DirtyMoe: Code Signing Certificate

decoded.avast.io/martinchlumecky/dirtymoe-3/

September 17, 2021

by Martin ChlumeckýSeptember 17, 202126 min read

Abstract

The DirtyMoe malware uses a driver signed with a revoked certificate that can be seamlessly loaded into the Windows kernel. Therefore, one of the goals is to analyze how Windows works with a code signature of Windows drivers. Similarly, we will be also interested in the code signature verification of user-mode applications since the user account control (UAC) does not block the application execution even if this one is also signed with a revoked certificate. The second goal is a statistical analysis of certificates that sign the DirtyMoe driver because the certificates are also used to sign other malicious software. We focus on the determination of the certificate's origin, prevalence, and a quick analysis of the top 10 software signed with these certificates.

Contrary to what often has been assumed, Windows loads a driver signed with a revoked certificate. Moreover, the results indicate that the UAC does not verify revocation online but only via the system local storage which is updated by a user manually. DirtyMoe certificates are used to sign many other software, and the number of incidents is still growing. Furthermore, Taiwan and Russia seem to be the most affected by these faux signatures.

Overall, the analyzed certificates can be declared as malicious, and they should be monitored. The UAC contains a significant vulnerability in its code signature verification routine. Therefore, in addition to the usual avoidance of downloading from usual sources, Windows users should also not fully rely on inbuilt protections only. Due to the flawed certificate verification, manual verification of the certificate is recommended for executables requiring elevated privileges.

1. Introduction

The DirtyMoe malware is a complex malicious backdoor designed to be modularized, undetectable, and untrackable. The main aim of DirtyMoe is cryptojacking and DDoS attacks. Basically, it can do anything that attackers want. The previous study, published at DirtyMoe: Introduction and General Overview, has shown that DirtyMoe also employs various self-protection and anti-forensics mechanisms. One of the more significant safeguards, defending DirtyMoe, is a rootkit. What is significant about the rootkit using is that it offers advanced techniques to hide malicious activity on the kernel layer. Since DirtyMoe is complex malware and has undergone long-term development, therefore, the malware authors have also implemented various rootkit mechanisms. The DirtyMoe: Rootkit Driver post examines the DirtyMoe driver functionality in detail.

However, another important aspect of the rootkit driver is a digital signature. The rootkits generally exploit the system using a Windows driver. Microsoft has begun requiring the code signing with certificates that a driver vendor is trusted and verified by a certification authority (CA) that Microsoft trusts. Nonetheless, Windows does not allow the loading of unsigned drivers since 64-bit Windows Vista and later versions. Although the principle of the code signing is correct and safe, Windows allows loading a driver that a certificate issuer has explicitly revoked the used certificate. The reasons for this behavior are various, whether in terms of performance or backward compatibility. The weak point of this certificate management is if malware authors steal any certificate that has been verified by Microsoft and revoked by CA in the past. Then the malware authors can use this certificate for malicious purposes. It is precisely the case of DirtyMoe, which still signs its driver with stolen certificates. Moreover, users cannot affect the codesign verification via the user account control (UAC) since drivers are loaded in the kernel mode.

Motivated by the loading of drivers with revoked certificates, the main issues addressed in this paper are analysis of mechanism how Windows operates with code signature in the kernel and user mode; and detailed analysis of certificates used for code signing of the DirtyMoe driver.

This paper first gives a brief overview of UAC and code signing of Windows drivers. There are also several observations about the principle of UAC, and how Windows manages the certificate revocation list (CRL). The remaining part of the paper aims to review in detail the available information about certificates that have been used to code signatures of the DirtyMoe driver.

Thus far, three different certificates have been discovered and confirmed by our research group as malicious. Drivers signed with these certificates can be loaded notwithstanding their revocation into the 64-bit Windows kernel. An initial objective of the last part is an analysis of the suspicious certificates from a statistical point of view. There are three viewpoints that we studied. The first one is the geological origin of captured samples signed with the malicious certificate. The second aspect is the number of captured samples, including a prediction factor for each certificate. The last selected frame of reference is a statistical overview about a type of the signed software because the malware authors use

the certificates to sign various software, e.g., cracked software or other popular user applications that have been patched with a malicious payload. We have selected the top 10 software signed with revoked certificates and performed a quick analysis. Categories of this software help us to ascertain the primarily targeted type of software that malware authors signed. Another significant scope of this research is looking for information about companies that certificates have probably been leaked or stolen.

2. Background

Digital signatures are able to provide evidence of the identity, origin, and status of electronic documents, including software. The user can verify the validity of signatures with the certification authority. Software developers use digital signatures to a code signing of their products. The software authors guarantee via the signature that executables or scripts have not been corrupted or altered since the products have been digitally signed. Software users can verify the credibility of software developers before run or installation.

Each code-signed software provides information about its issuer, and an URL points to the current certificate revocation list (CRL) known as the CRL Distribution Point. Windows can follow the URL and check the validity of the used certificate. If the verification fails, the User Account Control (UAC) blocks software execution that code-signature is not valid. Equally important is the code-signature of Windows drivers that use a different approach to digital signature verification. Whereas user-mode software signatures are optional, the Windows driver must be signed by a CA that Windows trusts. Driver signing is mandatory since 64-bit Windows Vista.

2.1 User Account Control and Digital Signature

Windows 10 prompts the user for confirmation if the user wants to run software with high permission. User Account Control (UAC) should help to prevent users from inadvertently running malicious software or other types of changes that might destabilize the Windows system. Additionally, UAC uses color coding for different scenarios – blue for a verified signature, yellow for an unverified, and red for a rejected certificate; see **Figure 1**.

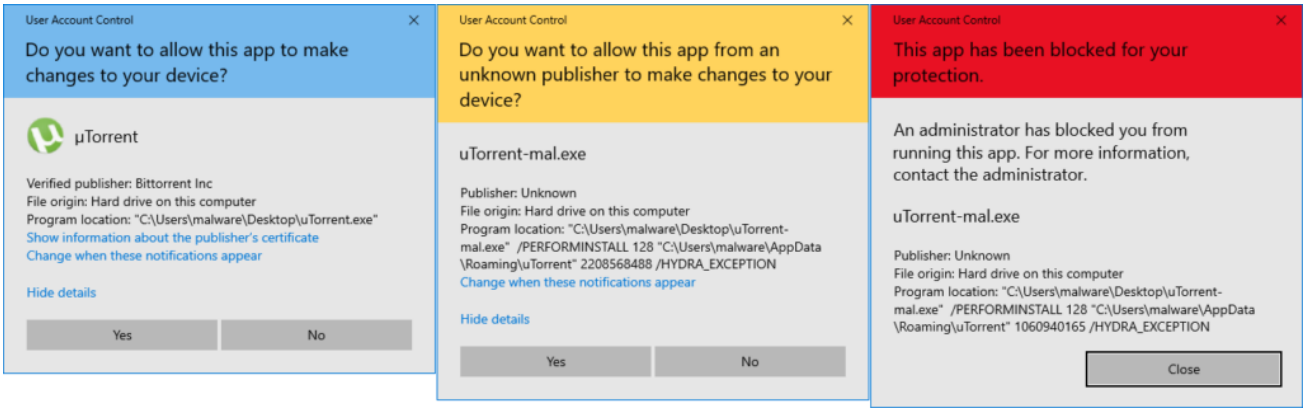

Figure 1: UAC of verified, unverified, and rejected software requiring the highest privileges

Unfortunately, users are the weakest link in the safety chain and often willingly choose to ignore such warnings. Windows itself does not provide 100% protection against unknown publishers and even revoked certificates. So, user's ignorance and inattention assist malware authors.

2.2 Windows Drivers and Digital Signature

Certificate policy is diametrically different for Windows drivers in comparison to other user-mode software. While user-mode software does not require code signing, code signing is mandatory for a Windows driver; moreover, a certificate authority trusted by Microsoft must be used. Nonetheless, the certificate policy for the windows drivers is quite complicated as MSDN demonstrates [1].

2.2.1 Driver Signing

Historically, 32-bit drivers of all types were unsigned, e.g., drivers for printers, scanners, and other specific hardware. Windows Vista has brought a new signature policy, but Microsoft could not stop supporting the legacy drivers due to backward compatibility. However, 64-bit Windows Vista and later has to require digitally signed drivers according to the new policy. Hence, 32-bit drivers do not have to be signed since these drivers cannot be loaded into the 64-bit kernels. The code signing of 32-bit drivers is a good practice although the certificate policy does not require it.

Microsoft has no capacity to sign each driver, more precisely file, provided by third-party vendors. Therefore, a cross-certificate policy is used to provide an instrument to create a chain of trust. This policy allows the release of a subordinate certificate that builds the chain of trust from the Microsoft root certification authority to multiple other CAs that Microsoft accredited. The current cross-certificate list is documented in [2]. In fact, the final driver signature is done with a certificate that has been issued by the subordinate CA, which Microsoft authorized.

The examples of the certificate chain illustrates **Figure 2** and the following output of `signtool`;
see **Figure 3**.

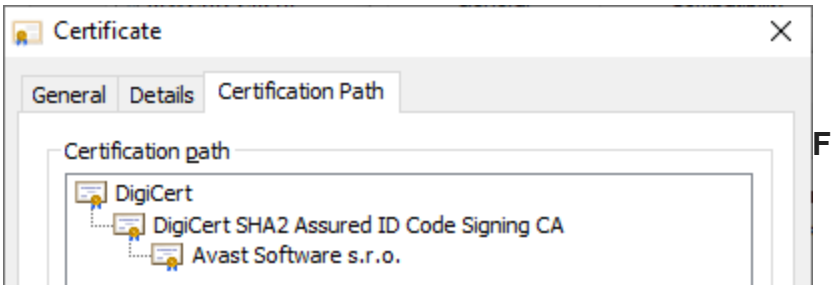

Figure 2: Certification chain of one Avast driver

```
signtool.exe verify /v /kp c:\Windows\System32\drivers\aswbidsdriver.sys
Verifying: c:\Windows\System32\drivers\aswbidsdriver.sys
Signing Certificate Chain:
    Issued to: DigiCert High Assurance EV Root CA
    Issued by: DigiCert High Assurance EV Root CA
    Expires:   10. 11. 2031 2:00:00
        Issued to: DigiCert High Assurance Code Signing CA-1
        Issued by: DigiCert High Assurance EV Root CA
        Expires:   10. 2. 2026 14:00:00
            Issued to: Avast Software s.r.o.
            Issued by: DigiCert High Assurance Code Signing CA-1
            Expires:   19. 10. 2022 14:00:00
The signature is timestamped: 8. 6. 2021 11:34:47
Successfully verified: c:\Windows\System32\drivers\aswbidsdriver.sys
Number of files successfully Verified: 1
Number of warnings: 0
Number of errors: 0
```

Figure 3: Output of signtool.exe for one Avast driver

2.2.2 **Driver Signature Verification**

For user-mode software, digital signatures are verified remotely via the CRL list obtained from certificate issuers. The signature verification of Windows drivers cannot be performed online compared with user-mode software because of the absence of network connection during the kernel bootup. Moreover, the kernel boot must be fast and a reasonable approach to tackle the signature verification is to implement a light version of the verification algorithm.

The Windows system has hardcoded root certificates of several CAs and therefore can verify the signing certificate chain offline. However, the kernel has no chance to authenticate signatures against CRLs. The same is applied to expired certificates. Taken together, this approach is implemented due to kernel performance and backward driver compatibility. Thus, leaked and compromised certificates of a trusted driver vendor causes a severe problem for Windows security.

3. CRL, UAC, and Drivers

As was pointed out in the previous section, CRL contains a list of revoked certificates. UAC should verify the chain of trust of run software that requires higher permission. The UAC dialog then shows the status of the certificate verification. It can end with two statuses, e.g., verified publisher or unknown publisher [3], see **Figure 1**.

However, we have a scenario where software is signed with a revoked certificate and run is not blocked by UAC. Moreover, the code signature is marked as the unknown publisher (yellow titled UAC), although the certificate is in CRL.

3.1 Cryptographic Services and CRL Storage

Cryptographic Services confirms the signatures of Windows files and stores file information into the `C:\Users\<user>\AppData\LocalLow\Microsoft\CryptnetUrlCache` folder, including the CRL of the verified file. The `CryptnetUrlCache` folder (cache) is updated, for instance, if users manually check digital signatures via "*Properties -> Digital Signature*" or via the `signtool` tools, as **Figure 3** and **Figure 4** illustrate. The certificate verification processes the whole chain of trust, including CRL and possible revocations deposits in the cache folder.

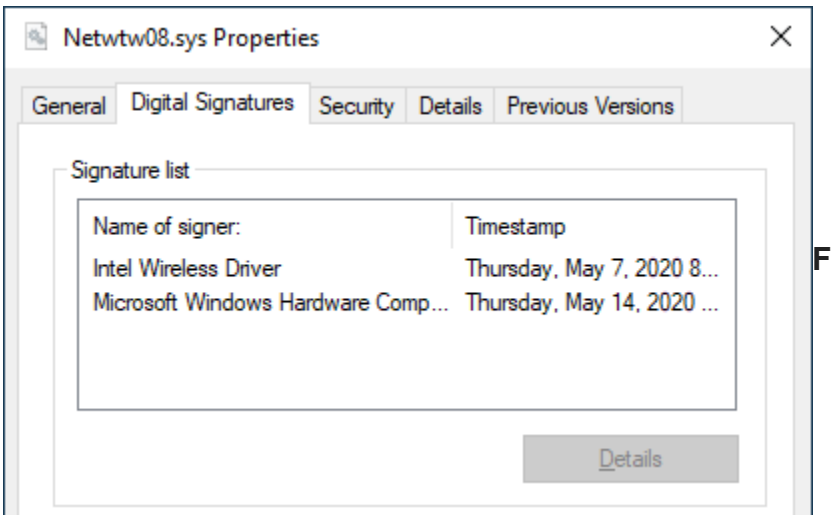

Figure 4: List of digital signature via Explorer

Another storage of CRLs is `Certificate Storage` accessible via the Certificate Manager Tool; see **Figure 5**. This storage can be managed by local admin or domain.

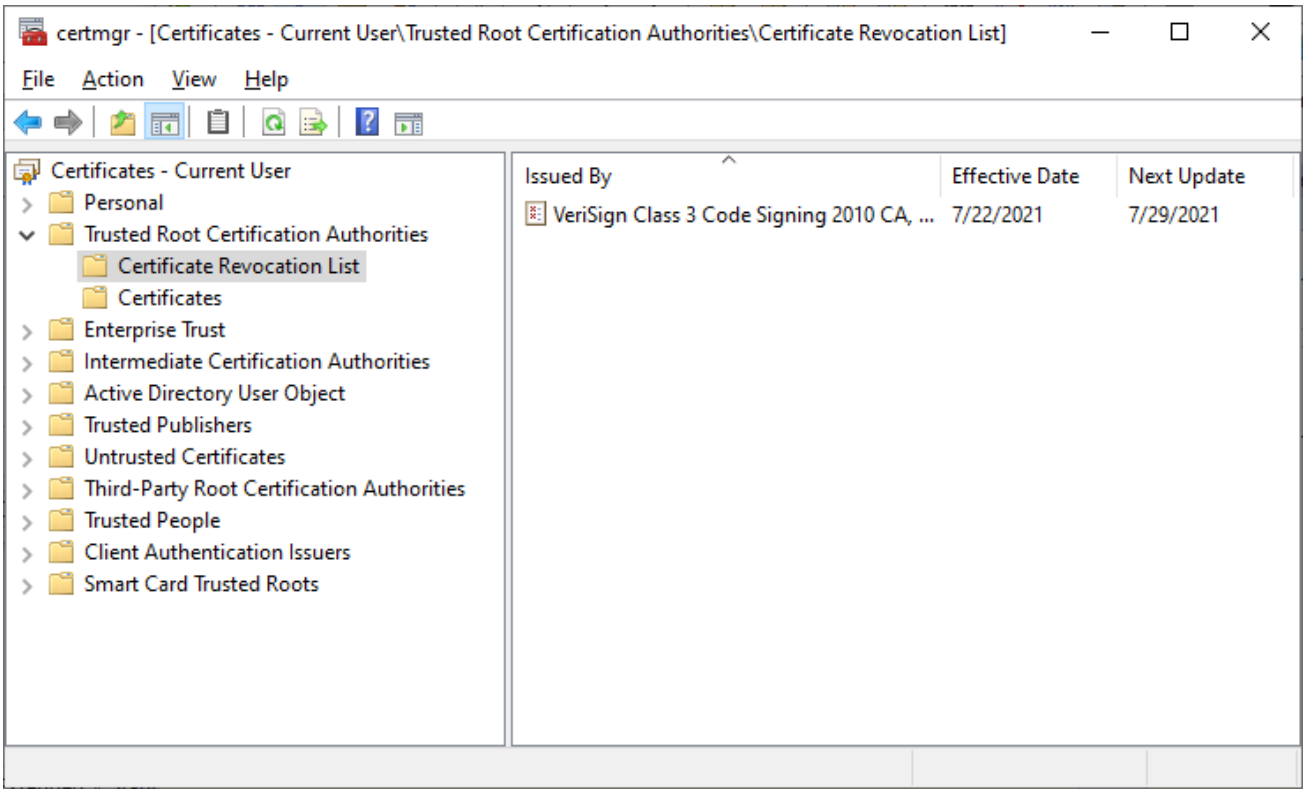

Figure 5: Certificate Manager Tool

3.2 CRL Verification by UAC

UAC itself does not verify code signature, or more precisely chain of trust, online; it is probably because of system performance. The UAC codesign verification checks the signature and CRL via `CryptnetUrlCache` and the system `Certificate storage`. Therefore, UAC marks malicious software, signed with the revoked certificate, as the unknown publisher because `CryptnetUrlCache` is not up-to-date initially.

Suppose the user adds appropriate CRL into `Certificate Storage` manually using this command:

```
certutil -addstore -f Root CSC3-2010.crl
```

In that case, UAC will successfully detect the software as suspicious, and UAC displays this message: "*An administrator has blocked you from running this app.*" without assistance from Cryptographic Services and therefore offline.

The Windows update does not include CRLs of cross-certificate authorities that Microsoft authorized to codesign of Windows files. This fact has been verified via Windows updates and version as follow:

- Windows Malicious Software Removal Tool x64 – v5.91 (KB890830)
- 2021-06 Cumulative update for Windows 10 Version 20H2 for x64-based System (KB5003690)
- Windows version: 21H1 (OB Build 19043.1110)

In summary, UAC checks digital signatures via the system local storage of CRL and does not verify code signature online.

3.3 Windows Driver and CRL

Returning briefly to the codesign of the Windows driver that is required by the kernel. The kernel can verify the signature offline, but it cannot check CRL since Cryptographic Services and network access are not available at boot time.

Turning now to the experimental evidence on the offline CRL verification of drivers. It is evident in the case described in Section 3.2 that UAC can verify CRL offline. So, the tested hypothesis is that the Windows kernel could verify the CRL of a loaded driver offline if an appropriate CRL is stored in `Certificate Storage`. We used the DirtyMoe malware to deploy the malicious driver that is signed with the revocated certificate. The corresponding CRL of the revoked certificate has been imported into `Certificate Storage`. However, the driver was still active after the system restart. It indicates that the rootkit was successfully loaded into the kernel, although the driver certificate has been revoked and the current CRL was imported into `Certificate Storage`. There is a high probability that the kernel avoids the CRL verification even from local storage.

4. Certificate Analysis

The scope of this research are three certificates as follow:

Beijing Kate Zhanhong Technology Co.,Ltd.
Valid From: 28-Nov-2013 (2:00:00)
Valid To: 29-Nov-2014 (1:59:59)
SN: `3c5883bd1dbcd582ad41c8778e4f56d9`
Thumbprint: `02a8dc8b4aead80e77b333d61e35b40fbbb010a0`
Revocation Status: Revoked on 22-May-2014 (9:28:59)
CRL Distribution Point: http://cs-g2-crl.thawte.com/ThawteCSG2.crl
IoCs:
`88D3B404E5295CF8C83CD204C7D79F75B915D84016473DFD82C0F1D3C375F968`
`376F4691A80EE97447A66B1AF18F4E0BAFB1C185FBD37644E1713AD91004C7B3`
`937BF06798AF9C811296A5FC1A5253E5A03341A760A50CAC67AEFEDC0E13227C`

Beijing Founder Apabi Technology Limited
Valid From: 22-May-2018 (2:00:00)
Valid To: 29-May-2019 (14:00:00)
SN: `06b7aa2c37c0876ccb0378d895d71041`
Thumbprint: `8564928aa4fbc4bbecf65b402503b2be3dc60d4d`
Revocation Status: Revoked on 22-May-2018 (2:00:01)
CRL Distribution Point: http://crl3.digicert.com/sha2-assured-cs-g1.crl
IoCs:

`B0214B8DFCB1CC7927C5E313B5A323A211642E9EB9B9F081612AC168F45BF8C2`
`5A4AC6B7AAB067B66BF3D2BAACEE300F7EDB641142B907D800C7CB5FCCF3FA2A`
`DA720CCAFE572438E415B426033DACAFBA93AC9BD355EBDB62F2FF01128996F7`

Shanghai Yulian Software Technology Co., Ltd. (上海域联软件技术有限公司)
Valid From: 23-Mar-2011 (2:00:00)
Valid To: 23-Mar-2012 (1:59:59)
SN: `5f78149eb4f75eb17404a8143aaeaed7`
Thumbprint: `31e5380e1e0e1dd841f0c1741b38556b252e6231`
Revocation Status: Revoked on 18-Apr-2011 (10:42:04)
CRL Distribution Point: http://csc3-2010-crl.verisign.com/CSC3-2010.crl
IoCs:
`15FE970F1BE27333A839A873C4DE0EF6916BD69284FE89F2235E4A99BC7092EE`
`32484F4FBBECD6DD57A6077AA3B6CCC1D61A97B33790091423A4307F93669C66`
`C93A9B3D943ED44D06B348F388605701DBD591DAB03CA361EFEC3719D35E9887`

4.1 Beijing Kate Zhanhong Technology Co.,Ltd.

We did not find any relevant information about *Beijing Kate Zhanhong Technology* company. Avast captured 124 hits signed with the Beijing Kate certificates from 2015 to 2021. However, prediction to 2021 indicates a downward trend; see **Figure 6**. Only 19 hits have been captured in the first 7 months of this year, so the prediction for this year (2021) is approx. 30 hits signed with this certificate.

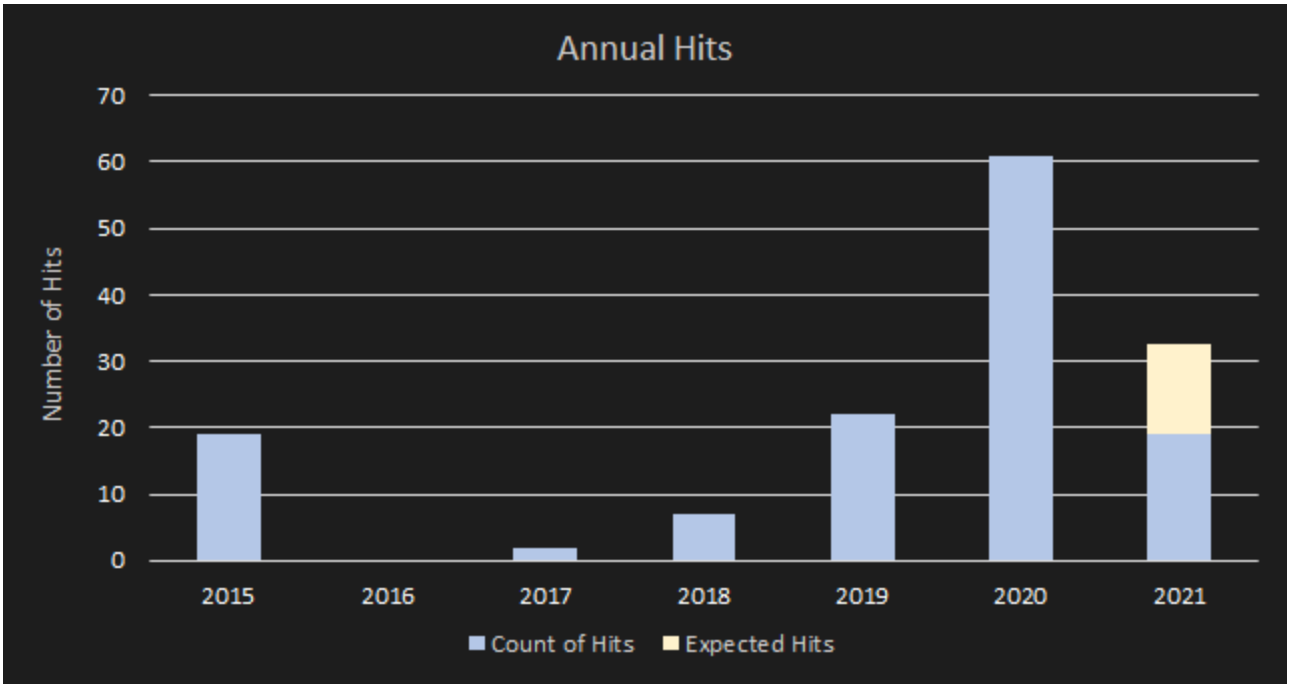

Figure 6: Number of hits and prediction of Beijing Kate

Ninety-five percent of total hits have been located in Asia; see **Table 1** for detailed GeoIP distribution of the last significant years.

Year / Country	CN	TW	UA	MY	KR	HK
2015	15	1		3		
2018	3	1				2
2019	19		2			
2020	12	46	1			
2021	11				8	
Total	60	48	3	3	8	2

Table 1: Geological distribution of Beijing Kate certificates

The malware authors use this certificate to sign Windows system drivers in 61 % of monitored hits, and Windows executables are signed in 16 % of occurrences. Overall, it seems that the malware authors focus on Windows drivers, which must be signed with any verified certificates.

4.2 Beijing Founder Apabi Technology Limited

The *Beijing Founder Apabi Technology Co., Ltd.* was founded in 2001, and it is a professional digital publishing technology and service provider under Peking University Founder Credit Group. The company also develops software for e-book reading like Apabi Reader [4].

The Beijing Founder certificate has been observed, as malicious, in 166 hits with 18 unique SHAs. The most represented sample is an executable called “Linking.exe” which was hit in 8 countries, most in China; exactly 71 cases. The first occurrence and peak of the hits were observed in 2018. The incidence of hits was an order of magnitude lower in the previous few years, and we predict an increase in the order of tens of hits, as **Figure 7** illustrates.

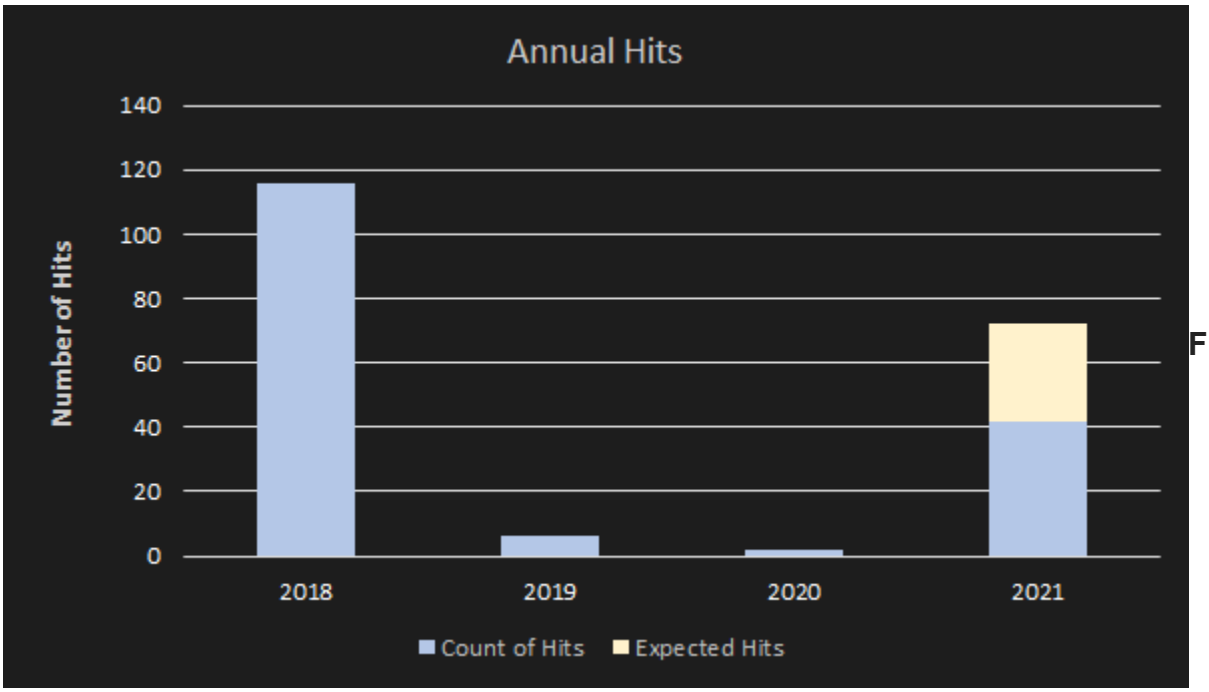

Figure 7: Number of hits and prediction of Beijing Founder

4.3 Shanghai Yulian Software Technology Co., Ltd.

The *Shanghai Yulian Software Technology Co.* was founded in 2005, which provides network security guarantees for network operators. Its portfolio covers services in the information security field, namely e-commerce, enterprise information security, security services, and system integration [5].

The *Shanghai Yulian Software* certificate is the most widespread compared to others. Avast captured this certificate in 10,500 cases in the last eight years. Moreover, the prevalence of samples signed with this certificate is on the rise, although the certificate was revoked by its issuer in 2011. In addition, the occurrence peak was in 2017 and 2020. We assume a linear increase of the incidence and a prediction for 2021 is based on the first months of 2021. And therefore, the prediction for 2021 indicates that this revoked certificate will also dominate in 2021; see the trend in **Figure 8**. We have no record of what caused the significant decline in 2018 and 2019.

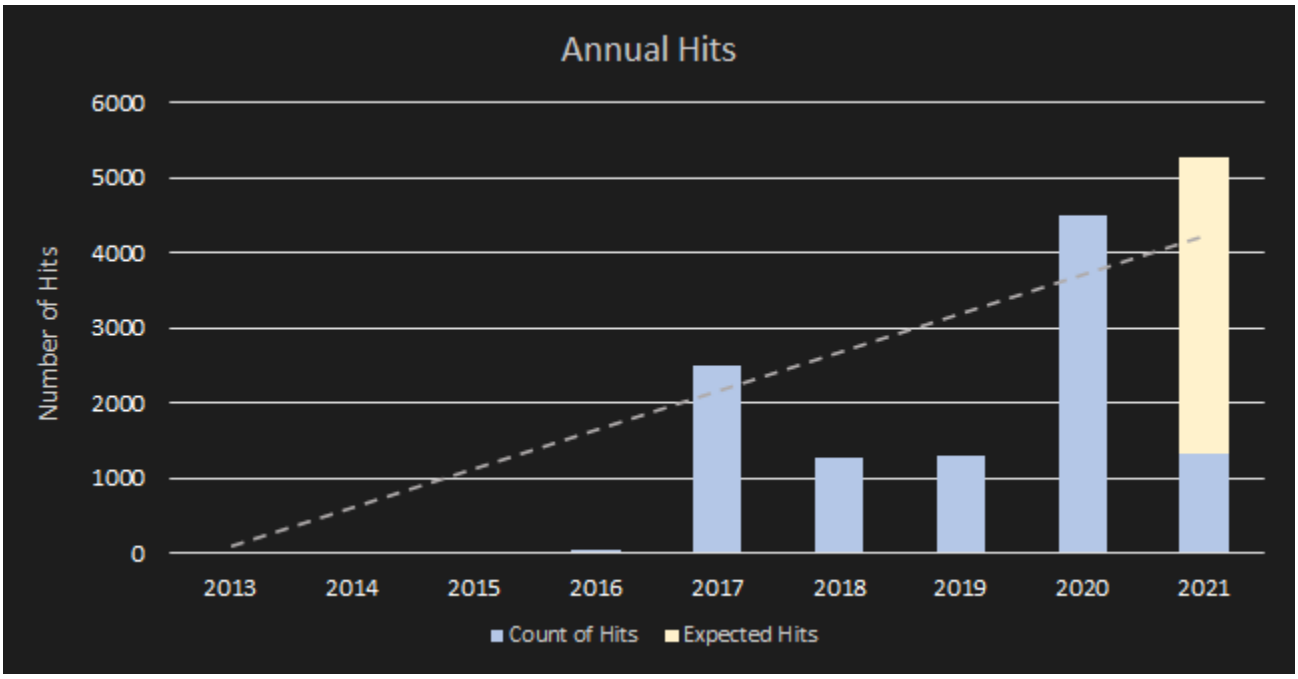

Figure 8: Number of hits and prediction of Shanghai Yulian Software

The previous certificates dominate only in Asia, but the *Shanghai Yulian Software* certificate prevails in Asia and Europe, as **Figure 9** illustrates. The dominant countries are Taiwan and Russia. China and Thailand are on a close hinge; see **Table 2** for detailed country distribution, several countries selected.

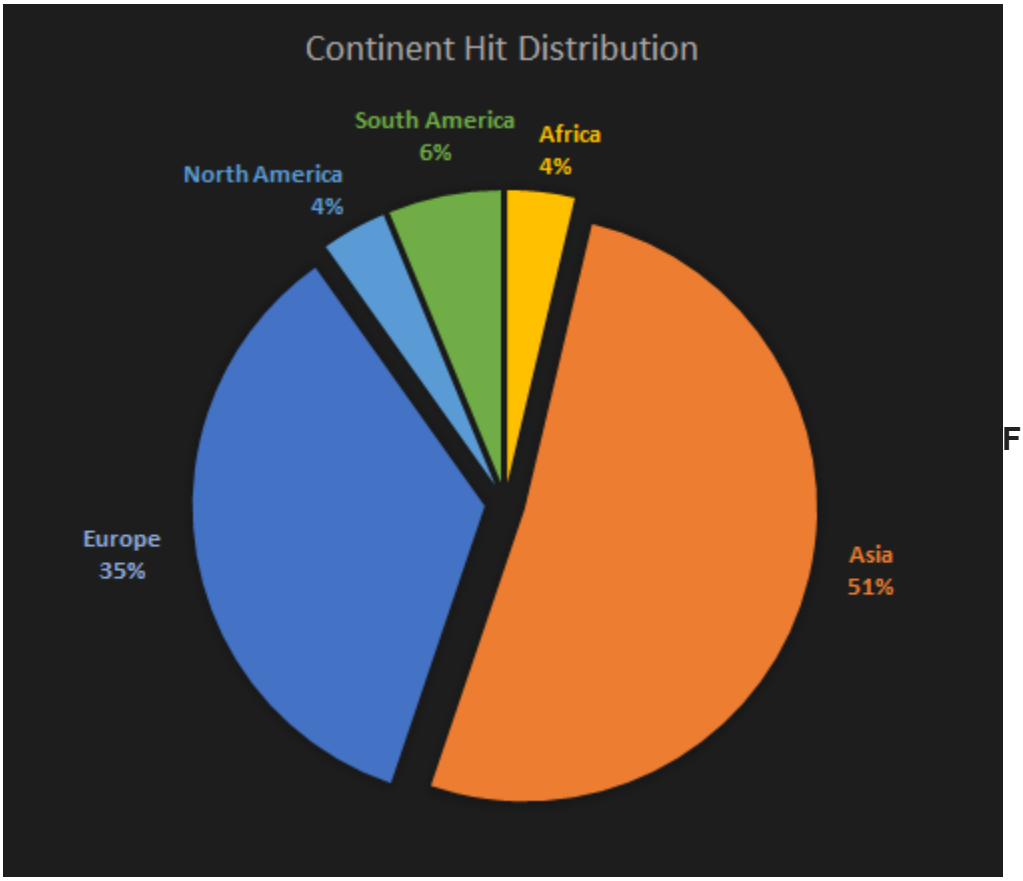

Figure 9: Distribution of hits in point of continent view for Shanghai Yulian Software

Year / Country	TW	RU	CN	TH	UA	BR	EG	HK	IN	CZ
2016	6		18			2				
2017	925	28	57			4				1
2018	265	31	79	4	3	26	4	1		2
2019	180	54	56	10	9	98	162	5	45	1
2020	131	1355	431	278	232	112	17	168	86	24
Total	1507	1468	641	292	244	242	183	174	131	28

Table 2: Geological distribution of Shanghai Yulian Software certificate

As in previous cases, the malware authors have been using *Shanghai Yulian Software* certificates to sign Windows drivers in 75 % of hits. Another 16 % has been used for EXE and DLL executable; see **Figure 10** for detailed distribution.

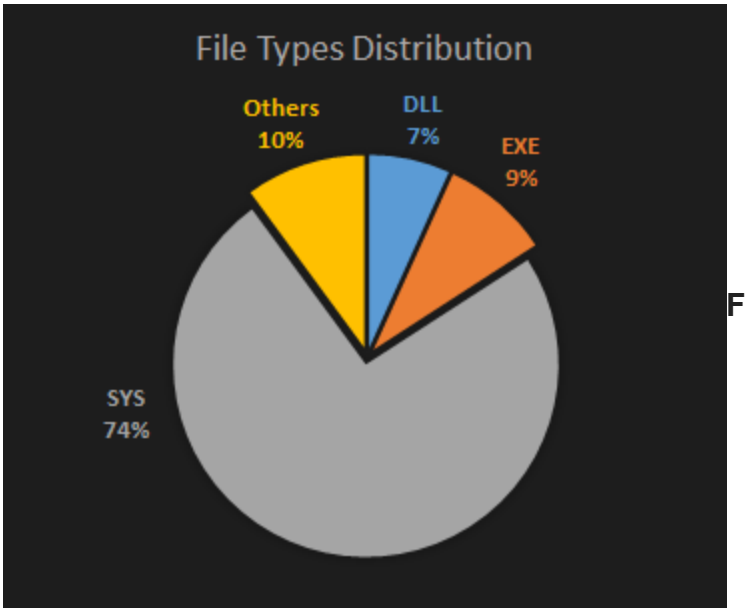

Figure 10: Distribution of file types for Shanghai Yulian Software

4.3.1 **Analysis of Top 10 Captured Files**

We summarize the most frequent files in this subchapter. **Table 3** shows the occurrence of the top 10 executables that are related to user applications. We do not focus on DirtyMoe drivers because this topic is recorded at DirtyMoe: Rootkit Driver.

Hit File Name	Number of Hits	Hit File Name	Number of Countries
WFP_Drive.sys	1056	Denuvo64.sys	81
LoginDrvS.sys	1046	WsAP-Filmora.dll	73
Denuvo64.sys	718	SlipDrv7.sys	20

WsAP-Filmora.dll	703	uTorrent.exe	47
uTorrent.exe	417	RTDriver.sys	41
SlipDrv7.sys	323	SlipDrv10.sys	42
LoginNpDriveX64.sys	271	SbieDrv.sys	22
RTDriver.sys	251	SlipDrv81.sys	16
SlipDrv10.sys	238	ONETAP V4.EXE	16
SbieDrv.sys	189	ECDM.sys	13

Table 3: The most common occurrence of the file names: hit count and number of countries
We have identified the five most executables signed with *Shanghai Yulian Software* certificate. The malware authors target popular and commonly used user applications such as video editors, communication applications, and video games.

1) WFP_Drive.sys

The WFP driver was hit in a folder of App-V (Application Virtualization) in `C:\ProgramData`. The driver can filter network traffic similar to a firewall [6]. So, malware can block and monitor arbitrary connections, including transferred data. The malware does not need special rights to write into the `ProgramData` folder. The driver and App-V's file names are not suspicious at first glance since App-V uses Hyper-V Virtual Switch based on the WFP platform [7].

Naturally, the driver has to be loaded with a process with administrator permission. Subsequently, the process must apply for a specific privilege called `SeLoadDriverPrivilege` (load and unload device drivers) to successful driver loading. The process requests the privilege via API call `EnablePriviliges` that can be monitored with AVs. Unfortunately, Windows loads drivers with revoked certificates, so the whole security mechanism is inherently flawed.

2) LoginDrvS.sys

Further, the `LoginDrvS` driver uses a similar principle as the WFP driver. The malware camouflages its driver in a commonly used application folder. It uses the `LineageLogin` application folder that is used as a launcher for various applications, predominantly video games. `LineageLogin` is implemented in Pascal [8]. The `LineageLogin` itself contains a suspicious embedded DLL. However, it is out of this topic. Both drivers (WFP_Drive.sys and LoginDrvS.sys) are presumably from the same malware author as their PDB paths indicate:

- `f:\projects\c++\win7pgkill\amd64\LoginDrvS.pdb` (10 unique samples)

- `f:\projects\c++\wfp\objfre_win7_amd64\amd64\WFP_Drive.pdb` (8 unique samples)

Similar behavior is described in *Trojan.MulDrop16.18994*, see [9].

3) Denuvo64.sys

Denuvo is an anti-piracy protection technique used by hundreds of programming companies around the world [10]. This solution is a legal rootkit that can also detect cheat mode and other unwanted activity. Therefore, Denuvo has to use a kernel driver.

The easy way to deploy a malicious driver is via cracks that users still download abundantly. The malware author packs up the malicious driver into the cracks, and users prepare a rootkit backdoor within each crack run. The malicious driver performs the function that users expect, e.g., anti-anti cheating or patching of executable, but it also services a backdoor very often.

We have revealed 8 unique samples of Denuvo malicious drivers within 1046 cases. The most occurrence is in Russia; see **Figure 11**. The most hits are related to video game cracks as follows: *Constructor, Injustice 2, Prey, Puyo Puyo Tetris, TEKKEN 7 – Ultimate Edition, Total War – Warhammer 2, Total.War – Saga Thrones of Britannia*.

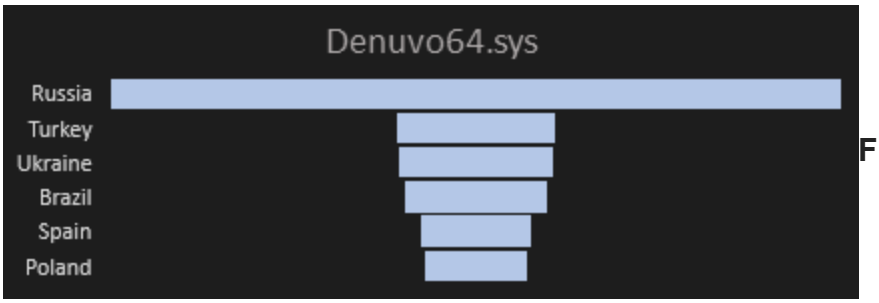

Figure 11: Distribution of Denuvo driver across the countries

4) WsAP-Filmora.dll

Wondershare Filmora is a simple video editor which has free and paid versions [11]. So, there is a place for cracks and malware activities. All hit paths point to crack activity where users tried to bypass the trial version of the Filmora editor. No PDB path has been captured; therefore, we cannot determine the malware authors given other samples signed with the same certificates. WsAP-Filmora.dll is widespread throughout the world (73 countries); see the detailed country distribution in **Figure 12**.

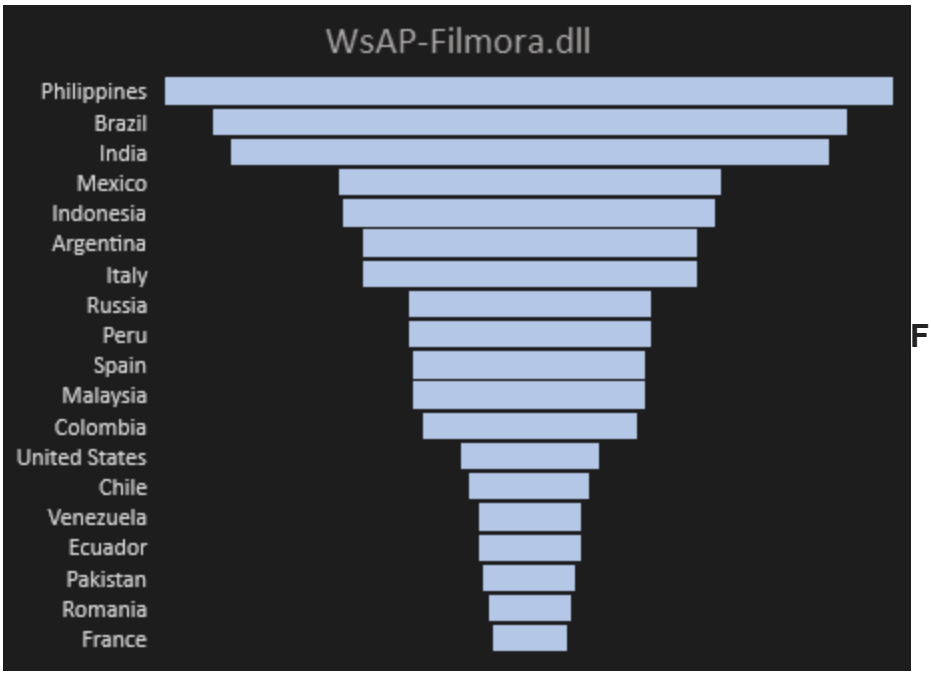

Figure 12: Filmora DLL distribution across the countries

5) µTorrent.exe

µTorrent is a freeware P2P client for the BitTorrent network [12]. Avast has detected 417 hits of malicious µTorrent applications. The malicious application is based on the original µTorrent client with the same functionality, GUI, and icon. If a user downloads and runs this updated µTorrent client, the application's behavior and design are not suspicious for the user.

The original application is signed with Bit Torrent, Inc. certificate, and the malicious version was signed with *Shanghai Yulian Software* certificate. Everyday users, who want to check the digital signature, see that the executable is signed. However, information about the signature's suspiciousness is located in details that are not visible at first glance, **Figure 13**.

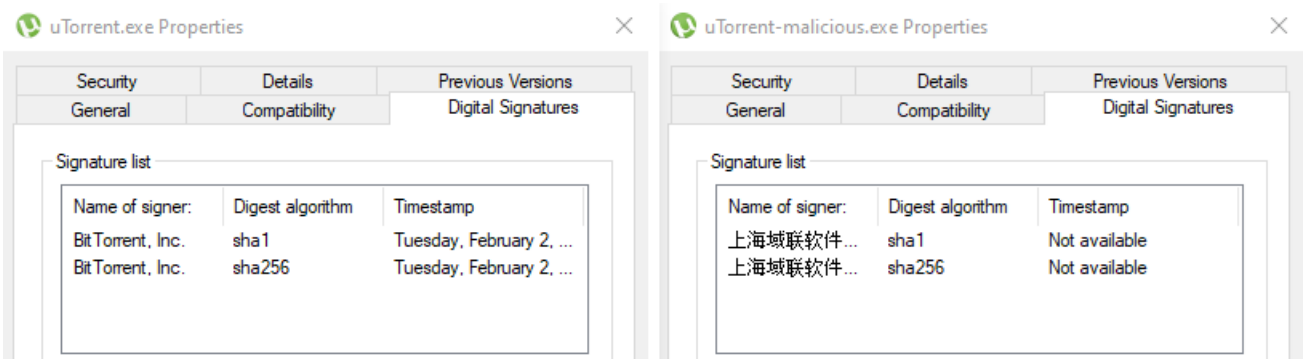

Figure 13: Digital signatures of original and malicious µTorrent client

The malicious signature contains Chinese characters, although the sample's prevalence points to Russia. Moreover, China has no hits. **Figure 14** illustrates a detailed country distribution of the malicious µTorrent application.

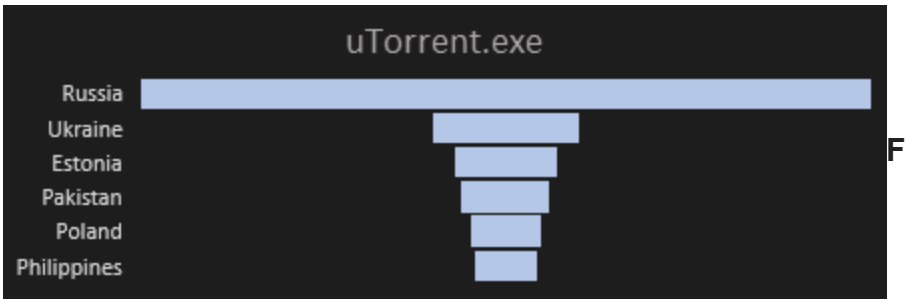

Figure 14: Distribution of µTorrent client across the countries

4.3.2 **YDArk Rootkit**

We have found one Github repository which contains a YDArk tool that monitors system activities via a driver signed with the compromised certificate. The last sign was done in May 2021; hence it seems that the stolen certificate is still in use despite the fact that it was revoked 10 years ago. The repository is managed by two users, both located in China. The YDArk driver has been signed with the compromise certificates by ScriptBoy2077, who admitted that the certificate is compromised and expired by this note:

"*I have signed the .sys file with a compromised, expired certificate, may it is helpful to those my friends who don't know how to load an unsigned driver.*"

The private key of the compromised certificate is not available on the clearnet. We looked for by the certificate name, serial number, hashes, etc.; hence we can assume that the certificate and private key can be located in the darknet and shared within a narrow group of malware authors. Consequently, the relationship between YDArk, more precisely ScriptBoy2077, and the DirtyMoe driver is compelling. Additionally, C&C DirtyMoe servers are also located in most cases in China, as we described in DirtyMoe: Introduction and General Overview.

5. Discussion

The most interesting finding of this research was that Windows allows loading drivers signed with revoked certificates even if an appropriate CRL is locally stored. Windows successfully verifies the signature revocation offline for user-mode applications via UAC. However, if Windows loads drivers with the revoked certificate into the kernel despite the up-to-date CRL, it is evident that the Windows kernel avoids the CRL check. Therefore, it is likely that the missing implementation of CRL verification is omitted due to the performance at boot time. It is a hypothesis, so further research will need to be undertaken to fully understand the process and implementation of driver loading and CRL verification.

Another important finding was that the online CRL is not queried by UAC in digital signatures verification for user-mode applications, which require higher permission. It is somewhat surprising that UAC blocks the applications only if the user manually checks the chain of trust before the application is run; in other words, if the current CRL is downloaded

and is up-to-date before the UAC run. So, the evidence suggests that neither UAC does not fully protect users against malicious software that has been signed with revoked certificates. Consequently, the users should be careful when UAC appears.

The DirtyMoe's driver has been signing with three certificates that we analyzed in this research. Evidence suggests that DirtyMoe's certificates are used for the code signing of other potentially malicious software. It is hard to determine how many malware authors use the revoked certificates precisely. We classified a few clusters of unique SHA samples that point to the same malware authors via PDB paths. However, many other groups cannot be unequivocally identified. There is a probability that the malware authors who use the revoked certificates may be a narrow group; however, further work which compares the code similarities of signed software is required to confirm this assumption. This hypothesis is supported by the fact that samples are dominantly concentrated in Taiwan and Russia in such a long time horizon. Moreover, leaked private keys are often available on the internet, but the revoked certificates have no record. It could mean that the private keys are passed on within a particular malware author community.

According to these data, we can infer that the *Shanghai Yulian Software Technology* certificate is the most common use in the wild and is on the rise, although the certificate was revoked 10 years ago. Geological data demonstrates that the malware authors target the Asian and European continents primarily, but the reason for this is not apparent; therefore, questions still remain. Regarding types of misused software, the analysis of samples signed with the *Shanghai Yulian Software* certificate confirmed that most samples are rootkits deployed together with cracked or patched software. Other types of misused software are popular user applications, such as video games, communication software, video editors, etc. One of the issues is that the users observe the correct behavior of suspicious software, but malicious mechanisms or backdoors are also deployed.

In summary, the DirtyMoe's certificates are still used for code signing of other software than the DirtyMoe malware, although the certificates were revoked many years ago.

6. Conclusion

The malware analysis of the DirtyMoe driver (rootkit) discovered three certificates used for codesign of suspicious executables. The certificates have been revoked in the middle of their validity period. However, the certificates are still widely used for codesign of other malicious software.

The revoked certificates are principally misused for the code signature of malicious Windows drivers (rootkits) because the 64bit Windows system has begun requiring the driver signatures for greater security. On the other hand, Windows does not implement the verification of signatures via CRL. In addition, the malware authors abuse the certificates to sign popular software to increase credibility, whereas the software is patched with malicious payloads. Another essential point is that UAC verifies the signatures against the local CRL,

which may not be up-to-date. UAC does not download the current version of CRL when users want to run software with the highest permission. Consequently, it can be a serious weakness.

One source of uncertainty is the origin of the malware authors who misused the revoked certificates. Everything points to a narrow group of the author primarily located in China. Further investigation of signed samples are needed to confirm that the samples contain payloads with similar functionality or code similarities.

Overall, these results and statistics suggest that Windows users are still careless when running programs downloaded from strange sources, including various cracks and keygens. Moreover, Windows has a weak and ineffective mechanism that would strongly warn users that they try to run software with revoked certificates, although there are techniques to verify the validity of digital signatures.

References

[1] Driver Signing
[2] Cross-Certificates for Kernel Mode Code Signing
[3] How Windows Handles Running Files Whose Publisher's Certificate(s) Have Been Revoked
[4] Beijing Founder Apabi Technology Limited
[5] Shanghai Yulian Software Technology Co., Ltd.
[6] WFP network monitoring driver (firewall)
[7] Hyper-V Virtual Switch
[8] Lineage Login
[9] Dr. Web: Trojan.MulDrop16.18994
[10] Denuvo
[11] Filmora
[12] µTorrent

Tagged asCertificate, DirtyMoe, Rootkit, series