De baas over Secure Boot deel 2

Hans Lunsing

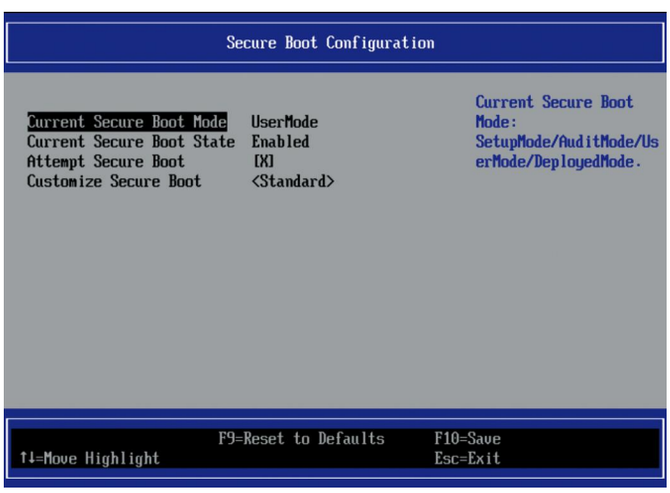
De vorige maal heb ik u laten zien hoe Secure Boot werkt, met welke methode u volledige zeggenschap kunt krijgen over Secure Boot, en hoe u de daarvoor benodigde digitale sleutel kunt aanmaken. Tot slot vertelde ik het een en ander over de verschillende UEFI-interfaces, over de mogelijkheid dat ze niet alles bieden wat we nodig hebben, en ik gaf aan dat wat ze wél bieden op uiteenlopende manieren is ingericht. Er is echter een virtuele UEFI met de naam OVMF, die als referentie dient voor actuele UEFI-implementaties en uitvoerige mogelijkheden biedt om Secure Boot naar eigen wensen in te richten.

Hoe wordt u Secure Boot de baas?

In deze aflevering laat ik aan de hand van de UEFI-interface van OVMF zien hoe u Secure Boot in instellingsmodus zet, uw eigen sleutel toevoegt aan de database met toegelaten sleutels en de platformsleutel overneemt. Voor het geval de UEFI-interface van uw computer niet de mogelijkheid biedt om sleutels toe te voegen aan de Secure Boot-databases, laat ik ook zien hoe het anders kan. Tot slot laat ik zien hoe EFI-programma's in de vorm van opstartladers, EFI-tools, enz. moeten worden ondertekend om ze onder Secure Boot te kunnen gebruiken.

Naar Instellingsmodus in OVMF

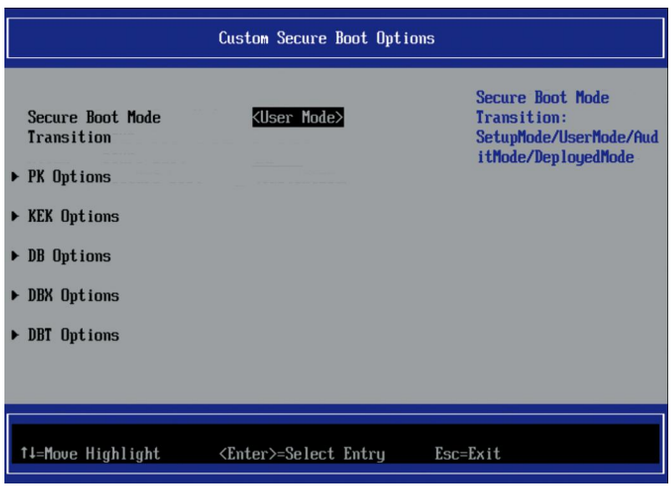
De interface van OVMF verschijnt als we direct na de start van de virtuele machine op F2 drukken. Dan kiezen we voor 'Device Manager' en vervolgens voor 'Secure Boot Configuration', waarna het scherm van afbeelding 1 verschijnt.



Afbeelding 1: Configuratie van Secure Boot in OVMF

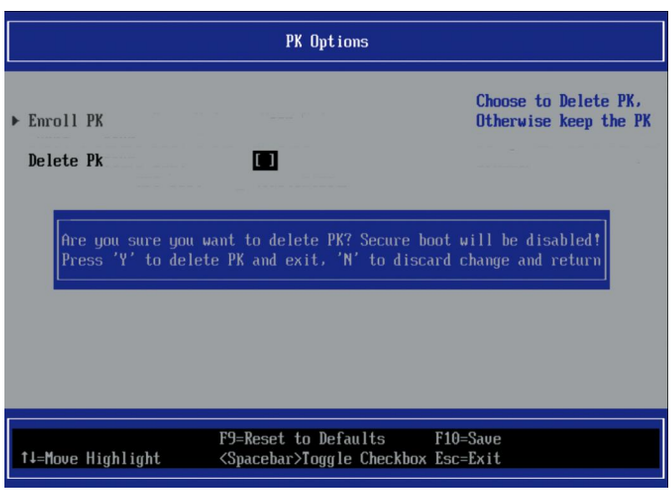
We gaan nu met de cursor naar <Standard>, drukken daar op Enter en kiezen 'Customized'. Onder 'Customize Secure Boot' verschijnt dan een regel 'Custom Secure Boot Options' met ervoor een pijltje. Als we met de cursor op die regel gaan staan en op Enter drukken verschijnt het scherm 'Custom Secure Boot Options' van afbeelding 2.

We zien dat Secure Boot nog in Gebruikersmodus (User Mode) staat, maar in transitie is naar een andere modus. Verder zien we voor elke Secure Boot-database een regel met daarvoor een pijltje. Om veranderingen in zo'n database aan te brengen, moeten we met de cursor op zo'n regel gaan staan en op Enter drukken. Eerst moeten we Secure Boot in



Afbeelding 2: Secure Boot-opties op maat

Instellingsmodus (Setup Mode) zetten door de platformsleutel te verwijderen. Ga daartoe met de cursor naar 'PK Options' en druk op Enter. Dan verschijnt een scherm 'PK Options' met de cursor op het invulvakje achter 'Delete PK'.
Als we dan op Enter drukken verschijnt het scherm van afbeelding 3 met de vraag of we PK echt willen verwijderen omdat dat betekent dat Secure Boot wordt gedeactiveerd.



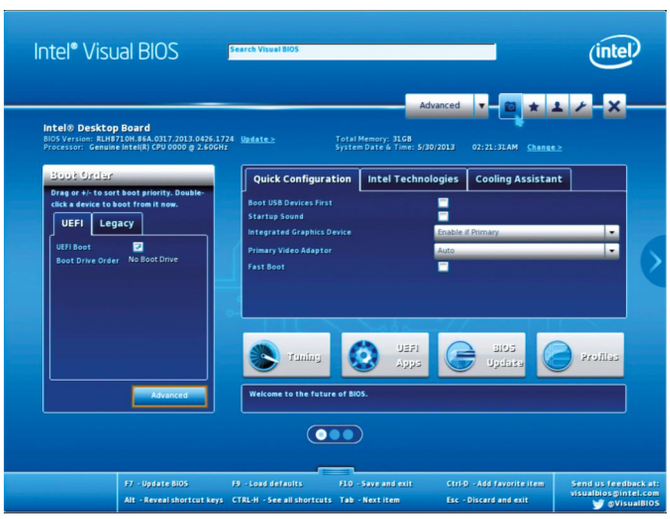
Afbeelding 3: PK-opties met waarschuwing

Wel, dat is precies wat we willen, en we drukken dus op 'Y'. Dan verschijnt de 'Device Manager' met de cursor op 'Secure Boot Configuration'. Op Enter drukken laat weer de 'Secure Boot Configuration' verschijnen, maar dat geeft nu aan dat Secure Boot gedeactiveerd (disabled) is.

Kies weer voor ‘Customized’ en ‘Custom Secure Boot Options’, waar nu staat dat Secure Boot in Instellingsmodus (Setup Mode) is. Nu kunnen we KEK, db en dbx naar eigen goeddunken inrichten. Zoals u ziet is er ook een regel ‘DBT Options’. Dat is een database die voorkomt in nieuwere versies van de UEFI-specificatie (versie 2.4 en hoger). Het doel daarvan is mij (nog) niet bekend.

Toevoegen van eigen sleutel ...

Tot zover is de UEFI-interface nodig om Secure Boot in Instellingsmodus te kunnen zetten. Het is niet gezegd dat hij ook de mogelijkheid biedt sleutels aan de afzonderlijke databases toe te voegen. OVMF, die als referentie dient, doet dat wel, en dat geldt bijvoorbeeld ook voor de Aptio Setup Utility van American Megatrends Inc. (AMI) uit 2012. De Intel Visual BIOS van mei 2013 doet het echter niet. In dat geval is er een alternatief in de vorm van de EFI-tools.



... via de UEFI-interface

Het toevoegen van uw eigen sleutel aan de Secure Boot-databases wijst zich eigenlijk vanzelf. In OVMF begint het op het scherm ‘Custom Secure Boot Options’, waar we net geëindigd zijn (afbeelding 2). Kies daar voor db, en eventueel voor KEK.

Om aan te geven in welk bestand de toe te voegen sleutel staat, verschijnt op zeker ogenblik het scherm ‘File Explorer’, met daarin een lijst van leesbare PCI-apparaten waarop het bestand zou kunnen worden gevonden. Ik ga ervan uit dat u uw eigen sleutelbestand in de map Sleutels op de EFI Systeem Partitie (ESP) hebt gezet. Die ESP is geformatteerd als FAT, die voor UEFI leesbaar is. We zoeken dus naar de ESP, maar de manier waarop de apparaten worden aangeduid is deels nogal cryptisch. Als een bestandssysteemlabel (volume label) ontbreekt ziet het er bijvoorbeeld zo uit:

```
‘NO VOLUME LABEL, [PciRoot(0x0)/Pci(0x7,0x0)/
  HD(1,GPT,244D4BD0- 6B96-481E-AC92-
  DF7344D1A4B9,0x800,0x100000)]
```

Dat is wat moeilijk herkenbaar. Het is daarom handig om vooraf aan de ESP het label ‘ESP’ toe te kennen. Dan wordt hij in dit voorbeeld getoond als:

```
‘ESP, [PciRoot(0x0)/Pci(0x7,0x0)/HD(1,GPT,
  244D4BD0-6B96-481E-AC92-DF7344D1A4B9,0x800
  0x100000)]
```

Meestal is de ESP overigens het eerste apparaat. Zorg dat de cursor erop staat en druk op Enter. U ziet dan de inhoud, die veel gemakkelijker te herkennen is. Als het om de ESP gaat moet er een map EFI zijn. Als die ontbreekt zit u verkeerd en moet u terug door de toets Esc in te drukken.

Bij het toevoegen van uw sleutel kunt u ook een eigenaarsidentificatie opgeven in de vorm van een GUID. U kunt er natuurlijk zelf een verzinnen, maar Linux (ook WSL in Windows 10) heeft standaard een heel gemakkelijke opdracht om een willekeurige GUID te genereren die bijna gegarandeerd nog niet in de wereld voorkomt: uuidgen. Ook Windows kent dat programma, zij het alleen als onderdeel van de Windows SDK, die ik al eerder noemde. Een GUID kan bovendien online worden gegenereerd via de URL https://www.guidgenerator.com/.

... via efitools

Als de UEFI-interface geen mogelijkheid biedt om een sleutel toe te voegen aan de Secure Boot-databases bieden de ‘efitools’ van James Bottomley uitkomst. Het gaat weliswaar om Linux-software, maar ook voor Windows gebruikers is het van belang, omdat het pakket tevens EFI-programma’s biedt die in een UEFI-shell kunnen worden gedraaid.

Ook Linux-gebruikers zullen zich overigens tot die EFI-programma’s moeten wenden, want het Linux-programma waarmee sleutels aan de Secure Boot-databases kunnen worden toegevoegd, weigert vanwege ontbrekende permissies, zelfs als het met rootrechten wordt uitgevoerd.

Ik heb voor installatie in de ESP een UEFI-pakket samengesteld dat bestaat uit:

- De EFI-programma’s van efitools, in de map \EFI\tools.
- Een drietal UEFI-shells in map \EFI\shells: versie 1, 2b en 2. Versie 2 werkt niet in oudere UEFI-implementaties. Versie 1 wel, maar 2b kan ook een alternatief zijn. Dat is een versie 1-shell waarin nieuwe functies van versie 2 zijn opgenomen. Een kopie van de versie 2-shell is opgenomen als \shellx64.efi, zodat de bootmanager ‘rEFInd’ hem als tool in zijn menu opneemt. Voor oudere UEFI-systemen zou dat versie 1 of 2b moeten zijn.
- De uitstekende rEFInd-bootmanager voor UEFI-systemen in map \EFI\refind.
- Een map \Sleutels met daarin de bij rEFInd geleverde openbare sleutels in DER-formaat van Microsoft, enkele Linux-distributies en rEFInd zelf.

Het pakket kunt u downloaden van http://tinyurl.com/UEFIpakket. De inhoud van het pakket kan zo, klaar voor gebruik, naar de ESP worden gekopieerd.

Alle EFI-software van rEFInd, dat zijn de bootmanager zelf met bijbehorende drivers en tools, is met de rEFInd-sleutel ondertekend. Voor gebruik van rEFInd terwijl Secure Boot aan staat, moet die sleutel (refind.cer) in de database met toegelaten sleutels worden opgenomen. De shells en de efitools zijn niet ondertekend. Om ze te kunnen gebruiken terwijl Secure Boot aan staat, moet u ze met uw eigen sleutel ondertekenen en moet u die sleutel in de database met toegelaten sleutels opnemen.

Om de efitools te kunnen gebruiken moet u toegang hebben tot een UEFI-shell. Het is mogelijk dat de UEFI-interface er een biedt. Als dat niet het geval is bent u aangewezen op een van de in het pakket opgenomen externe shells.

Sommige UEFI-interfaces bieden de mogelijkheid een EFI-shell te starten van de ESP. Hij moet dan (voor 64-bitsystemen) 'shellx64.efi' heten en in de rootmap (\) van de ESP staan, zoals ik in het UEFI-pakket heb opgenomen. Als dat ook niet kan, is een bootmanager nodig om tijdens het booten voor zo'n shell te kunnen kiezen. Als u al Linux en dus de bootmanager Grub gebruikt, kunt u het menu aanvullen met toegang tot een UEFI-shell. Neem daartoe in de Grub-configuratie het volgende op:

```
menuentry "UEFI Shell {
  search --no-floppy --set=root --file
  /shellx64.efi chainloader /shellx64.efi
}
```

Ook aan de Windows-bootmanager kan een extra ingang naar de UEFI-shell worden toegevoegd. Dat gaat met het programma 'bcdedit' (als Administrator) , waarin BCD staat voor Boot Configuration Data. Bcdedit is een opdrachtregelprogramma, maar er zijn gratis GUI-programma's beschikbaar die het werken ermee gemakkelijker maken, zoals 'EasyBCD' en 'Visual BCD' .

In plaats van Grub of de Windows-bootmanager kunt u ook rEFInd als bootmanager gebruiken. rEFInd zoekt in alle partities automatisch naar opstartladers en in de ESP naar EFI-programma's, zoals de shell, en toont die in zijn grafische menu. rEFInd is helemaal configureerbaar, maar zijn standaardinstelling voldoet meestal wel. Wel moet het systeem ertoe worden gebracht niet met Grub of Windows te booten, maar met rEFInd. Dat is een kwestie van de bootvolgorde veranderen. Linux heeft daarvoor de beschikking over het programma efibootmgr, terwijl het in Windows met het programma bcdedit kan. Het juiste bcdedit-commando hiervoor is:

```
bcdedit /set {bootmgr} path \EFI\refind\
  refind_x64.efi
```

In plaats van bcdedit kan in dit geval ook het grafische - EasyUEFI worden gebruikt. Bedenk wel dat vooral de oudere UEFI-implementaties nogal eens 'buggy' zijn en een vastgelegde bootvolgorde niet onthouden.

Eenmaal in de shell verschijnt een Mapping table waarin staat hoe de verschillende aangetroffen apparaten in de shell worden genoemd. Apparaten met een bekend bestandssysteem heten FS0, FS1, enz., en andere heten BLK0, BLK1, enz. Hierin staat FS voor File System, en BLK voor Block device. De ESP is in elk geval leesbaar, en meestal de eerste partitie met de aanduiding FS0. Soms is het FS1. Onder de Mapping table verschijnt een prompt, Shell>, met daarachter de cursor. De shell werkt eigenlijk net zo als de opdrachtprompt in Windows, maar de namen van de opdrachten zijn ontleend aan Linux / UNIX. In plaats van de standaard opdrachten zoals ls, cp en rm mogen ook de Windows-equivalenten worden gebruikt, in dit geval dir, copy en del.

Een lijst van alle op-drachten met korte beschrijving krijgt u met het commando help -b, waarin de optie -b er voor zorgt dat de lijst pagina-gewijs wordt gegeven. Meer informatie over een specifieke opdracht krijgt u met het commando help <opdracht> -b, waarin <opdracht> staat voor de naam van de opdracht.

Een voorbeeld is help ls, en voor de Windows-gebruikers onder ons help dir. Met de opdracht set krijgt u een lijst van alle omgevingsvariabelen binnen de shell, zoals het pad, maar ook de UEFI- en shell-versienummers. Net als in Windows is er in de shell geen verschil tussen hoofd- en kleine letters, en is de mappenscheider in paden geen voorwaartse slash (/) zoals in Linux, maar een achterwaartse (\) zoals in Windows. Voorts hoeft de bestandsextensie .efi bij het aanroepen van EFI-programma's niet te worden opgegeven. Hierin, en ook in het FAT-bestandssysteem van de ESP, zien we duidelijk de invloed van Microsoft.

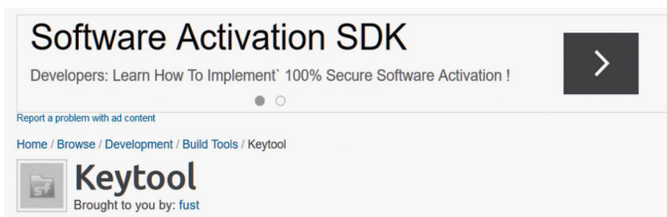
We gaan nu naar de ESP en daarin naar de map \EFI\tools:

```
fs0:
cd efi\tools
```

Welke tools er zijn zien we met de opdracht

```
ls *.efi
```

De tools waarmee we sleutels aan de Secure Boot databases kunnen toevoegen zijn 'KeyTool.efi' en 'UpdateVars.efi'. KeyTool is een menugestuurd programma, zodat dat zich zelf wel wijst (zij het in het Engels). UpdateVars is een optiegestuurd programma. Hoe het moet worden gebruikt staat in het bestand UpdateVars.help. UpdateVars heeft een voordeel ten opzichte van KeyTool: er kan bij het toevoegen van een sleutel ook een eigenaarsidentificatie (GUID) worden opgegeven. UpdateVars heeft wel de eigenaardigheid dat de toe te voegen sleutel in dezelfde map als UpdateVars.efi zelf moet staan. Naast UpdateVars.efi is er 'ReadVars.efi' waarmee de databases kunnen worden uitgelezen en de sleutels in bestanden kunnen worden opgeslagen.



Ook met het ingebouwde shellcommando dmpstore kunnen UEFI-variabelen, zoals de Secure Boot-databases, worden gelezen en geschreven, maar alleen in hun geheel en niet per afzonderlijke sleutel.

Na deze inleiding in de efitools en de shell bent u klaar om uw eigen sleutel aan de database met toegelaten sleutels (db) toe te voegen. Met UpdateVars gaat het zo:

```
cp \Sleutels\SelfSign.cer \efi\tools
  updatevars -g <GUID> -a db SelfSign.cer
rm SelfSign.cer
```

waarin <GUID> staat voor uw GUID, en uw certificaat SelfSign.cer heet.

Overnemen van de platformsleutel

U hebt nu uw eigen sleutel, en naar wens ook die van rEFInd, toegevoegd aan de database van toegelaten sleutels en uw eigen sleutel mogelijk ook aan de KEK-database. Dan is nu de tijd gekomen om uw eigenaarschap van het platform te bevestigen door uw sleutel als platformsleutel in te stellen. Dat gaat op dezelfde manier als het toevoegen van een sleutel aan de KEK- of db-databases, zij het dat er maar één platformsleutel is.

Nadat u dat gedaan hebt is Secure Boot onder uw beheer actief in de Gebruikersmodus. Voortaan mag alleen software worden gestart die is ondertekend met een sleutel die in de database van toegelaten sleutels is opgenomen. Dat zijn de sleutel van Microsoft voor Windows, de sleutel van Microsoft als CA voor derden, en uw eigen sleutel.
Nu is het zaak om gewenste en vertrouwde software met uw sleutel te onder-tekenen, te beginnen met de UEFI-shells en de efitools.

Hoe gaat ondertekenen in zijn werk?
Voor het ondertekenen van EFI-programmabestanden met uw sleutel hebt u de keus uit verschillende programma's. In Linux zijn dat met name de ‘sbsigntools’, maar ook ‘pesign’ en ‘osslsigncode’ zijn beschikbaar. De sbsigntools zijn het gemakkelijkst in gebruik en hebben heel wat functies ter beschikking, reden waarom ik die hier - voor zover nodig - zal behandelen. Een programma Voorbeeld.efi ondertekent u als volgt met uw sleutel SelfSign.crt:

```
sbsign --key SelfSign.key --cert SelfSign.crt
  --output Voorbeeld-signed.efi Voorbeeld.efi
```

SelfSign.key is uw privésleutel en SelfSign.crt uw openbare sleutel in PEM-formaat. Het ondertekende bestand heb ik Voorbeeld-signed.efi genoemd. Om te controleren of het bestand correct ondertekend is gebruikt u sbverify:

```
sbverify --cert SelfSign.crt Voorbeeld
  -signed.efi
```

Programmabestanden kunnen met meer dan één sleutel zijn ondertekend. Let wel dat niet elke UEFI-implementatie daar tegen kan. Als een programma onder Secure Boot niet kan worden gedraaid ondanks het feit dat het met een toegelaten sleutel is ondertekend, kan de oorzaak zijn dat er meer dan één ondertekening is. Met sbverify kunt u een lijst van ondertekeningen krijgen:

```
sbverify --list Voorbeeld-signed.efi
```

Het blijkt dat de --list optie niet in alle versies van sbverify voorkomt. Als hij ontbreekt kan het programma osslsigncode u ook een lijst van ondertekeningen geven:

```
osslsigncode verify Voorbeeld-signed.efi
```

Met het programma sbattach kan een ondertekening worden verwijderd:

```
sbattach --remove Voorbeeld-signed.efi
```

Als er meerdere ondertekeningen zijn, wordt standaard de eerste verwijderd. Een andere kan worden verwijderd door het volgnummer op te geven, bijvoorbeeld:

```
sbattach --remove --signum 2 Voorbeeld
  -signed.efi
```

In Windows 10 zijn de sbsigntools beschikbaar via het Windows Subsysteem voor Linux, maar Microsoft heeft er zelf ook software voor. Zo heeft de Windows Software Development Kit het programma ‘signtool’, en PowerShell het cmdlet ‘Set-AuthenticodeSignature’, een van de Security cmdlets.

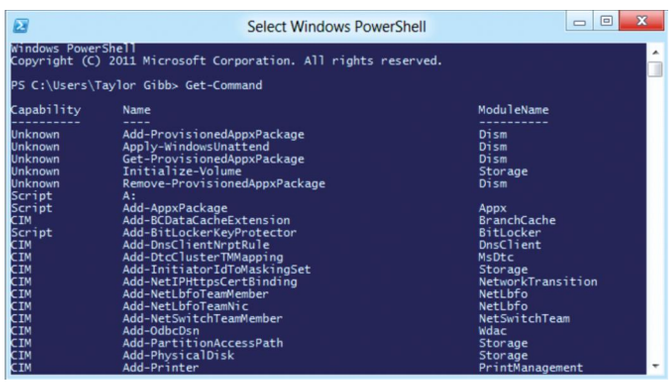
The proof of the pudding is in the eating
Nadat u alle gewenste EFI-programma's hebt ondertekend, kunt u de niet-ondertekende op de ESP vervangen door de door u ondertekende, en dan maar eens zien of het werkt.

Zoals ik al eerder schreef, vertonen met name oudere implementaties van UEFI nogal eens gebreken, zoals een niet goed verlopende authenticatie van ondertekeningen. Als voorbeeld noemde ik mijn Intel Visual BIOS, waarin alleen de door mij ondertekende shell kon worden gestart, maar geen enkele van de efitools. OVMF en de Aptio Setup Utility van AMI doen het overigens prima. Ik ben benieuwd wat uw ervaringen zijn!

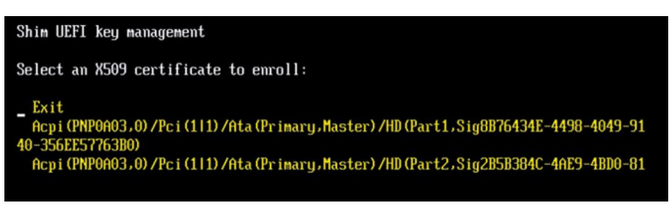
Tot slot

De Linux-efitools en ook sbsigntools hebben nog veel meer mogelijkheden dan ik hier beschreven heb. Zo kunnen sleutelcertificaten ook worden omgezet in een EFI Signature List (.esl) en ondertekend met een hogere sleutel om ze onder Secure Boot te kunnen toevoegen. Ook kunnen kentekens (controlesommen) van specifieke programmabestanden worden toegevoegd aan de database met toegelaten sleutels en kentekens, waardoor die programma's zonder ondertekening toch onder Secure Boot kunnen worden gebruikt. Voor u leuk om verder mee te experimenteren ... Ik hoor graag de resultaten.

Ik ben ook niet ingegaan op het gebruik van de Microsoft-tools voor het maken van en ondertekenen met sleutelcertificaten. De Windows-SDK heeft tools, maar ik heb de indruk dat die inmiddels zijn opgevolgd door PowerShell-cmdlets. Helaas heb ik met geen van beide ervaring. PowerShell is inmiddels open-source en ook beschikbaar voor Linux, en ik ben benieuwd of dit soort dingen dan zowel in Windows als in Linux op dezelfde manier kunnen worden gedaan. De toekomst zal het leren.



Linux-gebruikers hebben nog een andere manier om het Secure Boot-slot te omzeilen. De Linux-bootloader voor Secure Boot met de naam ‘shim’, die met een Microsoft-sleutel voor derden is ondertekend, maakt zelf buiten Secure Boot om gebruik van een database in nvram met de naam MokList (Mok staat voor Machine Owner Key), waarin de gebruiker (de Machine Owner) vertrouwde sleutels kan opnemen. Opstartladers en andere programma's die met zo'n sleutel zijn ondertekend, kunnen dan door shim en soms ook door de bootmanager Grub (zoals bij openSUSE) worden gestart. De Linux-distributies leveren een EFI-programma met de naam MokManager.efi mee om de MokList te kunnen beheren.



Met behulp het programma mokutil is het voor een fysiek aanwezige gebruiker mogelijk om Secure Boot binnen Linux via een getrapte en beveiligde procedure uit te schakelen. Geef daartoe als root het commando:

```
mokutil - disable-validation
```

Hierbij moet een eenmalig te gebruiken wachtwoord worden opgegeven. Na een herstart verschijnt de MokManager waarmee Secure Boot binnen Linux definitief kan worden uitgeschakeld na ingave van het wachtwoord. Na nog een herstart is het mogelijk om niet ondertekende modules, zoals die van VirtualBox en de Nvidia grafische driver, te laden.

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