Ansible Infrastructure Security and Backup Automation

This body of work demonstrates the use of Ansible to automate critical infrastructure security controls and system protection tasks across Linux environments. The focus is on enforcing secure network access, safeguarding kernel-level components, and validating results through direct system verification. Together, these automations reflect real-world operational workflows used in enterprise Linux administration.

The work spans development and staging systems and emphasizes repeatability, idempotency, and validation, ensuring changes are applied consistently and verified after execution.

Firewall Hardening and Access Control

The first automation enforces firewall hardening using firewalld by disabling unnecessary network exposure. Commonly exploited web ports are closed, and changes are applied both immediately and permanently. SSH access is tightly controlled using source-based rules, allowing administrative access only from approved networks while explicitly rejecting unauthorized sources.

Post-deployment verification confirms that only essential services remain accessible and that firewall rules are actively enforced, reducing the system’s attack surface.

Kernel Module Backup Automation

The second automation focuses on protecting critical system components by backing up Linux kernel modules prior to maintenance or upgrades. Using Ansible, kernel modules are archived from multiple hosts and stored on a centralized NFS share. Backups are compressed, host-specific, and consistently named using inventory identifiers.

This approach ensures that recovery artifacts are readily available and centrally managed, supporting faster rollback and improved operational resilience.

Ansible Kernel Module Backup Automation

This project demonstrates the use of Ansible to automate the backup of Linux kernel modules across multiple environments in a centralized and repeatable manner. The playbook is designed to capture critical system components prior to maintenance, upgrades, or troubleshooting activities, ensuring recovery artifacts are readily available if rollback is required.

The automation targets defined hosts and executes with elevated privileges to access system-level directories. Backups are stored on a shared NFS location, enabling centralized retention and easy retrieval across environments.

Automated Backup Workflow

The playbook performs the following actions:

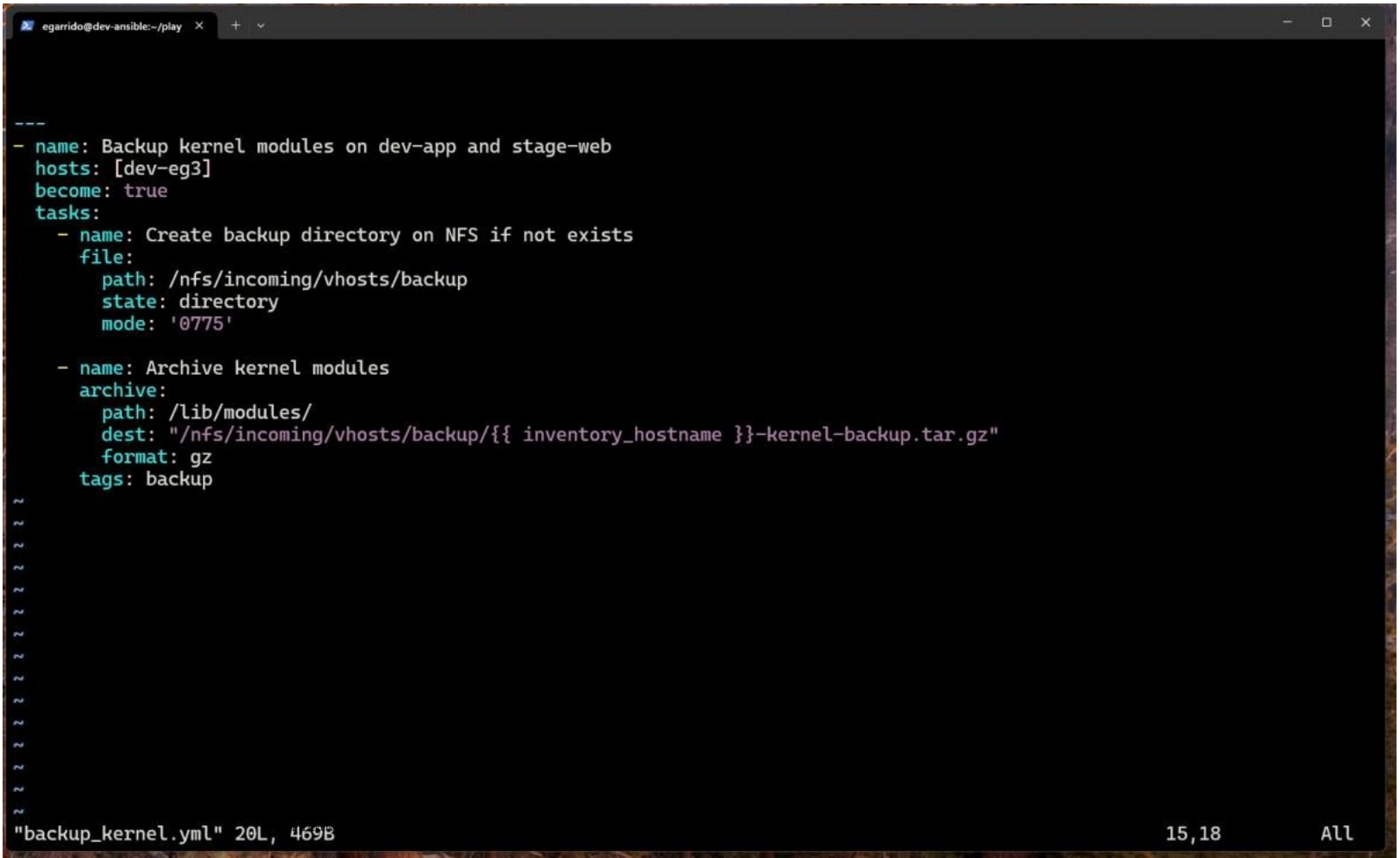
Ensures a dedicated backup directory exists on a mounted NFS share

Archives the contents of /lib/modules/ on each managed host

Creates a compressed, host-specific backup file using the inventory hostname

Applies consistent permissions to support shared access and retention policies

Each task is idempotent, allowing the playbook to be re-run safely without duplicating directories or introducing configuration drift.



Ansible Kernel Module Backup and Validation

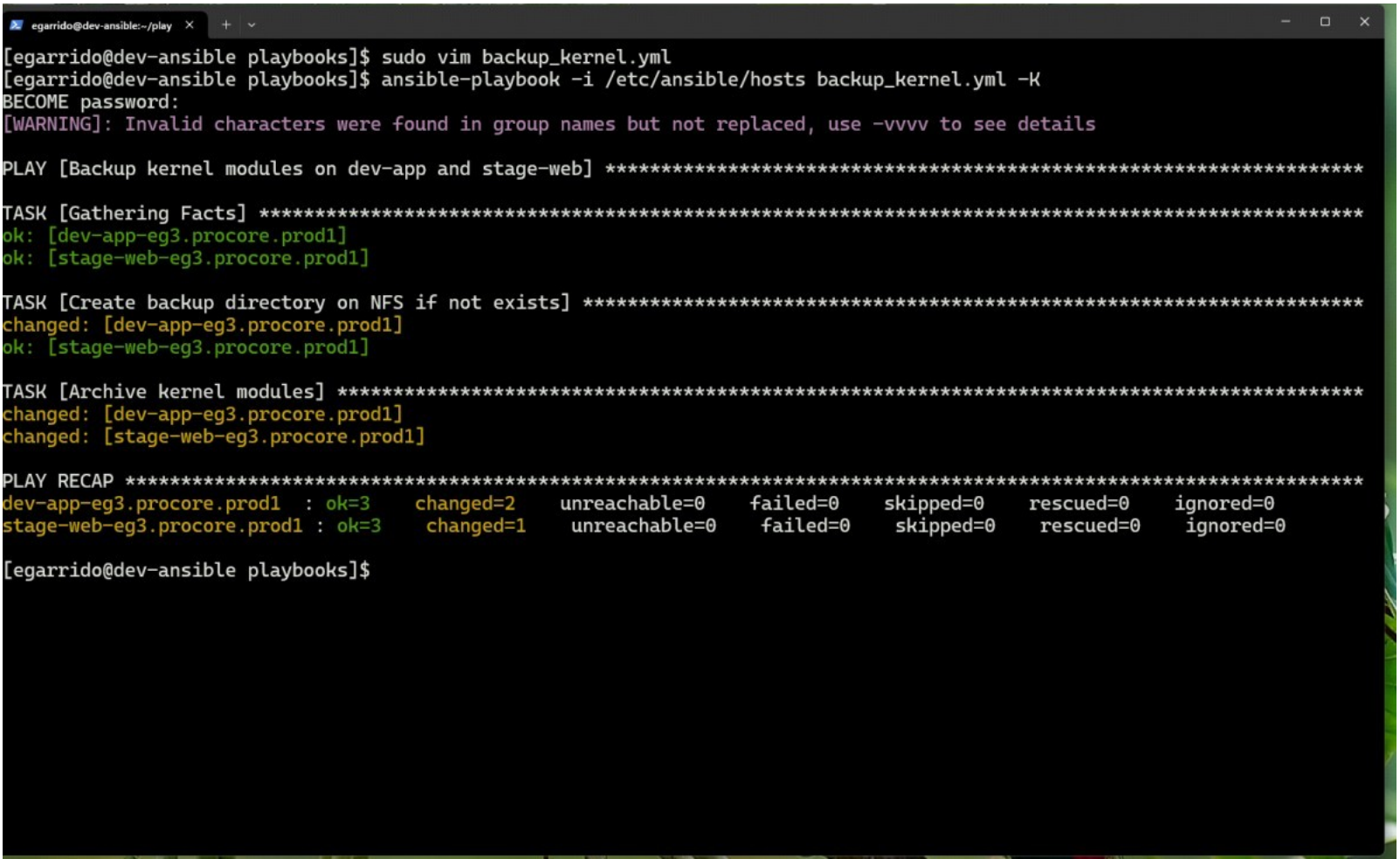
This project demonstrates the use of Ansible to automate the backup of Linux kernel modules across multiple hosts in a controlled and repeatable manner. The playbook is designed to protect critical system components prior to maintenance, upgrades, or troubleshooting by creating centralized, host-specific backup archives.

The automation targets both development and staging systems and runs with elevated privileges to access system directories. Backups are written to a shared NFS location, allowing for centralized storage, simplified management, and consistent retention across environments.

Execution Results and Verification

The playbook executes successfully on both development and staging hosts with no failures or unreachable systems. Directory creation is reported as changed only where required, and kernel module archives are generated for each host. The final play recap confirms successful completion with clear reporting of applied changes per system.

This output verifies that kernel module backups were created and stored centrally as intended.



Ansible Kernel Module Backup Verification

This project documents the validation of kernel module backups created through Ansible automation and stored on a centralized NFS share. After executing the backup playbook across development and staging systems, verification is performed directly on the filesystem to confirm successful archive creation, proper naming, and consistent storage across hosts.

The verification process ensures that each managed system produces a unique, host-specific backup artifact and that the backups are accessible from multiple environments, demonstrating centralized storage and cross-host visibility.

Backup Validation Process

Validation is performed using standard Linux commands to inspect the shared NFS backup directory:

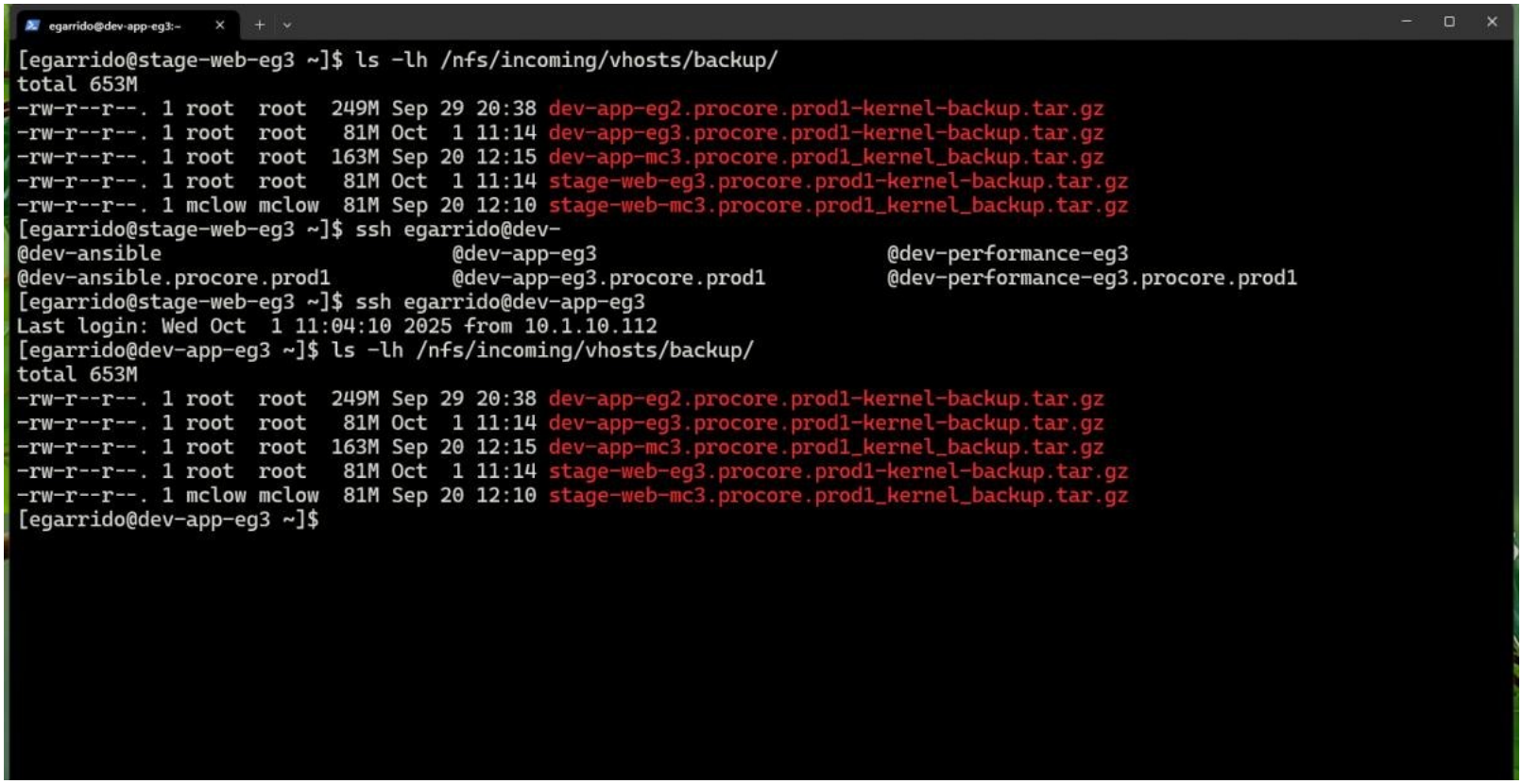
Lists backup files using ls -lh to confirm presence, size, and timestamps

Verifies that each archive follows a consistent naming convention based on the inventory hostname

Confirms backups are visible from multiple hosts, validating shared NFS access

Ensures kernel module archives are stored in compressed .tar.gz format for efficient retention

The output confirms that kernel module backups were successfully generated for multiple development and staging hosts and stored centrally as intended.



Summary

This work demonstrates the use of Ansible to automate firewall hardening, protect critical kernel components through centralized backups, and validate results through direct system verification. Together, these tasks showcase a complete operational workflow that reduces attack surface, improves recovery readiness, and enforces secure, repeatable infrastructure management practices suitable for enterprise Linux environment