

# **Enterprise Linux Host Hardening and Automation Readiness**

This project documents the standardized configuration and security hardening of multiple Linux hosts across development, performance, and staging environments in an enterprise-style infrastructure.

All systems are deployed on vSphere and run CentOS Stream 9, with operating system versions verified prior to configuration. Hostnames, networking, and system identity are aligned to ensure consistency with centralized services and automation tooling.

A major focus of this work is secure remote access. SSH is hardened on every host by disabling direct root login, enforcing strict authentication controls, and validating daemon configuration after each change. Password and key-based authentication behavior is explicitly reviewed to align with organizational security standards.

Network access is further restricted using firewall. Default SSH services and broad port rules are removed, and access is constrained using targeted rules that allow administrative connectivity only from trusted network ranges. These controls are applied consistently across application, performance, stage web, and automation hosts, demonstrating repeatable security enforcement.

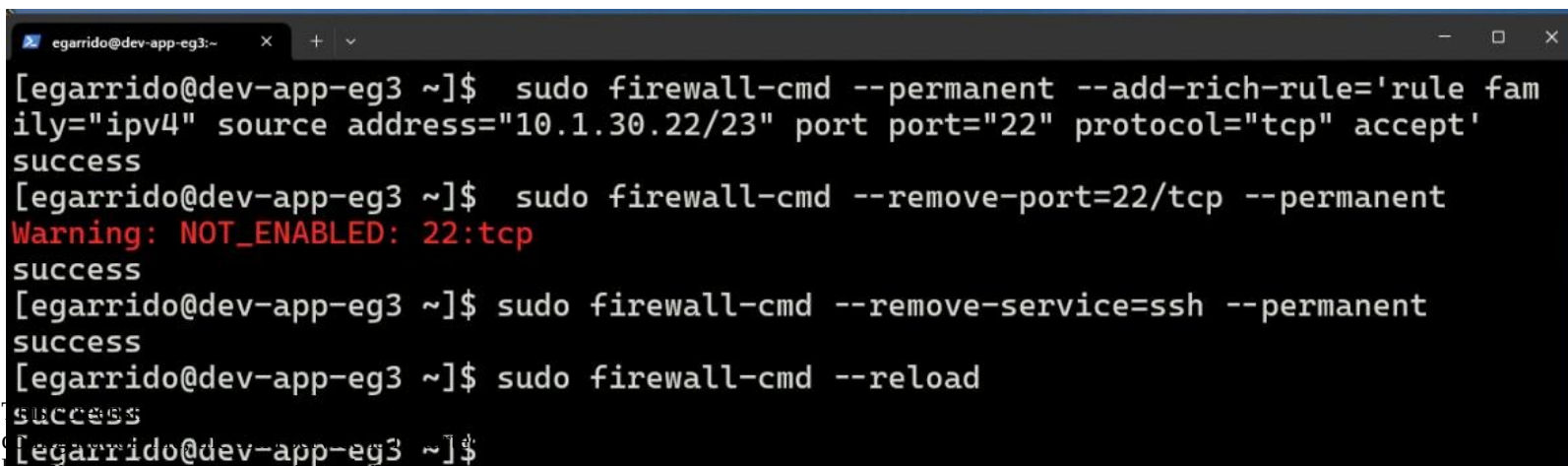
To support automation, SSH key-based trust is established bidirectionally between hosts, enabling passwordless access where appropriate. Host key conflicts and trust issues are identified, corrected, and validated to ensure reliable remote execution.

Connectivity and readiness for automation are verified using Ansible, confirming that all hardened hosts remain reachable and correctly configured for managed operations.

## **Security & Sanitization Notice**

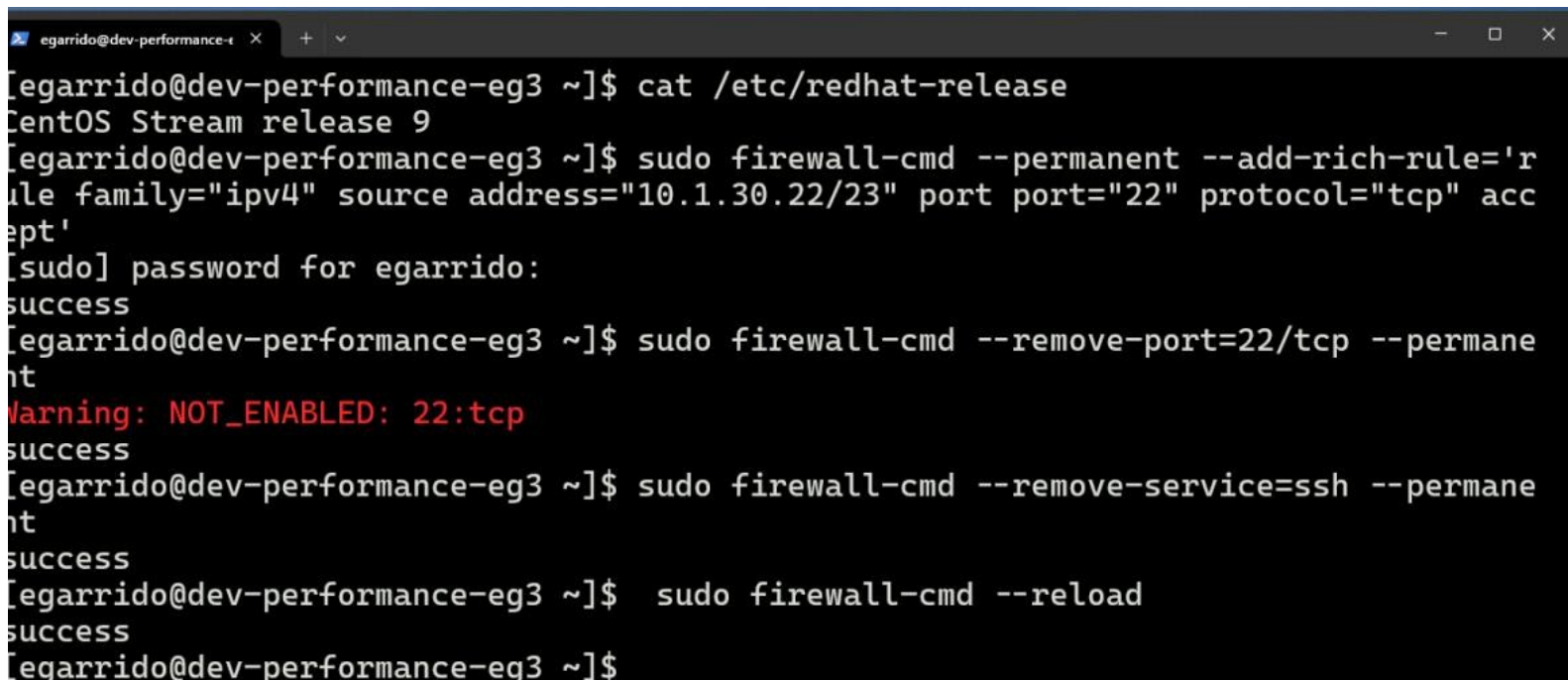
All IP addresses, hostnames, usernames, network ranges, timestamps, and environment-specific identifiers shown in screenshots, command output, and configuration files have been sanitized or obfuscated. No production credentials or sensitive infrastructure data are exposed in this repository.

This screenshot shows firewalld hardening and SSH access restriction on the application server. A rich rule is added to permanently allow SSH (TCP port 22) only from a specific trusted subnet, replacing broad access. Default SSH service and open port rules are then removed to eliminate unrestricted exposure, and the firewall configuration is reloaded to apply the changes. This enforces least-privilege network access while maintaining required administrative connectivity

A terminal window with a dark background and light text. The window title is 'egarrido@dev-app-eg3:~'. The terminal shows a series of commands and their outputs. The first command adds a rich rule for SSH access from a specific subnet. The second command removes the default SSH service rule, resulting in a warning. The third command removes the default SSH port rule. The fourth command reloads the firewall configuration. All commands are successful.

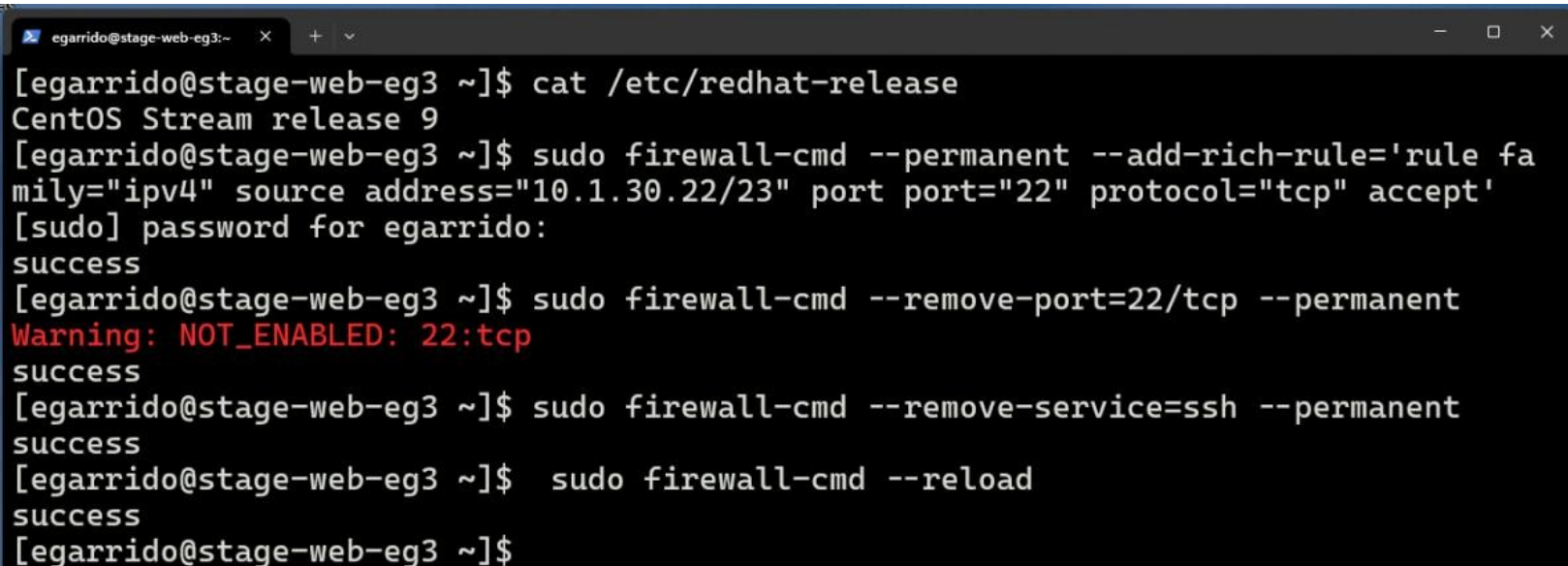
```
egarrido@dev-app-eg3:~  
[egarrido@dev-app-eg3 ~]$ sudo firewall-cmd --permanent --add-rich-rule='rule family="ipv4" source address="10.1.30.22/23" port port="22" protocol="tcp" accept'  
success  
[egarrido@dev-app-eg3 ~]$ sudo firewall-cmd --remove-port=22/tcp --permanent  
Warning: NOT_ENABLED: 22:tcp  
success  
[egarrido@dev-app-eg3 ~]$ sudo firewall-cmd --remove-service=ssh --permanent  
success  
[egarrido@dev-app-eg3 ~]$ sudo firewall-cmd --reload  
success  
[egarrido@dev-app-eg3 ~]$
```

This screenshot demonstrates firewalld hardening and OS verification on the performance server. The operating system is first confirmed as CentOS Stream 9, after which firewall rules are tightened to restrict SSH access. A permanent rich rule is added to allow TCP port 22 only from a trusted subnet, while the default open SSH service and generic port rule are removed. The firewall is then reloaded to apply the changes, enforcing least-privilege network access while preserving required administrative connectivity.

A terminal window with a dark background and light text. The window title is 'egarrido@dev-performance-eg3'. The terminal shows a series of commands and their outputs. The first command is 'cat /etc/redhat-release', which outputs 'CentOS Stream release 9'. The second command is 'sudo firewall-cmd --permanent --add-rich-rule='rule family="ipv4" source address="10.1.30.22/23" port port="22" protocol="tcp" accept'', which prompts for a password and then outputs 'success'. The third command is 'sudo firewall-cmd --remove-port=22/tcp --permanent', which outputs 'Warning: NOT\_ENABLED: 22:tcp' and 'success'. The fourth command is 'sudo firewall-cmd --remove-service=ssh --permanent', which outputs 'success'. The fifth command is 'sudo firewall-cmd --reload', which outputs 'success'. The prompt returns to the user's shell.

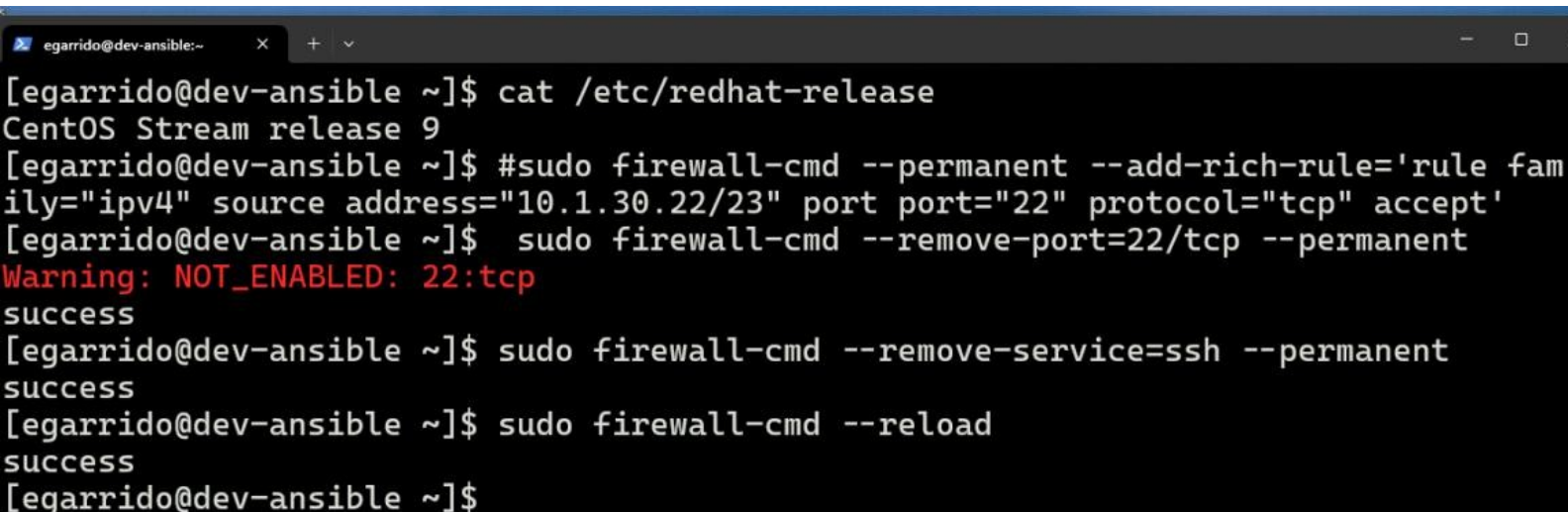
```
egarrido@dev-performance-eg3 ~]$ cat /etc/redhat-release
CentOS Stream release 9
egarrido@dev-performance-eg3 ~]$ sudo firewall-cmd --permanent --add-rich-rule='rule family="ipv4" source address="10.1.30.22/23" port port="22" protocol="tcp" accept'
[sudo] password for egarrido:
success
egarrido@dev-performance-eg3 ~]$ sudo firewall-cmd --remove-port=22/tcp --permanent
Warning: NOT_ENABLED: 22:tcp
success
egarrido@dev-performance-eg3 ~]$ sudo firewall-cmd --remove-service=ssh --permanent
success
egarrido@dev-performance-eg3 ~]$ sudo firewall-cmd --reload
success
egarrido@dev-performance-eg3 ~]$
```

This screenshot shows OS verification and firewall hardening on the stage web server. The operating system is confirmed as CentOS Stream 9, after which firewall rules are tightened using firewalld. A permanent rich rule is added to allow SSH access only from a trusted subnet, while the default SSH service and generic port rule are removed. The firewall configuration is then reloaded to apply the changes, enforcing restricted, least-privilege SSH access while maintaining required administrative connectivity



```
egarrido@stage-web-eg3:~  
[egarrido@stage-web-eg3 ~]$ cat /etc/redhat-release  
CentOS Stream release 9  
[egarrido@stage-web-eg3 ~]$ sudo firewall-cmd --permanent --add-rich-rule='rule family="ipv4" source address="10.1.30.22/23" port port="22" protocol="tcp" accept'  
[sudo] password for egarrido:  
success  
[egarrido@stage-web-eg3 ~]$ sudo firewall-cmd --remove-port=22/tcp --permanent  
Warning: NOT_ENABLED: 22:tcp  
success  
[egarrido@stage-web-eg3 ~]$ sudo firewall-cmd --remove-service=ssh --permanent  
success  
[egarrido@stage-web-eg3 ~]$ sudo firewall-cmd --reload  
success  
[egarrido@stage-web-eg3 ~]$
```

This screenshot shows OS verification and firewall rule cleanup on the Ansible control host. The system is confirmed to be running CentOS Stream 9, and the firewall configuration is adjusted to remove broad SSH exposure by deleting the default SSH service and generic port 22 rules. The firewall is then reloaded to apply the changes, aligning access control with a least-privilege security model for automation infrastructure.

A terminal window with a dark background and light text. The window title bar shows 'egarrido@dev-ansible:~' and standard window controls. The terminal output shows a series of commands and their results: checking the OS release, adding a specific rule for port 22, removing the default port 22 rule (which triggers a warning), removing the SSH service rule, and finally reloading the firewall.

```
egarrido@dev-ansible:~$ cat /etc/redhat-release
CentOS Stream release 9
egarrido@dev-ansible:~$ #sudo firewall-cmd --permanent --add-rich-rule='rule fam
ily="ipv4" source address="10.1.30.22/23" port port="22" protocol="tcp" accept'
egarrido@dev-ansible:~$ sudo firewall-cmd --remove-port=22/tcp --permanent
Warning: NOT_ENABLED: 22:tcp
success
egarrido@dev-ansible:~$ sudo firewall-cmd --remove-service=ssh --permanent
success
egarrido@dev-ansible:~$ sudo firewall-cmd --reload
success
egarrido@dev-ansible:~$
```

## Summary

This project demonstrates consistent security hardening and automation preparation across multiple Linux hosts in development, performance, and staging environments. Systems running CentOS Stream 9 are verified, SSH access is hardened, firewall rules are tightened to restrict access to trusted sources, and secure key-based authentication is validated. Ansible connectivity is confirmed after hardening to ensure all hosts remain manageable. All IP addresses, hostnames, usernames, and environment-specific details shown are sanitized for security purposes.