Abuse Elevation Control Mechanism: Setuid and Setgid

Other sub-techniques of Abuse Elevation Control Mechanism (6)

ID: T1548.001
Sub-technique of: T1548
ⓘ Tactics: Privilege Escalation, Defense Evasion
ⓘ Platforms: Linux, macOS
ⓘ Permissions Required: User
Version: 1.1
Created: 30 January 2020
Last Modified: 15 March 2023

Version Permalink

An adversary may abuse configurations where an application has the setuid or setgid bits set in order to get code running in a different (and possibly more privileged) user's context. On Linux or macOS, when the setuid or setgid bits are set for an application binary, the application will run with the privileges of the owning user or group respectively.[1] Normally an application is run in the current user's context, regardless of which user or group owns the application. However, there are instances where programs need to be executed in an elevated context to function properly, but the user running them may not have the specific required privileges.

Instead of creating an entry in the sudoers file, which must be done by root, any user can specify the setuid or setgid flag to be set for their own applications (i.e. Linux and Mac File and Directory Permissions Modification). The `chmod` command can set these bits with bitmasking, `chmod 4777 [file]` or via shorthand naming, `chmod u+s [file]`. This will enable the setuid bit. To enable the setgid bit, `chmod 2775` and `chmod g+s` can be used.

Adversaries can use this mechanism on their own malware to make sure they're able to execute in elevated contexts in the future.[2] This abuse is often part of a "shell escape" or other actions to bypass an execution environment with restricted permissions.

Alternatively, adversaries may choose to find and target vulnerable binaries with the setuid or setgid bits already enabled (i.e. File and Directory Discovery). The setuid and setguid bits are indicated with an "s" instead of an "x" when viewing a file's attributes via `ls -l`. The `find` command can also be used to search for such files. For example, `find / -perm +4000 2>/dev/null` can be used to find files with setuid set and `find / -perm +2000 2>/dev/null` may be used for setgid. Binaries that have these bits set may then be abused by adversaries.
[3]

Procedure Examples

ID	Name	Description
S0401	Exaramel for Linux	Exaramel for Linux can execute commands with high privileges via a specific binary with setuid functionality.[4]
S0276	Keydnap	Keydnap adds the setuid flag to a binary so it can easily elevate in the future.[2]

Mitigations

ID	Mitigation	Description