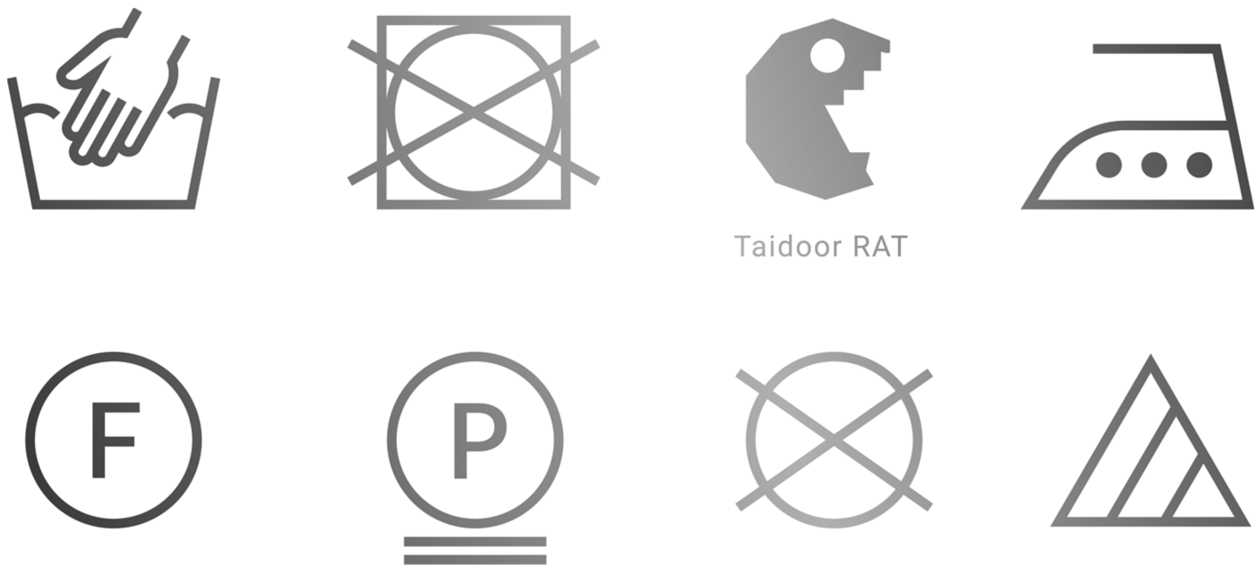
Taidoor - a truly persistent threat

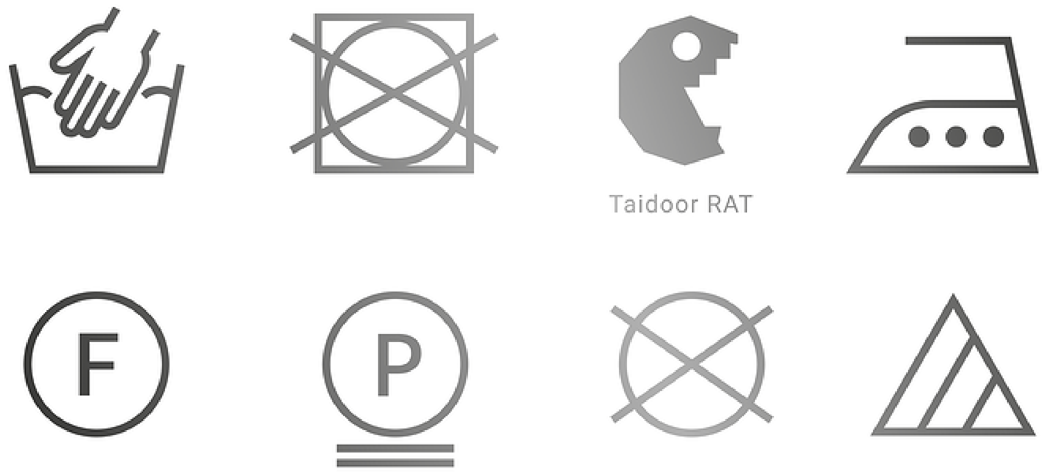
blog.reversinglabs.com/blog/taidoor-a-truly-persistent-threat



Threat Research | September 22, 2020

Blog Author
Karlo Zanki, Reverse Engineer at ReversingLabs. Read More...

When malware lasts longer than your washing machine



Introduction

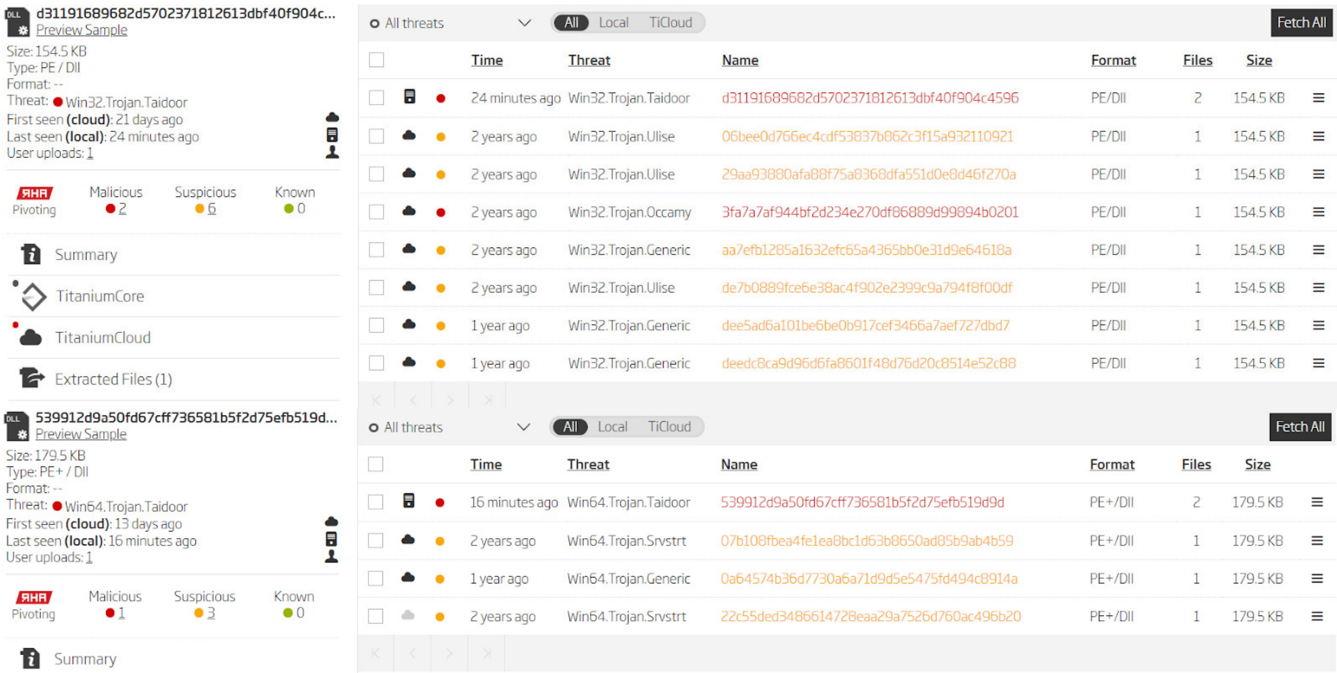
Government-supported actors usually conduct long-lasting activities in cyberspace, and to simplify such continuous processes, they often develop malicious tools with the intention of using them for a long period of time. Like any other malicious tool, they need to be stealthy, and when they get detected, some modifications are necessary to become undetectable again. Sometimes it can be as simple as changing the compromised infrastructure, while at other times creating a new version of the tool is required. When organizations put a lot of effort into creating a tool like that, they probably don't plan to use it in massive campaigns. It is expected to be used in smaller, targeted attacks; therefore, researchers won’t have too many samples for analysis at their disposal.

An example of such a tool is Taidoor RAT (remote access trojan), dating all the way back to 2008, whose new version was recently discovered and presented in a technical report released by the US government institutions. Taidoor is described as a Remote Access Trojan developed and used by cyber actors supported by the Chinese government. The new version of the RAT consists of two parts - a loader in a DLL form, and a main RAT module that comes as RC4-encrypted binary data. The loader first decrypts the encrypted main RAT module, and then executes its exported Start function. The report provides two samples for both the loader and the main encrypted RAT module. These samples come with only two C2 domains and one C2 IP.
In this blog, you will learn how our Titanium Platform can help you find more malware samples to extend the IOC lists and power up your defenses.

Taidoor

As already mentioned, the Taidoor RAT consists of two parts, a loader and the main RAT module. A few pivoting attempts on the loader samples didn’t produce any results, so we focused on the samples of the main RAT module. This makes sense, as they contain the malware configuration including C2 domains and IPs.

Two samples of the main RAT available from the threat report are encrypted with the RC4 algorithm, and aren’t suitable for pivoting in that form. The first step is to decrypt them and get them to their normal DLL format suitable for metadata extraction. Publicly available tools like CyberChef can be used to decrypt various encryption algorithms, including RC4. Once decrypted, the DLL is processed with Titanium Platform, and its metadata is extracted. The first thing to do is look at the files grouped into the same buckets based on the RHA1, our functional file similarity algorithm, for each of the samples.

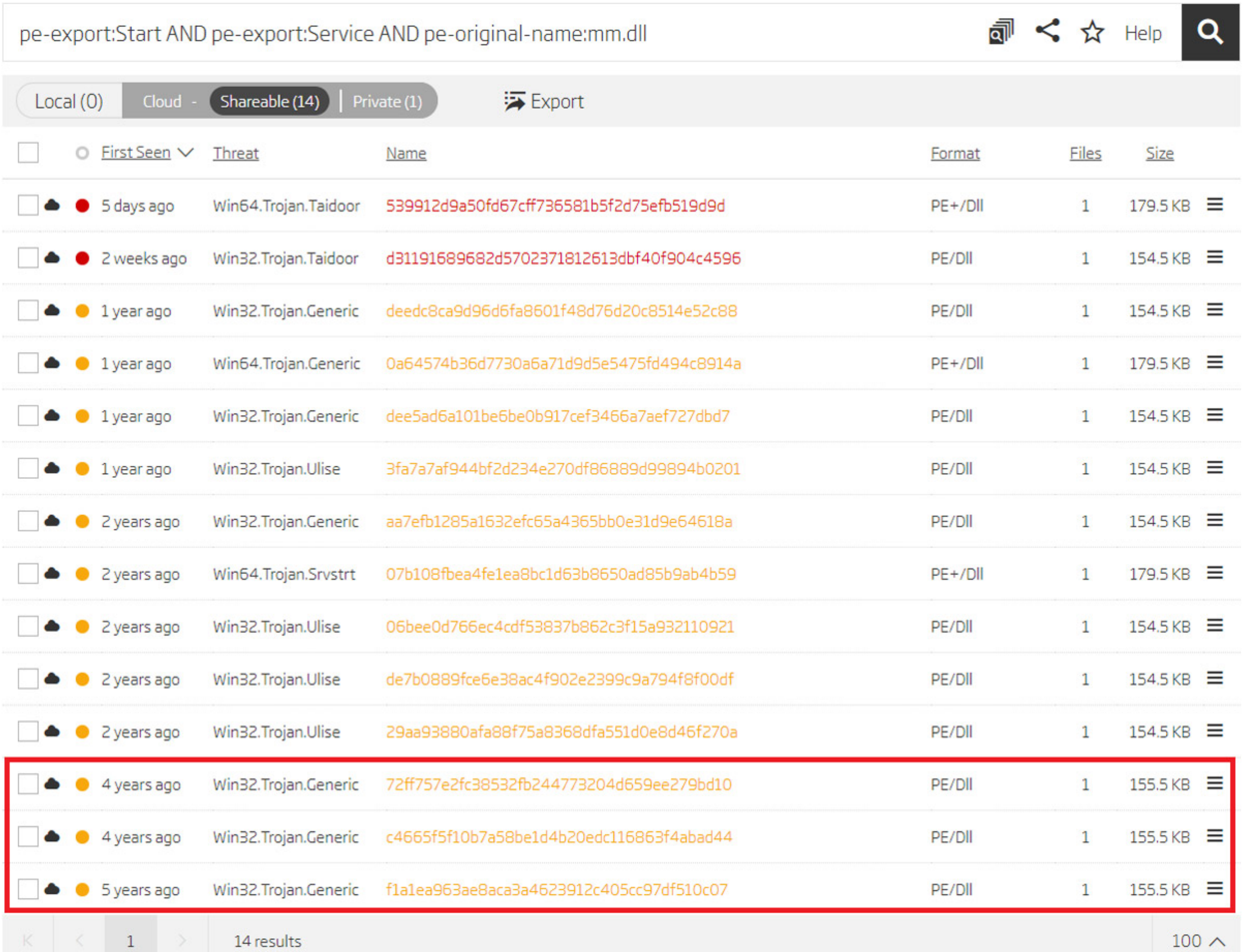


Similar files grouped by RHA1 algorithm

The RHA1 algorithm reveals ten more samples dating back to 2018 and 2019. However, there are additional options for pivoting. Looking at the samples' exports shows a specific combination of functions and original file name.

Samples exports

Running a Titanium Platform cloud search query with this specific combination finds the samples already discovered using the RHA1 file similarity algorithm, and also 3 additional samples.



pe-export:Start AND pe-export:Service AND pe-original-name:mm.dll

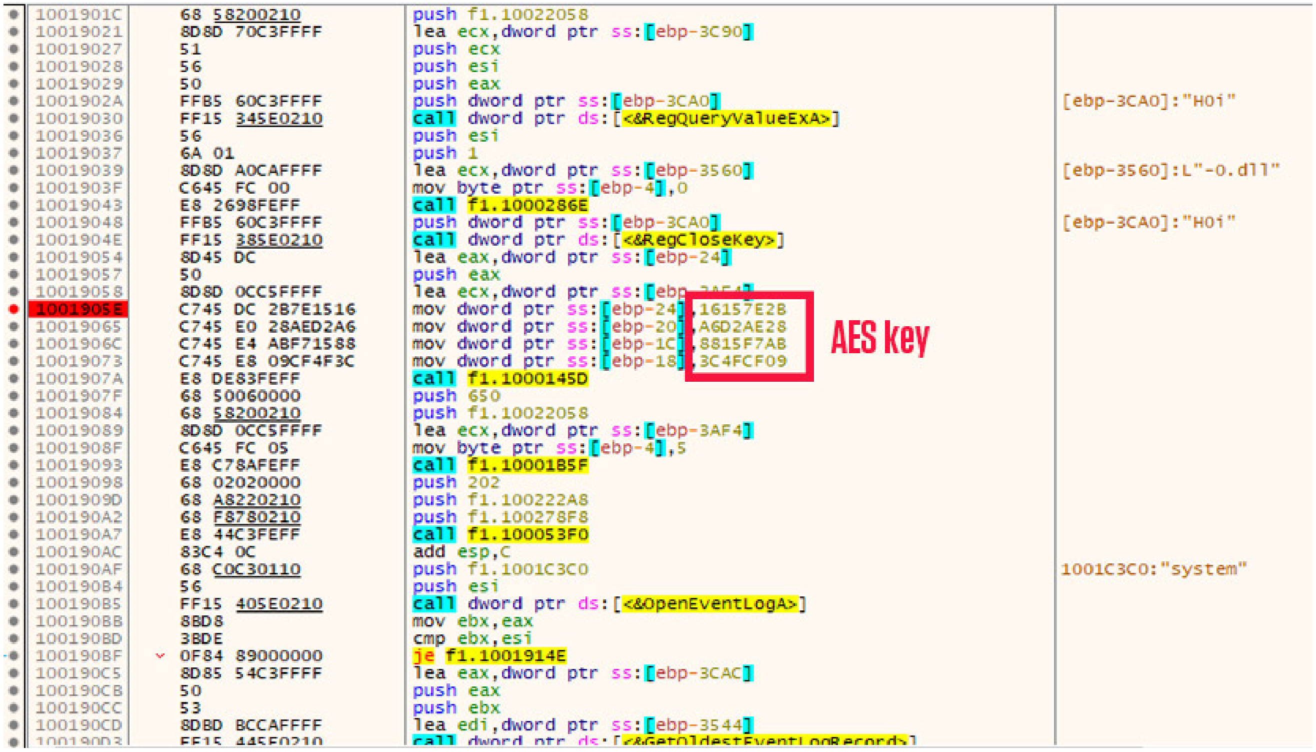
Local (0) | Cloud - Shareable (14) | Private (1) | Export

First Seen	Threat	Name	Format	Files	Size
5 days ago	Win64.Trojan.Taidoor	539912d9a50fd67cff736581b5f2d75efb519d9d	PE+/Dll	1	179.5 KB
2 weeks ago	Win32.Trojan.Taidoor	d31191689682d5702371812613dbf40f904c4596	PE/Dll	1	154.5 KB
1 year ago	Win32.Trojan.Generic	deedc8ca9d96d6fa8601f48d76d20c8514e52c88	PE/Dll	1	154.5 KB
1 year ago	Win64.Trojan.Generic	0a64574b36d7730a6a71d9d5e5475fd494c8914a	PE+/Dll	1	179.5 KB
1 year ago	Win32.Trojan.Generic	dee5ad6a101be6be0b917cef3466a7aef727dbd7	PE/Dll	1	154.5 KB
1 year ago	Win32.Trojan.Ulise	3fa7a7af944bf2d234e270df86889d99894b0201	PE/Dll	1	154.5 KB
2 years ago	Win32.Trojan.Generic	aa7efb1285a1632efc65a4365bb0e31d9e64618a	PE/Dll	1	154.5 KB
2 years ago	Win64.Trojan.Srvstrt	07b108fbea4fe1ea8bc1d63b8650ad85b9ab4b59	PE+/Dll	1	179.5 KB
2 years ago	Win32.Trojan.Ulise	06bee0d766ec4cdf53837b862c3f15a932110921	PE/Dll	1	154.5 KB
2 years ago	Win32.Trojan.Ulise	de7b0889fce6e38ac4f902e2399c9a794f8f00df	PE/Dll	1	154.5 KB
2 years ago	Win32.Trojan.Ulise	29aa93880afa88f75a8368dfa551d0e8d46f270a	PE/Dll	1	154.5 KB
4 years ago	Win32.Trojan.Generic	72ff757e2fc38532fb244773204d659ee279bd10	PE/Dll	1	155.5 KB
4 years ago	Win32.Trojan.Generic	c4665f5f10b7a58be1d4b20edc116863f4abad44	PE/Dll	1	155.5 KB
5 years ago	Win32.Trojan.Generic	f1a1ea963ae8aca3a4623912c405cc97df510c07	PE/Dll	1	155.5 KB

1 | 14 results | 100

Pivoting on exports and original name

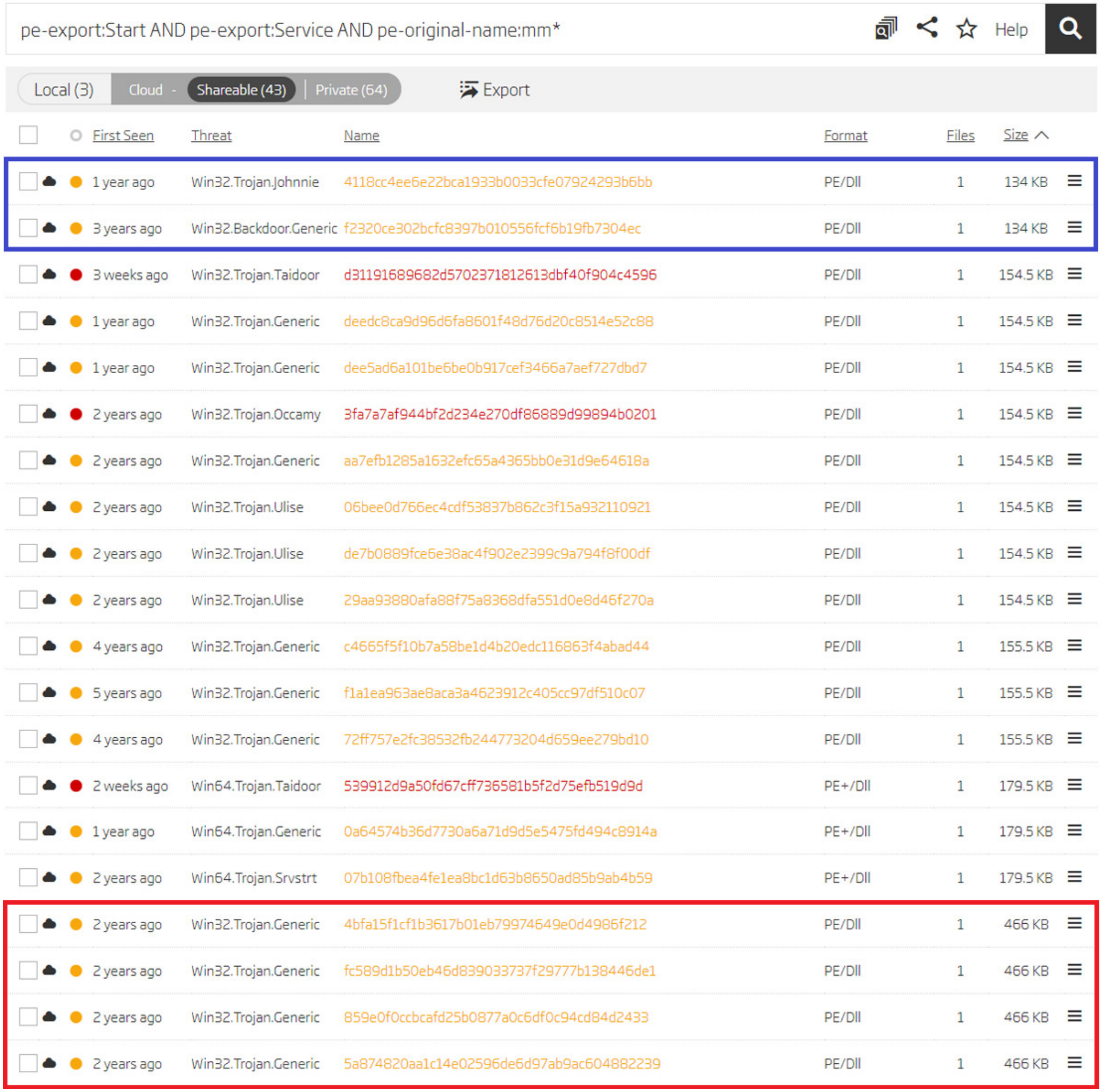
Analyzing these three samples shows that they are using the same AES key as described in the threat report. The significant difference is that they have another layer of encryption used to hide the part of the code responsible for configuration decryption, including the AES key and S-Box initialization.



Loading of AES decryption key in previously decrypted layer of code

Pivoting on these three samples didn't return any new results. However, there is something uncommon in this exported file name - "mm.dll". It is probably a shorthand for something.

What if there were some variations in the functionality of these samples? There is a chance that this *mm* prefix could have something appended to it. To check this assumption, a search query is constructed using the wildcard character to match samples that start with the *mm* prefix and at the same time contain the aforementioned exported functions *Start* and *Service*.

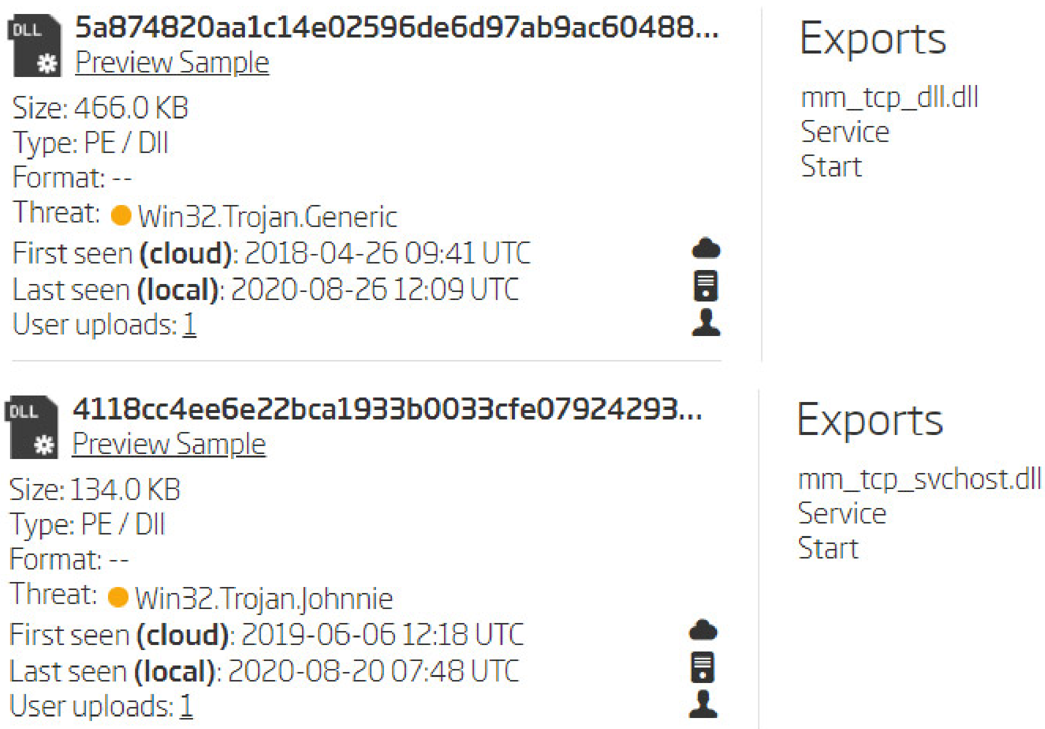

pe-export:Start AND pe-export:Service AND pe-original-name:mm*

Local (3) | Cloud - Shareable (43) | Private (64) | Export

First Seen	Threat	Name	Format	Files	Size
1 year ago	Win32.Trojan.Johnnie	4118cc4ee6e22bca1933b0033cfe07924293b6bb	PE/Dll	1	134 KB
3 years ago	Win32.Backdoor.Generic	f2320ce302bcfc8397b010556fcf6b19fb7304ec	PE/Dll	1	134 KB
3 weeks ago	Win32.Trojan.Taidoor	d31191689682d5702371812613dbf40f904c4596	PE/Dll	1	154.5 KB
1 year ago	Win32.Trojan.Generic	deedc8ca9d96d6fa8601f48d76d20c8514e52c88	PE/Dll	1	154.5 KB
1 year ago	Win32.Trojan.Generic	dee5ad6a101be6be0b917cef3466a7aef727dbd7	PE/Dll	1	154.5 KB
2 years ago	Win32.Trojan.Occamy	3fa7a7af944bf2d234e270df86889d99894b0201	PE/Dll	1	154.5 KB
2 years ago	Win32.Trojan.Generic	aa7efb1285a1632efc65a4365bb0e31d9e64618a	PE/Dll	1	154.5 KB
2 years ago	Win32.Trojan.Ulise	06bee0d766ec4cdf53837b862c3f15a932110921	PE/Dll	1	154.5 KB
2 years ago	Win32.Trojan.Ulise	de7b0889fce6e38ac4f902e2399c9a794f8f00df	PE/Dll	1	154.5 KB
2 years ago	Win32.Trojan.Ulise	29aa93880afa88f75a8368dfa551d0e8d46f270a	PE/Dll	1	154.5 KB
4 years ago	Win32.Trojan.Generic	c4665f5f10b7a58be1d4b20edc116863f4abad44	PE/Dll	1	155.5 KB
5 years ago	Win32.Trojan.Generic	f1a1ea963ae8aca3a4623912c405cc97df510c07	PE/Dll	1	155.5 KB
4 years ago	Win32.Trojan.Generic	72ff757e2fc38532fb244773204d659ee279bd10	PE/Dll	1	155.5 KB
2 weeks ago	Win64.Trojan.Taidoor	539912d9a50fd67cff736581b5f2d75efb519d9d	PE+/Dll	1	179.5 KB
1 year ago	Win64.Trojan.Generic	0a64574b36d7730a6a71d9d5e5475fd494c8914a	PE+/Dll	1	179.5 KB
2 years ago	Win64.Trojan.Srvstrt	07b108fbea4fe1ea8bc1d63b8650ad85b9ab4b59	PE+/Dll	1	179.5 KB
2 years ago	Win32.Trojan.Generic	4bfa15f1cf1b3617b01eb79974649e0d4986f212	PE/Dll	1	466 KB
2 years ago	Win32.Trojan.Generic	fc589d1b50eb46d839033737f29777b138446de1	PE/Dll	1	466 KB
2 years ago	Win32.Trojan.Generic	859e0f0ccbcafd25b0877a0c6df0c94cd84d2433	PE/Dll	1	466 KB
2 years ago	Win32.Trojan.Generic	5a874820aa1c14e02596de6d97ab9ac604882239	PE/Dll	1	466 KB

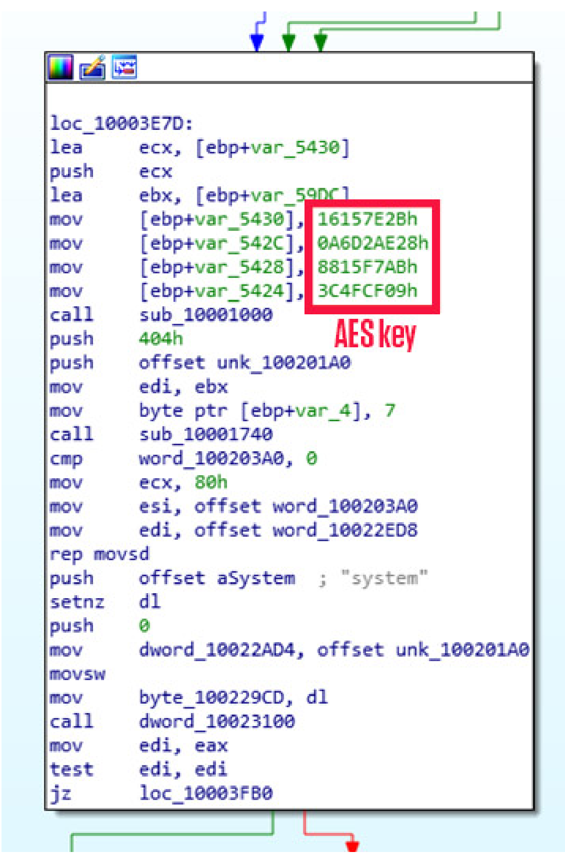
Pivoting on exports and original name using wildcard character

Ordering these search results by their size - and ignoring clean software and previously discovered samples - leaves just a few hashes we need to look at. These hashes are split in two groups based on their size. Looking at their exports reveals two new original file names - **mm_tcp_dll.dll** and **mm_tcp_svchost.dll**.

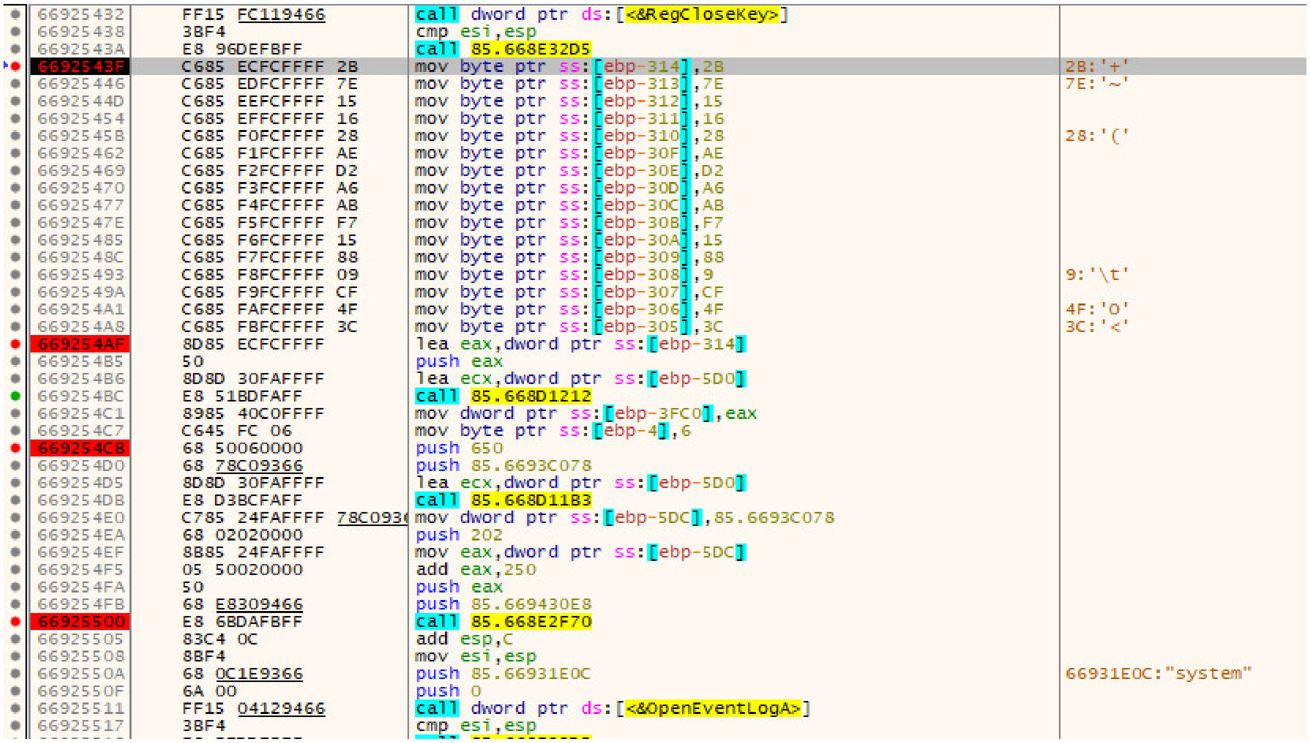


Samples exports

Analyzing these samples in disassembler confirms these are indeed Taidoor samples, since they perform the same AES decryption and use the same AES key. The 466 KB samples also have a layer of encryption used to hide the part of the code responsible for configuration decryption, as already described in the previous case.



Loading of AES decryption key in 134KB samples



Loading of AES decryption key in previously decrypted layer of code in 466KB samples

They also include the same strings present in the Taidoor samples mentioned in the referenced threat report.

0x040x230x190x340xfe0xc1

%%temp%%\%u

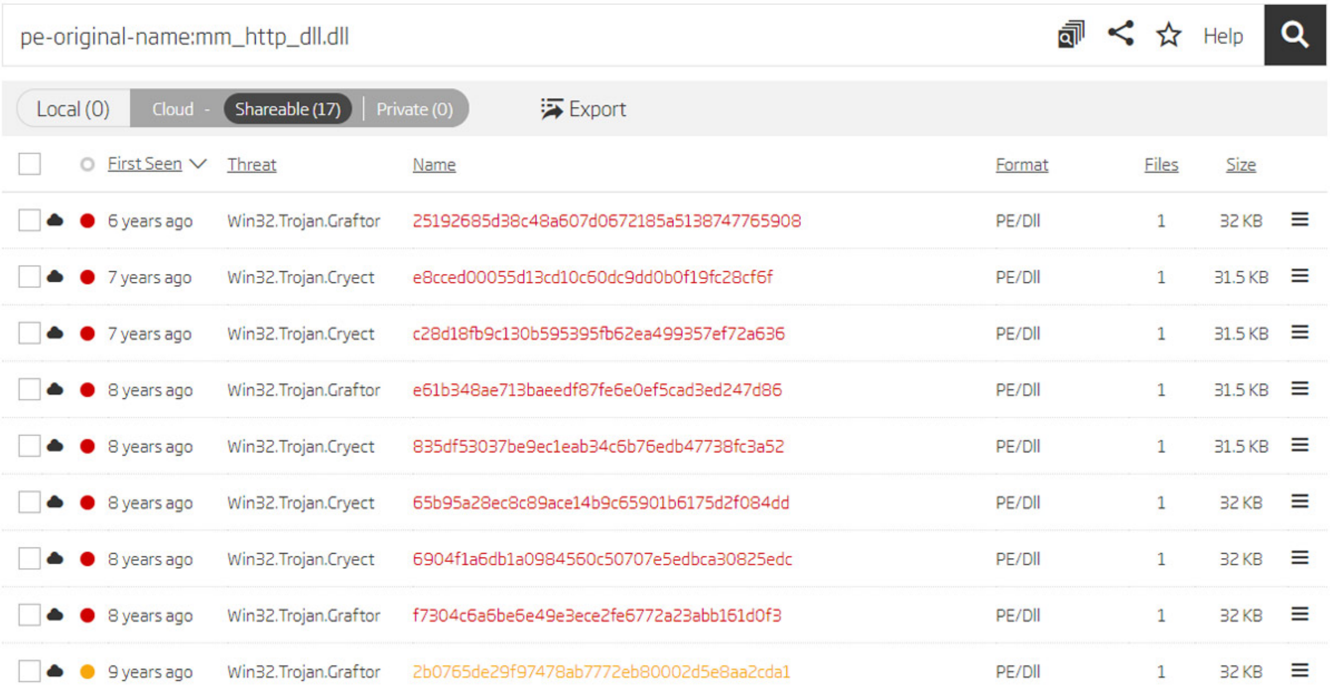
CONNECT %s:%d HTTP/1.0
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)
Host: %s
Accept: */*
Pragma: no-cache
Proxy-Connection: Keep-Alive
Content-Length: 0

Some of the strings found in Taidoor samples

One interesting thing that catches the eye are the pdb paths. Samples of the 134 KB size contain the pdb path **"C:\Users\john\Desktop\KD17.6_20170628\Release\mm_tcp_svchost.pdb"** and have the compilation timestamp set to 2017-07-04T09:20:21. Samples of the 466 KB size have their compilation timestamp set to 2016-04-06T08:44:22 and don't include any pdb paths, but looking at the contained strings reveals one very similar to the mentioned pdb path - **"c:\users\develop\desktop\kd15.1_aes_20160321\mm_tcp_dll\svchost.cpp"**. We can assume that in the "KD**VV.v_YYYYMMDD**" part of the string, the **VV.v** represents the malware version, and that **YYYYMMDD** represents the creation date. Unfortunately, pivoting variations on these pdb paths didn't result in any other samples besides the ones previously found.

There are still two discovered original file names left to pivot on. Pivoting on *mm_tcp_svchost.dll* doesn't result in anything new. On the other hand, pivoting on *mm_tcp_dll.dll* reveals around 70 more samples. Looking at them in detail indicates they are older versions of Taidoor dating all the way back to 2011. These samples have configuration encrypted with the RC4 algorithm, and not with AES like the version from the referenced threat report. We won't describe these samples here because they are a bit outdated and are more similar to the older versions of Taidoor already covered with technical reports from FireEye and TrendMicro.

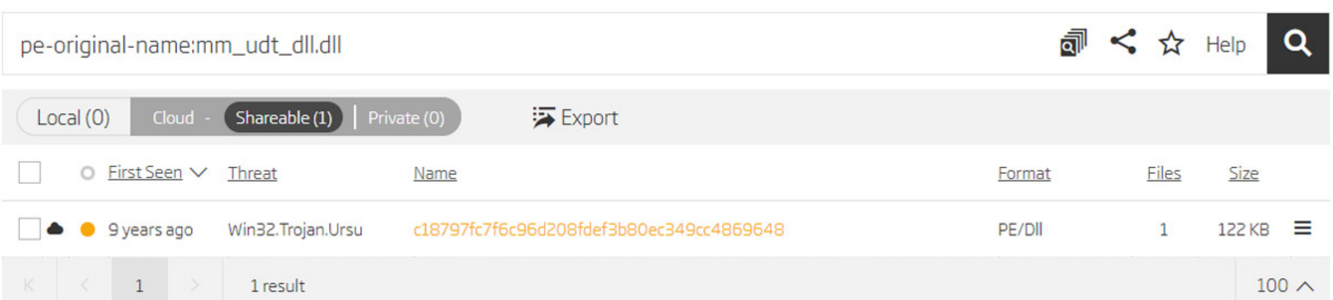
But if there is a sample named *mm_tcp_dll.dll*, is it possible that there is a sample using the http protocol for communication? We can guess that it would be named *mm_http_dll.dll* if it follows the naming convention. A search query indeed finds 17 samples with this name, and analyzing them in disassembler shows they also have the configuration encrypted with the same "**0xA1 0xA2**" RC4 key. But as illustrated in the following image, these samples are also 8 years old, and are not in the focus of interest of this blog post.



First Seen	Threat	Name	Format	Files	Size
6 years ago	Win32.Trojan.Graftor	25192685d38c48a607d0672185a5138747765908	PE/Dll	1	32 KB
7 years ago	Win32.Trojan.Cryect	e8cced00055d13cd10c60dc9dd0b0f19fc28cf6f	PE/Dll	1	31.5 KB
7 years ago	Win32.Trojan.Cryect	c28d18fb9c130b595395fb62ea499357ef72a636	PE/Dll	1	31.5 KB
8 years ago	Win32.Trojan.Graftor	e61b348ae713baeedf87fe6e0ef5cad3ed247d86	PE/Dll	1	31.5 KB
8 years ago	Win32.Trojan.Cryect	835df53037be9ec1eab34c6b76edb47738fc3a52	PE/Dll	1	31.5 KB
8 years ago	Win32.Trojan.Cryect	65b95a28ec8c89ace14b9c65901b6175d2f084dd	PE/Dll	1	32 KB
8 years ago	Win32.Trojan.Cryect	6904f1a6db1a0984560c50707e5edbca30825edc	PE/Dll	1	32 KB
8 years ago	Win32.Trojan.Graftor	f7304c6a6be6e49e3ece2fe6772a23abb161d0f3	PE/Dll	1	32 KB
9 years ago	Win32.Trojan.Graftor	2b0765de29f97478ab7772eb80002d5e8aa2cda1	PE/Dll	1	32 KB

Pivoting on original name

Another very interesting sample exists, originally named *mm_udt_dll.dll*, suggesting usage of the UDT protocol for the C2 communication. Since it is also quite old, it won't be examined in detail, but it uses the same RC4 key for configuration encryption, and it is likely that the only difference is the network protocol used for the communication.



First Seen	Threat	Name	Format	Files	Size
9 years ago	Win32.Trojan.Ursu	c18797fc7f6c96d208fdef3b80ec349cc4869648	PE/Dll	1	122 KB

1 1 result 100

Pivoting on original name

All collected samples have their configuration placed in the *.data* section, and use the same AES key "**2B 7E 15 16 28 AE D2 A6 AB F7 15 88 09 CF 4F 3C**" IV: "**00**". Configurations were extracted from all the samples and subsequently decrypted with the provided key. The last thing to do was to collect decrypted C2 domains and IPs, and create an IOC list provided at the end of this blog.

Development history

The samples were grouped based on the described findings, and one representative sample from each group was put into a table. The table was then sorted based on the compilation timestamp.

SHA1	Compile timestamp	First seen timestamp	Size	File Type	Original name
f1a1ea963ae8aca3a4623912c405cc97df510c07	2015-12-22 T07:15:44	12/23/2015 4:40	155.5KB	PE/Dll	mm.dll
859e0f0ccbcafd25b0877a0c6df0c94cd84d2433	2016-04-06 T08:44:22	7/18/2018 1:03	466KB	PE/Dll	mm_tcp_dll.dll
4118cc4ee6e22bca1933b0033cfe07924293b6bb	2017-07-04 T09:20:21	6/6/2019 12:18	134KB	PE/Dll	mm_tcp_svchost.dll
de7b0889fce6e38ac4f902e2399c9a794f8f00df	2018-04-25 T01:36:11	8/3/2018 13:31	154.5KB	PE/Dll	mm.dll
22c55ded3486614728eaa29a7526d760ac496b20	2018-09-21 T01:51:04	11/20/2018 13:19	179.5KB	PE+/Dll	mm.dll

Two oldest groups of samples have an extra layer of encryption used for hiding the part of the code responsible for AES decryption. At some point in 2017, the malware developers decided to simplify the code and move that extra protection layer, probably to another PE artifact (the loader component). The fact that the same AES key has been used for more than 4 years is a lucky coincidence which simplifies the IOC extraction for researchers and makes correlating samples easier. Older versions of Taidoor mentioned in this blog use a different encryption algorithm, but have a highly similar set of exported functions, and also very similar high-level program logic.

Conclusion

Like any other software product, malware families require a lot of maintenance and improvement to achieve long-term operability. Even though such continuous upgrading helps malware avoid detection mechanisms, it also results in related malware versions. These related versions must have some similarities, and this is an opportunity for security researchers to establish correlations between various malware campaigns and quickly find more threat samples.

As part of this research, and based on the analysis provided in the referenced threat report, 23 related samples were found and 40 new C2 IPs and domains extracted from their configurations.

Titanium Platform helps researchers get a more detailed insight into malware samples. Various search queries can be constructed from the extracted metadata, and can be used to find related threats. With Titanium Platform you can also establish time relationships between the samples, and together with the differences in malware functionality, this can enable you a better understanding of malware development history.

IOC list

The following link contains the data extracted from the newly discovered samples related to the Taidoor malware.

https://blog.reversinglabs.com/hubfs/Blog/Taidoor_SHA1_list.txt

https://blog.reversinglabs.com/hubfs/Blog/Taidoor_C2_list.txt

MORE BLOG ARTICLES