FUSE Using gocryptfs

Scenario:

Create and interact with a secure encrypted folder mounted via FUSE using gocryptfs.

1. Install gocryptfs

sudo apt update
sudo apt install gocryptfs

Command: sudo apt update and sudo apt install gocryptfs

Component	Description
sudo	**Superuser Do** – runs the following command with root privileges, required for system-level operations like package management.
apt	**Advanced Package Tool** – the command-line interface for handling packages on Debian-based systems (like Ubuntu), used to install, update, and remove software.
update	In the context of `apt`, it **updates the local package index** by retrieving the latest package lists from configured repositories. Does **not** install or upgrade packages.
install	In the context of `apt`, it tells `apt` to install the specified package(s) on the system.
gocryptfs	The **name of the package** to be installed – `gocryptfs` is a user-space encrypted filesystem that uses FUSE.

```
tiago-paquete@Linux:~$ sudo apt update
```

```
tiago-paquete@Linux:~$ sudo apt install gocryptfs
```

2. Prepare Encrypted & Mount Directories

mkdir -p ~/secure_data/encrypted ~/secure_data/decrypted

Command: mkdir -p ~/secure_data/encrypted ~/secure_data/decrypted

Component	Description
mkdir	**Make Directory** – command used to create one or more directories.
-p	**Parents** – tells `mkdir` to create parent directories as needed and **not to error** if the directories already exist.
~/secure_data/ encrypted	Path to the first directory to create. ~ represents the current user's home directory. This creates `secure_data/encrypted` within the home directory.
~/secure_data/ decrypted	**Path to the second directory** to create. Also located in the user's home directory under `secure_data/decrypted`.

```
tiago-paquete@Linux:~$ mkdir -p ~/secure_data/encrypted ~/secure_data/
decrypted
tiago-paquete@Linux:~$ ls ~/secure_data/encrypted ~/secure_data/
decrypted
/home/tiago-paquete/secure_data/decrypted:

/home/tiago-paquete/secure_data/encrypted:
tiago-paquete@Linux:~$
```

3. Initialize the Encrypted Filesystem

gocryptfs -init ~/secure_data/encrypted

Set a secure password.
This creates the gocryptfs.conf file with secure defaults.

Command: gocryptfs -init ~/secure_data/encrypted

Component	Description
gocryptfs	A user-space encrypted filesystem (FUSE-based). It allows you to encrypt a directory and mount it like a regular file system.
-init	Initializes a new encrypted filesystem in the specified directory. It creates a `gocryptfs.conf`and `masterkey` internally.
~/secure_data/ encrypted	Path to the directory where the encrypted filesystem should be initialized. ~ is a shortcut to the current user's home directory.

```
tiago-paquete@Linux:~$ gocryptfs -init ~/secure_data/encrypted
================================================================================
Choose a password for protecting your files.
Password:
Repeat:

Your master key is:

    bxxxx2-77xxxx06-xxxx1e63-3xxxxx03-
    xxxx676c-xxxxc1f1-c2xxxxdb-4xxxx5db

If the gocryptfs.conf file becomes corrupted or you ever forget your
password,
there is only one hope for recovery: The master key. Print it to a piece
of
paper and store it in a drawer. This message is only printed once.
The gocryptfs filesystem has been created successfully.
You can now mount it using: gocryptfs secure_data/encrypted MOUNTPOINT
================================================================================
```

4. Mount the Encrypted Filesystem

gocryptfs ~/secure_data/encrypted ~/secure_data/decrypted

Now:
You can access ~/secure_data/decrypted as a decrypted virtual filesystem.
Files written here are encrypted and stored in ~/secure_data/encrypted.

Command: gocryptfs ~/secure_data/encrypted ~/secure_data/decrypted

Component	Description
gocryptfs	A user-space encrypted filesystem based on FUSE (Filesystem in Userspace). It transparently encrypts files and directories.
~/secure_data/encrypted	Path to the **encrypted directory** (ciphertext view). This contains encrypted data and the `gocryptfs.conf` config file. The ~ symbol represents the current user's home directory.
~/secure_data/decrypted	Path to the **mount point** for the decrypted (plaintext) view. Once mounted, files in this directory appear decrypted and can be accessed like regular files.

```
tiago-paquete@Linux:~$ gocryptfs ~/secure_data/encrypted ~/secure_data/
decrypted
================================================================================
Password:
Decrypting master key
Filesystem mounted and ready.
================================================================================
```

5. Store a Secure File

echo "This is sensitive content" > ~/secure_data/decrypted/secret.txt

Command: echo "This is sensitive content" > ~/secure_data/decrypted/secret.txt

Component	Description
echo	**Echo (in bold)** – a built-in shell command used to output the given text or string to standard output (the terminal).
"This is sensitive content"	**String literal** – the text that will be printed or redirected; must be quoted to preserve spacing and special characters.
>	**Redirection operator** – redirects the output from the left-hand command (`echo`) into the file specified on the right-hand side. Overwrites the file if it already exists.
~/secure_data/decrypted/secret.txt	**File path** – specifies the destination file. ~ is a shortcut for the current user's home directory. The full path resolves to `/home/username/secure_data/decrypted/secret.txt`.

```
tiago-paquete@Linux:~$ echo "This is sensitive content" > ~/secure_data/decrypted/secret.txt
```

Check the encrypted version (don't open the decrypted path):
ls ~/secure_data/encrypted

```
tiago-paquete@Linux:~$ ls ~/secure_data/encrypted
=================================================================
gocryptfs.conf  gocryptfs.diriv  mdALriCyWQsosW13EvM5Dw
=================================================================
```

Output Explanation:

Item	Description
gocryptfs.conf	**Configuration file** – contains metadata and cryptographic configuration for the encrypted filesystem created with `gocryptfs`. It stores information like the encryption parameters (e.g., AES-GCM mode, scrypt parameters).
gocryptfs.diriv	**Directory IV (Initialization Vector)** – a per-directory initialization vector used by `gocryptfs` to randomize file name encryption. Each directory has its own `gocryptfs.diriv`to enhance security.
mdALriCyWQsosW13EvM5Dw	**Encrypted filename** – an example of an encrypted file or directory name. Since names are encrypted, this represents a file or folder whose original name is hidden. The content is also encrypted until decrypted via the mounted view.

6. Unmount the Decrypted View

fusermount -u ~/secure_data/decrypted

The contents are now secure and inaccessible without remounting and re-authenticating.

Command: fusermount -u ~/secure_data/decrypted

Component	Description
sudo	**Superuser Do** – runs the command that follows with elevated (root) privileges. Required when the command needs administrative access. *(Not present in this command but kept for format consistency.)*
fusermount	Command used to mount and unmount filesystems that are managed by FUSE (Filesystem in Userspace). It allows non-root users to manage FUSE mounts.
-u	Option that tells `fusermount` to unmount the specified FUSE-mounted directory.
~/secure_data/ decrypted	Path to the mounted directory to be unmounted. The ~ represents the current user's home directory.

```
tiago-paquete@Linux:~$ fusermount -u ~/secure_data/decrypted
```

Challenge Variation:

Mount your encrypted gocryptfs folder inside your LVM volume:

mkdir -p /backup/databk/secure_encrypted /backup/databk/secure_decrypted
gocryptfs -init /backup/databk/secure_encrypted
gocryptfs /backup/databk/secure_encrypted /backup/databk/secure_decrypted

Command: mkdir -p /backup/databk/secure_encrypted /backup/databk/secure_decrypted

Component	Description
mkdir	**Make Directory** – command used to create one or more directories.
-p	**Parent** – creates parent directories as needed; avoids error if the directories already exist.
/backup/databk/ secure_encrypted	**Directory Path** – the full path of the directory to be created for storing encrypted files.
/backup/databk/ secure_decrypted	**Directory Path** – the full path of the directory to be created for accessing decrypted content.

Command: gocryptfs -init /backup/databk/secure_encrypted

Component	Description
gocryptfs	**gocryptfs** – a user-space encrypted filesystem based on FUSE.
-init	**Initialize** – sets up a new encrypted filesystem at the specified path; creates a config file and key.
/backup/databk/ secure_encrypted	**Encrypted Directory** – the target path where the encrypted filesystem will be initialized.

Command: gocryptfs /backup/databk/secure_encrypted /backup/databk/secure_decrypted

Component	Description
gocryptfs	**gocryptfs** – runs the encrypted filesystem and mounts it.
/backup/databk/ secure_encrypted	**Encrypted Directory** – the source directory that holds the encrypted data.
/backup/databk/ secure_decrypted	**Mount Point** – the target directory where the decrypted view of the encrypted files is mounted.

Summary of What You Practice:

FUSE-based filesystem mounting/unmounting.
Secure encryption and access control.
Integration with LVM volumes.
User-space isolation with modern cryptographic security.