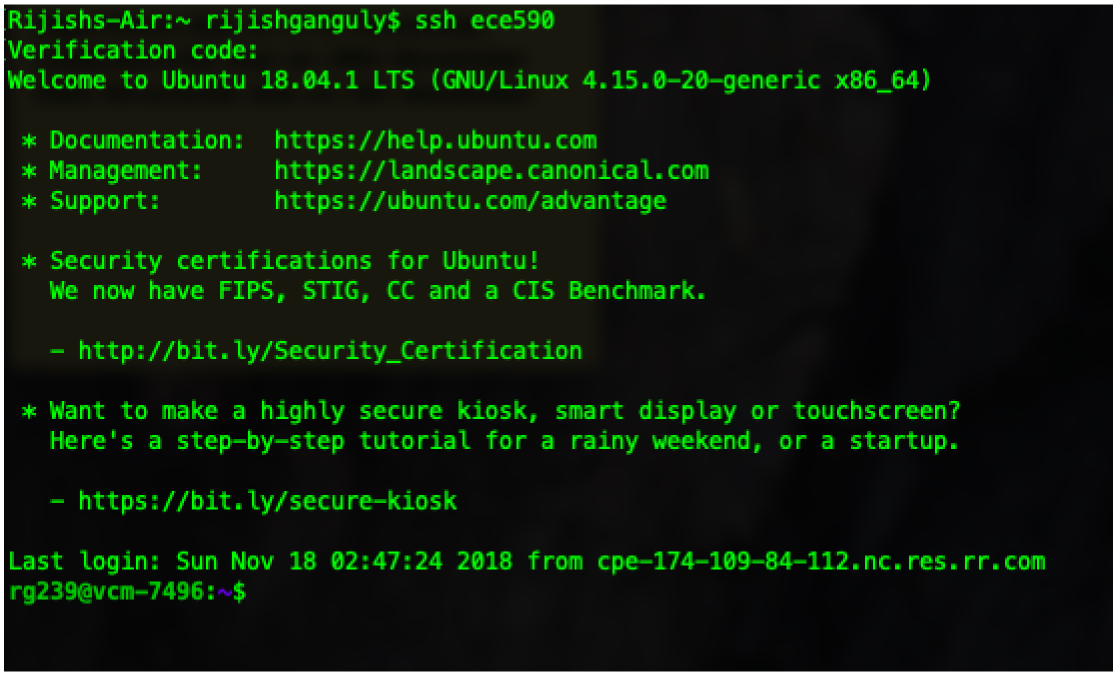
What command would you use to take full ownership from root to yourself?
sudo chown rijishganguly gosh.tar.gz

Question 14: SSH with MFA (5 points)

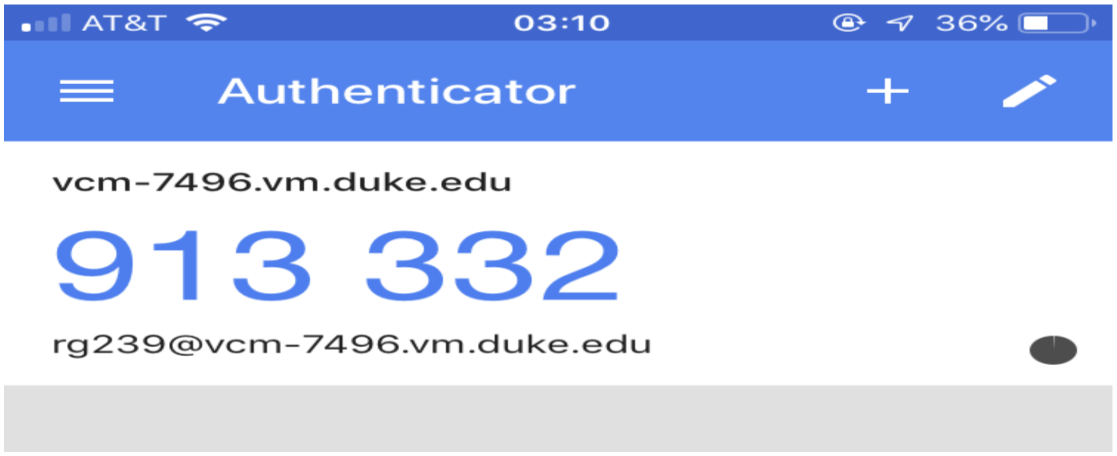
On your Linux VM, you are going to add some extra protections to the SSH server by adding Multi-Factor Authentication (MFA) to your account.

Some research will tell you how to do this using Google Authenticator or the desktop app Authy.

Show a screenshot of your SSH login with MFA code.



```
Rijishs-Air:~ rijishganguly$ ssh ece590
Verification code:
Welcome to Ubuntu 18.04.1 LTS (GNU/Linux 4.15.0-20-generic x86_64)

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

 * Security certifications for Ubuntu!
   We now have FIPS, STIG, CC and a CIS Benchmark.

   - http://bit.ly/Security_Certification

 * Want to make a highly secure kiosk, smart display or touchscreen?
   Here's a step-by-step tutorial for a rainy weekend, or a startup.

   - https://bit.ly/secure-kiosk

Last login: Sun Nov 18 02:47:24 2018 from cpe-174-109-84-112.nc.res.rr.com
rg239@vcm-7496:~$
```



vcm-7496.vm.duke.edu

913 332

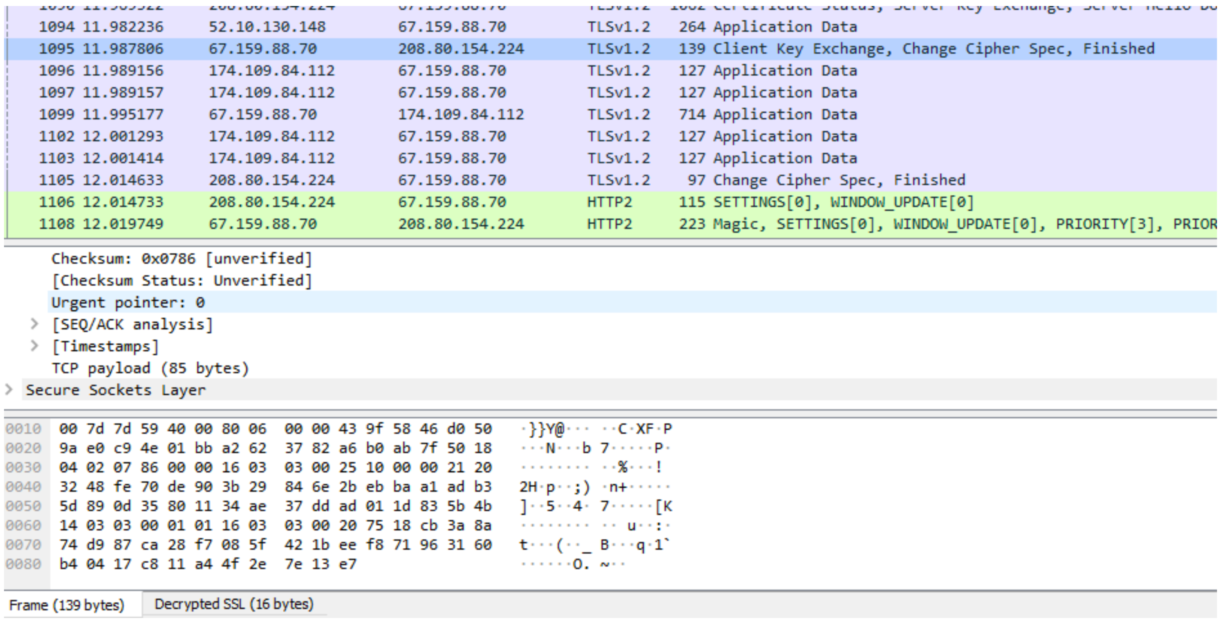
rg239@vcm-7496.vm.duke.edu

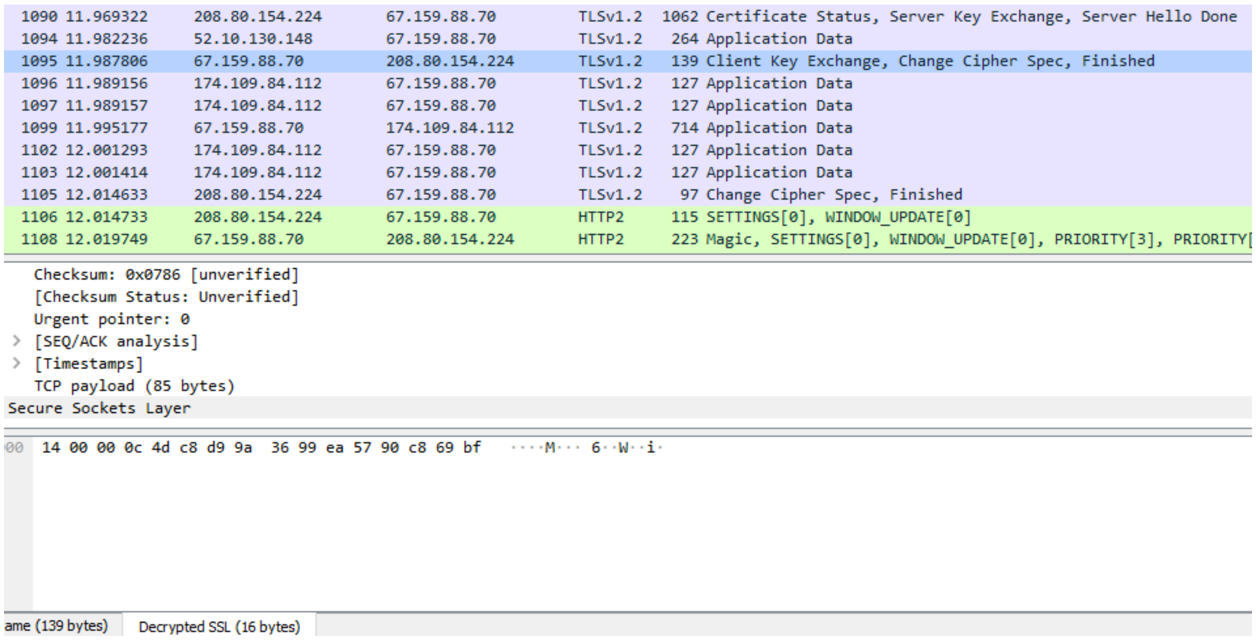
Google Authenticator code

Question 15: Decrypting SSL/TLS Traffic with Wireshark and Session Keys (5 points)

Follow this guide to decrypt some SSL/TLS traffic on your Windows VM. You will need to install Wireshark and Firefox. After setting up the necessary configuration of with environment variables and Wireshark, do the following.

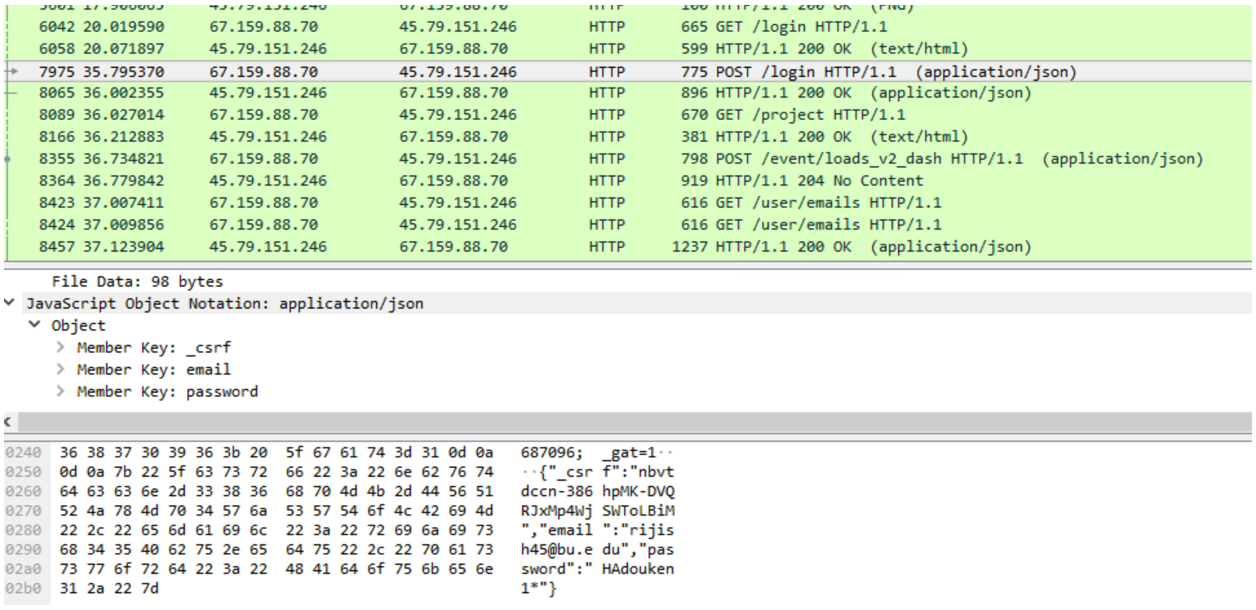
Capture some SSL traffic and show the encrypted and decrypted traffic.


```
1094 11.982236   52.10.130.148    67.159.88.70     TLSv1.2   264 Application Data
1095 11.987806   67.159.88.70     208.80.154.224   TLSv1.2   139 Client Key Exchange, Change Cipher Spec, Finished
1096 11.989156   174.109.84.112   67.159.88.70     TLSv1.2   127 Application Data
1097 11.989157   174.109.84.112   67.159.88.70     TLSv1.2   127 Application Data
1099 11.995177   67.159.88.70     174.109.84.112   TLSv1.2   714 Application Data
1102 12.001293   174.109.84.112   67.159.88.70     TLSv1.2   127 Application Data
1103 12.001414   174.109.84.112   67.159.88.70     TLSv1.2   127 Application Data
1105 12.014633   208.80.154.224   67.159.88.70     TLSv1.2    97 Change Cipher Spec, Finished
1106 12.014733   208.80.154.224   67.159.88.70     HTTP2     115 SETTINGS[0], WINDOW_UPDATE[0]
1108 12.019749   67.159.88.70     208.80.154.224   HTTP2     223 Magic, SETTINGS[0], WINDOW_UPDATE[0], PRIORITY[3], PRIOR

    Checksum: 0x0786 [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
  > [SEQ/ACK analysis]
  > [Timestamps]
    TCP payload (85 bytes)
> Secure Sockets Layer

0010  00 7d 7d 59 40 00 80 06  00 00 43 9f 58 46 d0 50   ·}}Y@···  ··C·XF·P
0020  9a e0 c9 4e 01 bb a2 62  37 82 a6 b0 ab 7f 50 18   ···N···b 7·····P·
0030  04 02 07 86 00 00 16 03  03 00 25 10 00 00 21 20   ········  ··%···!
0040  32 48 fe 70 de 90 3b 29  84 6e 2b eb ba a1 ad b3   2H·p··;) ·n+·····
0050  5d 89 0d 35 80 11 34 ae  37 dd ad 01 1d 83 5b 4b   ]··5··4· 7·····[K
0060  14 03 03 00 01 01 16 03  03 00 20 75 18 cb 3a 8a   ········ ·· u··:·
0070  74 d9 87 ca 28 f7 08 5f  42 1b ee f8 71 96 31 60   t···(··_ B···q·1`
0080  b4 04 17 c8 11 a4 4f 2e  7e 13 e7                  ······O. ~··

Frame (139 bytes)   Decrypted SSL (16 bytes)
```



```
1090 11.969322    208.80.154.224    67.159.88.70      TLSv1.2  1062 Certificate Status, Server Key Exchange, Server Hello Done
1094 11.982236    52.10.130.148     67.159.88.70      TLSv1.2   264 Application Data
1095 11.987806    67.159.88.70      208.80.154.224    TLSv1.2   139 Client Key Exchange, Change Cipher Spec, Finished
1096 11.989156    174.109.84.112    67.159.88.70      TLSv1.2   127 Application Data
1097 11.989157    174.109.84.112    67.159.88.70      TLSv1.2   127 Application Data
1099 11.995177    67.159.88.70      174.109.84.112    TLSv1.2   714 Application Data
1102 12.001293    174.109.84.112    67.159.88.70      TLSv1.2   127 Application Data
1103 12.001414    174.109.84.112    67.159.88.70      TLSv1.2   127 Application Data
1105 12.014633    208.80.154.224    67.159.88.70      TLSv1.2    97 Change Cipher Spec, Finished
1106 12.014733    208.80.154.224    67.159.88.70      HTTP2     115 SETTINGS[0], WINDOW_UPDATE[0]
1108 12.019749    67.159.88.70      208.80.154.224    HTTP2     223 Magic, SETTINGS[0], WINDOW_UPDATE[0], PRIORITY[3], PRIORITY[

    Checksum: 0x0786 [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
  > [SEQ/ACK analysis]
  > [Timestamps]
    TCP payload (85 bytes)
Secure Sockets Layer

00  14 00 00 0c 4d c8 d9 9a  36 99 ea 57 90 c8 69 bf   ····M··· 6··W··i·

ame (139 bytes)   Decrypted SSL (16 bytes)
```

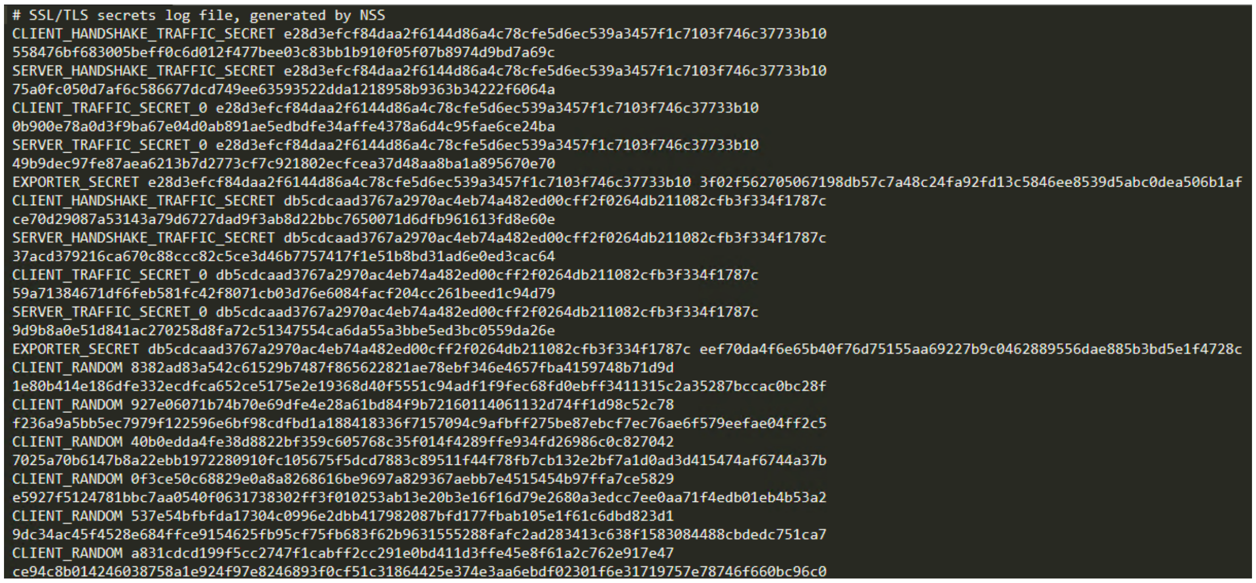
Try to show a decrypt password that would be used to log in to a “secure” website.



```
6042 20.019590   67.159.88.70     45.79.151.246    HTTP    665 GET /login HTTP/1.1
6058 20.071897   45.79.151.246    67.159.88.70     HTTP    599 HTTP/1.1 200 OK  (text/html)
7975 35.795370   67.159.88.70     45.79.151.246    HTTP    775 POST /login HTTP/1.1  (application/json)
8065 36.002355   45.79.151.246    67.159.88.70     HTTP    896 HTTP/1.1 200 OK  (application/json)
8089 36.027014   67.159.88.70     45.79.151.246    HTTP    670 GET /project HTTP/1.1
8166 36.212883   45.79.151.246    67.159.88.70     HTTP    381 HTTP/1.1 200 OK  (text/html)
8355 36.734821   67.159.88.70     45.79.151.246    HTTP    798 POST /event/loads_v2_dash HTTP/1.1  (application/json)
8364 36.779842   45.79.151.246    67.159.88.70     HTTP    919 HTTP/1.1 204 No Content
8423 37.007411   67.159.88.70     45.79.151.246    HTTP    616 GET /user/emails HTTP/1.1
8424 37.009856   67.159.88.70     45.79.151.246    HTTP    616 GET /user/emails HTTP/1.1
8457 37.123904   45.79.151.246    67.159.88.70     HTTP   1237 HTTP/1.1 200 OK  (application/json)

     File Data: 98 bytes
v JavaScript Object Notation: application/json
  v Object
    > Member Key: _csrf
    > Member Key: email
    > Member Key: password

0240  36 38 37 30 39 36 3b 20  5f 67 61 74 3d 31 0d 0a   687096;  _gat=1··
0250  0d 0a 7b 22 5f 63 73 72  66 22 3a 22 6e 62 76 74   ··{"_csr f":"nbvt
0260  64 63 63 6e 2d 33 38 36  68 70 4d 4b 2d 44 56 51   dccn-386 hpMK-DVQ
0270  52 4a 78 4d 70 34 57 6a  53 57 54 6f 4c 42 69 4d   RJxMp4Wj SWToLBiM
0280  22 2c 22 65 6d 61 69 6c  22 3a 22 72 69 6a 69 73   ","email ":"rijis
0290  68 34 35 40 62 75 2e 65  64 75 22 2c 22 70 61 73   h45@bu.e du","pas
02a0  73 77 6f 72 64 22 3a 22  48 41 64 6f 75 6b 65 6e   sword":" HAdouken
02b0  31 2a 22 7d                                        1*"}
```

Website – Overleaf.com

Show a sample from the session key file you set up as well.



```
# SSL/TLS secrets log file, generated by NSS
CLIENT_HANDSHAKE_TRAFFIC_SECRET e28d3efcf84daa2f6144d86a4c78cfe5d6ec539a3457f1c7103f746c37733b10
558476bf683005beff0c6d012f477bee03c83bb1b910f05f07b8974d9bd7a69c
SERVER_HANDSHAKE_TRAFFIC_SECRET e28d3efcf84daa2f6144d86a4c78cfe5d6ec539a3457f1c7103f746c37733b10
75a0fc050d7af6c586677dcd749ee63593522dda1218958b9363b34222f6064a
CLIENT_TRAFFIC_SECRET_0 e28d3efcf84daa2f6144d86a4c78cfe5d6ec539a3457f1c7103f746c37733b10
0b900e78a0d3f9ba67e04d0ab891ae5edbdfe34affe4378a6d4c95fae6ce24ba
SERVER_TRAFFIC_SECRET_0 e28d3efcf84daa2f6144d86a4c78cfe5d6ec539a3457f1c7103f746c37733b10
49b9dec97fe87aea6213b7d2773cf7c921802ecfcea37d48aa8ba1a895670e70
EXPORTER_SECRET e28d3efcf84daa2f6144d86a4c78cfe5d6ec539a3457f1c7103f746c37733b10 3f02f562705067198db57c7a48c24fa92fd13c5846ee8539d5abc0dea506b1af
CLIENT_HANDSHAKE_TRAFFIC_SECRET db5cdcaad3767a2970ac4eb74a482ed00cff2f0264db211082cfb3f334f1787c
ce70d29087a53143a79d6727dad9f3ab8d22bbc7650071d6dfb961613fd8e60e
SERVER_HANDSHAKE_TRAFFIC_SECRET db5cdcaad3767a2970ac4eb74a482ed00cff2f0264db211082cfb3f334f1787c
37acd379216ca670c88ccc82c5ce3d46b7757417f1e51b8bd31ad6e0ed3cac64
CLIENT_TRAFFIC_SECRET_0 db5cdcaad3767a2970ac4eb74a482ed00cff2f0264db211082cfb3f334f1787c
59a71384671df6feb581fc42f8071cb03d76e6084facf204cc261beed1c94d79
SERVER_TRAFFIC_SECRET_0 db5cdcaad3767a2970ac4eb74a482ed00cff2f0264db211082cfb3f334f1787c
9d9b8a0e51d841ac270258d8fa72c51347554ca6da55a3bbe5ed3bc0559da26e
EXPORTER_SECRET db5cdcaad3767a2970ac4eb74a482ed00cff2f0264db211082cfb3f334f1787c eef70da4f6e65b40f76d75155aa69227b9c0462889556dae885b3bd5e1f4728c
CLIENT_RANDOM 8382ad83a542c61529b7487f865622821ae78ebf346e4657fba4159748b71d9d
1e80b414e186dfe332ecdfca652ce5175e2e19368d40f5551c94adf1f9fec68fd0ebff3411315c2a35287bccac0bc28f
CLIENT_RANDOM 927e06071b74b70e69dfe4e28a61bd84f9b72160114061132d74ff1d98c52c78
f236a9a5bb5ec7979f122596e6bf98cdfbd1a188418336f7157094c9afbff275be87ebcf7ec76ae6f579eefae04ff2c5
CLIENT_RANDOM 40b0edda4fe38d8822bf359c605768c35f014f4289ffe934fd26986c0c827042
7025a70b6147b8a22ebb1972280910fc105675f5dcd7883c89511f44f78fb7cb132e2bf7a1d0ad3d415474af6744a37b
CLIENT_RANDOM 0f3ce50c68829e0a8a8268616be9697a829367aebb7e4515454b97ffa7ce5829
e5927f5124781bbc7aa0540f0631738302ff3f010253ab13e20b3e16f16d79e2680a3edcc7ee0aa71f4edb01eb4b53a2
CLIENT_RANDOM 537e54bfbfda17304c0996e2dbb417982087bfd177fbab105e1f61c6dbd823d1
9dc34ac45f4528e684ffce9154625fb95cf75fb683f62b9631555288fafc2ad283413c638f1583084488cbdedc751ca7
CLIENT_RANDOM a831cdcd199f5cc2747f1cabff2cc291e0bd411d3ffe45e8f61a2c762e917e47
ce94c8b014246038758a1e924f97e8246893f0cf51c31864425e374e3aa6ebdf02301f6e31719757e78746f660bc96c0
```

Note: Do not show us anything *actually* sensitive or important to you!

Question 16: Reverse Engineering (6 points)

Download this Linux binary called saltymd5. It is a program in the same tradition as the autograders from homeworks 2 and 3 -- it hashes a "secret salt" (which we now know is actually an HMAC key) plus the content of a provided file using the MD5 algorithm.

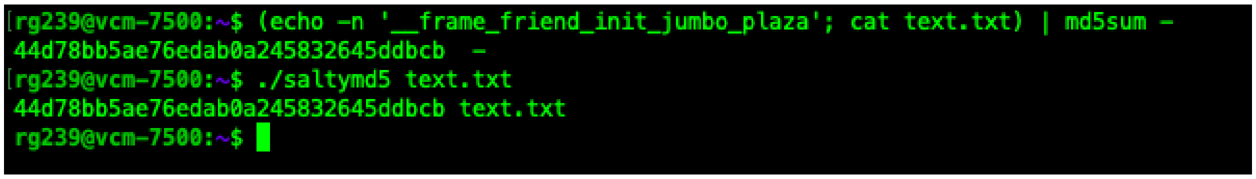
Use the tools of your choice to determine what the HMAC key is. Here is an example of how to check your answer:

```
$ (echo -n MyGuessOfWhatTheSaltIs ; cat somefile) | md5sum -
d451ed8c5d854d7f014d107756fa259a  -
$ ./saltymd5 somefile
0059a25e8f40099235e1e6335df0c9bd somefile
```

When you prepend the correct HMAC key to the input, regular old md5sum will give the same output as saltymd5. The hashes above don't match, so "MyGuessOfWhatTheSaltIs" isn't the correct HMAC key.

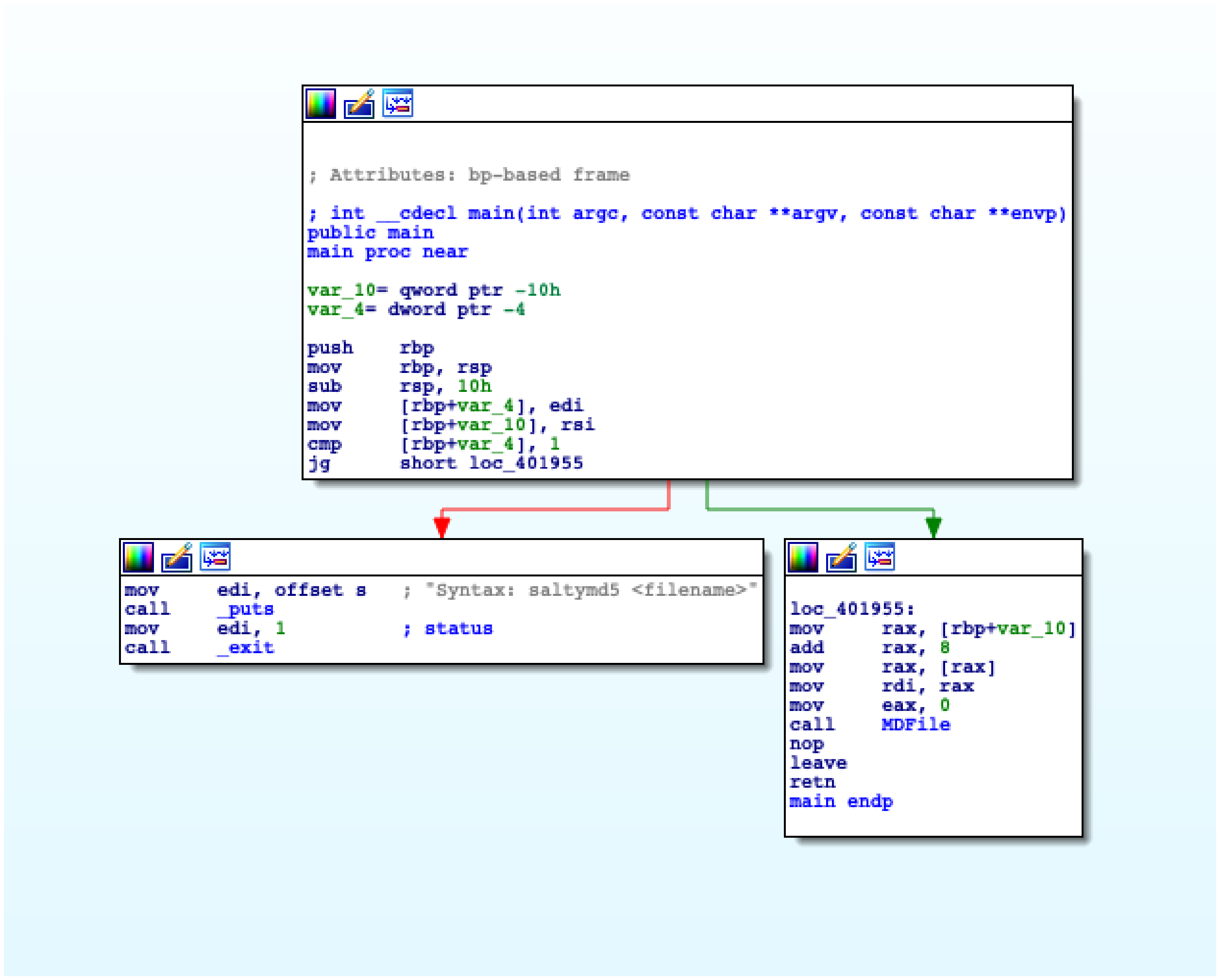
What is the HMAC key ("secret salt")? Show how you obtained your answer.

The HMAC key is __frame_friend_init_jumbo_plaza



```
[rg239@vcm-7500:~$ (echo -n '__frame_friend_init_jumbo_plaza'; cat text.txt) | md5sum -
 44d78bb5ae76edab0a245832645ddbcb  -
[rg239@vcm-7500:~$ ./saltymd5 text.txt
 44d78bb5ae76edab0a245832645ddbcb text.txt
 rg239@vcm-7500:~$ 
```

I disassembled the binary using IDA Pro and started analyzing the program flow. In the main function, we see a function call- MDFile

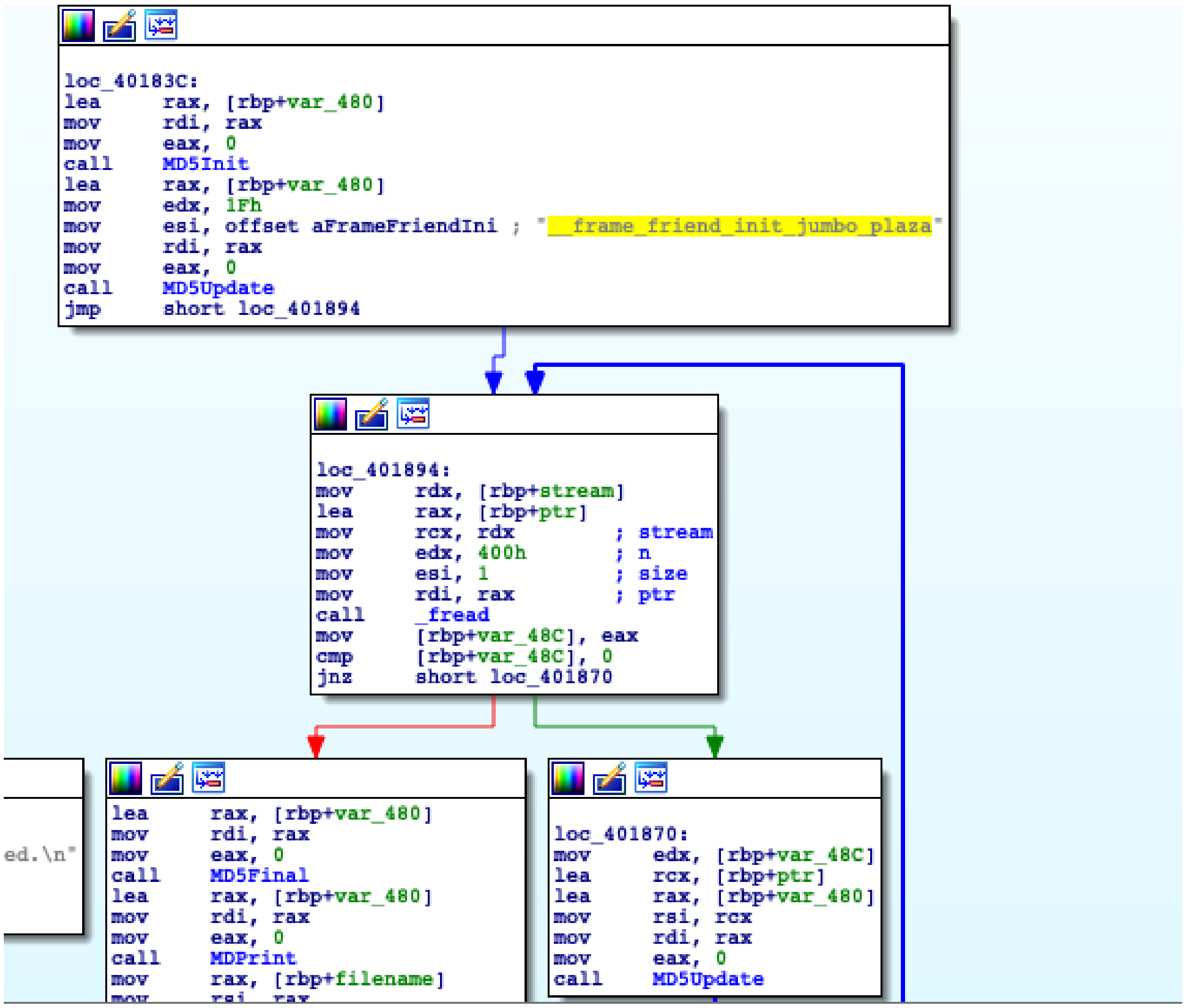


```
; Attributes: bp-based frame

; int __cdecl main(int argc, const char **argv, const char **envp)
public main
main proc near

var_10= qword ptr -10h
var_4= dword ptr -4

push    rbp
mov     rbp, rsp
sub     rsp, 10h
mov     [rbp+var_4], edi
mov     [rbp+var_10], rsi
cmp     [rbp+var_4], 1
jg      short loc_401955
```

```
mov     edi, offset s   ; "Syntax: saltymd5 <filename>"
call    _puts
mov     edi, 1          ; status
call    _exit
```

```
loc_401955:
mov     rax, [rbp+var_10]
add     rax, 8
mov     rax, [rax]
mov     rdi, rax
mov     eax, 0
call    MDFile
nop
leave
retn
main endp
```

When we analyze the MDFile function, we can observe that there is a loop in action which helps in traversing over the entire content of the file. The HMAC key is stored as offset aFrameFriendIni in the register rsi and gets prepended to the content of the file before the MD5 hash is applied.



```
loc_40183C:
lea     rax, [rbp+var_480]
mov     rdi, rax
mov     eax, 0
call    MD5Init
lea     rax, [rbp+var_480]
mov     edx, 1Fh
mov     esi, offset aFrameFriendIni ; "__frame_friend_init_jumbo_plaza"
mov     rdi, rax
mov     eax, 0
call    MD5Update
jmp     short loc_401894
```

```
loc_401894:
mov     rdx, [rbp+stream]
lea     rax, [rbp+ptr]
mov     rcx, rdx        ; stream
mov     edx, 400h       ; n
mov     esi, 1          ; size
mov     rdi, rax        ; ptr
call    _fread
mov     [rbp+var_48C], eax
cmp     [rbp+var_48C], 0
jnz     short loc_401870
```

```
ed.\n"
```

```
lea     rax, [rbp+var_480]
mov     rdi, rax
mov     eax, 0
call    MD5Final
lea     rax, [rbp+var_480]
mov     rdi, rax
mov     eax, 0
call    MDPrint
mov     rax, [rbp+filename]
```

```
loc_401870:
mov     edx, [rbp+var_48C]
lea     rcx, [rbp+ptr]
lea     rax, [rbp+var_480]
mov     rsi, rcx
mov     rdi, rax
mov     eax, 0
call    MD5Update
```

HINT: The free version of IDA Pro can make quick work of this problem.

Question 17: Social Engineering (5 points)

It is common for people to use URL shorteners to make long URLs easier to remember and to fit them in limited space, such as a tweet or QR code. A URL shortener is a simple service that takes a URL and gives an alias which, when visited, will redirect to the original URL.

At the same time, a common step in social engineering is to get a target to visit a URL. This could be to infect them with some form of malware, but most often it's just a simple and innocuous way to get the target's IP address and basic browser/OS details.

In this question, you will use a URL shortening service we have provided, and you will be able to login to create shortened URLs and see the IP addresses and User Agent strings of visitors to the URLs you create. You will induce someone you know to visit a shortened URL of your creation, and note the IP address and browser details in a screenshot.

Important note: You must follow the procedure below in order to complete this question within the bounds of the ethics agreement to which you have agreed.

Requirements:

- Choice of target: You must already know the "target" (the person who you're inducing to visit the URL), but they may not be another student enrolled in this course. The system the target is using must not be especially sensitive (e.g. a corporate-owned workstation, point of sale system, etc.).
- Duty to disclose: You must disclose that you are enrolled in a computer security course and want to show them a security demo, and that they are under no obligation to participate. You must indicate that no data loss or unauthorized access to their system will be incurred from this procedure if they participate. Lastly, when the interaction is complete (whether they visited the URL or not), you must disclose the entire nature of the exercise to them, including any data that was or would have been revealed. Further, ensure the target understands that this information is automatically disclosed to every website they visit, and does not by itself constitute a threat.
- Use of the URL shortener: When you create a shortened URL, do not use a private or sensitive destination URL, or a URL that contains objectionable content. Do not modify or interfere with other URL aliases that have been created.

- Go no further: Despite it being basically public data, you must not use the information obtained in any way other than to disclose it to the target and produce it below in this assignment.
- *DON'T SCREW UP: IN GENERAL, YOU MUST USE GOOD JUDGMENT IN KEEPING WITH THE ETHICAL STANDARDS SET FORTH FOR THE COURSE!*

A URL redirect service has been set up for your use at http://googz.us/. When you visit that URL, you will be redirected to the admin panel for Your Own URL Shortener (YOURLS), a web-based package used to create a URL shortener service. The username is "student" with the password "sec@590". Once you login, you can create URL aliases. I recommend you point your URL alias at something relevant to security, so that when the target arrives there, it is not obvious that the act of visiting the URL was itself the goal.

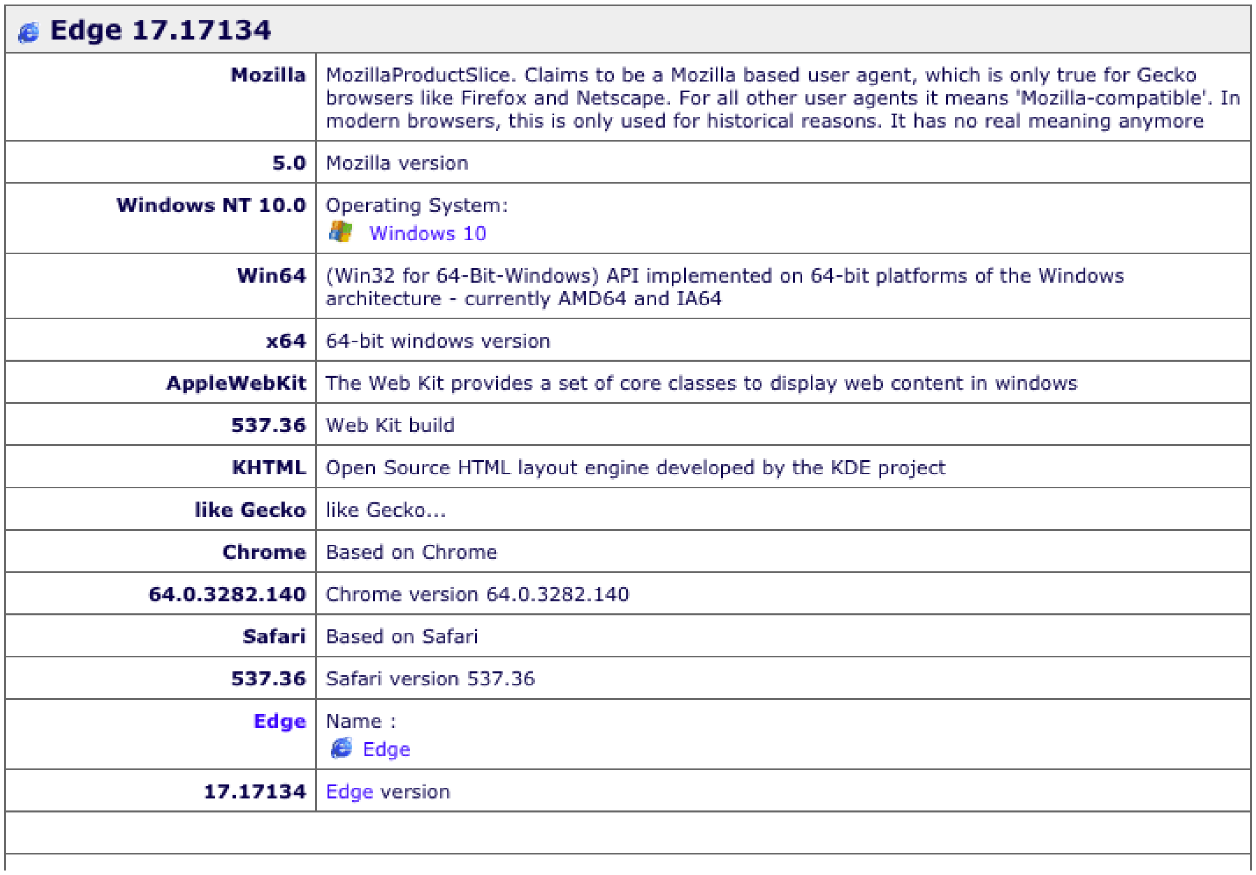
Also, YOURLS will generate alphanumerically aliases starting from '1' and incrementing in base-36, meaning that by default you'll have a URL like "googz.us/c". To create a more realistic short URL, choose an alias manually and have it be 5 to 7 random alphanumerics, such as "googz.us/8gf3sf".

YOURLS identifies clients as they use the service, but this information is summarized statistically rather than provided in full. Instead, to see who has visited what URL, you can view the site's HTTP access log here: http://googz.us/accesslog.php. From the log, you can find the request for the shortened URL that you created that was accessed by the target.

When you have succeeded, paste the IP address and user agent obtained below.

```
174.109.84.112 - - [19/Nov/2018:16:49:30 -0800] "GET /k HTTP/1.1" 301 341 "-"
"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/64.0.3282.140 Safari/537.36 Edge/17.17134"
```

Use a User Agent analyzer to determine their exact OS and browser version, show your findings below.



Edge 17.17134	
Mozilla	MozillaProductSlice. Claims to be a Mozilla based user agent, which is only true for Gecko browsers like Firefox and Netscape. For all other user agents it means 'Mozilla-compatible'. In modern browsers, this is only used for historical reasons. It has no real meaning anymore
5.0	Mozilla version
Windows NT 10.0	Operating System: Windows 10
Win64	(Win32 for 64-Bit-Windows) API implemented on 64-bit platforms of the Windows architecture - currently AMD64 and IA64
x64	64-bit windows version
AppleWebKit	The Web Kit provides a set of core classes to display web content in windows
537.36	Web Kit build
KHTML	Open Source HTML layout engine developed by the KDE project
like Gecko	like Gecko...
Chrome	Based on Chrome
64.0.3282.140	Chrome version 64.0.3282.140
Safari	Based on Safari
537.36	Safari version 537.36
Edge	Name : Edge
17.17134	Edge version

Describe how your social interaction went. How suspicious was the target to visit the URL despite your assurances?

Initially, despite my assurance, the target was reluctant to visit the link. However, after showing him the instructions for this question, he agreed to visit the link.

Without attempting to do so, describe how an attacker could induce a stranger to visit such a link, especially a stranger in a corporate or other firewalled environment. What strategies could help an attacker be successful in this pursuit?

An attacker can induce a stranger to visit such a link by creating a custom link with the organization name included within the link. Suppose an employee works at Fidelity Investments. I can create a link such as http://googz.us/fidelityorg
Creating a link with extension names of a reputable organization might increase the chance of the stranger clicking the link.