

NETMANAGEIT

Intelligence Report

Analysis and Detection of STEADY#URSA Attack

Campaign Targeting

Ukraine Military Dropping

New Covert SUBTLE-PAWS

PowerShell Backdoor

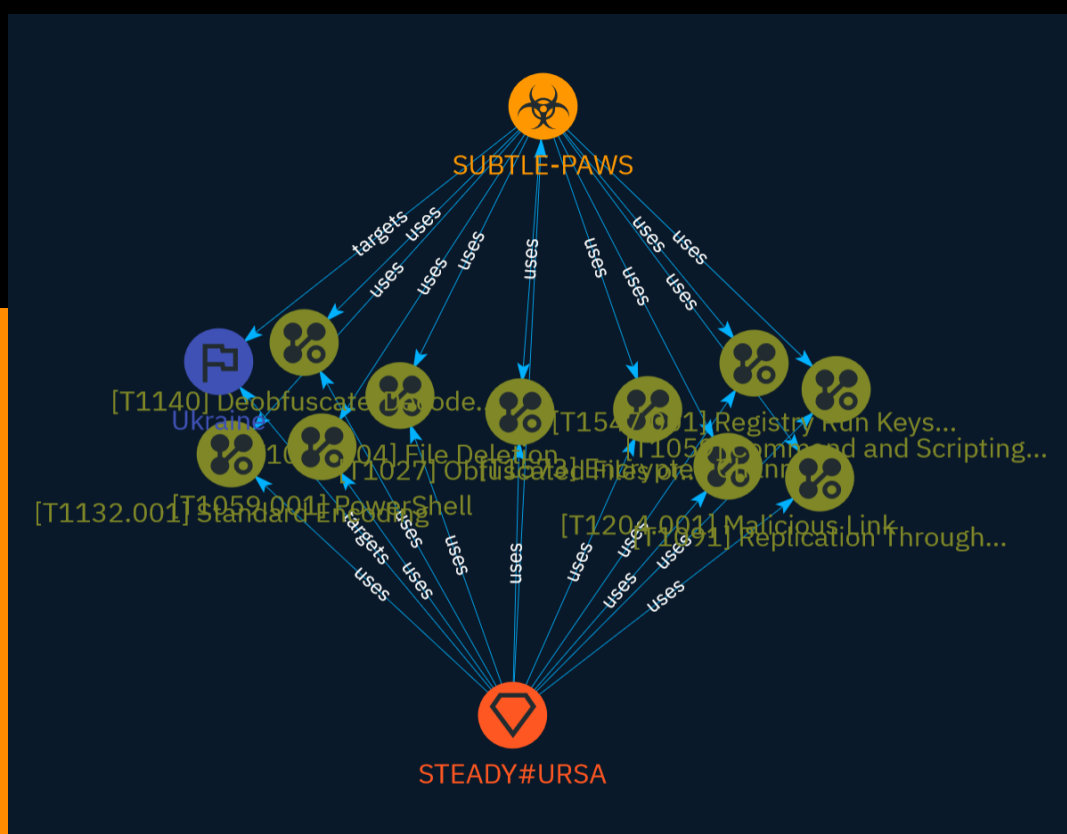


Table of contents

Overview

● Description	3
● Confidence	3
● Content	4

Entities

● Malware	5
● Intrusion-Set	6
● Attack-Pattern	7
● Country	22

External References

● External References	23
-----------------------	----

Overview

Description

An interesting campaign leveraging a new SUBTLE-PAWS PowerShell-based backdoor has been identified targeting Ukraine which follows stealthy tactics to evade detection and spreads by infecting USB drives. The malicious payload is delivered through compressed files, possibly through phishing emails. The exploitation chain involves the target executing a malicious shortcut file which loads and executes a new PowerShell backdoor payload code. While the initial execution is trivial, some complex late-stage execution and persistence methods are used. The backdoor establishes C2 communication, gathers system info, and spreads via USB drives. It uses obfuscation, randomization, and environment awareness for stealth.

Confidence

This value represents the confidence in the correctness of the data contained within this report.

15 / 100

Content

N/A

Malware

Name

SUBTLE-PAWS

Name

SUBTLE-PAWS

Intrusion-Set

Name

STEADY#URSA

Name

STEADY#URSA

Attack-Pattern

Name

Malicious Link

ID

T1204.001

Description

An adversary may rely upon a user clicking a malicious link in order to gain execution. Users may be subjected to social engineering to get them to click on a link that will lead to code execution. This user action will typically be observed as follow-on behavior from [Spearphishing Link](<https://attack.mitre.org/techniques/T1566/002>). Clicking on a link may also lead to other execution techniques such as exploitation of a browser or application vulnerability via [Exploitation for Client Execution](<https://attack.mitre.org/techniques/T1203>). Links may also lead users to download files that require execution via [Malicious File](<https://attack.mitre.org/techniques/T1204/002>).

Name

File Deletion

ID

T1070.004

Description

Adversaries may delete files left behind by the actions of their intrusion activity. Malware, tools, or other non-native files dropped or created on a system by an adversary (ex: [Ingress Tool Transfer](https://attack.mitre.org/techniques/T1105)) may leave traces to indicate to what was done within a network and how. Removal of these files can occur during an intrusion, or as part of a post-intrusion process to minimize the adversary's footprint. There are tools available from the host operating system to perform cleanup, but adversaries may use other tools as well.(Citation: Microsoft SDelete July 2016) Examples of built-in [Command and Scripting Interpreter](https://attack.mitre.org/techniques/T1059) functions include ``del`` on Windows and ``rm`` or ``unlink`` on Linux and macOS.

Name

Encrypted Channel

ID

T1573

Description

Adversaries may employ a known encryption algorithm to conceal command and control traffic rather than relying on any inherent protections provided by a communication protocol. Despite the use of a secure algorithm, these implementations may be vulnerable to reverse engineering if secret keys are encoded and/or generated within malware samples/configuration files.

Name

Registry Run Keys / Startup Folder

ID

T1547.001

Description

Adversaries may achieve persistence by adding a program to a startup folder or referencing it with a Registry run key. Adding an entry to the "run keys" in the Registry or startup folder will cause the program referenced to be executed when a user logs in. (Citation: Microsoft Run Key) These programs will be executed under the context of the user and will have the account's associated permissions level. The following run keys are created by default on Windows systems: *

`HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run` *

`HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce` *

`HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run` *

`HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunOnce` Run keys may exist under multiple hives.(Citation: Microsoft Wow6432Node 2018)(Citation:

Malwarebytes Wow6432Node 2016) The

`HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunOnceEx` is also available but is not created by default on Windows Vista and newer. Registry run key

entries can reference programs directly or list them as a dependency.(Citation: Microsoft Run Key) For example, it is possible to load a DLL at logon using a "Depend" key with

RunOnceEx: `reg add

HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnceEx\0001\Depend /v 1 /d "C:\temp\evil[.].dll" (Citation: Oddvar Moe RunOnceEx Mar 2018) Placing a program within a

startup folder will also cause that program to execute when a user logs in. There is a startup folder location for individual user accounts as well as a system-wide startup folder that will be checked regardless of which user account logs in. The startup folder path for the current user is `C:\Users\[Username]\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup`. The startup folder path for all users is `C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup`. The following Registry

keys can be used to set startup folder items for persistence: *

`HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders` *

`HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders` *

`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders` *

`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders` The following Registry keys can control automatic startup of services during boot: *

`HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServicesOnce` *

`HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunServicesOnce` *

`HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServices` *

`HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunServices` Using policy settings to specify startup programs creates corresponding values in either of two Registry keys: *

`HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run` *

HKKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run` Programs listed in the load value of the registry key

HKKEY_CURRENT_USER\Software\Microsoft\Windows NT\CurrentVersion\Windows` run automatically for the currently logged-on user. By default, the multistring `BootExecute` value of the registry key

HKKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Session Manager` is set to `autocheck autochk *`. This value causes Windows, at startup, to check the file-system integrity of the hard disks if the system has been shut down abnormally. Adversaries can add other programs or processes to this registry value which will automatically launch at boot. Adversaries can use these configuration locations to execute malware, such as remote access tools, to maintain persistence through system reboots. Adversaries may also use [Masquerading](https://attack.mitre.org/techniques/T1036) to make the Registry entries look as if they are associated with legitimate programs.

Name

PowerShell

ID

T1059.001

Description

Adversaries may abuse PowerShell commands and scripts for execution. PowerShell is a powerful interactive command-line interface and scripting environment included in the Windows operating system.(Citation: TechNet PowerShell) Adversaries can use PowerShell to perform a number of actions, including discovery of information and execution of code. Examples include the `Start-Process` cmdlet which can be used to run an executable and the `Invoke-Command` cmdlet which runs a command locally or on a remote computer (though administrator permissions are required to use PowerShell to connect to remote systems). PowerShell may also be used to download and run executables from the Internet, which can be executed from disk or in memory without touching disk. A number of PowerShell-based offensive testing tools are available, including [Empire](https://attack.mitre.org/software/S0363), [PowerSploit](https://attack.mitre.org/software/S0194), [PoshC2](https://attack.mitre.org/software/S0378), and PSAttack.(Citation: Github PSAttack) PowerShell commands/scripts can also be executed without directly invoking the `powershell.exe` binary through interfaces to PowerShell's underlying `System.Management.Automation` assembly DLL exposed through the .NET framework and

Windows Common Language Interface (CLI).(Citation: Sixdub PowerPick Jan 2016)(Citation: SilentBreak Offensive PS Dec 2015)(Citation: Microsoft PSfromCsharp APR 2014)

Name

Command and Scripting Interpreter

ID

T1059

Description

Adversaries may abuse command and script interpreters to execute commands, scripts, or binaries. These interfaces and languages provide ways of interacting with computer systems and are a common feature across many different platforms. Most systems come with some built-in command-line interface and scripting capabilities, for example, macOS and Linux distributions include some flavor of [Unix Shell](<https://attack.mitre.org/techniques/T1059/004>) while Windows installations include the [Windows Command Shell](<https://attack.mitre.org/techniques/T1059/003>) and [PowerShell](<https://attack.mitre.org/techniques/T1059/001>). There are also cross-platform interpreters such as [Python](<https://attack.mitre.org/techniques/T1059/006>), as well as those commonly associated with client applications such as [JavaScript](<https://attack.mitre.org/techniques/T1059/007>) and [Visual Basic](<https://attack.mitre.org/techniques/T1059/005>). Adversaries may abuse these technologies in various ways as a means of executing arbitrary commands. Commands and scripts can be embedded in [Initial Access](<https://attack.mitre.org/tactics/TA0001>) payloads delivered to victims as lure documents or as secondary payloads downloaded from an existing C2. Adversaries may also execute commands through interactive terminals/shells, as well as utilize various [Remote Services](<https://attack.mitre.org/techniques/T1021>) in order to achieve remote Execution. (Citation: Powershell Remote Commands)(Citation: Cisco IOS Software Integrity Assurance - Command History)(Citation: Remote Shell Execution in Python)

Name

Obfuscated Files or Information

ID

T1027

Description

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](https://attack.mitre.org/techniques/T1140) for [User Execution](https://attack.mitre.org/techniques/T1204). 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 use 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 abuse [Command Obfuscation](https://attack.mitre.org/techniques/T1027/010) to obscure commands executed from payloads or directly via [Command and Scripting Interpreter](https://attack.mitre.org/techniques/T1059). 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)

Name

Deobfuscate/Decode Files or Information

ID

T1140

Description

Adversaries may use [Obfuscated Files or Information](https://attack.mitre.org/techniques/T1027) to hide artifacts of an intrusion from analysis. They may require separate mechanisms to decode or deobfuscate that information depending on how they

intend to use it. Methods for doing that include built-in functionality of malware or by using utilities present on the system. One such example is the use of [certutil](https://attack.mitre.org/software/S0160) to decode a remote access tool portable executable file that has been hidden inside a certificate file.(Citation: Malwarebytes Targeted Attack against Saudi Arabia) Another example is using the Windows `copy /b` command to reassemble binary fragments into a malicious payload.(Citation: Carbon Black Obfuscation Sept 2016) Sometimes a user's action may be required to open it for deobfuscation or decryption as part of [User Execution](https://attack.mitre.org/techniques/T1204). 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)

Name

Replication Through Removable Media

ID

T1091

Description

Adversaries may move onto systems, possibly those on disconnected or air-gapped networks, by copying malware to removable media and taking advantage of Autorun features when the media is inserted into a system and executes. In the case of Lateral Movement, this may occur through modification of executable files stored on removable media or by copying malware and renaming it to look like a legitimate file to trick users into executing it on a separate system. In the case of Initial Access, this may occur through manual manipulation of the media, modification of systems used to initially format the media, or modification to the media's firmware itself. Mobile devices may also be used to infect PCs with malware if connected via USB.(Citation: Exploiting Smartphone USB) This infection may be achieved using devices (Android, iOS, etc.) and, in some instances, USB charging cables.(Citation: Windows Malware Infecting Android)(Citation: iPhone Charging Cable Hack) For example, when a smartphone is connected to a system, it may appear to be mounted similar to a USB-connected disk drive. If malware that is compatible with the connected system is on the mobile device, the malware could infect the machine (especially if Autorun features are enabled).

Name

Standard Encoding

ID

T1132.001

Description

Adversaries may encode data with a standard data encoding system to make the content of command and control traffic more difficult to detect. Command and control (C2) information can be encoded using a standard data encoding system that adheres to existing protocol specifications. Common data encoding schemes include ASCII, Unicode, hexadecimal, Base64, and MIME.(Citation: Wikipedia Binary-to-text Encoding)(Citation: Wikipedia Character Encoding) Some data encoding systems may also result in data compression, such as gzip.

Name

Malicious Link

ID

T1204.001

Description

An adversary may rely upon a user clicking a malicious link in order to gain execution. Users may be subjected to social engineering to get them to click on a link that will lead to code execution. This user action will typically be observed as follow-on behavior from [Spearphishing Link](<https://attack.mitre.org/techniques/T1566/002>). Clicking on a link may also lead to other execution techniques such as exploitation of a browser or application vulnerability via [Exploitation for Client Execution](<https://attack.mitre.org/techniques/T1203>). Links may also lead users to download files that require execution via [Malicious File](<https://attack.mitre.org/techniques/T1204/002>).

Name

File Deletion

ID

T1070.004

Description

Adversaries may delete files left behind by the actions of their intrusion activity. Malware, tools, or other non-native files dropped or created on a system by an adversary (ex: [Ingress Tool Transfer](<https://attack.mitre.org/techniques/T1105>)) may leave traces to indicate to what was done within a network and how. Removal of these files can occur during an intrusion, or as part of a post-intrusion process to minimize the adversary's footprint. There are tools available from the host operating system to perform cleanup, but adversaries may use other tools as well.(Citation: Microsoft SDelete July 2016) Examples of built-in [Command and Scripting Interpreter](<https://attack.mitre.org/techniques/T1059>) functions include ``del`` on Windows and ``rm`` or ``unlink`` on Linux and macOS.

Name

Encrypted Channel

ID

T1573

Description

Adversaries may employ a known encryption algorithm to conceal command and control traffic rather than relying on any inherent protections provided by a communication protocol. Despite the use of a secure algorithm, these implementations may be vulnerable to reverse engineering if secret keys are encoded and/or generated within malware samples/configuration files.

Name

Registry Run Keys / Startup Folder

ID

T1547.001

Description

Adversaries may achieve persistence by adding a program to a startup folder or referencing it with a Registry run key. Adding an entry to the "run keys" in the Registry or startup folder will cause the program referenced to be executed when a user logs in. (Citation: Microsoft Run Key) These programs will be executed under the context of the user and will have the account's associated permissions level. The following run keys are created by default on Windows systems: *

```
`HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run` *
```

```
`HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce` *
```

```
`HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run` *
```

```
`HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunOnce`
```

Run keys may exist under multiple hives.(Citation: Microsoft Wow6432Node 2018)(Citation: Malwarebytes Wow6432Node 2016) The

```
`HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunOnceEx`
```

is also available but is not created by default on Windows Vista and newer. Registry run key entries can reference programs directly or list them as a dependency.(Citation: Microsoft Run Key) For example, it is possible to load a DLL at logon using a "Depend" key with RunOnceEx: ``reg add`

```
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnceEx\0001\Depend /v 1 /d "C:\temp\evil[.].dll"
```

(Citation: Oddvar Moe RunOnceEx Mar 2018) Placing a program within a startup folder will also cause that program to execute when a user logs in. There is a startup folder location for individual user accounts as well as a system-wide startup folder that will be checked regardless of which user account logs in. The startup folder path for the current user is ``C:\Users\[Username]\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup``. The startup folder path for all users is ``C:\ProgramData\Microsoft\Windows\Start Menu\Programs\StartUp``. The following Registry keys can be used to set startup folder items for persistence: *

```
`HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders` *
```

```
`HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders` *
```

```
`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders` *
```

```
`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\User
```

```
`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\User
```


Shell Folders` The following Registry keys can control automatic startup of services during boot: *

`HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServicesOnce` *

`HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunServicesOnce` *

`HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServices` *

`HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunServices` Using policy settings to specify startup programs creates corresponding values in either of two Registry keys: *

`HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run` *

`HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run` Programs listed in the load value of the registry key

`HKEY_CURRENT_USER\Software\Microsoft\Windows NT\CurrentVersion\Windows` run automatically for the currently logged-on user. By default, the multistring `BootExecute` value of the registry key

`HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Session Manager` is set to `autocheck autochk *`. This value causes Windows, at startup, to check the file-system integrity of the hard disks if the system has been shut down abnormally. Adversaries can add other programs or processes to this registry value which will automatically launch at boot. Adversaries can use these configuration locations to execute malware, such as remote access tools, to maintain persistence through system reboots. Adversaries may also use [Masquerading](https://attack.mitre.org/techniques/T1036) to make the Registry entries look as if they are associated with legitimate programs.

Name

PowerShell

ID

T1059.001

Description

Adversaries may abuse PowerShell commands and scripts for execution. PowerShell is a powerful interactive command-line interface and scripting environment included in the Windows operating system.(Citation: TechNet PowerShell) Adversaries can use PowerShell to perform a number of actions, including discovery of information and execution of code. Examples include the `Start-Process` cmdlet which can be used to run an executable and the `Invoke-Command` cmdlet which runs a command locally or on a remote computer (though administrator permissions are required to use PowerShell to connect to remote

systems). PowerShell may also be used to download and run executables from the Internet, which can be executed from disk or in memory without touching disk. A number of PowerShell-based offensive testing tools are available, including [Empire](<https://attack.mitre.org/software/S0363>), [PowerSploit](<https://attack.mitre.org/software/S0194>), [PoshC2](<https://attack.mitre.org/software/S0378>), and PSAttack.(Citation: Github PSAttack) PowerShell commands/scripts can also be executed without directly invoking the `powershell.exe` binary through interfaces to PowerShell's underlying `System.Management.Automation` assembly DLL exposed through the .NET framework and Windows Common Language Interface (CLI).(Citation: Sixdub PowerPick Jan 2016)(Citation: SilentBreak Offensive PS Dec 2015)(Citation: Microsoft PSfromCsharp APR 2014)

Name

Command and Scripting Interpreter

ID

T1059

Description

Adversaries may abuse command and script interpreters to execute commands, scripts, or binaries. These interfaces and languages provide ways of interacting with computer systems and are a common feature across many different platforms. Most systems come with some built-in command-line interface and scripting capabilities, for example, macOS and Linux distributions include some flavor of [Unix Shell](<https://attack.mitre.org/techniques/T1059/004>) while Windows installations include the [Windows Command Shell](<https://attack.mitre.org/techniques/T1059/003>) and [PowerShell](<https://attack.mitre.org/techniques/T1059/001>). There are also cross-platform interpreters such as [Python](<https://attack.mitre.org/techniques/T1059/006>), as well as those commonly associated with client applications such as [JavaScript](<https://attack.mitre.org/techniques/T1059/007>) and [Visual Basic](<https://attack.mitre.org/techniques/T1059/005>). Adversaries may abuse these technologies in various ways as a means of executing arbitrary commands. Commands and scripts can be embedded in [Initial Access](<https://attack.mitre.org/tactics/TA0001>) payloads delivered to victims as lure documents or as secondary payloads downloaded from an existing C2. Adversaries may also execute commands through interactive terminals/shells, as well as utilize various [Remote Services](<https://attack.mitre.org/techniques/T1021>) in order to achieve remote Execution. (Citation: Powershell Remote Commands)(Citation: Cisco IOS Software Integrity Assurance - Command History)(Citation: Remote Shell Execution in Python)

Name

Obfuscated Files or Information

ID

T1027

Description

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](<https://attack.mitre.org/techniques/T1140>) for [User Execution](<https://attack.mitre.org/techniques/T1204>). 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 use 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 abuse [Command Obfuscation](<https://attack.mitre.org/techniques/T1027/010>) to obscure commands executed from payloads or directly via [Command and Scripting Interpreter](<https://attack.mitre.org/techniques/T1059>). 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)

Name

Deobfuscate/Decode Files or Information

ID

T1140

Description

Adversaries may use [Obfuscated Files or Information](<https://attack.mitre.org/techniques/T1027>) to hide artifacts of an intrusion from analysis. They may require separate mechanisms to decode or deobfuscate that information depending on how they intend to use it. Methods for doing that include built-in functionality of malware or by using utilities present on the system. One such example is the use of [certutil](<https://attack.mitre.org/software/S0160>) to decode a remote access tool portable executable file that has been hidden inside a certificate file.(Citation: Malwarebytes Targeted Attack against Saudi Arabia) Another example is using the Windows `copy /b`` command to reassemble binary fragments into a malicious payload.(Citation: Carbon Black Obfuscation Sept 2016) Sometimes a user's action may be required to open it for deobfuscation or decryption as part of [User Execution](<https://attack.mitre.org/techniques/T1204>). 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)

Name

Replication Through Removable Media

ID

T1091

Description

Adversaries may move onto systems, possibly those on disconnected or air-gapped networks, by copying malware to removable media and taking advantage of Autorun features when the media is inserted into a system and executes. In the case of Lateral Movement, this may occur through modification of executable files stored on removable media or by copying malware and renaming it to look like a legitimate file to trick users into executing it on a separate system. In the case of Initial Access, this may occur through manual manipulation of the media, modification of systems used to initially format the media, or modification to the media's firmware itself. Mobile devices may also be used to infect PCs with malware if connected via USB.(Citation: Exploiting Smartphone USB) This infection may be achieved using devices (Android, iOS, etc.) and, in some instances, USB

charging cables.(Citation: Windows Malware Infecting Android)(Citation: iPhone Charging Cable Hack) For example, when a smartphone is connected to a system, it may appear to be mounted similar to a USB-connected disk drive. If malware that is compatible with the connected system is on the mobile device, the malware could infect the machine (especially if Autorun features are enabled).

Name

Standard Encoding

ID

T1132.001

Description

Adversaries may encode data with a standard data encoding system to make the content of command and control traffic more difficult to detect. Command and control (C2) information can be encoded using a standard data encoding system that adheres to existing protocol specifications. Common data encoding schemes include ASCII, Unicode, hexadecimal, Base64, and MIME.(Citation: Wikipedia Binary-to-text Encoding)(Citation: Wikipedia Character Encoding) Some data encoding systems may also result in data compression, such as gzip.

Country

Name

Ukraine

Name

Ukraine

External References

-
- <https://www.securonix.com/blog/security-advisory-steadyursa-attack-campaign-targets-ukraine-military/>
-
- <https://otx.alienvault.com/pulse/65bd06dae1221bc7703eb73b>