Enterprise ~

Pricing

Open Source ~



> T1036

Atomic Red Team doc generat... Generated docs from job=generate-d... 819934c · 2 years ago

Preview Code Blame 225 lines (119 loc) · 6.39 KB

Raw L L E

T1548.001 - Setuid and Setgid

Description from ATT&CK

Notifications

Q

Y Fork 2.8k

Sign in

Sign up

Star 9.7k

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.(Citation: setuid man page) 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

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

[file] . This will enable the setuid bit. To enable the setgit bit, chmod 2775 and

Alternatively, adversaries may choose to find and target vulnerable binaries with the setuid or setgid bits already enabled (i.e. <u>File and Directory Discovery</u>). The setuid and setguid bits are indicated with an "s" instead of an "x" when viewing a file's attributes via 1s -1. 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.(Citation: GTFOBins Suid)

## **Atomic Tests**

chmod g+s can be used.

- Atomic Test #1 Make and modify binary from C source
- Atomic Test #2 Set a SetUID flag on file
- Atomic Test #3 Set a SetGID flag on file
- Atomic Test #4 Make and modify capabilities of a binary
- Atomic Test #5 Provide the SetUID capability to a file

> T1037.001

> T1037.002

> T1037.004

> **T1037.005** 

> **T1039** 

> **T1040** 

# Atomic Test #1 - Make and modify binary from C source

Make, change owner, and change file attributes on a C source code file

Supported Platforms: macOS, Linux

auto\_generated\_guid: 896dfe97-ae43-4101-8e96-9a7996555d80

#### Inputs:

Name	Description	Туре	Default Value
payload	hello.c payload	Path	PathToAtomicsFolder/T1548.001/src/hello.c

### Attack Commands: Run with sh! Elevation Required (e.g. root or admin)

cp #{payload} /tmp/hello.c
sudo chown root /tmp/hello.c
sudo make /tmp/hello
sudo chown root /tmp/hello
sudo chmod u+s /tmp/hello
/tmp/hello

#### Cleanup Commands:

sudo rm /tmp/hello
sudo rm /tmp/hello.c

Q

# Atomic Test #2 - Set a SetUID flag on file

This test sets the SetUID flag on a file in Linux and macOS.

Supported Platforms: macOS, Linux

auto\_generated\_guid: 759055b3-3885-4582-a8ec-c00c9d64dd79

### Inputs:

	Name	Description	Туре	Default Value
	file_to_setuid	Path of file to set SetUID flag	Path	/tmp/evilBinary

### Attack Commands: Run with sh! Elevation Required (e.g. root or admin)

sudo touch #{file\_to\_setuid}
sudo chown root #{file\_to\_setuid}
sudo chmod u+s #{file\_to\_setuid}

### Cleanup Commands:

sudo rm #{file\_to\_setuid}

## Atomic Test #3 - Set a SetGID flag on file

This test sets the SetGID flag on a file in Linux and macOS.

Supported Platforms: macOS, Linux

auto\_generated\_guid: db55f666-7cba-46c6-9fe6-205a05c3242c

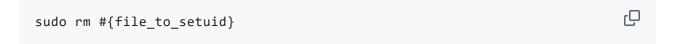
#### Inputs:

Name	Description	Туре	Default Value
file_to_setuid	Path of file to set SetGID flag	Path	/tmp/evilBinary

#### Attack Commands: Run with sh! Elevation Required (e.g. root or admin)

```
sudo touch #{file_to_setuid}
sudo chown root #{file_to_setuid}
sudo chmod g+s #{file_to_setuid}
```

#### **Cleanup Commands:**



# Atomic Test #4 - Make and modify capabilities of a binary

Make and modify <u>capabilities</u> of a C source code file. The binary doesn't have to modify the UID, but the binary is given the capability to arbitrarily modify it at any time with setuid(0). Without being owned by root, the binary can set the UID to 0.

**Supported Platforms:** Linux

auto\_generated\_guid: db53959c-207d-4000-9e7a-cd8eb417e072

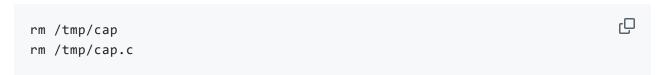
#### Inputs:

Name	Description	Туре	Default Value
payload	cap.c payload	Path	PathToAtomicsFolder/T1548.001/src/cap.c

### Attack Commands: Run with sh! Elevation Required (e.g. root or admin)

```
cp #{payload} /tmp/cap.c
make /tmp/cap
sudo setcap cap_setuid=ep /tmp/cap
/tmp/cap
```

### **Cleanup Commands:**



# Atomic Test #5 - Provide the SetUID capability to a file

This test gives a file the capability to set UID without using flags.

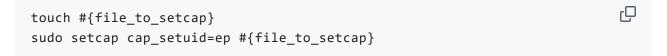
Supported Platforms: Linux

auto\_generated\_guid: 1ac3272f-9bcf-443a-9888-4b1d3de785c1

Inputs:

Name	Description	Туре	Default Value
file_to_setcap	Path of file to provide the SetUID capability	Path	/tmp/evilBinary

### Attack Commands: Run with sh! Elevation Required (e.g. root or admin)



Q

# Cleanup Commands:

rm #{file\_to\_setcap}