

Using Encryption to Enhance Confidentiality and Integrity (4e)

Fundamentals of Information Systems Security, Fourth Edition - Lab 05

Student:

James Sacharanski

Email:

sacharanskij@gmail.com

Time on Task:

4 hours, 30 minutes

Progress:

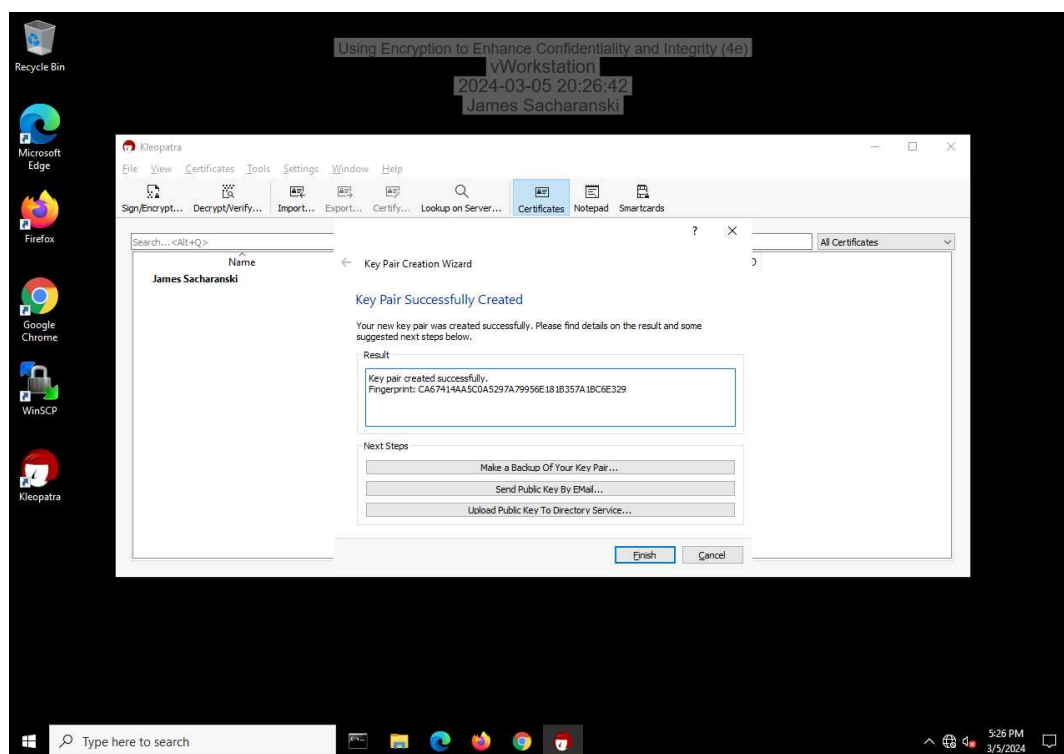
100%

Report Generated: Tuesday, March 5, 2024 at 9:52 PM

Section 1: Hands-On Demonstration

Part 1: Create and Exchange Asymmetric Encryption Keys

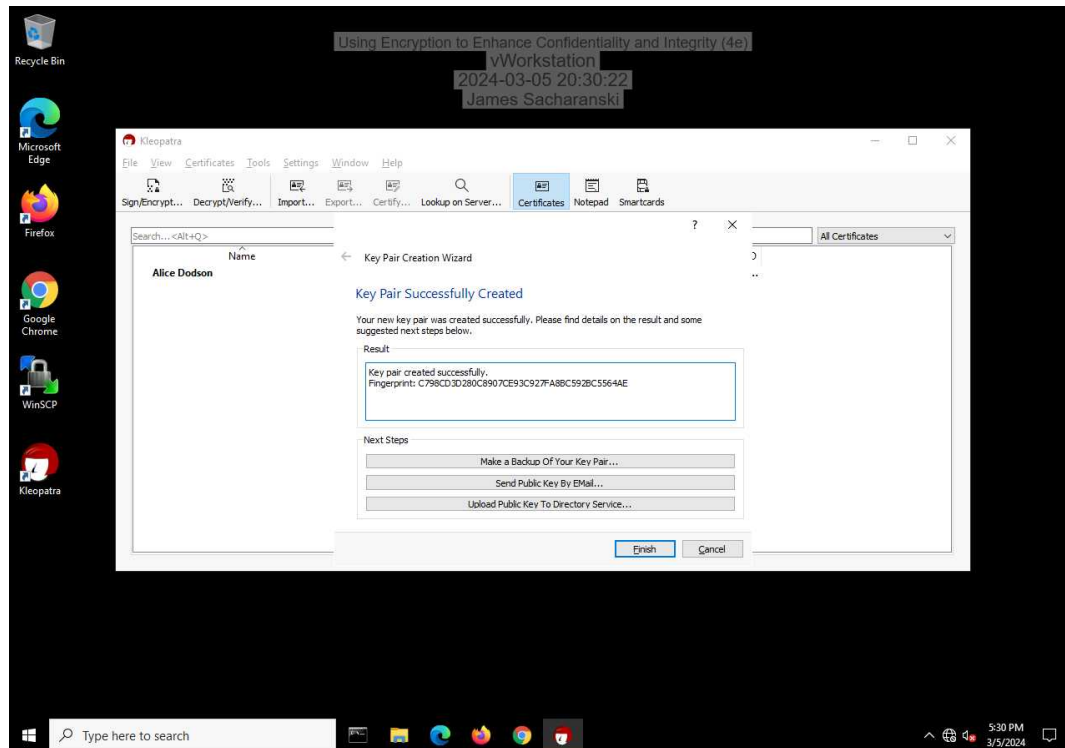
9. Make a screen capture showing the **fingerprint** for your key pair.



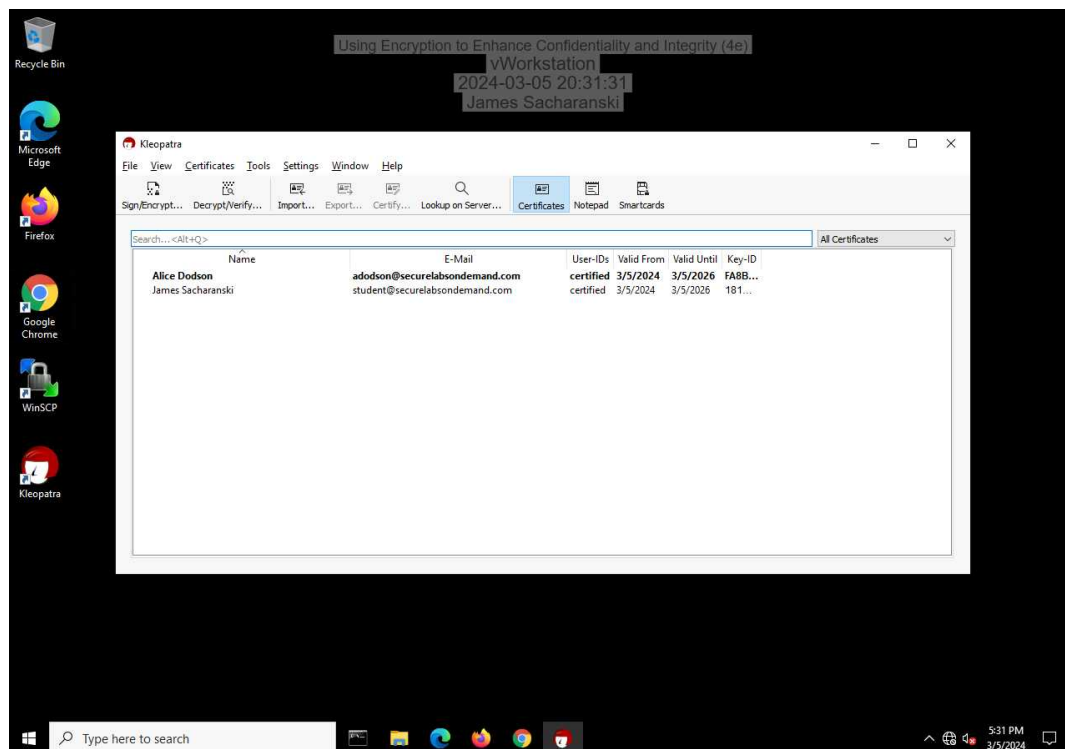
Using Encryption to Enhance Confidentiality and Integrity (4e)

Fundamentals of Information Systems Security, Fourth Edition - Lab 05

22. Make a screen capture showing the fingerprint for Alice's key pair.



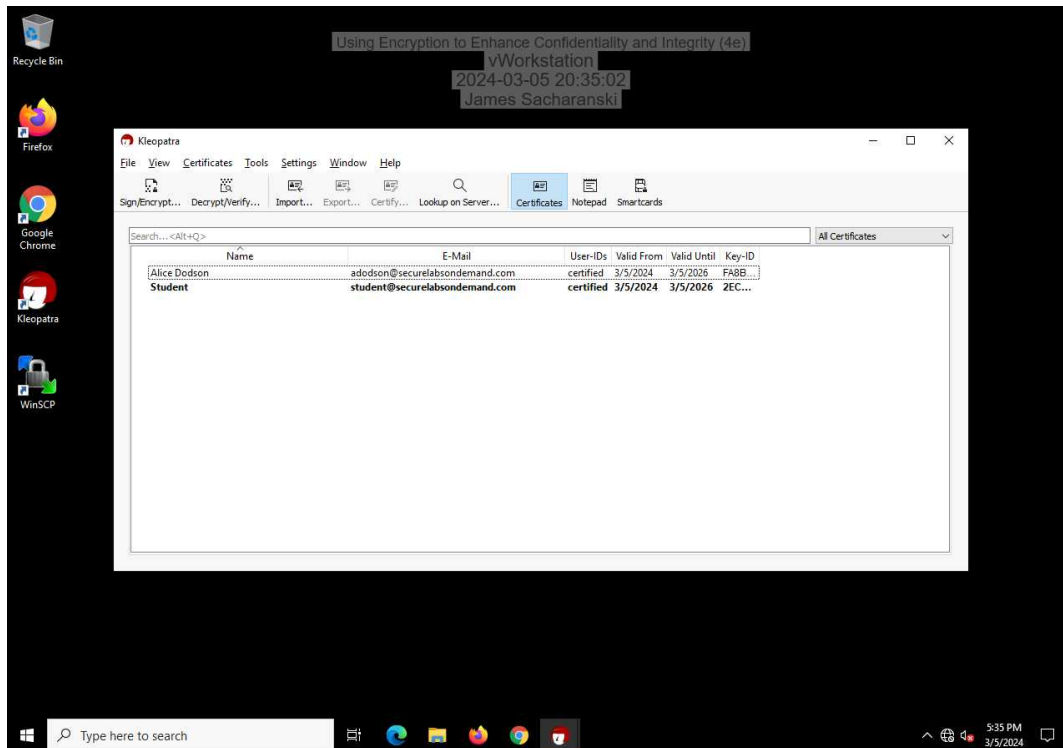
30. Make a screen capture showing your public key in Alice's certificate cache.



Using Encryption to Enhance Confidentiality and Integrity (4e)

Fundamentals of Information Systems Security, Fourth Edition - Lab 05

35. Make a screen capture showing Alice's public key in your certificate cache.

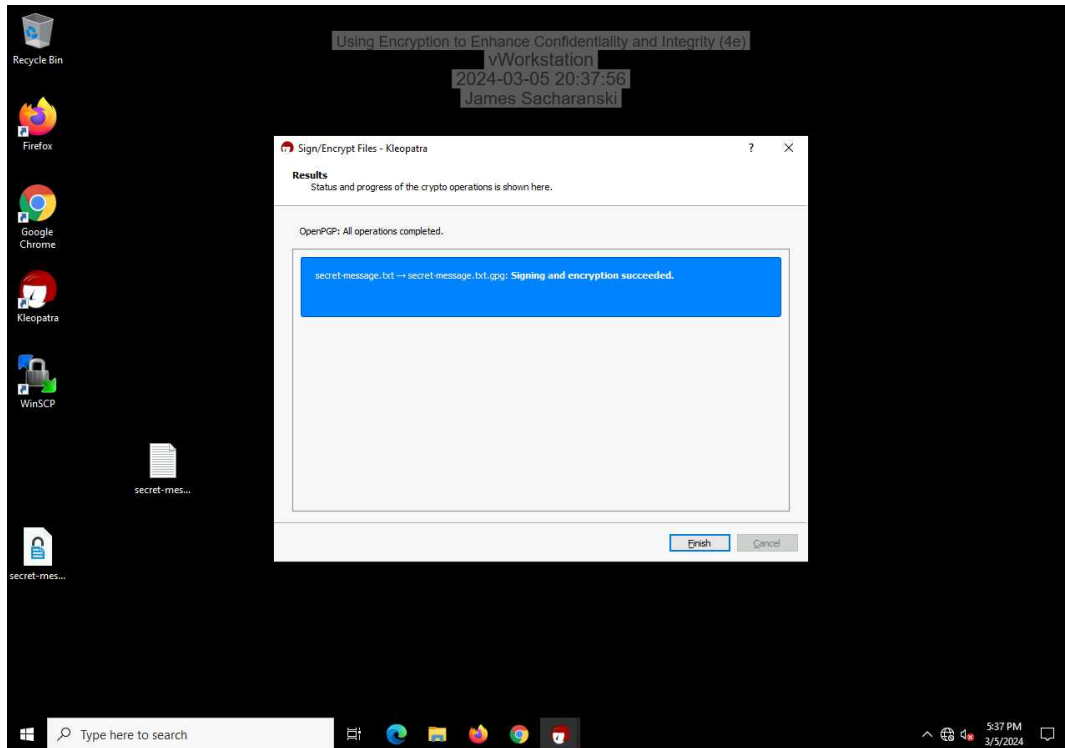


Part 2: Encrypt a File Using Asymmetric Encryption

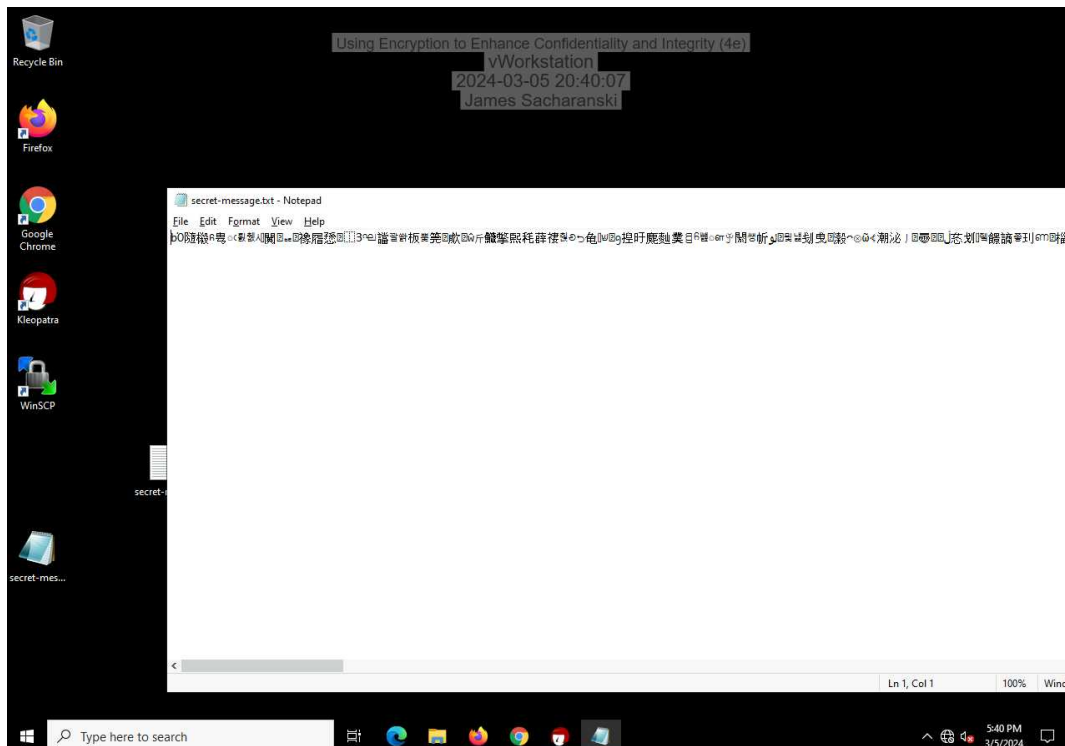
Using Encryption to Enhance Confidentiality and Integrity (4e)

Fundamentals of Information Systems Security, Fourth Edition - Lab 05

9. Make a screen capture showing the **successful signing and encryption message**.

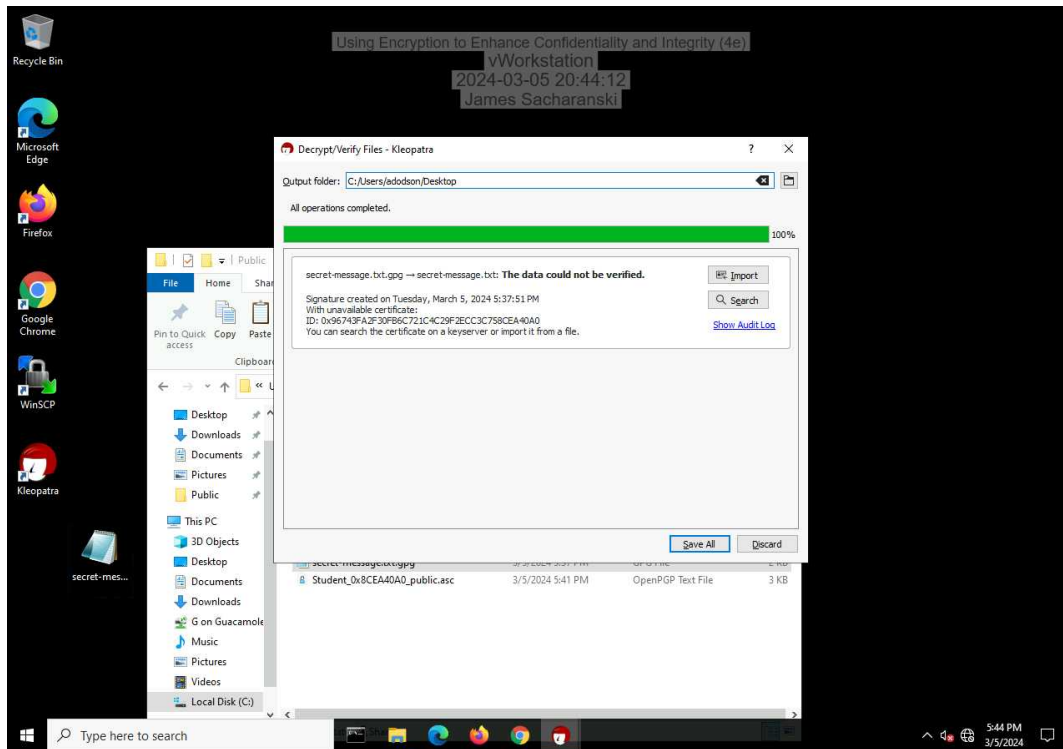


12. Make a screen capture showing the **ciphertext**.



Part 3: Decrypt a File Using Asymmetric Encryption

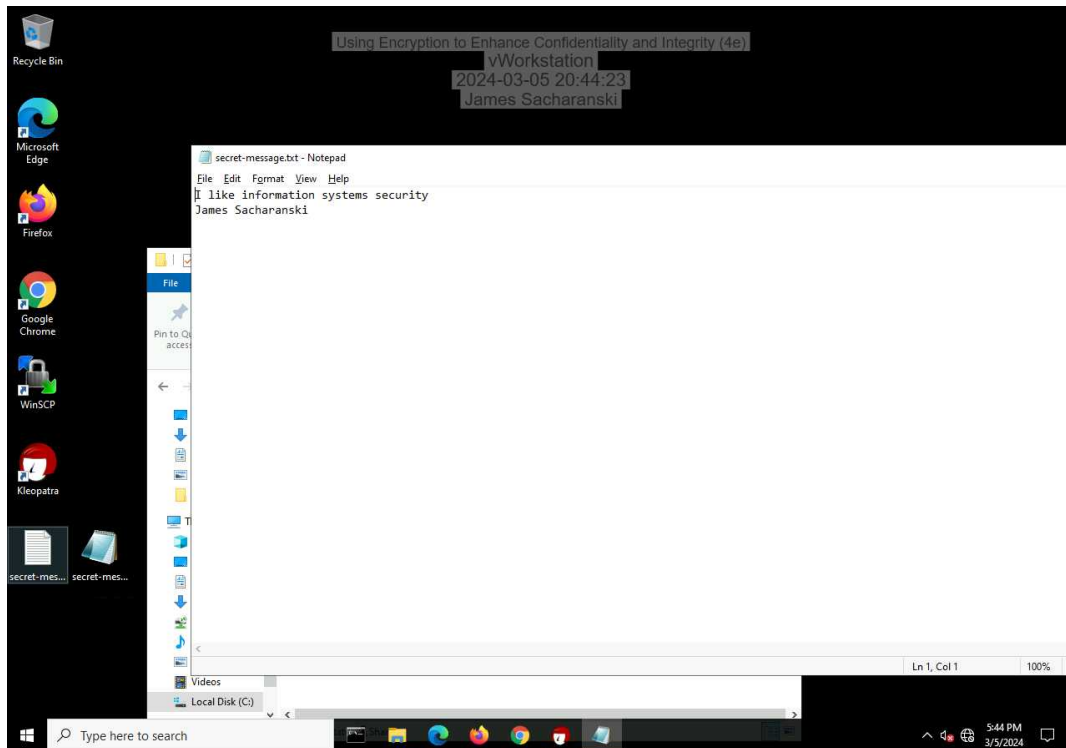
15. Make a screen capture showing the **Decrypt/Verify Files** window.



Using Encryption to Enhance Confidentiality and Integrity (4e)

Fundamentals of Information Systems Security, Fourth Edition - Lab 05

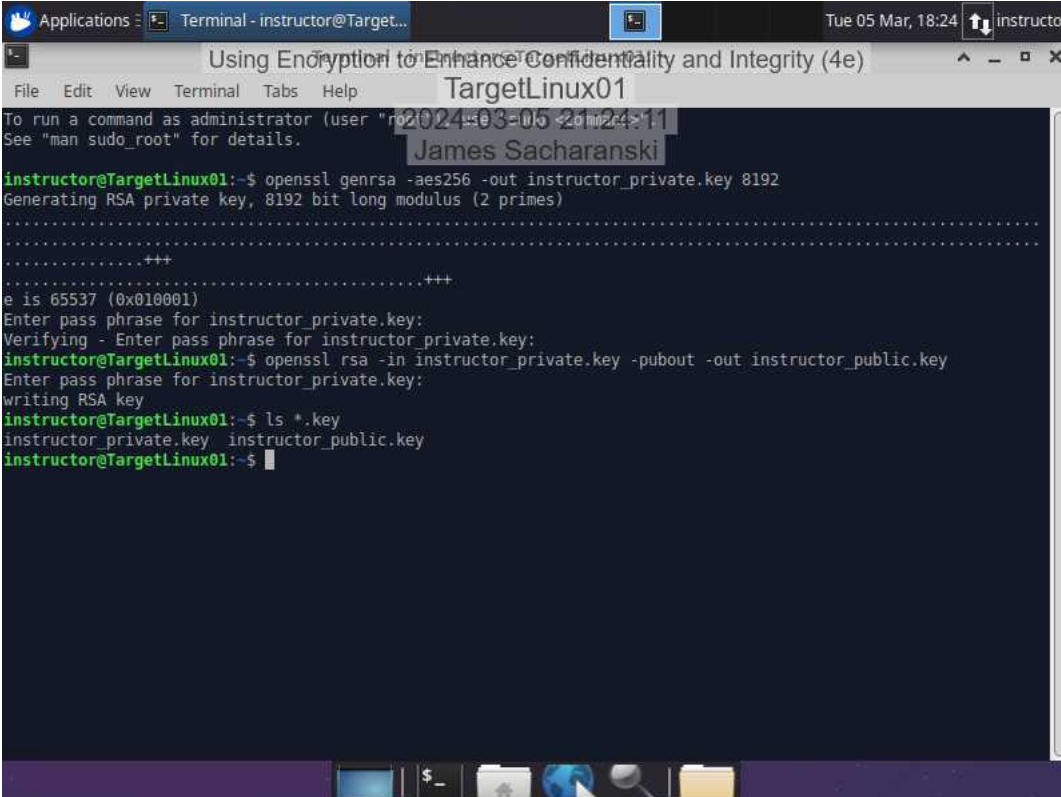
18. Make a screen capture showing the **decrypted secret-message.txt** file in Notepad.



Section 2: Applied Learning

Part 1: Create an Asymmetric Key Pair

10. Make a screen capture showing the instructor's key pair files.



The screenshot shows a terminal window titled "Terminal - instructor@Target..." with a dark background. The terminal output is as follows:

```
instructor@TargetLinux01:~$ openssl genrsa -aes256 -out instructor_private.key 8192
Generating RSA private key, 8192 bit long modulus (2 primes)
.....+++
.....+++
e is 65537 (0x010001)
Enter pass phrase for instructor_private.key:
Verifying - Enter pass phrase for instructor_private.key:
instructor@TargetLinux01:~$ openssl rsa -in instructor_private.key -pubout -out instructor_public.key
Enter pass phrase for instructor_private.key:
writing RSA key
instructor@TargetLinux01:~$ ls *.key
instructor_private.key  instructor_public.key
instructor@TargetLinux01:~$
```

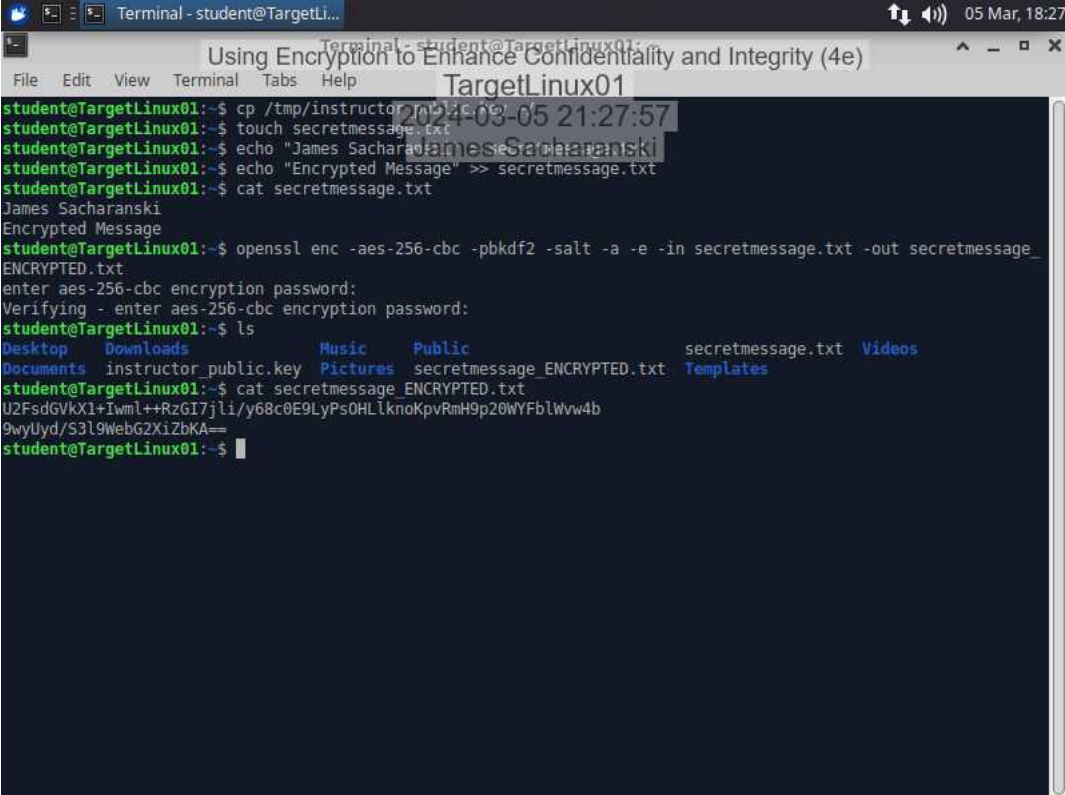
Overlaid on the terminal window are several semi-transparent text elements: "Using Encryption to Enhance Confidentiality and Integrity (4e)" at the top, "TargetLinux01" in the center, "2024-03-05 21:24:11" on the right, and "James Sacharanski" at the bottom.

Part 2: Encrypt a File Using Symmetric Encryption

11. Document the password you used to symmetrically encrypt the file.

james

13. Make a screen capture showing the ciphertext in the `secretmessage_ENCRYPTED.txt` file.

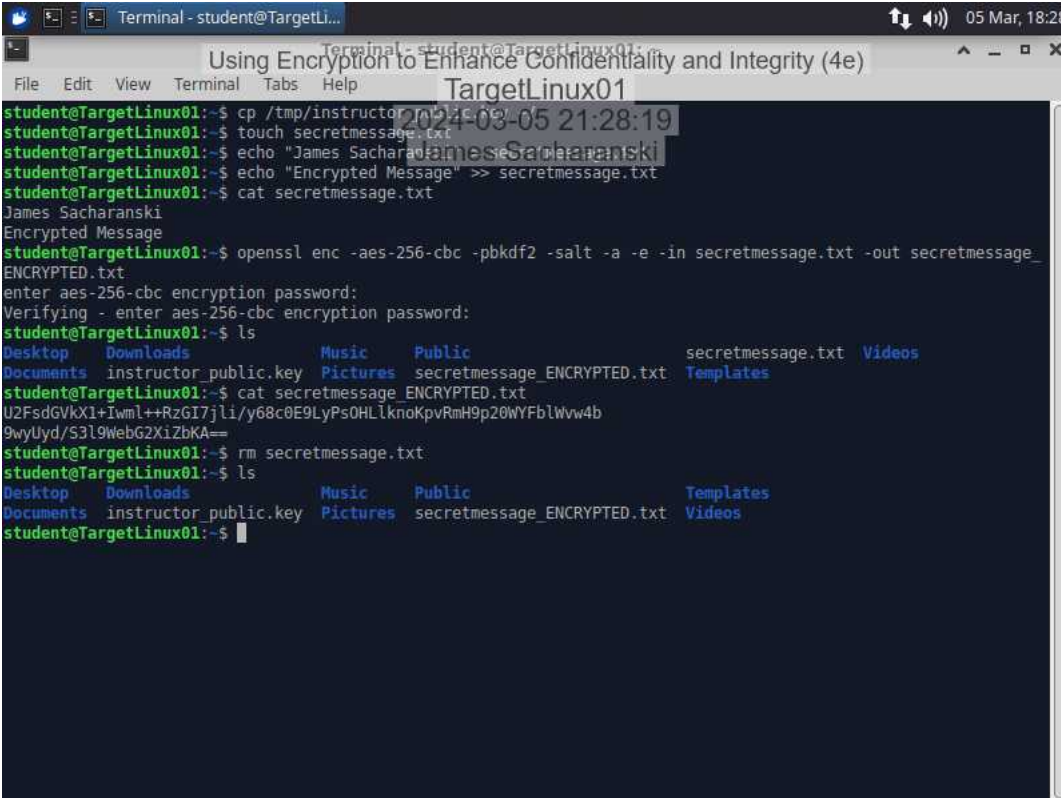
A terminal window titled "Terminal - student@TargetLinux01" with a menu bar (File, Edit, View, Terminal, Tabs, Help) and window controls. The terminal shows a series of commands and their outputs. A timestamp "2024-03-05 21:27:57" is visible. The commands include creating a file, adding content, and using openssl to encrypt the file. The final output shows the ciphertext in the file.

```
student@TargetLinux01:~$ cp /tmp/instructor_public.key .
student@TargetLinux01:~$ touch secretmessage.txt
student@TargetLinux01:~$ echo "James Sacharanski" >> secretmessage.txt
student@TargetLinux01:~$ echo "Encrypted Message" >> secretmessage.txt
student@TargetLinux01:~$ cat secretmessage.txt
James Sacharanski
Encrypted Message
student@TargetLinux01:~$ openssl enc -aes-256-cbc -pbkdf2 -salt -a -e -in secretmessage.txt -out secretmessage_ENCRYPTED.txt
enter aes-256-cbc encryption password:
Verifying - enter aes-256-cbc encryption password:
student@TargetLinux01:~$ ls
Desktop  Downloads  Music  Public  secretmessage.txt  Videos
Documents  instructor_public.key  Pictures  secretmessage_ENCRYPTED.txt  Templates
student@TargetLinux01:~$ cat secretmessage_ENCRYPTED.txt
U2FsdGVkX1+Iwml++RzGI7jli/y68c0E9LyPs0HLlknoKpvRmH9p20WYFb1Wvw4b
9wyUyd/S3l9WebG2XiZbKA==
student@TargetLinux01:~$
```


Using Encryption to Enhance Confidentiality and Integrity (4e)

Fundamentals of Information Systems Security, Fourth Edition - Lab 05

16. Make a screen capture showing the output of the ls command.

A terminal window titled 'Terminal - student@TargetLinux01' with a menu bar (File, Edit, View, Terminal, Tabs, Help) and a title bar (Using Encryption to Enhance Confidentiality and Integrity (4e)). The terminal shows a series of commands and their outputs. The user creates a file 'secretmessage.txt', adds content, and then encrypts it using 'openssl enc -aes-256-cbc -pbkdf2 -salt -a -e -in secretmessage.txt -out secretmessage_ENCRYPTED.txt'. After encryption, the user runs 'ls' which shows the encrypted file in the directory. The user then removes the original file and runs 'ls' again, showing only the encrypted file.

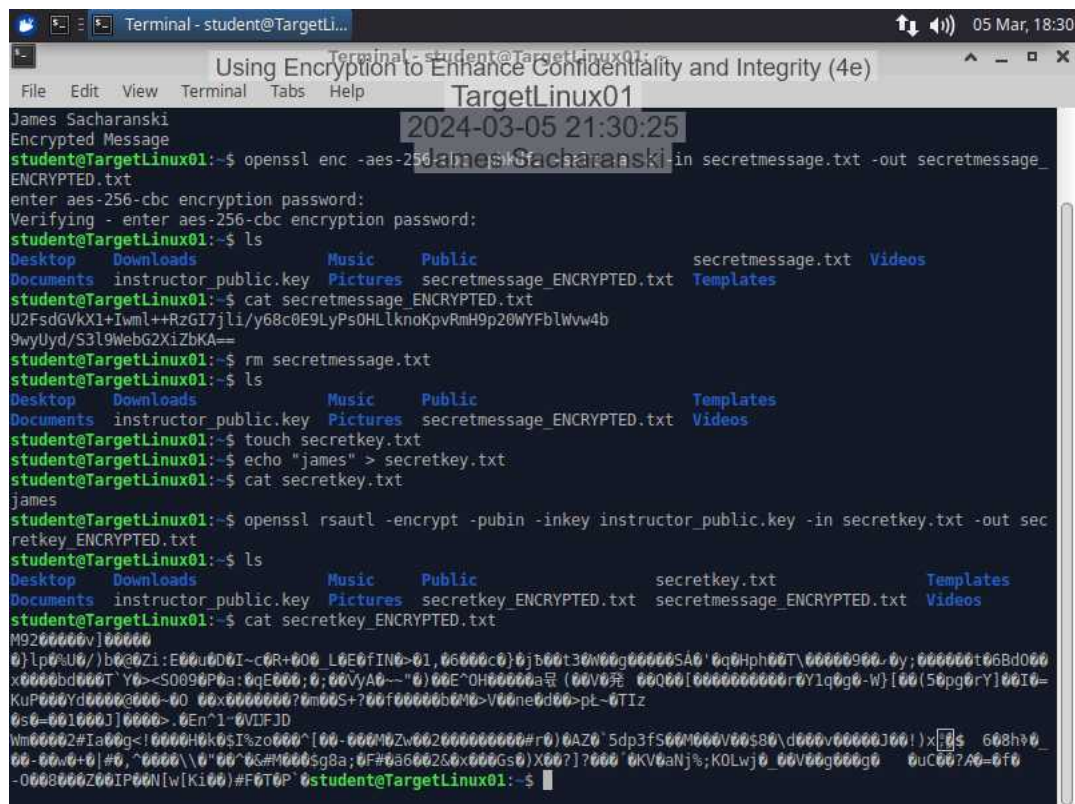
```
student@TargetLinux01:~$ cp /tmp/instructor/public.key .
student@TargetLinux01:~$ touch secretmessage.txt
student@TargetLinux01:~$ echo "James Sacharan" >> secretmessage.txt
student@TargetLinux01:~$ echo "Encrypted Message" >> secretmessage.txt
student@TargetLinux01:~$ cat secretmessage.txt
James Sacharan
Encrypted Message
student@TargetLinux01:~$ openssl enc -aes-256-cbc -pbkdf2 -salt -a -e -in secretmessage.txt -out secretmessage_
ENCRYPTED.txt
enter aes-256-cbc encryption password:
Verifying - enter aes-256-cbc encryption password:
student@TargetLinux01:~$ ls
Desktop  Downloads  Music  Public  secretmessage.txt  Videos
Documents  instructor.public.key  Pictures  secretmessage_ENCRYPTED.txt  Templates
student@TargetLinux01:~$ cat secretmessage_ENCRYPTED.txt
U2FsdGVkX1+Iwml++RzGI7jli/y68c0E9LyPs0HLlknokpvRmH9p20WYFb1Www4b
9wyUyd/S3l9WebG2XiZbKA==
student@TargetLinux01:~$ rm secretmessage.txt
student@TargetLinux01:~$ ls
Desktop  Downloads  Music  Public  Templates
Documents  instructor.public.key  Pictures  secretmessage_ENCRYPTED.txt  Videos
student@TargetLinux01:~$
```

Part 3: Transfer and Decrypt a File Using Hybrid Cryptography

Using Encryption to Enhance Confidentiality and Integrity (4e)

Fundamentals of Information Systems Security, Fourth Edition - Lab 05

6. Make a screen capture showing the encrypted contents of the `secretkey_ENCRYPTED.txt` file.

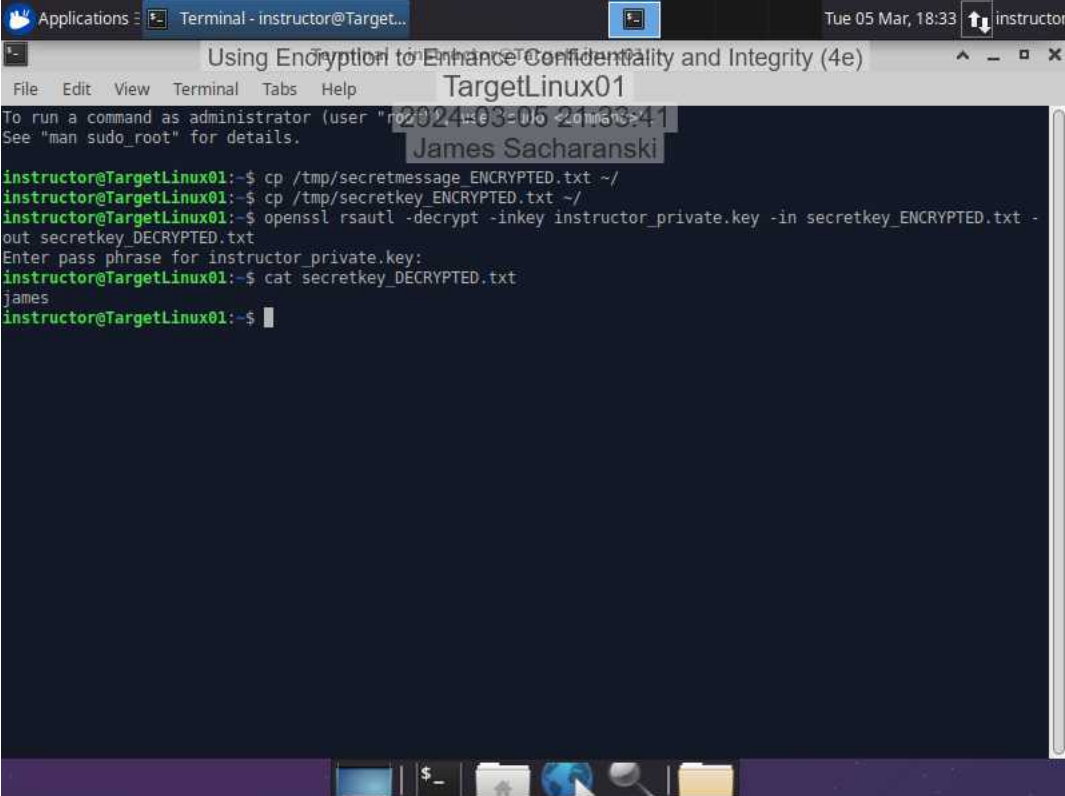


```
Terminal - student@TargetLinux01: ~
Using Encryption to Enhance Confidentiality and Integrity (4e)
TargetLinux01
James Sacharanski
Encrypted Message
2024-03-05 21:30:25
student@TargetLinux01:~$ openssl enc -aes-256-cbc -in secretmessage.txt -out secretmessage_ENCRYPTED.txt
enter aes-256-cbc encryption password:
Verifying - enter aes-256-cbc encryption password:
student@TargetLinux01:~$ ls
Desktop  Downloads  Music  Public  secretmessage.txt  Videos
Documents  instructor_public.key  Pictures  secretmessage_ENCRYPTED.txt  Templates
student@TargetLinux01:~$ cat secretmessage_ENCRYPTED.txt
U2FsdGVkX1+Iwml++RzGI7jli/y68c0E9LyPs0HLlknokpvmH9p20WYFb1Wvw4b
9wyUyd/S3L9WebG2XiZbKA==
student@TargetLinux01:~$ rm secretmessage.txt
student@TargetLinux01:~$ ls
Desktop  Downloads  Music  Public  secretmessage_ENCRYPTED.txt  Templates
Documents  instructor_public.key  Pictures
student@TargetLinux01:~$ touch secretkey.txt
student@TargetLinux01:~$ echo "james" > secretkey.txt
student@TargetLinux01:~$ cat secretkey.txt
james
student@TargetLinux01:~$ openssl rsautl -encrypt -pubin -inkey instructor_public.key -in secretkey.txt -out secretkey_ENCRYPTED.txt
student@TargetLinux01:~$ ls
Desktop  Downloads  Music  Public  secretkey.txt  Templates
Documents  instructor_public.key  Pictures  secretkey_ENCRYPTED.txt  secretmessage_ENCRYPTED.txt  Videos
student@TargetLinux01:~$ cat secretkey_ENCRYPTED.txt
M9200000v]00000
0}lp0W0/)b000Zi:E00u000I~c0R+000 L0E0fIN0>01,06000c0}0jb00t30w00g00000SA0'0q0Hph00T\00000900-0y;000000t06Bd000
x0000bd000T`Y0><S0090P0a:0qE000;0;00VyA0~~"0)00E^0H00000a0 (00V000 00Q00[000000000r0Y1q0g0-W][00(50pg0rY]00I0=
KuP000Yd00000000-00 00x0000000?0m00S+?00f00000b00%0>V00ne0d00>pt-0TIz
0s0=001000J]0000>.0En^1-0VIFJD
Wm00002#Ia00g<!0000H0k0$I%zo000^[00-000M0Zw00200000000#r0)0AZ0`5dp3fS000000V00$80\d000v00000J00!)x[0$ 608h0_
00-00w0+0|#0,^0000\0"00^0S#0000sgBa;0F#00600260x000Gs0)X00?]?000 0KV0aNj%;K0Lwj0_00V00g000g0 0uC00?A0=0f0
-0008000Z00IP00N[w[Ki00)#F0T0P'0student@TargetLinux01:~$
```

Using Encryption to Enhance Confidentiality and Integrity (4e)

Fundamentals of Information Systems Security, Fourth Edition - Lab 05

17. **Make a screen capture** showing the **decrypted contents of the secretkey_DECRYPTED.txt** file.

A terminal window titled "Terminal - instructor@Target..." is shown. The window has a menu bar with "File", "Edit", "View", "Terminal", "Tabs", and "Help". The title bar also includes "Applications", "Terminal - instructor@Target...", and a system clock showing "Tue 05 Mar, 18:33". The terminal content shows the following commands and output:

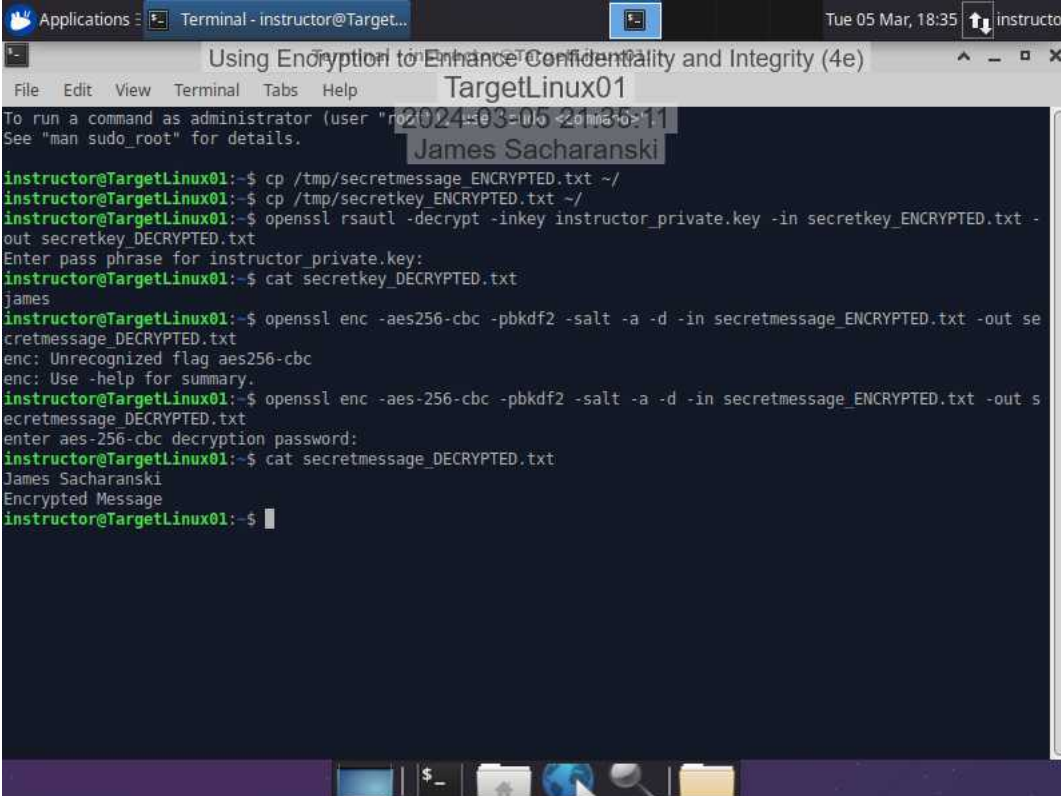
```
instructor@TargetLinux01:~$ cp /tmp/secretmessage_ENCRYPTED.txt ~/
instructor@TargetLinux01:~$ cp /tmp/secretkey_ENCRYPTED.txt ~/
instructor@TargetLinux01:~$ openssl rsautl -decrypt -inkey instructor_private.key -in secretkey_ENCRYPTED.txt -
out secretkey_DECRYPTED.txt
Enter pass phrase for instructor_private.key:
instructor@TargetLinux01:~$ cat secretkey_DECRYPTED.txt
james
instructor@TargetLinux01:~$
```

There are several redaction boxes over the terminal: a blue box with "2024-03-05 21:33:41" and "James Sacharanski" over the first two lines of output; a grey box with "TargetLinux01" over the prompt of the third line; and a grey box with "Using Encryption to Enhance Confidentiality and Integrity (4e)" over the menu bar.

Using Encryption to Enhance Confidentiality and Integrity (4e)

Fundamentals of Information Systems Security, Fourth Edition - Lab 05

21. Make a screen capture showing the contents of the `secretmessage_DECRYPTED` file.



The screenshot shows a terminal window titled "Terminal - instructor@Target..." with a menu bar containing "File", "Edit", "View", "Terminal", "Tabs", and "Help". The window displays the following commands and output:

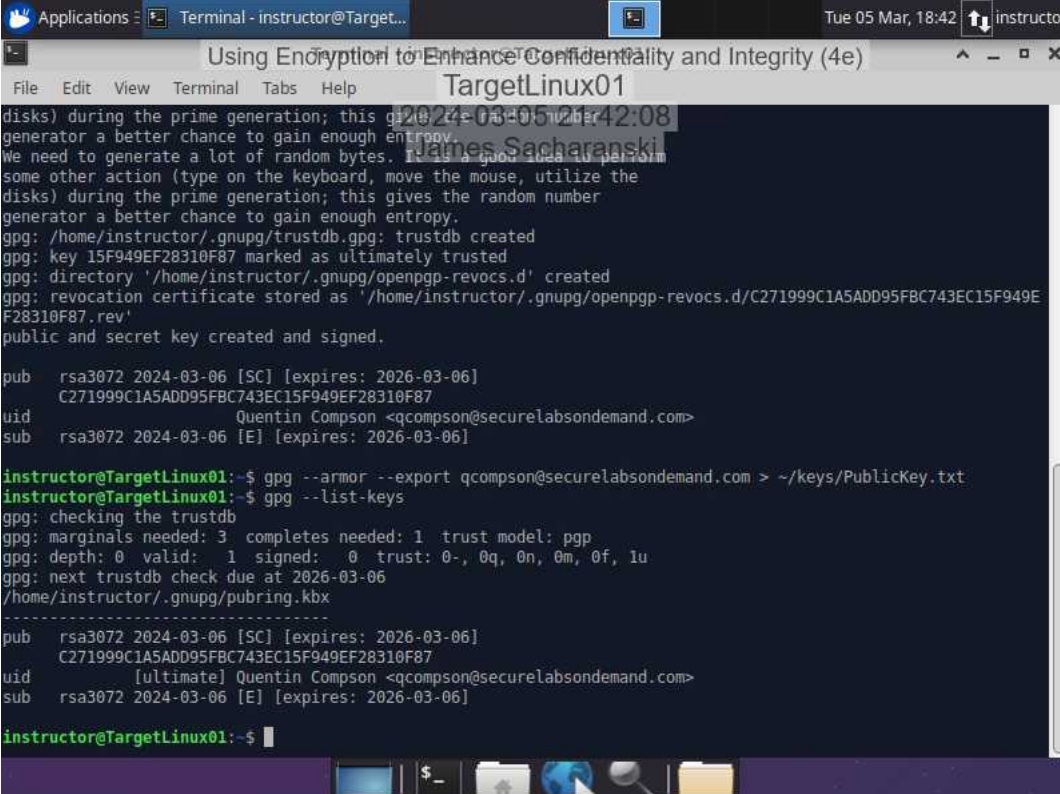
```
instructor@TargetLinux01:~$ cp /tmp/secretmessage_ENCRYPTED.txt ~/
instructor@TargetLinux01:~$ cp /tmp/secretkey_ENCRYPTED.txt ~/
instructor@TargetLinux01:~$ openssl rsautl -decrypt -inkey instructor_private.key -in secretkey_ENCRYPTED.txt -
out secretkey_DECRYPTED.txt
Enter pass phrase for instructor_private.key:
instructor@TargetLinux01:~$ cat secretkey_DECRYPTED.txt
james
instructor@TargetLinux01:~$ openssl enc -aes256-cbc -pbkdf2 -salt -a -d -in secretmessage_ENCRYPTED.txt -out se
cretmessage_DECRYPTED.txt
enc: Unrecognized flag aes256-cbc
enc: Use -help for summary.
instructor@TargetLinux01:~$ openssl enc -aes-256-cbc -pbkdf2 -salt -a -d -in secretmessage_ENCRYPTED.txt -out s
ecretmessage_DECRYPTED.txt
enter aes-256-cbc decryption password:
instructor@TargetLinux01:~$ cat secretmessage_DECRYPTED.txt
James Sacharanski
Encrypted Message
instructor@TargetLinux01:~$
```

The terminal output shows the successful decryption of the secret message, resulting in the text "James Sacharanski" and "Encrypted Message".

Section 3: Challenge and Analysis

Part 1: Digitally Sign a Document Using GPG

Make a screen capture showing the **key fingerprint** for the key pair you generated in this part of the lab.



The screenshot shows a terminal window titled "Terminal - instructor@TargetLinux01" with a timestamp of "Tue 05 Mar, 18:42". The window displays the output of GPG commands. It starts with instructions on how to generate a key pair, followed by the execution of `gpg --full-genkey`. The output shows the creation of a trust database, marking the key as ultimately trusted, and creating a revocation certificate. The key fingerprint is displayed as `pub rsa3072 2024-03-06 [SC] [expires: 2026-03-06] C271999C1A5ADD95FBC743EC15F949EF28310F87`. The user is identified as "Quentin Compson <qcompson@securelabsondemand.com>". The terminal then shows the execution of `gpg --armor --export qcompson@securelabsondemand.com > ~/keys/PublicKey.txt` and `gpg --list-keys`, which lists the generated key pair.

```
instructor@TargetLinux01:~$ gpg --full-genkey
gpg: disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
gpg: /home/instructor/.gnupg/trustdb.gpg: trustdb created
gpg: key 15F949EF28310F87 marked as ultimately trusted
gpg: directory '/home/instructor/.gnupg/openpgp-revocs.d' created
gpg: revocation certificate stored as '/home/instructor/.gnupg/openpgp-revocs.d/C271999C1A5ADD95FBC743EC15F949E
F28310F87.rev'
public and secret key created and signed.

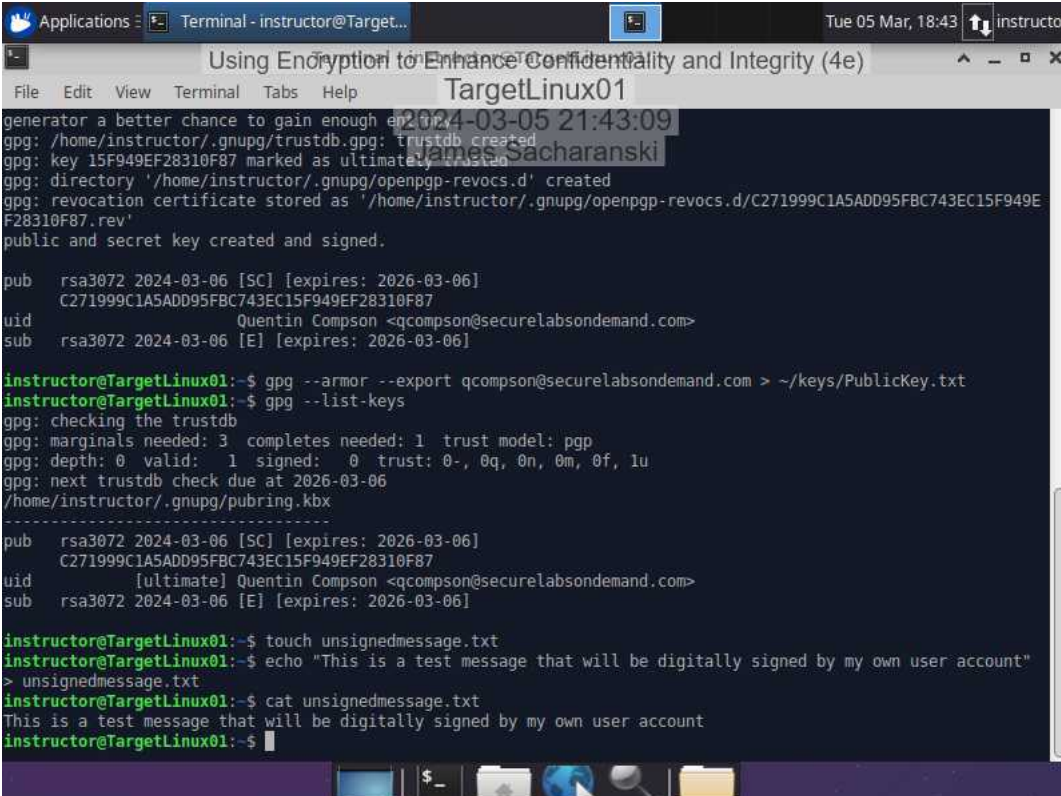
pub  rsa3072 2024-03-06 [SC] [expires: 2026-03-06]
     C271999C1A5ADD95FBC743EC15F949EF28310F87
uid          Quentin Compson <qcompson@securelabsondemand.com>
sub  rsa3072 2024-03-06 [E] [expires: 2026-03-06]

instructor@TargetLinux01:~$ gpg --armor --export qcompson@securelabsondemand.com > ~/keys/PublicKey.txt
instructor@TargetLinux01:~$ gpg --list-keys
gpg: checking the trustdb
gpg: marginals needed: 3  completes needed: 1  trust model: pgp
gpg: depth: 0  valid: 1  signed: 0  trust: 0-, 0q, 0n, 0m, 0f, 1u
gpg: next trustdb check due at 2026-03-06
/home/instructor/.gnupg/pubring.kbx
-----
pub  rsa3072 2024-03-06 [SC] [expires: 2026-03-06]
     C271999C1A5ADD95FBC743EC15F949EF28310F87
uid          [ultimate] Quentin Compson <qcompson@securelabsondemand.com>
sub  rsa3072 2024-03-06 [E] [expires: 2026-03-06]
```


Using Encryption to Enhance Confidentiality and Integrity (4e)

Fundamentals of Information Systems Security, Fourth Edition - Lab 05

Make a screen capture showing the contents of the unsignedmessage.txt file.



The screenshot shows a terminal window titled "Terminal - instructor@TargetLinux01" with a timestamp of "Tue 05 Mar, 18:43". The window displays the output of several GPG commands. It starts with a message about a better chance to gain enough entropy, followed by the creation of a trust database, marking a key as ultimately trusted, creating a revocation directory, and storing a revocation certificate. Then, a public and secret key pair is created and signed. The output shows the key details: a public key (rsa3072 2024-03-06 [SC] [expires: 2026-03-06]) and a subkey (rsa3072 2024-03-06 [E] [expires: 2026-03-06]). The user then exports the public key to a file, lists the keys, and checks the trust database. Finally, a file named "unsignedmessage.txt" is created and its contents are displayed: "This is a test message that will be digitally signed by my own user account".

```
generator a better chance to gain enough entropy
gpg: /home/instructor/.gnupg/trustdb.gpg: trustdb created
gpg: key 15F949EF28310F87 marked as ultimately trusted
gpg: directory '/home/instructor/.gnupg/openpgp-revocs.d' created
gpg: revocation certificate stored as '/home/instructor/.gnupg/openpgp-revocs.d/C271999C1A5ADD95FBC743EC15F949EF28310F87.rev'
public and secret key created and signed.

pub  rsa3072 2024-03-06 [SC] [expires: 2026-03-06]
     C271999C1A5ADD95FBC743EC15F949EF28310F87
uid          [ultimate] Quentin Compson <qcompson@securelabsondemand.com>
sub  rsa3072 2024-03-06 [E] [expires: 2026-03-06]

instructor@TargetLinux01:~$ gpg --armor --export qcompson@securelabsondemand.com > ~/keys/PublicKey.txt
instructor@TargetLinux01:~$ gpg --list-keys
gpg: checking the trustdb
gpg: marginals needed: 3  completes needed: 1  trust model: pgp
gpg: depth: 0  valid: 1  signed: 0  trust: 0-, 0q, 0n, 0m, 0f, 1u
gpg: next trustdb check due at 2026-03-06
/home/instructor/.gnupg/pubring.kbx
-----
pub  rsa3072 2024-03-06 [SC] [expires: 2026-03-06]
     C271999C1A5ADD95FBC743EC15F949EF28310F87
uid          [ultimate] Quentin Compson <qcompson@securelabsondemand.com>
sub  rsa3072 2024-03-06 [E] [expires: 2026-03-06]

instructor@TargetLinux01:~$ touch unsignedmessage.txt
instructor@TargetLinux01:~$ echo "This is a test message that will be digitally signed by my own user account"
> unsignedmessage.txt
instructor@TargetLinux01:~$ cat unsignedmessage.txt
This is a test message that will be digitally signed by my own user account
instructor@TargetLinux01:~$
```

Part 2: Verify the Digital Signature Using Kleopatra

Using Encryption to Enhance Confidentiality and Integrity (4e)

Fundamentals of Information Systems Security, Fourth Edition - Lab 05

Make a screen capture showing the successful signature verification on the signed message file.

