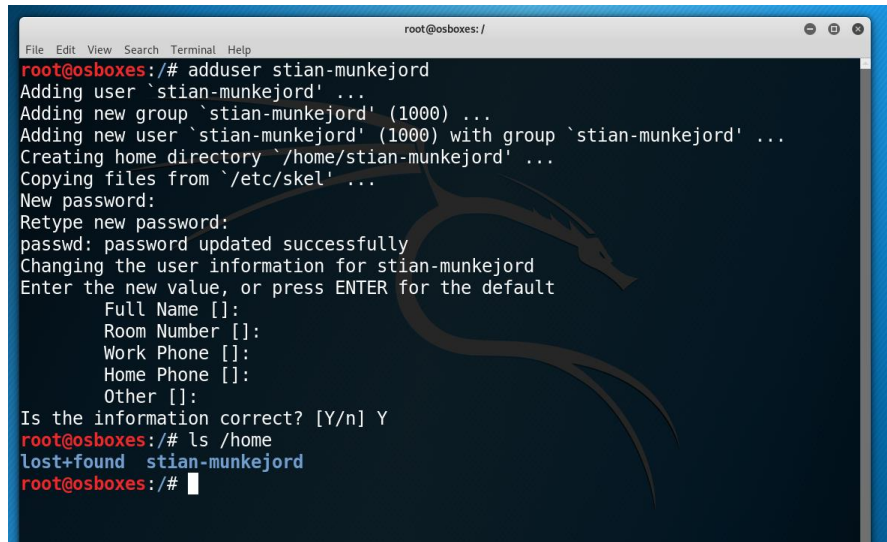


# Mandatory Assignment 2

## User Authentication (15 pts)

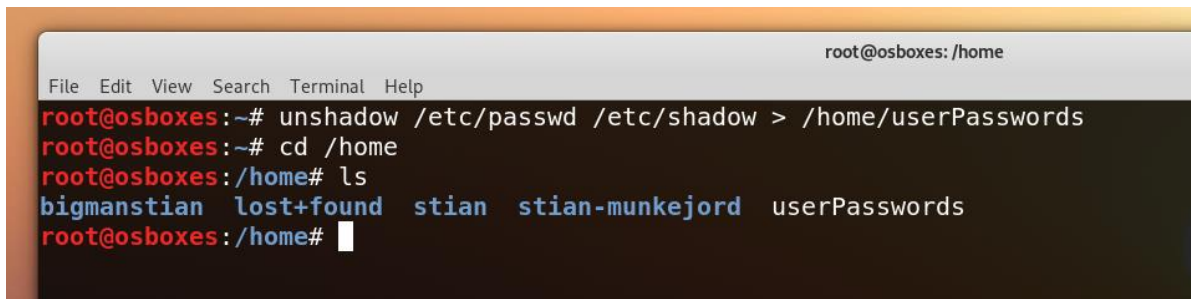
### Question 1.

- I. Add yourself (firstname-lastname) as a new user with a weak password in Kali Linux (Take a screenshot of the home directory);



```
root@osboxes: /
root@osboxes:~# adduser stian-munkejord
Adding user `stian-munkejord' ...
Adding new group `stian-munkejord' (1000) ...
Adding new user `stian-munkejord' (1000) with group `stian-munkejord' ...
Creating home directory `/home/stian-munkejord' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for stian-munkejord
Enter the new value, or press ENTER for the default
Full Name []:
Room Number []:
Work Phone []:
Home Phone []:
Other []:
Is the information correct? [Y/n] Y
root@osboxes:~# ls /home
lost+found  stian-munkejord
root@osboxes:~#
```

- II. Locate the password file of all users in Kali Linux and print its content (Take a screenshot of the home directory);



```
root@osboxes: /home
root@osboxes:~# unshadow /etc/passwd /etc/shadow > /home/userPasswords
root@osboxes:~# cd /home
root@osboxes:~# ls
bigmanstian  lost+found  stian  stian-munkejord  userPasswords
root@osboxes:~#
```

```

root@osboxes: /home# cat userPasswords
root:$6$08uKtWw/dpata2sE1B4j/HJuiw2lsuT7yBvTh3fioWj9KkUvPQT/10JT4rtBACAm4N1EFV4n4x6ndTN3wD9ASuH0jEQ0/1JqN.:0:0:root:/root:/bin/bash
daemon:*:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:*:2:2:bin:/bin:/usr/sbin/nologin
sys:*:3:3:sys:/dev:/usr/sbin/nologin
sync:*:4:65534:sync:/bin:/bin/sync
games:*:5:60:games:/usr/games:/usr/sbin/nologin
man:*:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:*:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:*:8:8:mail:/var/mail:/usr/sbin/nologin
news:*:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:*:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:*:11:11:proxy:/bin:/usr/sbin/nologin
www-data:*:33:33:www-data:/var/www:/usr/sbin/nologin
backup:*:34:34:backup:/var/backups:/usr/sbin/nologin
list:*:38:38:Mail Manager:/var/list:/usr/sbin/nologin
ircd:*:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:*:41:41:Gnats Bug-Reporting System (admin)/var/lib/gnats:/usr/sbin/nologin
nobody:*:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:*:100:65534:/nonexistent:/usr/sbin/nologin
systemd-timesync:*:101:102:systemd Time Synchronization,,:/run/systemd:/usr/sbin/nologin
systemd-networkd:*:102:103:systemd Network Management,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:*:103:104:systemd Resolver,,:/run/systemd:/usr/sbin/nologin
mysql:*:104:10:MySQL Server,,:/nonexistent:/bin/false
ntp:*:105:111:/nonexistent:/usr/sbin/nologin
messagebus:*:106:112:/nonexistent:/usr/sbin/nologin
arpwatch:*:107:114:ARP Watcher,,:/var/lib/arpwatch:/bin/sh
Debian-exim:*:108:115:/var/spool/exim4:/usr/sbin/nologin
uuidd:*:109:116:/run/uuidd:/usr/sbin/nologin
redsocks:*:110:117:/var/run/redsocks:/usr/sbin/nologin
tss:*:111:118:TPM2 software stack,,:/var/lib/tpm:/bin/false
nfsd:*:112:65534:/var/spool/nfs:/usr/sbin/nologin
iodine:*:113:65534:/var/run/iodine:/usr/sbin/nologin
stunnel4:*:114:121:/var/run/stunnel4:/usr/sbin/nologin
miredo:*:115:65534:/var/run/miredo:/usr/sbin/nologin
dnsmasq:*:116:65534:dnsmasq,,:/var/lib/misc:/usr/sbin/nologin
sshd:*:117:123:/nonexistent:/usr/sbin/nologin
postgres:*:118:125:PostgreSQL administrator,,:/var/lib/postgresql:/bin/bash
usbmux:*:119:46:usbmux daemon,,:/var/lib/usbmux:/usr/sbin/nologin
rkt:*:120:127:RealtimeKit,,:/proc:/usr/sbin/nologin
rpc:*:121:65534:/run/rpcbind:/usr/sbin/nologin
Debian-snmpp:*:122:129:/var/lib/snmpp:/bin/false
stard:*:123:65534:/var/lib/nfs:/usr/sbin/nologin
inetd:*:124:130:/var/lib/inetd:/usr/sbin/nologin
sshd:*:125:65534:/run/sshd:/usr/sbin/nologin
pulse:*:126:132:PulseAudio daemon,,:/var/run/pulse:/usr/sbin/nologin
speech-dispatcher:*:127:129:Speech Dispatcher,,:/var/run/speech-dispatcher:/bin/false
avahi:*:128:135:Avahi mDNS daemon,,:/var/run/avahi-daemon:/usr/sbin/nologin
saned:*:129:136:/var/lib/saned:/usr/sbin/nologin
colord:*:130:138:colord colour management daemon,,:/var/lib/colord:/usr/sbin/nologin
gocue:*:131:139:/var/lib/gocue:/usr/sbin/nologin
king-phisher:*:132:140:/var/lib/king-phisher:/usr/sbin/nologin
Debian-gdm:*:133:141:Gnome Display Manager:/var/lib/gdm3:/bin/false
dradis:*:134:142:/var/lib/dradis:/usr/sbin/nologin
beef-xxs:*:135:143:/var/lib/beef-xxs:/usr/sbin/nologin
systemd-coredump:*:136:144:systemd Core Dumper:/usr/sbin/nologin
stian-munkejord:$6$0cJ4H21zfA3pABdH8hTzcVqBcArYUEGyatce5ponCtXVuaCwxhvg3re5VzV1zxYbScYgV0RZUK1HEHntw16s0c1vA1B0Mq1KMAPX/:1000:1000:,,:/home/stian-munkejord:/bin/bash
root@osboxes: /home#

```

- III. Try to use the tool John Ripper (or Hashcat) to crack your password file. If it takes too long, try to re-create the user with even weaker password so that John Ripper (or Hashcat) can crack within one hour. Take a screenshot of the output of John Ripper (or Hashcat).

I only decrypted the hash to the user “stian-munkejord”, since the password: “osboxes.org” for the root user probably isn’t in the wordlist: “rockyou.txt”

```

File Edit View Search Terminal Help
root@osboxes:~# cp /usr/share/wordlists/rockyou.txt.gz /home
root@osboxes:~# cd /home
root@osboxes:~# gunzip rockyou.txt.gz
root@osboxes:~# john --wordlist=rockyou.txt /home/userPasswords
Using default input encoding: UTF-8
Loaded 2 password hashes with 2 different salts (sha512crypt, crypt(3) $6$ [SHA512 256/256 AVX2 4x])
Remaining 1 password hash
Cost 1 (iteration count) is 5000 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
0g 0:00:00:30 0.82% (ETA: 16:48:44) 0g/s 4612p/s 4612c/s 4612C/s lovemep1z..katong
0g 0:00:00:38 1.04% (ETA: 16:48:12) 0g/s 4637p/s 4637c/s 4637C/s 211077..141002
Session aborted
root@osboxes:~# john --show /home/userPasswords
stian-munkejord:qwerty:1000:1000:,,:/home/stian-munkejord:/bin/bash

1 password hash cracked, 1 left
root@osboxes:~#

```

## Access Control (35 pts)

### Question 2.

- I. *Describe the differences between discretionary access control model and mandatory access control model*

The difference between discretionary access control (DAC) and the mandatory access control (MAC) model, is where the restrictions are set. In a DAC model the owner of the object specifies who is granted access, this is commonly found most operating systems such as Windows and Linux. On the other side the MAC model gives each subject a security clearance where only the people with high enough clearance can view specific files.

- II. *File permissions in Linux can be also represented in digits from 0-7 for the owner, group and others with reading as the most significant bit (E.g., the value 6 represents the permission right rw - for a file). Suppose a file in Linux has the permission as the digits 764.*

- *What does this permission right indicate for the owner/user, group and others?*

The number indicate that the: User can: read, write, execute; group can: read, write; others can: read.

- *What is the letter representation of this permission right?*

The letter representation for the number 764 is: "rwxrw-r--"

### Question 3.

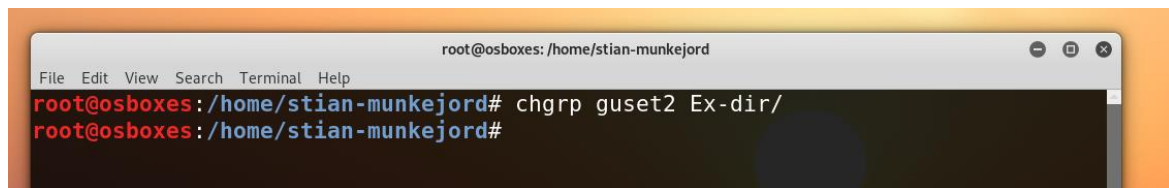
*In Kali Linux, create two users Guest2 and Guest3. Use the user (for yourself in Question 1) to create a directory Ex-dir containing a text file File1.txt.*

- I. *Display the owner and group for the directory Ex-dir; (Take a screenshot of the command and its output)*



```
stian-munkejord@osboxes: ~  
File Edit View Search Terminal Help  
stian-munkejord@osboxes:~$ mkdir Ex-dir  
stian-munkejord@osboxes:~$ ls -ld Ex-dir/  
drwxr-xr-x 2 stian-munkejord stian-munkejord 4096 Oct 14 16:41 Ex-dir/  
stian-munkejord@osboxes:~$
```

II. Change the directory's group as Guest2; (Take a screenshot of the command)



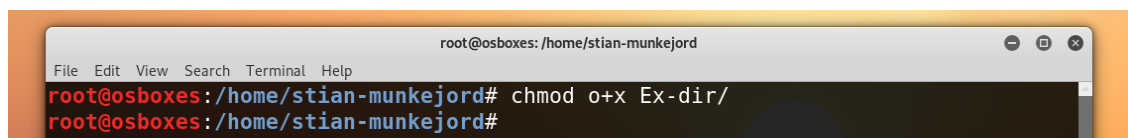
```
root@osboxes: /home/stian-munkejord
File Edit View Search Terminal Help
root@osboxes:/home/stian-munkejord# chgrp guset2 Ex-dir/
root@osboxes:/home/stian-munkejord#
```

III. Assign Guest2 the **read** and **write** permission to the directory. Create a file File2.txt in the directory;



```
root@osboxes: /home/stian-munkejord/Ex-dir
File Edit View Search Terminal Help
root@osboxes:/home/stian-munkejord# chmod g+w Ex-dir/
root@osboxes:/home/stian-munkejord# touch Ex-dir/File2.txt
root@osboxes:/home/stian-munkejord#
```

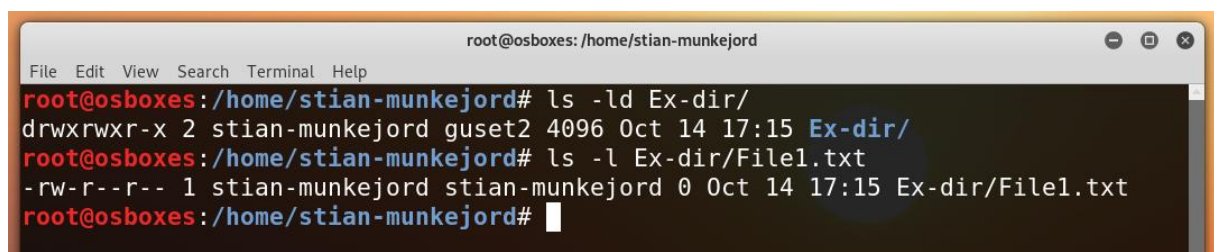
IV. Assign Guest3 (as other users) only execute permission to the directory



```
root@osboxes: /home/stian-munkejord
File Edit View Search Terminal Help
root@osboxes:/home/stian-munkejord# chmod o+x Ex-dir/
root@osboxes:/home/stian-munkejord#
```

V. Can Guest3 display the content of File1.txt in the directory? Justify your answer.

As default all “other” users can read the file when it’s created, and since I haven’t changed any permissions to File1.txt. The permission to the directory where the file is located is also readable for “others” and therefore the file is also readable for guest3.



```
root@osboxes: /home/stian-munkejord
File Edit View Search Terminal Help
root@osboxes:/home/stian-munkejord# ls -ld Ex-dir/
drwxrwxr-x 2 stian-munkejord guset2 4096 Oct 14 17:15 Ex-dir/
root@osboxes:/home/stian-munkejord# ls -l Ex-dir/File1.txt
-rw-r--r-- 1 stian-munkejord stian-munkejord 0 Oct 14 17:15 Ex-dir/File1.txt
root@osboxes:/home/stian-munkejord#
```

### Question 4.

A company has implemented a discretionary access control system for important documents and programs on a shared Linux server.

Figure 1 shows the current implementation of the DAC system. It only applies to the directory `/opt/company` (you can ignore all files/directories that are not within this directory). The list of users is shown in the output from the first `ls` command; the members of each group is shown in the output of the `tail` command; the permissions are shown in the output of the second `ls` command.

Assume the files are the objects, the subjects are the five users and there are four access rights: own, read, write and execute. Refer to Figure 4.2 and Table 2 in the textbook and complete the following sub questions.

```
root@node1:/opt/company# ls /home
pichaya surayut damienb steveng sumitra
root@node1:/opt/company# tail -3 /etc/group
staff:x:1010:steveng,pichaya,damienb,surayut,sumitra
eng:x:1011:steveng,pichaya
fin:x:1012:damienb,surayut
root@node1:/opt/company# ls -lR
.:
total 12
drwxr-xr-x 2 steveng eng 4096 Dec 30 11:27 engineering
drwxr-xr-x 2 damienb fin 4096 Dec 30 11:27 finance
drwxr-xr-x 4 steveng staff 4096 Dec 30 11:29 marketing
./engineering:
total 12
-rw-rw-r-- 1 steveng eng 8 Dec 30 11:27 designs.txt
-rwxrwx--- 1 pichaya eng 12 Dec 30 11:27 testscript
-rw-rw---- 1 pichaya eng 12 Dec 30 11:27 testresults.xls
./finance:
total 16
-rw-r--r-- 1 damienb fin 8 Dec 30 11:29 summary.pdf
-r--r----- 1 damienb fin 7 Dec 30 11:28 year12.xls
-r-----r-- 1 damienb fin 7 Dec 30 11:28 year13.xls
-rw-rw---- 1 surayut fin 7 Dec 30 11:28 year14.xls
./marketing:
total 4
drwxr-xr-x 2 steveng staff 4096 Dec 30 11:29 images
./marketing/images:
total 8
-rw-r--r-- 1 steveng staff 7 Dec 30 11:29 logo.png
-rw----- 1 steveng staff 5 Dec 30 11:29 logo.xcf
```

Figure 1: Implementation of a DAC system

I. Draw the Access Control Lists that illustrate the implementation on the Linux server;

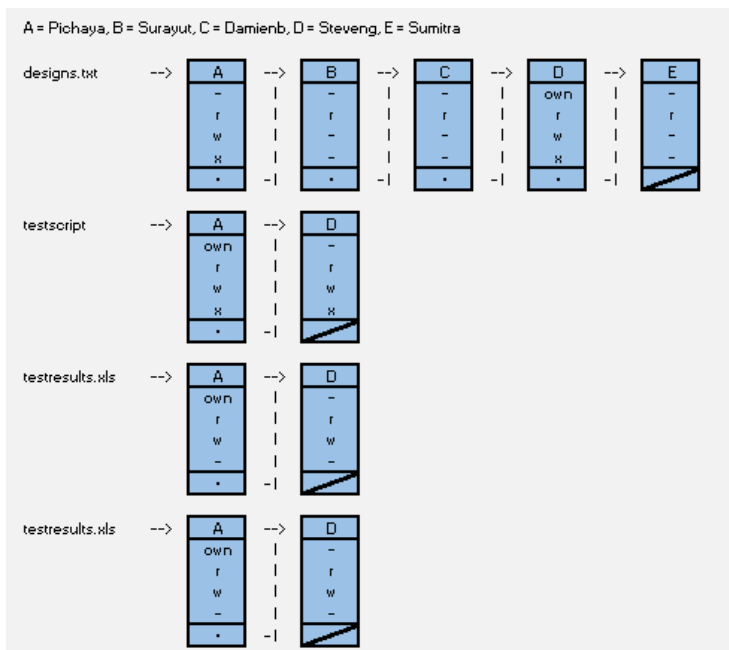


Figure 1.1

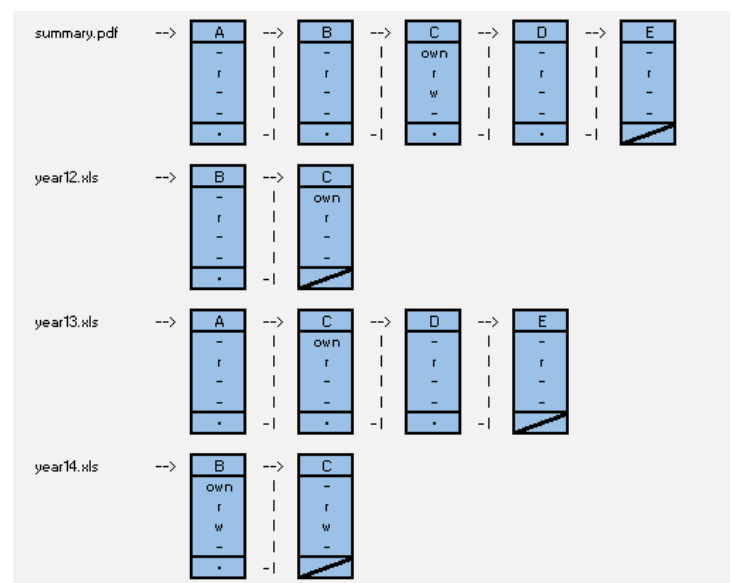


Figure 1.2

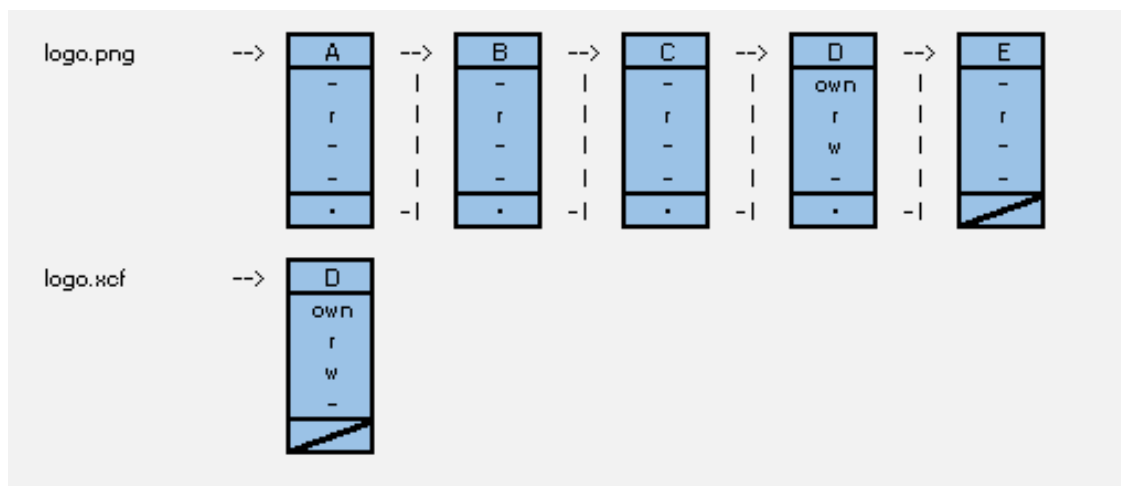


Figure 1.3

## II. Draw the Capability Lists that illustrate the implementation on the Linux server;



III. Build the Authorization Table that illustrate the implementation on the Linux server.

Subject	Access Mode	Objects
A	Read	designs.txt
A	Write	designs.txt
A	Execute	designs.txt
A	Own	testscripts
A	Read	testscripts
A	Write	testscripts
A	Execute	testscripts
A	Read	testresults.xls
A	Write	testresults.xls
A	Read	summary.pdf
A	Read	year13.xls
A	Read	logo.png
B	Read	designs.txt
B	Read	summary.pdf
B	Read	year12.xls
B	Own	year14.xls
B	Read	year14.xls
B	Write	year14.xls
B	Read	logo.png
C	Read	designs.txt
C	Own	summary.pdf
C	Read	summary.pdf
C	Write	summary.pdf
C	Own	year12.xls
C	Read	year12.xls
C	Own	year13.xls
C	Read	year13.xls
C	Read	year14.xls
C	Write	year14.xls
C	Read	logo.png

Figure 3.1

D	Own	designs.txt
D	Read	designs.txt
D	Write	designs.txt
D	Execute	designs.txt
D	Read	testresults.xls
D	Write	testresults.xls
D	Read	summary.pdf
D	Read	year12.xls
D	Own	logo.png
D	Read	logo.png
D	Write	logo.png
D	Own	logo.xcf
D	Read	logo.xcf
D	Write	logo.xcf
E	Read	designs.txt
E	Read	summary.pdf
E	Read	year13.xls
E	Read	logo.png

Figure 3.2

## Malware (20 pts)

### Question 5.

- I. Malicious software can be classified by propagation method or payload. Explain the difference between the three common propagation methods: worm, virus and social engineering;

The three common propagation methods are viruses, worms and social engineering. The difference of these different methods will I explain under:

Viruses hides inside another program and propagates itself to other programs, or even computers itself. Often the virus also includes some destructive functions to the computer.



Worms, on the other hand, can run independently and can propagate a complete working version of itself onto another host on the network. The worm usually consumes computer resources destructively.

Last, but not least we have social engineering which is all about tricking someone to give private information to compromise a computer system. An example on this is trojan horses. The trojan horse is a useful or seemingly useful program that contains hidden code of a malicious nature that executes when the program is invoked.

*II. Explain the difference between a normal virus, a metamorphic virus and a polymorphic virus, including discussing how easy they are to detect by anti-virus software*

A virus is a computer program, and its sole purpose to infect other computer programs. The two types of viruses we'll look at here are so called stealth viruses. These viruses are designed to slip under the installed anti-virus program of the host computer.

Firstly, we have **Polymorphic viruses**. These viruses inflict programs with an encrypted version of itself, and to decode it the virus also copy over a decryption module. Unlike normal encrypted viruses a polymorphic virus can modify the decryption module every time it infects a new file.

Secondly, we have **Metamorphic viruses**. Unlike polymorphic viruses the metamorphic viruses don't use encryption, therefore it doesn't need a decryption module. To achieve its ability to avoid antivirus detection most of the virus consists of a metaphoric engine. This engine completely rewrites itself every time it copies.

Finally, about the detection of the different types of viruses. Since polymorphic viruses changes its code, it makes the job for signature-based antivirus programs very difficult. However, the antivirus programs can decrypt the viruses using an emulator, or it can analyze its behavior to figure out the virus. The polymorphic virus therefore has some countermeasures to these solutions by the antivirus program. These involve reducing the speed of mutation and not to copy itself when there is already a copy on the computer. On the other hand, metamorphic viruses are just hard to detect in general because of its ever changing code of the virus.



## Question 6.

Computer viruses and worms have a common feature self-replication. A program, which takes no input and produces a copy of its own source code as its only output, is known as a quine. It implements a similar idea of computer viruses and worms. There are examples of quine program in different languages, e.g., one simple python quine example is as follows:

```
import sys;f=sys.stdout;x='import sys;f=sys.stdout;x=%s;f.write(x%%'x')';f.write(x%'x')
```

```
ChunleiLi:~ ChunleiLi$ cat quine2.py
import sys;f=sys.stdout;x='import sys;f=sys.stdout;x=%s;f.write(x%%'x')';f.write(x%'x')
ChunleiLi:~ ChunleiLi$ python2 quine2.py > copy.py
ChunleiLi:~ ChunleiLi$ cat copy.py
import sys;f=sys.stdout;x='import sys;f=sys.stdout;x=%s;f.write(x%%'x')';f.write(x%'x')
```

Write a python quine script by yourself (preferably with a part of your name embedded in the quine). Bonus points:

- Manage to include your first/last name in the quine script (+ 5 pts)
- Write a quine with more than 10 lines (+5 pts)

Python 3 quine:

```
#StianMunkejordLoveCybersecurity#StianMunkejordLoveCybersecurity#StianMunkejor
dLoveCybersecurity
myQuine = 'print("myQuine = " + repr(myQuine) + "\nName = " + repr("StianMunke
jord") + "\nChunleiFan = " + repr(ChunleiFan) + "\nif Name == " + repr("StianM
unkejord") + ":" + "\n    Name = " + repr("#") + " + Name + ChunleiFan" + "\nm
yList = []" + "\nfor i in range(3):" + "\n    myList.append(Name)" + "\nprint(
" + repr("") + ".join(myList))" + "\neval(myQuine))'
Name = 'StianMunkejord'
ChunleiFan = 'LoveCybersecurity'
if Name == 'StianMunkejord':
    Name = '#' + Name + ChunleiFan
myList = []
for i in range(3):
    myList.append(Name)
print(''.join(myList))
eval(myQuine)
```

To write this quine I followed the tutorial at<sup>1</sup>. To further develop the code, I used the same premise with the repr() and eval() function.

---

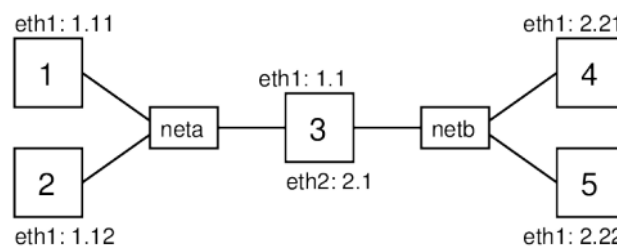
<sup>1</sup> <https://towardsdatascience.com/how-to-write-your-first-quine-program-947f2b7e4a6f>

The quine I wrote consists of a variable “myQuine” this variable contains the entire data structure of the program. To achieve this, I started to write a simple code, then copy it into the variable “myQuine”. The result is the code above, and when run it outputs a complete replica of itself.

## Firewalls (20 pts)

### Question 7.

Given the following topology of a network



where the router connecting the two subnets together, Nodes 1 and 2 are 'inside' of the subnet neta, Nodes 4 and 5 are 'outside' of neta and Node 3 with a firewall is the gateway of neta.

Try to write rules as general as possible. For example, although there are only two nodes outside, try to write rules such that the policy is achieved even if there were more than two nodes outside

Suppose the default policy is *ACCEPT* in the firewall on Node 3. Write packet filtering rules for the following goals (each of which is treated separately):

While writing these rules treat net a as the “inside” hosts and netb as the “outside” hosts

**Rule 1.** Block ping (icmp) packets being forwarded between the two subnets;

Default: Accept

Src IP	Src port	Dst IP	Dst port	Protocol	Action
*	*	192.168.*	*	ICMP	Drop

With this rule the two subnets cannot send the packets through the firewall which is the barrier between the two subnets.

**Rule 2.** Block ping packets coming into the firewall;

Default: Accept

Src IP	Src port	Dst IP	Dst port	Protocol	Action
*	*	192.168.1.1	*	ICMP	Drop

**Rule 3.** Prevent Node1 from SSHing to any outside nodes;

Default: Accept

Src IP	Src port	Dst IP	Dst port	Protocol	Action
192.168.1.11	*	192.168.2.*	22	TCP	Drop

Change the default policy as DROP and write packet filtering rules for the following goals:

**Rule 4.** Allow inside hosts can access outside websites;

Default: Drop

Src IP	Src port	Dst IP	Dst port	Protocol	Action
192.168.*	*	192.168.2.*	80	TCP	Accept
192.168.*	80	192.168.1.*	*	TCP	Accept

For simplicity we only write the rules with the port 80 which is HTTP. (HTTPS = port 443). In this instance the default policy is DROP, therefore we need to write two rules because when the client sends a TPC SYN the server responds with a TPC SYN ACK. If the SYN ACK is not responded to by the client, the connection would not be established. However, there is a potential security breach when following this system. A hacker can connect to the “netb” network and can bypass the firewall.

**Rule 5.** Allow outside hosts can SSH into Node1. No other access should be allowed.

Default: Drop

Src IP	Src port	Dst IP	Dst port	Protocol	Action
192.168.2.*	22	192.168.1.11	*	TCP	Accept

Since Node2 has a connection directly to Node1 without going through the firewall, this Node cannot be blocked to SSH into Node1 by the firewall.

## Transport-Layer Security (10 pts)

### Question 8.

- I. Use Wireshark to capture the traffic when you visit the website <https://mitt.uib.no/login/ldap>. Open the captured traffic Client Hello of the TLS protocol and take a screenshot of cipher suites initiated by the client (which is your browser)

231	3.298131	129.177.6.11	192.168.0.130	TLSv1.2	1099 Application Data
238	3.327223	192.168.0.130	129.177.6.218	TLSv1.2	603 Client Hello
253	3.347529	129.177.6.218	192.168.0.130	TLSv1.2	191 Server Hello, Change Cipher Spec, Encrypted Handshake Message
254	3.348300	192.168.0.130	129.177.6.218	TLSv1.2	105 Change Cipher Spec, Encrypted Handshake Message
255	3.348406	192.168.0.130	129.177.6.218	TLSv1.2	628 Application Data

Cipher Suites Length: 32	
✓	Cipher Suites (16 suites)
	Cipher Suite: Reserved (GREASE) (0x9a9a)
	Cipher Suite: TLS_AES_128_GCM_SHA256 (0x1301)
	Cipher Suite: TLS_AES_256_GCM_SHA384 (0x1302)
	Cipher Suite: TLS_CHACHA20_POLY1305_SHA256 (0x1303)
	Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 (0xc02b)
	Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)
	Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 (0xc02c)
	Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030)
	Cipher Suite: TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256 (0xc03a)
	Cipher Suite: TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (0xc03b)
	Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc013)
	Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014)
	Cipher Suite: TLS_RSA_WITH_AES_128_GCM_SHA256 (0x009c)
	Cipher Suite: TLS_RSA_WITH_AES_256_GCM_SHA384 (0x009d)
	Cipher Suite: TLS_RSA_WITH_AES_128_CBC_SHA (0x002f)
	Cipher Suite: TLS_RSA_WITH_AES_256_CBC_SHA (0x0035)
Compression Methods Length: 1	

- II. Check the traffic Server Hello of TLS protocol. Display the cipher suite offered by the server and explain how the cipher suite will protect subsequent HTTP traffics (Application Data) between your browser and the server.

238	3.327223	192.168.0.130	129.177.6.218	TLSv1.2	603 Client Hello
253	3.347529	129.177.6.218	192.168.0.130	TLSv1.2	191 Server Hello, Change Cipher Spec, Encrypted Handshake Message
254	3.348300	192.168.0.130	129.177.6.218	TLSv1.2	105 Change Cipher Spec, Encrypted Handshake Message
255	3.348406	192.168.0.130	129.177.6.218	TLSv1.2	628 Application Data

Session ID Length: 32	
Session ID: a5d0582a49430de6e1a57c6296d14290fef76933142a8f66...	
	Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030)
Compression Method: null (0)	

The cipher suite contains a set of algorithms which usually include a key exchange algorithm, a bulk encryption, and a message authentication code algorithm. A cipher suite can also include authentication algorithms which help the server and or the client to authenticate the other entity.

In our instance the protocol is Transport Layer Security (TLS), the key exchange is done with Elliptic Curve Diffie-Hellman Ephemeral (ECDHE), the authentication process uses the Rivest Shamir Adleman algorithm (RSA), encryption is accomplished by the Advanced Encryption Standard with 256bit key in Galois/Counter mode (AES 256 GCM) and finally it is all hashed with the Secure Hash Algorithm 385 (SHA385)

With this setup the two devices can exchange a key between themselves which is used to securely deliver the message. Further the bulk encryption algorithm is used to encrypt the data

being sent. Last but not least, the message authentication code algorithm provides integrity. This algorithm ensures the receiver that the information is coming from the right person.