De baas over Secure Boot (1)

Hans Lunsing

Secure Boot heeft ten doel bescherming te bieden tegen malware die zich in het bootproces heeft genesteld voordat het besturingssysteem wordt geladen, in het bijzonder rootkits en bootkits. Het doet dit door cryptografische verificatie van laders van besturingssystemen, drivers en andere software die vanuit de firmware, waarvan UEFI de interface vormt, wordt gestart.

Het Secure Boot-protocol, naar verluidt een initiatief van Microsoft, verscheen in november 2010 voor het eerst in specificatie 2.2 van UEFI, de opvolger van het oude BIOS. Microsoft stelde een werkende Secure Boot als eis voor de certificatie van computers voor zijn besturingssysteem Windows 8, dat in oktober 2012 werd uitgebracht. Sindsdien zijn alle nieuwe computers met Windows 8 en latere versies (8.1 en 10) uitgerust met UEFI met Secure Boot. Het aloude BIOS, daterend van eind jaren 70, zal daardoor geleidelijk verdwijnen.

Volledige zeggenschap

Om pc's voor Windows 8 te kunnen certificeren moeten hardwaremakers ervoor zorgen dat fysiek aanwezige gebruikers Secure Boot kunnen uitschakelen. Dat heeft natuurlijk als nadeel dat de pc dan ook niet meer beschermd is tegen root- en bootkits.

Om de eigenaar van een pc met gebruik van Secure Boot zelf te laten bepalen welke software mag starten en welke niet, stelt Microsoft aan certificatie bovendien de eis dat een fysiek aanwezige gebruiker volledige zeggenschap moet kunnen krijgen over Secure Boot op zijn pc *(take control!)*. Dat alleen een fysiek aanwezige gebruiker dit kan doen is een belangrijk veiligheidsvereiste om te verhinderen dat kwaadaardige software de touwtjes in handen neemt.

Voor Windows 10 geldt niet meer de eis dat Secure Boot moet kunnen worden uitgeschakeld. Het kan dan ook gebeuren dat u een Windows 10 pc koopt waarop dat niet meer mogelijk is. Voor zover ik weet geldt de eis dat een gebruiker de baas moet kunnen worden over Secure Boot echter nog wel.
Vorig jaar heb ik in de nummers 2015-1 en 2015-2 al eens over Secure Boot geschreven, de eerste keer in het kader van UEFI, de tweede keer in het kader van de manier waarop Linux met UEFI en Secure Boot omgaat. Dit keer gaat het om de manier waarop u de baas kunt worden over Secure Boot[1] . Ik veronderstel daarbij dat u een redelijk gevorderde computergebruiker bent, die niet terugschrikt voor wat werk op de opdrachtregel.
We beginnen met een wat uitgebreidere uitleg van Secure Boot dan ik vorig jaar heb gegeven.

Hoe werkt Secure Boot?

Secure Boot werkt met hetzelfde cryptografische systeem als dat wat voor de beveiliging van websites (https) gebruikt wordt: X.509.
X.509 is een PKI- *(Public Key Infrastructure)* standaard die werkt met een sleutelpaar: een openbare sleutel *(public key)* en een bijbehorende privésleutel *(private key)*. De privésleutel wordt geheimgehouden en wordt gebruikt bij de ondertekening van digitale bestanden met de openbare sleutel. De openbare sleutel kan door anderen worden gebruikt ter verificatie van de ondertekening van digitale bestanden, zodat zeker is dat ze van een vertrouwde bron komen. Sleutels worden opgeslagen en verspreid in de vorm van een certificaat.

Verificatie en SB-databases

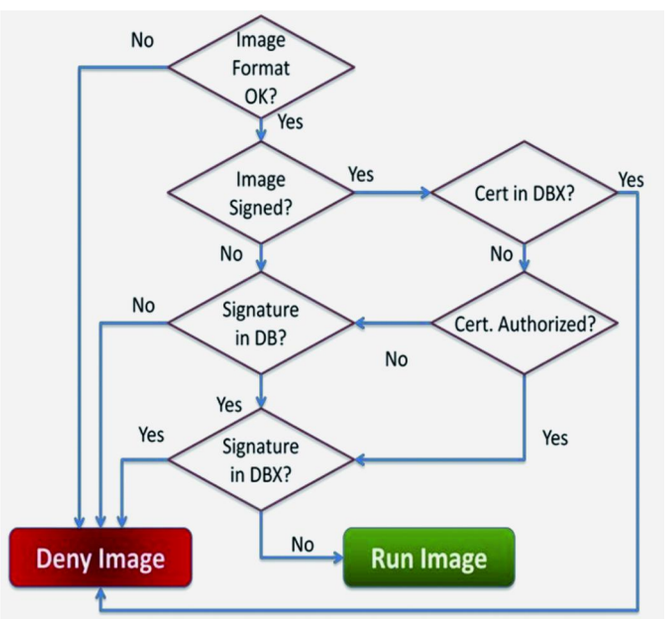
Om van Secure Boot te mogen starten moet een EFI-programmabestand — bijvoorbeeld een opstartlader van een besturingssysteem of een ervoor benodigde driver — zijn ondertekend met een toegelaten sleutelcertificaat *(certificate)*. In plaats daarvan kan een specifiek programma ook worden toegelaten op basis van zijn kenteken *(signature)* in de vorm van zijn cryptografische controlesom *(checksum of hash)*.

Ter verificatie van certificaten en kentekens zijn in het NVRAM *(Non-Volatile Random Access Memory)* van de UEFI-firmware twee EFI-variabelen opgenomen met een lijst van toegelaten en verboden certificaten en kentekens:

- Database van toegelaten certificaten en kentekens *(Authorized Signature Database,* EFI-variabele db).
- Database van verboden certificaten en kentekens *(Forbidden Signature Database,* EFI-variabele dbx).

Verificatie gaat als volgt in zijn werk:

- Als het programma niet is ondertekend, moet zijn kenteken zijn toegelaten (in db), maar niet verboden (in dbx).
- Als een programma wel is ondertekend, gaat het wat ingewikkelder:
  - Noch zijn certificaat, noch zijn kenteken mag verboden (dbx) zijn.
  - Óf zijn certificaat óf zijn kenteken moet (in db) toegelaten zijn. Zie het Verificatieschema hieronder.



Beveiliging van de databases

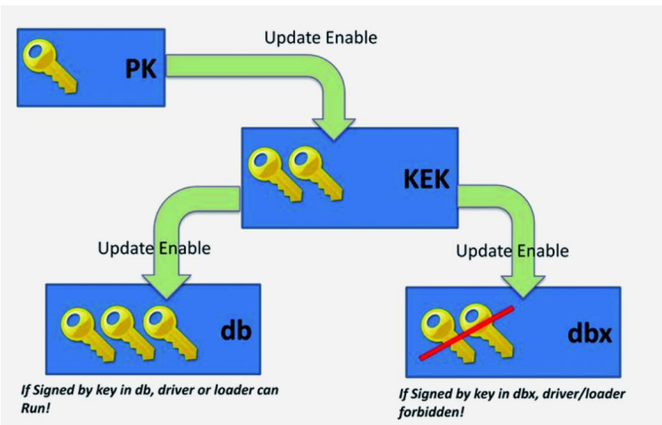
Wanneer Secure Boot actief is mogen aan beide databases (db en dbx) alleen sleutelcertificaten en kentekens worden toegevoegd die ondertekend zijn met een speciale sleutel, een KEK *(Key Exchange Key)*.
De NVRAM bevat ook daarvan een lijst:

- De Key Exchange Key-database (variabele KEK) bevat de openbare sleutels waarmee sleutelcertificaten en kentekens moeten zijn ondertekend om aan de db of de dbx te kunnen worden toegevoegd.

De KEK-database is op zijn beurt ook beveiligd door een speciale sleutel:

- De Platform Key (variabele PK) is de openbare sleutel waarmee sleutelcertificaten moeten zijn ondertekend om aan de KEK-database te kunnen worden toegevoegd.

De relaties tussen de verschillende databases blijken uit deze afbeelding.



Eigenaarsidentificatie

Aan elke sleutel en elk kenteken kan een eigenaarsidentificatie zijn toegekend in de vorm van een GUID (Globally Unique Identifier), maar dit is niet verplicht.
Zo'n GUID bestaat uit 32 hexadecimale cijfers gerangschikt in vijf groepen van resp. 8-4-4-4-12 cijfers. De GUID van Microsoft is 77fa9abd-0359-4d32-bd60-28f4e78f784b.
Als er in het geheel geen GUID is opgegeven wordt als GUID 00000000-0000-0000-0000-000000000000 gebruikt.

Cryptografische algoritmes

Voor de aanmaak van sleutels wordt het cryptografische algoritme RSA met een sleutellengte van 2048 bits aanbevolen, en voor controlesommen het cryptografische hash-algoritme SHA-2 met 256 bits. RSA staat voor Rivest, Shamir, Adleman, de drie ontwerpers van dat algoritme, terwijl SHA staat voor Secure Hash Algorithm.

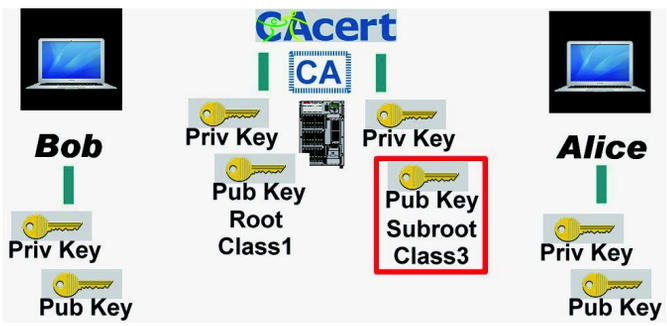
Ondertekening van bestanden

Voor ondertekening van EFI-programmabestanden met een openbare sleutel is de bijbehorende privésleutel nodig. De ondertekening moet voldoen aan de Microsoft Authenticode Specificatie[2]. Daarbij wordt de ondertekening niet los van het programmabestand opgeslagen, maar erin ingebed.
In het X.509-systeem worden sleutelcertificaten gewoonlijk in het PEM- (Privacy-enhanced Electronic Mail) tekstformaat met Base64-codering aangemaakt, maar ze dienen in het binaire DER-formaat (Distinguished Encoding Rules) aan de UEFI-firmware te worden aangeboden.

Certificaat Autoriteit

Iedereen kan een sleutelcertificaat aanmaken, maar dat wil nog niet zeggen dat dit bij beveiliging kan worden gebruikt. Daarvoor moet een certificaat ook nog algemeen worden vertrouwd. Hiertoe zijn Certificaatautoriteiten (CA: Certificate Authority) in het leven geroepen.
Dat zijn algemeen vertrouwde organisaties die bij hen ingediende verzoeken (CSR: Certificate Signing Request) tot het ondertekenen van een certificaat beoordelen en bij goedkeuring een door hen ondertekend certificaat afgeven.

Zo'n certificaat bevestigt het eigendom van de openbare sleutel in kwestie. Er zijn commerciële CA's, die tegen betaling hun diensten aanbieden, zoals Comodo, maar ook gratis CA's, zoals CAcert.



Zo werkt het voor de beveiliging van websites, maar bij Secure Boot gaat het anders. Daar ondertekent Microsoft als voorlopig enige CA geen certificaten, maar opstartbestanden van vertrouwde organisaties zoals de belangrijkste Linux-distributeurs, met zijn openbare CA-sleutel. Zo hoeft de hardwareleverancier bij de inrichting van Secure Boot geen rekening te houden met sleutelcertificaten van alle mogelijke vertrouwde bedrijven, maar kan hij volstaan met het sleutelcertificaat van Microsoft als Certificaat Autoriteit.

Toestanden van Secure Boot

Secure Boot kan in één van drie toestanden (modi) zijn:

- Als Secure Boot actief is, is het in Gebruikersmodus (User Mode). Daarbinnen kunnen weer twee modi worden onderscheiden:
  - Standaardmodus (Standard Mode): de toestand waarin het door de hardware-leverancier is geleverd,
  - Opmaatmodus (Custom Mode): de toestand waarin het door de gebruiker naar eigen wens is ingericht.

  In Gebruikersmodus kan de PK alleen worden vervangen als de nieuwe PK is ondertekend met de openbare sleutel van de oude. KEK's kunnen alleen worden toegevoegd of vervangen als de nieuwe KEK's zijn ondertekend met de openbare sleutel van de PK. Toegelaten en verboden certificaten en controlesommen kunnen alleen worden toegevoegd of vervangen als de nieuwe certificaten dan wel controlesommen zijn ondertekend met de openbare sleutel van een KEK.

- In Opmaatmodus geeft de UEFI-firmware een fysiek aanwezige gebruiker de mogelijkheid de PK te wissen. Dan komt Secure Boot in Instellingsmodus (Setup Mode), waarin die gebruiker KEK, db en dbx helemaal naar eigen smaak kan inrichten zonder dat authenticatie nodig is. Hij kan dan bijvoorbeeld zijn eigen certificaat aan zowel KEK als db toevoegen. Wanneer hij vervolgens zijn eigen certificaat als PK instelt, komt Secure Boot weer in Gebruikersmodus, maar nu helemaal onder zijn zeggenschap.

In Standaardmodus heeft de hardwareleverancier (OEM, ofwel Original Equipment Manufacturer) in de NVRAM van systemen met Windows 8 of hoger gewoonlijk de volgende certificaten opgenomen:

- De PK is een certificaat van de hardwareleverancier zelf als platformeigenaar. Zo is een Intel-moederbord uitgerust met een PK van Intel.
- De KEK bevat een 'Microsoft Corporation KEK CA'-sleutel, uitgegeven door de 'Microsoft Corporation Third Party Marketplace Root'. Dit betekent dat alleen Microsoft de lijsten van toegelaten en verboden certificaten en kentekens kan wijzigen.
- De db bevat twee sleutels van Microsoft:
  - De sleutel van Microsoft als leverancier van Windows: 'Microsoft Windows Production PCA', uitgegeven door de 'Microsoft Root Certificate Authority' voor verificatie van Windows, en

- De sleutel van Microsoft als UEFI-Certificaat Autoriteit: ‘Microsoft Corporation UEFI CA’, uitgave van de ‘Microsoft Corporation Third Party Marketplace Root’ voor verificatie van software van derden.

• De dbx kan kentekens van bekende malware bevatten.

Een hardwareleverancier mag er echter voor kiezen om de ‘Microsoft Corporation UEFI CA’ niet in de db op te nemen. Dat heeft tot gevolg dat software van derden, zoals Linux, op zulke systemen standaard niet start onder Secure Boot.

Hoe wordt u Secure Boot de baas?

Zolang Secure Boot actief is kunnen alleen door Microsoft CA ondertekende EFI-programma's worden gestart. Dat betreft niet alleen de opstartlader van Windows, maar ook die van de belangrijkste Linux-distributies als Ubuntu, Linux Mint als afgeleide van Ubuntu, openSUSE en Fedora, en misschien nog wel andere. Er zijn er echter ook heel wat die niet beschikken over een door Microsoft CA ondertekende opstartlader en daarom niet onder Secure Boot kunnen worden gestart. En mogelijk is er meer interessante software die aan dat euvel mank gaat.

Word uw eigen CA

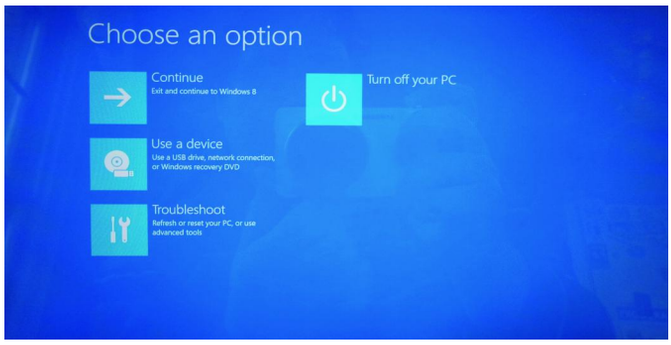
Wilt u toch zulke niet door Microsoft CA ondertekende software gebruiken, dan zit er niets anders op dan uw eigen CA te worden, de software met uw sleutel te ondertekenen (natuurlijk alleen als u die software vertrouwt!) en uw certificaat aan de Secure Boot-databases toe te voegen. Zolang Secure Boot actief is kan dat alleen met een autorisatie van Microsoft, en dat is nu juist niet de bedoeling. We moeten Secure Boot dus deactiveren. Omdat we Secure Boot wel willen blijven gebruiken, doen we dat niet door het helemaal uit te schakelen, maar door de platformsleutel PK te verwijderen.
Zoals we hierboven zagen, kan dat door Secure Boot eerst in Opmaatmodus *(Custom Mode)* te zetten. Na verwijdering van de platformsleutel komt Secure Boot in Instellingsmodus *(Setup Mode)*, waarna u uw eigen sleutel aan KEK en db kunt toevoegen. Tot slot bevestigt u uw eigenaarschap van het platform door uw sleutel als platformsleutel in te stellen. Dan staat Secure Boot weer in Gebruikersmodus *(User Mode)*, maar zal nu ook EFI-programma's laten starten die met uw sleutel ondertekend zijn.

UEFI-interface openen

Er zijn verschillende methoden om in de UEFI-interface te komen. Bij het opstarten van de computer verschijnt de UEFI-interface als u een bepaalde toets indrukt. Welke dat is hangt onder meer af van de fabrikant. Vaak is het F2, maar ook F1, F12, Esc en Del komen nogal eens voor. Raadpleeg daarvoor de handleiding bij uw computer[3].
Het is ook mogelijk om bij het afsluiten van Windows aan te geven dat u bij opnieuw opstarten in de UEFI-interface wilt komen. Dat kan op verschillende manieren, maar de gemakkelijkste is om bij het klikken op ‘Opnieuw opstarten’ van de computer de Shift-toets ingedrukt te houden. Dan verschijnt een scherm met verschillende keuzes waaronder ‘Problemen oplossen’ met ondertitel ‘Pc opnieuw instellen of geavanceerde opties bekijken’. Klik daarop en vervolgens op ‘Geavanceerde opties’. Dan verschijnt een scherm met onder meer ‘Instellingen voor UEFI-firmware’. Klik daarop, waarna u de pc opnieuw moet opstarten om de instellingen van de UEFI-firmware, waaronder Secure Boot, te kunnen wijzigen.
Hoe het instellen van Secure Boot in zijn werk gaat zullen we straks bekijken, maar eerst gaat u uw eigen digitale sleutel maken die u nodig hebt om zeggenschap over Secure Boot te krijgen.

Een eigen sleutel

Om de baas te worden over Secure Boot hebt u een eigen digitale sleutel nodig. Die maakt u met het programma **openssl**[4], dat standaard met Linux-distributies wordt meegeleverd, en ook voor Windows beschikbaar is. U vindt het



eveneens in het WSL (Windows Subsystem for Linux) dat sinds de Anniversary Update in Windows 10 beschikbaar is[5]. De opdracht om een zelf-ondertekend *(self-signed)* sleutelpaar (openbaar en privé) te maken luidt als volgt:

```
openssl req -x509 -newkey rsa:2048 -days 3650
  -nodes -subj '/CN=SelfSign/'
  -keyout SelfSign.key -out SelfSign.crt
```

Er wordt een sleutelpaar aangemaakt volgens cryptosysteem X.509 met cryptografisch algoritme RSA met 2048 bits. Voor dit paar is de naam *‘SelfSign’* (CN betekent Common Name) gekozen.
De privésleutel is opgenomen in het bestand **SelfSign.key**, en de openbare sleutel in het certificaatbestand **SelfSign.crt**. Ook die namen zijn vrij te kiezen, maar het is wel handig om de gegeven extensies aan te houden. Het certificaat is 3650 dagen (10 jaar) geldig, en de privésleutel wordt niet versleuteld *(-nodes)*. Beide sleutels zijn in PEM-tekstformaat. Dat ziet er als volgt uit:

```
-----BEGIN PRIVATE KEY-----
MIIEvgIBADANBgkqhkiG9w0BAQEFAASCBKgwg-
gSkAgEAAoIBAQDiqzpVuGe5+DJO
......
3NUwTraR4pDQEaq8bafW3HyumPCp2Yt+SDWksLJai+2aO
5X6uFCh34n/iuNpVDtOGbf1e3ycvFu9k1ccEpkRzLMw
-----END PRIVATE KEY-----

-----BEGIN CERTIFICATE-----
MIIC+TCCAeGgAwIBAgIJAJ6PUfgpJ0SGMA0GCS-
qGSIb3DQEBCwUAMBMxETAPBgNV
......
H5OJcEIW2QfCZhw+zGaoy6OGUXu+K6srQmHxckUBRS3sw
QdPCL8JoRT1D2x6ML8SCMspB0gkG6ZsSXPsBnDm35NEFN
5ys0yPrs84515Q8+WiDfxmBGo15/d7aKN7
-----END CERTIFICATE-----
```

Op de plaats van ..... staat nog veel meer code. De openbare sleutel moet echter in DER-formaat aan de UEFI-firmware worden aangeboden. Omzetting naar een ander formaat kan ook met **openssl**. Dat gaat zo:

```
openssl x509 -in SelfSign.crt -inform PEM
-out SelfSign.cer -outform DER
```

SelfSign.cer is het certificaat met openbare sleutel in het binaire DER-formaat.
De privésleutel is nodig voor het ondertekenen van berichten of bestanden en moet daarom geheim worden gehouden. Met de openbare sleutel kunnen anderen verifiëren of het bericht of bestand echt van u afkomstig is. In het kader van Secure Boot is het handig om de openbare sleutel, in welke vorm ook, op de ESP *(EFI System Partition)* te bewaren, bijvoorbeeld in een map met de naam ‘Sleutels’.
In Linux heeft alleen de root-gebruiker toegang tot de ESP onder /boot/efi. In Windows heeft alleen de Administrator toegang.

Open daartoe een opdrachtprompt met Administratorrechten en geef daarin de volgende opdracht:

```
mountvol U: /s
```

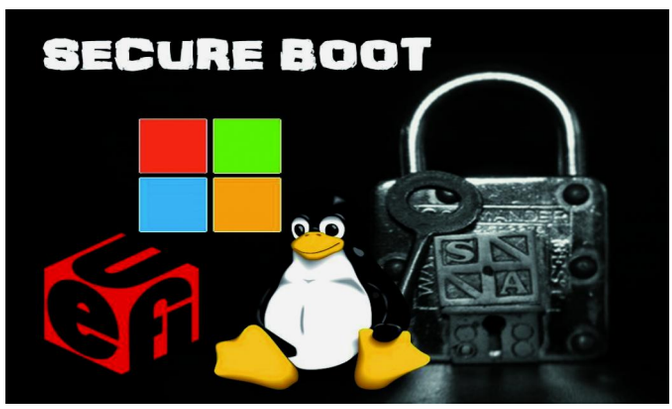
Hiermee wordt de ESP aangekoppeld als schijf U:. Hij kan weer worden ontkoppeld met de opdracht:

```
mountvol U: /d
```

Microsoft heeft zelf ook programma's voor het maken en converteren van sleutelcertificaten. Ze horen bij de Windows SDK *(Software Development Kit)*, die van de site van Microsoft kan worden gedownload[6] en in Windows 10 wordt geïnstalleerd in de map 'C:\Program Files (x86)\Windows Kits\10'. In plaats daarvan is het ook in PowerShell mogelijk met de PKI- *(Public Key Infrastructure)* Cmdlets[7].

Meerdere UEFI-implementaties

De wijze waarop u de platformsleutel verwijdert om Secure Boot in Instellingsmodus te zetten, vervolgens uw eigen sleutelcertificaat toevoegt en daarmee ook het platform overneemt, hangt af van de manier waarop de fabrikant zijn UEFI-interface heeft ingericht. Elke fabrikant doet het op zijn eigen manier. In de wat oudere Intel Visual BIOS (mei 2013) van mijn pc, bijvoorbeeld, kan het alleen maar op tamelijk grove wijze door alle Secure Boot-databases leeg te maken. Bestaande sleutels, in het bijzonder die van Microsoft, worden dan ook verwijderd. Het is wel mogelijk ze terug te zetten, maar dat is dan met inbegrip van de platformsleutel, waardoor de Instellingsmodus wordt verlaten. In zo'n geval is het zaak de bestaande sleutels naar een bestand op schijf te exporteren om ze later weer te kunnen toevoegen. De Aptio Setup Utility van American Megatrends Inc.(AMI) uit 2012, waarmee de laptop van mijn vrouw is uitgerust, doet het gelukkig veel beter.
Het komt zelfs voor dat de mogelijkheid om Secure Boot in de Instellingsmodus te zetten niet eens wordt geboden. Zo heb ik een Toshiba Satellite-laptop gezien waarin Secure Boot alleen maar kan worden uit- of ingeschakeld en verder niets. Waarschijnlijk komt dit weinig voor. De UEFI-interface van computers die voor Windows 8 zijn gecertificeerd moet een fysiek aanwezige gebruiker de mogelijkheid bieden Secure Boot in Instellingsmodus te zetten. Vermoedelijk geldt deze eis ook bij Windows 10.
UEFI-implementaties 'in het wild' hebben nogal eens bugs, waardoor onderdelen niet of niet goed werken. Een voorbeeld is mijn Intel Visual BIOS, waarin Secure Boot wel heel kieskeurig is in wat het toelaat en wat niet. Ook al stond mijn sleutelcertificaat keurig in de lijst van toegelaten certificaten, toch kon van de met mijn sleutel ondertekende programma's alleen de UEFI-shell worden gestart. Nu ja, dit Intel Visual BIOS is al enkele jaren oud, en wordt inmiddels niet meer door Intel ondersteund. Met het verstrijken van de tijd worden UEFI-implementaties gelukkig steeds beter. Er is een open-source virtuele UEFI-firmware met de naam OVMF (Open Virtual Machine Firmware). Deze wordt gemaakt door Tianocore[8], een door Intel in samenwerking met Collabnet opgezette organisatie, en is de door Intel geprefereerde implementatie van UEFI, helaas niet in mijn Intel Visual BIOS. OVMF kan door hardwaremakers als referentie voor hun eigen implementatie van UEFI worden gebruikt. OVMF werkt goed samen met QEMU *(Quick EMUlator)*, een open-source hardware-emulator waarmee virtuele machines kunnen worden gemaakt, te vergelijken met het bekende VirtualBox. QEMU is, net als VirtualBox, beschikbaar voor zowel Linux als Windows en kan in Linux gebruik maken van KVM *(Kernel Virtual Machine)* waarmee het toegang krijgt tot de snelle hardware virtualisatie van de processor. Tussen haakjes: ook de EFI-modus van VirtualBox gebruikt OVMF, maar helaas zonder Secure Boot.



Met QEMU heb ik de Secure Boot-implementatie van OVMF goed kunnen testen. Deze is heel uitgebreid, zelfs zo dat afzonderlijke sleutelcertificaten en controlesommen kunnen worden toegevoegd, dan wel verwijderd. Ook biedt OVMF een UEFI-shell van waaruit EFI-programma's, zoals opstartladers van besturingssystemen, maar ook tools voor onder meer wijziging van de Secure Boot-databases, kunnen worden gedraaid. Die shell is ook afzonderlijk verkrijgbaar en kan bij het opstarten van echte UEFI-hardware worden gebruikt als de UEFI-interface zelf geen shell biedt. Kortom, elke UEFI-implementatie zou zoals OVMF moeten zijn, maar helaas is dat niet het geval.

In de volgende SoftwareBus

Aan de hand van de UEFI-interface van OVMF zal ik de volgende keer laten zien hoe u Secure Boot naar eigen wensen inricht. Na afloop bent u voor Secure Boot de platformeigenaar omdat uw digitale sleutel de platformsleutel is, en staat uw sleutel in de database van toegelaten sleutels. Bedenk daarbij wel dat OVMF een ideale situatie weergeeft. In actuele UEFI-implementaties gaat het ongetwijfeld anders en kunnen er minder mogelijkheden zijn. Zo kan de optie ontbreken om via de UEFI-interface individuele sleutels toe te voegen. Daarvoor is wel een oplossing, waarop ik dan ook zal ingaan.

Voor vragen, commentaar enz. ben ik beschikbaar via https://www.compusers.nl/user/15/contact. Hiervoor moet u wel als lid zijn ingelogd.

Noten
1 Zie http://www.rodsbooks.com/efi-bootloaders/controlling-sb.html
2 Zie http://tinyurl.com/authenticode
3 Zie ook http://pcsupport.about.com/od/fixtheproblem/a/biosaccess_pc.htm
4 https://www.openssl.org/ Windows-versies beschikbaar via https://wiki.openssl.org/index.php/Binaries.
5 Zie voor meer informatie: https://linux.compusers.nl/node/1063
6 Voor Windows 10: https://developer.microsoft.com/en-us/windows/downloads/windows-10-sdk
7 Voor Windows 10: https://technet.microsoft.com/en-us/library/hh848636.aspx
8 Zie http://www.tianocore.org