Black Basta Ransomware Campaign Drops Zbot, DarkGate, and Custom Malware

Tyler McGraw

Last updated at Fri, 17 Jan 2025 21:25:06 GMT

Executive Summary

Beginning in early October, Rapid7 has observed a resurgence of activity related to the ongoing social engineering campaign being conducted by Black Basta ransomware operators. Rapid7 initially reported the discovery of the novel social engineering campaign back in May, 2024, followed by an update in August 2024, when the operators updated their tactics and malware payloads and began sending lures via Microsoft Teams. Now, the procedures followed by the threat actors in the early stages of the social engineering attacks have been refined again, with new malware payloads, improved delivery, and increased defense evasion.

Overview

The social engineering attacks are still initiated in a similar manner. Users within the target environment will be email bombed by the threat actor, which is often achieved by signing up the user's email to

numerous mailing lists simultaneously. After the email bomb, the threat actor will reach out to the impacted users. Rapid7 has observed the initial contact still occurs primarily through usage of Microsoft Teams, by which the threat actor, as an external user, will attempt to call or message the impacted user to offer assistance. The account domains in use include both Azure/Entra tenant subdomains (e.g., username[@]tenantsubdomain[.]onmicrosoft[.]com) and custom domains (e.g., username[@]cofincafe[.]com).

In many cases, Rapid7 has observed that the threat actor will pretend to be a member of the target organization's help desk, support team, or otherwise present themself as IT staff. Below are examples of Microsoft Teams display names observed, by Rapid7, to be in use by operators. The display names may or may not be padded with whitespace characters. Rapid7 has also observed threat actors use a first and last name, as the chat display name and/or account username, to impersonate an IT staff member within the targeted organization.

Operator Chat Display Name
Help Desk
HELP DESK
Help Desk Manager
Technical Support
Administracion

If the user interacts with the lure, either by answering the call or messaging back, the threat actor will attempt to get the user to install or execute a remote management (RMM) tool, including, but not limited to, QuickAssist, AnyDesk, TeamViewer, Level, or ScreenConnect. Rapid7 has also observed attempts to leverage the OpenSSH client, a native Windows utility, to establish a reverse shell. In at least one instance, the threat actor shared a QR code with the targeted user. The purpose of the QR code is unconfirmed but appears to be an attempt to bypass MFA after stealing a user's credentials. The URL embedded within the QR code adheres to the following format: `hxxps://<company_name>[.]qr-<letter><number>[.]com`.

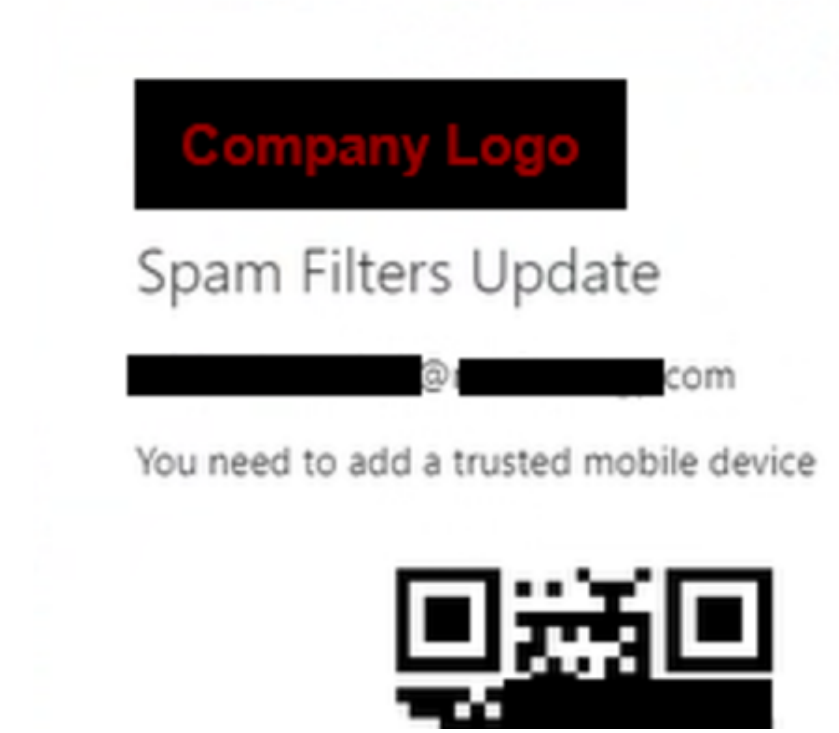

Figure 1. A QR code (obfuscation by Rapid7) sent by an operator.

In a majority of cases, Rapid7 has observed that the operator, after gaining access to the user's asset via RMM tool, will then attempt to download and execute additional malware payloads. In one case handled by Rapid7, the operator requested more time — potentially to hand off the access to another member of the group.

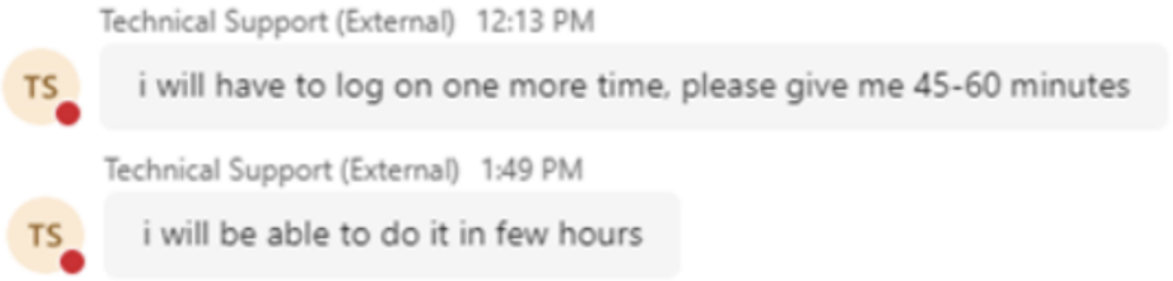

Figure 2. An operator stalls for time.

The payload delivery methods vary per case, but have included external compromised SharePoint instances, common file sharing websites, servers rented through hosting providers, or even direct upload to the compromised asset in the case of RMM tool remote control. In one case, the operator used the group's custom credential harvester to dump the user's credentials, the results for which were subsequently uploaded to a file sharing site — publicly exposing the stolen credentials. SharePoint has been used to distribute copies of AnyDesk portable, likely to circumvent security measures that would prevent the user from downloading it directly from `anydesk[.]com`. Such attempts have been blocked by web proxy in previous cases.

The overall goal following initial access appears to be the same: to quickly enumerate the environment and dump the user's credentials. When possible, operators will also still attempt to steal any available VPN configuration files. With the user's credentials, organization VPN information, and potential MFA bypass, it may be possible for them to authenticate directly to the target environment.

Rapid7 has observed usage of the same credential harvesting executable, previously reported as `AntiSpam.exe`, though it is now delivered in the form of a DLL and most commonly executed via rundll32.exe. Whereas before it was an unobfuscated .NET executable, the program is now commonly contained within a compiled 64-bit DLL loader. Rapid7 has analyzed at least one sample that has also been obfuscated using the group's custom packer. The newest versions of the credential harvester now save output to the file `123.txt` in the user's `%TEMP%` directory, an update from the previous `qwertyuio.txt` file, though versions of the DLL distributed earlier in the campaign would still output to the previous file.

Update

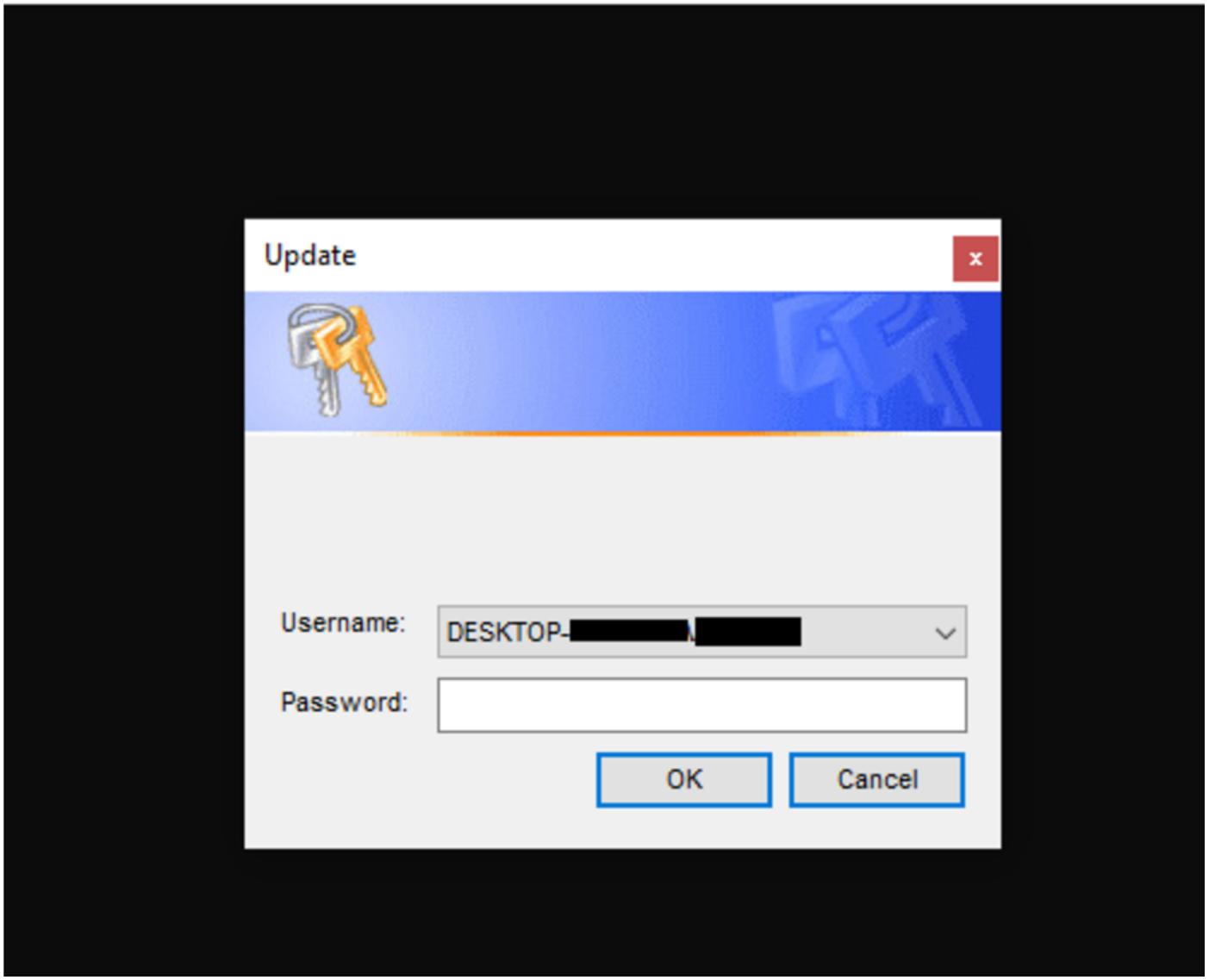

Figure 3. The credential harvesting prompt shown to the user upon executing the DLL (redaction by Rapid7).

The credential harvester is most commonly followed by the execution of a loader such as Zbot (a.k.a. Zloader) or DarkGate. This can then serve as a gateway to the execution of subsequent payloads in memory, facilitate data theft, or otherwise perform malicious actions. Rapid7 has also observed operators distributing alternate payload archives containing Cobalt Strike beacon loaders and a pair of Java payloads containing a user credential harvester variant and a custom multi-threaded beacon by which to remotely execute PowerShell commands. In some cases, operators have sent the user a short command, via Teams, which will then begin an infection chain after execution by the targeted user.

Rapid7 continues to observe inconsistent usage of the group's custom packer to deliver various malware payloads, including their custom credential harvester. A YARA rule is now publicly available that can be used to detect the packer. For example, this packer was used to deliver several obfuscated versions of Black Basta ransomware, obtained via open source intelligence, which directly links operators to the ongoing social engineering campaign.

At the time of writing, the threat actors behind the campaign continue to update both their strategy for gaining initial access and the tools subsequently used. For example, around the time the most recent campaign activity began, Rapid7 observed the delivery of a timestamped and versioned payload archive,

`171024_V1US.zip` (2024-10-17, version 1, US), which, when compared to a more recently delivered archive, `171124_V15.zip` (2024-11-17, version 15), highlights the rapid iteration being undertaken. Many of the payloads being delivered follow a similar pattern as previous activity and often consist of a legitimate file where an export or function entry point has been overwritten to jump to malicious code, and the result is signed with a likely stolen code signing certificate.

Intrusions related to the campaign should be taken seriously — the intent goes beyond typical phishing activity. Past campaign activity has led to the deployment of Black Basta ransomware. While Rapid7 has handled a high volume of incidents related to the current social engineering campaign across a variety of customer environments, to date, every case has been contained before the operator was able to move laterally beyond the targeted user's asset.

Technical Analysis

Initial Access

Each attack is preceded by the targeted user receiving an often overwhelming amount of emails. An operator will then attempt to contact the user via Microsoft Teams, either via messaging or calling, by which they will pretend to offer assistance. Operators will attempt to impersonate the organization's help desk, such as using the names of existing staff members.

During this social engineering stage, operators often need to troubleshoot with the user to establish remote control of the user's asset. Based on the environment, for example, RMM tool downloads or execution may be blocked (often some, but not all) or QuickAssist may be disabled, causing the operator to cycle through their options at establishing a foothold. One of the most common first steps after gaining either the confidence of the user, or remote access, is to execute a custom credential harvester.

Credential Harvesting

The credential harvester used by operators, for example `SafeStore.dll` (SHA256: 3B7E06F1CCAA207DC331AFD6F91E284FEC4B826C3C427DFFD0432FDC48D55176), is an updated version of the previously analyzed program `AntiSpam.exe`. The DLL variant of the credential harvester is executed by a command like the following example:

```
rundll32.exe SafeStore.dll,epaas_request_clone
```

The module will quickly execute three enumeration commands to gather system information — `systeminfo`, `route print`, `ipconfig /all` — and then prompt the user for their password. The user's credentials are appended onto a new line of the text file `123.txt` with each attempt, after the enumeration command output, regardless of whether the credentials are correct. If the user enters the wrong password, they will be prompted to try again. The output for the enumeration commands and the user's credentials were saved to the file `qwertyuio.txt` in older versions of the harvester, but are now saved to `123.txt`, within the user's `%TEMP%` directory. The enumeration commands within the updated version are executed via successive calls to `CreateProcessA`.

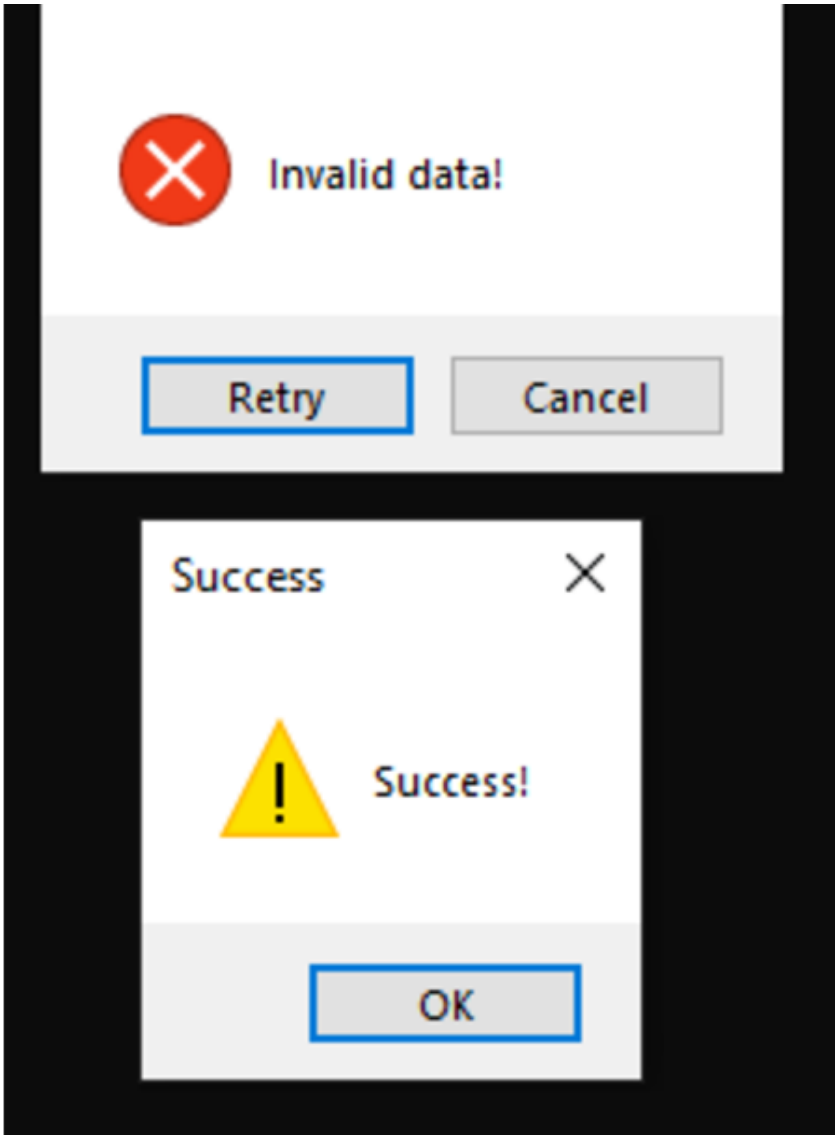

Figure 4. Success and failure messages for the credential harvester.

Based on analysis of one credential harvester sample, `EventCloud.dll`, the program was present in shellcode form. The shellcode is decrypted from the `Cursor Group 880` resource embedded within the executable, using the XOR key `5A 3C 77 6E 33 30 4D 38 4F 38 40 78 41 58 51 30 42 5F 3F 67 71 00`, and then injected locally. The following strings which were extracted from the shellcode show the output file and list dynamically loaded libraries:

Credential Harvester Strings	-	-	-	-
cmd.exe /c	%s%s	%s%s%s%s	123.txt	ooki
Update	filter kb_outl	Need credentials to update...	Username:	Password:
ntdll.dll	Gdi32.dll	user32.dll	msvcrt.dll	ucrtbase.dll
Comctl32.dll	Advapi32.dll	kernel32.dll	-	-

The Java variant of the credential harvester, `identity.jar`, provides a similar prompt to the user, though when a password is entered it is appended, without the username, to a .txt file with a random 10-letter alphabetic name to the current working directory. The cancel button on the prompt, shown below, is not functional and the prompt is drawn on top of other windows, meaning that it will not close until the user

has entered their password correctly.

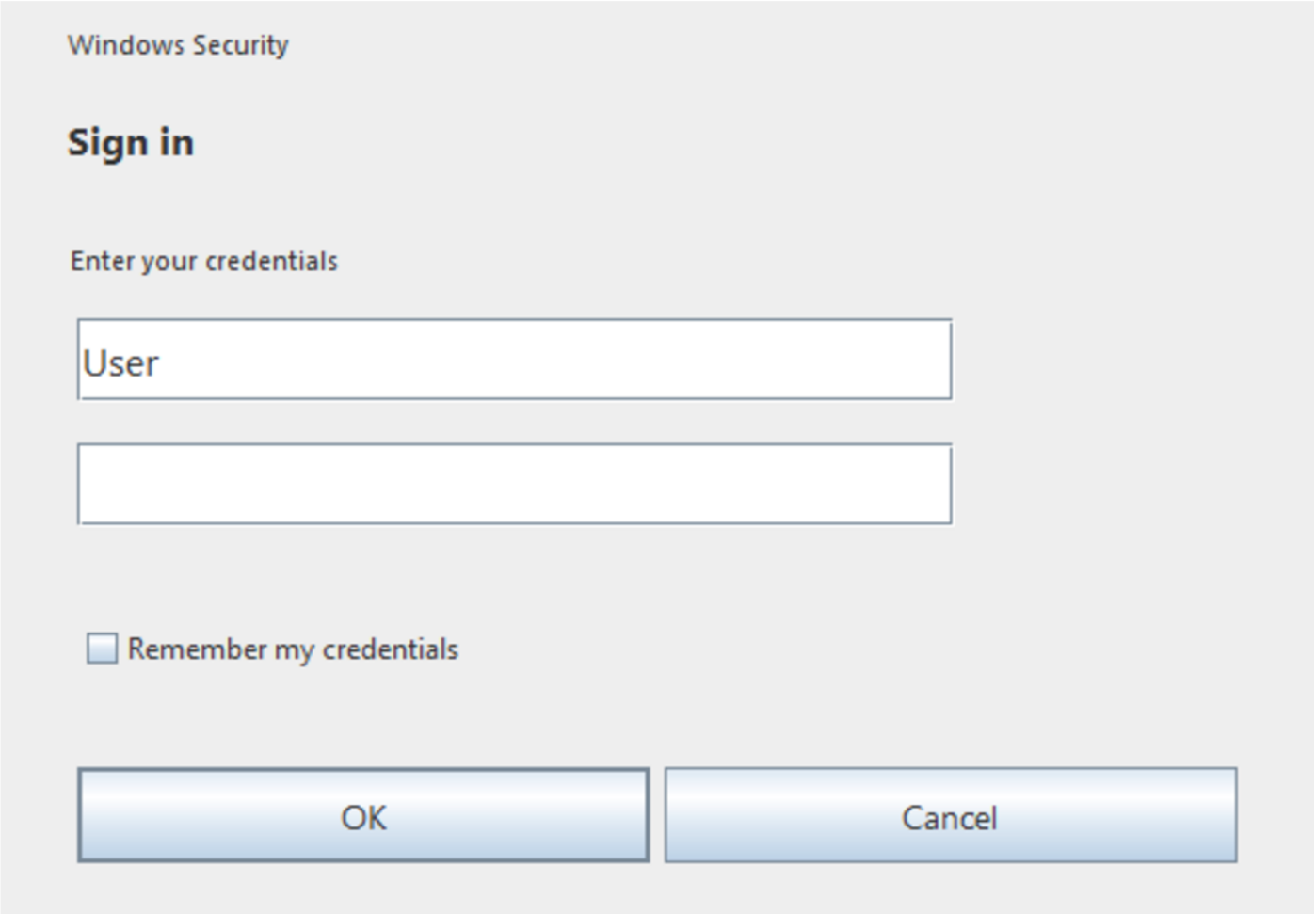

Figure 5. The credential harvesting prompt created by `identity.jar`.

Malware Payloads

Following execution of a credential harvester, an operator will typically infect the asset with Zbot or DarkGate. One of the Zbot samples delivered after initial access, `SyncSuite.exe` (SHA256: DB34E255AA4D9F4E54461571469B9DD53E49FEED3D238B6CFB49082DE0AFB1E4) contains similar functionality and strings to other Zbot/Zloader samples previously reported by ZScaler. However, in addition to previously observed strings, the sample also contains encrypted strings for an embedded command help menu, error messages, and more. Rapid7 observed the embedded malware version was `2.9.4.0`.

Upon execution, the malware will copy itself to a random folder within the `%APPDATA%` directory. If the file does not have its original filename however, the process will immediately exit. The malware also contains the functionality to establish persistence either via a Run key at `HKCU\Software\Microsoft\Windows\CurrentVersion\Run` or a scheduled task named after the executable, which executes the malware copy in %APPDATA% whenever the user logs on. After collecting the hostname, username, and the installation date from the `InstallDate` value contained within the registry key `HKLM\Software\Microsoft\Windows NT\CurrentVersion`, this data is concatenated (delimited by underscore characters) and encrypted, along with other config information. It is then stored within the user's registry inside a random key created at `HKCU\Software\Microsoft\`. The analyzed sample will also load a fresh copy of ntdll.dll to avoid hooking, which is then used to perform calls to

NTAPI functions. `SyncSuite.exe` ultimately injects itself into a suspended instance of `msedge.exe`, created using `NtCreateUserProcess` and executed via `ResumeThread`, a technique known as Process Hollowing.

All of the strings used by the malware are stored encrypted within the .rdata section along with the configuration. The strings are decrypted using an obfuscated loop that is ultimately a simple XOR operation with the hard coded key `16 EB D5 3E AA E6 51 09 14 D3 DF 18 AD D6 1B BD BE`, which is also stored in the .rdata section. The configuration is decrypted using an RC4 key, `F3 F9 F7 FB FA F3 F7 F7 FF F5 F2 F3 FA FD FE F2` for this sample. The decrypted configuration for `SyncSuite.exe` can be seen below, with empty rows removed. The configuration contains a different public RSA key and botnet ID than the one previously shared by ThreatLabz, indicating that the campaign is being run by a different affiliate. All decrypted strings from `SyncSuite.exe` can be seen in the Zbot Strings section following other Indicators of Compromise.

Offset(d)	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	Decoded text
00000000	00	00	00	00	54	65	73	74	00	00	00	00	00	00	00	00	....Test........
00000016	00	00	00	00	00	00	00	00	00	31	2E	30	00	00	00	00	.........1.0....
00000032	00	00	00	00	00	00	00	00	00	00	00	00	00	00	68	74	..............ht
00000048	74	70	73	3A	2F	2F	62	69	67	64	65	61	6C	63	65	6E	tps://bigdealcen
00000064	74	65	72	2E	77	6F	72	6C	64	2F	00	00	00	00	00	00	ter.world/......
00000080	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000688	00	00	00	00	00	00	00	00	03	00	00	00	11	00	00	00	....
00000704	2D	2D	2D	2D	2D	42	45	47	49	4E	20	50	55	42	4C	49	-----BEGIN PUBLI
00000720	43	20	4B	45	59	2D	2D	2D	2D	2D	0A	4D	49	47	66	4D	C KEY-----.MIGfM
00000736	41	30	47	43	53	71	47	53	49	62	33	44	51	45	42	41	A0GCSqGSIb3DQEBA
00000752	51	55	41	41	34	47	4E	41	44	43	42	69	51	4B	42	67	QUAA4GNADCBiQKBg
00000768	51	44	43	59	2B	55	46	74	76	55	35	63	6C	74	47	70	QDCY+UFtvU5cltGp
00000784	43	45	35	45	46	6C	2B	48	66	62	33	0A	53	38	37	73	CE5EFl+Hfb3.S87s
00000800	74	43	4A	64	48	68	53	36	74	75	79	79	61	59	6A	4F	tCJdHhS6tuyyaYjO
00000816	74	37	78	49	41	56	33	6B	46	63	36	42	6C	57	78	6B	t7xIAV3kFc6BlWxk
00000832	6D	4F	6D	6E	54	57	64	30	71	74	37	47	54	30	6F	2B	mOmnTWd0qt7GT0o+
00000848	74	44	32	75	54	66	37	7A	50	66	52	33	0A	74	6B	6D	tD2uTf7zPfR3.tkm
00000864	70	33	76	47	58	79	4E	5A	58	6A	52	39	30	6C	77	53	p3vGXyNZXjR90lwS
00000880	48	4B	73	32	32	6B	73	66	4F	67	6D	5A	70	4E	64	62	HKs22ksfOgmZpNdb
00000896	5A	2B	5A	48	56	6E	34	6F	7A	62	70	45	2F	63	47	58	Z+ZHVn4ozbpE/cGX
00000912	7A	7A	6F	2F	6B	39	33	7A	2B	50	36	4A	6B	0A	63	68	zzo/k93z+P6Jk.ch
00000928	58	5A	38	4E	77	46	5A	4D	38	41	52	72	63	6A	65	51	XZ8NwFZM8ARrcjeQ
00000944	49	44	41	51	41	42	0A	2D	2D	2D	2D	2D	45	4E	44	20	IDAQAB.-----END
00000960	50	55	42	4C	49	43	20	4B	45	59	2D	2D	2D	2D	2D	0A	PUBLIC KEY-----.
00000976	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000992	00	00	00	00	00	00	00	00	00	00	00	00	68	74	74	70	............http
00001008	73	3A	2F	2F	66	6F	72	64	6E	73	2F	63	6F	72	70	72	s://fordns/corpr
00001024	6F	6F	74	2F	20	7E	20	64	6E	73	3A	2F	2F	6E	73	31	oot/ ~ dns://ns1
00001040	2E	62	72	6F	77	6E	73	77	65	72	2E	63	6F	6D	00	00	.brownswer.com..
00001200	00	00	00	00	2D	3D	98	9A	08	08	04	04	08	08	08	08	....-=˜š........
00001216	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00001232	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00001248	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....

Figure 6. The decrypted Zbot configuration for `SyncSuite.exe` (1264 bytes).

Rapid7 has also observed the delivery of DarkGate malware following initial access. One payload archive

contained both a DarkGate infection initiation script, `test.vbs`, and an executable copy of the DarkGate malware itself, `SafeFilter.exe` (SHA256: EF28A572CDA7319047FBC918D60F71C124A038CD18A02000C7AB413677C5C161), though this copy is packed using the group's custom packer. The final payload containing the DarkGate malware, after several layers of decrypting and loading, contains the version string `7.0.6`. If the folder `c:\debugg` exists on the system when the malware is executed it will display the version number via `MessageBoxA`. The configuration for this sample can be seen below along with hard coded commands. Notably, the campaign ID for the sample appears to be `drk2`.

Figure 7. DarkGate displays its version using a debug message box.

The configuration is decrypted with the key `ckcilIcconnh` within a customized XOR loop near the beginning of execution to reveal CRLF delimited options. However, due to the implementation of the decryption loop, the keyspace is effectively reduced to that of a single byte (0-255), after the first byte. This makes the XOR key for the majority of the config `0x60`, for this sample allowing for the encrypted data to be trivially bruteforced.

Key-Value Pair (`SafeFilter.exe` DarkGate Config)	**Description**
0=179.60.149[.]194|	C2 domains or IP addresses, delimited with '|' characters
8=No	If enabled and the file `C:\ProgramData\hedfdfd\Autoit3.exe` does not exist, call `MessageBoxTimeoutA` using keys 11 and 12 and a timeout of 1770ms.
11=Error	Used by key 8 as a message box title.
12=PyKtS5Q	The string `Error`, base64 encoded with the custom alphabet `zLAxuU0kQKf3sWE7ePRO2imyg9GSpVoYC6rhlX48ZHnvjJDBNFtMd1I5acwbqT+=.` Used by key 8 as a message box caption.

Key-Value Pair (`SafeFilter.exe` DarkGate Config)	Description
13=6	Unknown
14=Yes	Unknown
15=80	C2 communication port.
1=Yes	Enables infection.
32=Yes	If enabled, attempt bypass of detected security products. For example, enables calls to `RtlAdjustPrivilege` and `NtRaiseHardError` to cause a crash if `hdkcgae` is not present in `C:\temp\` and a Kaspersky product has been detected.
3=No	If disabled, do an anti-vm display check.
4=No	If enabled, compare system drive size to key 18. If below, exit.
18=100	Minimum drive size in GB.
6=No	If enabled and key 3 is disabled, check the display for known virtual machine display strings using `EnumDisplayDevicesA`. If matched, exit. Failed to match properly when tested.
7=No	If enabled, compare system RAM to key 19. If below, exit.
19=4096	Minimum RAM size in MB.
5=No	If enabled, check the registry key `ProcessorNameString` at `HKLM\HARDWARE\DESCRIPTION\System\CentralProcessor\0` for `xeon`. If found, exit.
21=No	Unknown
22	Not present in the config for this sample, but is still checked for in the code. If enabled, set the variant string to `DLL`, otherwise `?`.
23=Yes	If enabled, set the variant string to `AU3` for Autoit3 payloads.
31=No	If enabled, set the variant string to `AHK` for AutoHotKey payloads.
25=drk2	Campaign ID
26=No	Unknown
27=rsFxMyDX	Decryption key, also used to bound/find payloads stored within other files.
28=No	Unknown
29=2	Unknown
35=No	Unknown
tabla=IsUiPQ4&atzM5N=0($"3]TGfyK8JYwvO61SAF{ndrDuol29*RkmqCpgxeX[EH,V)}7jbZBc.WLh	Unknown

DarkGate Hard-coded Commands
/c cd /d "C:\Users\User\AppData\Roaming<browser_dir>" && move <browser_name> <browser_name><random_alphabet_string>

DarkGate Hard-coded Commands
/c cd /d "C:\Users\User\AppData\Local" && move <browser_name> <browser_name><random_alphabet_string>
/c cmdkey /delete:
/c cmdkey /list > c:\temp\cred.txt
/c del /q /f /s C:\Users\User\AppData\Roaming\Mozilla\firefox*
/c ping 127.0.0.1 & del /q /f /s c:\temp & del /q /f /s C:\ProgramData\hedfdfd\ & rmdir /s /q C:\ProgramData\hedfdfd\
/c shutdown -f -r -t 0
/c shutdown -f -s -t 0
/c wmic ComputerSystem get domain > C:\ProgramData\hedfdfd\fcadaab

During execution, DarkGate will hash certain strings and use the result to create or check files at the directories `C:\ProgramData\hedfdfd`(mainfolder) and `C:\temp\`. The hashing algorithm uses a randomized key generated at runtime, so the hashes across infections will be different. Commonly used strings and their resultant hash, for the analysis environment, are shown below.

Path String	DarkGate Custom Hash
mainfolder	hedfdfd
logsfolder	fhhcfhh
settings	dhkbbfc
domain	fcadaab
mutex0	hfgdced
mutex1	cekchde
au3	dgfeabe
c.txt	adfcbdd
cc.txt	dehgaba
script	daaadeh
fs.txt	hdkcgae

DarkGate may also change its behavior if a known security product is detected. This is achieved by using `CreateToolhelp32Snapshot` and related functions to loop through running processes which are compared to a hard-coded list. The malware will also check for known installation directories using `GetFileAttributesA`. If a security product is found, a flag will be set which may alter the execution path. Only the following products had associated flags:

DarkGate “Supported” Security Products	-	-	-	-
Windows Defender	Sophos	Quick Heal	MalwareBytes	Panda Security
Norton/Symantec	ESET/ Nod32	Kaspersky	Avast	SentinelOne

DarkGate "Supported" Security Products	-	-	-	-
Bitdefender	-	-	-	-

At the end of the first execution of the DarkGate payload, it will then attempt to inject itself into a host process. First, DarkGate will select the injection target by searching a list of hard coded directories for any executable that contains the string `updatecore.exe`, subdirectories included. The path `C:\Program Files (x86)\Microsoft\EdgeUpdate\` is searched first, with the fallback being `C:\Program Files (x86)\Microsoft\EdgeUpdate\MicrosoftEdgeUpdate.exe`. If a matching Edge executable is not found, the path `C:\Program Files (x86)\Google\Update\` is then searched. If that also fails, the malware will attempt to use `C:\Windows\Microsoft.NET\Framework\v4.0.30319\msbuild.exe`.

After successfully choosing the injection target, DarkGate will then inject itself into the target process using shellcode, terminating the original instance of the final DarkGate payload after executing the shellcode. When creating an instance of the target process to inject, DarkGate will also attempt to spoof the parent process ID (PPID) of the injection target by enumerating running processes for accessibility using `OpenProcess` and then randomly selecting one from an assembled list. The PPID of the target is then updated using `UpdateProcThreadAttribute` prior to creation with `CreateProcessA`.

Execution of the injected process is coordinated by checking for the presence of two file based mutexes within `C:\ProgramData\hedfdfd\` (mainfolder). Each instance of the DarkGate malware checks both of the file-based mutexes. The file mutex usage is checked via calls to `CreateFileA` using an exclusive share mode flag (0) and a creation disposition of `CREATE_ALWAYS`, which means that if the mutex is already in usage by another DarkGate instance the call will fail. If the call to both mutexes created by DarkGate, `hfgdced` and `cekchde`, fails, DarkGate will exit. As a result of having two mutexes, DarkGate will typically run within two injected process instances at the same time, so if one process is terminated, the remaining instance will spawn another. If a DarkGate instance is spawned and both calls to open the file based mutexes fail, indicating two existing DarkGate instances, the new instance will terminate. This technique is rarely used by malware developers and highlights the sophistication of DarkGate malware.

DarkGate will unconditionally log keystrokes as well as clipboard data that is under 1024 bytes. The logged data is stored encrypted at `C:\ProgramData\hedfdfd\fhhcfhh` (mainfolder\logsfolder) within files named `<date>.log`. The logged data may be sent directly to the C2 address contained within the config. A thread is also created to persist on infected systems by creating the Run key `daaadeh` (script) at `HKCU\Software\Microsoft\Windows\CurrentVersion\Run`. The Run key will point to the copies of `Autoit3.exe` and the compiled AU3 script payload `dgfeabe.a3x` (au3) created at `C:\ProgramData\hedfdfd` (mainfolder), with the former executing the latter every time the user logs on. When the AU3 script is executed, DarkGate reinfects the system. The thread continuously monitors the text within the infected user's active window however, sleeping 1500ms between checks, and will delete the registry key if a blacklisted application is detected. This list includes popular analysis tools such as Process Hacker, Process Monitor, Task Manager, and even the Windows Registry Editor.

The DarkGate sample executed by `SafeFilter.exe` contains 78 remote commands, some of which can be seen below with their intended function. Every loop, the malware will re-send the text of the active window, user idle time, and whether or not the malware instance has admin rights, before checking for a command.

Command ID	Function
1000	Sleep for a randomized amount of time.
1004	Use MessageBoxA to display the message `test msg`.
1044,1045,1046	Click the user's mouse at specified screen coordinates using `SetCursorPos` and successive calls to `mouse_event`. 1044 for double left-click. 1045 for single left click. 1046 for single right click.
1049	Create a remote shell via `powershell.exe`.
1059	Terminate process by PID.
1061	Inject DarkGate shellcode into a specified process or an Edge/Chrome process if none is selected. The shellcode is then executed via `ResumeThread`.
1062,1063,1064	Inject DarkGate shellcode into a specified process or `cmd.exe` if none is selected. The shellcode is then executed via `CreateRemoteThread`.
1066	Remove infection files by using `cmd.exe` to delete the staging directories `C:\ProgramData\hedfdfd` and `c:\temp\`.
1071	Steal `sitemanager.xml` and `recentservers.xml` from `%APPDATA%\FileZilla\` if present.
1079	If admin, delete stored credentials found using cmdkey.
1080	Rename browser directories for Firefox, Chrome, and Brave if present after terminating the related browser executable. Attempt to steal Opera cookies if present, after terminating the process.
1081	Use NTAPI calls `RtlAdjustPrivilege` and `NtRaiseHardError` to crash the system.
1083	Use the `shutdown` command to turn the system off.
1084	Use the `shutdown` command to restart the system.
1089	If 1=Yes in config, reinfect system with AU3 payloads.
1093	Create a remote shell via `cmd.exe`.
1097	Infect system with AU3 variant. Creates the files `script.a3x` and `Autoit3.exe` in `c:\temp` and then executes `script.a3x` via `Autoit3.exe` using `CreateProcessA`.
1104	Infect system with AHK variant. Creates the files `script.ahk`, `test.txt`, and `AutoHotkey.exe` in `c:\temp` and then executes `script.ahk` via `AutoHotkey.exe` using `CreateProcessA`.
1108	Infect system with DLL variant. Creates the files `libcurl.dll`, `test.txt`, and `GUP.exe` in `c:\temp` and then executes `GUP.exe` via `CreateProcessA`.
1111	Create the files `ransom.txt` and `decrypter.exe` in `c:\temp`. Terminate `decrypter.exe` if already running and then execute `decrypter.exe` using `CreateProcessA`. Likely ransomware deployment method.

DarkGate Remote Command Related Strings	-	-	-	-
U_Binder	U_BotUpdate	U_Constantes	U_FTPRecovery	U_FileManager
U_FileManagerMisc	U_GetScreens	U_HVNC	U_HVNC_7	
U_HWID	U_InfoRecovery	U_InjectOnFly	U_Keylogger	U_LNKStartup
U_MemExecute	U_MemExecuteMisc	U_RemoteScreen	U_SysApi	U_SysNtReadWrite

DarkGate Remote Command Related Strings	-	-	-	-
U_miniclipboard	u_AntiAntiStartup	u_Antis	u_AudioRecord	u_CustomBase64
u_ExtraMisc	u_HollowInstall	u_InjectEP	u_InvokeBSOD	u_RDPRecovery
u_Ransomware	u_ReadCookies	u_ReverseShell	u_RootkitMutex	u_Settings
u_SettingsPad	u_ShellcodeEP	u_UnlockCookies	u_loadpe	hxxps://ipinfo[.]io/ip

Mitigation Guidance

Rapid7 recommends taking the following precautions to limit exposure to these types of attacks:

Restrict the ability for external users to contact users via Microsoft Teams to the greatest extent possible. This can be done for example by blocking all external domains or creating a white/black list. Microsoft Teams will allow all external requests by default. For more information, see this reference.

Standardize remote management tools within the environment. For unapproved tools, block known hashes and domains to prevent usage. Hash blocking can be done, for example, via Windows AppLocker or an endpoint protection solution.

Provide user awareness training regarding the social engineering campaign. Familiarize users with official help desk and support procedures to enable them to spot and report suspicious requests.

Standardize VPN access. Traffic from known low cost VPN solutions should be blocked at a firewall level if there is no business use case.

Rapid7 Customers

InsightIDR, Managed Detection and Response, and Managed Threat Complete customers have existing detection coverage through Rapid7's expansive library of detection rules. Rapid7 recommends installing the Insight agent on all applicable hosts to ensure visibility into suspicious processes and proper detection coverage. Below is a non-exhaustive list of detections that are deployed and will alert on behavior related to this activity:

Detections
Suspicious Chat Request - Potential Social Engineering Attempt
Initial Access - Potential Social Engineering Session Initiated Following Chat Request
Suspicious Conversation - Potential Social Engineering Message Interaction
Attacker Technique - Process Executed Using Nt Object Path
Suspicious Process - Enumeration Burst via ShellExecute
Attacker Technique - Renamed Kaspersky Dump Writer

Detections
Ransomware - Possible Black Basta Related Binary Execution
Credential Access - Steal or Forge Kerberos tickets
Suspicious Process - Diskshadow (Windows Server) Delete Shadow Copies
Non-Approved Application - Remote Management and Monitoring (RMM) Tools

MITRE ATT&CK Techniques

Tactic	Technique	Procedure
Resource Development	T1587.001: Develop Capabilities: Malware	The threat actor is actively developing new malware to distribute.
Impact	T1498: Network Denial of Service	The threat actor overwhelms email protection solutions with spam.
Initial Access	T1566.004: Phishing: Spearphishing Voice	The threat actor calls impacted users and pretends to be a member of their organization's IT team to gain remote access.
Defense Evasion	T1140: Deobfuscate/Decode Files or Information	The threat actor encrypts some zip archive payloads with a password.
Defense Evasion	T1055.002: Process Injection: Portable Executable Injection	Multiple payloads executed by the threat actor utilize local PE injection.
Defense Evasion	T1620: Reflective Code Loading	Multiple payloads executed by the threat actor load and execute shellcode.
Credential Access	T1649: Steal or Forge Authentication Certificates	The threat actor has distributed numerous signed malware payloads.
Credential Access	T1056.001: Input Capture: Keylogging	The threat actor runs an executable that harvests the user's credentials.
Credential Access	T1558.003: Steal or Forge Kerberos Tickets: Kerberoasting	The threat actor has performed Kerberoasting after gaining initial access.
Discovery	T1033: System Owner/User Discovery	The threat actor enumerates asset and user information within the environment after gaining access.
Command and Control	T1572: Protocol Tunneling	The threat actor has attempted to use SSH reverse tunnels.
Command and Control	T1219: Remote Access Software	The threat actor has used QuickAssist, AnyDesk, ScreenConnect, TeamViewer, Level, and more, to facilitate remote access.

Indicators of Compromise

Indicators of compromise are available here.

Subscribe Now