Operating System Hardening on Service Hosts

The goal of the hardening applied to the hosts that make up the infrastructure is to ensure a secure, predictable, and resilient environment against attacks. These machines are not designed for personal use, but exclusively to run critical services; therefore, the implemented measures aim to minimize the attack surface, reinforce system integrity, and guarantee the reliable operation of the services.

1. User Management and Shell Restrictions

All system users are configured as accounts without login capability, except for a single user who uses a restricted shell. We can't apply no login to that user because it is used for ssh login. Since these machines are not intended for interactive use, any administrative action requires explicit privilege escalation.

The purpose of this measure is to:

- Prevent the misuse of human accounts on systems where interactive access is unnecessary.

- Ensure that service or technical accounts cannot be used as an entry point.

This approach reduces the risk of account abuse, mitigates lateral movement, and provides clear control over who can perform which actions.

2. Simulated Installations “Fake Install” for File System Structures

Mechanisms are implemented to simulate installations or structures required by certain services without affecting the base system. This approach allows:

- Maintaining controlled environments for specific services.

- Avoiding vulnerabilities of old, not-maintained filesystems.

This separation enhances the operational robustness of the host and prevents unauthorized modifications.

3. Integrity Control with AIDE

To ensure the integrity of key files, a monitoring system is configured to regularly analyze system directories, service configuration files, and application components. The objective is to detect any unauthorized or unexpected modification, whether malicious or accidental.

This verification applies to:

- Configuration of critical services such as NGINX web servers, PHP engines, or custom MariaDB applications.
- System and service logs.
- Operating system components and essential binaries.
- Scheduler elements (cron) and the system service manager.
- Remote access and firewall configuration.

Periodic verification ensures that any suspicious alteration is identified quickly. Automating this process through scheduled tasks enables continuous monitoring without manual intervention.

4. Kernel Hardening

Security policies are applied directly to the kernel to reinforce network behavior and prevent insecure practices. These measures aim to:

- Prevent unwanted routing and ensure that the system does not act as a router.
- Reject redirects or ICMP messages that could be exploited to redirect traffic or scan the network.
- Implement anti-spoofing filters to mitigate spoofing attacks.
- Enable protection mechanisms against denial-of-service attacks, such as those based on saturating TCP connections.
- Reduce exposure to ambiguous or malformed traffic.

These measures significantly enhance network defense, reduce the likelihood of traffic manipulation, and prevent the server from participating in unauthorized or malicious activities.

5. Hardening with AppArmor

Strict profiles are enabled and applied to control the behavior of each service. AppArmor allows defining which files, directories, and capabilities each system process may use, isolating it from the rest.

For services that already have profiles, strengthened variants are applied. For custom services or those without pre-existing profiles—such as the NGINX web server—specific profiles are generated using behavior-analysis tools.

However, to avoid operational risks or unexpected interruptions in critical services, these profiles are initially executed in complain mode. This approach makes it possible to observe and evaluate the actual behavior of each service before enabling strict enforcement. Later, a detailed review will allow migrating them to full enforcement mode without compromising system stability.

6. Automatic Security Updates

An automatic update system is enabled, focusing exclusively on security patches to avoid unplanned changes that could affect production services. The goal is to:

- Keep the system protected against newly discovered vulnerabilities.
- Reduce the exposure window to known attacks.
- Ensure service continuity by avoiding automatic restarts or potentially disruptive updates.

The process is automated and periodically reviewed via scheduled tasks to guarantee proper operation.