# Old certificate, new signature: Open-source tools forge signature timestamps on Windows drivers

By Chris Neal

TUESDAY, JULY 11, 2023 13:07

THREAT ADVISORY

- Cisco Talos has observed threat actors taking advantage of a Windows policy loophole that allows the signing and loading of cross-signed kernel mode drivers with signature timestamp prior to July 29, 2015.
- Actors are leveraging multiple open-source tools that alter the signing date of kernel mode drivers to load malicious and unverified drivers signed with expired certificates.

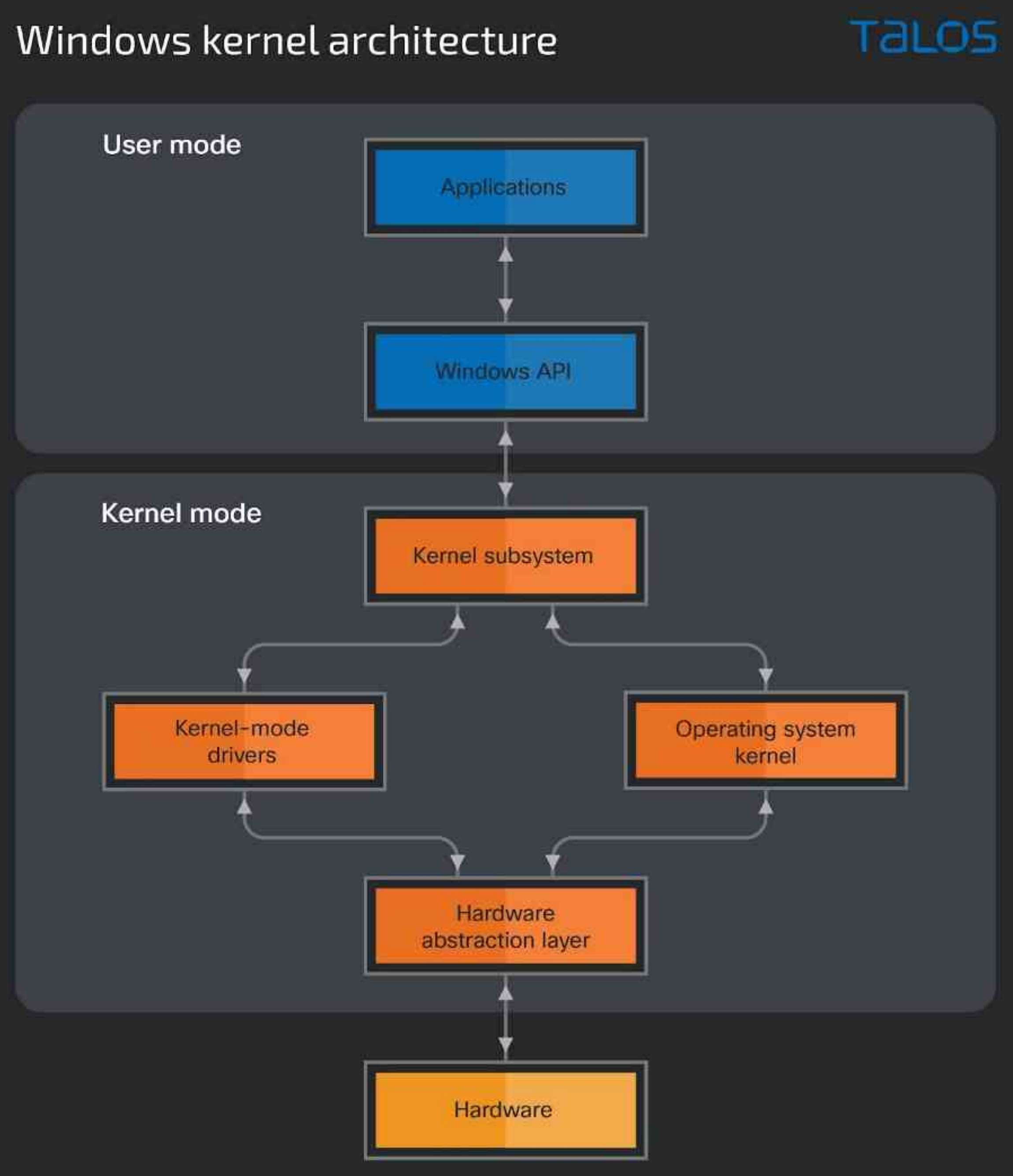
- We have observed over a dozen code signing certificates with keys and passwords contained in a PFX file hosted on GitHub used in conjunction with these open source tools.
- The majority of drivers we identified that contained a language code in their metadata have the Simplified Chinese language code, suggesting the actors using these tools are frequently used by native Chinese speakers.
- Cisco Talos has further identified an instance of one of these open-source tools being used to re-sign cracked drivers to bypass digital rights management (DRM).
- We have released a second blog post alongside this one demonstrating real-world abuse of this loophole by an undocumented malicious driver named RedDriver.

*During the research phase for this blog post we reached out to Microsoft to notify them of our findings. In response, Microsoft has blocked all certificates discussed in this blog and has released an advisory. We would like to thank the Microsoft team for their assistance and cooperation in mitigating this threat.*

## Malicious drivers pose a significant threat

The Windows operating system (OS) is split into two layers, or “modes”: user mode, where the files and applications that users interact with reside, and the kernel mode, where kernel mode drivers and the underpinnings of Windows perform the necessary functions to run the system. Drivers can facilitate communication between these modes via the Windows API through a series of functions contained within system libraries.

Splitting the operating system into two modes creates a highly controlled logical barrier between the average user and the Windows kernel. This barrier is critical to maintaining the integrity and security of the OS, as access to the kernel provides complete access to a system. As such, leveraging a malicious driver can allow an attacker to pass through this barrier, resulting in total compromise of the target system.



*Windows kernel architecture*

Starting in Windows Vista 64-bit to combat the threat of malicious drivers, Microsoft began to require kernel-mode drivers to be digitally signed with a certificate from a verified certificate authority. Without signature enforcement, malicious drivers would be extremely difficult to defend against as they can easily evade anti-malware software and endpoint detection. Requiring signatures on drivers is the most critical component of defending against malicious kernel mode drivers.

Another issue that malicious drivers pose is the difficulty of conducting analysis of samples. Typical sandboxes used to analyze malware do not have the capability to monitor all of the behavior of a driver, meaning much of the analysis must be conducted manually. To further the difficulties posed by driver analysis, threat actors have begun to use code obfuscation on the drivers they deploy with tools such as VMProtect.

From an attacker's perspective, the advantages of leveraging a malicious driver include, but are not limited to, evasion of endpoint detection, the ability to manipulate system and user mode processes, and maintained persistence on an infected system. These advantages provide a significant incentive for attackers to discover ways to bypass the Windows driver signature policies.

Cisco Talos has observed threat actors taking advantage of a Windows policy loophole that allows the forging of signatures on kernel-mode drivers, thereby bypassing the certificate policies within Windows. As we will demonstrate below, this is facilitated by the use of open-source tooling and non-revoked certificates that either expired before or were issued prior to July 29, 2015.

## Windows driver policy loophole allows signature timestamp forging

Starting with Windows 10 version 1607, Microsoft updated its driver signing policy to no longer allow new kernel-mode drivers that have not been submitted to, and signed by its Developer Portal. This process is intended to ensure that drivers meet Microsoft's requirements and security standards. In an effort to maintain the functionality and compatibility of older drivers, Microsoft created exceptions for the following:

1. *The PC was upgraded from an earlier release of Windows to Windows 10, version 1607.*
2. *Secure Boot is off in the BIOS.*
3. *Drivers was [sic] signed with an end-entity certificate issued prior to July 29th 2015 that chains to a supported cross-signed CA.*

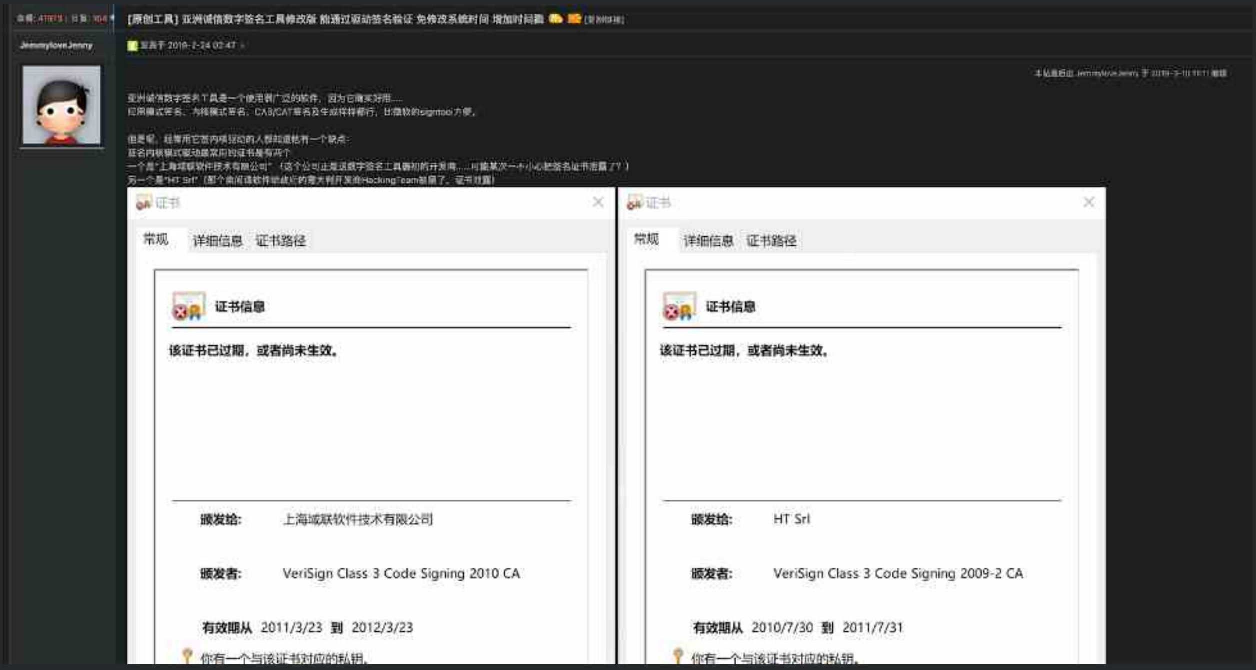
The third exception creates a loophole that allows a newly compiled driver to be signed with non-revoked certificates issued prior to or expired before July 29, 2015, provided that the certificate chains to a supported cross-signed certificate authority. If a driver is successfully signed this way, it will not be prevented from being installed and started as a service. As a result, multiple open source tools have been developed to exploit this loophole. This is a known technique though often overlooked despite posing a serious threat to Windows systems and being relatively easy to perform due in part to the tooling being publicly available.

## Actors leveraging open-source tools to take advantage of loophole

Talos has observed multiple threat actors taking advantage of the aforementioned Windows policy loophole to deploy thousands of malicious, signed drivers without submitting them to Microsoft for verification. During our research we identified threat actors leveraging HookSignTool and FuckCertVerifyTimeValidity, signature timestamp forging tools that have been publicly available since 2019 and 2018 respectively, to deploy these malicious drivers. While they have gained some popularity within the game cheat development community, we have observed the use of these tools on malicious Windows drivers unrelated to game cheats. In our blog post revealing RedDriver, a driver-based browser hijacker, we demonstrate a real-world example of HookSignTool being used in a malicious context.

### HookSignTool

HookSignTool is a driver signature forging tool that alters the signing date of a driver during the signing process through a combination of hooking into the Windows API and manually altering the import table of a legitimate code signing tool. It was originally released in 2019 on the Chinese software cracking forum "52pojie[.]cn" by its author "JemmyLoveJenny" and has been available on GitHub since at least 2020.


查看: 41873 | 回复: 164

[原创工具] 亚洲诚信数字签名工具修改版 能通过驱动签名验证 免修改系统时间 增加时间戳

JemmyloveJenny

发表于 2019-2-24 02:47

本帖最后由 JemmyloveJenny 于 2019-3-10 11:11 编辑

亚洲诚信数字签名工具是一个使用最广泛的软件，因为它确实好用……
应用模式签名、内核模式签名、CAB/CAT签名及生成样样都行，比微软的signtool方便。

但是呢，经常用它签内核驱动的人都知道他有一个缺点：
签名内核模式驱动最常用的证书是有两个
一个是"上海域联软件技术有限公司"（这个公司正是该数字签名工具最初的开发商……可能某次一不小心把签名证书泄露了？）
另一个是"HT Srl"（[illegible]HackingTeam被黑了，证书泄露）

证书 | 常规 详细信息 证书路径

证书信息

该证书已过期，或者尚未生效。

颁发给: 上海域联软件技术有限公司

颁发者: VeriSign Class 3 Code Signing 2010 CA

有效期从 2011/3/23 到 2012/3/23

你有一个与该证书对应的私钥。

证书 | 常规 详细信息 证书路径

证书信息

该证书已过期，或者尚未生效。

颁发给: HT Srl

颁发者: VeriSign Class 3 Code Signing 2009-2 CA

有效期从 2010/7/30 到 2011/7/31

你有一个与该证书对应的私钥。

*Original release of HookSignTool on 52pojie*

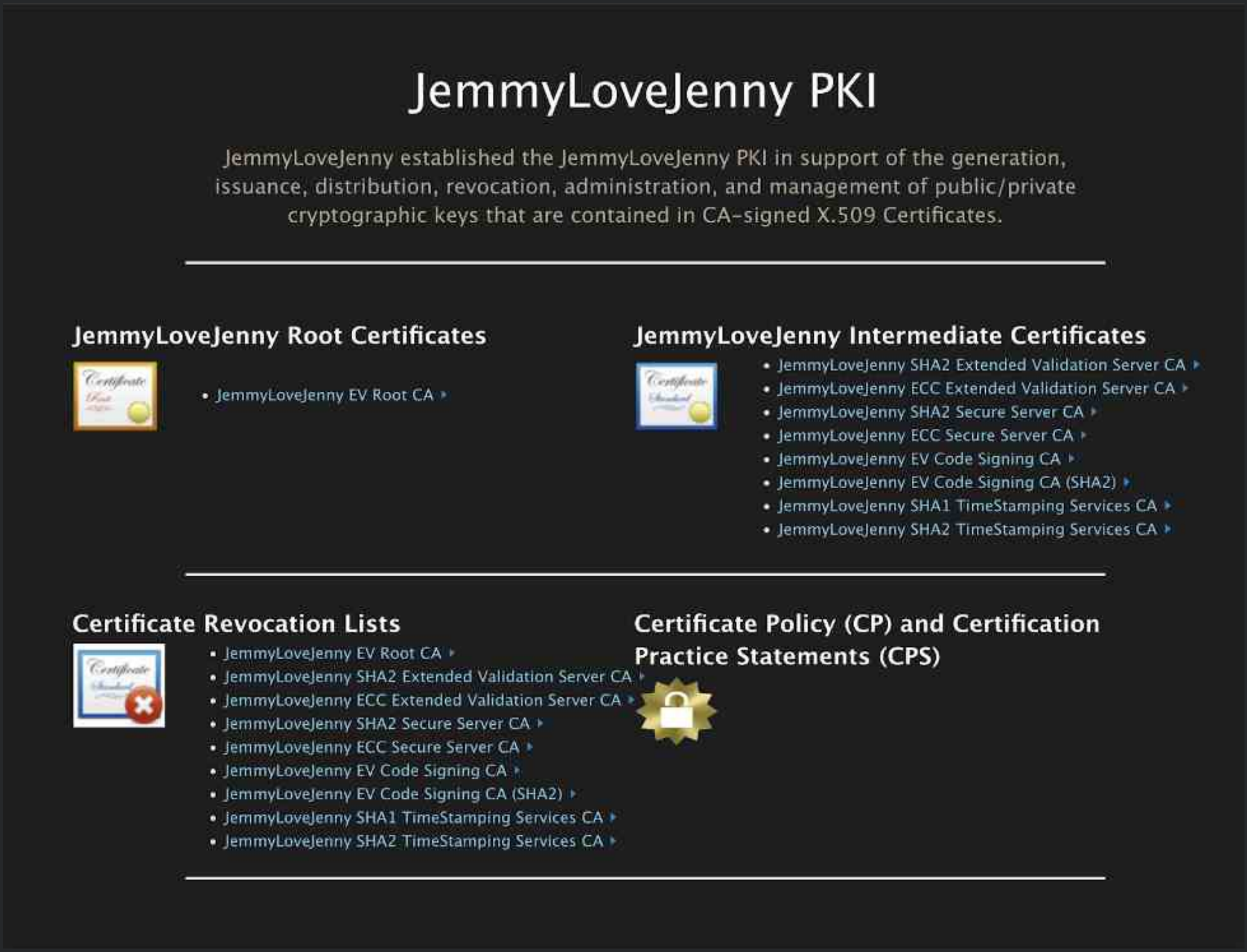
In the readme file on the HookSignTool Github, the author specifically mentions the use of the Chinese code signing utility "Digital Signature Tool for Asian Integrity" (用于亚洲诚信数字签名工

具) in conjunction with HookSignTool, although HookSignTool can theoretically be used with any code signing utility. This universal compatibility is achieved through the use of Microsoft Detours, a software package used for instrumenting and monitoring Windows API calls.

Before HookSignTool can be used, several steps must be taken to render it functional. First, it must be compiled and manually added into the import table of the legitimate signing tool as "*HookSigntool.dll!attach*". To specify the date to forge, the user can pass a date as a command line argument or create a configuration file in the same directory as the code signing tool. Upon execution of the legitimate code signing tool, HookSignTool is loaded into the process and can then intercept and alter the signing date of the target driver.

HookSignTool requires another edit of the code signing tool; the URL strings of the timestamp authorities embedded within the legitimate tool must be replaced with the string "{CustomTimestampMarker-SHA1}" in hexadecimal with 0x00 in between each character. For example, "http[:]//timestamp.digicert.com" would be replaced with hexadecimal shown below:

7b 00 43 00 75 00 73 00 74 00 6f 00 6d 00 54 00 69 00 6d 00 65 00 73 00 74 00 61 00 6d 00 70 00 4d 00 61 00 72 00 6b 00 65 00 72 00 2d 00 53 00 48 00 41 00 31 00 7d 00 00 00 00 00
During the signing process, HookSignTool searches for this tag and replaces it with "https://pki[.]jemmylovejenny[.]tk/". In addition, the JemmyLoveJenny EV Root CA certificate must also be installed on the system signing the driver, available on the author's personal website as shown below:



JemmyLoveJenny PKI

JemmyLoveJenny established the JemmyLoveJenny PKI in support of the generation, issuance, distribution, revocation, administration, and management of public/private cryptographic keys that are contained in CA-signed X.509 Certificates.

JemmyLoveJenny Root Certificates

- JemmyLoveJenny EV Root CA

JemmyLoveJenny Intermediate Certificates

- JemmyLoveJenny SHA2 Extended Validation Server CA
- JemmyLoveJenny ECC Extended Validation Server CA
- JemmyLoveJenny SHA2 Secure Server CA
- JemmyLoveJenny ECC Secure Server CA
- JemmyLoveJenny EV Code Signing CA
- JemmyLoveJenny EV Code Signing CA (SHA2)
- JemmyLoveJenny SHA1 TimeStamping Services CA
- JemmyLoveJenny SHA2 TimeStamping Services CA

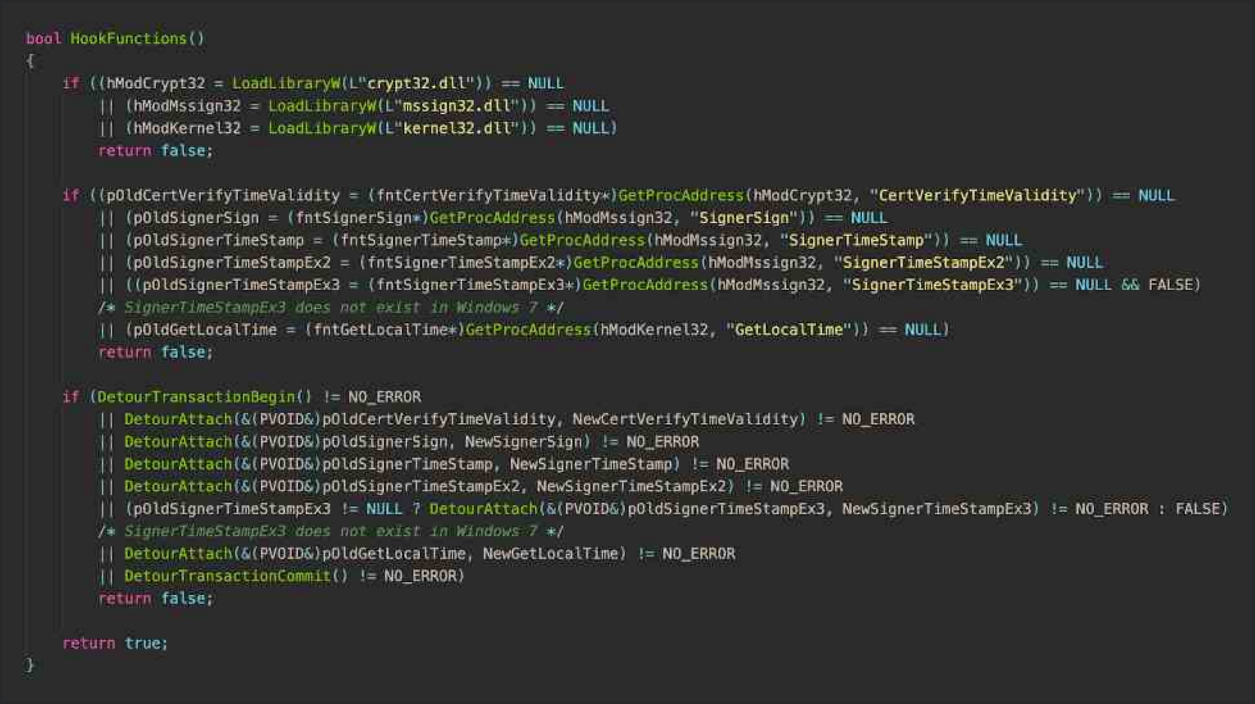
Certificate Revocation Lists

- JemmyLoveJenny EV Root CA
- JemmyLoveJenny SHA2 Extended Validation Server CA
- JemmyLoveJenny ECC Extended Validation Server CA
- JemmyLoveJenny SHA2 Secure Server CA
- JemmyLoveJenny ECC Secure Server CA
- JemmyLoveJenny EV Code Signing CA
- JemmyLoveJenny EV Code Signing CA (SHA2)
- JemmyLoveJenny SHA1 TimeStamping Services CA
- JemmyLoveJenny SHA2 TimeStamping Services CA

Certificate Policy (CP) and Certification Practice Statements (CPS)

*JemmyLoveJenny's personal website containing the required certificates*

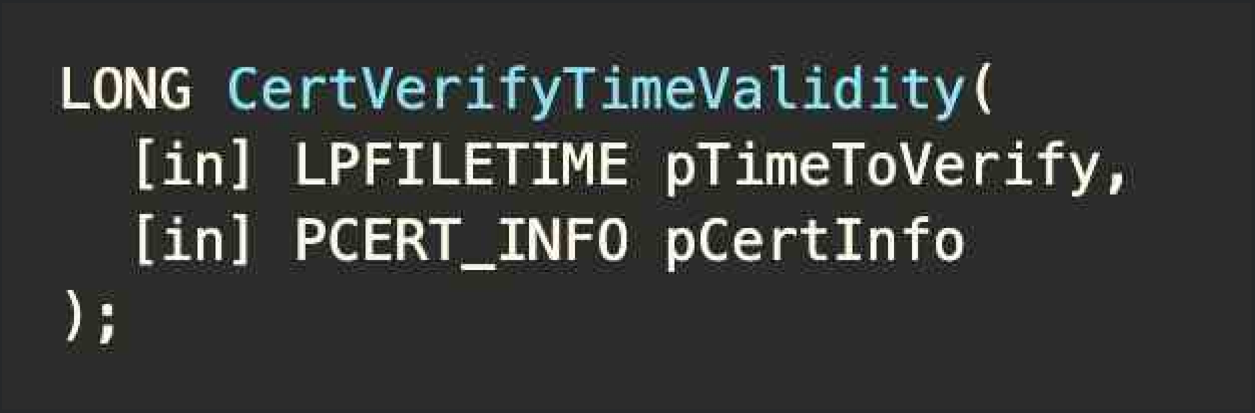
HookSignTool calls the Detours function "DetourAttach" with two parameters: a pointer to the Windows API function to attach to, and a pointer to the "detour" function. The implementation within HookSignTool is shown below:



```
bool HookFunctions()
{
    if ((hModCrypt32 = LoadLibraryW(L"crypt32.dll")) == NULL
        || (hModMssign32 = LoadLibraryW(L"mssign32.dll")) == NULL
        || (hModKernel32 = LoadLibraryW(L"kernel32.dll")) == NULL)
        return false;

    if ((pOldCertVerifyTimeValidity = (fntCertVerifyTimeValidity*)GetProcAddress(hModCrypt32, "CertVerifyTimeValidity")) == NULL
        || (pOldSignerSign = (fntSignerSign*)GetProcAddress(hModMssign32, "SignerSign")) == NULL
        || (pOldSignerTimeStamp = (fntSignerTimeStamp*)GetProcAddress(hModMssign32, "SignerTimeStamp")) == NULL
        || (pOldSignerTimeStampEx2 = (fntSignerTimeStampEx2*)GetProcAddress(hModMssign32, "SignerTimeStampEx2")) == NULL
        || ((pOldSignerTimeStampEx3 = (fntSignerTimeStampEx3*)GetProcAddress(hModMssign32, "SignerTimeStampEx3")) == NULL && FALSE)
        /* SignerTimeStampEx3 does not exist in Windows 7 */
        || (pOldGetLocalTime = (fntGetLocalTime*)GetProcAddress(hModKernel32, "GetLocalTime")) == NULL)
        return false;

    if (DetourTransactionBegin() != NO_ERROR
        || DetourAttach(&(PVOID&)pOldCertVerifyTimeValidity, NewCertVerifyTimeValidity) != NO_ERROR
        || DetourAttach(&(PVOID&)pOldSignerSign, NewSignerSign) != NO_ERROR
        || DetourAttach(&(PVOID&)pOldSignerTimeStamp, NewSignerTimeStamp) != NO_ERROR
        || DetourAttach(&(PVOID&)pOldSignerTimeStampEx2, NewSignerTimeStampEx2) != NO_ERROR
        || (pOldSignerTimeStampEx3 != NULL ? DetourAttach(&(PVOID&)pOldSignerTimeStampEx3, NewSignerTimeStampEx3) != NO_ERROR : FALSE)
        /* SignerTimeStampEx3 does not exist in Windows 7 */
        || DetourAttach(&(PVOID&)pOldGetLocalTime, NewGetLocalTime) != NO_ERROR
        || DetourTransactionCommit() != NO_ERROR)
        return false;

    return true;
}
```

*Implementation of Detours in HookSignTool.*

One of the most critical Windows API functions HookSignTool and FuckCertVerifyTimeValidity attaches to, as made clear by the latter's name, is CertVerifyTimeValidity, a function that verifies that the signing date of a given file is valid.



```
LONG CertVerifyTimeValidity(
  [in] LPFILETIME pTimeToVerify,
  [in] PCERT_INFO pCertInfo
);
```

*CertVerifyTimeValidity prototype.*

By attaching to the *CertVerifyTimeValidity* function, HookSignTool performs a "detour" to a custom implementation of *CertVerifyTimeValidity* named *NewCertVerifyTimeValidity*. This allows

HookSignTool to pass a custom time in the “pTimeToVerify” parameter, thereby allowing an invalid time to be verified.

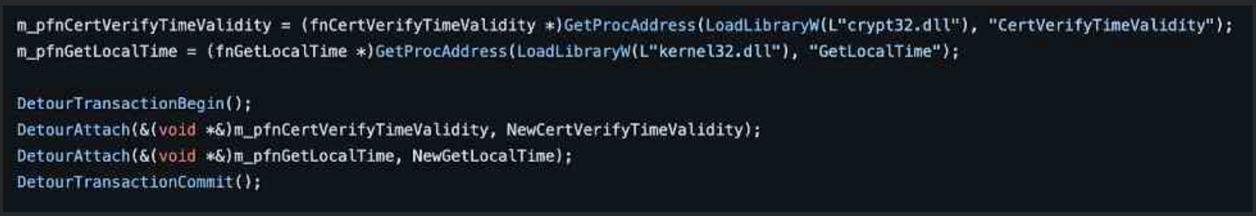
To change the signing timestamp during execution, HookSignTool again uses the DetourAttach function to attach to the Windows API function GetLocalTime and detours to another function named *NewGetLocalTime*. This detour replaces the local time with the date supplied by the user that is within the valid range for the certificate being used. Once both GetLocalTime and CertVerifyTimeValidity are detoured, HookSignTool can supply and verify an illegitimate timestamp for the target binary.

By default, HookSignTool leaves artifacts within the signature it forges, enabling users to identify when it has been used on a given driver. One of these artifacts, “JemmyLoveJenny”, is a reference to the author of HookSignTool. Another artifact left behind is “Fake Timestamp Responder”, a rather conspicuous name that replaces the real timestamp authority name. However, the attacker can manually change the names of these artifacts if they have the skillset to do so, increasing the difficulty of identifying when HookSignTool has been used.

## FuckCertVerifyTimeValidity

Originally released on GitHub on December 13th, 2018, FuckCertVerifyTimeValidity (aka FuckCertVerify) does not have as much functionality as HookSignTool, though provides the same end result of forging a signature timestamp. The tool is likely to have been developed for signing game cheating software, and since its release has been copied and uploaded into different GitHub repositories.

FuckCertVerifyTimeValidity works in a similar fashion to HookSignTool in that it uses the Microsoft Detours package to attach to the “CertVerifyTimeValidity” API call and sets the timestamp to a chosen date. Like HookSignTool, a function must be added to the import table of the legitimate signing tool, but in this case it’s “FuckCertVerifyTimeValidity.dll!test”. Unlike HookSignTool, FuckCertVerifyTimeValidity does not leave artifacts in the binary that it signs, making it very difficult to identify when this tool has been used.



```
m_pfnCertVerifyTimeValidity = (fnCertVerifyTimeValidity *)GetProcAddress(LoadLibraryW(L"crypt32.dll"), "CertVerifyTimeValidity");
m_pfnGetLocalTime = (fnGetLocalTime *)GetProcAddress(LoadLibraryW(L"kernel32.dll"), "GetLocalTime");

DetourTransactionBegin();
DetourAttach(&(void *&)m_pfnCertVerifyTimeValidity, NewCertVerifyTimeValidity);
DetourAttach(&(void *&)m_pfnGetLocalTime, NewGetLocalTime);
DetourTransactionCommit();
```

*FuckCertVerifyTimeValidity attaching to Windows API*

## Stolen and expired certificates frequently used with these tools

To successfully forge a signature, HookSignTool and FuckCertVerifyTimeValidity require a non-revoked code signing certificate that expired or was issued before July 29, 2015, along with the private key and password. During our research, we identified a PFX file hosted on GitHub in a fork of FuckCertVerifyTimeValidity that contained more than a dozen expired code signing certificates frequently used with both tools to forge signatures. Many of these certificates are non-revoked and are widely used to forge signatures on game cheating software as well as malicious drivers. Stolen certificates from the 2015 Hacking Team leaks are included within the file, as well as certificates from a leak posted on a Chinese language software cracking forum. However, it is unclear how these certificates were obtained prior to the leaks. Below is a full list of owner names contained within these certificates:

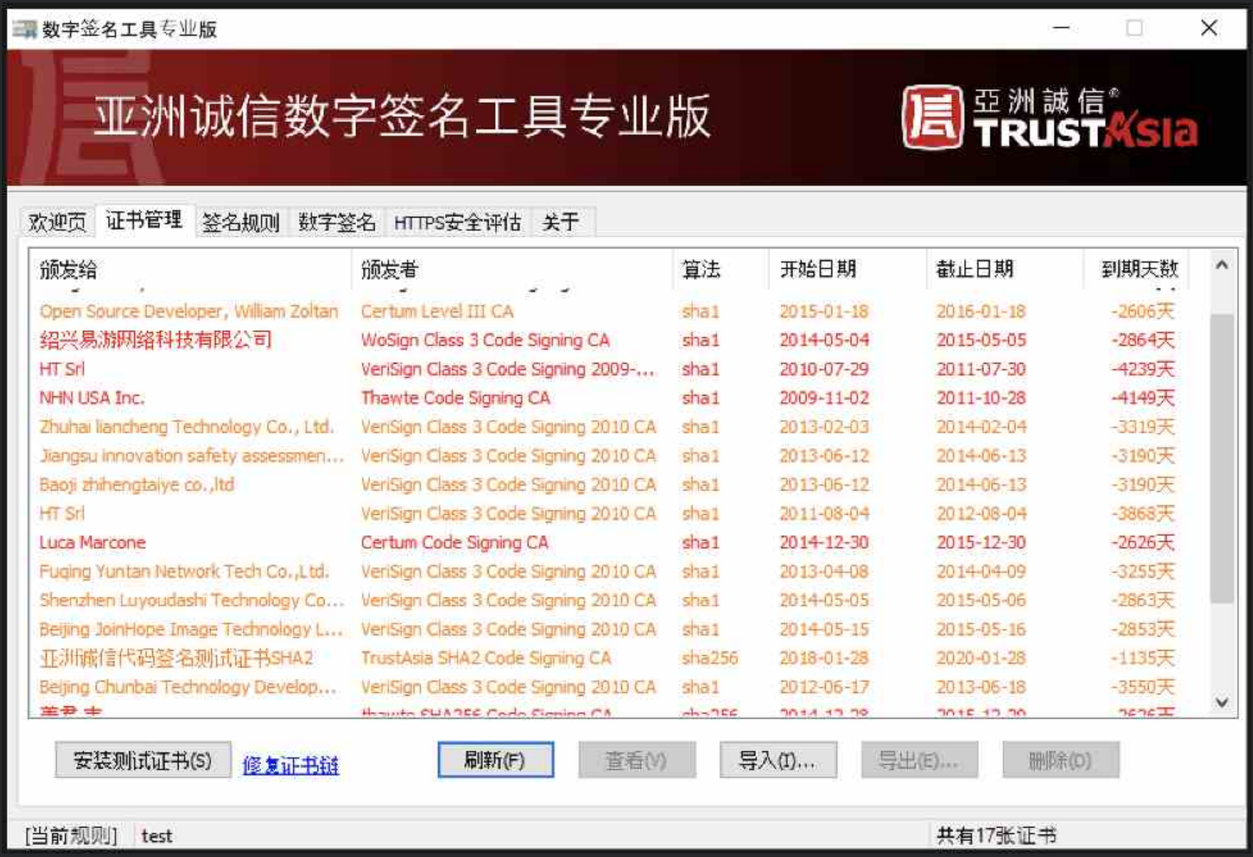
Hacking Team Certificates:

- Open Source Developer, William Zoltan
- Luca Marcone
- HT Srl

Other certificates:

- Beijing JoinHope Image Technology Ltd.
- Shenzhen Luyoudashi Technology Co., Ltd.
- Jiangsu innovation safety assessment Co., Ltd.
- Baoji zhihengtaiye co.,ltd
- Zhuhai liancheng Technology Co., Ltd.
- Fuqing Yuntan Network Tech Co.,Ltd.
- Beijing Chunbai Technology Development Co., Ltd
- 绍兴易游网络科技有限公司
- 善君 韦
- NHN USA Inc.

It is worth noting that any inconsistent capitalization of names in the list above comes directly from the certificates; these errors should not be corrected when used for detection purposes.


数字签名工具专业版

亚洲诚信数字签名工具专业版

欢迎页 | 证书管理 | 签名规则 | 数字签名 | HTTPS安全评估 | 关于

| 颁发给 | 颁发者 | 算法 | 开始日期 | 截止日期 | 到期天数 |
|---|---|---|---|---|---|
| Open Source Developer, William Zoltan | Certum Level III CA | sha1 | 2015-01-18 | 2016-01-18 | -2606天 |
| 绍兴易游网络科技有限公司 | WoSign Class 3 Code Signing CA | sha1 | 2014-05-04 | 2015-05-05 | -2864天 |
| HT Srl | VeriSign Class 3 Code Signing 2009-... | sha1 | 2010-07-29 | 2011-07-30 | -4239天 |
| NHN USA Inc. | Thawte Code Signing CA | sha1 | 2009-11-02 | 2011-10-28 | -4149天 |
| Zhuhai liancheng Technology Co., Ltd. | VeriSign Class 3 Code Signing 2010 CA | sha1 | 2013-02-03 | 2014-02-04 | -3319天 |
| Jiangsu innovation safety assessmen... | VeriSign Class 3 Code Signing 2010 CA | sha1 | 2013-06-12 | 2014-06-13 | -3190天 |
| Baoji zhihengtaiye co.,ltd | VeriSign Class 3 Code Signing 2010 CA | sha1 | 2013-06-12 | 2014-06-13 | -3190天 |
| HT Srl | VeriSign Class 3 Code Signing 2010 CA | sha1 | 2011-08-04 | 2012-08-04 | -3868天 |
| Luca Marcone | Certum Code Signing CA | sha1 | 2014-12-30 | 2015-12-30 | -2626天 |
| Fuqing Yuntan Network Tech Co.,Ltd. | VeriSign Class 3 Code Signing 2010 CA | sha1 | 2013-04-08 | 2014-04-09 | -3255天 |
| Shenzhen Luyoudashi Technology Co... | VeriSign Class 3 Code Signing 2010 CA | sha1 | 2014-05-05 | 2015-05-06 | -2863天 |
| Beijing JoinHope Image Technology L... | VeriSign Class 3 Code Signing 2010 CA | sha1 | 2014-05-15 | 2015-05-16 | -2853天 |
| 亚洲诚信代码签名测试证书SHA2 | TrustAsia SHA2 Code Signing CA | sha256 | 2018-01-28 | 2020-01-28 | -1135天 |
| Beijing Chunbai Technology Develop... | VeriSign Class 3 Code Signing 2010 CA | sha1 | 2012-06-17 | 2013-06-18 | -3550天 |

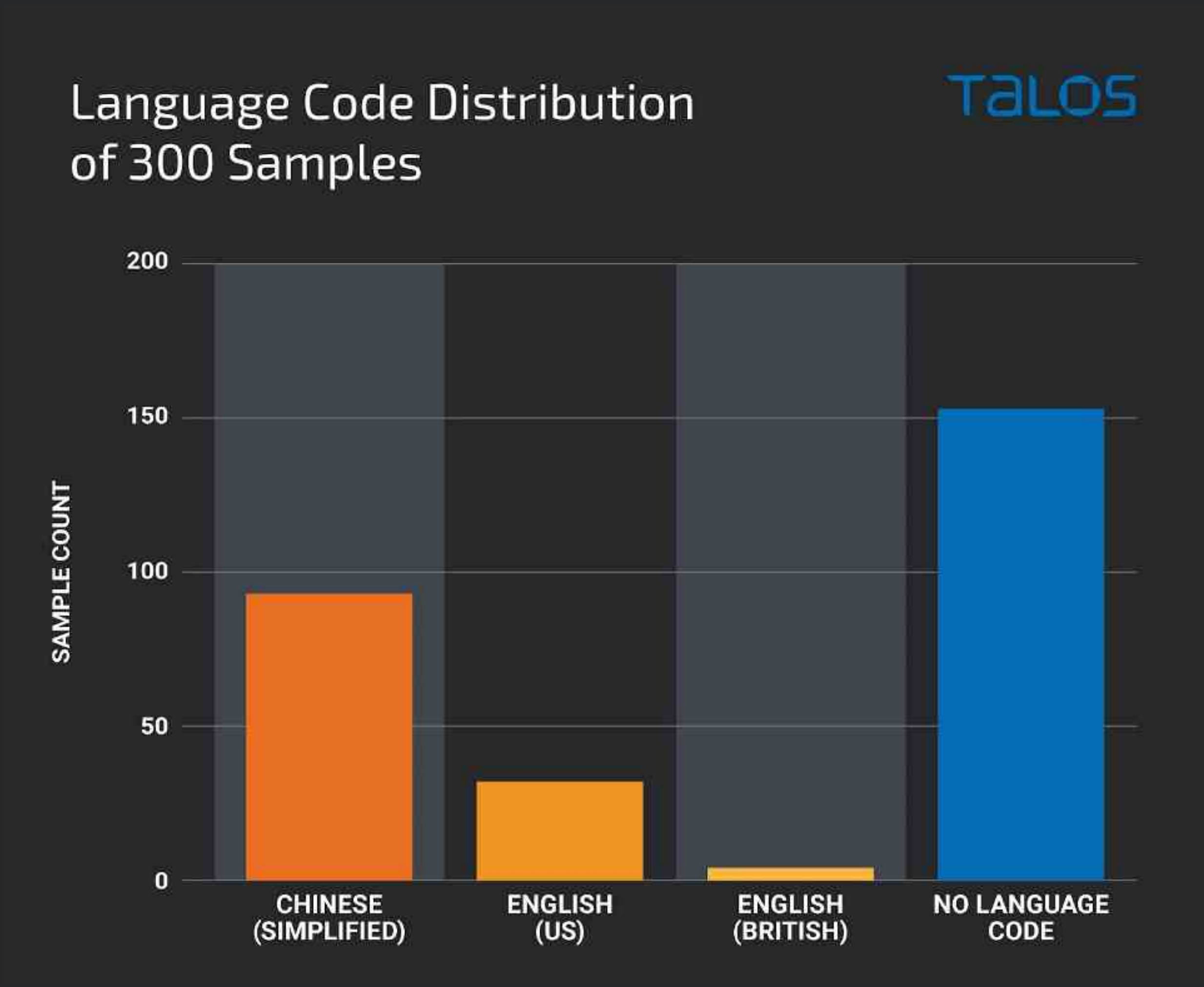
*PFX file imported into "Digital Signature Tool for Asian Integrity"*

We identified another certificate being used in the same manner, although it is not contained within the aforementioned PFX file.

- 北京汇聚四海商贸有限公司 (Beijing Shihai Trading Co Ltd)

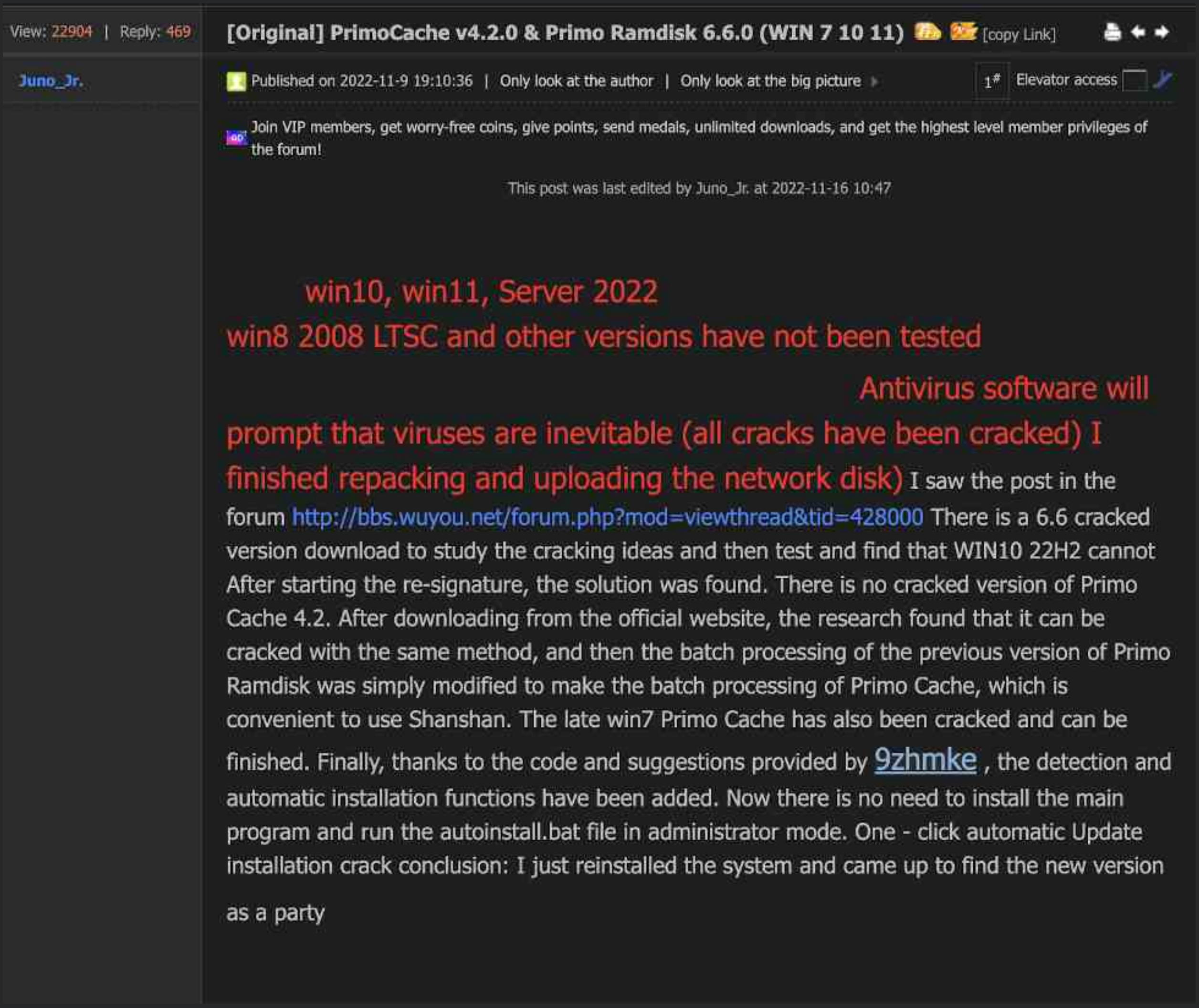
## Actors using these tools predominantly Chinese speakers

Although HookSignTool has been available since 2019, it's popularity and usage appears to be popular with native Chinese speakers. While it is unclear as to why its popularity has not spread further, it is likely that language barriers have played a part. The authors of both HookSignTool and FuckCertVerifyTimeValidity appear to be native Chinese speakers based on the language used in their respective GitHub repositories. In a set of 300 random samples we identified that contained HookSignTool artifacts, 30% contained the "Chinese (Simplified)" language code within the metadata, 10% contained "English (US)" and 1% "English (British)", while 51% contained no language code. Of the samples that *did* contain a language code, "Chinese (Simplified)" accounted for 71% of the sample set.



## Resigned cracked drivers used to bypass DRM

The signing of malicious drivers isn't the only issue that arises from the existence of these tools. During our research, we encountered HookSignTool being used to re-sign drivers after being patched to bypass digital rights management. Specifically, in the Chinese software cracking forum "bbs[.]wuyou[.]net", the user "Juno_Jr." released a cracked version of "PrimoCache" on Nov. 9, 2022.



View: 22904 | Reply: 469

[Original] PrimoCache v4.2.0 & Primo Ramdisk 6.6.0 (WIN 7 10 11) [copy Link]

Juno_Jr.

Published on 2022-11-9 19:10:36 | Only look at the author | Only look at the big picture

1# Elevator access

Join VIP members, get worry-free coins, give points, send medals, unlimited downloads, and get the highest level member privileges of the forum!

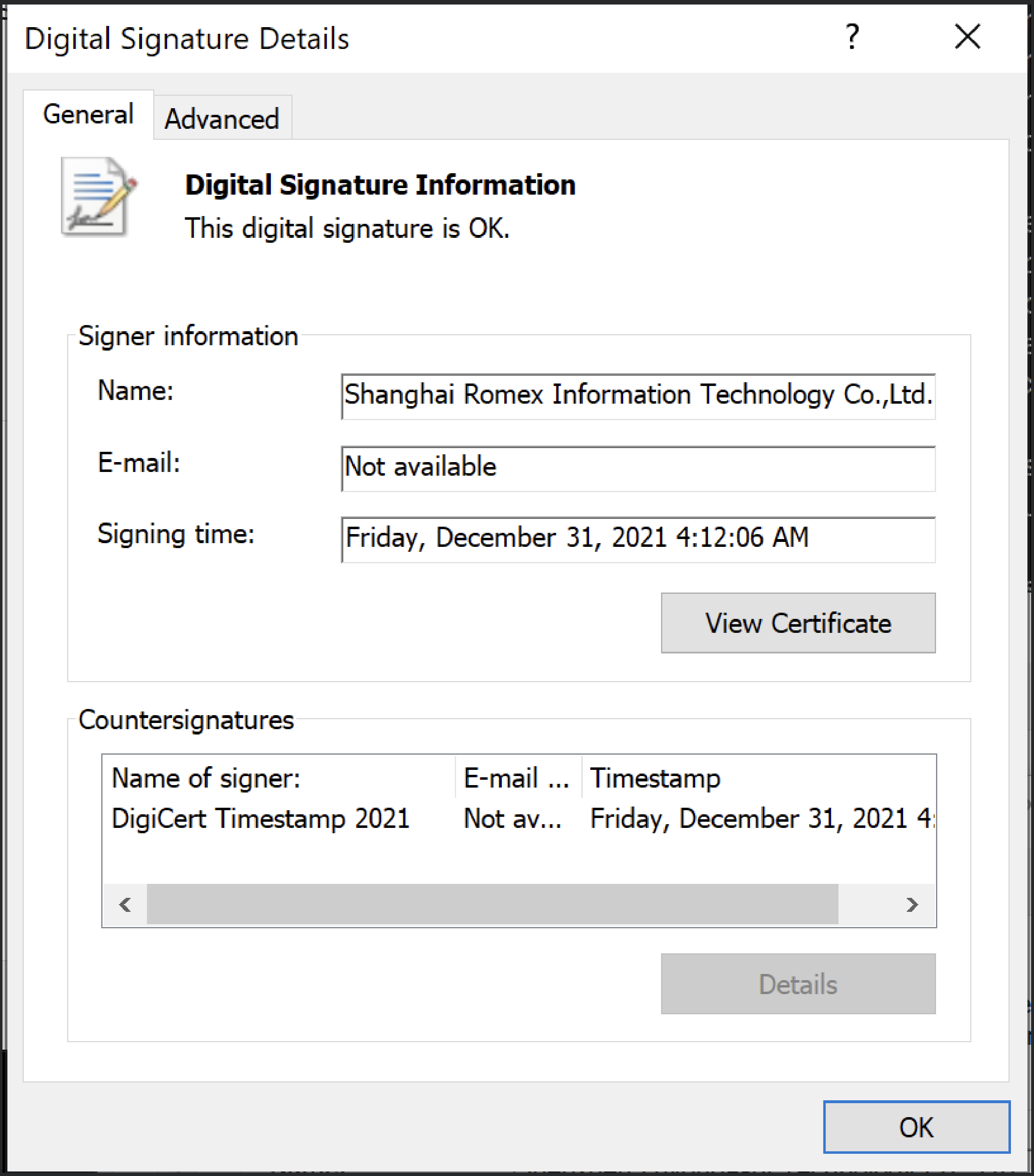
This post was last edited by Juno_Jr. at 2022-11-16 10:47

win10, win11, Server 2022
win8 2008 LTSC and other versions have not been tested

Antivirus software will prompt that viruses are inevitable (all cracks have been cracked) I finished repacking and uploading the network disk) I saw the post in the forum http://bbs.wuyou.net/forum.php?mod=viewthread&tid=428000 There is a 6.6 cracked version download to study the cracking ideas and then test and find that WIN10 22H2 cannot After starting the re-signature, the solution was found. There is no cracked version of Primo Cache 4.2. After downloading from the official website, the research found that it can be cracked with the same method, and then the batch processing of the previous version of Primo Ramdisk was simply modified to make the batch processing of Primo Cache, which is convenient to use Shanshan. The late win7 Primo Cache has also been cracked and can be finished. Finally, thanks to the code and suggestions provided by 9zhmke , the detection and automatic installation functions have been added. Now there is no need to install the main program and run the autoinstall.bat file in administrator mode. One - click automatic Update installation crack conclusion: I just reinstalled the system and came up to find the new version as a party

*Release of a cracked version of PrimoCache (Translated from Chinese)*

To be cracked successfully, the PrimoCache driver needed to be re-signed after patching in order to pass integrity checks within Windows. In the cracked version released on the aforementioned forum post, the patched driver was re-signed with a certificate originally issued to “Shenzhen Luyoudashi Technology Co., Ltd.”, which is contained in the PFX file on GitHub.


Digital Signature Details

General | Advanced

**Digital Signature Information**
This digital signature is OK.

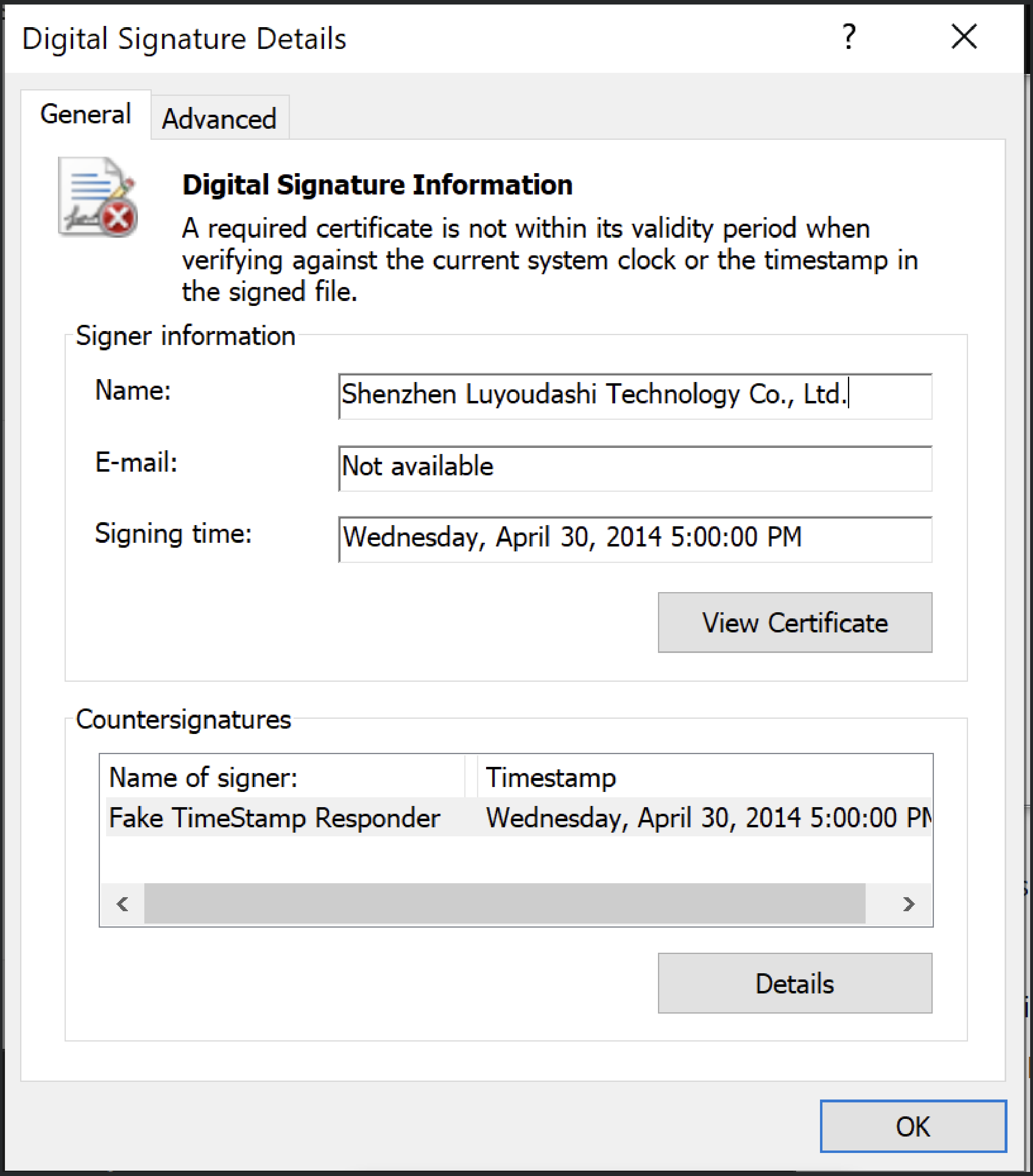
Signer information

| | |
|---|---|
| Name: | Shanghai Romex Information Technology Co.,Ltd. |
| E-mail: | Not available |
| Signing time: | Friday, December 31, 2021 4:12:06 AM |

Countersignatures

| Name of signer: | E-mail ... | Timestamp |
|---|---|---|
| DigiCert Timestamp 2021 | Not av... | Friday, December 31, 2021 4: |

*Legitimate PrimoCache driver signature*


Digital Signature Details

General | Advanced

**Digital Signature Information**

A required certificate is not within its validity period when verifying against the current system clock or the timestamp in the signed file.

Signer information

| | |
|---|---|
| Name: | Shenzhen Luyoudashi Technology Co., Ltd. |
| E-mail: | Not available |
| Signing time: | Wednesday, April 30, 2014 5:00:00 PM |

Countersignatures

| Name of signer: | Timestamp |
|---|---|
| Fake TimeStamp Responder | Wednesday, April 30, 2014 5:00:00 PM |

*Resigned PrimoCache driver signature*

Despite the warning displayed in the digital signature details above, the cracked driver for PrimoCache still functions properly when installed on Windows 10. This ability to resign a cracked driver removes a significant roadblock when attempting to bypass DRM checks in a signed driver. Generally, a signed driver will not execute if it has been tampered with due to integrity checks within the Windows operating system.

## Ramifications, risk and defense

HookSignTool and FuckCertVerifyTimeValidity present a serious threat as the installation of malicious drivers can provide an attacker kernel-level access to a system. These tools can also be used to re-sign a cracked driver to bypass DRM, which may lead to a loss of sales for an organization through software piracy. Aside from these risks, running unverified drivers even if no malicious intent is present can cause damage to a system if the driver is not written properly.

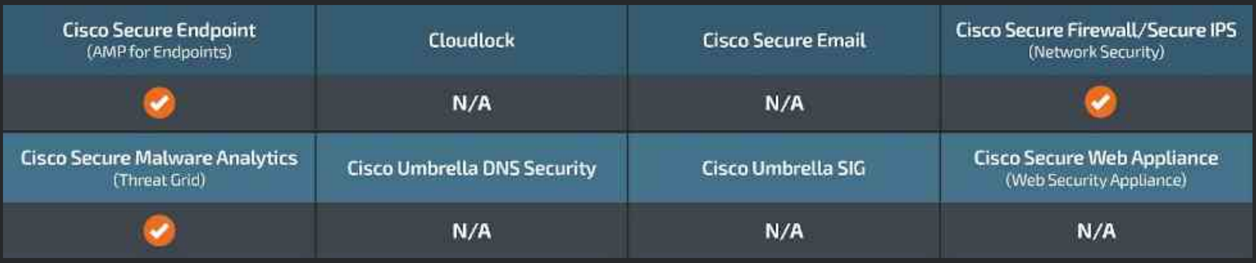
Cisco Talos recommends blocking the certificates mentioned in this blog post, as malicious drivers are difficult to detect heuristically and are most effectively blocked based on file hashes or the certificates used to sign them. Comparing the signature timestamp to the compilation date of a driver can sometimes be an effective means of detecting instances of timestamp forging. However, it is important to note that compilation dates can be altered to match signature timestamps.

Microsoft, in response to our notification, has blocked all certificates discussed in this blog post. Please refer to the advisory published by Microsoft for further information on their response.

Microsoft implements and maintains a driver block list within Windows, although it is focused on vulnerable drivers rather than malicious ones. As such, this block list should not be solely relied upon for blocking rootkits or malicious drivers.

Cisco Talos has created coverage for the certificates discussed in this blog and will continue to monitor this threat activity to inform future protections. Additionally, we will report any future findings regarding this threat to Microsoft.

## Coverage



| Cisco Secure Endpoint (AMP for Endpoints) | Cloudlock | Cisco Secure Email | Cisco Secure Firewall/Secure IPS (Network Security) |
|---|---|---|---|
| ✓ | N/A | N/A | ✓ |
| **Cisco Secure Malware Analytics (Threat Grid)** | **Cisco Umbrella DNS Security** | **Cisco Umbrella SIG** | **Cisco Secure Web Appliance (Web Security Appliance)** |
| ✓ | N/A | N/A | N/A |

Cisco Secure Endpoint (formerly AMP for Endpoints) is ideally suited to prevent the execution of the malware detailed in this post. Try Secure Endpoint for free here.

Cisco Secure Firewall (formerly Next-Generation Firewall and Firepower NGFW) appliances such as Threat Defense Virtual, Adaptive Security Appliance and Meraki MX can detect malicious activity associated with this threat.

Cisco Secure Malware Analytics (Threat Grid) identifies malicious binaries and builds protection into all Cisco Secure products.

Open-source Snort Subscriber Rule Set customers can stay up to date by downloading the latest rule pack available for purchase on Snort.org. Snort SIDs for this threat are 61805-61831 for Snort 2 and 300555-300568 for Snort 3.

ClamAV detections are available for this threat:

- Revoked.CRT.HookSignTool-9999982-1
- Revoked.CRT.HookSignTool-9999983-1
- Revoked.CRT.HookSignTool-9999979-1
- Revoked.CRT.HookSignTool-10000379-0
- Revoked.CRT.HookSignTool-10000385-0
- Revoked.CRT.HookSignTool-10000392-0
- Revoked.CRT.HookSignTool-10000396-0
- Revoked.CRT.HookSignTool-10000400-0
- Revoked.CRT.HookSignTool-10000420-0
- Revoked.CRT.HookSignTool-10000424-0
- Revoked.CRT.HookSignTool-10000428-0

**RELATED CONTENT**

Undocumented driver-based browser hijacker RedDriver targets Chinese speakers and internet cafes