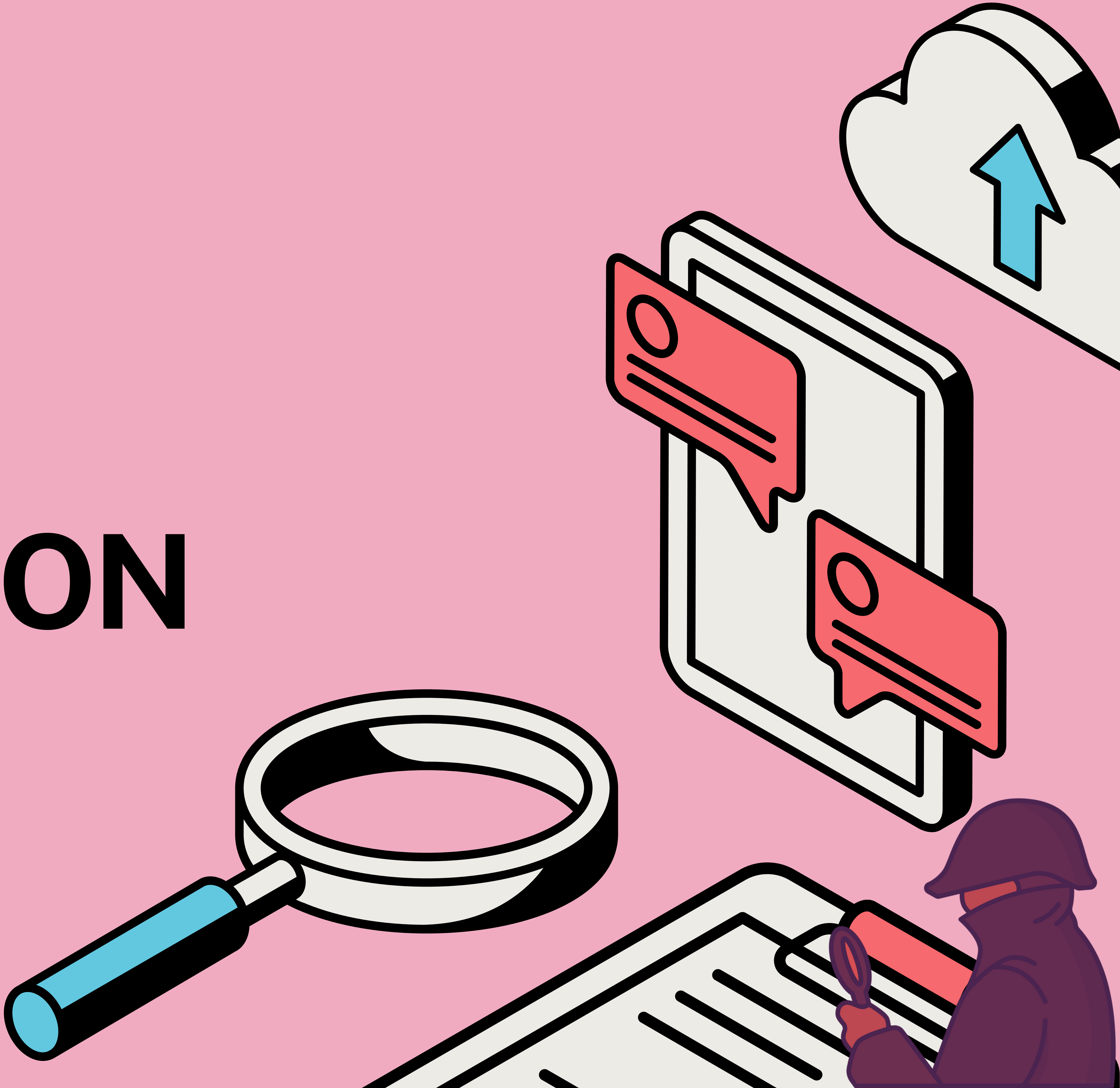
TEAM

INSE 6130 PROJECT PRESENTATION



Date: April 12, 2023

In this document:

1. Introduction of the project
2. General Attack Flow and Attacks Performed
    a. General Attack Flow
    b. Attacks Performed
3. General Defense Introduction
    a. General idea of defense deployed and algorithm
4. Solutions provided against the attacks
5. Program Source Code
6. Conclusion
7. Questions?

1.

Introduction of the project

A roleplay...

Breaking News!

Company,inc - Company that helps manage student grades, has been attacked by suspected former employees. The company is in distress and is shut for resolution and maintenance.

2.

General Attack Flow and Attacks Performed

2.a.
General Attack Flow

Four different types of attacks were performed on the company.

- The idea was mainly to gain root access to the company
- and steal data

Attacks Performed

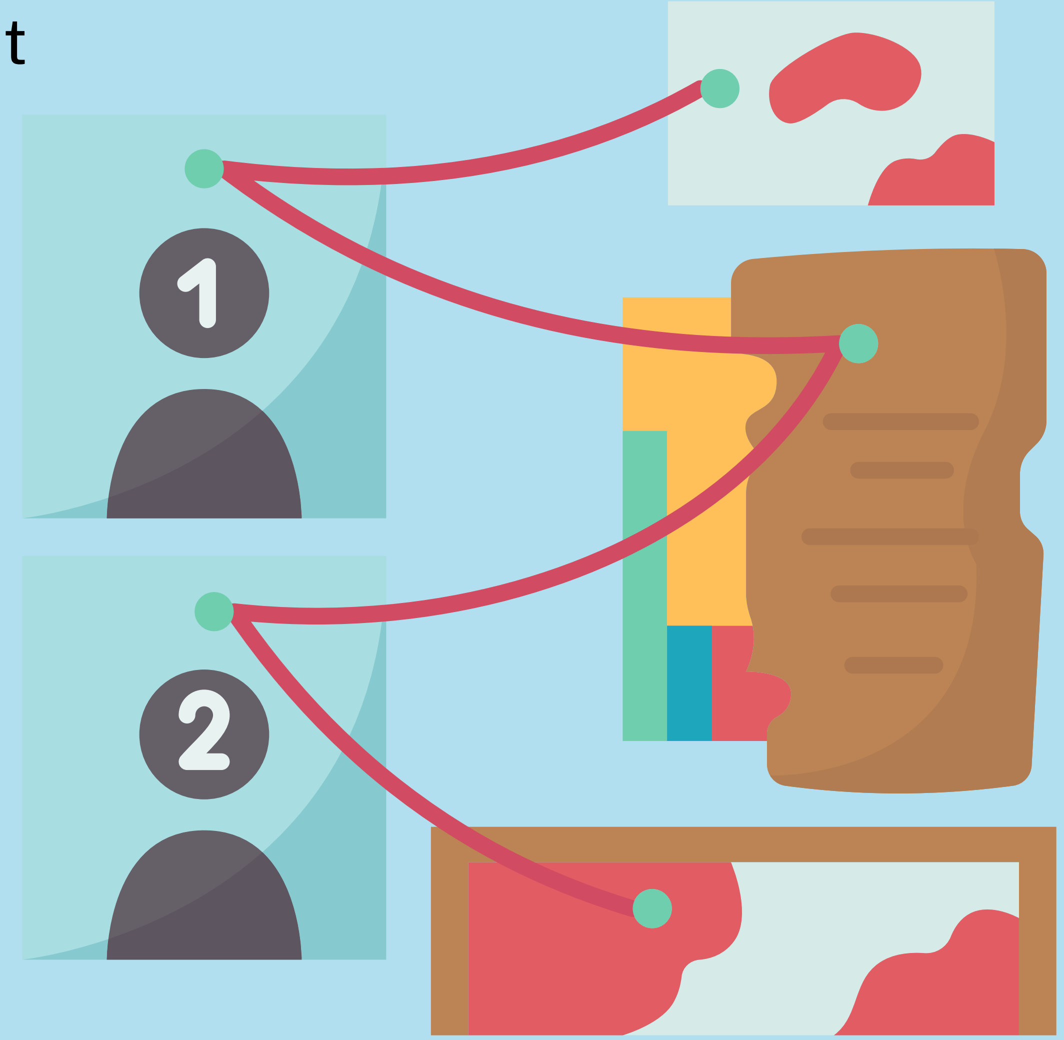
1. RunC Attack
2. Priviledge Escalation using docker sockets
3. Priviledge Escalation using volume mounts
4. Priviledge Escalation using docker groups
5. Abusing Exposed Docker Registry

3.

General Defense Introduction

3.a.
General idea of defense deployed and algorithm

Proposed a costly solution, which was rejected by the company.inc due to budget
Some of the most generic fixes were checked

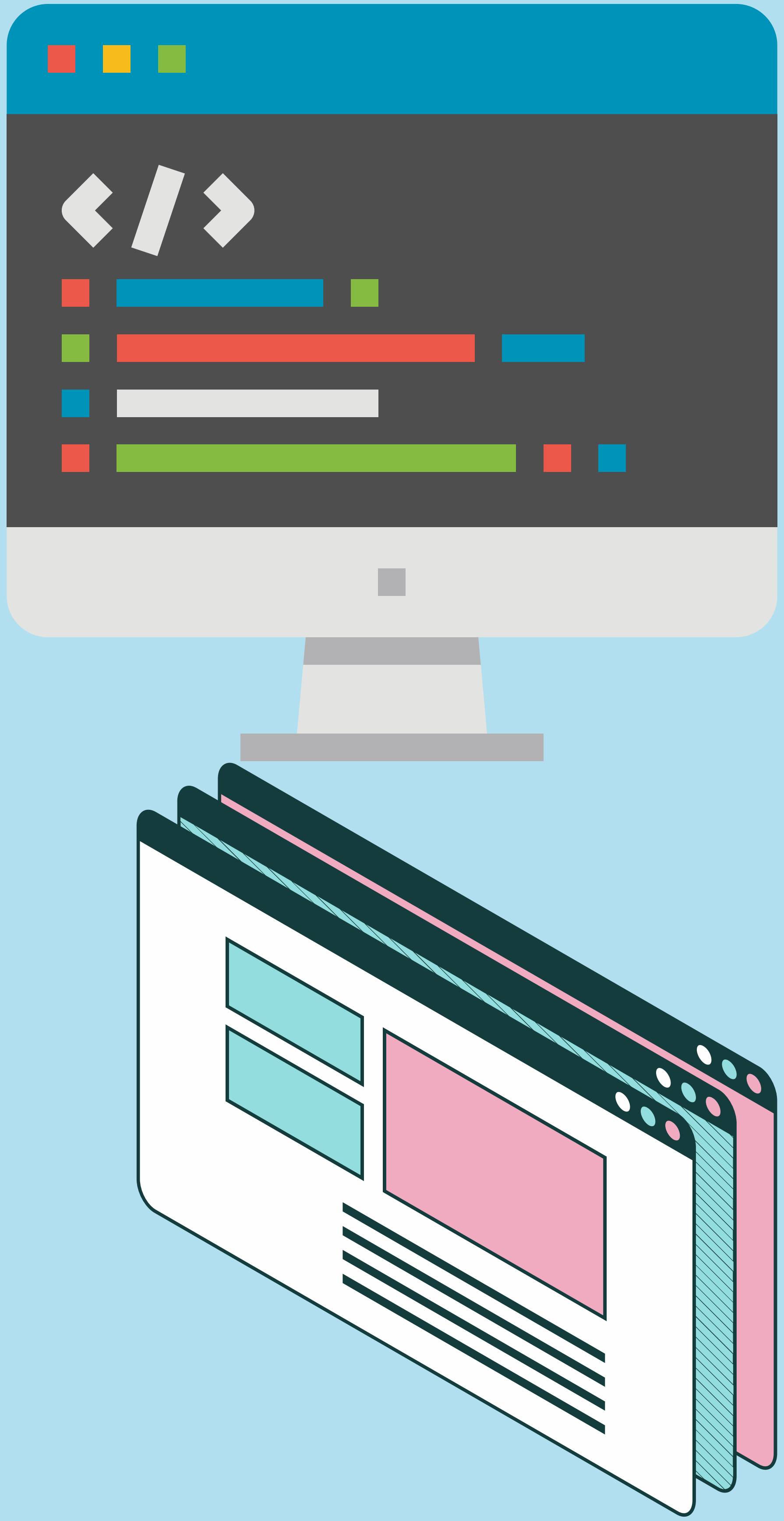


Common Solutions, which is too straightforward

- Update docker
- Check CVEs regularly
- Use Official Images
- Limit Container Capabilities
- Use Trusted Repositories

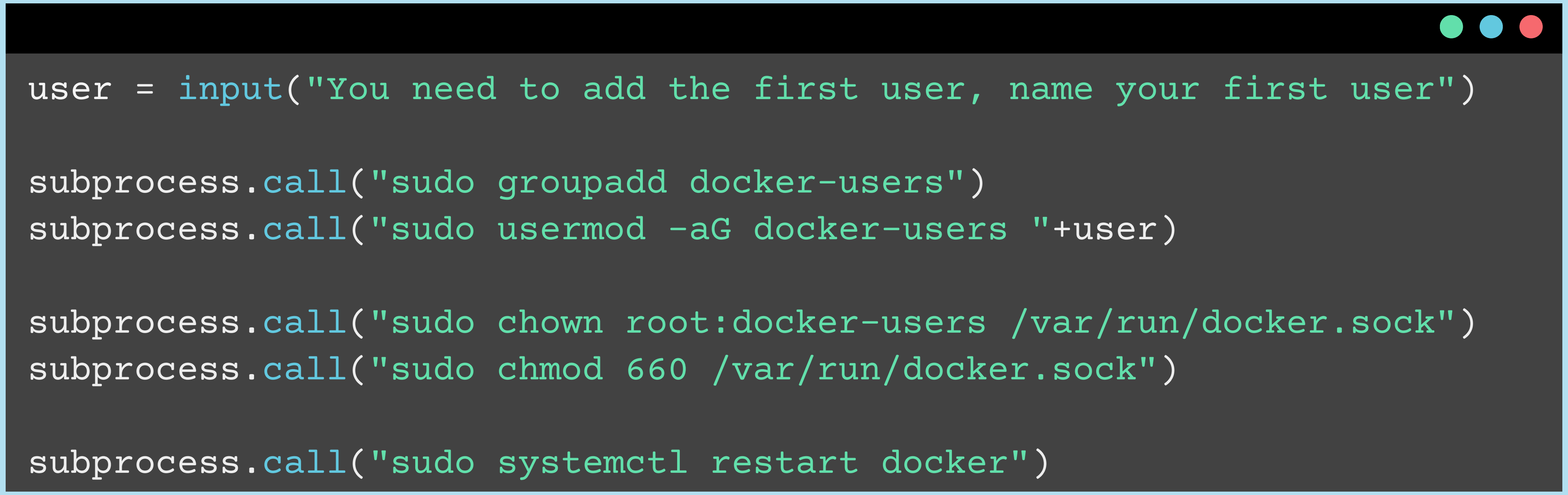
4.
Solutions provided against the attacks

- **Fix for RunC attack** - make login groups
- **To fix Priviledge Escalation attacks** - make login groups, only assign certain users to have specific accesses - Separation of Priviledges.
- **Fixing Abusing Docker Registry Attack** - Implement Authentication to containers.
- **Fix against any attacks which abuse docker cp** - Run docker cp when container is closed.

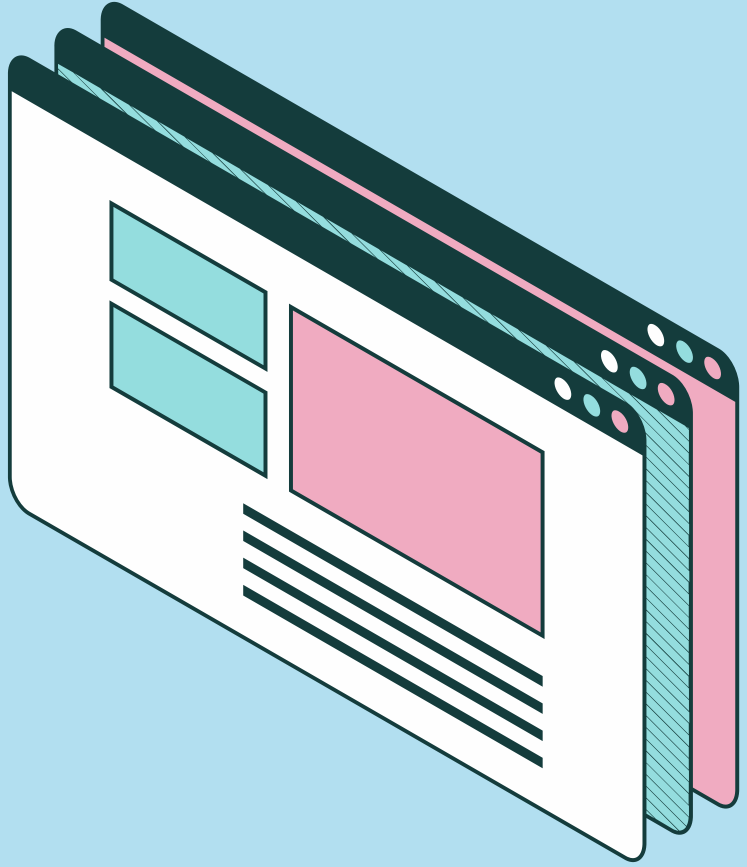


4.
Program Source Code

Patch files to make login groups - two files
Part - 1: Initiation

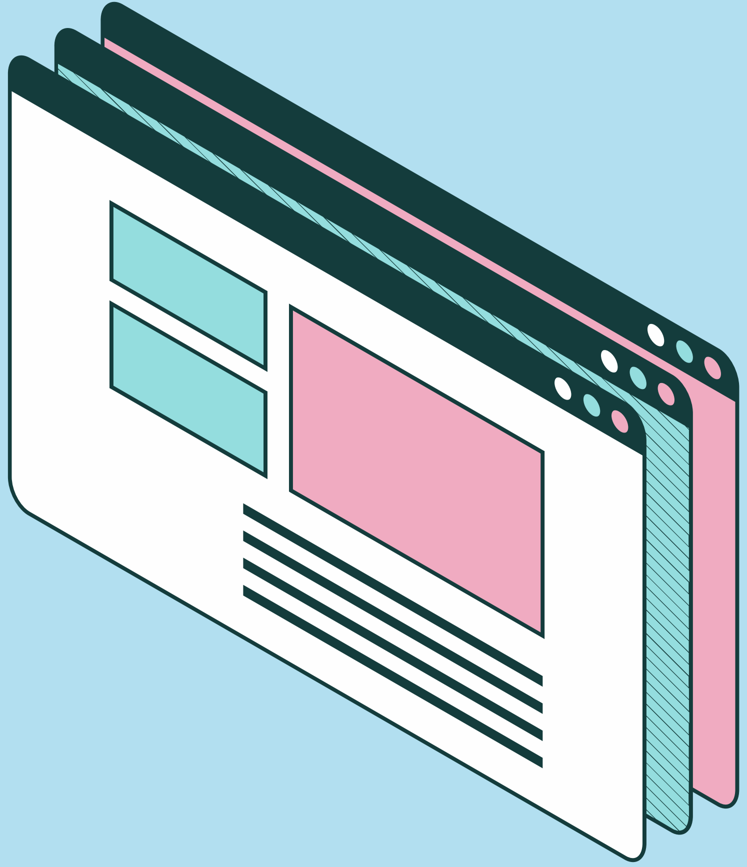


```
user = input("You need to add the first user, name your first user")

subprocess.call("sudo groupadd docker-users")
subprocess.call("sudo usermod -aG docker-users "+user)

subprocess.call("sudo chown root:docker-users /var/run/docker.sock")
subprocess.call("sudo chmod 660 /var/run/docker.sock")

subprocess.call("sudo systemctl restart docker")
```

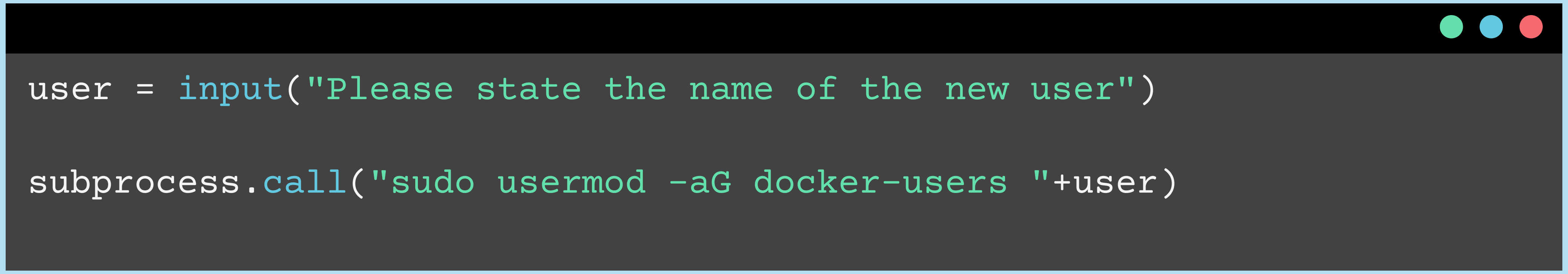


4.

Program Source Code



Patch files to make login groups - two files
Part - 1: Add new users



```
user = input("Please state the name of the new user")

subprocess.call("sudo usermod -aG docker-users "+user)
```

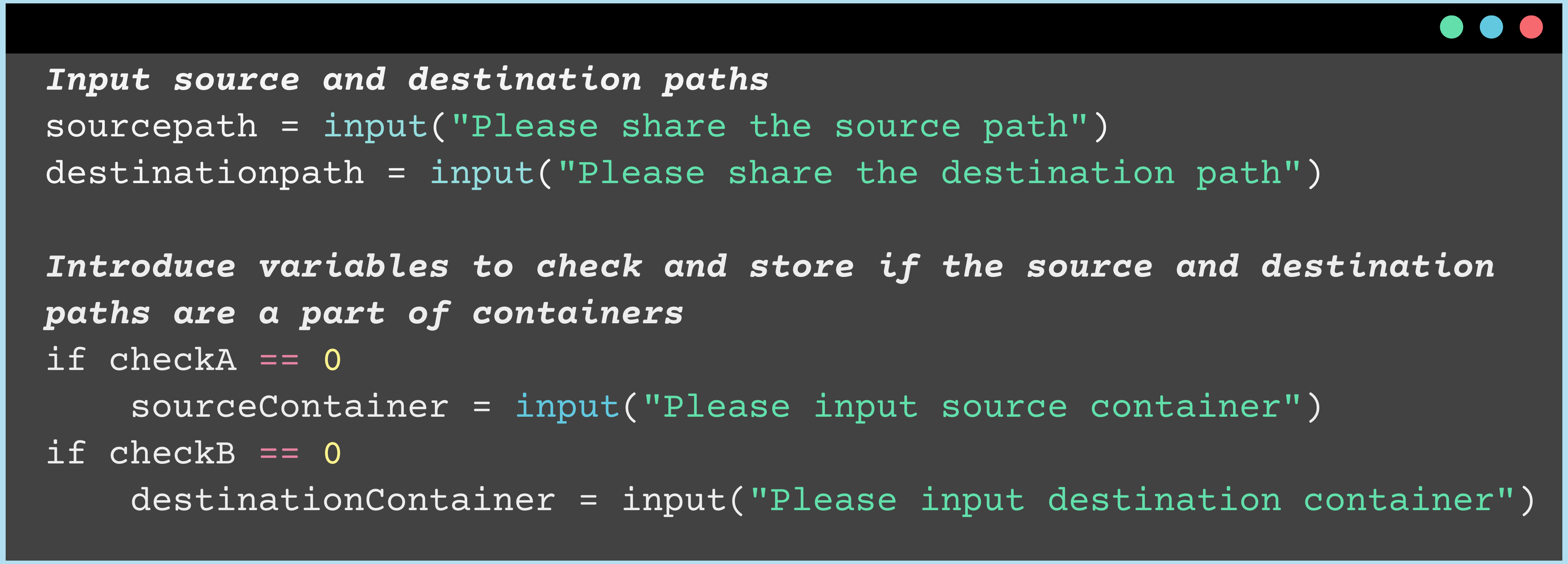
Design principle used:

- Fail-Safe Defaults
- Separation of Priviledges

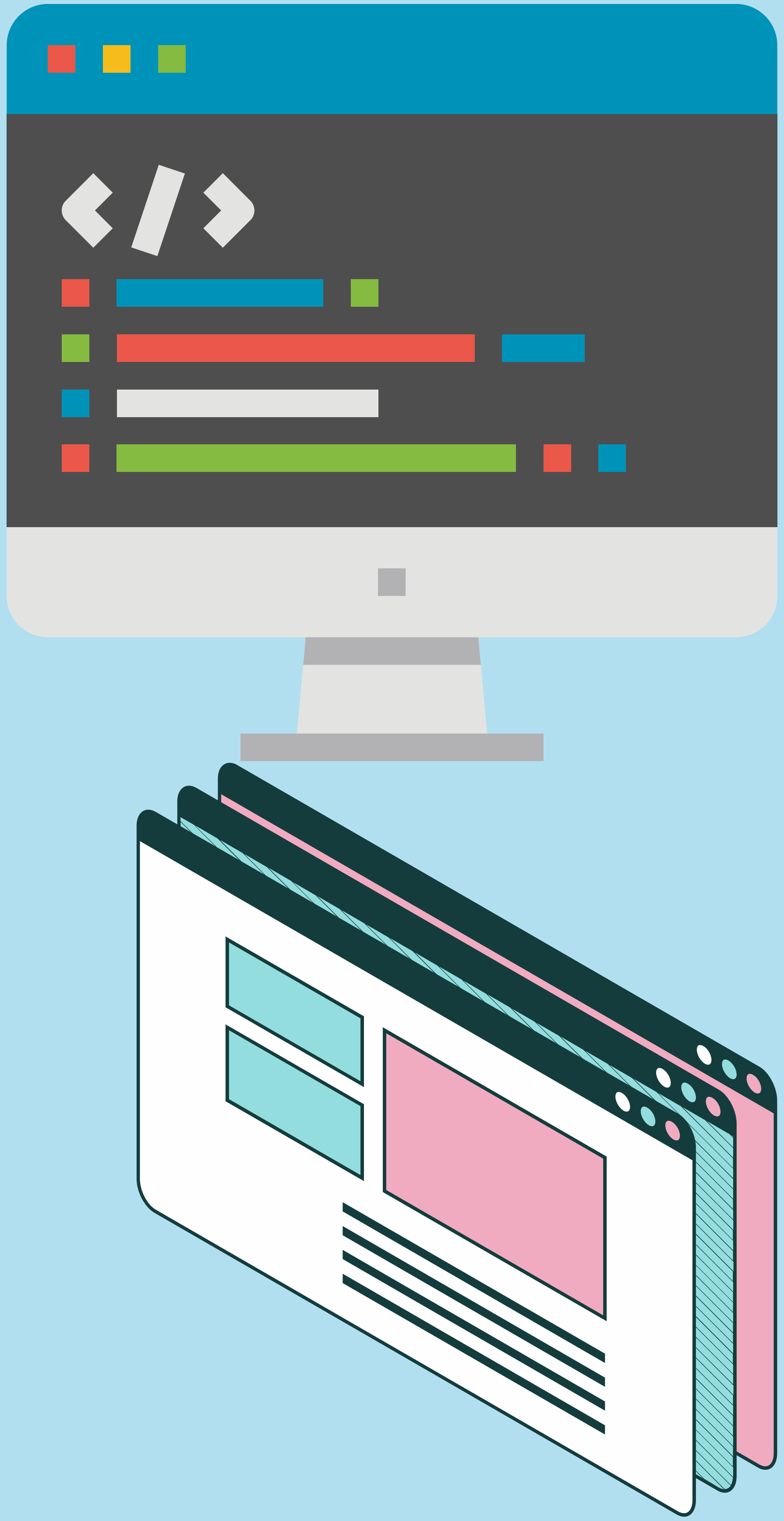
Fixes RunC and Priviledge Escalation Attacks

4.
Solutions provided against the attacks

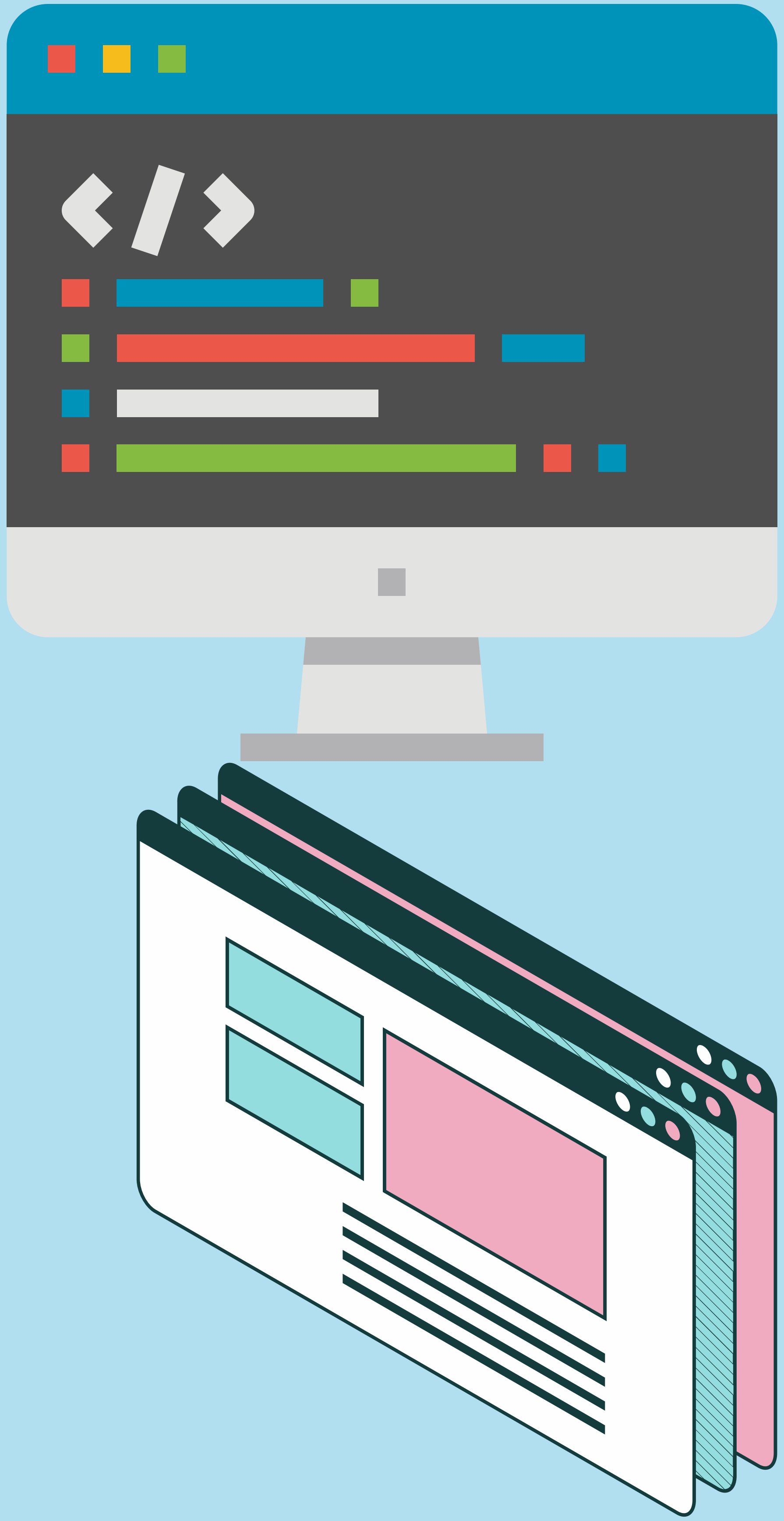
Patch Files to fix abuse of docker cp



```
Input source and destination paths
sourcepath = input("Please share the source path")
destinationpath = input("Please share the destination path")

Introduce variables to check and store if the source and destination
paths are a part of containers
if checkA == 0
    sourceContainer = input("Please input source container")
if checkB == 0
    destinationContainer = input("Please input destination container")
```



4.
Program Source Code

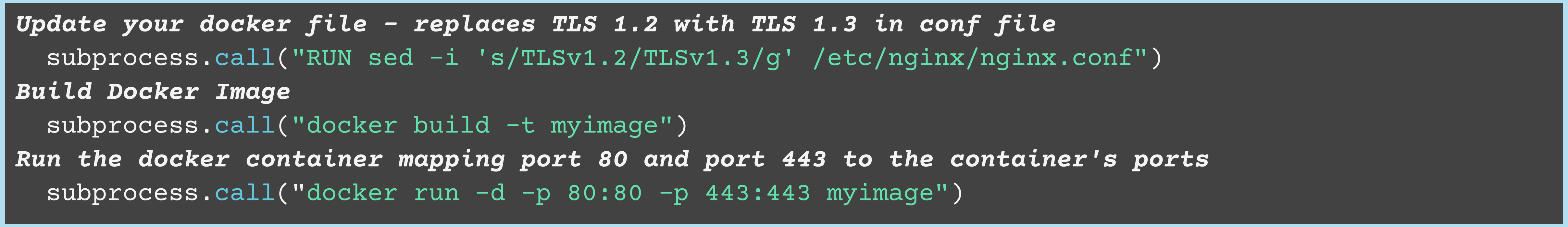


Fix against Abuse of Docker Registry

- Enable Authentication on the containers - you could use plugins such as MySQL or PostgreSQL
- Every time the users use a different container, they will be posed with login page.

Fixes against possible attacks related to abuse of docker sockets

- Enable TLS 1.3



```
Update your docker file - replaces TLS 1.2 with TLS 1.3 in conf file
  subprocess.call("RUN sed -i 's/TLSv1.2/TLSv1.3/g' /etc/nginx/nginx.conf")
Build Docker Image
  subprocess.call("docker build -t myimage")
Run the docker container mapping port 80 and port 443 to the container's ports
  subprocess.call("docker run -d -p 80:80 -p 443:443 myimage")
```

5.
Conclusion

Following is the conclusion of the project

1 The attacks were performed in isolation.

2 The problem statement (Attacks) were clearly identified and understood.

3 The attacks were documented and studied before fix.

4 CVEs were checked for more details on the attacks.

5 Solutions were proposed, discussed and debated upon.

6 Algorithms of the fix were designed and patch files were made

7 Patch files were deployed and tested

Questions?

Thank you.