1. Introduction:

Given that we find ourselves in an age where cyberthreats are becoming more sophisticated and widespread, it is imperative that our computing systems are adequately protected by strong, multilayered security protection. In the vast array of tools for protecting from unauthorized access, privilege escalation, and abuse of resources, three technologies developed on Linux platform attract particular attention: SECCOMP, SELinux and chroot. They all focus on different parts of system protection, but as whole, they are an example of the principle of defense-in-depth: applying several aspects of defense in such a way that the failure of one aspect stays compensated by the functioning of the others together.

This paper examines these mechanisms in detail, dissecting their underlying concepts, virtues, and flaws. It contrasts their place within contemporary security architectures, and draws on common use cases — ensuring that containerized workloads are hardened, or that compliance critical systems are secured. Admins and developers who understand how SECCOMP, SELinux, and chroot work together can stand at odds and also can create a resilient environment that remains ahead of an evolving threat landscape, while still remaining usable and performing well.

2. Basic Concepts:

i) SECCOMP (Secure Computing Mode):

SECCOMP is a security feature for the Linux kernel that limits what system calls a process may use. It has, by default, the two following modes:

- Strict Mode: Allows only exit(), sigreturn(), read(), and write() syscalls, blocking all others.
- Filter Mode (seccomp-bpf): Uses Berkeley Packet Filter (BPF) rules to create custom allowlists/denylists of syscalls, enabling granular control. For example, Docker uses this to block risky syscalls like mount() or reboot() in containers.

The principle objective of SECCOMP is to limit the attack surface through the concept of least privilege. It is popular in a containerized environment (Kubernetes, Docker) to confine untrusted workloads.

ii) SELinux (Security-Enhanced Linux) :

SELinux performs mandatory access control (MAC) on the system and is built into the Linux kernel. It is a type of security labeling system developed by the NSA that assigns labels (security contexts) to files and processes and network port, for controlling their access, restrictions among processes, and controlling of communications among them. Key features include:

- Type Enforcement: Rules specify which processes (e.g., httpd_t) can access resources (e.g., httpd_sys_content_t).
- Role-Based Access Control (RBAC): Restricts users and roles to predefined privileges.
- Default-Deny Policy: Blocks all actions unless explicitly permitted by the policy.

SELinux is essential for securing servers, segregating multi-tenant environments, and meeting regulatory requirements such as HIPAA.

iii) chroot (Change Root) :

Chroot prevents processes from accessing files outside their directory by simulating the default directory as a subdirectory. Although frequently referred to as a “jail” it is not a secure mechanism for restricting access; rather it consists of a filesystem chroot isolation capability.

Key characteristics:

- Use Cases: The testing of legacy software, system restoration and dependency management.
- Limitations: chroot is bypassable for root and doesn't confine syscalls or kernel interactions.

3. Comparative Analysis :

Feature	SECCOMP	SELinux	chroot
Focus	Syscall filtering	Mandatory access control (MAC)	Filesystem isolation
Granularity	Process-level syscall restrictions	Resource-level (files, ports, etc.)	Directory-level isolation
Security Strength	High (prevents kernel exploits)	Very high (comprehensive policies)	Low (no kernel-level controls)
Complexity	Moderate (BPF rules)	High (policy configuration)	Low
Performance Overhead	Minimal	Moderate	Minimal
Common Use Cases	Containers, browsers (Chrome/Chromium)	Servers, compliance-critical systems	Testing, recovery environments

4. Use Cases :

SECCOMP

1. Container Security: Docker and Kubernetes block containers from dangerous syscalls (e.g., clone() for creating namespaces) by the default SECCOMP profiles.

2. Browser Sandboxing: Chrome/Chromium uses SECCOMP to limit the Adobe Flash and renderer processes.

3. Systemd Services: It is used in Linux services such as OpenSSH to restrict post-authentication operations.

SELinux

1. Web Server Protection: This restricts Apache (httpd_t) to /var/www/html and does not allow it to access /tmp or databases.

2. Multi-Level Security (MLS): The governments systems are using SELinux as means of keeping data confidential (for examples, if data is classified or unclassified).

3. Android: Implements app sandboxing and blocks root access.

chroot

1. Legacy Software Testing: Run old apps in an isolated environment on the host OS without affecting it.

2. System Recovery: If error occurs, it can be just recovered by booting off of a live CD, chroot and then do whatever we want from our live CD.

3. Build Environments: Isolate the build process from its environment, guarding against complex path-related dependency conflicts (typical in package builds) and ensure a known and static environment as much as possible.

5. Combined Use for Defense-in-Depth :

These new interlock mechanisms are also placed over one another in many modern systems:

- For example, a container might use chroot to isolate the filesystem, SECCOMP to prevent unwanted syscall invocations, and SELinux to apply MAC policy.
- Example: Kubernetes' seccompProfile: RuntimeDefault and SELinux labels for multi-tenancy security.

Conclusion :

- SECCOMP is really good when it comes to syscall-level sandboxing for containers and untrusted application.
- SELinux is a full-featured MAC targeted towards enterprise and compliance-based systems.
- chroot is a crude method for basic isolation that doesn't provide any strong security guarantees.

For maximum protection, combine these mechanisms with other practices (e.g., network policies, AppArmor) to address diverse threat vectors.

References :

1. Wikipedia contributors. (2025, February 19). *Seccomp*. https://en.wikipedia.org/wiki/Seccomp

2. Roshan, I. (2025, January 29). Understanding SeCComp — A comprehensive guide - Google Cloud - Community - Medium. *Medium*. https://medium.com/google-cloud/understanding-seccomp-a-comprehensive-guide-b1c519167df3

3. *What is SELinux?* (n.d.). https://www.redhat.com/en/topics/linux/what-is-selinux

4. Wikipedia contributors. (2025, May 11). *Chroot*. https://en.wikipedia.org/wiki/Chroot

5. *Chroot Simplified: A beginner's guide to secured Linux sandboxing*. (2023, May 28). https://www.lenovo.com/us/en/glossary/what-is-chroot/

6. *"Seccomp security profiles for Docker."* (2024, December 17). Docker Documentation. https://docs.docker.com/engine/security/seccomp/

7. ARMO. (2024, September 26). *What is Seccomp in Kubernetes?* https://www.armosec.io/glossary/seccomp/

8. Wikipedia contributors. (2025, April 3). *Security-Enhanced Linux*. Wikipedia. https://en.wikipedia.org/wiki/Security-Enhanced_Linux

9. *Chapter 1. Getting started with SELinux | Red Hat Product Documentation*. (n.d.). https://docs.redhat.com/en/documentation/red_hat_enterprise_linux/8/html/using_selinux/getting-started-with-selinux_using-selinux#getting-started-with-selinux_using-selinux

THE END