Analysis Report on Lazarus Threat Group's Volgmer and Scout Malware

asec.ahnlab.com/en/57685/

By Sanseo October 13, 2023

Table of Contents

The seemingly state-sponsored Lazarus threat group has records of activity that date back to 2009. In the early days, their activities were mostly focused on Korea, but since 2016, the group has been attacking the defense, advanced technology, and finance sectors worldwide. The Lazarus group usually employed spear phishing and supply chain attacks, usually disguising the malware as legitimate programs in their attack process. **[1]**

For the last few years, the group launched watering hole attacks to attack multiple Korean enterprises and organizations in the fields of defense, satellite, software, and media. Their method for initial access involved the exploitation of a security vulnerability of a Korean financial security certification software. **[2]** Even after initial access, the threat actor exploited vulnerabilities in web security software or enterprise asset management programs during lateral movement. **[3]** The Lazarus group attacks not only ordinary PCs but also server systems for the purpose of using them as malware distribution or C&C servers. **[4] [5]**

Because the Lazarus threat group has been active since a long time ago, there are many attack cases and various malware strains are used in each case. In particular, there is also a wide variety of backdoors used for controlling the infected system after initial access. AhnLab Security Emergency response Center (ASEC) is continuously tracking and analyzing attacks by the Lazarus group, and in this post, we will analyze Volgmer and Scout, the two major malware strains used in their attacks.

Volgmer is a backdoor that has been used by the Lazarus threat group since 2014. Volgmer, which usually runs by being registered as a service, is installed with a name that disguises it as a legitimate file. It differs from other malware in the fact that it encrypts and saves the configuration data in the registry key "HKLM\SYSTEM\CurrentControlSet\Control\WMI\Security". ASEC identified that since 2014, Volgmer underwent many changes and has been used in attacks until about 2021. We also confirmed that since 2022, a downloader named Scout has been used in attacks instead of Volgmer. The basic operating mechanism of Scout is similar to the previous one, with the only difference in the actual features. The payload it downloads is presumed to be a backdoor for controlling the infected system.

```
; PDB File Name : Y:\Development\RT\Windows\Scout\Scout v2.1\Engine\Engine\x64\Penetrator\Engine.pdb
; OS type       :  MS Windows
; Application type:  DLL
```

Figure 1. PDB information of the Scout downloader

Scout has been in use for attacks since around 2022. While there are many instances where the specific attack cases could not be confirmed, there are cases where the initial access process was identified. For example, it was found alongside other pieces of malware in the attack case mentioned above, where a security vulnerability of a Korean financial security certification software was exploited. Much like the Lazarus group's ordinary activities, its targets include multiple Korean enterprises and organizations in the defense, manufacturing, ICT, and financial sectors. The threat actor used this malware to control the infected systems. There has also been a case of BYOVD (Bring Your Own Vulnerable Driver), where the threat actor leveraged a vulnerable driver module of a hardware supplier to disable security products. **[6] [7]**

This blog post will analyze the initial version of Volgmer backdoor that was identified first and the later version that began to be used in attacks in around 2017. Afterward, we will analyze the Scout downloader and also cover the dropper that was used to install Scout.

1. Analysis of Volgmer Backdoor

The oldest record regarding Volgmer is presumed to be the "Trojan.Volgmer" malware analysis page published by Symantec in 2014. **[8]** (link currently unavailable) Volgmer continued to be used in later attacks, and in 2017, CISA (Cybersecurity and Infrastructure Security Agency) of the U.S. also mentioned Volgmer when they disclosed the malware used by the Lazarus group. **[9]** (link currently unavailable) According to AhnLab's ASD logs, the Volgmer malware type disclosed by Symantec was detected from at least 2014 to 2015, and there are records of a similar variant being used in attacks until 2016.

An updated version of Volgmer was found to have been used since 2017, and there were records of its use until around 2021. As determined by the comparison of the C&C command routines despite a few differences, this type can be considered as the same type as the

backdoor used in the attack case shared by Kaspersky in 2021 where the backdoor was disguised as a DeFi application. **[10]** There were no other cases of Volgmer being used in attacks after the emergence of the Scout downloader around 2022.

Here we will analyze the initial version of Volgmer in the past before analyzing the later version of Volgmer used between 2017 and 2020. The initial version of Volgmer will be briefly analyzed even if it is an old malware strain, since there are many functional similarities with other malware that came after it. Subsequently, the later version of Volgmer will be analyzed; while this type has a different C&C command routine, its flow of operation is notably almost identical to the past version of Volgmer.

1.1. Initial Version of Volgmer

1.1.1. Analysis of Volgmer Dropper

Because Volgmer is a DLL-type backdoor, it needs malware that installs it. A dropper was identified alongside the initial version of Volgmer, and this dropper installs Volgmer by creating a password-protected compressed version in the resource area before registering it as a service. The dropper also checks the number of arguments, recognizes Korean operating environments, and even checks the version of Windows operating environments, and if these do not match pre-configured conditions, it either displays a message box or deletes itself. A batch file is used for self-deletion, and the use of the file name “pdm.bat” is notable.

The encrypted configuration data is decrypted during execution. This contains the registry key which will include the configuration data with the C&C server addresses, the string used to register the malware as a service, and the file “pdm.bate” to be used for self-deletion. The 0x10 byte-sized key used for decryption is still used by the malware from the Andariel group, a subsidiary group of Lazarus. **[11]**

> **Key**: 74 61 51 04 77 32 54 45 89 95 12 52 12 02 32 73

One of the characteristics of Volgmer is that it follows a certain logic to randomly generate strings for the name of the Volgmer DLL file to be created, as well as the name and description of the service to be registered. These strings are created by combining the following strings contained in the decrypted configuration data.

- **String A**: svc, mgmt, mgr, enum, app, bg, c, d, ex, f, g, h, i, k, l, m, net, o, p, q, rm, sec, ti, up, vol, win, dc, ud
- **String B**: Service, Management, Manager, Enumerator, Application, Background, Control, Desktop, Extension, Function, Group, Host, Intelligent, Key, Layer, Multimedia, Network, Operation, Portable, Quality, Remote, Security, TCP/IP, User Profile, Volume, Windows, Device, Update

For example, the file name is a combination of four selections from the “String A” list, resulting in names such as “hlrmenum.dll”. Likewise, service-related items are created by combining four selections from the “String B” list, as follows.

- **Service name**: “[Host Layer Remote Enumerator]”
- **Service description**: “The [Host Layer Remote Enumerator] is an essential service for management of Windows System. If the service is stopped or disabled, Windows will be able to damaged seriously.”
- **Service DLL path**: “C:\Windows\system32{hlrmenum}.dll”

The Volgmer dropper decompresses “MYRES” in the resource area to obtain the Volgmer DLL and configuration files. These files are compressed with the ZIP compression algorithm and password-protected with the following password.

Password for the compressed file: “!1234567890 dghtdhtrhgfjnui$%^^&fdt”

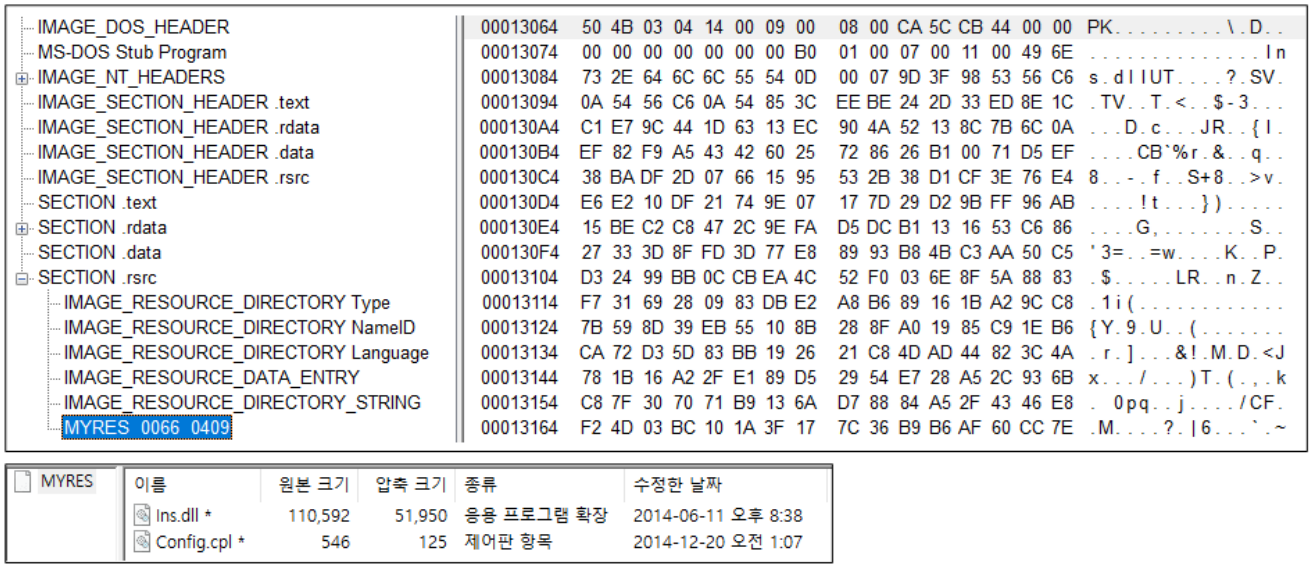

Figure 2. Password-protected compressed DLL and configuration file in the resource area

After creating the Volgmer DLL in the path %SystemDirectory%, the dropper sets the time configuration information to be the same as the Notepad (notepad.exe) file. This timestomping is one of the major anti-forensic techniques employed for the purpose of evading timeline analysis. Besides the timestomping technique, the Lazarus group uses a variety of anti-forensic techniques such as file deletion and data concealment in their attack process, and this trend continues to this day. **[12]**

Before being written into the registry key, the decompressed configuration file is encrypted with the same method as the algorithm used for decrypting the configuration data. This data includes C&C server addresses and is later read, decrypted, and used by Volgmer. After these processes are complete, it uses the generated service configuration data to register Volgmer as a service and executes it.

- **Registry Key – 1**: HKLM\SYSTEM\CurrentControlSet\Control\WMI\Security / 125463f3-2a9c-bdf0-d890-5a98b08d8898
- **Registry Key – 2**: HKLM\SYSTEM\CurrentControlSet\Control\WMI\Security / f0012345-2a9c-bdf8-345d-345d67b542a1

1.1.2. Analysis of Volgmer Backdoor

Volgmer, running as a service, decrypts the registry value above to obtain the configuration data. As shown below, the configuration data consists of the signature string “cgi_config”, the ID, and C&C server addresses. Additionally, the ID value is NULL when the dropper is generated, but afterward, Volgmer uses the infected system’s hardware information to create an ID value.

Offset	Size	Data
0x00	0x0A	“cgi_config”
0x0A	0x08	Victim ID
0x12	0x04	The first C&C server’s IP address
0x16	0x04	The first C&C server’s port number
0x1A	0x04	The second C&C server’s IP address
0x1E	0x04	The second C&C server’s port number
…	…	…

Table 1. Configuration data

Volgmer obtains one of the C&C servers from the list in the configuration data and connects to it. It then transmits an HTTP packet, which is created using a random combination of strings, similar to when the service and file name were created. One HTTP request method is selected among “GET”, “POST”, or “HEAD”, and one of the eight User Agent strings is selected. A characteristic of Volgmer is that the User Agent strings contain a typo, where it reads “Mozillar” instead of “Mozilla”. After these processes, it transmits an arbitrarily selected HTTP packet, and then uses the RIPEMD-160 hash to perform the verification process with the C&C server.

```
.data:10016064 off_10016064    dd offset aGet          ; DATA XREF: fn_makeUserAgent2+4E↑r
.data:10016064                                        ; "GET "
.data:10016068                 dd offset aPost         ; "POST "
.data:1001606C                 dd offset aHead         ; "HEAD "
.data:10016070 off_10016070    dd offset aTp11         ; DATA XREF: fn_makeUserAgent2:loc_10009B5B↑r
.data:10016070                                        ; fn_makeUserAgent1:loc_10009EE6↑r
.data:10016070                                        ; "TP/1.1"
.data:10016074                 dd offset aTp10         ; "TP/1.0"
.data:10016078 off_10016078    dd offset aAmd64        ; DATA XREF: fn_makeUserAgent2:loc_10009BE8↑r
.data:10016078                                        ; "AMD64\r\n"
.data:1001607C                 dd offset aAmd32        ; "AMD32\r\n"
.data:10016080 off_10016080    dd offset aAcceptEncoding
.data:10016080                                        ; DATA XREF: fn_makeUserAgent2+226↑r
.data:10016080                                        ; "Accept-Encoding: gzip, deflate\r\n"
.data:10016084                 dd offset aAcceptEncoding_0 ; "Accept-Encoding: compress, deflate\r\n"
.data:10016088                 dd offset aAcceptEncoding_1 ; "Accept-Encoding: deflate\r\n"
.data:1001608C                 dd offset aAcceptEncoding_2 ; "Accept-Encoding: gzip, compress, deflat"...
.data:10016090                 dd offset aAcceptEncoding_3 ; "Accept-Encoding: gzip, compress\r\n"
.data:10016094 off_10016094    dd offset aUserAgentMozil
.data:10016094                                        ; DATA XREF: fn_makeUserAgent2+25F↑r
.data:10016094                                        ; "User-Agent: Mozillar/5.0 (compatible; M"...
.data:10016098                 dd offset aUserAgentMozil_0 ; "User-Agent: Mozillar/5.0 (compatible; M"...
.data:1001609C                 dd offset aUserAgentMozil_1 ; "User-Agent: Mozillar/5.0 (compatible; M"...
.data:100160A0                 dd offset aUserAgentMozil_2 ; "User-Agent: Mozillar/5.0 (compatible; M"...
.data:100160A4                 dd offset aUserAgentMozil_3 ; "User-Agent: Mozillar/5.0 (compatible; M"...
.data:100160A8                 dd offset aUserAgentMozil_4 ; "User-Agent: Mozillar/5.0 (compatible; M"...
.data:100160AC                 dd offset aUserAgentMozil_5 ; "User-Agent: Mozillar/5.0 (compatible; M"...
.data:100160B0                 dd offset aUserAgentMozil_6 ; "User-Agent: Mozillar/5.0 (compatible; M"...
.data:100160B4                 dd offset aUserAgentMozil_4 ; "User-Agent: Mozillar/5.0 (compatible; M"...
.data:100160B8                 dd offset aUserAgentMozil_4 ; "User-Agent: Mozillar/5.0 (compatible; M"...
.data:100160BC off_100160BC    dd offset aCom          ; DATA XREF: fn_makeUserAgent2+2E8↑r
.data:100160BC                                        ; ".com\r\n"
.data:100160C0                 dd offset aOrg          ; ".org\r\n"
.data:100160C4                 dd offset aUn           ; ".un\r\n"
```

Figure 3. Strings used for creating the HTTP packet

After the verification process with the C&C server is complete, information on the infected system is transmitted over two batches. The first includes information such as whether or not the currently running system is a virtual machine, currently running security programs, and installed software.

Offset	Size	Data
0x00	0x08	ID
0x08	0x08	NULL
0x10	0x04	Execution Flag
0x14	0x04	Time
0x18	0x04	Scan for virtual machine environments
0x1C	0x04	Scan for installed software
0x20	0x04	Scan for security programs
0x24	0x04	Scan for debugging
0x28	0x04	Scan to check if it is running in the svchost.exe process

Table 2. Data transmitted to the C&C server – 1

Next, it collects a variety of information such as the computer name, network information, hardware information, language, installed antivirus software, and running services, before sending them to the C&C server. Additionally, running services are scanned through the port that is currently being listened to; targets include FTP, SSH, DNS, HTTP, SMB, RDP, MS-SQL, and VNC.

Offset	Data
0x0000	IP address
0x0004	Computer name
0x0084	CPU information
0x0184	Number of processors
0x0188	Windows version
0x02D0	MAC address
0x02D6	Malware name
0x0316	Malware file name
0x0358	Sleep time
0x035C	Installed antivirus software
0x0360	Locale information
0x03E0	List of services in use
0x03E8	“DING”
0x03EC	“PADD”
0x03F0	“INGX”
0x03F4	“XPAD”

Table 3. Data transmitted to the C&C server – 2

Afterward, Volgmer can receive commands from the C&C server to run features such as file-related tasks, command execution, reverse shell, etc.

Command	Feature
0x1000	Transmit system information

Command	Feature
0x1009	Modify C&C address list (registry key)
0x100A	Transmit C&C address list
0x100B	Download file
0x100C	Upload and delete file
0x100D	Upload file
0x100E	Execute file
0x100F	Download and execute file
0x1010	Delete file
0x1011	Set sleep time
0x1012	Reverse shell

Table 4. C&C commands

There is also a variant of the initial version of Volgmer. While it has the same C&C command routines, the major differences include the fact that the signature string used in the configuration data was “config_reg” instead of “cgi_config” and that the registry key where the configuration data is saved was changed to the following.

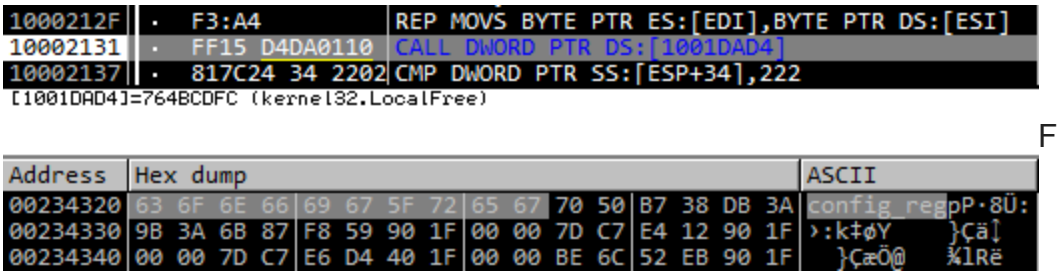

Figure 4. Changed signature string

- **Registry Key – 1**: HKLM\SYSTEM\CurrentControlSet\Control\WMI\Security / 2d54931A-47A9-b749-8e23-311921741dcd
- **Registry Key – 2**: HKLM\SYSTEM\CurrentControlSet\Control\WMI\Security / c72a93f5-47e6-4a2a-b13e-6AFE0479cb01

1.2. Later Version of Volgmer

A new version of Volgmer started being used in 2017. While there are differences in the C&C command routine between this and the initial version, there are various similarities including the major characteristic that the configuration data is encrypted and saved in the registry key “HKLM\SYSTEM\CurrentControlSet\Control\WMI\Security” for use. Other similarities can be

seen in the fact that the dropper runs after being registered as a service because it is in a service DLL format and that Volgmer DLL's file name or the strings used when registering it as a service are created by randomly combining certain strings.

The later version of Volgmer covered in this section has the same C&C command routine as the backdoor in the report released by Kaspersky in 2021. **[13]** Because the Lazarus group is known for using a variety of backdoors, it is presumed that the backdoors that were often used in attacks in Korea were altered and used in other attacks as well. Here we will analyze the later version of Volgmer backdoor DLL, and the malware thought to be the dropper that installs this will be covered alongside the analysis of Scout later on.

1.2.1. Analysis of Volgmer Backdoor

The later version of Volgmer decrypts the configuration data saved in a certain registry key to obtain the C&C addresses. When creating the Volgmer backdoor, the dropper creates a file name by randomly combining certain strings and uses the Hex value of the first four letters of this file name as the registry value where the configuration data will be stored. This will be covered in more detail later on. Accordingly, Volgmer references the first four letters of its own file name and reads the following registry value while running.

> **Registry Key**: HKLM\SYSTEM\CurrentControlSet\Control\WMI\Security / [First four letters of the file name]-5903-ed41-902f-e93a29dafef5

The read data is decrypted using the RC4 algorithm. While later versions of Volgmer all use the RC4 algorithm, they are largely divided into two types depending on the decryption method. The first is a type that uses a manually implemented RC4 algorithm, and the other type uses Crypto API to obtain the SHA-1 hash and uses the resulting value to perform RC4 decryption. The method of manually-implemented RC4 algorithm uses a 4-byte key for decryption, and the routine that uses Crypto API uses a 4-byte key to obtain the SHA-1 hash before using this as the RC4 key. Additionally, the first 0x10 size of the SHA-1 value is used as the RC4 algorithm key. For example, the SHA-1 hash of the value "DE A7 00 00" is "8f919e6d8970faede0b10cfd5f82da53a83ca34d", but the value "8766fe8380b144907efa286a814c2241″ is used as the RC4 key.

```
if ( !CryptAcquireContextW(&phProv, 0i64, L"Microsoft Enhanced Cryptographic Provider v1.0", 1u, 0)
  && !CryptAcquireContextW(&phProv, 0i64, L"Microsoft Enhanced Cryptographic Provider v1.0", 1u, 8u) )
{
  return 0i64;
}
if ( !CryptCreateHash(phProv, 0x8004u, 0i64, 0, &phHash) )// CALG_SHA1 : 0x00008004
{
LABEL_9:
  CryptReleaseContext(phProv, 0);
  return 0i64;
}
if ( !CryptHashData(phHash, a3, 4u, 0) || !CryptDeriveKey(phProv, 0x6801u, phHash, 0, phKey) )// CALG_RC4 : 0x00006801
{
LABEL_8:
  CryptDestroyHash(phHash);
  goto LABEL_9;
}
if ( !CryptEncrypt(phKey[0], 0i64, 1, 0, pbData, &pdwDataLen, dwBufLen) )
{
  CryptDestroyKey(phKey[0]);
  goto LABEL_8;
```

Figure 5. Decryption routine using the SHA-1 hash and RC4 algorithm

- **RC4 Key (manually implemented)**: E2 28 00 00
- **RC4 Key (Crypto API)**: DE A7 00 00

Volgmer selects one of the C&C server addresses from the configuration data and connects to it. The later version of Volgmer and the Scout downloader to be covered later use the HTTP protocol to communicate with the C&C server. All identified C&C addresses used https. The POST method is used for initial connection to the C&C server or when receiving commands from it; different parameters have been used depending on the point in time. The most recent version of Volgmer which appeared after 2020 uses a similar parameter as that mentioned in Kaspersky's report of 2021. Additionally, the period and types mentioned in this blog post only refer to instances where cases of attacks have been identified and may differ depending on the cases that have not actually been confirmed.

Period	Parameter Format
2017 – 2019	secgb=[param1]&secdata=[param2]
2019	sessions=[param1]&secinfo=[param2]
2020	jsessid=[param1]&cookie=[param2]

Table 5. Format of the HTTP request to the C&C server

Body

Name	Value
secgb	00000141f
secdata	60d49d98785e1a520001777faAB0

Figure 6. Packet used in the authentication process with the C&C server

Period	Request Type	Parameter #1	Parameter #1 Structure	Parameter #2	Parameter #2 Structure

Period	Request Type	Parameter #1	Parameter #1 Structure	Parameter #2	Parameter #2 Structure
2017 – 2019	Initial access	secgb	Random (0x9)	secdata	message ID – “60D49D98” (0x08), victim ID (0x08), Random (0x8) + C&C address (Base64)
	C&C communication	secgb	Random (0x9)	secdata	message ID – “60D49D99” (0x08), victim ID (0x08), Random (0x8) + 0x60D49D94 (RC4)
	Send results	secgb	Random (0x9)	secdata	message ID – “60D49D99”, etc. (0x08), victim ID (0x08), Random (0x8) + Data (RC4)
2019	Initial access	sessions	Random (0x6)	secinfo	message ID – “60D49D98” (0x08), victim ID (0x08), Random (0x8) + C&C address (Base64)
	C&C communication	sessions	Random (0x6)	secinfo	message ID – “60D49D99” (0x08), victim ID (0x08), Random (0x8) + 0x60D49D94 (RC4)
	Send results	sessions	Random (0x6)	secinfo	“60D49D99” (0x08), victim ID (0x08), Random (0x8) + Data (RC4)

Period	Request Type	Parameter #1	Parameter #1 Structure	Parameter #2	Parameter #2 Structure
From 2020 onwards	Initial access	cookie	Random (0x10) + 0x60D49D94 (Base64) + Random (0x10)	jsessid	message ID – “60D49D99” (0x08), victim ID (0x08), Random (0x8)
	C&C communication	cookie	Random (0x10) + 0x60D49D94 (RC4) + Random (0x10)	jsessid	message ID – “60D49D99” (0x08), victim ID (0x08), Random (0x8)
	Send results	cookie	Random (0x10) + Data (RC4) + Random (0x10)	jsessid	message ID – “60D49D99” (0x08), victim ID (0x08), Random (0x8)

Table 6. Argument parameters

The message ID “60D49D98″ is used for initial communication with the C&C server. For the data, the C&C server address is encrypted in Base64 and transmitted. Afterward, the 0x60D49D94 value is transmitted with the message ID “60D49D99″ to receive commands. The value 0x60D49D94 is not only used for requests but also for responding. This is because Volgmer verifies communications with the C&C server by checking whether the decrypted response value is 0x60D49D94.

message ID	Feature
60D49D98	Establish connection
60D49D99	Request command
60D49DB6	Transmit system information

Table 7. Types of msg IDs

The Base64 algorithm is used for data encryption upon initial authentication. Afterward, the RC4 algorithm is used for encryption and decryption. There are types of Volgmer that also manually implement the RC4 algorithm or use the Crypto API, and Volgmer is particular for using a different RC4 key for each type.

- **RC4 Key (manually implemented)**: 8D 52 00 00
- **RC4 Key (Crypto API)**: A3 D5 00 00

Volgmer provides features to control the infected system, much like a typical backdoor. The following is a list of commands for a certain version of Volgmer backdoor. Volgmer types categorized as the later version mostly support the same commands.

Command	Feature
0x60D49D94	Default response
0x60D49D95	Default
0x60D49D97	Set the wait time using the default value
0x60D49D9F	Set the wait time using the received value
0x60D49DA0	Transmit system information (computer name, Windows version, architecture, IP information, etc.)
0x60D49DA1	Look up drive information
0x60D49DA2	Look up list of files
0x60D49DA3	Look up list of processes
0x60D49DA4	Terminate process
0x60D49DA5	Set the current task path
0x60D49DA6	Scan (connect to the received address)
0x60D49DA7	Timestomping
0x60D49DA8	Reverse shell
0x60D49DA9	Delete file
0x60D49DAA	Execute program
0x60D49DAB	Execute program (with certain privileges)
0x60D49DAC	Execute program (as an administrator)
0x60D49DAD	Download files
0x60D49DAE	Transfer file contents
0x60D49DAF	Transfer file (compressed in cab format)
0x60D49DB0	Look up directory

Command	Feature
0x60D49DB1	Send configuration data
0x60D49DB2	Download and apply configuration data
0x60D49DB3	Apply the current time to the configuration data
0x60D49DB4	Sleep
0x60D49DB5	Transmit information (module name, etc.)
0x60D49DB7	Delete and terminate the “CMb*.a-p” file

Table 8. List of commands

A notable point is that the aforementioned timestomping and file deletion techniques often employed by the Lazarus group are supported as commands. The timestomping command changes the timestamp of the file at a path received from the C&C server to the timestamp of another file in another path received alongside the first piece of information. Instead of simply deleting the file, the file deletion command overwrites it with the value “0x5F 00 00 00 00 …” before deletion to prevent recovery.

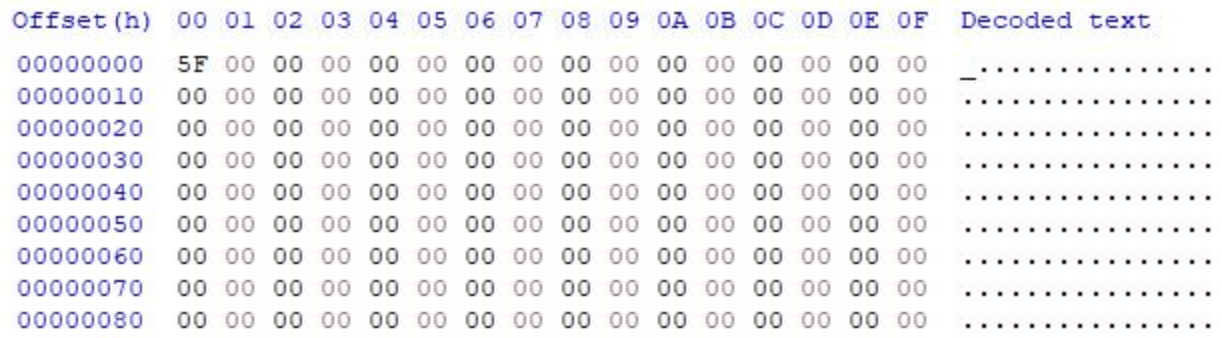

Figure 7. File overwritten before being deleted

Although there are records of PebbleDash being recently used by another threat group, it is by default a backdoor known to have been used by the Lazarus group. When a new drive or session is created, PebbleDash supports the feature of terminating the wait routine and initializing communication with the C&C server (i.e., being activated). **[14]** This is because PebbleDash has a long wait time in the process of communicating with the C&C server and it is difficult for it to respond to changes in the infected system in real-time. These features are also found in other backdoors of the Lazarus group, with the major one being Volgmer. Aside from Volgmer, there was a case where this feature was supported by the OpenCarrot backdoor mentioned in a report from SentinelOne. **[15]**

Additionally, among the files that Volgmer deletes periodically, the file “BIT*.tmp” is presumed to be for deleting the CAB file which is created while executing the 0x60D49DAF command responsible for the file transfer feature. However, the file named “CMb*.a-p” deleted by the 0x60D49DB7 command is not observed with the current analysis target, Volgmer, alone.

2. Analysis of Scout Downloader

Following its first use in 2014, Volgmer was used until around 2021. In 2022, a downloader began being detected. This is similar to Volgmer, but instead of having backdoor features, it is a downloader that downloads another malware from an external source and executes it in the memory area. While the downloaded payload could not be procured, there are three notable points about it. First is that it is being detected after the end of Volgmer's active period. The second is that its communication method with the C&C server and loading of the configuration data are the same as Volgmer. Lastly, it also has records of being created by a similar dropper.

This malware is largely categorized into two types depending on the routine. One was mainly observed in the first half of 2022 and the other was distributed from late 2022 to 2023. The second type has a few more routines added in comparison to the first, and as mentioned in the PDB information above, the presence of the keyword "Scout" and the version v2.x allow us to assume that this is an improved version of Scout that was distributed in the first half of 2022. Accordingly, we will classify the type distributed in the first half of 2022 as Scout v1 and the type being identified from late 2022 to 2023 as Scout v2. Additionally, the period and types mentioned in this blog post only refer to instances where cases of attacks have been identified and may differ depending on the cases that have not actually been confirmed.

Most Windows versions of malware are created with a character user interface (CUI) to run in the background without the user being aware. The Scout downloader characteristically creates a window when running, like Graphical User Interface (GUI) programs. Of course, the window size is set to 0 and is not actually noticeable to the user, and this is likely to disguise the malware as a legitimate program.

```
while ( wcsnicmp(&data_config2[260 * v32], L"http", 4ui64) )
{
  if ( ++v32 >= 5 )
    return 0i64;
}
*&WndClass.cbClsExtra = 0i64;
WndClass.hbrBackground = GetStockObject(4);
WndClass.hCursor = LoadCursorW(0i64, 0x7F00);
WndClass.lpszMenuName = 0i64;
WndClass.hIcon = LoadIconW(0i64, 0x7F00);
WndClass.hInstance = hInstance;
WndClass.lpfnWndProc = fn_Wndproc;
WndClass.lpszClassName = L"Windows";
WndClass.style = 3;
RegisterClassW(&WndClass);
hWnd = CreateWindowExW(0, L"Windows", L"Windows", 0xCF0000u, 0, 0, 0, 0, 0i64, 0i64, hInstance, 0i64);
ShowWindow(hWnd, 0);
TickCount64 = GetTickCount64();
srand(TickCount64);
v34 = rand();
data_flag = 'ecqc';
data_rand0_4 = v34 % 5;
SendMessageW(hWnd, 0x5450u, 0i64, 0i64);
while ( GetMessageW(&Msg, hWnd, 0, 0) )
{
  TranslateMessage(&Msg);
  DispatchMessageW(&Msg);
```

Figure 8. The routine that creates a window titled “Windows”

Unlike the Volgmer backdoor that saved the configuration data in the registry key, there is a type of Scout downloader where the configuration data is in the Overlay area at the end of the file. In this case, Scout uses the string transmitted as an argument for the decryption key.

```
0001FFC0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0001FFD0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0001FFE0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0001FFF0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00020000  51 BC F9 54 07 1E 50 55 75 98 A3 09 68 AA BB A2  Q¼ùT..PUu˜£.hª»¢
00020010  8C B7 AF 1F 30 92 D6 AF C7 22 5E CB 96 21 DB 2A  Œ·¯.0'Ö¯Ç"^Ë–!Û*
00020020  C7 12 66 CC 22 BE C8 98 90 5D 8D 5B 9E CA 5B 5A  Ç.fÌ"¾È˜.].[žÊ[Z
00020030  AC 05 25 85 09 F1 D8 B9 54 FE A2 F0 1E D0 56 E1  ¬.%….ñØ¹Tþ¢ð.ÐVá
00020040  4C 3D 83 FF 6D FC A9 5E E8 2C 47 4A E1 82 BF 9E  L=ƒÿmü©^è,GJá‚¿ž
```

Figure 9. Configuration data in the Overlay area

In this section, we will analyze the Scout downloader; for this, we will first cover the dropper that installs Scout. This dropper contains the actual malware and registry value in the Overlay area at the end of the file, and as these are encrypted with the RC4 algorithm, the string given as an argument is used as the RC4 key for decryption. While the argument has not been identified, there are AhnLab ASD logs that show Scout having been created.

action	parentProcess_path	currentProcess_path	currentProcess_size	targetFile_path	targetFile_size
CreateFile	%SystemRoot%\system32\cmd.exe	%SystemRoot%\wagent.dat	374928	%SystemRoot%\system32\gpkImgmt.dll	224772
SetFileAttributes	%SystemRoot%\system32\cmd.exe	%SystemRoot%\wagent.dat	374928	%SystemRoot%\system32\gpkImgmt.dll	224772

Figure 10. The dropper that creates Scout

In addition, an examination of the same type of dropper observed in around 2021 reveals a routine that configures the registry key used by the Volgmer backdoor, and it also uses the same RC4 key for encrypting the registry value. This shows that the threat actor installed the

Volgmer backdoor until 2021 and from 2022 used the same dropper to install the Scout downloader.

2.1. Dropper (Volgmer, Scout)

The dropper is classified as either the Volgmer dropper or Scout dropper depending on the malware it creates. The differences are in the registry key for writing configuration data to and the RC4 key for encrypting the configuration data. Other than those, the actual routines are identical. However, it is divided into the injector type and the service registration type according to the way it installs the malware.

Like the initial version of the Volgmer dropper, the injector-type dropper creates the file name randomly; a random 2-5 character string is randomly generated, then one of the “svc”, “mgr”, or “mgmt” strings is selected randomly and added. If the generated file name is “bnsvc.dll”, the hex value of the first four letters is set as the registry value where the configuration data will be saved. For example, the hex value of “bnsv” is “62 6E 73 76″. In this case, the registry value where Volgmer and Scout’s configuration data is to be stored is as follows.

- **Volgmer**: HKLM\SYSTEM\CurrentControlSet\Control\WMI\Security / “626e7376-5903-ed41-902f-e93a29dafef5”
- **Scout**: HKLM\SYSTEM\CurrentControlSet\Control\WMI\Security / “626e7376-2790-10f2-dd2a-d92f482d094f”

Afterward, it uses the string given as an argument for the RC4 key, decrypts the encrypted DLL (i.e., Volgmer or Scout) added to the overlay area at the end of the file, and creates it under the name generated above in the %SystemDirectory% directory. The configuration data that contains the C&C addresses is also decrypted and saved in the registry value created in the stage above. Because the malware created in the %SystemDirectory% directory has its timestamp set to recent time, this is changed to the timestamp information of the calculator (i.e., calc.exe).

It is unclear whether the malware runs as intended after all the procedures up to this point, but the service is registered to the registry settings of the security package, and then the created DLL is injected into the lsass.exe process.

> **Security Package Registry Key**: HKLM\SYSTEM\CurrentControlSet\Control\Lsa / Security Packages

The service registration type is different starting in the name generation routine. First, it obtains the netsvcs service group from the following registry key, then searches for each service in the “HKLM\SYSTEM\CurrentControlSet\Services\” entry and selects a service that is not currently registered.

> HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Svchost / netsvcs

If the selected service is “LogonHours”, it is registered to the netsvcs service, and the service DLL is created in the %SystemDirectory% directory under the name “LogonHourss.dll” with an extra “s”.

```
RegSetValueExW(hKey, L"ServiceDll", 0, 2u, (const BYTE *)data_dllPath, 2 * v0 + 2);
RegCloseKey(hKey);
RegCloseKey(phkResult);
v3 = OpenSCManagerW(0i64, 0i64, 0xF003Fu);
ServiceW = CreateServiceW(
             v3,
             Destination,
             DisplayName,
             0xF01FFu,
             0x20u,
             2u,
             1u,
             L"%SystemRoot%\\System32\\svchost.exe -k netsvcs -p",
             0i64,
             0i64,
             0i64,
             L"LocalSystem",
             0i64);
v5 = ServiceW;
if ( ServiceW )
{
  StartServiceW(ServiceW, 0, 0i64);
```

Figure 11. The routine for service registration

2.2. Analysis of Scout Downloader

2.2.1. Scout Downloader V1

Like Volgmer, Scout performs a file name-based lookup of the registry value where the encrypted configuration data is saved. RC4 is used for the encryption algorithm, and identified instances of Scout all use Crypto API to decrypt the configuration data.

- **Registry Key**: HKLM\SYSTEM\CurrentControlSet\Control\WMI\Security / [First four letters of the file name]-2790-10f2-dd2a-d92f482d094f
- **RC4 Key (Crypto API)**: F9 A3 DE 48

After decrypting the configuration data, it creates a window titled “Windows” as mentioned above. Its actual routine is implemented in the registered procedure. Scout characteristically has each feature implemented based on Windows messages. For example, first, it uses the SendMessageW() API to send a 0x5450 message which branches out into the 0x5451 message. The 0x5451 message connects to the C&C server and branches out into different routes depending on whether the authentication was successful or not.

```
    data_rand0_4 = (data_rand0_4 + 1) % 5;
    if ( v9 >= 5 )
    {
      v16 = 0x546D;
      goto LABEL_39;
    }
  }
case 0x5452u:
  fn_downNload(hWnd);
  return DefWindowProcW(hWnd, Msg, wParam, lParam);
case 0x5453u:
  if ( hInternet )
  {
    InternetCloseHandle(hInternet);
    hInternet = 0i64;
  }
  if ( hConnect )
  {
    InternetCloseHandle(hConnect);
    hConnect = 0i64;
  }
  if ( qword_18001FA08 )
  {
    InternetCloseHandle(qword_18001FA08);
    qword_18001FA08 = 0i64;
  }
  data_rand0_4 = (data_rand0_4 + 1) % 5;
  Sleep(60000 * data_sleep);
  break;
case 0x546Du:
  buf_res[0] = 0i64;
```

Figure 12. Windows message-based routine

Message Number	Feature
0x5450	Starting routine
0x5451	Connect to the C&C server and authenticate
0x5452	Download additional payload and execute it in the memory area (PE)
0x5453	Reset the C&C server address and reattempt to connect to the server
0x546D	Reattempt to connect to the C&C server

Table 9. Features of each message – Scout v1

Unlike Volgmer, the initial version of Scout had a single parameter and the string “param” was used. As it is a downloader, it has a simple structure with only two message IDs used: “184D0382” and “0E8AFD28”. The string “cqce” is encrypted and transmitted during the initial authentication process, and in order for the C&C server authentication process to be successful, the transmitted data string must be “1111”.

Request Type	Parameter	Parameter Structure	Data Structure

Request Type	Parameter	Parameter Structure	Data Structure
Initial access	param	Random (0x8), victim ID (0x08), message ID – “184D0382” (0x08), Data (Base64)	Random (0x4), “cqce”, C&C URL
Download	param	Random (0x8), victim ID (0x08), message ID – “0E8AFD28” (0x08), Data (RC4)	Random (0x4), configuration data

Table 10. Format of the HTTP request to the C&C server

Figure 13. Packet used in the authentication process with the C&C server

The RC4 key used in communication with the C&C server is a 0x20 byte-sized key instead of the 4-byte key used for decrypting the configuration data. The download process involves first receiving the size of the payload to be downloaded. At this stage, the 0x20 byte-sized RC4 key is used, and when the encrypted payload with the defined size is downloaded, a 0x20 byte NULL data value is used as the RC4 key.

> **RC4 Key (Crypto API)**: 54 A6 BA C3 13 98 DB 1A 62 45 23 12 A8 83 71 82 4E 74 D2 38 00 00 00 00 00 00 00 00 00 00 00 00

The payload downloaded and decrypted with the RC4 key scans the “MZ” signature, then executes it in the memory area.

2.2.2. Scout Downloader V2

Since the second half of 2022, types with a few more routines added have been observed, but the actual features are the same. Among these types, there are multiple pieces of malware for which the PDB information still exists; this shows that the threat actor named the malware “Scout”.

```
Y:\Development\RT\Windows\Scout\Scout v2.1\Engine\Engine\x64\Penetrator\Engine.pdb
Y:\Development\RT\Windows\Scout\Scout v2.1\Engine\Engine\x64\Lsass\Engine.pdb
Y:\Development\RT\Windows\Scout\Scout v2.1\Engine\Engine\x64\Netsvc\Engine.pdb
Z:\Development\RT\Windows\Scout\Scout v2.2\Engine\Engine\x64\Netsvc\Engine.pdb
Z:\Development\RT\Windows\Scout\Scout v2.3\Engine\Engine\x64\Lsass\Engine.pdb
```

Scout versions v2.x support more commands (i.e., messages) in comparison to the past versions of Scout. While the Windows messages of the past versions have a simple flow, in versions v2.x, it downloads messages from the C&C server and uses them as commands. Accordingly, it can also execute commands for changing the configuration data aside from downloading additional payloads, Also, past versions encrypted the string “cqce” and transmitted it to the C&C server. In versions v2.x, the first string “bqce” is transmitted, but depending on the message command received from the C&C server, the values “fqce”, “eqce”, “dqce”, or “cqce” can be transmitted.

Message Number	Feature
0x5450	Starting routine. Connect to the C&C server and authenticate. (Flag : “bqce”)
0x5451	Download message.
0x5452	Download additional payload and execute it in the memory area (Shellcode). Download message.
0x5453	Download configuration data.
0x5454	Change settings data.
0x5455	Download additional payload and execute it in the memory area (PE).
0x5456	Download additional payload and inject (PE).
0x5457	Reattempt to connect to the C&C server. (Flag : “eqce”)
0x5458	Reattempt to connect to the C&C server. (Flag : “fqce”)
0x5459	Self-delete.
0x545A	Reattempt to connect to the C&C server. (Flag : “cqce”)
0x545B	Reattempt to connect to the C&C server. (Flag : “dqce”)

Table 11. Features of each message – Scout v2

Figure 14. Added Windows messages

In comparison to the past versions, the method of communication with the C&C server for versions up to v2.2 used the “param” parameter, like the past versions. From v2.3, “jsessionid” is used as the parameter, and the RC4 key value is also different.

Request Type	Parameter	Parameter Structure	Data Structure
Initial access	jsessionid	Random (0x5), victim ID (0x08), message ID – “184D0382” (0x08) + Data (Base64)	Flag, C&C addresses
Downloading commands	jsessionid	Random (0x5), victim ID (0x08), message ID – “0E8AFD28” (0x08)	
Download settings	jsessionid	Random (0x5), victim ID (0x08), message ID – “0E8AFD28” (0x08) + Data (RC4)	Configuration data

Table 12. Format of the HTTP request to the C&C server

- **RC4 Key (param)**: 54 A8 BA C3 E3 98 DB 1A 6D 45 23 12 A8 83 71 82 4E 74 D2 38 00 00 00 00 00 00 00 00 00 00 00 00
- **RC4 Key (jsessionid)**: 73 D3 FE CC 23 AA 74 BA 53 47 88 32 73 11 19 AC FF D3 14 08 00 00 00 00 00 00 00 00 00 00 00 00

3. Conclusion

The Lazarus group is one of the very dangerous groups that are highly active worldwide, using various attack vectors such as spear phishing and supply chain attacks. Recently, the group exploited a security vulnerability in a Korean financial security authentication software in their initial access process and also exploited vulnerabilities in web security software or enterprise asset management programs in the lateral movement process.

Security managers of enterprises must identify assets that may be exposed to threat actors through attack surface management and always apply the latest security patches. Users must be particularly cautious of attachments to emails from unknown sources or executable files downloaded from web pages. Users should also apply the latest patch for OS and programs such as Internet browsers, and update V3 to the latest version to prevent such malware infection in advance.

File Detection

– Backdoor/Win.Lazardoor.C5233133 (2022.09.07.00)
– Backdoor/Win32.Agent.C3351518 (2019.07.25.00)
– Backdoor/Win32.Agent.R283184 (2019.07.25.00)
– Backdoor/Win64.Agent.C3371791 (2019.08.08.03)
– Data/BIN.Encoded (2022.10.05.00)
– Data/BIN.Encoded (2023.03.08.00)
– Data/BIN.EncPe (2022.09.07.00)
– Dropper/Win.Agent.C5499468 (2023.10.02.00)
– Dropper/Win32.Agent.C3371843 (2019.08.08.03)
– Dropper/Win64.Agent.C3371802 (2019.08.08.03)
– Malware/Win64.Generic.C4065063 (2020.04.16.07)
– Trojan/Win.Lazardoor.C4979367 (2022.02.24.02)
– Trojan/Win.Lazardoor.C4979368 (2022.02.24.03)
– Trojan/Win.Lazardoor.C5037872 (2022.03.31.00)
– Trojan/Win.Lazardoor.R474265 (2022.02.24.03)
– Trojan/Win.Lazardoor.R482731 (2022.04.07.01)
– Trojan/Win.Lazardoor.R495643 (2022.06.04.00)
– Trojan/Win.Lazardoor.R500179 (2022.06.24.00)
– Trojan/Win.LazarLoader.C5194304 (2022.07.06.01)
– Trojan/Win.LazarLoader.C5196326 (2022.07.06.03)
– Trojan/Win.LazarLoader.C5196363 (2022.07.06.04)
– Trojan/Win.LazarLoader.C5196414 (2022.07.07.00)
– Trojan/Win.LazarLoader.C5201772 (2022.07.11.03)
– Trojan/Win.LazarLoader.C5210732 (2022.07.19.00)
– Trojan/Win.LazarLoader.C5211408 (2022.07.20.01)
– Trojan/Win.LazarLoader.C5233120 (2022.09.07.00)
– Trojan/Win.LazarLoader.R480766 (2022.03.31.00)
– Trojan/Win.LazarLoader.R491208 (2022.05.10.02)
– Trojan/Win.LazarLoader.R500065 (2022.06.22.03)
– Trojan/Win.LazarLoader.R501218 (2022.06.28.03)
– Trojan/Win.Scout.R536659 (2022.11.30.00)
– Trojan/Win32.Agent.C876729 (2015.06.03.00)
– Trojan/Win32.Agent.R128643 (2014.12.17.00)
– Trojan/Win32.Akdoor.C3450548 (2019.08.29.04)
– Trojan/Win32.Backdoor.R174379 (2016.02.16.05)
– Trojan/Win32.Dllbot.C715400 (2015.02.12.04)
– Trojan/Win32.Ghost.C695717 (2015.01.27.05)
– Trojan/Win64.Agent.R274329 (2019.06.04.03)
– Trojan/Win64.Akdoor.R289258 (2019.08.29.00)

Behavior Detection
– Execution/MDP.Behavior.M10661

IOC
A portion of the file hash and a list of confirmed C&C addresses will be released on AhnLab TIP.

MD5
– 1ecd83ee7e4cfc8fed7ceb998e75b996: Volgmer Dropper initial version Type 1
– 35f9cfe5110471a82e330d904c97466a: Volgmer Backdoor initial version Type 1 (civolmgmt.dll)
– 5dd1ccc8fb2a5615bf5656721339efed: Volgmer Backdoor initial version Type 1 (divolenum.dll)
– 9a5fa5c5f3915b2297a1c379be9979f0: Volgmer Backdoor initial version Type 1 (fqrmsvc.dll)
– a545f548b09fdf61405f5cc07e4a7fa1: Volgmer Backdoor initial version Type 1
– eb9db98914207815d763e2e5cfbe96b9: Volgmer Backdoor initial version Type 1 (bgmsecenum.dll)
– fe32303e69b201f9934248cc06b32ef8: Volgmer Backdoor initial version Type 1 (xkupsvc.dll)
– 85b6e4ea8707149b48e41454cbd0d5ad: Volgmer Dropper initial version Type 2
– 64965a88e819fb93dbabafc4e3ad7b6c: Volgmer Backdoor initial version Type 2 (idefsrv.dll)
– 6da7d8aec65436e1350f1c0dfc4016b7: Volgmer Backdoor initial version Type 2
– e3d03829cbec1a8cca56c6ae730ba9a8: Volgmer Backdoor initial version Type 2 (hssvc.dll)
– 0171c4a0a53188fe6f9c3dfcc5722be6: Volgmer Backdoor later version (sbiimgr.dll)
– 17eacf4b4ae2ca4b07672dcc12e4d66d: Volgmer Backdoor later version (eqpkamgmt.dll)
– 1e2acecce7b5e9045b07d65e9e8afe1f: Volgmer Backdoor later version (lrmons.dll)
– 226cc1f17c4625837b37b5976acbd68e: Volgmer Backdoor later version (Exwtr.dll)
– 3e6119ebfacd1d88acbd2ca460c70b49: Volgmer Backdoor later version (helpsvcs.dll)
– 4753679cef5162000233d69330208420: Volgmer Backdoor later version (olesvc.bin)
– 5473fa2c5823fbab2b94e8d5c44bc7b4: Volgmer Backdoor later version (NWCWorkstations.dll)
– 570a4253ae80ee8c2b6b23386e273f3a: Volgmer Backdoor later version (Nlas.dll)
– 5c87373eef090bed525b80aef398ee8a: Volgmer Backdoor later version
– 693afaedf740492df2a09dfcc08a3dff: Volgmer Backdoor later version (ddmgr.dll)
– 6e21cc6669ada41e48b369b64ec5f37b: Volgmer Backdoor later version (ntmgr.dll)
– 72756e6ebb8274d9352d8d1e7e505906: Volgmer Backdoor later version (fhcmgr.dll)
– 8b3ec4b9c7ad20af418e89ca6066a3ad: Volgmer Backdoor later version (xbmgr.dll)
– 947124467bd04b7624d9b31e02b5ee7f: Volgmer Backdoor later version (hgiezmgmt.dll)
– 9a87f19609f28d7f7d76f9759864bd08: Volgmer Backdoor later version
– b1225fa644eebafba07f0f5e404bd4fd: Volgmer Backdoor later version (lrmons.dll)

– cf2ff5b59c638a06d8b81159b9a435ea: Volgmer Backdoor later version (tzmgr.dll)
– d52b5d8c20964333f79ff1bce3385d0b: Volgmer Backdoor later version (bqmgr.dll)
– e273803ae6724a714b970dd86ca1acd0: Volgmer Backdoor later version (fnsysN.dll)
– ea5d322648ff108b1c9cbdd1ef4a5959: Volgmer Backdoor later version (ntmgr.dll)
– 44fa8daa347ef5dd107bf123b4688797: Volgmer Dropper later version (ExwtrSvc.exe)
– 7f953c6988d829c9c4ac2002572c9055: Volgmer Dropper later version (ExwtrSvc.exe)
– c2ab2a8ffdc18c24080e889a634ef279: Volgmer Dropper later version (fmSysM.exe)
– 05bb1d8b7e62f4305d97042f07c64679: Scout Downloader v1 (Comms.db)
– 0b78347acf76d4bb66212bf9a41b9fb9: Scout Downloader v1 (gpklmgmt.dll)
– 0ed86587124f08325cd8f3d3d2556292: Scout Downloader v1 (bnsvc.dll)
– 35943aa640e122fcb127b2bfd6e29816: Scout Downloader v1 (helpsvcs.dll)
– 394b05394ebb9b239a063a6b5839edb9: Scout Downloader v1 (oxmgmt.dll)
– 5496adcd712d4378950ba62ad4c2423b: Scout Downloader v1 (gokimgmt.dll)
– 64cac69ab1e9108e0035f9ce38b47db7: Scout Downloader v1 (bnsvc.dll)
– 695e5b8dc9615ec603fe2cbb7326a50f: Scout Downloader v1 (helpsvcs.dll)
– c07e04d388fb394ac190aace51c03c33: Scout Downloader v1 (helpsvcs.dll)
– c41eb1ea59fab31147c5b107cc1c5a51: Scout Downloader v1 (tfbgmmgmt.dll)
– cc5a8a15d5808002e62d5daf2d4f31b3: Scout Downloader v1 (Comms.bin)
– 0b746394c9d23654577f4c0f2a39a543: Scout Downloader v2 (mib.cfg)
– 225cdc9b452b6d5a3f7616dcc9333d7d: Scout Downloader v2 (Keys.dat)
– 43f218d3a4b2199468b00a0b43f51c79: Scout Downloader v2 (wdsvc.dll)
– 4b1f1db4f169ca6b57015b313d665045: Scout Downloader v2 (olesvc.bin)
– 80d34f9ca10b0e8b49c02139e4615b7a: Scout Downloader v2 (NWCWorkstations.dll)
– 855e26d530e69ddc77bb19561fb19d90: Scout Downloader v2 (mib.bin)
– 9ec3a4257564658f651896abc608680e: Scout Downloader v2 (SRServices.dll)
– a76624578ed42cceba81c76660977562: Scout Downloader v2 (eppagent.bin)
– b517e7ad07d1182feb4b8f61549ff233: Scout Downloader v2 (usoshared.bin)
– fa868a38ceeb46ee9cf8bd441a67ae27: Scout Downloader v2 (ose.bin)
– 1f1a3fe0a31bd0b17bc63967de0ccc29: Scout Downloader v2 – Encoded (configmanager.tlb)
– fa3e49c877a95f37fd25dbd62f9e274c: Scout Downloader v2 – Encoded (event.dat)
– 202a7eec39951e1c0b1c9d0a2e24a4c4: Loader – Scout Downloader v2 (helpsvcs.dll)
– b457e8e9d92a1b31a4e2197037711783: Loader – Scout Downloader v2 (wpnsvc.dll)
– 8543667917a318001d0e331aeae3fb9b: Config – Scout Downloader v2 (C_68656c.NLS)
– c16a6178a4910c6f3263a01929f306b9: Scout Downloader v2 (C_77706e.NLS)
– 1c89fb4aee20020bfd75713264df97cd: Dropper – Scout Downloader
– 76f02ab112b8e077544d0c0a6e0c428a: Dropper – Scout Downloader (wAgent.dat)
– 7ba37d662f19bef27c3da2fd2cee0e3a: Dropper – Scout Downloader (wAgent.dat)
– 7f0e773397808b4328ad11d6948a683f: Dropper – Scout Downloader (Comms.bin)
– bf5d815597018fe7f3dfb52d4f7e1f65: Dropper – Scout Downloader

Subscribe to AhnLab's next-generation threat intelligence platform 'AhnLab TIP' to check related IOC and detailed analysis information.

Categories:Malware Information

Tagged as:Lazarus,Scout,Volgmer