Implement Access Control (AppArmor)

Verification and configuration of Mandatory Access Control (MAC) using AppArmor to restrict program capabilities and enforce the principle of least privilege.

Verification & Reporting Commands:

1. Create the reporting script

```
nano apparmor-report.sh
```

2. Insert script

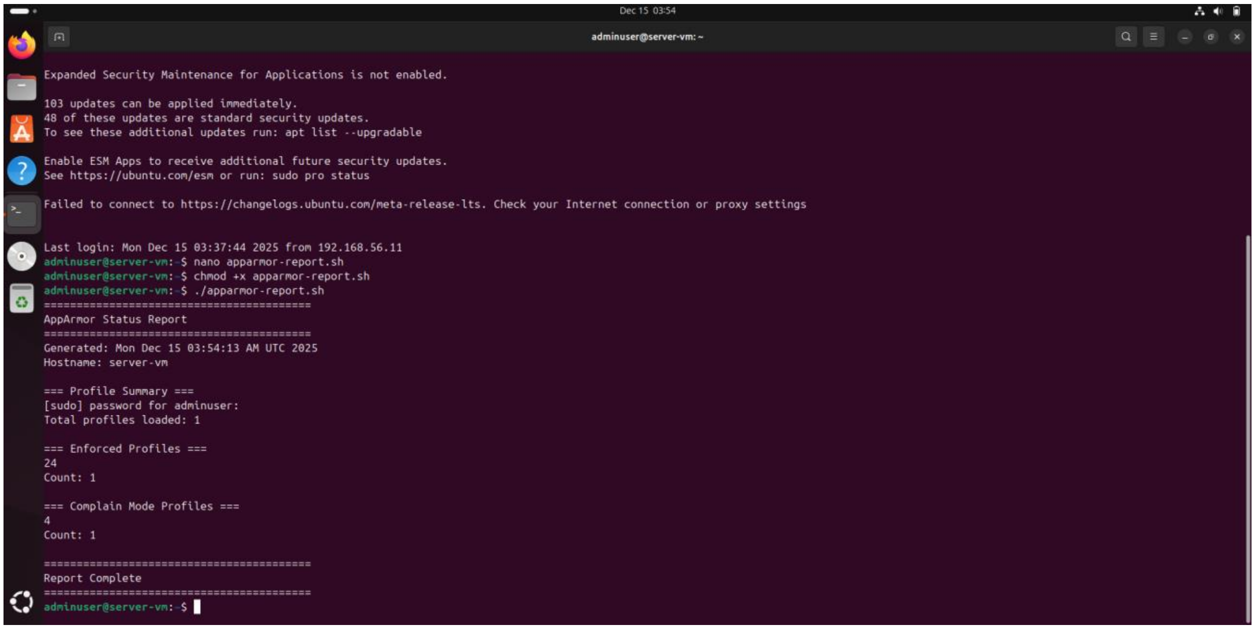
```
#!/bin/bash
# AppArmor Status Report Script
# Reports on all AppArmor profiles and their status

echo "========================================="
echo "AppArmor Status Report"
echo "========================================="
echo "Generated: $(date)"
echo "Hostname: $(hostname)"
echo ""

# Check if AppArmor is installed
if ! command -v aa-status &> /dev/null; then
   echo "ERROR: AppArmor is not installed"
   exit 1
fi

echo "=== Profile Summary ==="
# Count total profiles loaded
total_profiles=$(sudo aa-status --profiled | wc -l)
echo "Total profiles loaded: $total_profiles"
echo ""

echo "=== Enforced Profiles ==="
sudo aa-status --enforced
enforced_count=$(sudo aa-status --enforced | wc -l)
echo "Count: $enforced_count"
echo ""
echo "=== Complain Mode Profiles ==="
sudo aa-status --complaining

complain_count=$(sudo aa-status --complaining | wc -l)
echo "Count: $complain_count"
echo ""

echo "========================================"
echo "Report Complete"
echo "========================================"
```

3. Make executable

```
chmod +x apparmor-report.sh
```

4. Run script

```
./apparmor-report.sh
```



```
Expanded Security Maintenance for Applications is not enabled.

103 updates can be applied immediately.
48 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings


Last login: Mon Dec 15 03:37:44 2025 from 192.168.56.11
adminuser@server-vm:~$ nano apparmor-report.sh
adminuser@server-vm:~$ chmod +x apparmor-report.sh
adminuser@server-vm:~$ ./apparmor-report.sh
========================================
AppArmor Status Report
========================================
Generated: Mon Dec 15 03:54:13 AM UTC 2025
Hostname: server-vm

=== Profile Summary ===
[sudo] password for adminuser:
Total profiles loaded: 1

=== Enforced Profiles ===
24
Count: 1

=== Complain Mode Profiles ===
4
Count: 1

========================================
Report Complete
========================================
adminuser@server-vm:~$ 
```

Configure automatic security updates

Implementation of unattended upgrades to ensure the server automatically applies critical security patches, minimizing the vulnerability window.

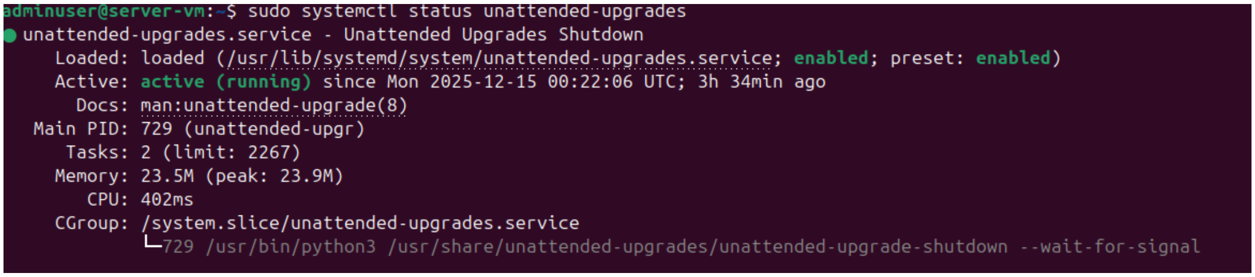
Configuration Commands:

1. Install the package
sudo apt update && sudo apt install unattended-upgrades

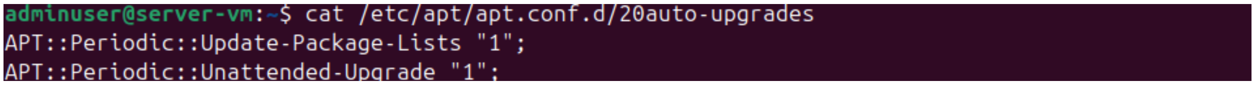
2. Configure the service
sudo dpkg-reconfigure -plow unattended-upgrades

3. Verify the service is active
sudo systemctl status unattended-upgrades

4. Verify configuration file creation
cat /etc/apt/apt.conf.d/20auto-upgrades



```
adminuser@server-vm:~$ sudo systemctl status unattended-upgrades
● unattended-upgrades.service - Unattended Upgrades Shutdown
     Loaded: loaded (/usr/lib/systemd/system/unattended-upgrades.service; enabled; preset: enabled)
     Active: active (running) since Mon 2025-12-15 00:22:06 UTC; 3h 34min ago
       Docs: man:unattended-upgrade(8)
   Main PID: 729 (unattended-upgr)
      Tasks: 2 (limit: 2267)
     Memory: 23.5M (peak: 23.9M)
        CPU: 402ms
     CGroup: /system.slice/unattended-upgrades.service
             └─729 /usr/bin/python3 /usr/share/unattended-upgrades/unattended-upgrade-shutdown --wait-for-signal
```



```
adminuser@server-vm:~$ cat /etc/apt/apt.conf.d/20auto-upgrades
APT::Periodic::Update-Package-Lists "1";
APT::Periodic::Unattended-Upgrade "1";
```

Configure fail2ban for enhanced intrusion detection

Installation and configuration of Fail2Ban to protect the SSH service against brute-force attacks by banning IP addresses after repeated failed login attempts.

Configuration Commands:

1. Install Fail2Ban

```
sudo apt install fail2ban
```

2. Create local configuration (preserve default jail.conf)

```
sudo cp /etc/fail2ban/jail.conf /etc/fail2ban/jail.local
```

3. Configure SSH Jail

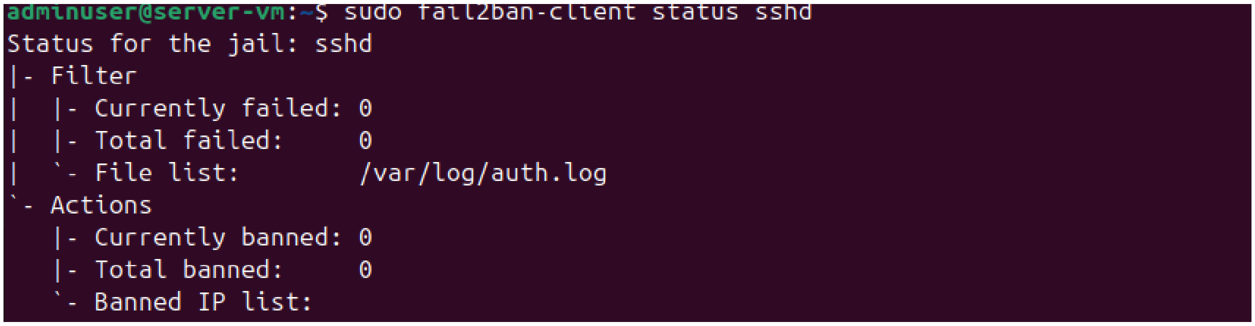
```
sudo nano /etc/fail2ban/jail.local
```

4. Change to

```
[sshd]
enabled = true
port = 22
filter = sshd
logpath = /var/log/auth.log
maxretry = 3
bantime = 600
findtime = 600
```

5. Restart and Verify

```
sudo systemctl restart fail2ban
sudo fail2ban-client status ssh
```



```
adminuser@server-vm:~$ sudo fail2ban-client status sshd
Status for the jail: sshd
|- Filter
|  |- Currently failed: 0
|  |- Total failed:     0
|  `- File list:        /var/log/auth.log
`- Actions
   |- Currently banned: 0
   |- Total banned:     0
   `- Banned IP list:
```

Security Baseline Verification Script (security-baseline.sh)

A script deployed on the Server to automatically verify that the security controls implemented, and they are active and correctly configured.

1. Edit script

```
nano ./security-baseline.sh
```

2. Add

Script Content:

```
#!/bin/bash
# Security Baseline Verification Script
# Verifies all security configurations from Phase 5

echo "========================================"
echo "Security Baseline Verification Report"
echo "========================================"
echo "Generated: $(date)"
echo "Hostname: $(hostname)"
echo ""

# Colour codes for output
RED='\033[0;31m'
GREEN='\033[0;32m'
YELLOW='\033[1;33m'
NC='\033[0m' # No Colour

# 1. Check SSH configuration
echo "=== SSH Security Configuration ==="
```

```
# Check password authentication
echo -n "Password Authentication: "
if grep -q "^PasswordAuthentication no" /etc/ssh/sshd_config.d/*.conf; then
    echo -e "${GREEN}DISABLED${NC} (Secure)"
else
    echo -e "${RED}ENABLED${NC} (Warning: Should be disabled)"
fi

# Check root login
echo -n "Root Login via SSH: "
if grep -q "^PermitRootLogin no" /etc/ssh/sshd_config.d/*.conf; then
    echo -e "${GREEN}DISABLED${NC} (Secure)"
else
    echo -e "${RED}ENABLED${NC} (Warning: Should be disabled)"
fi

# Check public key authentication
echo -n "Public Key Authentication: "
if grep -q "^PubkeyAuthentication yes" /etc/ssh/sshd_config.d/*.conf; then
    echo -e "${GREEN}ENABLED${NC} (Secure)"
else
    echo -e "${YELLOW}DISABLED${NC} (Warning: Should be enabled)"
fi
echo ""

# 2. Check Firewall Configuration
echo "=== Firewall Configuration ==="
```

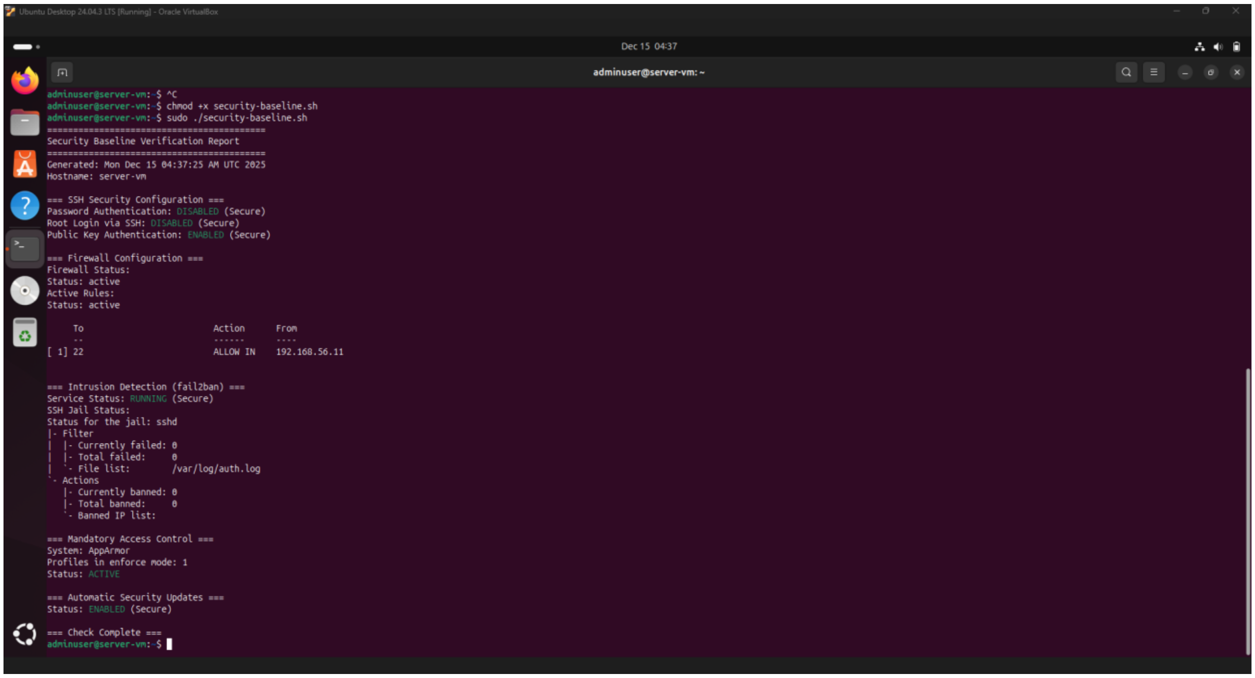
```
if command -v ufw &> /dev/null; then
    echo "Firewall Status:"
    sudo ufw status | grep "Status"
    echo "Active Rules:"
    sudo ufw status numbered
else
    echo -e "${RED}UFW not installed${NC}"
fi
echo ""

# 3. Check Intrusion Detection (fail2ban)
echo "=== Intrusion Detection (fail2ban) ==="
if systemctl is-active --quiet fail2ban; then
    echo -e "Service Status: ${GREEN}RUNNING${NC} (Secure)"
    echo "SSH Jail Status:"
    sudo fail2ban-client status sshd 2>/dev/null || echo "SSH jail not configured"
else
    echo -e "Service Status: ${RED}NOT RUNNING${NC} (Warning)"
fi
echo ""

# 4. Check Mandatory Access Control (AppArmor)
echo "=== Mandatory Access Control ==="
if command -v aa-status &> /dev/null; then
    echo "System: AppArmor"
    enforced=$(sudo aa-status --enforced 2>/dev/null | wc -l)
    echo "Profiles in enforce mode: $enforced"
    if [ "$enforced" -gt 0 ]; then
        echo -e "Status: ${GREEN}ACTIVE${NC}"
    else
        echo -e "Status: ${YELLOW}INSTALLED BUT NO ENFORCED PROFILES${NC}"
    fi
else
    echo -e "${RED}AppArmor not installed${NC}"
fi
echo ""

# 5. Check Automatic Updates
echo "=== Automatic Security Updates ==="
if systemctl is-enabled unattended-upgrades &> /dev/null; then
    echo -e "Status: ${GREEN}ENABLED${NC} (Secure)"
else
    echo -e "Status: ${YELLOW}DISABLED${NC} (Warning)"
fi
echo ""

echo "=== Check Complete ==="
```

3. Make executable

```
chmod +x security-baseline.sh
```

4. Run script

```
./security-baseline.sh
```



```
adminuser@server-vm:~$ ^C
adminuser@server-vm:~$ chmod +x security-baseline.sh
adminuser@server-vm:~$ sudo ./security-baseline.sh
==========================================
Security Baseline Verification Report
==========================================
Generated: Mon Dec 15 04:37:25 AM UTC 2025
Hostname: server-vm

=== SSH Security Configuration ===
Password Authentication: DISABLED (Secure)
Root Login via SSH: DISABLED (Secure)
Public Key Authentication: ENABLED (Secure)

=== Firewall Configuration ===
Firewall Status:
Status: active
Active Rules:
Status: active

     To                         Action      From
     --                         ------      ----
[ 1] 22                         ALLOW IN    192.168.56.11


=== Intrusion Detection (fail2ban) ===
Service Status: RUNNING (Secure)
SSH Jail Status:
Status for the jail: sshd
|- Filter
|  |- Currently failed: 0
|  |- Total failed:     0
|  `- File list:        /var/log/auth.log
`- Actions
   |- Currently banned: 0
   |- Total banned:     0
   `- Banned IP list:

=== Mandatory Access Control ===
System: AppArmor
Profiles in enforce mode: 1
Status: ACTIVE

=== Automatic Security Updates ===
Status: ENABLED (Secure)

=== Check Complete ===
adminuser@server-vm:~$ 
```

Remote Monitoring Script

A script deployed on the Workstation that connects to the server via SSH to collect performance metrics (CPU, Memory, Disk) without requiring an interactive session.

1. Edit script

```
nano ./monitor-server.sh
```

2. Make executable

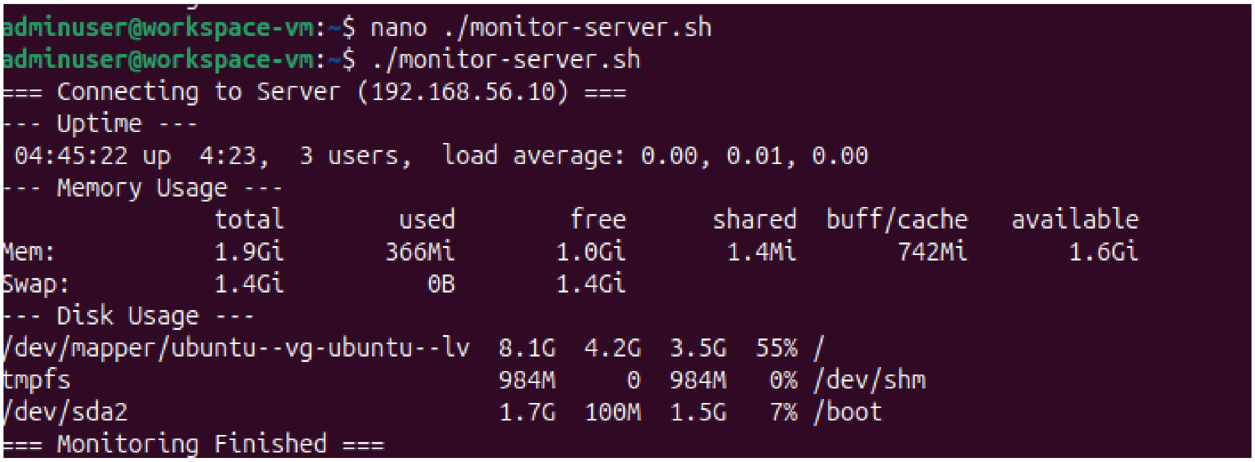
```
chmod +x monitor-server.sh
```

3. Execution Command (Workstation):

```
./monitor-server.sh
```

Script Content:

```
#!/bin/bash
# Remote Server Monitoring Script
# Runs on Workstation, connects via SSH

# Define Server IP and User
SERVER_IP="192.168.56.10"
USER="adminuser"

echo "=== Connecting to Server ($SERVER_IP) ==="

# 1. Collect System Uptime
echo "--- Uptime ---"
ssh $USER@$SERVER_IP "uptime"

# 2. Collect Memory Usage (Human Readable)
echo "--- Memory Usage ---"
ssh $USER@$SERVER_IP "free -h"

# 3. Collect Disk Usage (Physical drives only)
echo "--- Disk Usage ---"
ssh $USER@$SERVER_IP "df -h | grep '/dev/'"

echo "=== Monitoring Finished ==="
```



```
adminuser@workspace-vm:~$ nano ./monitor-server.sh
adminuser@workspace-vm:~$ ./monitor-server.sh
=== Connecting to Server (192.168.56.10) ===
--- Uptime ---
 04:45:22 up  4:23,  3 users,  load average: 0.00, 0.01, 0.00
--- Memory Usage ---
               total        used        free      shared  buff/cache   available
Mem:           1.9Gi       366Mi       1.0Gi       1.4Mi       742Mi       1.6Gi
Swap:          1.4Gi          0B       1.4Gi
--- Disk Usage ---
/dev/mapper/ubuntu--vg-ubuntu--lv  8.1G  4.2G  3.5G  55% /
tmpfs                              984M     0  984M   0% /dev/shm
/dev/sda2                          1.7G  100M  1.5G   7% /boot
=== Monitoring Finished ===
```