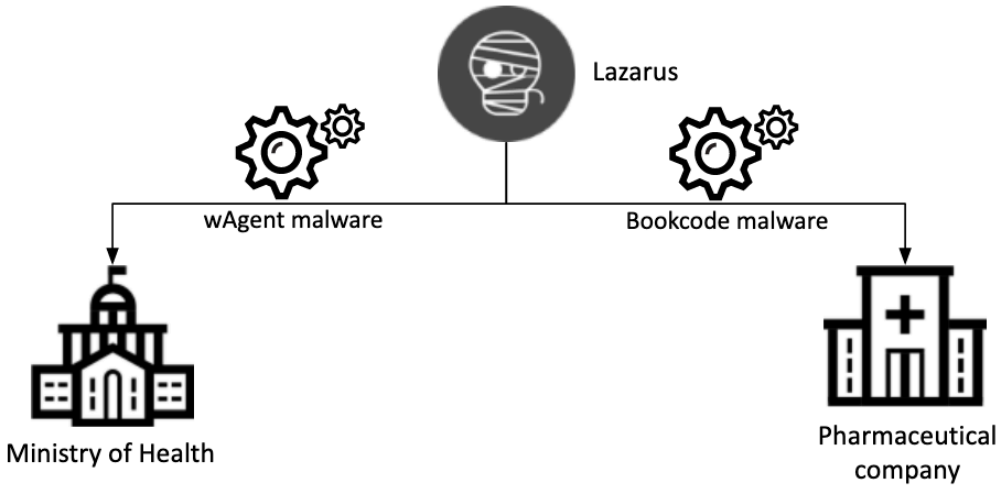
Lazarus covets COVID-19-related intelligence

By Seongsu Park

Published: 2020-12-23 · Archived: 2026-04-05 13:55:52 UTC

As the COVID-19 crisis grinds on, some threat actors are trying to speed up vaccine development by any means available. We have found evidence that actors, such as the Lazarus group, are going after intelligence that could help these efforts by attacking entities related to COVID-19 research.

While tracking the Lazarus group's continuous campaigns targeting various industries, we discovered that they recently went after COVID-19-related entities. They attacked a pharmaceutical company at the end of September, and during our investigation we discovered that they had also attacked a government ministry related to the COVID-19 response. Each attack used different tactics, techniques and procedures (TTPs), but we found connections between the two cases and evidence linking those attacks to the notorious Lazarus group.



Relationship of recent Lazarus group attack

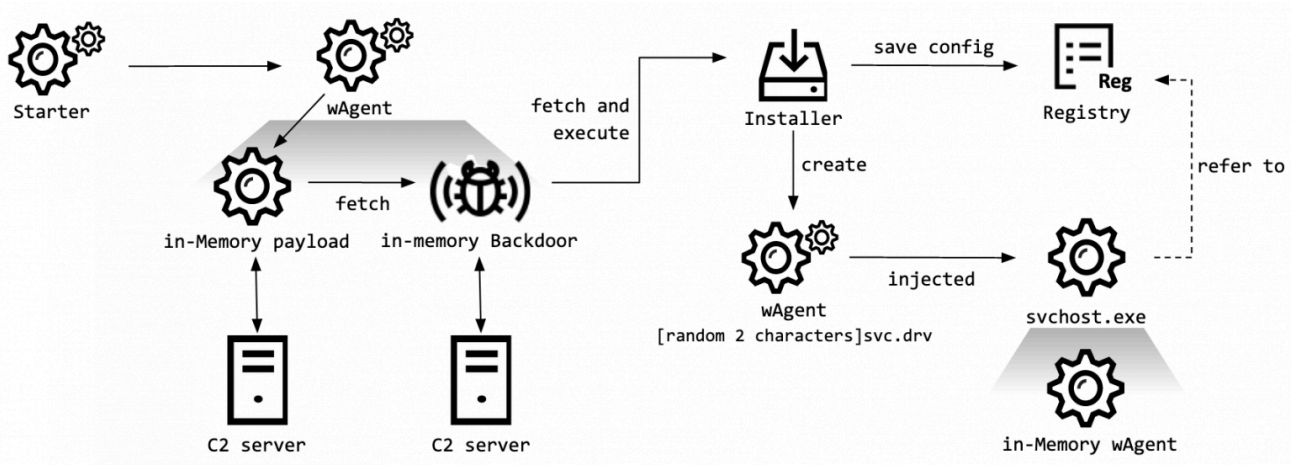
In this blog, we describe two separate incidents. The first one is an attack against a government health ministry: on October 27, 2020, two Windows servers were compromised at the ministry. We were unable to identify the infection vector, but the threat actor was able to install a sophisticated malware cluster on these servers. We already knew this malware as 'wAgent'. It's main component only works in memory and it fetches additional payloads from a remote server.

The second incident involves a pharmaceutical company. According to our telemetry, this company was breached on September 25, 2020. This time, the Lazarus group deployed the Bookcode malware, previously reported by ESET, in a supply chain attack through a South Korean software company. We were also able to observe post-exploitation commands run by Lazarus on this target.

Both attacks leveraged different malware clusters that do not overlap much. However, we can confirm that both of them are connected to the Lazarus group, and we also found overlaps in the post-exploitation process.

wAgent malware cluster

The malware cluster has a complex infection scheme:



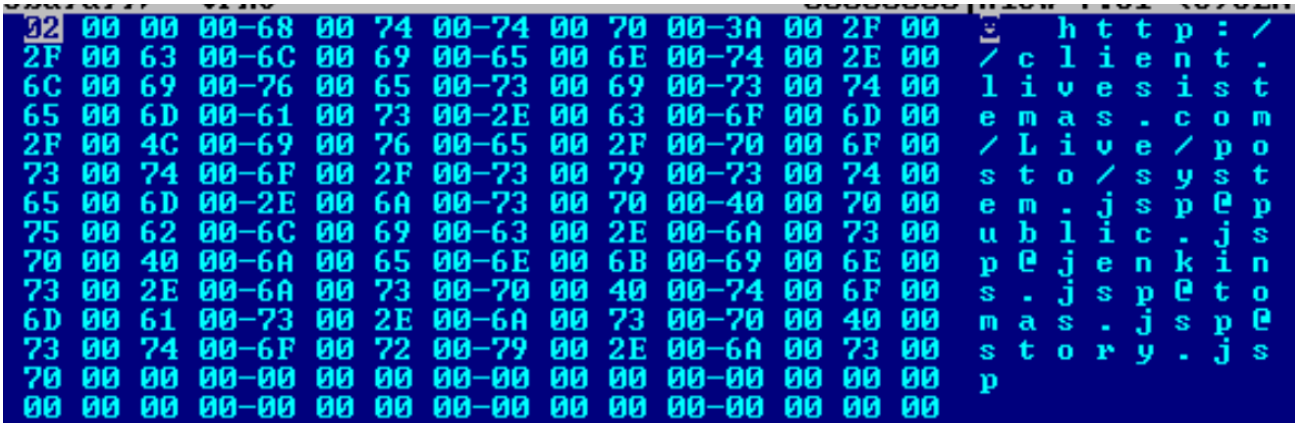
Infection scheme of the wAgent malware cluster

Unfortunately, we were unable to obtain the starter module used in this attack. The module seems to have a trivial role: executing wAgent with specific parameters. One of the wAgent samples we collected has fake metadata in order to make it look like the legitimate compression utility XZ Utils.

According to our telemetry, this malware was directly executed on the victim machine from the command line shell by calling the Thumbs export function with the parameter:

```
c:\windows\system32\rundll32.exe  C:\Programdata\Oracle\javac.dat, Thumbs 8IZ-VU7-109-S2MY
```

The 16-byte string parameter is used as an AES key to decrypt an embedded payload – a Windows DLL. When the embedded payload is loaded in memory, it decrypts configuration information using the given decryption key. The configuration contains various information including C2 server addresses, as well as a file path used later on. Although the configuration specifies two C2 servers, it contains the same C2 server twice. Interestingly, the configuration has several URL paths separated with an '@' symbol. The malware attempts to connect to each URL path randomly.



C2 address in the configuration

When the malware is executed for the first time, it generates identifiers to distinguish each victim using the hash of a random value. It also generates a 16-byte random value and reverses its order. Next, the malware concatenates this random 16-byte value and the hash using '@' as a delimiter. *i.e.*: *82UKx3vnjQ791PL2@29312663988969*

POST parameter names (shown below) are decrypted at runtime and chosen randomly at each C2 connection. We've previously seen and reported to our Threat Intelligence Report customers that a very similar technique was used when the Lazarus group attacked cryptocurrency businesses with an evolved downloader malware. It is worth noting that Tistory is a South Korean blog posting service, which means the malware author is familiar with the South Korean internet environment:

*plugin course property **tistory** tag vacon slide parent manual themes product notice portal articles category doc entry isbn tb idx tab maincode level bbs method thesis content blogdata tname*

The malware encodes the generated identifier as base64 and POSTs it to the C2. Finally, the agent fetches the next payload from the C2 server and loads it in memory directly. Unfortunately, we couldn't obtain a copy of it, but according to our telemetry, the fetched payload is a Windows DLL containing backdoor functionalities. Using this in-memory backdoor, the malware operator executed numerous shell commands to gather victim information:

```
cmd.exe /c ping -n 1 -a 192.[redacted]

cmd.exe /c ping -n 1 -a 192.[redacted]

cmd.exe /c dir \\192.[redacted]\c$

cmd.exe /c query user

cmd.exe /c net user [redacted] /domain

cmd.exe /c whoami
```

Persistent wAgent deployed

Using the wAgent backdoor, the operator installed an additional wAgent payload that has a persistence mechanism. After fetching this DLL, an export called SagePlug was executed with the following command line parameters:

```
rundll32.exe c:\programdata\oracle\javac.io, SagePlug 4GO-R19-0TQ-HL2A
c:\programdata\oracle\~TMP739.TMP
```

4GO-R19-0TQ-HL2A is used as a key and the file path indicates where debugging messages are saved. This wAgent installer works similarly to the wAgent loader malware described above. It is responsible for loading an embedded payload after decrypting it with the 16-byte key from the command line. In the decrypted payload, the malware generates a file path to proceed with the infection:

- *C:\Windows\system32\[random 2 characters]svc.drv*

This file is disguised as a legitimate tool named SageThumbs Shell Extension. This tool shows image files directly in Windows Explorer. However, inside it contains an additional malicious routine.

While creating this file, the installer module fills it with random data to increase its size. The malware also copies cmd.exe's creation time to the new file in order to make it less easy to spot.

For logging and debugging purposes, the malware stores information in the file provided as the second argument (*c:\programdata\oracle\~TMP739.TMP* in this case). This log file contains timestamps and information about the infection process. We observed that the malware operators were checking this file manually using Windows commands. These debugging messages have the same structure as previous malware used in attacks against cryptocurrency businesses involving the Lazarus group. More details are provided in the Attribution section.

After that, the malware decrypts its embedded configuration. This configuration data has a similar structure as the aforementioned wAgent malware. It also contains C2 addresses in the same format:

- *hxxps://iski.silogica[.]net/events/serial.jsp@WFRForms.jsp@import.jsp@view.jsp@cookie.jsp*
- *hxxp://sistema.celllab[.]com.br/webrun/Navbar/auth.jsp@cache.jsp@legacy.jsp@chooseIcon.jsp@customZoom.jsp*
- *hxxp://www.bytecortex.com[.]br/eletronicos/digital.jsp@exit.jsp@helpform.jsp@masks.jsp@Functions.jsp*
- *hxxps://sac.najatelecom.com[.]br/sac/Dados/ntlm.jsp@loading.jsp@access.jsp@local.jsp@default.jsp*

The malware encrypts configuration data and stores it as a predefined registry key with its file name:

- *HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\eventlog\Application\Emulate – [random 2 characters]svc*

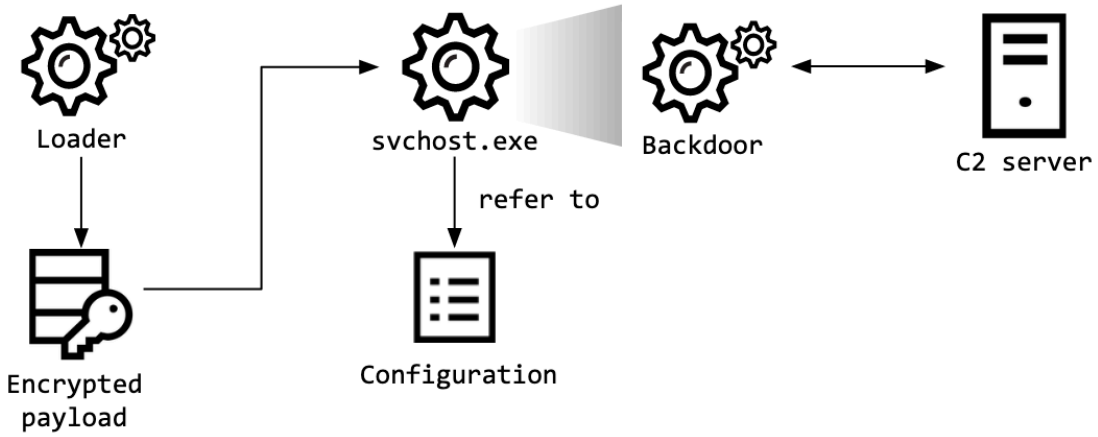
It also takes advantage of the Custom Security Support Provider by registering the created file path to the end of the existing registry value. Thanks to this registry key, this DLL will be loaded by lsass.exe during the next startup.

- *HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa – Security Packages : kerberos msv1_0 schannel wdigest tspkg pku2u* ***[random 2 characters]svc.drv***

Finally, the starter module starts the *[random 2 characters]svc.drv* file in a remote process. It searches for the first svchost.exe process and performs DLL injection. The injected *[random 2 characters]svc.drv* malware contains a malicious routine for decrypting and loading its embedded payload. The final payload is wAgent, which is responsible for fetching additional payloads from the C2, possibly a fully featured backdoor, and loading it in the memory.

Bookcode malware cluster

The pharmaceutical company targeted by Lazarus group's Bookcode malware is developing a COVID-19 vaccine and is authorized to produce and distribute COVID-19 vaccines. We previously saw Lazarus attack a software company in South Korea with Bookcode malware, possibly targeting the source code or supply chain of that company. We have also witnessed the Lazarus group carry out spear phishing or strategic website compromise in order to deliver Bookcode malware in the past. However, we weren't able to identify the exact initial infection vector for this incident. The whole infection procedure confirmed by our telemetry is very similar to the one described in ESET's latest publication on the subject.



Bookcode infection procedure

Although we didn't find the piece of malware tasked with deploying the loader and its encrypted Bookcode payload, we were able to identify a loader sample. This file is responsible for loading an encrypted payload named gmslogmgr.dat located in the system folder. After decrypting the payload, the loader finds the Service Host Process (svchost.exe) with *winmgmt, ProfSvc* or *Appinfo* parameters and injects the payload into it. Unfortunately, we couldn't acquire the encrypted payload file, but we were able to reconstruct the malware actions on the victim machine and identify it as the Bookcode malware we reported to our Threat Intelligence Report customers.

Upon execution, the Bookcode malware reads a configuration file. While previous Bookcode samples used the file *perf91nc.inf* as a configuration file, this version reads its configuration from a file called *C_28705.NLS*. This Bookcode sample has almost identical functionality as the malware described in the comprehensive report recently published by Korea Internet & Security Agency (KISA). As described on page 57 of that report, once the malware is started it sends information about the victim to the attacker's infrastructure. After communicating with the C2 server, the malware provides standard backdoor functionalities.

Post-exploitation phase

The Lazarus group's campaign using the Bookcode cluster has its own unique TTPs, and the same modus operandi was used in this attack.

- Extracting infected host information, including password hashes, from the registry sam dump.
- Using Windows commands in order to check network connectivity.
- Using the WakeMeOnLan tool to scan hosts in the same network.

After installing Bookcode on September 25, 2020, the malware operator started gathering system and network information from the victim. The malware operator also collected a registry sam dump containing password hashes:

- exe /c "reg.exe save hklm\sam %temp%\~reg_sam.save > "%temp%\BD54EA8118AF46.TMP~" 2>&1″
- exe /c "reg.exe save hklm\system %temp%\~reg_system.save > "%temp%\405A758FA9C3DD.TMP~" 2>&1″

In the lateral movement phase, the malware operator used well-known methodologies. After acquiring account information, they connected to another host with the "net" command and executed a copied payload with the "wmic" command.

- exe /c “netstat -aon | find “ESTA” > %temp%\~431F.tmp
- exe /c “net use \\172.[redacted] “[redacted]” /u:[redacted] > %temp%\~D94.tmp” 2>&1″
- wmic /node:172.[redacted] /user:[redacted] /password:”[redacted]” process call create “%temp%\engtask.exe” > %temp%\~9DC9.tmp” 2>&1″

Moreover, Lazarus used ADfind in order to collect additional information from the Active Directory. Using this utility, the threat actor extracted a list of the victim’s users and computers.

Infrastructure of Bookcode

As a result of closely working with the victim to help remediate this attack, we discovered an additional configuration file. It contains four C2 servers, all of which are compromised web servers located in South Korea.

- hxxps://www.kne.co[.]kr/upload/Customer/BBS.asp
- hxxp://www.k-kiosk[.]com/bbs/notice_write.asp
- hxxps://www.gongim[.]com/board/ajax_Write.asp
- hxxp://www.cometnet[.]biz/framework/common/common.asp

One of those C2 servers had directory listing enabled, so we were able to gain insights as to how the attackers manage their C2 server:

```
2020-11-13 오전 8:47          144 _ICEBIRD007.dat
2012-09-11 오후 5:54         7645 Customer_Session.asp
2020-12-02 오후 12:37          51 ~F05990302ERA.jpg
```

Attacker files listed on a compromised website

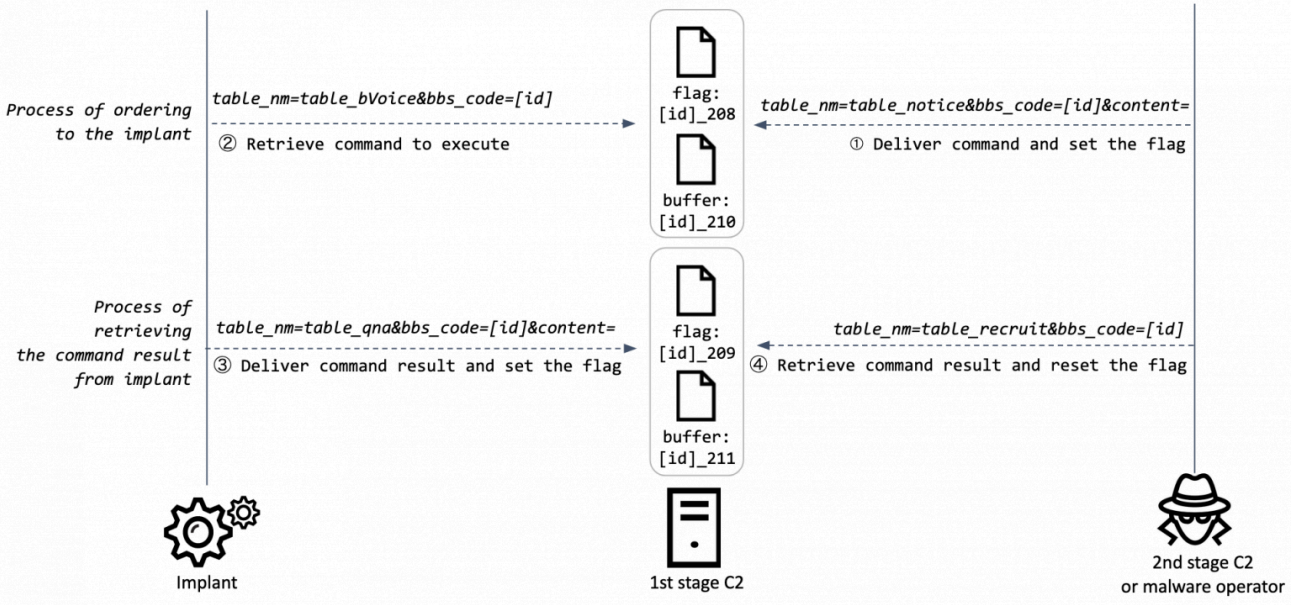
We discovered several log files and a script from the compromised server, which is a “first-stage” C2 server. It receives connections from the backdoor, but only serves as a proxy to a “second-stage” server where the operators actually store orders.

File name	Description
_ICEBIRD007.dat	A log file containing the identifier of victims and timestamps.
~F05990302ERA.jpg	Second-stage C2 server address: hxxps://www.locknlockmall[.]com/common/popup_left.asp
Customer_Session.asp	Malware control script.

Customer_Session.asp is a first-stage C2 script responsible for delivering commands from the next-stage C2 server and command execution results from the implant. In order to deliver proper commands to each victim, the *bbs_code* parameter from the implants is used as an identifier. The script uses this identifier to assign commands to the correct victims. Here is how the process of sending an order for a particular victim works:

1. 1 The malware operator sets the corresponding flag(*[id]_208*) of a specific implant and saves the command to the variable([id]_210).

2. 2 The implant checks the corresponding flag(*[id]_208*) and retrieves the command from the variable(*[id]_210*) if it is set.
3. 3 After executing the command, the implant sends the result to the C2 server and sets the corresponding flag.
4. 4 The malware operator checks the flag and retrieves the result if the flag is set.



Logic of the C2 script

Besides implant control features, the C2 script has additional capabilities such as updating the next-stage C2 server address, sending the identifier of the implant to the next-stage server or removing a log file.

table_nm value	**Function name**	**Description**
table_qna	qnaview	Set [id]_209 variable to TRUE and save the “content” parameter value to [id]_211.
table_recruit	recuritview	If [id]_209 is SET, send contents of [id]_211 and reset it, and set [ID]_209 to FALSE.
table_notice	notcieview	Set [id]_208 and save the “content” parameter value to [id]_210.
table_bVoice	voiceview	If [id]_208 is SET, send contents of [id]_210 and reset it, and set [id]_208 to FALSE.
table_bProduct	productview	Update the ~F05990302ERA.jpg file with the URL passed as the “*target_url*” parameter.

table_community	communityview	Save the identifier of the implant to the log file. Read the second-stage URL from *~F05990302ERA.jpg* and send the current server URL and identifier to the next hop server using the following format: *bbs_type=qnaboard&table_id=[base64ed identifier] &accept_identity=[base64 encoded current server IP]&redirect_info=[base64ed current server URL]*
table_free	freeview	Read *_ICEBIRD007.dat* and send its contents, and delete it.

Attribution

We assess with high confidence that the activity analyzed in this post is attributable to the Lazarus group. In our previous research, we already attributed the malware clusters used in both incidents described here to the Lazarus group. First of all, we observe that the wAgent malware used against the health ministry has the same infection scheme as the malware that the Lazarus group used previously in attacks on cryptocurrency businesses.

- Both cases used a similar malware naming scheme, generating two characters randomly and appending “svc” to it to generate the path where the payload is dropped.
- Both malicious programs use a Security Support Provider as a persistence mechanism.
- Both malicious programs have almost identical debugging messages.

Here is a side-by-side comparison of the malware used in the ministry of health incident, and the malware (4088946632e75498d9c478da782aa880) used in the cryptocurrency business attack:

Debugging log from ministry of health case	**Debugging log of cryptocurrency business case**
15:18:20 **Extracted Dll :** *[random 2bytes]*svc.drv 15:59:32 **Reg Config Success !** 16:08:45 **Register Svc Success !** 16:24:53 Injection Success, Process ID : 544	**Extracted Dll :** *[random 2bytes]*svc.dll Extracted Injecter : *[random 2bytes]*proc.exe **Reg Config Success !** **Register Svc Success !** Start Injecter Success !

Regarding the pharmaceutical company incident, we previously concluded that Bookcode is exclusively used by the Lazarus group. According to our Kaspersky Threat Attribution Engine (KTAE), one of the Bookcode malware samples (MD5 0e44fcafab066abe99fe64ec6c46c84e) contains lots of code overlaps with old Manuscrypt variants.

Analysis: Sample 0e44fcafab066abe99fe64ec6c46c84e

Size: 249856
Matched attribution entities: Manuscrypt (100%), Zoxpng (1%)

Similar samples (64)

MD5	Size	Genotypes (matched / total)	Strings (matched / total)	Similarity	Attribution entity
bcdce86c22dafa25175bc32b4ec7bbae	250368	480 / 977	23 / 36	64%	Manuscrypt
4350aa8b8305b905d29022dfbfc01c0d	249856	480 / 977	21 / 36	58%	Manuscrypt
6d73dbc24a9099f45d2800f3cf554bdb	274432	462 / 927	21 / 36	58%	Manuscrypt
7515f26dfedd8d0b984e913b59ecf90b	274432	462 / 927	21 / 36	58%	Manuscrypt
7835ec1556f1663abc47aedcd7fccdc9	274432	462 / 927	21 / 36	58%	Manuscrypt
858dd856570ae3fb563075274d0601d9	274432	462 / 927	21 / 36	58%	Manuscrypt

Kaspersky Threat Attribution Engine results for Bookcode

Moreover, the same strategy was used in the post-exploitation phase, for example, the usage of ADFind in the attack against the health ministry to collect further information on the victim's environment. The same tool was deployed during the pharmaceutical company case in order to extract the list of employees and computers from the Active Directory. Although ADfind is a common tool for the post-exploitation process, it is an additional data point that indicates that the attackers use shared tools and methodologies.

Conclusions

These two incidents reveal the Lazarus group's interest in intelligence related to COVID-19. While the group is mostly known for its financial activities, it is a good reminder that it can go after strategic research as well. We believe that all entities currently involved in activities such as vaccine research or crisis handling should be on high alert for cyberattacks.

Indicators of compromise

wAgent

dc3c2663bd9a991e0fbec791c20cbf92 %programdata%\oracle\javac.dat
26545f5abb70fc32ac62fdab6d0ea5b2 %programdata%\oracle\javac.dat
9c6ba9678ff986bcf858de18a3114ef3 %programdata%\grouppolicy\Policy.DAT

wAgent Installer

4814b06d056950749d07be2c799e8dc2 %programdata%\oracle\javac.io, %appdata%\ntuser.dat

wAgent compromised C2 servers

> http://client.livesistemas[.]com/Live/posto/system.jsp@public.jsp@jenkins.jsp@tomas.jsp@story.jsp
>
> hxxps://iski.silogica[.]net/events/serial.jsp@WFRForms.jsp@import.jsp@view.jsp@cookie.jsp

hxxp://sistema.celllab[.]com.br/webrun/Navbar/auth.jsp@cache.jsp@legacy.jsp@chooseIcon.jsp@customZoom.jsp hxxp://www.bytecortex.com[.]br/eletronicos/digital.jsp@exit.jsp@helpform.jsp@masks.jsp@Functions.jsp hxxps://sac.najatelecom.com[.]br/sac/Dados/ntlm.jsp@loading.jsp@access.jsp@local.jsp@default.jsp

wAgent file path

%SystemRoot%\system32\[random 2 characters]svc.drv

wAgent registry path

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\eventlog\Application\Emulate - [random 2 characters]svc

Bookcode injector

5983db89609d0d94c3bcc88c6342b354 %SystemRoot%\system32\scaccessservice.exe, rasprocservice.exe

Bookcode file path

%SystemRoot%\system32\C_28705.NLS %SystemRoot%\system32\gmslogmgr.dat

Bookcode compromised C2 servers

hxxps://www.kne.co[.]kr/upload/Customer/BBS.asp hxxp://www.k-kiosk[.]com/bbs/notice_write.asp hxxps://www.gongim[.]com/board/ajax_Write.asp hxxp://www.cometnet[.]biz/framework/common/common.asp hxxps://www.locknlockmall[.]com/common/popup_left.asp

MITRE ATT&CK Mapping.

Tactic	Technique.	Technique Name.
Execution	T1059.003	Command and Scripting Interpreter: Windows Command Shell
	T1569.002	System Services: Service Execution
Persistence	T1547.005	Boot or Logon Autostart Execution: Security Support Provider
	T1543.003	Create or Modify System Process: Windows Service
Privilege Escalation	T1547.005	Boot or Logon Autostart Execution: Security Support Provider
	T1543.003	Create or Modify System Process: Windows Service
	T1055.001	Process Injection: Dynamic-link Library Injection
Defense Evasion	T1070.006	Indicator Removal on Host: Timestomp
	T1055.001	Process Injection: Dynamic-link Library Injection
	T1140	Deobfuscate/Decode Files or Information
	T1027.001	Obfuscated Files or Information: Binary Padding
Credential Access	T1003.002	OS Credential Dumping: Security Account Manager
Discovery	T1082	System Information Discovery
	T1033	System Owner/User Discovery
	T1049	System Network Connections Discovery
Lateral Movement	T1021.002	SMB/Windows Admin Shares
Command and Control	T1071.001	Application Layer Protocol: Web Protocols
	T1132.001	Data Encoding: Standard Encoding
Exfiltration	T1041	Exfiltration Over C2 Channel
