System Protection Mechanisms: SECCOMP, SELinux, and chroot

System protection functions in modern operating systems help keep the system safe, deny unauthorized access, and avoid threats from harming the computer. The report explains, compares, and explains how to put into action these three mechanisms: SECCOMP, SELinux, and chroot.

1. Basic Idea Behind the Mechanisms

SECCOMP (Secure Computing Mode):

SECCOMP is a part of the Linux kernel that helps control what tasks a program can do by preventing it from making certain system calls. It works like a filter that decides which system calls can go through and which ones get stopped, all based on certain set rules. There are two main modes: SECCOMP Strict Mode lets apps run just the most basic syscalls like reading, writing, quitting, and responding to signals, while SECCOMP-BPF lets admins set up more detailed rules to control which syscalls can be used. This helps keep apps safer by only letting them use safe functions, which stops them from getting access to things that could harm the system.

SELinux (Security-Enhanced Linux):

SELinux is a security part of the Linux kernel that enforces Mandatory Access Control (MAC). Unlike DAC, where the user can decide who has access, the access rules in SELinux come from the system and cannot be altered by most regular users. It manages who has access to files, devices, or any system resources by using labels and specific rules in its policies.

chroot (Change Root):

The chroot mechanism changes where the root directory points for a process and all the programs that come from it. After moving the file to the new root, access to outside data gets blocked, preventing contacts with the rest of the file system. It sets up an isolated box often to safely test or use untrusted programs.

2. Relative Comparison

Feature	**SECCOMP**	**SELinux**	**chroot**
Type	System Call Filtering	Mandatory Access Control	Filesystem Isolation
Granularity	Fine-grained (per syscall)	Very Fine (per user, process)	Coarse (directory-level)
Setup Complexity	Moderate	High (complex policy setup)	Simple
Root Privilege	Required for enabling/filtering	Required for policy management	Required
Isolation Level	Medium	High	Low
Common Use Cases	Sandboxing	Enterprise-level security	Safe testing, legacy app running
Limitations	Syscall-level only	High complexity and performance	Not a true sandbox, can be escaped

3. Use Cases

SECCOMP:

- Used in container environments like Docker to restrict container processes from accessing harmful syscalls.
- Applied in browsers (e.g., Chromium) to sandbox rendering processes.
- Helps in securing system daemons and custom applications with known syscall patterns.

SELinux:

- Protects sensitive files and services (like web servers, databases) from unauthorized access.
- Used in government, military, and enterprise servers for strict security enforcement.
- Enables the auditing of access events to track suspicious activity.

chroot:

- Used in web hosting environments to isolate different user directories.
- Helps in recovery tasks by booting into a different root file system.
- Suitable for running old or legacy applications that need a separate environment.

Conclusion

All these system parts are crucial for protecting Linux systems, but they act in different ways. With SECCOMp, you can restrict what a process can do with a syscall, while SELinux can control which processes can access which files, and chroot keeps each process limited to its own file system. Choosing which tools to use depends on how much security is needed and what the systems are used for.