

## Report of FRA Challenge Påsk 2020

1. Description
2. Traffic Analysis
3. Decrypting the plans
4. Answer the questions

## 1. Description

“Vad har en av nätverkets klienter för sig? Förstå trafiken och svara på frågorna. Tillhör tjänsten underrättelseanalytiker till Cyberförsvaret”

“What is one of the network clients up to? Understand the traffic and answer the questions. Belongs to intelligence analysis of the swedish National Cyber Defence”

The following analysis was conducted with the help of the kali linux operating system and python.

Provided files:

- readme.txt
- f1.py
- trafik.pcap

Readme.txt stored the following text.

“

2020-03-31

*Klienten 172.10.0.5 har betett sig konstigt. Svara på frågorna nedan så gott du kan.*

*trafik.pcap innehåller inspelad trafik.*

*f1.py ser ut att vara relaterad på något sätt.*

*Frågor:*

- 1. Vilka kommandon körs i första steget?*
- 2. Vilka kommandon körs i andra steget?*
- 3. Vilket lösenord har user2?*
- 4. När sker anfallet?*
- 5. Vilka vapen kommer de använda?*

*Lycka till!*

“

f1.py stored the following code.

“

```
#!/usr/bin/python3
import sys
import subprocess as s
import commands as cm

class A(object):
    def __init__(self, k):
        # self.r = sub.run(cm.c[str(k)])
        self.r = s.run(cm.c[str(k)], stdout=s.PIPE).stdout.decode()
        print(str(self))

    def __str__(self):
        # return A.e(self.r.stdout.decode())
        return A.e(self.r)

    @staticmethod
    def h(x):
        return "%0.2x" % ord(x)
        # return repr(chr(ord(x)))

    @staticmethod
    def e(x):
        o = ''.join([str(A.h(c)) for c in x])
        return o

    @staticmethod
    def d(x):
        o = ''
        i = 0
        while i < len(x):
            o += str(chr(int(x[i:i+2], 16)))
            i += 2
        return o
```

“

trafik.pcap was opened in wireshark and the following tcp stream was found. ("right click -> follow -> tcp stream")

[illegible]

The first two commands “3vil\$shell” executes are using f1.py, specifically the constructor of class A. The code in f1.py is used to execute local commands. There is also an import statement for something called “commands” . There is a python module called “commands”. However, it has been removed from python as of v.2.6. Hence, it is more likely there is another python file called “commands.py”.

There was a new program created, called “hex.py”, with the intent to convert hex code to ascii text with the following code.

```
“
#!/usr/bin/python3
import codecs as codecs;
print("paste hex")
print(codecs.decode(codecs.decode(input(),'hex'),'ascii'))
“
```

hex.py was used to decode the first 2 outputs.

First commands gave the output

```
“
total 84
drwxr-xr-x 10 user1 user1 4096 Mar 25 08:08 .
drwxr-xr-x 17 user1 user1 4096 Mar 25 08:06 ..
drwxrwx--- 2 user1 user1 4096 Feb  3 08:10 backup
-rw-rw-r-- 1 user1 user1 1096 Mar 25 07:37 bash_history
-rwxrwx--- 1 user1 user1 173 Feb  1 08:37 .bashrc
-rwxrwx--- 1 user1 user1 173 Feb  1 08:37 commands.py
drwxr-xr-x 2 user1 user1 4096 Feb  3 07:54 Documents
drwxr-xr-x 2 user1 user1 4096 Feb  3 07:54 Downloads
-rwxrwx--- 1 user1 user1 596 Feb  1 10:12 f1.py
-rwxrwx--- 1 user1 user1 6430 Feb  2 02:24 nc
-rwxrwx--- 1 user1 user1 9847 Mar 25 07:23 nc2
drwxr-xr-x 2 user1 user1 4096 Feb  3 07:54 Pictures
drwxrwx--- 3 user1 user1 4096 Feb  1 06:39 private
drwxr-xr-x 2 user1 user1 4096 Mar 25 08:08 __pycache__
-rw-rw-r-- 1 user1 user1 5503 Mar 25 08:05 README
drwxr-xr-x 2 user1 user1 4096 Feb  3 07:54 Videos
drwxrwx--- 6 user1 user1 4096 Mar 25 07:39 work
“
```

The first command was most likely “ls -la” which shows all files (including hidden files) within the current directory along with their privileges, owner, etc. The output shows a file called “commands.py” which points to the previous hypothesis of the import of “commands”. The output also shows that the file was created on Feb 1st 08:37 and the f1.py file was created at the same date at 10:12. Which points to them being related.

The second command had this output.

“

```
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp      0      0 localhost:8307          localhost:59580        ESTABLISHED
tcp      0      0 user:43215              worker:ssh              ESTABLISHED
tcp      0      0 localhost:55412         localhost:https        ESTABLISHED
tcp    509      0 user1:40145            webproxy.myweb:http-alt CLOSE_WAIT
tcp      0      0 localhost:https         localhost:54789        ESTABLISHED
tcp      0      0 localhost:44223         localhost:1111         ESTABLISHED

Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags   Type       State      I-Node  Path
unix    2      []     DGRAM      33489     /run/user/1000/systemd/notify
unix    2      []     DGRAM      23543     /run/user/120/systemd/notify
unix    3      []     DGRAM      15950     /run/systemd/notify
unix    9      []     DGRAM      15961     /run/systemd/journal/socket
unix    2      []     DGRAM      15983     /run/systemd/journal/syslog
unix   29      []     DGRAM      16002     /run/systemd/journal/dev-log
unix    3      []     STREAM     CONNECTED 50044
unix    2      []     STREAM     CONNECTED 46503
unix    3      []     STREAM     CONNECTED 40504
unix    3      []     STREAM     CONNECTED 36404
unix    3      []     STREAM     CONNECTED 41170     /run/systemd/journal/stdout
“
```

Most likely “netstat -l” which shows the machine’s connected server sockets. The intruder is most likely the second connection from the top

“tcp 0 0 user:43215 worker:ssh ESTABLISHED”.

One theory of why the intruder would execute this command is to see if he/she is alone in the system.

The next command used method “d” in class A of f1.py. The input was decoded to ascii.

“

```
import sys
import subprocess as s
import commands as cm

class B(object):
    @staticmethod
    def e(x):
        o = cm.c["func1"](x.encode())
        return o.decode()

    @staticmethod
    def d(x):
        o = cm.c["func2"](x.encode())
        return o.decode()

def run(cmd):
    dec = eval(B.d(cmd))
    r = s.run(dec, stdout=s.PIPE).stdout.decode()
    # print(r)
    print(B.e(r))
```

“

The third command was the following.

```
<3v1l$hell:~#> python3 -c "import f1 as f;
r=f.A.d('696d706f7274207379730a696d706f7274
20202020407374617469636d6574686f640a2020202
2657475726e206f2e6465636f646528290a0a202026
656e636f64652829290a20202020202020207265747
02072203d20732e72756e286465632c207374646f75
290a');print(r)" > f2.py
```

It uses method “d” in class “A” of f1.py. Method “d” decodes the hex code to ascii characters. At the last line of the picture above one can see that the command also writes the return statement to a new file called “f2.py”.

The following code is now in f2.py. It also uses “subprocess” and “commands”.

“

```
import sys
import subprocess as s
import commands as cm

class B(object):
    @staticmethod
    def e(x):
        o = cm.c['func1'](x.encode())
        return o.decode()

    @staticmethod
    def d(x):
        o = cm.c['func2'](x.encode())
        return o.decode()

def run(cmd):
    dec = eval(B.d(cmd))
    r = s.run(dec, stdout=s.PIPE).stdout.decode()
    # print(r)
    print(B.e(r))
```

“

“f2.py” seems to also run local commands however it decodes the inputs and encodes the outputs to base 64 instead of hex.

Next command and output was:

```
<3v1l$hell:~#> python3 -c "import f2 as f;e=f.B.e('d');print(e)"
ZA==
```

The command uses the new python code (“f2.py”), specifically class “B” method “e”. The argument was “d” and the output was printed. The output reads “ZA==” which is “d” encoded to base 64. Since there was a new python class added this was most likely a test to see if the code was runnable.



```
<3vil$hell:#> python3 -c "import f2 as f;f.run('wydscyysJy1sYSdd')"
```

```
kali@kali:~/Desktop/challengePask2020$ echo -n 'WydsycysJy1sYSdd' | base64 --decode
['ls', '-la']kali@kali:~/Desktop/challengePask2020$
```

” [‘ls’, ‘-la’] “

“

```
drwxr-xr-x 10 user1 user1 4096 Mar 25 08:08 .
drwxr-xr-x 17 user1 user1 4096 Mar 25 08:06 ..
drwxrwx--- 2 user1 user1 4096 Feb 3 08:10 backup
-rw-rw-r-- 1 user1 user1 1096 Mar 25 07:37 bash_history
-rwxrwx--- 1 user1 user1 173 Feb 1 08:37 .bashrc
-rwxrwx--- 1 user1 user1 173 Feb 1 08:37 commands.py
drwxr-xr-x 2 user1 user1 4096 Feb 3 07:54 Documents
drwxr-xr-x 2 user1 user1 4096 Feb 3 07:54 Downloads
-rwxrwx--- 1 user1 user1 596 Feb 1 10:12 f1.py
-rw-r--r-- 1 user1 user1 391 Mar 25 08:08 f2.py
-rwxrwx--- 1 user1 user1 6430 Feb 2 02:24 nc
-rwxrwx--- 1 user1 user1 9847 Mar 25 07:23 nc2
drwxr-xr-x 2 user1 user1 4096 Feb 3 07:54 Pictures
drwxrwx--- 3 user1 user1 4096 Feb 1 06:39 private
drwxr-xr-x 2 user1 user1 4096 Mar 25 08:08 __pycache__
-rw-rw-r-- 1 user1 user1 5503 Mar 25 08:05 README
drwxr-xr-x 2 user1 user1 4096 Feb 3 07:54 Videos
drwxrwx--- 6 user1 user1 4096 Mar 25 07:39 work
```

*['ls', '-la', 'backup/']*

```
drwxrwx--- 2 user1 user1 4096 Feb  3 08:10 .
drwxr-xr-x 10 user1 user1 4096 Mar 25 08:08 ..
-rwxrwx--- 1 user1 user1   0 Feb  1 03:08 .bashhist
-rwxrwx--- 1 user1 user1 126 Feb  3 08:10 pw_vault
-rwxrwx--- 1 user1 user1 352 Feb  2 01:38 shadow copy
```



['cat','backup/pw\_vault']

Accounts:

User	Pass
root	
user1	sommar2019
user2	HorseHatBatteryStaple
user3	

['cat','backup/shadow\_copy']

root:4c523473b193736d7e00ec6a9be1480a6b10ace5184784b3fabb331fcae40357 // notes: root-priviledges pw:  
--  
user1:d146c62ff4a1b1552bfe162b86d7a656a26d2ad1cf812c4fb3eea0c272bc313b // notes: non\_root-priv pw:  
sommar2020  
user2:6008534dbeb34eb413a308f06067bb7f7060582e0b5b3d85c3e35bd07948c437 // notes: root\_priviledges  
pw: HorseHatBatteryStaple  
user3:

['ls','private']

bank\_details  
photos

['ls','work']

mails  
meeting protocols  
prospects  
scripts  
secret\_plans.zip

With these five commands the intruder learned potentially sensitive information about the system.

For example;

- hashed passwords for two users
- the two actual passwords
- potentially the privileges of these two users.

Moreover the location of;

- banking information
- private pictures
- Work mails
- meeting protocols
- Prospects
- Scripts
- "Secret plans"
- Bash history of user 1

"f2.py" was created by the intruder on March 25th 08:08 which must be the time of this traffic. Considering that the text file storing the password was created on February 3:d there is a high probability that the passwords were up to date with the current passwords of the user accounts.

The next command is different when compared to the previous five. This command creates a new python file, similar to the command that created "f2.py".

```
<3v1l$hell:~#> python3 -c "import f2 as f;
r=f.B.d('aw1wb3J0IHN1YnByb2Nlc3MgYXMGc3V1Cgpl
ByYW5nZSgyNTYpOgogICAgICAgIHggPSAoeCArIGJveF
iAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
Nl0pKQogICAgcmV0dXJuICcnLmpvaW4ob3V0KQoKCmRL
gCiAgICByZXR1cm4gJycuam9pbihbc3RyKGgoYykpIGZ
BzdHl0Y2hyKGluZCh4W2k6aSsyXSwgMTYpKSskKICAgIC
CgpLnN0cm1wKCKKICAgIGV4Y2VwdCBFeGNlcHRpb24gY
ICAgawYgZiBpcyBub3QgTm9uZTogICMgZiBpcyBzZXQs
gICAgIG0gPSBzdWIucnVuKGntZCwgc3Rkb3V0PXM1Yi5
91dDogXG4nICsgbS5zdGRvdXQuZGVjb2RlKCkgKyAnXG
gppZiBfX25hbWVfXyA9PSAnX19tYWluX18nOgogICAgal
LmFyZ3ZbMl0pCgoK');print(r)" > f3.py
```

The decoded input which would be the content of "f3.py":

"

```
import subprocess as sub

def ed(data, key):
    # encryptor / decryptor
    # return value is binary
    x = 0
    box = list(range(256))
    for i in range(256):
        x = (x + box[i] + ord(key[i % len(key)])) % 256
        box[i], box[x] = box[x], box[i]
    x, y = 0, 0
    out = []
    for char in data:
        x = (x + 1) % 256
        y = (y + box[x]) % 256
        box[x], box[y] = box[y], box[x]
        out.append(chr(ord(char) ^ box[(box[x] + box[y]) % 256]))
    return ''.join(out)

def h(x):
    # hex one character
    return "%0.2x" % ord(x)

def c(s):
    # encode a string s, calls h(x) on each character
    return ''.join([str(h(c)) for c in s])

def g(x):
    # un-hex a hex-string into a ascii-string
    o = ''
    i = 0
    while i < len(x):
        o += str(chr(int(x[i:i+2], 16)))
        i += 2
    return o

def execute(cmd, f=None):
    try:
        with open('pass.txt', 'r') as _f:
            pw = _f.read().strip()
    except Exception as e:
        # print(e)
        pw = 'default_pass'
    # print(pw)
    cmd = g(cmd)
    cmd = ed(cmd, pw)
    # print(cmd)
    if f is not None: # f is set,
        with open(f, 'w') as _f:
            _f.write(cmd)
            _f.write('\n')
    else:
        cmd = eval(cmd)
        m = sub.run(cmd, stdout=sub.PIPE, stderr=sub.PIPE)
        # print('out', m.stdout.decode())
        # print('err', m.stderr.decode())
        mout = 'stdout: \n' + m.stdout.decode() + '\n'
        mout += 'stderr: \n' + m.stderr.decode()
        # print(mout)
        mout = c(ed(mout, pw))
        print(mout)
```

```

if __name__ == '__main__':
    import sys
    if len(sys.argv) == 2:
        execute(sys.argv[1])
    if len(sys.argv) == 3:
        execute(sys.argv[1], sys.argv[2])

```

Comparable to the other python files it uses the module “subprocess” to execute local commands. However it does not use the other class called “commands”. This means that this class can be run by itself as the others could not. The main method sends the inputs to the execute method. If the execute method finds a file called “pass.txt” the input is executed locally, with the help of “subprocess”. If the method does not find a text file called “pass.txt”, a new file is written with the file name of the second input and the content of the first input.

The execute method sends the inputs via the g method which essentially with the help of method “c” and “h” decode the input from hex code to ascii code. The method then sends the input to the “ed” method before the execution of the command. If there is only one input there is a default key sent(“default\_pass”) The method also sends the output of the local command execution via the “ed” method before printing it. The “ed” method takes two arguments, the content and a key then passes the arguments through a symmetric encryption. With some analysis and searching it was found to be RC4 encryption [1].

The next command uses the new “f3.py” code and sends two arguments. As the intruder executed the command “ls -la” in the directory previously and there was no “pass.txt” file one could assume that the program will write the first argument to a new file with the next argument as file name.

```

<3vil$hell:~#> python3 f3.py 1b17574cd639192cd70e file.txt

```

f3.py was created and modified to decrypt the last inputs and outputs. The input was sent to the new python program and since the hypothesis was that there was no “pass.txt” present the key “default\_pass” was used as the key.

```

#!/usr/bin/python3
def ed(data, key):
    # encryptor / decryptor
    # return value is binary
    x = 0
    box = list(range(256))
    for i in range(256):
        x = (x + box[i] + ord(key[i % len(key)])) % 256
        box[i], box[x] = box[x], box[i]
    x, y = 0, 0
    out = []
    for char in data:
        x = (x + 1) % 256
        y = (y + box[x]) % 256
        box[x], box[y] = box[y], box[x]
        out.append(chr(ord(char) ^ box[(box[x] + box[y]) % 256]))
    return ''.join(out)

```

```

def h(x):
    # hex one character
    return "%0.2x" % ord(x)

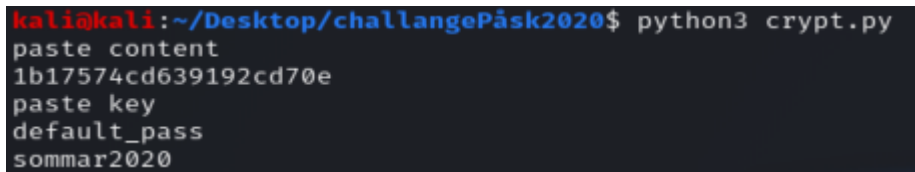
def c(s):
    # encode a string s, calls h(x) on each character
    return ''.join([str(h(c)) for c in s])

def g(x):
    # un-hex a hex-string into a ascii-string
    o = ''
    i = 0
    while i < len(x):
        o += str(chr(int(x[i:i+2], 16)))
        i += 2
    return o

print("paste content")
con = input()
print("paste key")
key = input()
print(ed(g(con),key))

```

The output of the new program was “sommar2020”. Which means that most likely there was a new file created called “file.txt” with the content “sommar2020”.



```

kali@kali:~/Desktop/challengePask2020$ python3 crypt.py
paste content
1b17574cd639192cd70e
paste key
default_pass
sommar2020

```

Next command and output:

```
<3vil$hell: #> python3 f3.py 335f575790670c7a8c5218e71c0e7f98dc6a1b3a6f9bc73424cc2f521b0c5e4ec23f113cef340ebd0c1379cdca6d61
```

These were decrypted in the same manner as the last command, with the key “default\_pass”.

```
['mv','file.txt','pass.txt']
```

*stdout:*

*stderr:*

Essentially this means that the file.txt was renamed to pass.txt. Hence, “sommar2020” will most likely be used as the key for the following commands. The “stdout” and “stderr” is hardcoded in “f3.py” to separate the output and the error codes.

The next command was decoded in the same manner as the previous command, however it uses the key “sommar2020” instead of “default\_pass”.

This command showed that indeed there was a file called pass.txt in the directory.

```
['ls','-la']
```

*stdout:*

*total 96*

```
drwxr-xr-x 10 user1 user1 4096 Mar 25 08:09 .
drwxr-xr-x 17 user1 user1 4096 Mar 25 08:06 ..
drwxrwx--- 2 user1 user1 4096 Feb  3 08:10 backup
-rw-rw-r-- 1 user1 user1 1096 Mar 25 07:37 bash_history
-rwxrwx--- 1 user1 user1 173 Feb  1 08:37 .bashrc
-rwxrwx--- 1 user1 user1 173 Feb  1 08:37 commands.py
drwxr-xr-x 2 user1 user1 4096 Feb  3 07:54 Documents
drwxr-xr-x 2 user1 user1 4096 Feb  3 07:54 Downloads
-rwxrwx--- 1 user1 user1 596 Feb  1 10:12 f1.py
-rw-r--r-- 1 user1 user1 391 Mar 25 08:08 f2.py
-rw-r--r-- 1 user1 user1 1747 Mar 25 08:09 f3.py
-rwxrwx--- 1 user1 user1 6430 Feb  2 02:24 nc
-rwxrwx--- 1 user1 user1 9847 Mar 25 07:23 nc2
-rw-r--r-- 1 user1 user1 11 Mar 25 08:09 pass.txt
drwxr-xr-x 2 user1 user1 4096 Feb  3 07:54 Pictures
drwxrwx--- 3 user1 user1 4096 Feb  1 06:39 private
drwxr-xr-x 2 user1 user1 4096 Mar 25 08:08 __pycache__
-rw-rw-r-- 1 user1 user1 5503 Mar 25 08:05 README
drwxr-xr-x 2 user1 user1 4096 Feb  3 07:54 Videos
drwxrwx--- 6 user1 user1 4096 Mar 25 07:39 work
```

*stderr:*

With this command the intruder located the asset that was of interest ("secret\_plans.zip")  
['ls','work/']

*stdout:*  
mails  
meeting protocols  
prospects  
scripts  
secret\_plans.zip

*stderr:*

This command tries to decompress the zip file. However the intruder learns that the file is password protected.

(The output for this command was not encrypted.)

['unzip','work/secret\_plans.zip']

*File is password-protected.*

The intruder views the history of user 1's commands and finds the password for the zip is  
"supersecretpassword"

['cat','bash\_history']

*stdout:*  
sudo apt search wireshark  
sudo apt install wireshark  
python  
sudo apt install python  
sudo apt install python3.6  
sudo apt install ipython3  
which python  
python  
python3  
history  
man history  
ld  
ls  
ping 8.8.8.8  
sudo wireshark  
ipython  
ipython3  
python3  
ipython3  
netstat  
ifconfig  
ls -la backup/  
cat backup/pw\_vault  
ls -la backup/  
cat backup/pw\_vault  
ls -la backup/  
python  
python3  
ipython3  
ls backup/  
ls  
ls



```

ls -la
cd work
ls
cd ..
ls
ls -la
cd work
ls
cd scripts
cd ..
cd plans
ls
cat attack_plans_VERY_SECRET.txt
cat drawing_of_our_new_wepon_VERY_SECRET.txt
nano drawing_of_our_new_wepon_VERY_SECRET.txt
cat drawing_of_our_new_wepon_VERY_SECRET.txt
nano drawing_of_our_new_wepon_VERY_SECRET.txt
nano attack_plans_VERY_SECRET.txt
nano drawing_of_our_new_wepon_VERY_SECRET.txt
cat drawing_of_our_new_wepon_VERY_SECRET.txt
cd ..
man zip
zip -r -P supersecretpassword secret_plans.zip plans
rm -rf plans
cd ..
private
cd private/
ls
cd ..
ls
ipython3
history
ls
ls
cd ..
cd ..
cd ..
ls
man unzip
history
cat .bash_history
ipython
ipython3
ls
mv plans plans2
ipython3
ipython
ipython3
ipython3
netstat

stderr:

```

This command gives the intruder a hex dump of the zip file. Which could be recreated to a copy of the zip file.

```
['hd','-v','work/secret_plans.zip']
```

```

stdout:
00000000  50 4b 03 04 0a 00 00 00 00 00 84 7b 79 50 00 00 |PK.....{yP..|
00000010  00 00 00 00 00 00 00 00 00 00 06 00 1c 00 70 6c |.....p|
00000020  61 6e 73 2f 55 54 09 00 03 78 6a 7b 5e 8a 6a 7b |ans/UT...xj{^.j|
00000030  5e 75 78 0b 00 01 04 e8 03 00 00 04 e8 03 00 00 |^ux.....|

```

00000040 50 4b 03 04 14 00 09 00 08 00 80 7b 79 50 a0 2d |PK.....{yP-|  
00000050 9f 8d 22 01 00 00 87 04 00 00 22 00 1c 00 70 6c |.. ".....".....p|  
00000060 61 6e 73 2f 61 74 74 61 63 6b 5f 70 6c 61 6e 73 |ans/attack\_plans|  
00000070 5f 56 45 52 59 5f 53 45 43 52 45 54 2e 74 78 74 |\_VERY\_SECRET.txt|  
00000080 55 54 09 00 03 6f 6a 7b 5e f3 6a 7b 5e 75 78 0b |UT...oj{^ux.|  
00000090 00 01 04 e8 03 00 00 04 e8 03 00 00 bb 56 7f 4a |.....V.J|  
000000a0 94 9c 6f e5 a3 14 44 8b 19 5a 8d e0 ed b3 14 29 |..o...D.Z....)|  
000000b0 19 0a 45 94 01 0a b5 da 98 48 99 15 79 d1 25 36 |..E.....H..y.%6|  
000000c0 e8 e6 70 fb 9c f1 fd 78 ba db 5a 2e ad 27 b4 db |..p....x.Z...|  
000000d0 9f 89 09 62 e0 00 32 92 90 a7 69 32 48 ed ae 3d |..b..2...i2H..=|  
000000e0 d0 5f c4 42 46 c0 bb 9d 70 95 13 a2 73 df 98 5c |..BF...p...s..|  
000000f0 7e ac b4 b7 45 a9 e2 9d 6c 7d 05 49 0b 3f d1 91 |~...E...}..I.?.|  
00000100 b7 30 e7 5e 06 32 dc c2 90 05 cd ad b3 55 34 1f |.0.^2.....U4.|  
00000110 8b 10 87 38 fc 99 16 07 50 53 d5 d1 75 b4 fd f8 |...8....PS..u...|  
00000120 54 5f 9b e6 cd 98 33 50 fc 68 bb f9 2a d2 16 76 |T....3P.h...\*.v|  
00000130 4b fe d1 37 d7 ac e1 0b 64 10 01 45 a7 96 d8 2e |K..7....d.E....|  
00000140 8c 79 a8 0e 58 c4 ed ac 7f 55 7f e2 1b ef 1d 68 |.y.X....U.....h|  
00000150 95 6f 67 9b ba 55 56 1e 8b 5f 1a 2c 18 c2 a3 a7 |.og..UV.....|  
00000160 10 43 a8 aa 65 86 aa d9 ed c9 5d fd ec d3 d7 62 |.C.e.....].b|  
00000170 ae b6 f7 0d 98 c4 fc 22 af 11 a4 3e e2 92 f2 eb |.....">....|  
00000180 ed 59 88 3f 86 62 65 12 b9 69 5c ad 35 2e f6 b3 |.Y?.be..l.5...|  
00000190 b3 65 ed 4f 81 22 9b 39 88 39 46 8a 2b ad 71 93 |.e.O..9.9F+.q.|  
000001a0 92 46 9e 20 96 15 fc dd c3 65 89 2e 37 a6 c6 d4 |.F.....e..7...|  
000001b0 e0 9a d1 8a 91 ca b2 87 a4 5c 91 4a 3a 5c 50 4b |.....\J:|PK|  
000001c0 07 08 a0 2d 9f 8d 22 01 00 00 87 04 00 00 50 4b |...-.. ".....PK|  
000001d0 03 04 14 00 09 00 08 00 84 7b 79 50 e0 dd b7 d4 |.....{yP...|  
000001e0 eb 00 00 00 ce 02 00 00 2f 00 1c 00 70 6c 61 6e |...../...plan|  
000001f0 73 2f 64 72 61 77 69 6e 67 5f 6f 66 5f 6f 75 72 |s/drawing\_of\_our|  
00000200 5f 6e 65 77 5f 77 65 61 70 6f 6e 5f 56 45 52 59 |\_new\_weapon\_VERY|  
00000210 5f 53 45 43 52 45 54 2e 74 78 74 55 54 09 00 03 |\_SECRET.txtUT...|  
00000220 78 6a 7b 5e f3 6a 7b 5e 75 78 0b 00 01 04 e8 03 |xj{^ux.....|  
00000230 00 00 04 e8 03 00 00 8c 34 0f 86 17 1a 87 29 f5 |.....4.....)|  
00000240 ae 0e 36 7f 9e d1 20 46 ae 75 00 fc 4a 53 de 84 |..6... F.u...JS..|  
00000250 42 45 00 d7 5e 17 1a ae 29 72 c6 90 19 bb db |BE..Yu...)r....|  
00000260 8a 9f f9 c8 f7 b5 37 a6 7d bd 3d fb ec 7d 61 b6 |.....7.}.=.}a.|  
00000270 5c 48 2e 25 05 56 bd ac 1e 4c bb 36 2c 60 68 96 |\\H.%V...L.6.`h.|  
00000280 9f 3a 21 2c e5 2f 49 0b 92 9d c1 b8 df 66 3d 89 |.:./.....f=|  
00000290 5a ae 51 fb 3b db 31 85 15 17 2c ea a4 bd 43 e7 |Z.Q...1.....C.|  
000002a0 b4 34 02 6a cc 6b d5 bb 14 55 a2 d0 01 c8 07 97 |.4.j.k...U.....|  
000002b0 13 d4 2f 2e d0 20 43 04 99 45 11 eb 9a 15 7c 1f |../. C..E....|  
000002c0 b7 68 c0 7d ae 26 8c 63 5e a0 92 9d 2a 8a 7a 75 |.h.}&.c^...\*.zu|  
000002d0 5b 31 d2 a1 a7 08 b8 a5 fe 6b 6f 66 b7 75 1d bd |[1.A.....kof.u..|  
000002e0 1f 49 37 3d 51 0e 7e e5 9e 40 96 e9 40 62 20 f2 |.I7=Q.~..@..@b..|  
000002f0 6a 47 9e cc 9a e3 72 10 83 10 78 b1 cd ad 7c 2c |jG....r...x...|  
00000300 8d 06 42 dc ed 7a 08 af ce 2b a6 83 47 07 2d 57 |..B.z...+.G.-W|  
00000310 dc 8a 69 81 d8 cb 8f 18 30 75 3a 67 af 86 b1 45 |..i.....Ou:g...E|  
00000320 fe 17 50 4b 07 08 e0 dd b7 d4 eb 00 00 00 ce 02 |..PK.....|  
00000330 00 00 50 4b 01 02 1e 03 0a 00 00 00 00 00 84 7b |..PK.....{f|  
00000340 79 50 00 00 00 00 00 00 00 00 00 00 00 06 00 |yP.....|  
00000350 18 00 00 00 00 00 00 00 10 00 ed 41 00 00 00 00 |.....A....|  
00000360 70 6c 61 6e 73 2f 55 54 05 00 03 78 6a 7b 5e 75 |plans/UT...xj{^u|  
00000370 78 0b 00 01 04 e8 03 00 00 04 e8 03 00 00 50 4b |x.....PK|  
00000380 01 02 1e 03 14 00 09 00 08 00 80 7b 79 50 a0 2d |.....{yP-|  
00000390 9f 8d 22 01 00 00 87 04 00 00 22 00 18 00 00 00 |.. ".....".....|  
000003a0 00 00 00 00 00 00 b4 81 40 00 00 00 70 6c 61 6e |.....@...plan|  
000003b0 73 2f 61 74 74 61 63 6b 5f 70 6c 61 6e 73 5f 56 |s/attack\_plans\_V|  
000003c0 45 52 59 5f 53 45 43 52 45 54 2e 74 78 74 55 54 |ERY\_SECRET.txtUT|  
000003d0 05 00 03 6f 6a 7b 5e 75 78 0b 00 01 04 e8 03 00 |...oj{^ux.....|  
000003e0 00 04 e8 03 00 00 50 4b 01 02 1e 03 14 00 09 00 |.....PK.....|  
000003f0 08 00 84 7b 79 50 e0 dd b7 d4 eb 00 00 00 ce 02 |...{yP.....|  
00000400 00 00 2f 00 18 00 00 00 00 00 01 00 00 00 b4 81 |../.....|  
00000410 ce 01 00 00 70 6c 61 6e 73 2f 64 72 61 77 69 6e |....plans/drawin|  
00000420 67 5f 6f 66 5f 6f 75 72 5f 6e 65 77 5f 77 65 61 |g\_of\_our\_new\_wea|  
00000430 70 6f 6e 5f 56 45 52 59 5f 53 45 43 52 45 54 2e |pon\_VERY\_SECRET.|  
00000440 74 78 74 55 54 05 00 03 78 6a 7b 5e 75 78 0b 00 |txtUT...xj{^ux..|  
00000450 01 04 e8 03 00 00 04 e8 03 00 00 50 4b 05 06 00 |.....PK...|  
00000460 00 00 00 03 00 03 00 29 01 00 00 32 03 00 00 00 |.....).....2....|

```
00000470 00
00000471
```

```
|.|
```

stderr:

### 3. Decrypting the plans.

To convert the hexdump to a zip file, the hypothesis was to:

- Write the hexdump to a text file.
- Cut the addresses and ascii code
- Convert the hex code to bit code
- Convert the bit file to a zip file

The hexdump was written to a text file. The cut command in linux was used to remove the addresses and ascii code.

```
00000430 70 6f 6e 5f 56 45 52 59 5f 53 45 43 52 45 54 2e pon_VERY_SECRET.
00000440 74 78 74 55 54 05 00 03 78 6a 7b 5e 75 78 0b 00 txtUT ... xj{^ux ..
00000450 01 04 e8 03 00 00 04 e8 03 00 00 50 4b 05 06 00 .....PK ...
00000460 00 00 00 03 00 03 00 29 01 00 00 32 03 00 00 00 .....) ... 2....
00000470 00 .|
00000471

kali@kali:~/Desktop/challengePask2020$ cut -d " " -f 2-19 hexdump.txt > hexdump2.txt
kali@kali:~/Desktop/challengePask2020$ cat hexdump2.txt
50 4b 03 04 0a 00 00 00 00 00 84 7b 79 50 00 00
00 00 00 00 00 00 00 00 00 00 06 00 1c 00 70 6c
61 6e 73 2f 55 54 09 00 03 78 6a 7b 5e 8a 6a 7b
5e 75 78 0b 00 01 04 e8 03 00 00 04 e8 03 00 00
```

For the conversion and decompressing the following commands were used. The password found in the bash history ("*supersecretpassword*") was used to decrypt the zip.

```
kali@kali:~/Desktop/challengePask2020$ xxd -r -p hexdump2.txt binary.bin
kali@kali:~/Desktop/challengePask2020$ mv binary.bin plans.zip
kali@kali:~/Desktop/challengePask2020$ unzip plans.zip
Archive: plans.zip
  creating: plans/
[plans.zip] plans/attack_plans_VERY_SECRET.txt password:
  inflating: plans/attack_plans_VERY_SECRET.txt
  inflating: plans/drawing_of_our_new_weapon_VERY_SECRET.txt
kali@kali:~/Desktop/challengePask2020$
```

xxd was used to do the conversion from hex to binary, then the contents of the binary file was moved to a zip file.

The zip contained the following two files:

```
kali@kali:~/Desktop/challengePask2020$ cd plans/ && ls
attack_plans_VERY_SECRET.txt  drawing_of_our_new_weapon_VERY_SECRET.txt
kali@kali:~/Desktop/challengePask2020/plans$
```

Contents of "attack\_plans\_VERY\_SECRET.txt":

"

*VERY VERY SUPERDUPER SECRET DOCUMENT*

=====  
WARNING  
DO NOT DISTRIBUTE TO ANYONE!!!  
ABSOLUTELY NOT TO THE ENEMY!!!  
=====

*When?*

*Timing is essential. Therefore we start the attack AT DAWN to take the enemy by surprise!*

*Attack plan:*


OUR FORCES		ENEMY FORCES
	\	
	=====\	=====
	=====/	
	/	
===		
===	==	
(>'')> (>'')> (>'')>	<("<) <("<) <("<) <("<) <("<	
(>'')> (>'')> (>'')>	<("<) <("<) <("<) <("<) <("<	
(>'')> (>'')> (>'')>	<("<) <("<) <("<) <("<) <("<	
(>'')> (>'')> (>'')>	<("<) <("<) <("<) <("<) <("<	

"

“

=====

=====



“

#### 4. Answer the questions

“

*Frågor:*

- 1. Vilka kommandon körs i första steget?*
- 2. Vilka kommandon körs i andra steget?*
- 3. Vilket lösenord har user2?*
- 4. När sker anfallet?*
- 5. Vilka vapen kommer de använda?*

“

Translation

“Questions:

1. Which commands were run in the first act?
2. Which commands were run in the second act?
3. What is user2's password?
4. When is the attack?
5. What weapons will they use?

“

1. The first commands were setting up encryption for the traffic. First using hex code then base 64 and then hex code of rc4 encryption.

2. The second act the intruder looked around the directories and found the asset of interest along with the password to unlock it.

3. User 2's password was found in the "pw\_vault" and is "HorseHatBatteryStaple"

4. According to the "attack\_plans\_VERY\_SECRET.txt" the attack will be launched at dawn.

5. Looking at the "drawing\_of\_our\_new\_weapon\_VERY\_SECRET.txt" it appears to be swords and intercontinental ballistic missiles.