Walk though

**Exercise 1 (20 marks)**

The virtual machine 2023-ecm2426-ca has several regular user accounts configured. In this exercise, you should explore these accounts and their access rights.

**E.1.1.** (5 marks):

To execute the question started with ---> awk -F: '$3 >= 2000 { print $1,$3 }' /etc/passwd

This returned the list:

nobody 65534

axxy075 2000

wxxn190 2001

bxxn580 2002

exxn563 2003

exxy014 2004

dxxl909 2005

rxxe878 2006

axxx905 2007

I was unsure about this so I then used the command --->

“””

lh@ML-RefVm-464034:~$ awk -F: '$3 >= 2000 { print $1,$3 }' /etc/passwd | while IFS= read -r user; do

username=$(echo "$user" | cut -d' ' -f1)

id=$(echo "$user" | cut -d' ' -f2)

echo "Checking user: $username with ID: $id"

if groups $username | grep -q staff; then

echo "User $username is a member of staff"

echo "$user"

else

echo "User $username is not a member of staff"

fi

done

“””

This retuned --->

Checking user: nobody with ID: 65534

User nobody is not a member of staff

Checking user: axxy075 with ID: 2000

User axxy075 is a member of staff

axxy075 2000

Checking user: wxxn190 with ID: 2001

User wxxn190 is not a member of staff

Checking user: bxxn580 with ID: 2002

User bxxn580 is not a member of staff

Checking user: exxn563 with ID: 2003

User exxn563 is not a member of staff

Checking user: exxy014 with ID: 2004

User exxy014 is a member of staff

exxy014 2004

Checking user: dxxl909 with ID: 2005

User dxxl909 is not a member of staff

Checking user: rxxe878 with ID: 2006

User rxxe878 is a member of staff

rxxe878 2006

Checking user: axxx905 with ID: 2007

User axxx905 is a member of staff

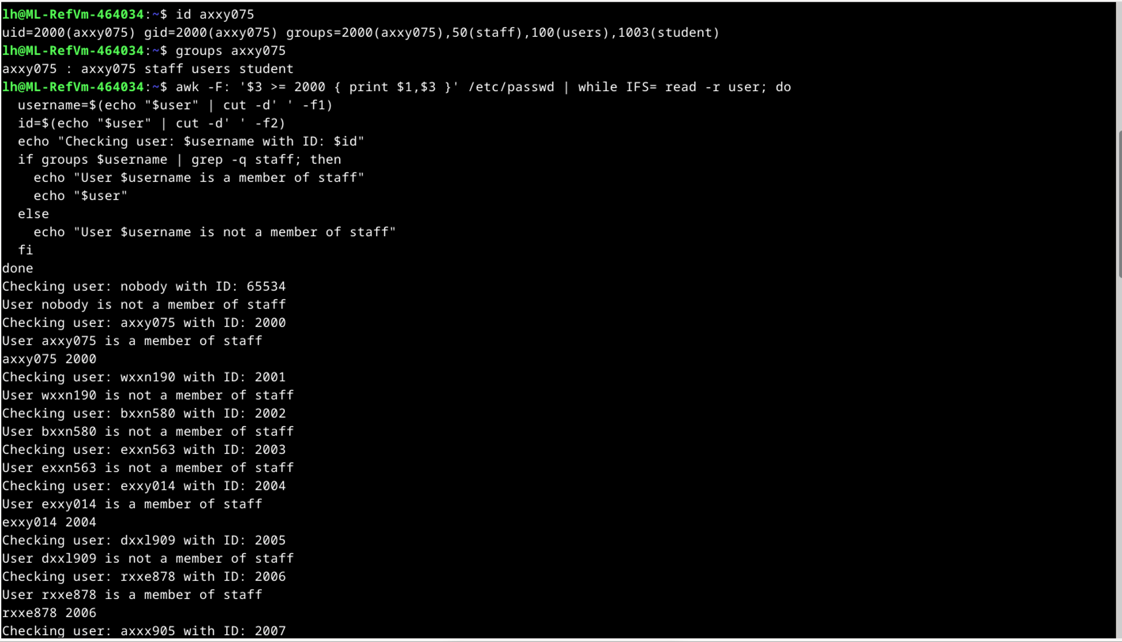
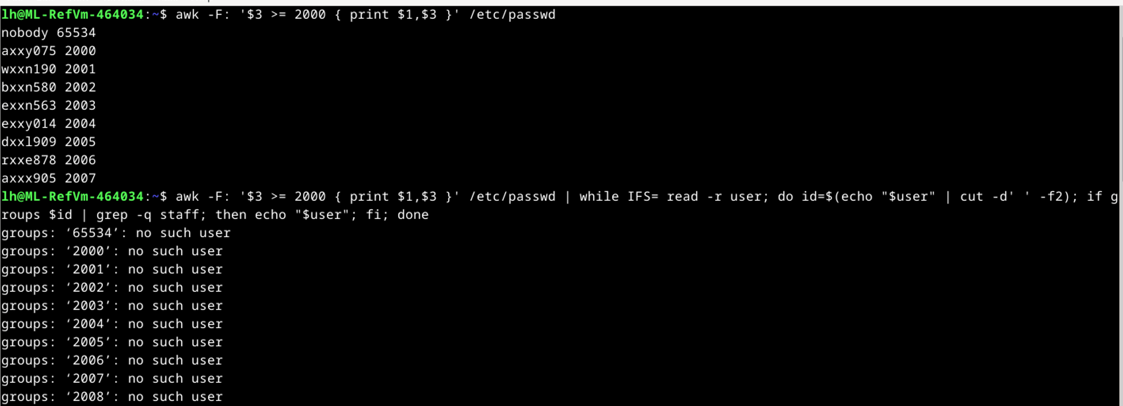
axxx905 2007

Checking user: rxxs447 with ID: 2008

User rxxs447 is a member of staff

rxxs447 2008

from there I was able to find all users that had id 2000 or higher and a member of staff



(other potential useful code: awk -F: '$3 >= 2000 { print $1,$3 }' /etc/passwd | while IFS= read -r user; do id=$(echo "$user" | cut -d' ' -f2); if groups $id | grep -q staff; then echo "$user"; fi; done)

**E.1.2.** (5 marks):

To execute this question, I first used the command --->. ls -la /home/axxy075

This returned a list of files in her directory

Noticed file password.txt in directory

To read file ---> cat /home/axxy075/password.txt

Retuned 4 codes:

IZQI4FCLQ5

eFGoDSamR0

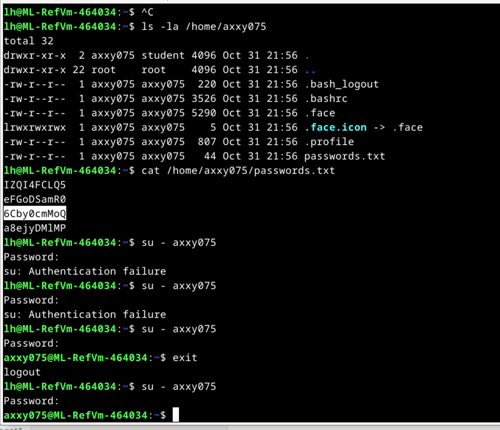
6Cby0cmMoQ

a8ejyDMlMP

I then attempted to log in using process of elimination

Code used was ---> su – axxy075

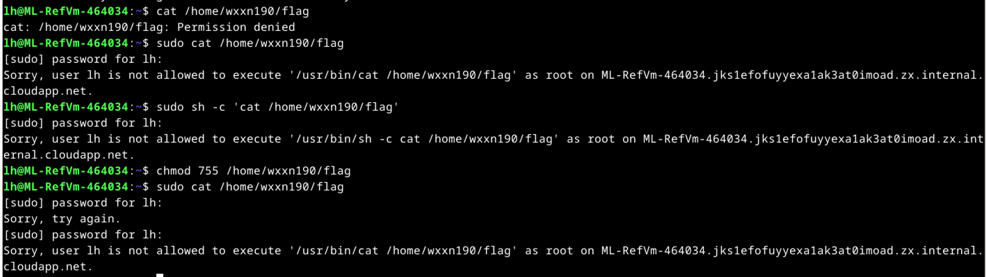
6Cby0cmMoQ had successful login



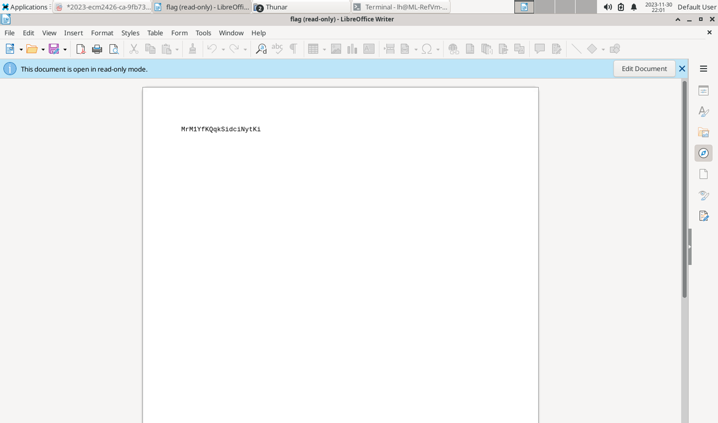
**E.1.3.**(5marks):

As what can be seen by the code I attempted trial and error until I noticed that the code that work was a simple ----> chmod 755 /home/wxxn190/flag

I then opened the file in a libre equivalent of word then read the file



MrM1YfKQqkSidciNytKi



**E.1.4.**(5marks):

I started by using ---> getfacl /home/wxxn190/flag

This returned --->

getfacl: Removing leading '/' from absolute path names

# file: home/wxxn190/flag

# owner: lh

# group: financials

user::rwx

user:bxxn580:r-x

user:exxn563:r-x

group::r--

mask::r-x

other::r-x



There are two users and financial group which consist of wxxn190, dxxl1909, rxxe878

**Exercise 2 (15 marks)**

**E.2.1.** (2 marks):

Signature Algorithm: 1.2.840.113549.1.1.11

In summary, the certificate is signed using the SHA-256 with RSA Encryption algorithm.

**E.2.2.** (2 marks):

The key length used by the signature algorithm (RSA) for signing the certificate can be determined from the "Public Key Info" section of the certificate information. In this case, the key size is mentioned under the "Key Size" field. The key size used for the RSA algorithm in this certificate is 4096 bits. This indicates that a 4096-bit RSA key pair is employed for the signature algorithm used to sign the certificate.

Public Key Info

Key Algorithm: RSA

Key Parameters: 05 00

Key Size: 4096

**E.2.3.** (3 marks):

To determine if the certificate is valid for the domain www1.exeter.ac.uk, you should look at the "Subject Alternative Names" extension in the certificate information. This extension lists additional domain names for which the certificate is valid.

Based on the information, the certificate is not valid for the domain www1.exeter.ac.uk.

This was the code in the certificate --->

**“””**

Subject Alternative Names

DNS: smpt.strathclyde.ac.uk www.portsmouth.ac.uk \*.wolverhampton.ac.uk \*.middlesex.ac.uk [www.dundee.ac.uk](http://www.dundee.ac.uk)

**“””**

The Subject Alternative Names list only includes domains such as smpt.strathclyde.ac.uk, [www.portsmouth.ac.uk](http://www.portsmouth.ac.uk), \*.wolverhampton.ac.uk, \*.middlesex.ac.uk, and [www.dundee.ac.uk](http://www.dundee.ac.uk). It does not explicitly mention www1.exeter.ac.uk.

**E.2.4.** (3 marks):

The certificate is a wildcard SSL certificate that is valid for all subdomains of exeter.ac.uk. This means that it is valid for www3.exeter.ac.uk.

the Common Name (CN) is \*.exeter.ac.uk. The asterisk (\*) in the CN field is a wildcard character, which means the certificate is valid for all subdomains of exeter.ac.uk.

Therefore, the certificate is valid for the domain www3.exeter.ac.uk.

Please note that the certificate is only valid if the domain is accessed over HTTPS, which is the protocol that uses SSL/TLS certificates for secure communication. If the domain is accessed over HTTP, the browser will not use the SSL/TLS certificate, and the connection will not be secure [**9**](https://www.123-reg.co.uk/support/ssl-certificates/verifying-domain-ownership-with-html-or-dns/).

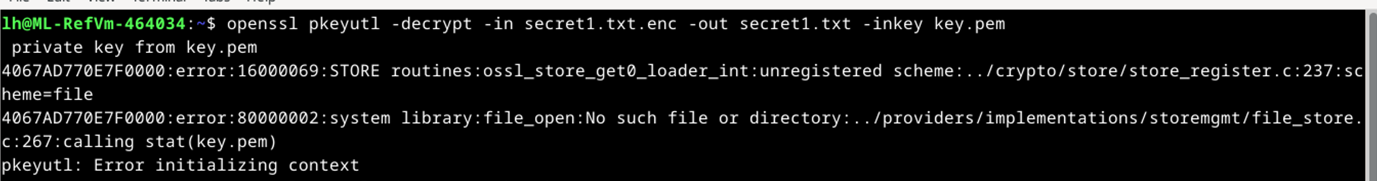
(double check this one)

**E.2.5.** (5 marks):

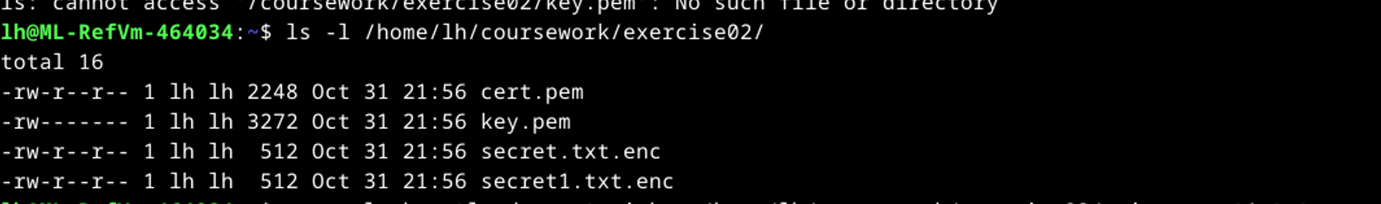
To start exercise by trying --->

openssl rsautl -decrypt -in secret1.txt.enc -out secret1.txt -inkey key.pem

but got an error relating to wrong openSLL type “rsautl” and location



then I checked the reading permissions of the file ---> ls -l /home/lh/coursework/exercise02/



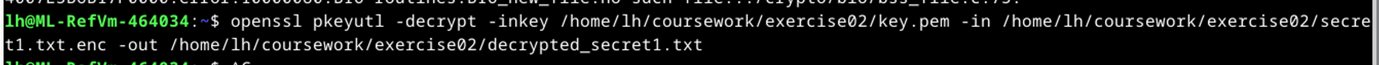
part 1

I was using a new code so I made a copy “secret.txt.enc” called “secret1.txt.enc” in case anything happned:

I was able to make an output.txt called “decrypt\_secret1.txt”

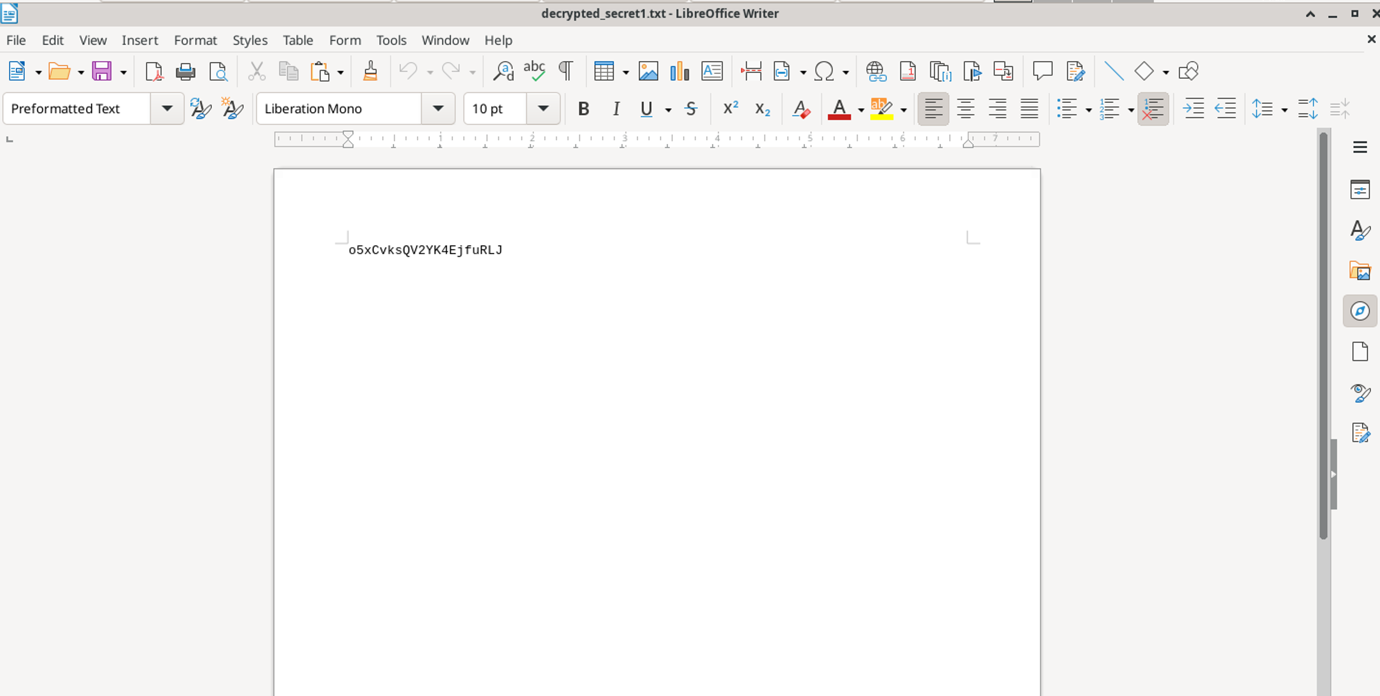
And using the key in the key.pem file

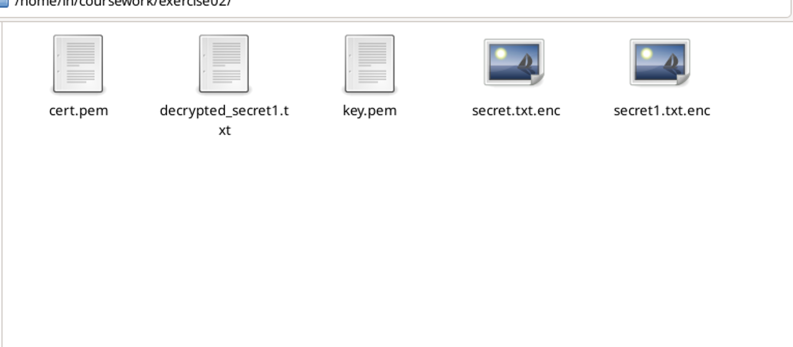
The code --->

openssl pkeyutl -decrypt -inkey /home/lh/coursework/exercise02/key.pem -in /home/lh/coursework/exercise02/secret1.txt.enc -out /home/lh/coursework/exercise02/decrypted\_secret1.txt

I then got a new txt file called”decrypt\_secret1.txt in the exercise02 folder and opened it and got this:

The code is the same in both files

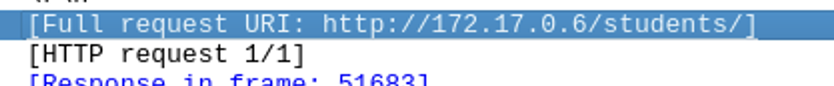
o5xCvksQV2YK4EjfuRLJ



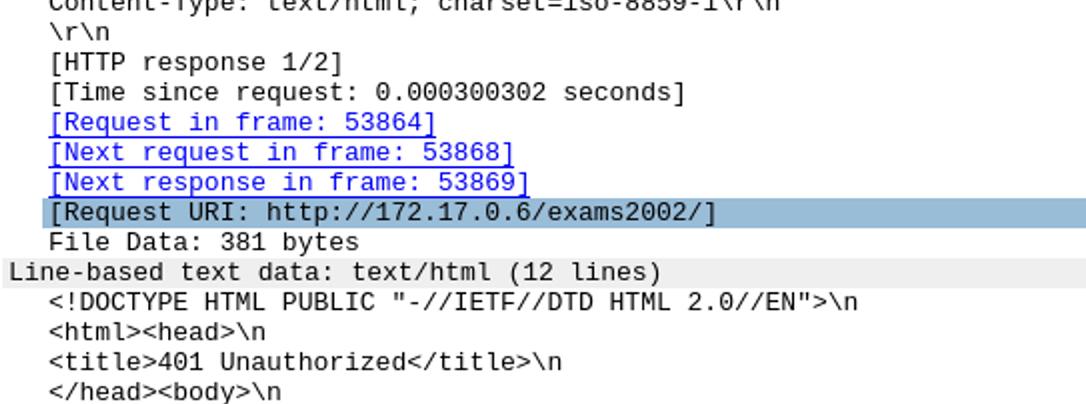
**Exercise 3**

**(15 marks)**

**E.3.1.**(5marks):

[http://172.17.0.6/exams2002/](http://172.17.0.6/exams2002/)

[http://172.17.0.6/exams2002/  
](http://172.17.0.6/exams2002/)

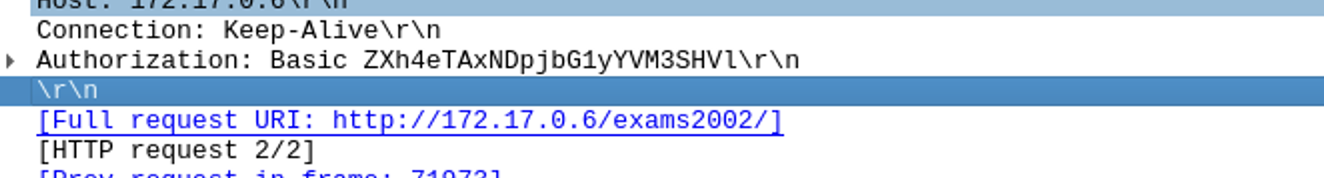


**E.3.2.** (5 marks):

# Trial 1

Got the code from inspecting pacjets

Basic ZXh4eTAxNDpjbG1yYVM3SHVl



To then get the user that got acess I used the code -->

**“””**

import base64

encoded\_string = "ZXh4eTAxNDpjbG1yYVM3SHVl"

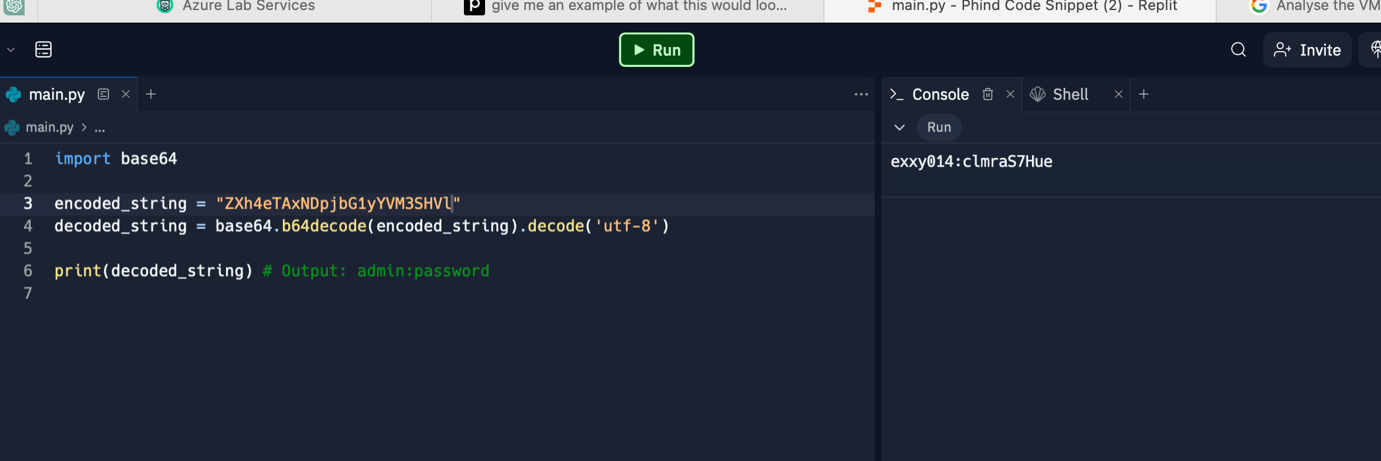
decoded\_string = base64.b64decode(encoded\_string).decode('utf-8')

print(decoded\_string) # Output: admin:password

**“””**

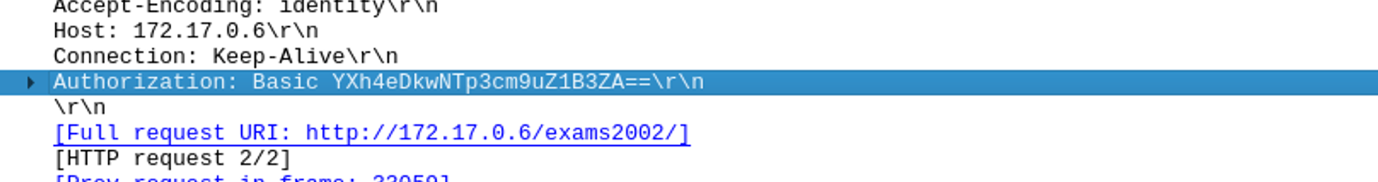
this retuned --> exxy014:clmraS7Hue

so the user that access the packet is exxy014



# Trial

This one here if you put code through same python software you get: axxx905:wrongPwd



**E.3.3.**(5marks):

Got this from earlier

exxy014:clmraS7Hue

so the password is: clmraS7Hue

**Exercise 4**

Types: Agent B,F;

Number NB,NF;

Symmetric\_key Key;

Function pk,f

Knowledge: B: F,pk(B),pk(F),inv(pk(B)),f;

F: <fact>, inv(pk(F)),f

Actions:

F->B: F,{NF,F}pk(B)

B->F: {f(NF),NB,B,Key}pk(F)

?->?: ?

Goals:

B \*->\* F: Key

**E.4.1** (5 marks):

<Fact>: F, pk(B)

**E.4.2** (2 marks):

In the protocol "ecm2426," the last message required to complete the protocol securely is initiated by Agent B. Agent B needs to send the last message in the protocol. Specifically, the last action is: --->

B->F: {f(NF),NB,B,Key}pk(F)

Therefore, Agent B is responsible for sending the last message to complete the protocol securely.

**E.4.3** (3 marks):

In the protocol "ecm2426," the last message required to complete the protocol securely is received by Agent F. The relevant action is: --->

B->F: {f(NF),NB,B,Key}pk(F)

Therefore, Agent F is the recipient of the last message required to complete the protocol securely

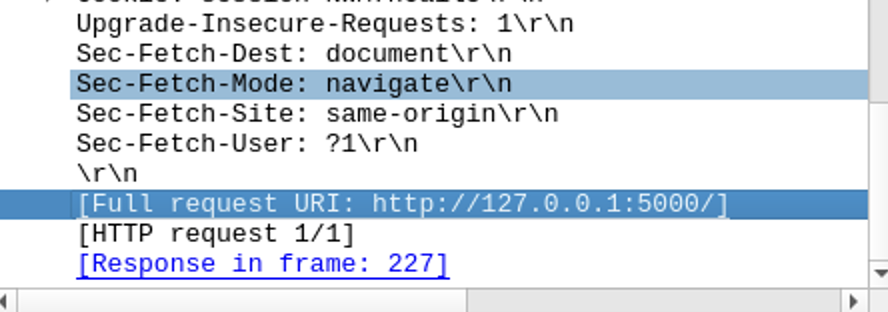
**E.4.4** (5 marks):

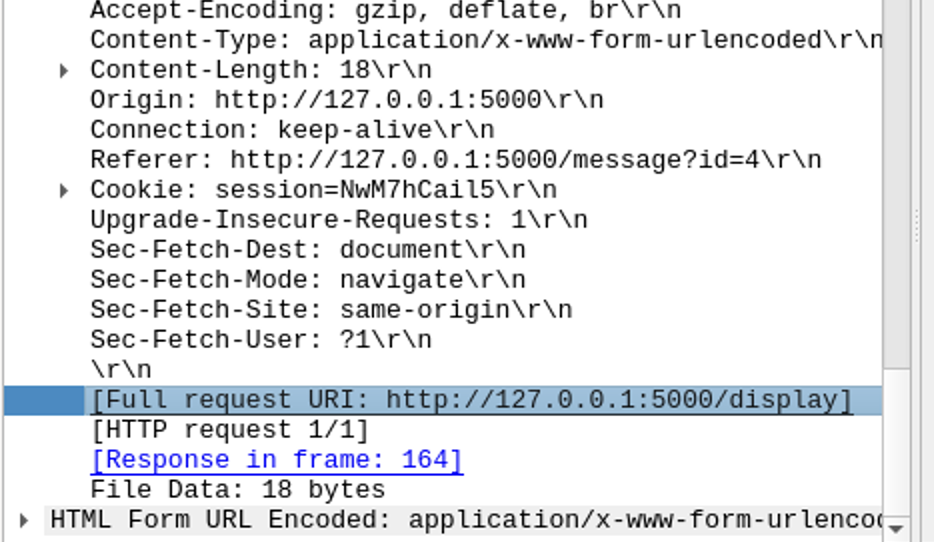
The last message exchanged in the last protocol step required to complete the "ecm2426" protocol securely is as follows:

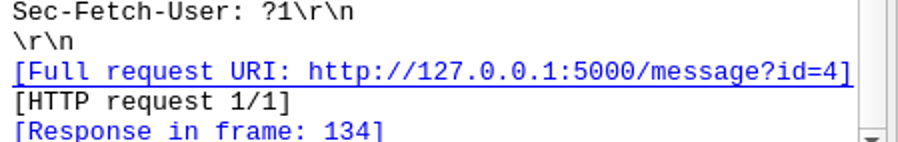
B->F: {f(NF), NB, B, Key}pk(F)

In this message, Agent B sends an encrypted payload to Agent F. The payload includes the values f(NF), NB, B, and Key, encrypted with the public key of Agent F (pk(F)). This message is crucial for completing the protocol securely as it provides the necessary information for Agent F to obtain the symmetric key (Key).

**E5.1.** (5 marks):



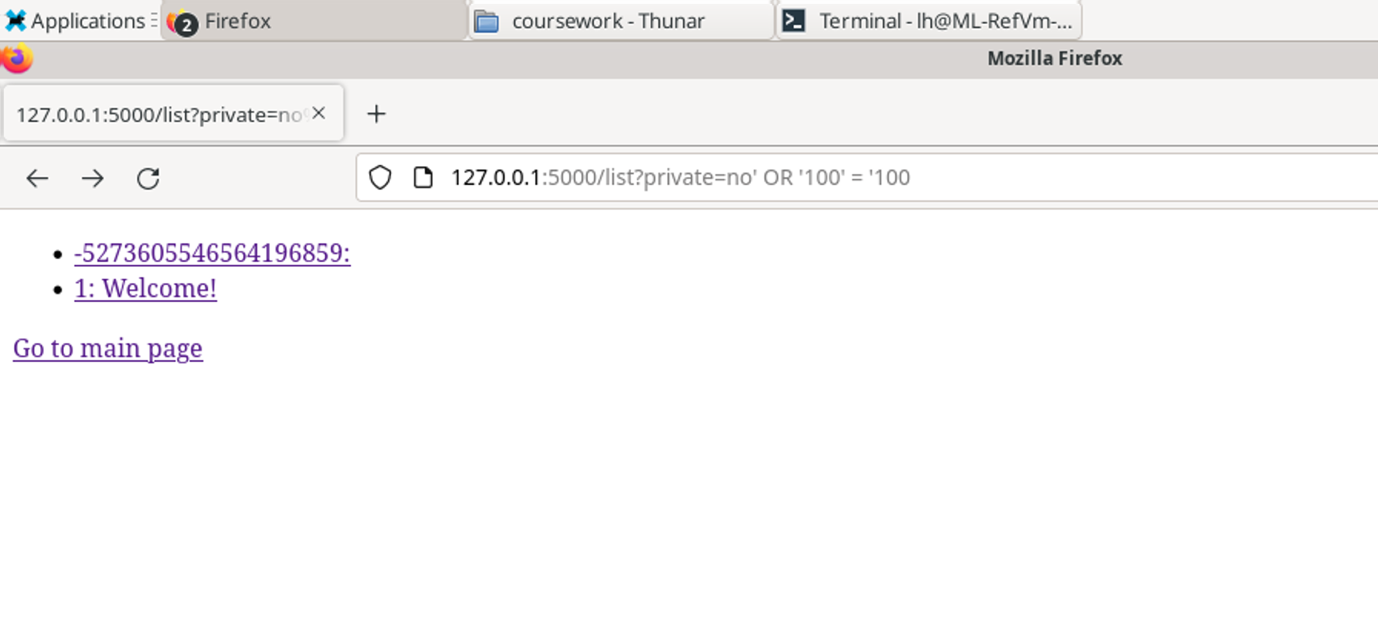




<http://127.0.0.1:5000/list?private=no%27%20OR%20%27100%27%20=%20%27100>

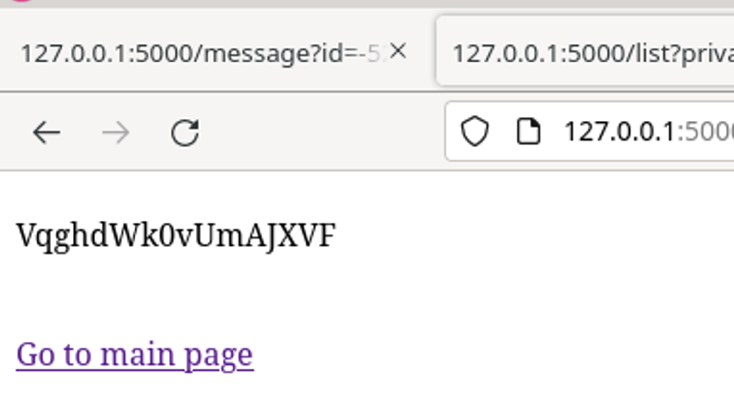
[http://127.0.0.1:5000/list?private=no’ OR ‘100’ = ’100](http://127.0.0.1:5000/list?private=no’%20OR%20‘100’%20=%20’100)  
  
<http://127.0.0.1:5000/list?private=no%27%20OR%20%27100%27%20=%20%27100>

127.0.0.1:5000/list?private=no' OR '100' = '100



**E5.2.** (5 marks):

The seceret message looked like : VqghdWk0vUmAJXVF

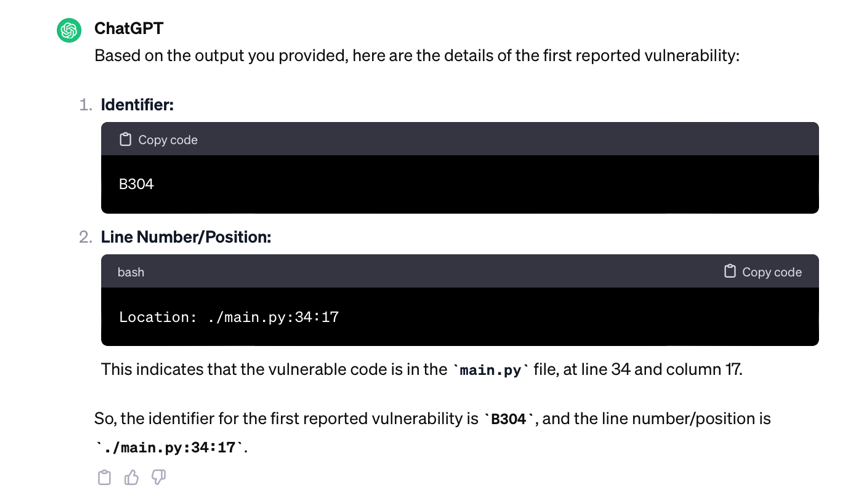


**E5.3.** (5 marks):

**E5.4.** (5 marks):

E6.1

doesn't change the fact that Bandit has identified vulnerabilities in the specified code. The identified vulnerability, in this case, is related to the use of an insecure cipher (DES) and has the identifier **B304** with the location in the **main.py** file at line **34** and column **17**.



E6.2

The first reported vulnerability involves the use of an insecure cipher (DES), and it is marked with the identifier **B304**. This vulnerability is considered high severity, but whether it can be exploited depends on the specific context and the overall security architecture of the application.

In general, the use of insecure ciphers, such as DES, is a security risk. DES is an outdated and weak cryptographic algorithm, and it's generally recommended to use more secure alternatives like AES.

E6.3

he second reported vulnerability involves again the use of an insecure cipher (DES), and it is marked with the identifier **B304**. This vulnerability is also considered high severity. Here are the details:

* Identifier: B304
* Line Number/Position: ./main.py:41:17

e.6.4

yes:   
 Yes, the second reported vulnerability (B304) involves the use of an insecure cipher (Cryptodome.Cipher.DES.new) and is considered a high-severity issue. While Bandit provides information about the security risk, it does not explicitly state whether an issue can be exploited. However, the use of an insecure cipher like DES is generally considered a serious security concern, and an attacker with the knowledge of cryptographic weaknesses in DES could potentially exploit this vulnerability.

e.6.5

The third reported vulnerability has the identifier B608, and the line number/position is 145:16. This vulnerability is flagged as a potential SQL injection vector through string-based query construction.

e.6.6

Yes, the third reported vulnerability (B608) is flagged as a potential SQL injection vector through string-based query construction. SQL injection vulnerabilities can potentially be exploited by attackers to manipulate or inject malicious SQL queries into the application's database queries, leading to unauthorized access or modification of data. It is considered a medium-severity vulnerability with medium confidence.