UEFI Secure Boot

Table of contents

All kernel modules included in MLNX_OFED for RHEL and SLES are signed with x.509 key to support loading the modules when Secure Boot is enabled.

Enrolling NVIDIA's x.509 Public Key On your Systems

In order to support loading MLNX_OFED drivers when an OS supporting Secure Boot boots on a UEFI-based system with Secure Boot enabled, the NVIDIA x.509 public key should be added to the UEFI Secure Boot key database and loaded onto the system key ring by the kernel.
Follow these steps below to add the NVIDIA's x.509 public key to your system:

> **Note**
>
> Prior to adding the NVIDIA's x.509 public key to your system, please make sure:
>
> - The 'mokutil' package is installed on your system
> - The system is booted in UEFI mode

1. Download the x.509 public key.

```
# wget http://www.mellanox.com/downloads/ofed/mlnx_signing_key_pub.der
```

> **Note**
>
> As of version 23.04, the builds for SLES15 sp4 and sp5 are being singed with a newer signing key. The corresponding public key can be downloaded from https://www.mellanox.com/downloads/ofed/nv_nbu_kernel_signing_key_pub.

2. Add the public key to the MOK list using the mokutil utility.
   You will be asked to enter and confirm a password for this MOK enrollment request.

```
# mokutil --import mlnx_signing_key_pub.der
```

3. Reboot the system.

The pending MOK key enrollment request will be noticed by shim.efi and it will launch MokManager.efi to allow you to complete the enrollment from the UEFI console. You will need to enter the password you previously associated with this request and confirm the enrollment. Once done, the public key is added to the MOK list, which is persistent. Once a key is in the MOK list, it will be automatically propagated to the system key ring and subsequent will be booted when the UEFI Secure Boot is enabled.

> **Note**
>
> To see what keys have been added to the system key ring on the current boot, install the 'keyutils' package and run: #keyctl list %:.system_keyring

Removing Signature from Kernel Modules

The signature can be removed from a signed kernel module using the 'strip' utility which is provided by the 'binutils' package.

```
# strip -g my_module.ko
```

The strip utility will change the given file without saving a backup. The operation can be undone only by resigning the kernel module. Hence, we recommend backing up a copy prior to removing the signature.

1. Remove the signature.

```
# rpm -qa | grep -E "kernel-ib|mlnx-ofa_kernel|iser|srp|knem|mlnx-rds|mlnx-nfsrdma|mlnx-nvme|mlnx-rdma-rxe" | xargs rpm -ql | grep "\.ko$" | xargs strip -g
```

Once the signature is removed, a message as the below will no longer be presented upon module loading:

```
"Request for unknown module key 'Mellanox Technologies signing key: 61feb074fc7292f958419386ffdd9d5ca999e403' err -11"
```

However, please note that a message similar to the following will be presented:

```
"my_module: module verification failed: signature and/or required key missing - tainting kernel"
```

This message is presented once, only upon first module boot that either has no signature or whose key is not in the kernel key ring. Therefore, this message may go unnoticed. Once the system is rebooted after unloading and reloading a kernel module, the message will appear (this message cannot be eliminated).

2. Update the initramfs on RHEL systems with the stripped modules.

```
mkinitrd /boot/initramfs-$(uname -r).img $(uname -r) --force
```