What is PATH hijacking?

PATH hijacking is privilege escalation that exploits how Linux finds executables. When a privileged program (a SUID binary, a sudo-allowed command, or a root cron job) calls another command by name rather than by full path, the system searches the directories in the PATH variable in order. If an attacker can put a malicious binary of that name in a directory searched first, the privileged program runs the attacker's code with its privileges. The fix is using absolute paths in privileged programs.

HOW IT WORKS

01 The abuse and payload

The attacker finds a privileged program that calls a binary by name, then hijacks the lookup:

- Identify a SUID binary or sudo command that runs another command relatively (inspect with `strings`/`ltrace`).
- Create a malicious binary named like the called command: `echo '/bin/bash -p' > /tmp/service && chmod +x /tmp/service`
- Put the attacker directory first in PATH and run the privileged program: `export PATH=/tmp:$PATH` then trigger it.
- The privileged program finds `/tmp/service` first and runs the attacker's shell as root.

Documented techniques shown for defenders.

> THE ROOT CAUSE
>
> *PATH hijacking only works when a privileged program calls a command by name. Using absolute paths (-* `/usr/sbin/service`*) in privileged programs removes it entirely.*

HOW TO DEFEND

- Use absolute paths for every command inside SUID binaries, sudo-allowed scripts, and root cron jobs.
- Set a safe, fixed PATH in privileged scripts rather than inheriting the user's.
- Use `secure_path` in sudoers so sudo ignores the user's PATH.
- Avoid writable directories early in PATH, and never put . (current directory) in PATH.
- Review privileged programs for relative command calls.

SOURCES

[1] MITRE ATT&CK: Privilege Escalation (TA0004)

[2] NIST SP 800-115 Technical Guide to Security Testing

[3] Linux man-pages: setuid(2)

Find the privilege-escalation paths before an attacker does.

securelayer7.net/learn/privilege-escalation/what-is-path-hijacking

Open online