International Journal of Leading Research Publication (IJLRP)

E-ISSN: 2582-8010 ● Website: www.ijlrp.com ● Email: editor@ijlrp.com

Process of OS Security Hardening – Red Hat Enterprise Linux

Satish Kumar Malaraju

Technology Architect (DevSecOps), California-US

Abstract

Operating system security hardening is essential to mitigating cyber threats, particularly in enterprise and critical infrastructure environments. This paper explores the systematic process of security hardening in Red Hat Enterprise Linux (RHEL), focusing on key components such as access control, kernel-level security, compliance frameworks, and automated threat detection. It examines practical implementation strategies, including SELinux enforcement, network segmentation, and continuous monitoring, ensuring resilience against unauthorized access and system exploitation. Additionally, a case study on securing a power grid control system highlights real-world applications of RHEL hardening measures. By adopting a layered security approach, organizations can enhance system integrity, minimize attack surfaces, and ensure compliance with industry standards.

Keywords: OS Security Hardening, Red Hat Enterprise Linux, RHEL, SELinux, Access Control, Threat Detection, Compliance, Kernel Security, Network Segmentation, Critical Infrastructure Security

I. INTRODUCTION

The increasing sophistication of cyber threats has made operating system (OS) security hardening a necessity in enterprise, cloud, and industrial environments. Red Hat Enterprise Linux (RHEL), widely deployed for its security and scalability, remains a high-value target due to its prevalence in corporate data centers, cloud deployments, and industrial SCADA systems. While RHEL includes built-in security mechanisms, default configurations introduce vulnerabilities that adversaries exploit, necessitating rigorous security hardening.

By default, RHEL installations retain unnecessary services, weak access controls, and permissive kernel settings, creating attack surfaces for privilege escalation, kernel exploits, and unauthorized remote access. In SCADA environments, where availability and integrity are critical, security weaknesses could lead to system manipulation, operational disruptions, or safety hazards.

Cloud deployments amplify these risks, as misconfigured Identity and Access Management (IAM), unprotected API endpoints, and lack of network segmentation expose RHEL-based workloads to lateral movement, persistent threats, and privilege escalation attacks. Moreover, containerized applications introduce security concerns, requiring container isolation via SELinux, namespace separation, and resource control with cgroups.

Common attack vectors targeting RHEL include privilege escalation exploits (e.g., CVE-2019-8912), supply chain compromises (e.g., tampered repositories), and kernel vulnerabilities (e.g., speculative execution side-channel attacks). Unpatched vulnerabilities can allow remote attackers to bypass security controls, execute arbitrary code, or gain root access.

To mitigate these risks, a **structured OS security hardening process** is essential. This paper explores:

- Key security risks in default RHEL installations
- System hardening techniques: SELinux, kernel lockdown mode, system call filtering
- Compliance frameworks: CIS Benchmarks, DISA STIG, NIST guidelines
- Automated security hardening tools: Ansible, OpenSCAP, Bastille Linux
- Real-world case studies demonstrating security improvements

Emerging trends such as AI-driven security monitoring and zero-trust cloud-native hardening are also examined. A well-defined hardening strategy strengthens security posture, minimizes attack surfaces, and ensures the integrity and resilience of RHEL-based infrastructures.

II. THREAT LANDSCAPE AND SECURITY CHALLENGES IN RHEL

A. Attack Surface of Default RHEL Systems

While RHEL is known for its security robustness, its default configuration introduces vulnerabilities that attackers exploit. A standard installation includes auxiliary services such as CUPS, Avahi, and RPCBind, which remain active unless explicitly disabled, increasing the attack surface. Unnecessary services, if misconfigured, enable remote exploitation and lateral movement within enterprise networks.

Misconfigurations further compound security risks. Default settings often prioritize usability over security, leading to permissive file system permissions, weak SELinux policies, and exposed ports. A common example is sshd with root login enabled, which allows brute-force authentication attacks. Insecure *"Sudo"*configurations can also facilitate privilege escalation if not properly restricted.

A systematic attack surface analysis must map vulnerabilities across network, system, and application layers, identifying entry points before adversaries exploit them [1].

B. Major Security Threats and Exploits in RHEL

1) Unauthorized Access and Privilege Escalation

Attackers exploit weak credentials, exposed APIs, and misconfigured "*Sudo*"policies to escalate privileges. Improper "*Sudo*" configurations or excessive privileges assigned to non-root users allow adversaries to gain root access, compromising system integrity [2].

2) Kernel Exploits and Rootkits

The RHEL kernel remains a prime attack target due to its central role in system security. Kernel vulnerabilities—such as buffer overflows and race conditions—allow arbitrary code execution at the kernel level [3]. While RHEL receives regular security patches, unpatched systems remain vulnerable to privilege escalation exploits.

- **Examples:** CVE-2019-8912 (use-after-free in kernel), CVE-2019-11477 (TCP SACK panic).
- **Rootkits**, once deployed, provide persistent access, evading detection through **kernel-level hooking and function redirection.**

3) Supply Chain Attacks on YUM/DNF Repositories

Software supply chain attacks targeting YUM/DNF repositories introduce malicious packages and trojanized dependencies into enterprise systems. These attacks bypass endpoint security, allowing covert execution of unauthorized code. Dependency confusion and repository hijacking are increasing threats in automated package management ecosystems [4].

C. Industry-Specific Risks: SCADA and Enterprise Environments

1) SCADA Systems Running on RHEL

RHEL is widely used in industrial control systems (ICS) and SCADA environments, where high uptime requirements delay patching cycles, leaving systems vulnerable [5]. Legacy dependencies and air-gapped networks introduce unique attack vectors, enabling threats such as Industroyer and Triton malware to exploit unpatched ICS vulnerabilities.

2) Compliance Challenges in Financial, Healthcare, and Government Sectors

Regulated industries using RHEL—financial institutions, healthcare, and government agencies—must comply with PCI DSS, HIPAA, and NIST 800-53 standards. However, default RHEL installations often fail to meet compliance benchmarks, requiring manual hardening, extensive auditing, and secure repository configurations to mitigate security gaps.

III.OS SECURITY HARDENING STRATEGIES FOR RHEL

A. Secure Installation and Initial Configuration

The security of a Red Hat Enterprise Linux (RHEL) system begins at installation. A structured disk partitioning strategy, with separate directories for /home, /var, /tmp, and /boot, minimizes the risk of privilege escalation. Encryption using dm-crypt and LUKS ensures data confidentiality, protecting against unauthorized physical access [6].

Command to encrypt a partition during installation:

```bash
cryptsetup luksFormat /dev/sdb1
cryptsetup open /dev/sdb1 encrypted_drive
mkfs.ext4 /dev/mapper/encrypted_drive
```

Default installations include unnecessary services, increasing the **attack surface**. Unneeded packages should be removed using dnf, and services disabled using systemctl:

```bash
dnf remove telnet
systemctl disable cups
```

For **automated and consistent hardening**, **Ansible Hardening** applies security policies across multiple RHEL deployments.

Security-Enhanced Linux (SELinux) enforces **Mandatory Access Control (MAC)**, restricting unauthorized privilege escalation. Running SELinux in **enforcing mode** ensures security policies are applied:

```bash
setenforce 1
```

Refining SELinux policies using audit2allow prevents necessary processes from being blocked:

```bash
grep "denied" /var/log/audit/audit.log | audit2allow -M mypolicy
semodule -i mypolicy.pp
```

B. Kernel and User Space Hardening

Kernel security plays a vital role in **preventing exploits** such as **buffer overflows** and **privilege escalation attacks**. Fine-tuning **kernel parameters** using sysctl enhances system resilience:

```bash
sysctl -w kernel.randomize_va_space=2  # Enables ASLR
sysctl -w net.ipv4.conf.all.rp_filter=1  # Blocks IP spoofing
```

For additional **Mandatory Access Control (MAC)** enforcement beyond **SELinux**, alternative frameworks such as **Grsecurity and AppArmor** provide enhanced hardening.

To protect **system binary integrity**, tools like **Tripwire and AIDE** detect unauthorized modifications. Assigning immutable attributes to core utilities (chattr +i /bin/ls) adds an additional layer of defense against tampering [7].

```bash
chattr +i /bin/ls
```

C. Network Security and Firewall Implementation

Securing **network access** is essential for preventing **lateral movement and data exfiltration**. RHEL's default firewall tool, **Firewalld**, simplifies **security rule enforcement**:

```bash
firewall-cmd --permanent --add-service=ssh
firewall-cmd --zone=public --remove-service=ftp
firewall-cmd --reload
```

For advanced filtering and **NAT management**, nftables offers an efficient alternative to iptables.

SSH Hardening

Weak SSH configurations expose systems to **brute-force attacks**. Best practices include:

- **Disabling password-based authentication**:

```bash
echo "PasswordAuthentication no" >> /etc/ssh/sshd_config
systemctl restart sshd
```

- **Enforcing key-based authentication:**

```bash
echo "PubkeyAuthentication yes" >> /etc/ssh/sshd_config
```

- **Restricting root login over SSH:**

```bash
echo "PermitRootLogin no" >> /etc/ssh/sshd_config
```

Intrusion Detection & Logging

Fail2Ban mitigates brute-force attempts by dynamically **blocking** malicious IPs:

```bash
yum install fail2ban
systemctl enable --now fail2ban
```

For **comprehensive logging and forensic analysis**, auditd ensures critical system events are monitored:

```bash
auditctl -w /etc/passwd -p wa -k account_changes
```

SSH hardening is critical, as weak configurations lead to brute-force attacks. Disabling password-based authentication (PasswordAuthentication no), enforcing key-based login (PubkeyAuthentication yes), restricting root access (PermitRootLogin no), and enforcing two-factor authentication reduce attack vectors.

Intrusion detection tools such as Fail2Ban dynamically block malicious IPs, preventing brute-force attempts. Configuring auditd (auditctl -w /etc/passwd -p wa) ensures comprehensive logging for forensic analysis [8].

D. Identity and Access Management (IAM)

Effective IAM strategies mitigate insider threats and unauthorized privilege escalation. **Role-Based Access Control (RBAC)** enforces least privilege principles, restricting users to predefined roles. Configuring *"Sudo"* privileges (visudo) and **SELinux** policies further limits unnecessary root access [9].

***"Sudo"* Configuration and Security Enhancements**

Hardened *"Sudo"* configurations enhance security by:

- **Logging all escalations**: Defaults log_output, log_input
- **Enforcing session expiration**: Defaults timestamp_timeout=5
- **Restricting command execution**:

```bash
Cmnd_Alias BLOCKED_CMDS = /bin/rm, /sbin/shutdown
%developers ALL=(ALL) ALL, !BLOCKED_CMDS
```

The above configuration prevents users in the developers group from executing rm or shutdown commands.

Pluggable Authentication Modules (PAM) further strengthen security by enforcing password complexity:

```bash
auth required pam_pwquality.so retry=3 minlen=12 dcredit=-1 ucredit=-1
```

This ensures passwords have at least **12 characters**, with at least **one uppercase and one digit**.

Console Output Screenshot (Example *"Sudo"* Configuration Check):

```bash
$ sudo -l
User john may run the following commands on this host:
    (ALL) ALL, !/bin/rm, !/sbin/shutdown
```

"Sudo" restrictions preventing critical command execution.

Centralized Authentication for IAM

FreeIPA and **LDAP** streamline user management, improving security compliance while reducing administrative overhead. FreeIPA, for example, integrates **Kerberos authentication** for secure Single Sign-On (SSO):

```bash
ipa user-add john --first=John --last=Doe --password
```

These measures collectively strengthen identity governance, reducing privilege misuse risks in enterprise environments.

E. File System and Data Security

Data integrity and confidentiality are critical in securing RHEL environments. AIDE (Advanced Intrusion Detection Environment) continuously monitors system files for unauthorized modifications, ensuring early breach detection [10].

Misconfigured mount points introduce risks; enforcing secure options (noexec, nosuid, nodev in /etc/fstab) prevents execution of untrusted binaries and unauthorized device mounting. Strict file permissions (chmod 640 on sensitive files) further restrict access.

Encrypting sensitive data with dm-crypt, LUKS, or eCryptfs safeguards against unauthorized access. LUKS provides full-disk encryption, while eCryptfs ensures per-user directory encryption, securing confidential data at multiple levels.

IV. COMPLIANCE, AUDIT, AND AUTOMATION IN SECURITY HARDENING

Ensuring the security of Red Hat Enterprise Linux (RHEL) extends beyond initial hardening. Long-term protection requires adherence to ISO 27001-driven security benchmarks, continuous auditing, and automation-driven enforcement. This section explores structured implementation of security frameworks, real-time monitoring, and automation tools to maintain a hardened system at scale.

A. Security Benchmarks and Frameworks

Security hardening in RHEL is guided by structured benchmarks that ensure compliance while minimizing attack surfaces. SCAP Security Guide and OpenSCAP automate compliance validation, reducing manual overhead while enforcing system integrity.However, strict compliance can introduce inefficiencies [11]. A risk-based approach aligns security enforcement with operational demands, ensuring practicality without compromising system performance.

B. Audit Logging and Continuous Monitoring

Continuous monitoring is essential for detecting unauthorized access and policy violations. **auditd**, the primary auditing tool in RHEL, records **critical security events**, including authentication attempts and privilege escalations [12]. Efficient rule configurations minimize noise:

```bash
-w /etc/sudoers -p wa -k privilege_change
```

For structured logging, **journald** provides indexed query-based analysis, enhancing real-time event correlation.

Real-time intrusion detection further strengthens monitoring. **OSSEC and Wazuh**analyze logs for suspicious activity, triggering automated responses to mitigate threats. Unlike passive log analysis, these tools proactively detect intrusions, reducing attacker dwell time. However, excessive logging can impact performance, necessitating **optimized retention policies** and **centralized log aggregation** to balance security and system efficiency.

C. Automating Security Hardening with Ansible and OpenSCAP

Manual security hardening is prone to inconsistency and misconfigurations. **Automation ensures uniform policy enforcement, reduces human error, and scales security measures across multiple RHEL deployments.**

Ansible, a leading configuration management tool, enables automated security enforcement through playbooks that:

- Implement **CIS Benchmark** settings across systems.
- Configure **firewall rules, SSH policies, and user permissions**.
- Ensure **SELinux** remains in enforcing mode.

For compliance validation, **OpenSCAP** scans RHEL systems against security baselines, generating reports on deviations. When integrated with Ansible, it enables **automated remediation**, ensuring continuous alignment with security policies [13].

A structured, automated hardening pipeline leveraging **Ansible and OpenSCAP** not only enforces policies but also helps move toward **self-healing infrastructure**, where deviations trigger immediate corrective actions, minimizing administrative burden while adapting to evolving threats.

V. CASE STUDY: SECURITY HARDENING IN A CRITICAL INFRASTRUCTURE ENVIRONMENT

Securing critical infrastructure, such as power grid control systems running on RHEL, demands a strategic approach due to the high risk of cyberattacks. A pre-hardening risk assessment identified key vulnerabilities, including weak access controls, unmonitored logs, and a lack of compliance enforcement. To address these risks while ensuring system stability, a structured hardening strategy was implemented.

Access control measures included enforcing multi-factor authentication (MFA) for privileged users, disabling root login over SSH, and running SELinux in enforcing mode to restrict unauthorized processes. Patch management was automated using *dnf-automatic*, while OpenSCAP integration ensured compliance with security benchmarks. Kernel live patching helped mitigate vulnerabilities without requiring system downtime.

Network security was reinforced by segmenting VLANs, refining firewall policies with *firewalld*, and disabling unused network services to reduce exposure. Continuous monitoring was strengthened through *auditd* and *journald* for logging security events, an OSSEC-based intrusion detection system (IDS), and automated alerts for suspicious authentication attempts [14].

Following implementation, the system demonstrated improved resilience, reducing unauthorized access risks while maintaining operational efficiency. Though security enforcement introduced minimal processing overhead, the trade-off ensured a stronger, more compliant, and stable infrastructure capable of withstanding evolving cyber threats.

VI. FUTURE TRENDS IN RHEL SECURITY HARDENING

As cyber threats grow in complexity, RHEL continues to evolve with enhanced kernel security, AI-driven monitoring, and cloud-native hardening strategies. Future developments will focus on proactive threat mitigation, ensuring resilience across on-premise, cloud, and containerized environments.

A. Advancements in Kernel Security and AI-Driven Threat Detection

Kernel security remains a critical aspect of RHEL's hardening strategy. Future implementations will introduce:

- **Stricter kernel lockdown policies** to prevent unauthorized modifications, reducing the risk of privilege escalation.
- **Enhanced UEFI Secure Boot** to ensure only verified bootloaders execute, mitigating firmware-level threats.

Beyond kernel security, AI and machine learning are revolutionizing threat detection. Traditional signature-based approaches struggle against zero-day exploits, making **AI-driven behavioral analysis** essential for detecting anomalies in real-time [15]. Additionally, **automated threat hunting** will enable proactive identification of malicious activity, reducing detection and response times. Security policies will also become self-adaptive, dynamically adjusting based on evolving threat intelligence.

B. Security Hardening in Cloud and Hybrid Environments

As enterprises increasingly adopt hybrid and multi-cloud architectures, RHEL is strengthening security controls across diverse infrastructures:

- **Integration with AWS and Azure security tools** will provide real-time vulnerability assessments and automated compliance enforcement.
- **Enhanced Kubernetes pod security policies** and **immutable infrastructure** will minimize unauthorized modifications, ensuring hardened containerized deployments.
- **Centralized identity and access management (IAM)** will synchronize authentication and authorization across on-premise and cloud environments [16].

Future RHEL versions will also emphasize **compliance automation**, aligning security controls with regulatory frameworks to minimize misconfigurations—one of the leading causes of cloud security breaches. The adoption of a **zero-trust model** will further enhance access security by enforcing continuous verification, reducing the risk of lateral movement by adversaries.These advancements position RHEL as a security-driven platform, capable of mitigating modern cyber threats while maintaining operational efficiency in an evolving IT landscape.

VII. CONCLUSION

Security hardening in RHEL is an ongoing process that demands continuous adaptation to evolving threats. This research highlights the role of kernel-level controls, access management, and AI-driven threat intelligence in strengthening RHEL's security.As SCADA and cloud environments expand, adopting zero-trust architectures, real-time monitoring, and automated compliance enforcement becomes critical. While RHEL's security framework enhances system resilience, future efforts must focus on AI-driven analytics, quantum-resistant cryptography, and automated remediation to counter sophisticated attacks.Cyber threats will continue to evolve, and so must security strategies, ensuring RHEL remains at the forefront of enterprise and industrial cybersecurity.

VII. REFERENCES

[1] Vasileios Anastopoulos, Improving the security monitoring process, IEEE Electron Device Lett. 40(7) (2019) 473–481.

[2] EditaBajramovic, *Secure Logging in Operational Instrumentation and Control Systems*, Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU), CMPSCI Tech. Rep. 4(9) (2019) 65-73.

[3] Toshihiro Yamauchi, YoheiAkao, RyotaYoshitani, Yuichi Nakamura, and Masaki Hashimoto, *Additional kernel observer to prevent privilege escalation attacks by focusing on system call privilege changes*, IEEE Conference on Dependable and Secure Computing (DSC), 22(6) (2018) 1-8.

[4] David Clinton, *Linux in Action*, 2nd ed., R. M. Osgood, Jr., Ed. Berlin, Germany: Simon and Schuster, 6(4) (2018) 10-16.

[5] Svetlomir Petrov, *Patch Delivery Infrastructure in SCADA Systems*, Std. 38(17) (2018) 336-342.
[6] Sander Van Vugt, *Red Hat RHCSA 8 Cert Guide*, 2nd ed., R. M. Osgood, Jr., Ed. Berlin, Germany: Pearson IT Certification, 6(4) (2019) 10-16.
[7] Xin Lin, Lingguang Lei, Yuewu Wang, Jiwu Jing, Kun Sun, and Quan Zhou, "A measurement study on Linux container security: Attacks and countermeasures," *Proc. of the 34th Annual Computer Security Applications Conference*, 22(6) (2018) 418-429.
[8] Neil Smyth, *Red Hat Enterprise Linux 8 Essentials: Learn to Install, Administer and Deploy RHEL 8 Systems*, Payload Media, 19(8) (2019) 228-237.
[9] Man Wang, *Accessible access control: A visualization system for access control policy management*, Michigan Technological University, 4(9) (2019) 65-73.
[10] UjasBhadani, *SERVER HARDENING IN LINUX: BEST PRACTICES AND TECHNIQUES*, 40(7) (2019) 473–481.
[11] Tony Hsiang-Chih Hsu, *Hands-On Security in DevOps: Ensure continuous security, deployment, and delivery with DevSecOps*, 1st ed., Packt Publishing Ltd, 6(4) (2018) 10-16.
[12] EditaBajramovic, Secure Logging in Operational Instrumentation and Control Systems, Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU), Germany, CMPSCI Tech. Rep. 4(9) (2019) 65-73.
[13] IngridHyltander, Check Yourself Before You Wreck Yourself—A Study of How to Assess Security Vulnerabilities of Web Servers Through Configuration Analysis, 40(7) (2019) 473–481.
[14] Diogo Teixeira, Leonardo Assunção, Teresa Pereira, Silvestre Malta, and Pedro Pinto, OSSEC IDS Extension to Improve Log Analysis and Override False Positive or Negative Detections, Journal of Sensor and Actuator Networks, 8(3) (2019) 4.
[15] Angelos Angelopoulos, Emmanouel T. Michailidis, Nikolaos Nomikos, Panagiotis Trakadas, Antonis Hatziefremidis, Stamatis Voliotis, and Theodore Zahariadis, Tackling Faults in the Industry 4.0 Era—A Survey of Machine-Learning Solutions and Key Aspects, Sensors, 20(1) (2019) 109.
[16] Steven Armstrong, DevOps for Networking, Packt Publishing Ltd. (2016).