Статья Windows10 - Custom Kernel Signers

xss.is/threads/48595

1. Что такое Custom Kernel Signers?

Мы знаем, что в Windows10 есть строгие требования к драйверу режима ядра. Одним из требований является то, что драйверы должны быть подписаны EV сертификатом, которому доверяет Microsoft. Более того, начиная с выпуска 1607, новые драйверы надо загружать на Windows Hardware Portal для получения подписи от Microsoft. Даже если драйвер использует самоподписанный сертификат, без включения режима TestSigning, Windows10 все равно отказывается его загружать, даже если сертификат был установлен в Windows Certificate Store (`certlm.msc` или `certmgr.msc`). Это означает, что в Windows10 есть независимое хранилище сертификатов для драйвера режима ядра.

Custom Kernel Signers(CKS) - это политика, поддерживаемая Windows10 (возможно, начиная с выпуска 1703). Полное название политики - `CodeIntegrity-AllowConfigurablePolicy-CustomKernelSigners` . Она позволяет пользователям решать, каким сертификатам в ядре доверять, а каким нет. Кстати, эта политика может потребовать включения другой политики - `CodeIntegrity-AllowConfigurablePolicy` .

Обычно **CKS** отключается по умолчанию во всех редакциях Windows10, кроме **Windows10 China Government Edition**.

Если ПК с Windows10 удовлетворяет следующим условиям:

1. Включена политика CodeIntegrity-AllowConfigurablePolicy-CustomKernelSigners. (Возможно CodeIntegrity-AllowConfigurablePolicy также необходима).
2. Включена функция SecureBoot.

если у пользователя есть UEFI Platform Key, то можно добавить сертификат в хранилище сертификатов ядра, для запуска любого драйвера, подписанного сертификатом на этом ПК.

Если вы заинтересованы в других политиках, то можете поискать тут.

2. Как включить данную опцию?

2.1 Требования

1. У вас должны быть права администратора.

2. Вам временно необходима Windows10 версии Enterprise или Education. Почему? Потому что вам надо выполнить команду `ConvertFrom-CIPolicy` в Powershell, которую нельзя выполнить в других версиях Windows 10.
3. Вы можете устанавливать UEFI Platform Key.

2.2 Создание сертификатов и установка Platform Key(PK)

Пожалуйста, проследуйте этой инструкции, чтобы создать сертификаты. После этого вы получите следующие файлы:
Code:

```
// самоподписанный корневой сертификат удостоверяющего центра
localhost-root-ca.der
localhost-root-ca.pfx

// сертификат уровня ядра, выпущенный самоподписанным корневым сертификатом удостоверяющего
центра
localhost-km.der
localhost-km.pfx

// сертификат UEFI Platform Key, выпущенный самоподписанным корневым сертификатом
удостоверяющего центра
localhost-pk.der
localhost-pk.pfx
```

Установку Platform Key в UEFI вам придется сделать самим, так как разные UEFI требуют разных действий. Здесь я вам покажу, как это делать в VMware.

2.2.1 Установка PK в VMware

Если имя вашей виртуальной машины VMware - `TestVM` и у нее есть SecureBoot, то в ее папке будет два файла: `TestVM.nvram` и `TestVM.vmx` . Вы можете установить PK следующим образом:

1. Выключите виртуальную машину.
2. Удалите TestVM.nvram. Это сбросит настройки UEFI виртуальной машины, при следующем запуске.

3. Откройте TestVM.vmx текстовым редактором и добавьте в конец следующие две строки:
   Code:

   ```
   uefi.allowAuthBypass = "TRUE"
   uefi.secureBoot.PKDefault.file0 = "localhost-pk.der"
   ```

   Первая строка позволяет вам управлять ключами SecureBoot в UEFI.
   Вторая строка делает файл `localhost-pk.der` , в папке виртуальной машины, стандартным UEFI PK. Если `localhost-pk.der` находится вне папки виртуальной машины, то укажите полный путь.

Теперь запустите TestVM и ваш PK будет установлен.

2.3 Создание правил сертификата подписи кода уровня ядра

Запустите Powershell от имени администратора в Windows10 версии Enterprise/Education.

1. Выполните New-CIPolicy для создания новой CI (Code Integrity - Целостность Кода) политики. Убедитесь, что ОС не заражена вредоносными программами.
   Code:

   ```
   New-CIPolicy -FilePath SiPolicy.xml -Level RootCertificate -ScanPath
   C:\windows\System32\
   ```

   Данная команда просканирует всю папку System32 и займет некоторое время. Если вы не хотите запускать процесс сканирования, вы можете использовать мой готовый файл SiPolicy.xml.
2. Выполните Add-SignerRule для добавления нашего сертификата подписи кода уровня ядра в SiPolicy.xml.
   Code:

   ```
   Add-SignerRule -FilePath .\SiPolicy.xml -CertificatePath .\localhost-km.der -Kernel
   ```

3. Выполните ConvertFrom-CIPolicy для сериализации SiPolicy.xml и получения бинарного файла SiPolicy.bin
   Code:

   ```
   ConvertFrom-CIPolicy -XmlFilePath .\SiPolicy.xml -BinaryFilePath .\SiPolicy.bin
   ```

На данном этапе, наши правила созданы. Сгенерированный файл можно использовать в любых версиях Windows 10, как только он будет подписан PK сертификатом. С этого момента, нам больше не требуется Windows 10 версии Enterprise/Education.

2.4 Правила политики подписания и применение политик

1. Для подписи SiPolicy.bin, мы должны использовать PK сертификат. Если у вас есть Windows SDK, вы можете подписать его, используя signtool.
   Code:

   ```
   signtool sign /fd sha256 /p7co 1.3.6.1.4.1.311.79.1 /p7 . /f .\localhost-pk.pfx /p
   <password of localhost-pk.pfx> SiPolicy.bin
   ```

   Пожалуйста, пропишите пароль от вашего localhost-pk.pfx в <password of localhost-pk.pfx>.
   После этого вы получите файл SiPolicy.bin.p7 в текущей директории.
2. Переименуйте SiPolicy.bin.p7 в SiPolicy.p7b и скопируйте SiPolicy.p7b в EFI\Microsoft\Boot\
   Code:

   ```
   # run powershell as administrator
   mv .\SiPolicy.bin.p7 .\SiPolicy.p7b
   mountvol x: /s
   cp .\SiPolicy.p7b X:\EFI\Microsoft\Boot\
   ```

2.5 Включение CustomKernelSigners

Переменная, отвечающая за состояние **CKS**, хранится в значении ProductPolicy, чей ключ - `HKLM\SYSTEM\CurrentControlSet\Control\ProductOptions` .

Хотя администраторы могут изменять это значение, оно будет сброшено сразу же после изменения. Это происходит потому, что это значение является просто отображением переменной в ядре после того, как ядро инициализировано. Единственный способ изменить переменную - это вызвать ExUpdateLicenseData. Однако, этот API может быть вызван только в режиме ядра или косвенно путем вызова NtQuerySystemInformation, с помощью SystemPolicyInformation. К сожалению, последний способ успешно работает только тогда, когда вызов делает защищенный процесс.

Таким образом, мы можем модифицировать значение только тогда, когда ядро не завершило инициализацию. Есть ли у нас шанс? Да, в этом нам поможет Windows Setup Mode.

Я написал программу, которая облегчит включение **CKS**. Код находится в папке EnableCustomKernelSigners, а саму программу EnableCKS.exe можно скачать здесь. Конечно, вы можете написать такую программу сами.

Два раза кликните на EnableCKS.exe и вы увидите
Code:

```
[+] Succeeded to open "HKLM\SYSTEM\Setup".
[+] Succeeded to set "CmdLine" value.
[+] Succeeded to set "SetupType" value.

Reboot is required. Are you ready to reboot? [y/N]
```

Введите y для перезагрузки. Система войдет в Setup Mode. EnableCKS.exe запустится автоматически и включит следующие две политики
Code:

```
CodeIntegrity-AllowConfigurablePolicy
CodeIntegrity-AllowConfigurablePolicy-CustomKernelSigners
```

По окончанию работы, система перезагрузится еще раз и вернется в нормальный режим работы.

2.6 Постоянный CustomKernelSigners

Теперь Вы сможете загружать драйвера, подписанные с помощью localhost-km.pfx. Но подождите минутку. В течение 10 минут, **CKS** будет сброшен до выключенного состояния, с помощью sppsvc, за исключением случаев, когда у вас стоит Windows10 China Government Edition. Не волнуйтесь, сброс вступает в силу только при следующем запуске системы.

Поэтому мы должны загрузить драйвер для непрерывного вызова ExUpdateLicenseData, чтобы сохранить **CKS**. Я собрал драйвер с именем cksdrv.sys, который можно скачать [на этой странице]](https://github.com/HyperSine/Windows10-CustomKernelSigners/releases). Код находится в папке CustomKernelSignersPersistent.

cksdrv.sys не подписан. Вы должны подписать его, с помощью localhost-km.pfx, чтобы его можно было загрузить в ядро.
Code:

```
signtool sign /fd sha256 /ac .\localhost-root-ca.der /f .\localhost-km.pfx /p <password of
localhost-km.pfx> /tr http://sha256timestamp.ws.symantec.com/sha256/timestamp cksdrv.sys
```

Пожалуйста пропишите пароль в `<password of localhost-km.pfx>` от вашего `localhost-km.pfx` .

Затем скопируйте cksdrv.sys в c:\windows\system32\drivers и запустите cmd от имени администратора:
Code:

```
sc create ckspdrv binpath=%windir%\system32\drivers\ckspdrv.sys type=kernel start=auto
error=normal
sc start ckspdrv
```

Если все сделали правильно, `ckspdrv.sys` будет успешно добавлен, что также подтверждает, что наша политика применилась правильно.

Теперь вы можете загрузить любой драйвер, подписанный `localhost-km.pfx` . Развлекайтесь и наслаждайтесь~

Оригинал
автор перевода Thatskriptkid