

## FCS Assignment 3

**Name:** Meetakshi Setiya

**Roll no:** 2019253

### Question 1.

Yes, even if the stolen data was encrypted, the organisation must notify the security breach. This is because:

- Everyone in the organisation has to be alerted and it has to be made sure that any further damages or data loss is prevented.
- It is very important to discuss the severity and the scope of the breach along with the legal repercussions.
- The organisation also has to make sure that the exploited vulnerability is fixed and that any compromised system is shut down or suspended. This should be done to make sure that no further unauthorized access is possible.
- Besides, legislatures like The General Data Protection Regulation (GDPR) require the organisations to mandatorily notify the affected users that the security of their data has been compromised. Anything otherwise might lead to legal ramifications for the organisation.
- There is always a possibility that even if encrypted data is stolen, and there is no ample security measure taken, the attacker can attempt to steal the encryption key too and in that case, the harm will be irreversible.

Example: In 2021, a hacker used data scraping techniques to exploit LinkedIn's API before dumping an information data set containing details of 500M of its customers. LinkedIn argued that no private data was exposed, the hacker posed a scraped data sample containing phone numbers, geolocation records, gender, social media details of the users. This was plenty of information for malicious parties to start social engineering and other attacks on the victims of the breach. In November 2015, the database of the website Abandonia was breached, giving the attackers access to the data (IP address, username and encrypted passwords) of more than 700k users registered on the website. The passwords were salted and encrypted using MD5 which were easily cracked. The breach was not even made known to its 700k+ victims until sometime later.

## **Question 2.**

Both anonymization and pseudonymisation are practices used to secure sensitive or private data about individuals and prevent exposing the said data to unauthorized personnel.

Anonymization: Anonymization is the process of protecting private or sensitive information by encryption or complete removal of identifiers (like IP addresses, devices etc.) that can be used to identify an individual. Since anonymization involves permanent replacement of the selected data with unrelated characters, the data once anonymized cannot be re-identified directly. Anonymizing data however, can destroy the value that the data holds for an organization.

Additionally, GDPR does not see anonymized data as personal data anymore.

Pseudonymization: Pseudonymization is the process of switching sensitive details, like name, email etc with false information- like a pseudonym or an alias such that it cannot be attributed to a specific subject without the use of additional information. A good example would be the replacement of names with temporary IDs. Pseudonymization is a reversible process, unlike anonymization, which means that even though the data was changed for the purpose of its concealment/de-identification, it can still be re-identified if necessary (using a proper key). According to GDPR, a pseudonym is still considered to be personal data because of its reversible nature.

*Given data:* Patient John Doe has a gamma-GT value of 83U/L

*Anonymized data:* A patient has has a gamma-GT value of 83U/L

*Pseudonymised data:* Patient 1221B has a gamma-GT value of 83U/L  
(Key mapping table: Patient 1221B → John Doe)

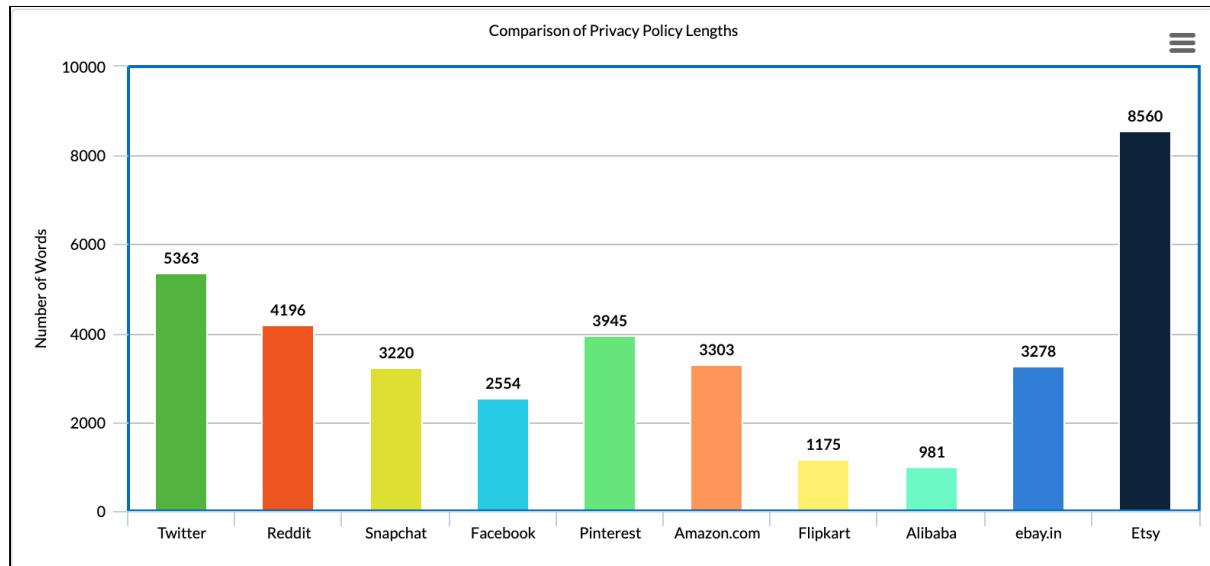
## **Question 3.**

The social media websites I chose are- Twitter, Reddit, Snapchat, Facebook and Pinterest.

The e-commerce websites I chose are- Amazon.com, ebay.in, Flipkart, Alibaba and Etsy.

a.

Bar chart showing the lengths of privacy policy for 5 different social media websites and 5 e-commerce websites is given below:



b.

I used [Webfx](#) to get the text complexity metrics of the various privacy policies.

The major ones are summarised below:

Website	No. of words	No. of complex words	Average words per sentence	Flesch Kincaid Reading Ease*	Flesch Kincaid Grade Level#
-----	-----	<b>Social media</b>	<b>websites</b>	-----	-----
Twitter	5363	840	23.12	46.3	12.5
Reddit	4196	793	21.52	39.6	13.1
Snapchat	3220	512	19.75	50.5	11.1
Facebook	2554	406	21.11	49.5	11.6
Pinterest	3945	661	21.68	47.7	12
-----	-----	<b>E-commerce</b>	<b>websites</b>	-----	-----

Amazon.com	3303	670	23.43	37.2	13.9
Flipkart	1175	212	18.65	47.1	11.3
Alibaba	981	197	23.93	35.5	14.3
ebay.in	3278	657	26.02	35.6	14.8
Etsy	8560	1603	25.86	37.2	14.5

\*The Flesch Kincaid readability scores are the most widely used measures of readability. The score is based on a ranking scale of 0-100, and the higher the score the better. For most businesses, 65 is a good target.

#The Flesch Kincaid readability scores are the most widely used measures of readability. The score is based on the American school grade one would need to be in to comprehend the material on a page.

### C.

Observations:

- Every privacy policy scores low on the readability ease front and thus requires some scrutiny and effort to go through them. Average complexity of words used (total number of words/complex words) also bordered to a ratio of roughly 6:1.
- On comparing the complexity of vocabulary used, the ratio of total number of words to complex words in the text was on average higher for the social media websites (nearing approx 6) than the e-commerce websites.
- Out of the social media websites, Snapchat scored the highest in readability score and had a lesser average number of words per sentence. It was also comparatively shorter than the privacy policies of its counterparts (except facebook). Interestingly, Snapchat is also reported to be a safer social media app than its counterparts.
- Out of the social networking sites, Pinterest's privacy policy had the least ratio of total words to complex words - 5.3:1. On going through the document, I found it comparatively easier to comprehend on a first glance.
- Facebook is reportedly the most used social media platform, even in 2021, yet its privacy policy had the least word count of the rest of the social media platforms'. The complexity ratio of the text however, was on the higher end- ie. 6.29.

- Etsy's privacy policy was the longest and the hardest to go through because of long sentences and the sheer length of the document.
- Out of the ecommerce websites, Alibaba's privacy policy text was the shortest. It was, in fact, the shortest privacy policy text in the table. The ratio of total words to complex words was also the least: around 4.9.
- Flipkart's privacy policy text has the least sentence length out of every privacy policy text present in the table. The complexity ratio is about 5.5. The text was also on the shorter end of the range- a mere 1175 words long.
- Amazon.com is reportedly the biggest e-commerce website by traffic. Its privacy policy was neither too long, nor too short, by word count. Nor was the complexity ratio as high as that of some of the other websites in this table. In fact it was about 4.92.
- For Amazon.com, Flipkart and Alibaba, the order of popularity is the same as the order of privacy policy's length, is the same as the enlisted order.
- Additionally, ebay.in scored the highest in the Flesch Kincaid Grade Level readability test scale. Snapchat scored the highest in Flesch Kincaid reading ease scale, and Flipkart had the lowest average number of words per sentence.

#### **Question 4.**

a.

Given below is a screenshot of the metadata of the image 4\_img.jpg as shown by exiftool.

Location- 36 deg 50' 30.18" S, 174 deg 46' 32.89" E

City- Auckland

Country- New Zealand

```
[1573 ~ /desktop >> exiftool 4_img.jpg
ExifTool Version Number      : 12.34
File Name                   : 4_img.jpg
Directory                   : .
File Size                   : 83 KiB
File Modification Date/Time : 2021:11:02 14:22:09+05:30
File Access Date/Time       : 2021:11:02 14:22:20+05:30
File Inode Change Date/Time: 2021:11:02 14:22:19+05:30
File Permissions            : -rw-r--r--
File Type                   : JPEG
File Type Extension         : jpg
MIME Type                   : image/jpeg
JFIF Version                : 1.01
Exif Byte Order              : Big-endian (Motorola, MM)
Image Description            : Malana
X Resolution                 : 1
Y Resolution                 : 1
Resolution Unit              : None
Y Cb Cr Positioning         : Centered
GPS Version ID               : 2.3.0.0
GPS Latitude Ref             : South
GPS Longitude Ref            : East
Image Width                  : 719
Image Height                 : 496
Encoding Process              : Progressive DCT, Huffman coding
Bits Per Sample               : 8
Color Components              : 3
Y Cb Cr Sub Sampling         : YCbCr4:2:0 (2 2)
Image Size                   : 719x496
Megapixels                   : 0.357
GPS Latitude                 : 36 deg 50' 30.18" S
GPS Longitude                : 174 deg 46' 32.89" E
GPS Position                 : 36 deg 50' 30.18" S, 174 deg 46' 32.89" E
1574 ~ /desktop >> ]
```

**b.**

IIIT Delhi's GPS coordinates are: 28.5455536926139, 77.27307041485886

The command that can be used to change the location of the image to IIIT Delhi is:

```
exiftool -GPSLatitude="28.5455536926139"
-GPSLongitude="77.27307041485886" 4_img.jpg
```

Here is a screenshot showing the changed GPS location:

```
[1579 ~ ~/desktop >> exiftool -GPSLatitude="28.5455536926139" -GPSLongitude="77.27307041485886" 4_img.jpg
 1 image files updated
[1580 ~ ~/desktop >> exiftool 4_img.jpg
ExifTool Version Number      : 12.34
File Name                   : 4_img.jpg
Directory                   : .
File Size                   : 83 Kib
File Modification Date/Time : 2021:11:02 14:49:59+05:30
File Access Date/Time       : 2021:11:02 14:50:01+05:30
File Inode Change Date/Time: 2021:11:02 14:49:59+05:30
File Permissions            : -rw-r--r--
File Type                   : JPEG
File Type Extension         : jpg
MIME Type                   : image/jpeg
JFIF Version                : 1.01
Exif Byte Order              : Big-endian (Motorola, MM)
Image Description            : Malana
X Resolution                 : 1
Y Resolution                 : 1
Resolution Unit              : None
Y Cb Cr Positioning         : Centered
GPS Version ID               : 2.3.0.0
GPS Latitude Ref             : South
GPS Longitude Ref            : East
Image Width                  : 719
Image Height                 : 496
Encoding Process             : Progressive DCT, Huffman coding
Bits Per Sample              : 8
Color Components              : 3
Y Cb Cr Sub Sampling        : YCbCr4:2:0 (2 2)
Image Size                   : 719x496
Megapixels                   : 0.357
GPS Latitude                 : 28 deg 32' 43.99" S
GPS Longitude                : 77 deg 16' 23.05" E
GPS Position                 : 28 deg 32' 43.99" S, 77 deg 16' 23.05" E
1581 ~ ~/desktop >>
```

#### c.

I used the 'md5' command on macOS to find the md5 checksums of the image before and after changing its location.

The command I used was :

```
md5 4_img.jpg
```

```
[1575 ~ ~/desktop >> md5 4_img.jpg
MD5 (4_img.jpg) = 50bb962c9c4dbd36efcc06a5c9c6462d
[1576 ~ ~/desktop >> exiftool -GPSLatitude="28.5455536926139" -GPSLongitude="77.27307041485886" 4_img.jpg
 1 image files updated
[1577 ~ ~/desktop >> md5 4_img.jpg
MD5 (4_img.jpg) = 7f659256b4a414b5f30180c53ff433d9
1578 ~ ~/desktop >> ]
```

The two checksums are different. This is because md5 hash is a hash of the complete binary image file. For two md5 hashes to match, the images as well as their metadata must be completely identical. Hence any change to the file metadata will produce a new hash value as is visible in the screenshot.

#### d.

I changed the extension of the image from jpg to png. On comparing the metadata of the two variants, I found them to be exactly equal. Also, the md5 hash for both formats was the same as well. The screenshot is given below:

```

[1594 ~ /desktop >> exiftool 4_img.jpg
ExifTool Version Number      : 12.34
File Name                   : 4_img.jpg
Directory                   : .
File Size                   : 83 KiB
File Modification Date/Time : 2021:11:02 15:00:06+05:30
File Access Date/Time       : 2021:11:02 15:00:19+05:30
File Inode Change Date/Time: 2021:11:02 15:00:17+05:30
File Permissions            : -rw-r--r--
File Type                   : JPEG
File Type Extension         : jpg
MIME Type                   : image/jpeg
JFIF Version                : 1.01
Exif Byte Order              : Big-endian (Motorola, MM)
Image Description            : Malana
X Resolution                 : 1
Y Resolution                 : 1
Resolution Unit              : None
Y Cb Cr Positioning         : Centered
GPS Version ID               : 2.3.0.0
GPS Latitude Ref             : South
GPS Longitude Ref            : East
GPS Width                    : 719
Image Height                  : 496
Encoding Process              : Progressive DCT, Huffman coding
Bits Per Sample               : 8
Color Components              : 3
Y Cb Cr Sub Sampling         : YCbCr4:2:0 (2 2)
Image Size                   : 719x496
Megapixels                   : 0.357
GPS Latitude                 : 36 deg 50' 30.18" S
GPS Longitude                 : 174 deg 46' 32.89" E
GPS Position                 : 36 deg 50' 30.18" S, 174 deg 46' 32.89" E
[1595 ~ /desktop >> exiftool 4_img.png
ExifTool Version Number      : 12.34
File Name                   : 4_img.png
Directory                   : .
File Size                   : 83 KiB
File Modification Date/Time : 2021:11:02 14:57:08+05:30
File Access Date/Time       : 2021:11:02 14:58:01+05:30
File Inode Change Date/Time: 2021:11:02 14:57:59+05:30
File Permissions            : -rw-r--r--
File Type                   : JPEG
File Type Extension         : jpg
MIME Type                   : image/jpeg
JFIF Version                : 1.01
Exif Byte Order              : Big-endian (Motorola, MM)
Image Description            : Malana
X Resolution                 : 1
Y Resolution                 : 1
Resolution Unit              : None
Y Cb Cr Positioning         : Centered
GPS Version ID               : 2.3.0.0
GPS Latitude Ref             : South
GPS Longitude Ref            : East
GPS Width                    : 719
Image Height                  : 496
Encoding Process              : Progressive DCT, Huffman coding
Bits Per Sample               : 8
Color Components              : 3
Y Cb Cr Sub Sampling         : YCbCr4:2:0 (2 2)
Image Size                   : 719x496
Megapixels                   : 0.357
GPS Latitude                 : 36 deg 50' 30.18" S
GPS Longitude                 : 174 deg 46' 32.89" E
GPS Position                 : 36 deg 50' 30.18" S, 174 deg 46' 32.89" E
[1596 ~ /desktop >> md5 4_img.png
MD5 (4_img.png) = 50bb962c9c4dbd36efcc06a5c9c6462d
[1597 ~ /desktop >> md5 4_img.jpg
MD5 (4_img.jpg) = 50bb962c9c4dbd36efcc06a5c9c6462d
1598 ~ /desktop >> █

```

No, we cannot trust the extension of a file for forensic purposes. We should always parse the file to make sure that the content is exactly what the file extension specifies. File extensions are only used so that the operating system can call an appropriate program to open the file. Changing the file extension will not change the file type. Someone can, for example, upload a file called "file.mp4" which actually is a malicious script that could harm the service/system.

### **Question 5.**

DNS traffic monitoring can be used to deanonymize the users of Tor network. Monitoring DNS traffic requests and combining it with various fingerprinting techniques can be used to create a new type of "DNS-enhanced website fingerprinting attack." These kinds of passive attacks can not only uncover the IP address of the client but also the physical location of servers, sometimes. These attacks work particularly well against websites that do not have frequent traffic since their DNS records stand out. For example, an attacker could be monitoring the DNS traffic and if a user A visits a sparingly-visited website on tor, they can look up the exit node traffic on the said website and identify the IP address that requested the service of that website.

A research team also discovered that roughly one-third of DNS requests sent through Tor exit relays are routed through Google's public resolvers. Therefore, resolving DNS queries through Tor service ensures a significantly higher level of anonymity than making the requests directly because of DNS leaks. This also prevents the ISP and the resolved from knowing which domain was resolved by the user.

### **Question 6.**

I will be installing Tor on an Ubuntu 18.04 Virtual Machine.

First, install apt-transport -https to use https-repositories and curl to download the repo key.

```
sudo apt install apt-transport-https curl
```

Then, add Tor repo and its key using these commands:

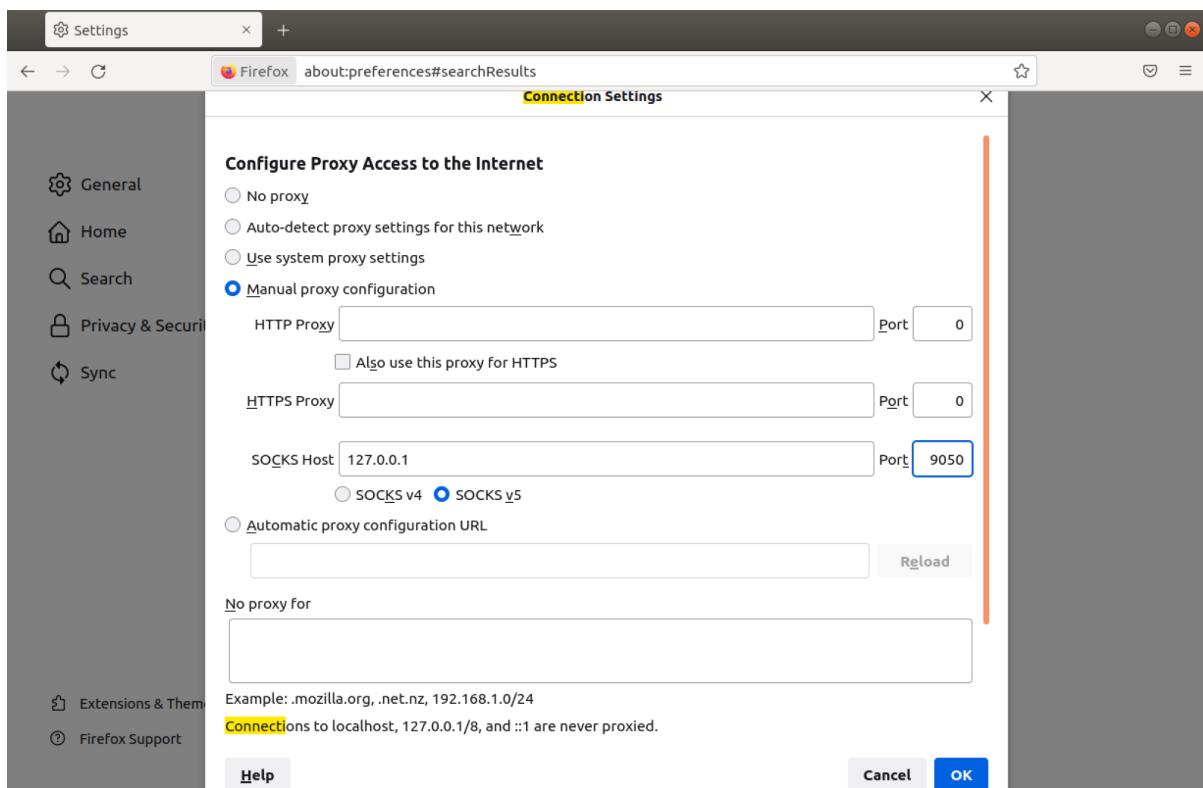
```
sudo -i
echo "deb https://deb.torproject.org/torproject.org/ $(lsb_release -cs) main" > /etc/apt/sources.list.d/tor.list
curl https://deb.torproject.org/torproject.org/A3C4F0F979CAA22CDBA8F512EE8CBC9E886DDD89.asc | gpg --import
apt update
```

```
root@ubuntu:~# echo "deb https://deb.torproject.org/torproject.org/ $(lsb_release -cs) main" > /etc/apt/sources.list.d/tor.list
root@ubuntu:~# curl https://deb.torproject.org/torproject.org/A3C4F0F979CAA22CDBA8F512EE8CBC9E886DD089.asc | gpg --import
gpg: directory '/root/.gnupg' created
gpg: keybox '/root/.gnupg/pubring.kbx' created
  % Total    % Received % Xferd  Average Speed   Time     Time   Current
          Dload  Upload   Total Spent   Left  Speed
100 49890  100 49890    0     0  54524      0 --:-- --:-- --:-- 54465
gpg: key EE8CBC9E886DD089: 1 duplicate signature removed
gpg: key EE8CBC9E886DD089: 82 signatures not checked due to missing keys
gpg: /root/.gnupg/trustdb.gpg: trustdb created
gpg: key EE8CBC9E886DD089: public key "deb.torproject.org archive signing key" imported
gpg: Total number processed: 1
gpg:           imported: 1
gpg: no ultimately trusted keys found
root@ubuntu:~# gpg --export A3C4F0F979CAA22CDBA8F512EE8CBC9E886DD089 | apt-key add -
OK
root@ubuntu:~# apt update
Hit:1 https://download.docker.com/linux/ubuntu bionic InRelease
Hit:2 http://us.archive.ubuntu.com/ubuntu bionic InRelease
Get:3 http://us.archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
Get:4 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]
Get:5 http://us.archive.ubuntu.com/ubuntu bionic-backports InRelease [74.6 kB]
Get:6 https://deb.torproject.org/torproject.org bionic InRelease [3,524 B]
Get:7 https://deb.torproject.org/torproject.org bionic/main i386 Packages [2,090 B]
Get:8 https://deb.torproject.org/torproject.org bionic/main amd64 Packages [2,092 B]
Fetched 260 kB in 4s (61.7 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
40 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@ubuntu:~#
```

Now, install torsocks, tor-geoipdb and deb.torproject.org-keyring along with tor.

```
sudo apt install tor tor-geoipdb torsocks
deb.torproject.org-keyring
```

Once tor is installed, configure Firefox to connect to Tor SOCKS5 proxy on localhost:9050



Now, Tor can be successfully run on Firefox.

A screenshot of a Firefox browser window displaying the Tor Project's 'Congratulations' page at https://check.torproject.org. The page features a large green onion logo. The main text reads: 'Congratulations. This browser is configured to use Tor.' Below it, it says 'Your IP address appears to be: 185.220.100.252'. A note below states: 'Please refer to the [Tor website](#) for further information about using Tor safely. You are now free to browse the Internet anonymously. For more information about this exit relay, see: [Relay Search](#)'. There's a blue 'Donate to Support Tor' button, and links for 'Tor Q&A Site | Volunteer | Run a Relay | Stay Anonymous'. At the bottom, a footer notes: 'The Tor Project is a US 501(c)(3) non-profit dedicated to the research, development, and education of online anonymity and privacy. [Learn More »](#)' and 'JavaScript is enabled.'

To get the hostname of the onion service on ubuntu, I followed these steps:

First, navigate to /etc/tor and create a directory tor\_host using these commands:

```
sudo -i  
cd /etc/tor  
mkdir tor_host
```

Now, modify the permissions of the tor\_host directory using the given command:

```
chmod 700 tor_host
```

The next step is to open the torrc file and do the appropriate configurations to set up an onion service. I then put these two lines in the location-hidden services section:

HiddenServiceDir /etc/tor/tor\_host

HiddenServicePort 80 127.0.0.1:80

According to the Tor project,

The **HiddenServiceDir** line specifies the directory which should contain information and cryptographic keys for the onion service and the

**HiddenServicePort** line specifies a *virtual port*.

```
| GNU nano 2.9.3                                     torrc  
  
## authentication methods, to prevent attackers from accessing it.  
#HashedControlPassword 16:872860B76453A77D60CA2BB8C1A7042072093276A3D701AD684053EC4C  
#CookieAuthentication 1  
  
##### This section is just for location-hidden services ####  
  
## Once you have configured a hidden service, you can look at the  
## contents of the file ".../hidden_service/hostname" for the address  
## to tell people.  
##  
## HiddenServicePort x:y:z says to redirect requests on port x to the  
## address y:z.  
  
#HiddenServiceDir /var/lib/tor/hidden_service/  
#HiddenServicePort 80 127.0.0.1:80  
  
#HiddenServiceDir /var/lib/tor/other_hidden_service/  
#HiddenServicePort 80 127.0.0.1:80  
#HiddenServicePort 22 127.0.0.1:22  
  
HiddenServiceDir /etc/tor/tor_host  
HiddenServicePort 80 127.0.0.1:80  
  
##### This section is just for relays #####  
#  
## See https://www.torproject.org/docs/tor-doc-relay for details.  
  
## Required: what port to advertise for incoming Tor connections.  
#ORPort 9001
```

Now, after saving the torrc file, restart the Tor service using the command `tor`.

Now, when I checked the contents of the `tor_host` directory, the file hostname contained the hostname autogenerated by tor.

```
vasilisa@ubuntu:/etc/tor$ sudo ls tor_host
authorized_clients  hostname  hs_ed25519_public_key  hs_ed25519_secret_key
vasilisa@ubuntu:/etc/tor$ sudo cat tor_host/hostname
vwojiogorepq2at5t5neu4b4skjsa3n5267fd6rdhwovcpv33cgrtid.onion
vasilisa@ubuntu:/etc/tor$
```

The auto generated hostname is :

`vwojiogorepq2at5t5neu4b4skjsa3n5267fd6rdhwovcpv33cgrtid.onion`

## Question 7.

a.

One way of doing password less authentication is using SSH private-public key pairs.

First, use `ssh-keygen` to generate the RSA key pair. I also used a passphrase to secure my private key.

```
ssh-keygen -t rsa
```

```
1614 ~  » ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/Users/meetakshi/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /Users/meetakshi/.ssh/id_rsa.
Your public key has been saved in /Users/meetakshi/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:rrsRsACs/MRrEzJ12x7BKjLrv/ePT/4lW9E+U5wyARE meetakshi@Meetakshis-MacBook-Air.local
The key's randomart image is:
+--[RSA 3072]----+
|o   .  Eo    |
| o . o .   .  |
|o + o + .   .  |
|.* * = o     o..|
| 0 = o S     + oo|
| . = +       = .|
| . . . . . o + |
| . . . o+     = o|
| .oo ==o+.o   |
+---[SHA256]----+
```

Now, check the generated keys and view the public key using `cat`.

```
ls -al .ssh
```

```
cat .ssh/id_rsa.pub
```

```
1619 ~ ~ >>> cat .ssh/id_rsa.pub iiitd@192.168.2.237
ls -al .ssh
total 24
drwxr-x---  6 meetakshi staff 168 Nov  2 21:25 .
drwxr-x--- 49 meetakshi staff 168 Nov  2 23:02 ..
-rw-r--r--  1 meetakshi staff 2475 Nov  2 21:26 id_rsa
-rw-r--r--  1 meetakshi staff 592 Nov  2 21:26 id_rsa.pub
-rw-r--r--  1 meetakshi staff 565 Nov  2 21:27 known_hosts
1620 ~ ~ >>> cat .ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAQABAAAQgDDfn8eJMV6T4QhH5NrVNsBUnseelkubvh1I0uGJUAWTiE5j8mc0/Jizz8ChTd7h80nduH9zLBUwMgN8m1KnaNuuuuz9E23Fc2b2Wzqmp4/cY4FL9MFZ2kJOXTzI3stgtB66PFNUPrn+49fb0SdJtAMlo152vWc
GDx212+1RgB/x/yVBDA1251w3vsEsut0SWAON4Quq0CFfZSL7R1R6K60/Yet3qnACOnV70JXKD/550hxH1RaWnTs72mfYfKDUvn/VhukXENQhQeOgtt6TBVY60UBAsk8YlCQHxK445jte9HVE+pZj0z2K9Hpp633qeNuNLNm15BwydgVHTZGY/68HE73Ki
6z0MKQuPsgKAT9k326zq9EEChS+5QO/6nNu0z/hsuzLkmf/CfZfy1MuhrwSRydApil2k8e3+sKSwRkSSKieec8EINRkej/UghIRqJ3eY0nM6rDBdqgx/KQKwbNnkWmzQjYzOzNbmc= meetakshi@meetakshi-MacBook-Air.local
```

Now, use ssh-copy-id to propagate the public key to the VM since password auth is currently enabled.

```
ssh-copy-id iiitd@192.168.2.237
```

```
1622 ~ ~ >>> ssh-copy-id iiitd@192.168.2.237
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/Users/meetakshi/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
iiitd@192.168.2.237's password:

Number of key(s) added:      1

Now try logging into the machine, with:  "ssh 'iiitd@192.168.2.237'"
and check to make sure that only the key(s) you wanted were added.
```

Now, as the prompt says, ssh into the VM. A prompt for passphrase appears. I entered the passphrase I used to secure my keys during generation.

```
1624 ~ ~ >>> ssh iiitd@192.168.2.237
Enter passphrase for key '/Users/meetakshi/.ssh/id_rsa':
Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 4.15.0-76-generic x86_64)

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

[ System information as of Tue Nov  2 23:06:20 IST 2021
[

  System load:  0.19          Processes:           113
  Usage of /:   29.9% of 15.68GB   Users logged in:     1
  Memory usage: 44%            IP address for ens32: 192.168.2.237
  Swap usage:   39%

 * Super-optimized for small spaces – read how we shrank the memory
   footprint of MicroK8s to make it the smallest full K8s around.

   https://ubuntu.com/blog/microk8s-memory-optimisation

 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

61 updates can be applied immediately.
36 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

New release '20.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Tue Nov  2 22:20:12 2021 from 192.168.52.246
manpath: can't set the locale; make sure $LC_* and $LANG are correct
```

Now, if I check the list of authorized keys using cat,

```

~ - cat .ssh/authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQCB0oD79gbxwU6x8BGhv17e7eBj3+sc94k5gXZoOpSN8qPyvC4Jmhz19E3ntswNph0DtjzVoxbycnOnnedMiM0muwqBA17xamNqYU0gti1f9QXeeEowEn+taQ1aysvPA8BbnYt8hBCb9Pm0wUdcS+9LW
Qijgnhrtnf/sk7zCRhYo1ksUuF1x66xJmUa8twxxvrv219We6LNqNkgXi+wn20037890uHwJ085DnayPN1seosAR779KivC4K0VMsn0T0zzIrcGcTj1u+jyQf1YCN8C0lQxjHrJadg0AS8Tp1
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQCB0oD79gbxwU6x8BGhv17e7eBj3+sc94k5gXZoOpSN8qPyvC4Jmhz19E3ntswNph0DtjzVoxbycnOnnedMiM0muwqBA17xamNqYU0gti1f9QXeeEowEn+taQ1aysvPA8BbnYt8hBCb9Pm0wUdcS+9LW
ql.sWEFQ-PwEMKMSOBKwK+/EjTA+138u8ghMsCxjpy1kbK+Cq130Tfx9GuuK2Jb+3+5bR7qNNU12z7hSwC6czhb0Ezok4jLkm3tdDDQ+Xen9f3AQhjzkLSHmA1bsh01oKtow/w9adazVkyzqalDz21gJhMU0v95
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQCB0oD79gbxwU6x8BGhv17e7eBj3+sc94k5gXZoOpSN8qPyvC4Jmhz19E3ntswNph0DtjzVoxbycnOnnedMiM0muwqBA17xamNqYU0gti1f9QXeeEowEn+taQ1aysvPA8BbnYt8hBCb9Pm0wUdcS+9LW
CM7264U8SzXlnf+80kp183Xy+VwPy1k56pHPw1kXVfy8Rkhn50a3xVLtTSc1zdf7x1qvXRV9do/lnqSYOccbwlOrohnAL8rOn/oxFwMaRaBhK4hn+4ea5q1udc2N95YKGfF8phnnhd1/H4VKdbsFz1GK19/4.19/
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQCB0oD79gbxwU6x8BGhv17e7eBj3+sc94k5gXZoOpSN8qPyvC4Jmhz19E3ntswNph0DtjzVoxbycnOnnedMiM0muwqBA17xamNqYU0gti1f9QXeeEowEn+taQ1aysvPA8BbnYt8hBCb9Pm0wUdcS+9LW
GDx12+1Rqd/B/y/VBOA1251w3tsvEsu103AONk4Qu9CffzsL7R1R6K60q/VEt3qAGNvft703XX0/56gxh1RaNwTs72ufYGDDBAsK9YCOHx44j1te9H/Eo+pJ3zk9HpgdJ3qeNu1Nm/5BwydgHtZQY//8HBS73ki
Bz0MKQu9SgkAT9k335Zq9rEcHs+5GQC/6eNuZC/HJuzLkmF/CfZy1MuHzwSRVdApil2k8aj+sKBWBkSSKleac8EINRkej/UgHRqJ3eYJY0nW6rD8dQQXkQIKwbN2nKWXzQjzOzNboM= meetakshi@Meetakshi's-MacBook-Air.local

```

My ssh public key is listed here.

## b.

Using (ssh) public keys is a safer method than using passwords. Some of the reasons why this is true are:

- Since keys do not have to be memorized or are human generated, they are long and complex and thus, are more difficult to brute-force or guess than passwords.
- SSH keys are not sent to the server, unlike passwords. If an attacker takes over the server, they can easily view the password if proper security measures are not followed. In ssh, the private key remains on the client side, and no secret value is sent to the server. Hence, keys are in this way more robust than passwords.
- Where a user can use multiple systems during password authentication, authentication using a public key can only come from a system where the private key resides.
- Keys can even be password protected to provide another layer of security.

All in all, keys provide a method of maintaining a degree of entropy and manageability while also providing a method of access that doesn't impede with the actions of a legitimate user.

## c.

My public ssh key is :

```

AAAAAB3NzaC1yc2EAAAQABAAQCB0oD79gbxwU6x8BGhv17e7eBj3+sc94k5gXZoOpSN8qPyvC4Jmhz19E3ntswNph0DtjzVoxbycnOnnedMiM0muwqBA17xamNqYU0gti1f9QXeeEowEn+taQ1aysvPA8BbnYt8hBCb9Pm0wUdcS+9LW
z8ChTd7h8OpdUH9rUBwuMgN0gm1KnaNUuuaz9E23Fc2b2WrqmP4/cY4FL9MFZ2kJGXTZrl3stgtB66PFFNNUPn+49f
bGSdjAMlois52vWcGDx2I2+1LRqB/ky/VBOA125lWE3tsvE5utO3WAONk4Qu9CffzsL7R1RGk6Oq/yEt3qnACGNvf7O
JKXD/550hxBjHjRaNwTSt72mFYGfKDUsn/VhUqKxENQDHQeOgl6TBVY6GDBBAsK0YCGMxK446jdtE9HvEo+pzjGzk9
Hgp6j3qeNuM1LNmY5Bwy6gVHtZGY/6BHE73ki8z0MOKQu9SgkAT9k3j5Zq9rEcHs+5GQC/6eNuZC/HJuzLkmF/CfZ
y1MuHzwSRVdApil2k8aj+sKBWBkSSKleac8EINRkej/UgHRqJ3eYJY0nW6rD8dQQXkQIKwbN2nKWXzQjzOzNboM=

```

I used the yes command to generate the text file having the letter M in each line.I will use it to encrypt the text files.

```
[1628 ~ ~/desktop >> yes M | head -n 10 > out10.txt  
[1629 ~ ~/desktop >> yes M | head -n 10000 > out10k.txt  
[1630 ~ ~/desktop >> yes M | head -n 10000000 > out10m.txt  
1631 ~ ~/desktop >> █
```

1. For AES encryption. The passphrase was my public ssh key.

```
1631 ~ ~/desktop >> openssl aes-256-cbc -salt -a -e -in out10.txt -out aesout10.txt  
enter aes-256-cbc encryption password:  
1633 ~ ~/desktop >> openssl aes-256-cbc -salt -a -e -in out10.txt -out aesout10.txt  
enter aes-256-cbc encryption password:  
Verifying - enter aes-256-cbc encryption password:  
1634 ~ ~/desktop >> openssl aes-256-cbc -salt -a -e -in out10k.txt -out aesout100k.txt  
enter aes-256-cbc encryption password:  
Verifying - enter aes-256-cbc encryption password:  
1635 ~ ~/desktop >> openssl aes-256-cbc -salt -a -e -in out100m.txt -out aesout100m.txt  
enter aes-256-cbc encryption password:  
Verifying - enter aes-256-cbc encryption password:  
1636 ~ ~/desktop >> █
```

2. For RC4 encryption. The passphrase was my public ssh key.

```
1641 ~ ~/desktop >> openssl rc4 -in out100k.txt -out rc4out100k.txt  
enter rc4 encryption password:  
Verifying - enter rc4 encryption password:  
1642 ~ ~/desktop >> openssl rc4 -in out10.txt -out rc4out10.txt  
enter rc4 encryption password:  
Verifying - enter rc4 encryption password:  
1643 ~ ~/desktop >> openssl rc4 -in out100m.txt -out rc4out100m.txt  
enter rc4 encryption password:  
Verifying - enter rc4 encryption password:  
1644 ~ ~/desktop >> █
```

3. RSA gave error with the key

## Question 8.

a.

I configured the firewall settings using iptables - Linux's built-in firewall. The INPUT chain is for all packets meant for the host computer i.e. packets going into local sockets. It controls the behaviour of incoming connections. Since the ping to my machine had to fail (disabled), I added a rule to the INPUT chain to drop all icmp packets from the incoming traffic. I did this using the command-

```
sudo iptables -A INPUT -p icmp --icmp-type echo-request -j REJECT
```

The screenshot below shows that pinging the localhost (loopback interface) failed but my machine could still ping google.com

```
vasilisa@ubuntu:~$ sudo iptables -A INPUT -p icmp --icmp-type echo-request -j REJECT
vasilisa@ubuntu:~$ ping -c 1 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
From 127.0.0.1 icmp_seq=1 Destination Port Unreachable
--- 127.0.0.1 ping statistics ---
1 packets transmitted, 0 received, +1 errors, 100% packet loss, time 0ms

vasilisa@ubuntu:~$ ping -c 1 www.google.com
PING www.google.com (142.250.192.164) 56(84) bytes of data.
64 bytes from del11s11-in-f4.1e100.net (142.250.192.164): icmp_seq=1 ttl=117 time=13.5 ms
--- www.google.com ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 13.575/13.575/13.575/0.000 ms
vasilisa@ubuntu:~$
```

Additionally, here is what the firewall configuration looks like now, with a new rule added to the INPUT chain:

```
vasilisa@ubuntu:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
REJECT    icmp --  anywhere             anywhere            icmp echo-request reject-with icmp-port-unreachable
```

## b.

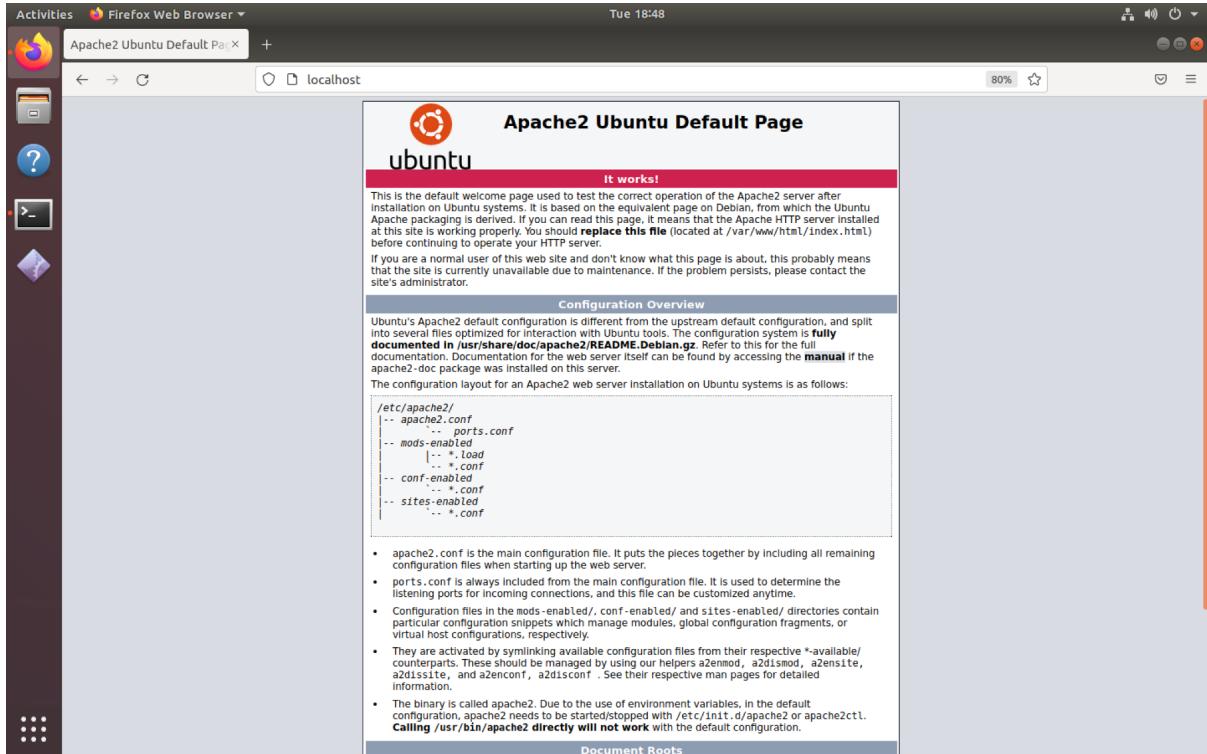
I will host the apache web server on my machine. Apache can be installed using this command:

```
sudo apt-get install apache2
```

Start the server using this command:

```
sudo service apache2 start
```

By default, apache listens on HTTP port 80. If we go to localhost:80 on the host machine, we see the apache default page.



Since my virtual machine and host machine both are in a private network, I can directly access the webpage from my host machine (without doing any port forwarding or such). Now, to only allow my mobile phone to access the web page, I will configure the firewall of my virtual machine using iptables to drop all connections to port 80 of the virtual machine except the one from my mobile phone.

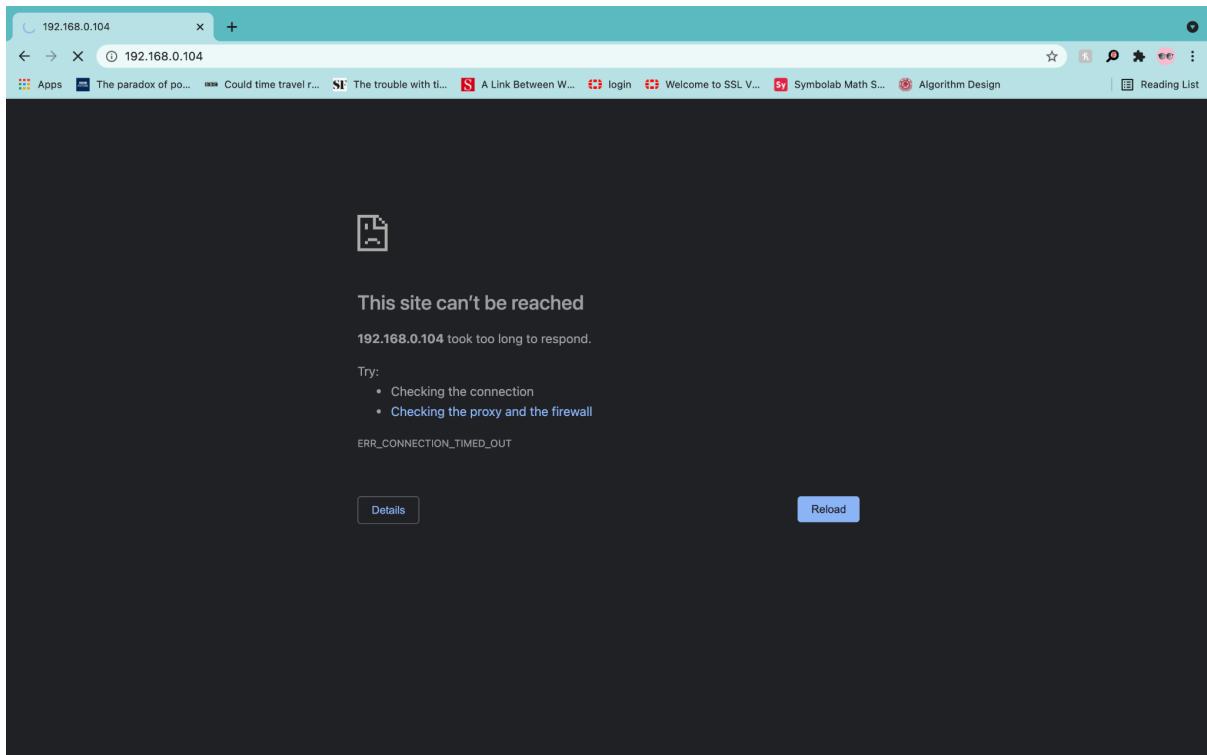
I used the following commands to update the INPUT chain policy of the firewall:

```
sudo iptables -A INPUT -p tcp --dport 80 -s <phone ip address> -j ACCEPT
sudo iptables -A INPUT -p tcp --dport 80 -j DROP
```

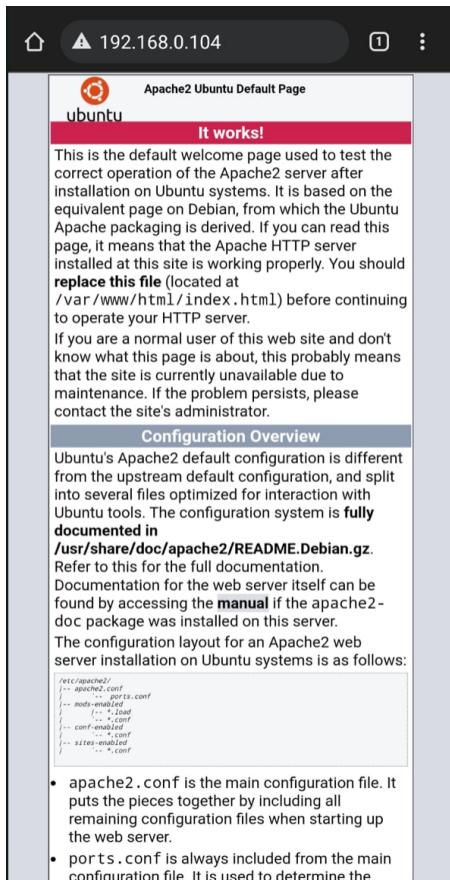
A screenshot of the updated rules-

```
vasilisa@ubuntu:~$ sudo iptables -A INPUT -p tcp --dport 80 -s 192.168.0.103 -j ACCEPT
vasilisa@ubuntu:~$ sudo iptables -A INPUT -p tcp --dport 80 -j DROP
vasilisa@ubuntu:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target      prot opt source          destination
ACCEPT      tcp  --  192.168.0.103    anywhere        anywhere        tcp dpt:http
DROP       tcp  --  anywhere        anywhere        tcp dpt:http
```

Now, if I access the webpage using my laptop, I am not able to.



Whereas, on my mobile, I see the default apache web page.



## Question 9.

Screenshots of WebGoat assignments:

### 1. Introduction

#### 1.1. WebGoat

The screenshot shows the Firefox browser window with the URL `localhost:8080/WebGoat/start.mvc#lesson/WebWolfIntroduction.lesson/3`. The title bar says "Activities Firefox Web Browser" and the time is "Tue 17:52". The main content area displays the "Landing page" for the "WebWolf" challenge. It includes a sidebar with navigation links like "General", "(A1) Injection", "(A2) Broken Authentication", etc. The main content area has a "Reset lesson" button and a "Requests" section showing a JSON payload. A note at the bottom says "For this exercise you need to login to WebWolf first." and "Please be aware that after resetting the password the user will receive an error page. In a real attack scenario the user would probably see a normal success page (this is due to a limit what we can control with WebGoat)".

#### 1.2. WebWolf

The screenshot shows the Firefox browser window with the URL `localhost:8080/WebGoat/start.mvc#lesson/WebGoatIntroduction.lesson`. The title bar says "Activities Firefox Web Browser" and the time is "Tue 17:50". The main content area displays the "What is WebGoat?" section. It includes a sidebar with navigation links like "General", "(A1) Injection", "(A2) Broken Authentication", etc. The main content area has a "Reset lesson" button and a "What is WebGoat?" section with text about the application's purpose and a note: "Now, while we in no way condone causing intentional harm to any animal, goat or otherwise, we think learning everything you can about security vulnerabilities is essential to understanding just what happens when even a small bit of unintended code gets into your applications." There is also a "The WebGoat Team" link.

## 2. General

### 2.1. HTTP Basics

The screenshot shows a Firefox browser window with three tabs open: 'localhost:9090/WebGoat/lesson/1', 'localhost:8080/WebGoat/start.mvc#lesson/HttpBasics.lesson/2', and 'localhost:8080/WebGoat/start.mvc#lesson/HttpBasics.lesson/2'. The main content area displays the 'HTTP Basics' lesson from WebGoat. The sidebar on the left lists various challenges, with 'HTTP Basics' currently selected. The main content area shows a quiz question: 'What type of HTTP command did WebGoat use for this lesson. A POST or a GET.' The user has answered 'POST' and provided the magic number '72'. A success message at the bottom says 'Congratulations. You have successfully completed the assignment.'

### 2.2. HTTP Proxies

The screenshot shows a Firefox browser window with three tabs open: 'localhost:9090/WebGoat/lesson/1', 'localhost:8080/WebGoat/start.mvc#lesson/HttpProxies.lesson/5', and 'localhost:8080/WebGoat/start.mvc#lesson/HttpProxies.lesson/5'. The main content area displays the 'HTTP Proxies' lesson from WebGoat. The sidebar on the left lists various challenges, with 'HTTP Proxies' currently selected. The main content area shows instructions for configuring a breakpoint filter in ZAP. A 'Break Points' dialog box is open, showing a configuration for a 'RequestHeader' breakpoint that matches 'POST' and ignores case. A note at the bottom of the page cautions against using the green/red button to deactivate breakpoints.

## 2.3. Developer Tools

The screenshot shows a Firefox browser window with the title bar "Activities Firefox Web Browser" and the address bar showing "localhost:8080/WebGoat/start.mvc#lesson/ChromeDevTools.lesson/5". The main content area displays the "Developer Tools" lesson from WebGoat. On the left is a sidebar with icons for Introduction, General, HTTP Basics, HTTP Proxies, Developer Tools (which is selected), CIA Triad, Crypte Basics, and Writing new lesson. The main content area has a heading "Developer Tools" and a sub-section "Try It! Working with the Network tab". It contains instructions to find a specific HTTP request and read a randomized number from it. A button labeled "Click this button to make a request: Go!" is shown, along with a text input field containing "67.87810684624279" and a "check" button. Below this, a message says "Correct, Well Done."

## 2.4. CIA Triad

The screenshot shows a Firefox browser window with the title bar "Activities Firefox Web Browser" and the address bar showing "localhost:8080/WebGoat/start.mvc#lesson/CIA.lesson/4". The main content area displays the "CIA Triad" lesson from WebGoat. On the left is a sidebar with icons for Introduction, General, HTTP Basics, HTTP Proxies, Developer Tools (selected), CIA Triad (selected), Crypte Basics, and Writing new lesson. The main content area has a heading "CIA Triad" and a "Reset lesson" button. It contains a quiz section with four numbered questions about how an intruder can harm the CIA security goals of confidentiality, integrity, and availability. Each question has multiple choice options with some being correct (indicated by a blue checkmark). At the bottom, there is a "Submit answers" button and a message "Congratulations. You have successfully completed the assignment."

## 2.5. Crypto basics

The screenshot shows the 'Crypto Basics' lesson page from the WebGoat application. The left sidebar lists various security topics, and the main content area displays the 'Crypto Basics' section. It includes sections on Java cacerts, protecting private keys, and assignments. An assignment involves retrieving a secret message from a Docker container and decrypting it using OpenSSL. The user has entered the command 'docker run -d webgoat/assignments:findthesecret' and the decrypted message 'U2FsdGVkX199jgh5oANEIfdtCxEvIcLiI+v+5loE+VCuy6I0b+5byb5DXp32RpmT02E1p1f55cQN+DHbwCP1VRFQanOmhBUp07as=' into a text input field. Below this, the user has entered 'default\_secret' into a 'post the answer' field and clicked 'post the answer'. A success message 'Congratulations, you did it!' is displayed.

## 2.6. Writing a new lesson

The screenshot shows the 'Writing new lesson' page from the WebGoat application. The left sidebar lists various security topics, and the main content area displays the 'Step 4: Add an assignment to your lesson' section. It provides an example of a Spring REST controller for an assignment. The code defines a class 'SampleAttack' that extends 'AssignmentEndpoint'. It includes annotations like @RestController, @Autowired, and @PostMapping. The controller handles a POST request to '/lesson-template/sample-attack' and returns a response based on the value of 'param1'. The code uses UserSessionData to store session information and outputs feedback messages. A note at the bottom states that every assignment is just a Spring RestController.

## (A1) Injection

### 1. SQL Injection (intro)

The screenshot shows a Firefox browser window with three tabs open: "localhost:9090/WebWolf/reload" (active), "localhost:8080/WebGoat/Wireshark", and "localhost:8080/WebGoat/start.mvc#lesson/SqlInjection.lesson/12". The main content area displays the "SQL Injection (intro)" lesson from the WebGoat application. The sidebar on the left lists various security topics under "A1 Injection", with "SQL Injection (intro)" currently selected. The main content area has a title "Compromising Availability" and a paragraph explaining that after compromising confidentiality and integrity, the goal is now to compromise availability. It notes that deleting or changing accounts, or dropping entire databases, violates availability. Below this, a section titled "It is your turn!" instructs the user to delete the "access\_log" table. A text input field contains the SQL command: "Action contains: ';' drop table access\_log; --". A button labeled "Search logs" is present. A success message at the bottom states: "Success! You successfully deleted the access\_log table and that way compromised the availability of the data."

### 2. SQL Injection (advanced)

SQL injection (advanced)

Now it is time for a quiz! It is recommended to do all SQL injection lessons before trying the quiz. Answer all questions correctly to complete the assignment.

- What is the difference between a prepared statement and a statement?
  - Solution 1: Prepared statements are statements with hard-coded parameters.
  - Solution 2: Prepared statements are not stored in the database.
  - Solution 3: A statement is faster.
  - Solution 4: A statement has got values instead of a prepared statement
- Which one of the following characters is a placeholder for variables?
  - Solution 1: \*
  - Solution 2: =
  - Solution 3: ?
  - Solution 4: !
- How can prepared statements be faster than statements?
  - Solution 1: They are not static so they can compile better written code than statements.
  - Solution 2: Prepared statements are compiled once by the database management system waiting for input and are pre-compiled this way.
  - Solution 3: Prepared statements are stored and wait for input it raises performance considerably.
  - Solution 4: Oracle optimized prepared statements. Because of the minimal use of the databases resources it is faster.
- How can a prepared statement prevent SQL-injection?
  - Solution 1: Prepared statements have got an inner check to distinguish between input and logical errors.
  - Solution 2: Prepared statements use the placeholders to make rules what input is allowed to use.
  - Solution 3: Placeholders can prevent that the users input gets attached to the SQL query resulting in a separation of code and data.
  - Solution 4: Prepared statements always read inputs literally and never mixes it with its SQL commands.
- What happens if a person with malicious intent writes into a register form :Robert); DROP TABLE Students;-- that has a prepared statement?
  - Solution 1: The table Students and all of its content will be deleted.
  - Solution 2: The input deletes all students with the name Robert.
  - Solution 3: The database registers 'Robert' and deletes the table afterwards.
  - Solution 4: The database registers 'Robert' ). DROP TABLE Students;--.

Submit answers

Congratulations. You have successfully completed the assignment.

### 3. SQL injection (mitigation)

SQL Injection (mitigation)

In this assignment try to perform an SQL injection through the ORDER BY field. Try to find the ip address of the [webgoat-prd](#) server, guessing the complete ip address might take too long so we give you the last part: [xxx.130.219.202](#).

Note: The submit field of this assignment is NOT vulnerable to an SQL injection.

	Hostname	IP	MAC	Status	Description
<input type="checkbox"/>	webgoat-tst	192.168.2.1	EE:FF:33:44:AB:CD	success	Test server
<input type="checkbox"/>	webgoat-acc	192.168.3.3	EF:12:FE:34:AA:CC	danger	Acceptance server
<input type="checkbox"/>	webgoat-dev	192.168.4.0	AA:BB:11:22:CC:DD	success	Development server
<input type="checkbox"/>	webgoat-pre-prod	192.168.6.4	EF:12:FE:34:AA:CC	danger	Pre-production server

IP address webgoat-prd server: 104.130.219.202

Submit

Congratulations. You have successfully completed the assignment.

### 4. Path Traversal

The screenshot shows a Firefox browser window with the URL `localhost:8080/WebGoat/start.mvc#lesson/PathTraversal.lesson/`. The page title is "Path traversal". On the left, there's a sidebar with a navigation tree for the WebGoat application, including sections like "Introduction", "General", "(A1) Injection", "(A2) Broken Authentication", and "Path traversal". The main content area displays a "Zip Slip assignment" challenge. It features a placeholder for a user profile picture and a form with fields for "Full Name" (containing "test"), "Email" (containing "test@test.com"), and "Password" (containing "\*\*\*\*"). A blue "Update" button is at the bottom of the form. Below the form, a message says "Congratulations. You have successfully completed the assignment." The Firefox developer tools are visible at the bottom of the browser window.

## (A2) Broken Authentication

### 1. Authentication Bypass

The screenshot shows a Firefox browser window with the URL `https://localhost:8080/WebGoat/start.mvc#lesson/AuthBypass.lesson/`. The page title is "Authentication Bypasses". On the left, there's a sidebar with a navigation tree for the WebGoat application, including sections like "Introduction", "General", "(A2) Injection", and "(A2) Broken Authentication". The main content area displays an "Auth Bypasses" challenge under "2FA Password Reset". It shows a "Step 2: Enter any answer for security questions." form with a dropdown menu for "Answer security question" and a text input field for "What's the name of your first pet?". Below this is a "Continue" button. Further down, there's a "Step 3: Using a proxy, remove "securityQuestion" and "securityQuestionId" from the post data." section with a complex URL. At the bottom, a success message reads: "Congrats, you have successfully verified the account without actually verifying it. You can now change your password!"

## 2. JWT Tokens

Activities Firefox Web Browser Sun 23:35

WebGoat 80%

localhost:8080/WebGoat/start.mvc#lesson/JWT.lesson/10

**JWT tokens**

Show hints Reset lesson

1 2 3 4 5 6 7 8 9 10 11 12

**Final challenge**

Below you see two accounts, one of Jerry and one of Tom. Jerry wants to remove Tom's account from Twitter, but his token can only delete his account. Can you try to help him and delete Tom's account?

St. ... M. ... Domain File Initiator Type Transferred Size Headers Cookies Request Response Timings Stack Trace

200	GET	localhost...	lessononview.mvc	jquery.m...	json	1.07 KB	86...	200	POST http://localhost:8080/WebGoat/JWT/final/delete?token=eyJ0eXAiOiJKV1QiLCJrWQoOigCaFja2ViJyBtVklPTiBzXWVjY3QgU1pHvnNaWFJw1m1kVWVhM0DhIGzb20gSU5GT1JnQV8T05fUuNIRU1BLNz1JFTV99UVvSjyA1LshmFr2yjikhTMjU1n0JpJz1McijxVzJhB2F0lRva2VuE1iaWkkZ2XILCj9YQj0E1MjQyHTA5MDQslm4cG16MjYxODkwNTMwNCw1Xvkjod22z29hdc5vcmclCjzwDlOij0b21A2d2Vz29hdCjz20lCj1c2vYbmPfZS16Rvb5ItkVtWbjodcPQHd5YmdvYxQY29tiwUm9z5i6WjYD7QXk0.eJbl_vuokmhTciU8bd06mW-lWkxBkOBmN_E4MEz1
200	GET	localhost...	lessonmenu.mvc	jquery.m...	json	7.40 KB	7.1...	200	Status 200 OK
200	GET	localhost...	lessonmenu.mvc	jquery.m...	json	1.07 KB	86...	200	Version HTTP/1.1
200	GET	localhost...	lessonmenu.mvc	jquery.m...	json	7.40 KB	7.1...	200	Transferred 425 B (196 B size)
200	GET	localhost...	lessonmenu.mvc	jquery.m...	json	1.07 KB	86...	200	Referrer Policy unsafe-url

Response Headers (229 B) Connection: keep-alive

4479 requests | 22.77 MB / 19.85 MB transferred | Finish: 205.71 min

### 3. Password Reset

Activities Firefox Web Browser ▾

Mon 00:18

WebGoat x WebWolf +

localhost:8080/WebGoat/start.mvc#lesson/PasswordReset.lesson/5 80% ⌂ ⓘ ⓘ ⓘ

## Password reset

Show hints Reset lesson

1 2 3 4 5 6 7 8

### Creating the password reset link

When creating a password reset link you need to make sure:

- It is a unique link with a random token
- It can only be used once
- The link is only valid for a limited amount of time.

Sending a link with a random token means an attacker cannot start a simple DOS attack to your website by starting to block users. The link should not be usable more than once which makes it impossible to change the password again. The time out is necessary to restrict the attack window, having a link opens up a lot of possibilities for the attacker.

### Assignment

Try to reset the password of Tom ([tom@webgoat-cloud.org](mailto:tom@webgoat-cloud.org)) to your own choice and login as Tom with that password. Note: it is not possible to use OWASP ZAP for this lesson, also browsers might not work, command line tools like curl and the like will be more successful for this attack.

Tom always resets his password immediately after receiving the email with the link.

Account Access

@ tom@webgoat-cloud.org

\*\*\*\*\*

Access

Forgot your password?

Congratulations. You have successfully completed the assignment.



#### 4. Secure passwords

Secure Passwords

How long could it take to brute force your password?

In this assignment you have to type in a password which is strong enough (at least 4/4).

After you finished this assignment we highly recommend you to try some of the passwords below to see why they are no good choices:

- password
- johnsmith
- 2018/10/4
- 1992home
- abcabc
- ffgff
- pouz
- @dmin

**>Password**   Show password

You have succeeded! The password is secure enough.

Your Password: \*\*\*\*\*  
Length: 35  
Estimated guesses needed to crack your password: 1652001119999999800000000000000000  
Score: 4/4  
Estimated cracking time: 292471208677 years 195 days 15 hours 30 minutes 7 seconds  
Score: 4/4  
Estimated cracking time in seconds: 292471208677 years 195 days 15 hours 30 minutes 7 seconds

## (A3) Sensitive Data Exposure

### 1. Insecure Login

Insecure Login

Let's try

Click the "log in" button to send a request containing login credentials of another user. Then, write these credentials into the appropriate fields and submit to confirm. Try using a packet sniffer to intercept the request.

CaptainJack

Congratulations. You have successfully completed the assignment.

Network Tab (Developer Tools):

Stage	M...	Domain	File	Initiator	Type	Transferred	Size
200	GET	localhost...	lessonoverview.mvc	jquery.mi...	json	382 B	15...
200	GET	localhost...	lessonmenu.mvc	jquery.mi...	json	7.40 KB	7.1...
200	GET	localhost...	lessonoverview.mvc	jquery.mi...	json	382 B	15...
200	GET	localhost...	lessonmenu.mvc	jquery.mi...	json	7.40 KB	7.1...
200	GET	localhost...	lessonoverview.mvc	jquery.mi...	json	382 B	15...
200	GET	localhost...	lessonmenu.mvc	jquery.mi...	json	7.40 KB	7.1...
200	GET	localhost...	lessonoverview.mvc	jquery.mi...	json	382 B	15...

Request Headers:

- password: "BlackPearl"
- username: "CaptainJack"

## (A4) XML External Entities

### 1. XXE

The screenshot shows a Firefox browser window titled "Activities Firefox Web Browser". The address bar shows "localhost:8080/WebGoat/start.mvc#lesson/XXE.lesson/10". The main content area displays the "XXE" assignment from WebGoat. It includes a sidebar with navigation links like Introduction, General, (A1) Injection, etc. The main content area has a heading "Blind XXE assignment" and instructions about using WebWolf to serve a DTD. Below this is a "Photo" section showing a black and white cat with a hanger around its neck, with the text "HUMAN" above it and "I REQUEST YOUR ASSISTANCE" below it. At the bottom, there's a comment section with a "Submit" button.

## (A5) Broken Access Control

### 1. Insecure Direct Object References

The screenshot shows a Firefox browser window titled "Activities Firefox Web Browser". The address bar shows "localhost:8080/WebGoat/IDOR.lesson/4". The main content area displays the "Insecure Direct Object References" assignment from WebGoat. It includes a sidebar with navigation links like Introduction, General, (A1) Injection, etc. The main content area has a heading "Playing with the Patterns" and instructions about viewing another profile. Below this is a "View Profile" button and an "Edit Another Profile" section with instructions. At the bottom, there's a "Network" tab in the developer tools showing a list of network requests, including a PUT request to "/WebGoat/IDOR/profile/2342388".

## 2. Missing Function Level Access Control

The screenshot shows a Firefox browser window with the title 'WebGoat'. The URL is 'localhost:8080/WebGoat/start.mvc#lesson/MissingFunctionAC.lesson/2'. The page content is titled 'Missing Function Level Access Control'. On the left, there's a sidebar with navigation links for various security topics. The main area has a heading 'Just Try It' and a note about how some applications use client-side controls for access control. Below this is a form field labeled 'Your Hash:' containing the value '7WbCu+MrABC6doU+XGU='.

**Network Tab (Firefox DevTools):**

Sta	Me	Domain	File	Initiator	Typ	Trans.	Size
200	GET	loc...	lessonoverview.mvc	jquer...	j...	567 B	338
200	GET	loc...	lessonmenu.mvc	jquer...	j...	7.40 KB	7.17
200	GET	loc...	lessonoverview.mvc	jquer...	j...	567 B	338
200	GET	loc...	lessonmenu.mvc	jquer...	j...	7.40 KB	7.17
200	GET	loc...	lessonoverview.mvc	jquer...	j...	567 B	338
200	GET	loc...	users	NetUt...	j...	351 B	123
200	GET	loc...	lessonmenu.mvc	jquer...	j...	7.40 KB	7.17

At the bottom of the Network tab, it says '393 requests | 1.46 MB / 1.55 MB transferred | Finish: 15.52 min'.

## (A7) Cross-Site Scripting (XSS)

### 1. Cross-Site Scripting

The screenshot shows a Firefox browser window with the title 'WebGoat'. The URL is 'localhost:8080/WebGoat/start.mvc#lesson/CrossSiteScripting.lesson/11'. The page content is a quiz titled 'Reset lesson'. It contains several questions with multiple-choice answers. The first question is 'Are trusted websites immune to XSS attacks?' with four options: Solution 1 (checkbox), Solution 2 (checkbox), Solution 3 (checkbox), and Solution 4 (radio button). The second question is 'When do XSS attacks occur?' with four options: Solution 1 (checkbox), Solution 2 (checkbox), Solution 3 (radio button), and Solution 4 (checkbox). The third question is 'What are Stored XSS attacks?' with four options: Solution 1 (checkbox), Solution 2 (checkbox), Solution 3 (checkbox), and Solution 4 (checkbox). The fourth question is 'What are Reflected XSS attacks?' with four options: Solution 1 (checkbox), Solution 2 (checkbox), Solution 3 (checkbox), and Solution 4 (radio button). The fifth question is 'Is JavaScript the only way to perform XSS attacks?' with four options: Solution 1 (checkbox), Solution 2 (checkbox), Solution 3 (checkbox), and Solution 4 (radio button). At the bottom, there's a 'Submit answers' button and a message 'Congratulations. You have successfully completed the assignment.'

## (A8) Insecure Deserialization

There is an application error in this exercise:

<https://github.com/WebGoat/WebGoat/issues/1129>

## (A9) Vulnerable Components

There is an application error in part 12 of this exercise. I just did part 5.

<https://github.com/WebGoat/WebGoat/issues/1134>

The screenshot shows a Firefox browser window with the title 'WebGoat'. The address bar indicates the URL is `localhost:8080/WebGoat/start.mvc#lesson/VulnerableComponents.lesson/4`. The sidebar on the left lists various challenges, with 'Vulnerable Components' currently selected. The main content area displays a slide titled 'The exploit is not always in "your" code'. It compares two versions of the jquery-ui library: 'jquery-ui:1.10.4' and 'jquery-ui:1.12.0 Not Vulnerable'. The 1.10.4 section shows a code snippet where clicking 'OK' in a dialog executes `OK<script>alert('XSS')</script> Go!`, leading to an XSS attack. The 1.12.0 section states that this exploit was prevented by an upgrade to a non-vulnerable version of the library.

## (A8) Request Forgeries

### 1. Cross-site Request Forgeries

## 2. Server-side Request Forgeries

## Client Side

## 1. Bypass Front End Restrictions

Activities Firefox Web Browser

Mon 06:24

WebGoat

localhost:8080/WebGoat/start.mvc#lesson/BypassRestrictions.lesson/2

80% 80%

Bypass front-end restrictions

Reset lesson

Validation

Often, there is some mechanism in place to prevent users from sending altered field values to server, such as validation before sending. Most of popular browsers such as Chrome don't allow editing scripts during runtime. We will have to circumvent the validation some other way.

Task

Send a request that does not fit the regular expression above the field in all fields.

Field 1: exactly three lowercase characters("a-z){3}\$

abc

Field 2: exactly three digits("0-9){3}\$

123

Field 3: letters, numbers, and space only"(a-zA-Z0-9 )\$

abc 123 ABC

Inspector Console Debugger Network Style Editor Performance Memory Storage Accessibility Application

Headers Cookies Request Response Timings Stack Trace

Filter URLs

Sta Me Domain File Initiator Typ Trans... Siz... Headers Cookies Request Response Timings Stack Trace

200 GET /loc... lessonmenu.mvc jquer... j... 608 B 379

200 GET /loc... lessonmenu.mvc jquer... j... 608 B 379

200 GET /loc... lessonmenu.mvc jquer... j... 608 B 379

200 GET /loc... lessonmenu.mvc jquer... j... 608 B 379

200 POST /loc... /WebGoat/BypassRestrictionsNetU... j... 445 B 216

200 GET /loc... lessonmenu.mvc jquer... j... 607 B 378

200 GET /loc... lessonmenu.mvc jquer... j... 7.39 KB 7.17

200 GET /loc... lessonmenu.mvc jquer... j... 607 R 378

112 requests 408.05 KB / 433.09 KB transferred | Finish: 4.44 min

## 2. Client-Side Filtering

Activities Firefox Web Browser

Mon 06:38

WebGoat

localhost:8080/WebGoat/start.mvc#lesson/ClientSideFiltering.lesson/2

80% 80%

Client side filtering

Reset lesson

One of the responses contains the answer

No need to pay if you know the code ...

Samsung Galaxy S8

Samsung - (124421 reviews)

PRICE US \$0.00

COLOR: Black

CAPACITY: 64 GB / 128 GB

QUANTITY: 1

CHECKOUT CODE:

Buy

Congratulations. You have successfully completed the assignment.

## 3. HTML Tampering

The screenshot shows the 'HTML tampering' challenge in the WebGoat application. The main page title is 'HTML tampering'. Below it, there's a section titled 'Try it yourself' with the sub-instruction: 'In an online store you ordered a new TV, try to buy one or more TVs for a lower price.' A shopping cart summary is shown:

Product	Quantity	Price	Total
55" M5510 White Full HD Smart TV by Samsung	1	\$2999.99	\$2999.99

Below the cart, the total is \$2999.99 and shipping costs are \$0.00. The developer tools Network tab shows a list of requests, with the last request being a POST to 'lessonoverview.mvc' with the following JSON response:

```

{
  "lessonCompleted": true,
  "feedback": "Well done, you just bought a TV at a discount!",
  "output": null,
  "assignment": "HtmlTamperingTask",
  "attemptWasMade": true
}

```

## Challenges

### 1. Admin lost password

The screenshot shows the 'Admin lost password' challenge in the WebGoat application. The main page title is 'Admin lost password'. The page contains a 'Reset lesson' button and a 'Sign in' button. Below the sign-in button is a text input field containing a flag: '5d57a19a-02c0-4a63-b236-9c56d33dc1ff'. At the bottom of the page, a success message reads: 'Congratulations you have solved the challenge!!'

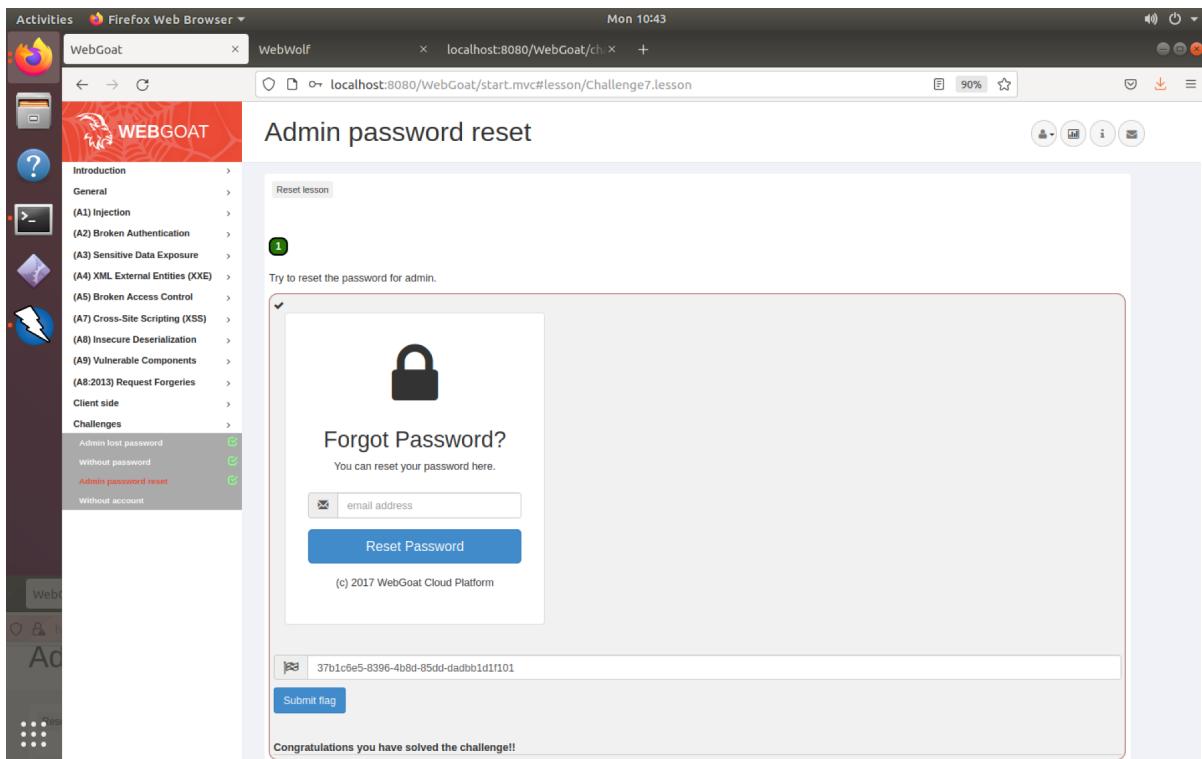
Observation: The password was in the logo file. I downloaded the logo shown and opened it in a text editor.

## 2. Without password

The screenshot shows a Firefox browser window titled 'WebGoat'. The URL is 'localhost:8080/WebGoat/start.mvc#'. The main content area displays the 'Without password' challenge. On the left, there's a sidebar with navigation links like 'Introduction', 'General', '(A1) Injection', etc. A specific link '(A9) Vulnerable Components' is highlighted with a red box. The main content area has a 'Reset lesson' button and a note: 'Can you login as Larry?'. Below is a 'LOGIN' form with fields for 'username' (containing 'Larry') and 'password' (containing '\*\*\*\*\*'). There's a 'Remember me' checkbox and a 'Log In' button. Below the form is a 'Forgot Password?' link. At the bottom, there's a text input field containing '504fb108-3dc9-4eff-b2dc-fb54d018fa81' and a 'Submit flag' button. A message at the bottom says 'Congratulations you have solved the challenge!!'

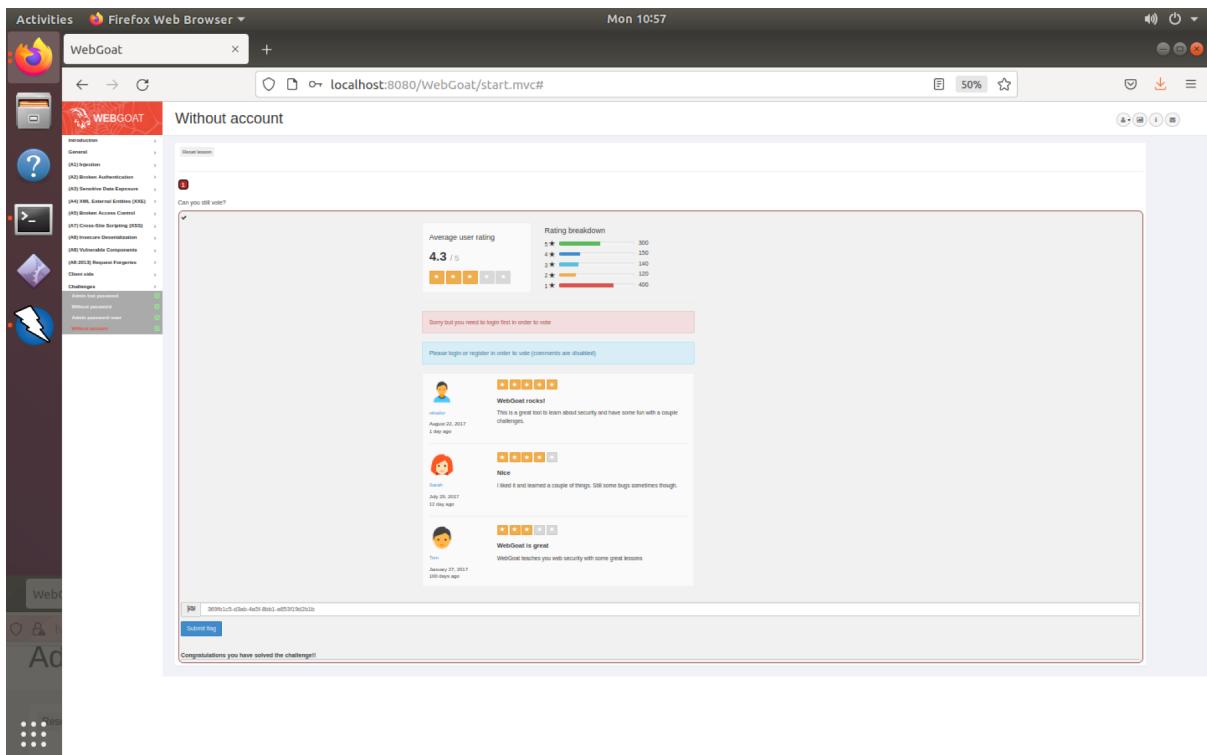
Observation: Sending an invalid request showed the SQL query being used. A simple SQL injection in the password field (`p' or '1'='1`) completed the assignment.

## 3. Admin Password reset



Observation: I obtained the admin-password-reset constant token from  
<https://github.com/WebGoat/WebGoat/blob/develop/webgoat-lessons/challenge/src/main/java/org/owasp/webgoat/challenges/SolutionConstants.java>  
After this, I changed the token of the link in mail box to the admin password reset link and made a new link:  
<http://localhost:8080/WebGoat/challenge/7/reset-password/375afe1104f4a487a73823c50a9292a2>

#### 4. Without Account



Observation: An OPTIONS request at the /vote/{num} endpoint showed that only HEAD/GET/OPTIONS request was valid. HEAD request at the /vote/{num} endpoint did the deed.