**For secure connection, we are going to make 2 keys:**

1. Private key: stay on personal computer (to identify the computer we are using right now)
2. Public key: this is going to be uploaded to the server

**How is it going to work?**

Whenever we try to connect, the system will look to those keys and see whether they matches or not. (Actually, they are not the same but will do some calculations to see if they match or not) to be able to connect.

**How to generate these keys:**

ssh-keygen -t rsa -b 4096

-t rsa: specify the RSA Algorithm

-b 4096: Sets the key size to 4096 bits for stronger security.

**After generating the key, it will ask where to save the key 🡪 Don’t change anything stay at the default file.**

**Then it will ask to set passphrase (Optional) to protect it using password**

**Once you are done, the private and public keys will be generated and saved in their corresponding files**

**To copy the public key to the server you have to go to the file that contain the public key in the user, copy the key, and paste in a file called authorized\_keys that exist in the server user**

**To connect to the server we use:**

Ssh user@server\_ip

* No need to write a password because we have the private key in our computer, it automatically read the key whenever we want to connect
* **To know the server ip:**
* hostname -I

**To disable password login:**

1. Connect to the server using: ssh user@server\_ip
2. Enter the ssh file using text editor: nano /etc/ssh/sshd\_config
3. Set PasswordAuthentication no
4. Set PubkeyAuthentication yes
5. Uncomment step (3,4) in the sshd\_config file
6. Save changes
7. Reload the ssh using the command: sudo systemctl restart sshd

<https://chatgpt.com/share/67b4f0e2-3938-8010-8206-cb01fb4d817d>

**How to enable automatic security updates :**

1. Log in to the server
2. Install the unattended upgrade package using the command:

Sudo apt-get install unattended-upgrades -y

1. Install update dash notifier common package (for automatic reboots) using the command:

Sudo apt-get install update-notifier-common -y

1. Edit the 50 unattended dash upgrade file using the command:

sudo nano /etc/apt/apt.conf.d/50unattended-upgrades

1. Remove the // and change the false to true from the line that contains:

Unattended-Upgrade::Automatic-Reboot “false”;

1. Add the following line to log updates:

Unattended-Upgrade::LogFile "/var/log/security\_updates.log";

Issue the command Cat /var/run/reboot-required to see if a reboot is required, if yes, reboot the machine and that’s it

**How to configure MOTD Message:**

1. Open the motd file using the command:

Sudo nano /etc/motd

1. Add the custom message “Welcome to the Ubuntu administration Lab”

**Shell Script:**

#!/bin/bash

# Log file to monitor

AUTH\_LOG="/var/log/auth.log" # Use /var/log/secure for CentOS/RHEL

# User to monitor

MONITORED\_USER="dev\_lead1"

# File to store blocked IPs

BLOCKED\_IPS\_FILE="/tmp/blocked\_ips.txt"

# File to track lock status of user

LOCKED\_USER\_FILE="/tmp/locked\_users.txt"

# Time to unblock IPs (24 hours)

UNBLOCK\_TIME=86400 # 24 hours in seconds

# Ensure necessary files exist

touch "$BLOCKED\_IPS\_FILE" "$LOCKED\_USER\_FILE"

# Function to unblock IPs after 24 hours

unblock\_old\_ips() {

local temp\_file="/tmp/temp\_ips.txt"

> "$temp\_file" # Create a temporary file

while read -r line; do

ip=$(echo "$line" | awk '{print $1}')

timestamp=$(echo "$line" | awk '{print $2}')

current\_time=$(date +%s)

if (( current\_time - timestamp >= UNBLOCK\_TIME )); then

echo "Unblocking IP $ip..."

iptables -D INPUT -s "$ip" -j DROP

else

echo "$ip $timestamp" >> "$temp\_file"

fi

done < "$BLOCKED\_IPS\_FILE"

mv "$temp\_file" "$BLOCKED\_IPS\_FILE"

}

# Function to block an IP

block\_ip() {

local ip=$1

local current\_time=$(date +%s)

if grep -q "$ip" "$BLOCKED\_IPS\_FILE"; then

echo "IP $ip is already blocked."

return

fi

echo "Blocking IP $ip..."

iptables -A INPUT -s "$ip" -j DROP

echo "$ip $current\_time" >> "$BLOCKED\_IPS\_FILE"

}

# Function to lock the user account

lock\_user() {

if grep -q "$MONITORED\_USER" "$LOCKED\_USER\_FILE"; then

echo "User $MONITORED\_USER is already locked."

return

fi

echo "Locking user $MONITORED\_USER due to repeated failed attempts..."

sudo usermod -L "$MONITORED\_USER"

echo "$MONITORED\_USER" >> "$LOCKED\_USER\_FILE"

}

# Function to monitor failed login attempts

monitor\_logins() {

tail -Fn0 "$AUTH\_LOG" | while read -r line; do

if echo "$line" | grep -q "Failed password for $MONITORED\_USER"; then

ip=$(echo "$line" | grep -oE "([0-9]{1,3}\.){3}[0-9]{1,3}")

if [ -n "$ip" ]; then

block\_ip "$ip"

# Count failed attempts in the last 10 lines

failed\_attempts=$(grep -E "Failed password.\*$MONITORED\_USER" "$AUTH\_LOG" | tail -n 10 | wc -l)

if (( failed\_attempts >= 5 )); then

lock\_user

fi

else

echo "Could not extract IP from log: $line"

fi

fi

done

}

# Run the functions

unblock\_old\_ips

monitor\_logins