**Title: Heritage Vault 2.0 Business Scenario Solution**

Name : Ashley Musihiwa

Student ID: 0159553

Email: [asm408@ensign.edu](mailto:asm408@ensign.edu)

IT 370: Advanced Linux

Instructor: Joseph Keene

Email: [joseph.keene@ensign.edu](mailto:joseph.keene@ensign.edu)

Department: Information Technology, Ensign College

Assignment Due Date: 1 February 2025

Table of Contents

[Introduction 3](#_Toc189429416)

[Proposed Security Measures 3](#_Toc189429417)

[1. File Auditing: Keeping an Eye on Critical Data 3](#_Toc189429418)

[2. User Access Control & Root Privilege Management 4](#_Toc189429419)

[3. Robust Backup Strategy 4](#_Toc189429420)

[4. Archival & Data Integrity with cpio & dd 5](#_Toc189429421)

[5. Fine-Tuned Access Control with ACLs & Special Permissions 5](#_Toc189429422)

[6. Securing Default File Permissions (umask Configuration) 6](#_Toc189429423)

[Ongoing Compliance & Monitoring 6](#_Toc189429424)

[1. Regular Log Audits 6](#_Toc189429425)

[2. Automated Compliance Checks 6](#_Toc189429426)

[Conclusion 7](#_Toc189429427)

[Next Steps: 7](#_Toc189429428)

[References 8](#_Toc189429429)

# Introduction

Protecting our genealogical data is a top priority. To ensure the **confidentiality, integrity, and availability** of our systems, we must implement robust security measures. This proposal outlines a structured approach to **Linux server security**, focusing on key areas such as **file auditing, user access control, root privilege management, backup strategies, access control lists (ACLs), and special permissions**. Additionally, we will integrate **automated scripts** to enforce these best practices efficiently.

## Proposed Security Measures

### 1. File Auditing: Keeping an Eye on Critical Data

Monitoring file access is essential for detecting unauthorized changes. By using **auditd and ausearch**, we can track who accesses or modifies sensitive files.

**Implementation:**

* Set up real-time auditing for critical directories.
* Generate alerts for unauthorized access attempts.
* Automate auditing with the following script:

*auditctl -w /var/log/heritage\_data/ -p wa -k heritage\_audit*

*ausearch -k heritage\_audit --start today*

**Benefit:** Ensures complete visibility into file access and modifications.

### 2. User Access Control & Root Privilege Management

Minimizing unnecessary privileges reduces security risks. We propose implementing **strict user access policies**:

**Key Strategies:**

* Enforce **least privilege access** via sudo configurations.
* Strengthen authentication using pam\_pwquality.
* Disable **direct root login** and enforce **SSH key authentication**.

**Configuration Update:** Modify */etc/ssh/sshd\_config* to enhance security:

*PermitRootLogin no*

*PasswordAuthentication no*

*PubkeyAuthentication yes*

**Benefit:** Prevents unauthorized root access and improves system security.

### 3. Robust Backup Strategy

Data loss can be catastrophic, making **automated backups** a necessity. We propose using **rsync and tar** for scheduled encrypted backups.

**Implementation:**

* Schedule periodic backups using cron.
* Store encrypted backups in secure locations.
* Automate the backup process:

*0 2 \* \* \* rsync -avz /heritage\_data/ /backup/heritage\_data/*

**Benefit:** Ensures data recovery in case of failure or cyber incidents.

### 4. Archival & Data Integrity with cpio & dd

Structured data archiving helps preserve **historical logs and configurations**.

🔹 **Implementation:**

* Use cpio for efficient archival.
* Utilize dd for full system cloning to remote backup servers.
* Automate archiving of log files:

*find /var/log -name "\*.log" | cpio -ov > /backup/log\_archive.cpio*

**Benefit:** Maintains an organized, decentralized and accessible archive of critical system logs.

### 5. Fine-Tuned Access Control with ACLs & Special Permissions

Enhancing file permissions ensures that only authorized personnel can access critical files.

**Key Strategies:**

* Implement **Access Control Lists (ACLs)** for granular permissions.
* Secure executables using **SUID, SGID, and Sticky Bit**.

**Example ACL Command:**

*setfacl -m u:backup\_admin:rwx /heritage\_data/*

**Benefit:** Adds an extra layer of security, limiting access to essential users only.

### 6. Securing Default File Permissions (umask Configuration)

By setting **default file permissions**, we prevent inadvertent data exposure.

🔹 **Implementation:** Modify /etc/profile to enforce secure defaults:

*umask 027*

**Benefit:** Ensures that newly created files have restrictive permissions by default.

## Ongoing Compliance & Monitoring

### 1. Regular Log Audits

* Conduct frequent reviews of system logs.
* Implement automated scanning for anomalies.

### 2. Automated Compliance Checks

* Utilize **Lynis** for periodic security assessments.
* Run system audits using:

*lynis audit system*

**Benefit:** Ensures that security policies remain enforced and up to date.

#### Alternatively : To ensure that the security script runs automatically on all hosts upon startup, follow these steps:

**1. Create a Unified Security Script**

Combine all security measures into a single Bash script (security\_init.sh). Example:

# Enable file auditing

*auditctl -w /var/log/heritage\_data/ -p wa -k heritage\_audit*

# Enforce SSH security settings

*sed -i 's/^PermitRootLogin.\*/PermitRootLogin no/' /etc/ssh/sshd\_config*

*sed -i 's/^PasswordAuthentication.\*/PasswordAuthentication no/' /etc/ssh/sshd\_config*

*sed -i 's/^PubkeyAuthentication.\*/PubkeyAuthentication yes/' /etc/ssh/sshd\_config*

*systemctl restart sshd*

# Set up automated backups

*echo "0 2 \* \* \* rsync -avz /heritage\_data/ /backup/heritage\_data/" | crontab -*

# Secure default file permissions

echo "umask 027" >> /etc/profile

# Set ACL permissions

*setfacl -m u:backup\_admin:rwx /heritage\_data/*

# Run security audit

*lynis audit system*

echo "Security initialization completed."

Save this script as /usr/local/bin/security\_init.sh and make it executable:

*chmod +x /usr/local/bin/security\_init.sh*

**2. Configure the Script to Run on Startup**

**Systemd Service (Recommended)**

Create a systemd service file:

*nano /etc/systemd/system/security\_init.service*

Add the following content:

*[Unit]*

*Description=Security Initialization Script*

*After=network.target*

*[Service]*

*ExecStart=/usr/local/bin/security\_init.sh*

*Type=oneshot*

*RemainAfterExit=yes*

*[Install]*

*WantedBy=multi-user.target*

*Enable and start the service:*

*systemctl enable security\_init.service*

*systemctl start security\_init.service*

**3. Deploy the Script Across All Hosts**

Use **Ansible** to automate deployment:

1. Create an Ansible playbook (deploy\_security.yml):

*- hosts: all*

*become: yes*

*tasks:*

*- name: Copy security script*

*copy:*

*src: security\_init.sh*

*dest: /usr/local/bin/security\_init.sh*

*mode: 0755*

*- name: Create systemd service*

*copy:*

*dest: /etc/systemd/system/security\_init.service*

*content: |*

*[Unit]*

*Description=Security Initialization Script*

*After=network.target*

*[Service]*

*ExecStart=/usr/local/bin/security\_init.sh*

*Type=oneshot*

*RemainAfterExit=yes*

*[Install]*

*WantedBy=multi-user.target*

*- name: Enable and start service*

*systemd:*

*name: security\_init*

*enabled: yes*

*state: started*

1. Run the playbook:

*ansible-playbook -i inventory deploy\_security.yml*

**Final Testing**

Reboot a host and check if the script runs:

*systemctl status security\_init.service*

This method ensures **automated, consistent security enforcement** across all Linux hosts upon startup.

# Conclusion

By implementing these security measures, **Heritage Vault 2.0** will strengthen its defenses against cyber threats, unauthorized access, and system vulnerabilities. This structured approach aligns with **industry best practices** for Linux security and ensures that our genealogical data remains protected.

### Next Steps:

* **Implement and test** these security measures, one by one,
* **Combine and automate** all measures into a single script which runs upon startup of all hosts**.**
* **Monitor** system performance and security logs.
* **Regularly update** configurations to stay ahead of emerging threats.

# References

Kennesaw State University. (2021). *Linux server best practices*. Retrieved from <https://www.kennesaw.edu/uits/docs/standards-procedures/linux-server-best-practices.pdf>

TuxCare. (2024). *10 best Linux server security practices for sysadmin in 2024*. Retrieved from <https://tuxcare.com/blog/10-best-linux-server-security-practices-for-sysadmin-in-2024/>

Wiz.io. (2024). *8 essential Linux security best practices*. Retrieved from <https://www.wiz.io/academy/linux-security-best-practices>

Plesk. (2024). *How to secure your Linux server: A detailed guide*. Retrieved from <https://www.plesk.com/blog/various/how-to-secure-your-linux-server-a-detailed-guide/>

Sternum IoT. (2023). *Linux security hardening: 19 best practices with Linux commands*. Retrieved from <https://sternumiot.com/iot-blog/linux-security-hardrining-19-best-practices-with-linux-commands/>