T1547.015 - Boot or Logon Autostart Execution: Login Items

Description from ATT&CK

> Adversaries may add login items to execute upon user login to gain persistence or escalate privileges. Login items are applications, documents, folders, or server connections that are automatically launched when a user logs in.(Citation: Open Login Items Apple) Login items can be added via a shared file list or Service Management Framework.(Citation: Adding Login Items) Shared file list login items can be set using scripting languages such as [AppleScript] (https://attack.mitre.org/techniques/T1059/002), whereas the Service Management Framework uses the API call `SMLoginItemSetEnabled`.
> Login items installed using the Service Management Framework leverage `launchd`, are not visible in the System Preferences, and can only be removed by the application that created them.(Citation: Adding Login Items)(Citation: SMLoginItemSetEnabled Schroeder 2013) Login items created using a shared file list are visible in System Preferences, can hide the application when it launches, and are executed through LaunchServices, not launchd, to open applications, documents, or URLs without using Finder.(Citation: Launch Services Apple Developer) Users and applications use login items to configure their user environment to launch commonly used services or applications, such as email, chat, and music applications.
>
> Adversaries can utilize AppleScript and Native API calls to create a login item to spawn malicious executables.(Citation: ELC Running at startup) Prior to version 10.5 on macOS, adversaries can add login items by using AppleScript to send an Apple events to the "System Events" process, which has an AppleScript dictionary for manipulating login items.(Citation: Login Items AE) Adversaries can use a command such as `tell application "System Events" to make login item at end with properties /path/to/executable`.(Citation: Startup Items Eclectic)(Citation: hexed osx.dok analysis 2019)(Citation: Add List Remove Login Items Apple Script) This command adds the path of the malicious executable to the login item file list located in `~/Library/Application`

> malware 2017)(Citation: CheckPoint Dok)(Citation: objsee netwire backdoor 2019)

Atomic Tests

- Atomic Test #1 - Persistence by modifying Windows Terminal profile
- Atomic Test #2 - Add macOS LoginItem using Applescript

T1003.001
T1003.002
T1003.003
T1003.004
T1003.005
T1003.006
T1003.007
T1003.008
T1003
T1006
T1007
T1010
T1012
T1014
T1016
T1018
T1020
T1021.001
T1021.002
T1021.003
T1021.006
T1027.001
T1027.002
T1027.004
T1027.006
T1027
T1030
T1033
T1036.003
T1036.004
T1036.005
T1036.006
T1036
T1037.001
T1037.002
T1037.004
T1037.005
T1039

Atomic Test #1 - Persistence by modifying Windows Terminal profile

Modify Windows Terminal settings.json file to gain persistence. Twitter Post

Supported Platforms: Windows

auto_generated_guid: ec5d76ef-82fe-48da-b931-bdb25a62bc65

Inputs:

Name	Description	Type	
calculator	Test program used to imitate a maliciously called program.	string	calculator.exe
settings_json_def	Default file for Windows Terminal to replace the default profile with a backdoor to call another program.	path	~\AppData\Local\Packages\Microsoft.Win
settings_json_tmp	Temp file for Windows Terminal.	path	~\AppData\Local\Temp\settings.json
wt_exe	Windows Terminal executable.	path	~\AppData\Local\Microsoft\WindowsApps

Attack Commands: Run with `powershell`!

```
mv #{settings_json_def} #{settings_json_tmp}
Invoke-WebRequest "https://github.com/redcanaryco/atomic-red-team/blob/m
wt.exe
```

Cleanup Commands:

```
mv -Force #{settings_json_tmp} #{settings_json_def}
taskkill /F /IM "#{calculator}" > $null
```

Dependencies: Run with `powershell`!

Description: Windows Terminal must be installed

Check Prereq Commands:

```
if (Test-Path #{wt_exe}) {exit 0} else {exit 1}
```

Get Prereq Commands:

```
$(rm ~\AppData\Local\Packages\Microsoft.DesktopAppInstaller_8wekyb3d8bbw
```

Atomic Test #2 - Add macOS LoginItem using Applescript

Runs osascript on a file to create new LoginItem for current user. NOTE: Will popup dialog prompting user to Allow or Deny Terminal.app to control "System Events" Therefore, it can't be automated until the TCC is granted. The login item launches Safari.app when user logs in, but there is a cleanup script to remove it as well. In addition to the `osascript` Process Events, file modification events to `/Users/*/Library/Application Support/com.apple.backgroundtaskmanagementagent/backgrounditems.btm` should be seen.

Supported Platforms: macOS

auto_generated_guid: 716e756a-607b-41f3-8204-b214baf37c1d

Inputs:

Name	Description	Type	Default Value
scriptfile	path to Applescript source to add Safari LoginItem.	string	PathToAtomicsFolder/T1547.015/src/add_logi
cleanup_script	path to Applescript source to delete Safari LoginItem.	string	PathToAtomicsFolder/T1547.015/src/remove_

Attack Commands: Run with `bash`!

```
osascript #{scriptfile}
```

Cleanup Commands:

```
osascript #{cleanup_script}
```