Defining a Secure Boot Policy on Dell Servers

This Dell Deployment and Configuration Guide explains how to use the System Setup utility to configure Secure Boot, a BIOS feature which helps protect the pre-boot environment.

Dell Server BIOS Development
September 2014
# Table of contents

Revisions ................................................................................................................................. 2

Executive summary .................................................................................................................. 4

1 Introduction................................................................................................................................ 5
   1.1 What is Secure Boot? ........................................................................................................ 5
   1.2 Why is Secure Boot valuable? .......................................................................................... 5
   1.3 What does this guide explain? .......................................................................................... 5

2 Secure Boot Overview ............................................................................................................. 6
   2.1 Db and Dbx control image execution .............................................................................. 6
   2.2 PK and KEK control policy updates ............................................................................... 7

3 Configuring and Enabling Secure Boot .................................................................................. 9
   3.1 Standard and Custom Policies ....................................................................................... 9
   3.2 Viewing policy components ......................................................................................... 9
   3.3 Using the standard policy .............................................................................................. 11
   3.4 Creating a custom policy .............................................................................................. 12
      3.4.1 Deleting policy objects ......................................................................................... 12
      3.4.2 Importing policy objects ....................................................................................... 13
      3.4.3 Restoring standard policy objects ......................................................................... 14

A Terminology .......................................................................................................................... 15

B Additional Resources ............................................................................................................ 16
Executive summary

As software-based attacks grow more sophisticated, system administrators must employ a wider variety of defenses. Many solutions include protection for the operating system (OS) environment, but neglect the firmware which executes before the OS boots.

Attackers find this pre-boot environment lucrative. Pre-boot malware avoids OS privilege levels, escapes detection by OS anti-malware tools, and even survives re-installation of the OS. If an attacker injects malware into the pre-boot environment, administrators may find it difficult to remove, if they detect it at all.

This guide discusses Secure Boot, a Basic Input/Output System (BIOS) feature that protects the pre-boot environment. Readers will learn how Secure Boot works and how to create a Secure Boot policy in the System Setup utility. By applying the steps described in this guide, system administrators will expand their defenses into the pre-boot realm.
1 Introduction

1.1 What is Secure Boot?
Secure Boot is a system BIOS feature that prevents unauthorized software execution in the pre-boot environment. The system administrator defines the Secure Boot policy, which the BIOS applies to pre-boot software modules. If any module does not satisfy the policy, the system BIOS neither loads nor executes the module.

The Unified Extensible Firmware Interface (UEFI) Forum - an industry body that develops standards for pre-boot software - defines Secure Boot in the UEFI specification. Computer system vendors, expansion card vendors, and operating system providers collaborate on this specification to promote interoperability. As a portion of the UEFI specification, Secure Boot represents an industry-wide standard for security in the pre-boot environment.

1.2 Why is Secure Boot valuable?
Technology solutions without Secure Boot may be vulnerable to firmware rootkits and bootkits. Attackers use firmware rootkits to hide malicious code in device firmware or system firmware, while bootkits infect the software that boots the operating system. In addition to compromising the security of the system, firmware rootkits and bootkits can escape detection in operating system tools and survive re-installation of the operating system. Secure Boot guards against these attacks because it prevents execution of unauthorized pre-boot code.

1.3 What does this guide explain?
This guide provides an overview of Secure Boot and describes how to define a Secure Boot policy on a Dell server. Readers will learn the purpose of each Secure Boot policy component and the steps necessary to enable Secure Boot using the System Setup utility. The intended audience of this guide is system administrators or IT personnel who deploy and configure Dell servers. This guide does not address Secure Boot configuration on client systems such as notebooks or workstations.

Secure Boot relies on asymmetric cryptography techniques. An explanation of these techniques falls outside the scope of this document, but Appendix B lists references with more details. Readers should familiarize themselves with public-key infrastructure (PKI), certificates, digital signatures, and image digests before reading this guide.

Dell provides a separate guide for configuring Secure Boot from Lifecycle Controller. To find this guide, search for "Using Lifecycle Controller to Configure UEFI Secure Boot and OS Deployment" in the Dell TechCenter website (http://www.delltechcenter.com).
2 Secure Boot Overview

Before booting, the system BIOS launches a variety of code modules including device firmware, diagnostics, and operating system loaders. Secure Boot aims to distinguish trusted modules from untrusted modules. Before the system BIOS loads a module into memory, Secure Boot checks whether the module has authorization to execute. If a module does not have authorization, the system BIOS continues without loading or executing the module.

How does Secure Boot distinguish between trusted and untrusted modules? The Secure Boot policy, which the system administrator defines, specifies the rules for authorization. The policy contains keys and certificates for signed modules, and image digests for unsigned modules. Providers of pre-boot code modules sign the code with private keys, and the Secure Boot policy contains public keys needed to verify the digital signatures. Signature verification gives the system BIOS assurance that the module came from a trusted entity, and that other entities have not tampered with the module.

A Secure Boot policy consists of four components: the Platform Key (PK), the Key Exchange Key database (KEK), the Authorized Signature Database (db), and the Forbidden Signature Database (dbx). The system BIOS uses the first two components (PK and KEK) to verify changes to the Secure Boot policy itself. The last two components (db and dbx) help the system BIOS determine whether to execute a pre-boot image.

**Note:** Secure Boot policies contain public keys only; private keys do not reside anywhere in the system. The system BIOS uses public keys to verify signatures, while module providers use private keys to sign modules. Owners of private keys use specialized hardware and techniques for protecting the keys, such as Hardware Security Modules (HSMs), secure Smart Cards, or a Key Management System (KMS). Neither the system BIOS nor Secure Boot require private keys during the boot process.

2.1 Db and Dbx control image execution

First, consider the authorization of pre-boot images. The Authorized Signature Database (db) contains public keys, certificates, and image digests for images authorized to execute. If a pre-boot image includes a digital signature, the BIOS verifies the signature using the keys and certificates in db. If a pre-boot image does not contain a digital signature, the system BIOS determines the digest (also known as a hash value) of the image and compares it against the image digests in db. The BIOS executes the image only if it verifies the digital signature using a key in db, or finds the digest in db.

The Forbidden Signature Database (dbx) specifies images which must not execute even if they are allowed by db. Like db, dbx may contain public keys, certificates, or hash values. The BIOS will not execute an image if it verifies the image’s digital signature with a key in dbx, or finds the image’s hash value in dbx.

In other words, db acts as a “whitelist” and dbx acts as a “blacklist.” To execute an image, Secure Boot must verify that the image is on the “whitelist” and not on the “blacklist.” If Secure Boot does not find the image in either list, the system BIOS will not execute the image. Similarly, if Secure Boot finds the image in both lists, the system BIOS will not execute the image.
Note: The Secure Boot policy applies to all pre-boot code images, including device firmware and operating system boot loaders. When installing expansion cards or operating systems, make sure that db includes information to authorize the images (and dbx does not forbid them). Otherwise, the images will not execute.

2.2 PK and KEK control policy updates

Second, consider changes to the Secure Boot policy itself. Periodically, administrators might add or remove entries in the policy, and attackers might attempt malicious updates to the policy. Anyone wanting to modify db or dbx must sign their modifications with the private PK or KEK. In this way, the BIOS can use the public keys in PK and KEK to verify updates to db and dbx. Thus, if an attacker attempts to modify db or dbx, the signature verification with PK and KEK fails (since the attacker does not possess the private PK or KEK), and the system BIOS does not permit the modifications.

Further, any agent wanting to modify PK or KEK must possess the private half of PK. PK acts as a master key; anyone with access to the private half of PK can modify any portion of the Secure Boot policy. Figure 1 illustrates the relationship between PK, KEK, db, and dbx.

![Diagram of PK, KEK, Db, and Dbx relationships]

Figure 1  Relationship between PK, KEK, Db, and Dbx
The Secure Boot policy contains only one key in PK, but multiple keys may reside in KEK. Normally, either the platform manufacturer or platform owner maintains the private key corresponding to the public PK. Third parties (such as operating system providers and device providers) maintain the private keys corresponding to the public keys in KEK. In this way, platform owners or third parties may add or remove entries in db or dbx.

Notice that the owner of a private KEK possesses similar authority as the owner of a private PK. Like the private PK owner, owners of private KEKs can authorize or prevent module execution by updating db and dbx. The private PK owner possesses slightly more authority; they can modify the contents of KEK or PK.

In summary, the Secure Boot policy uses db and dbx to authorize pre-boot image execution. For an image to execute, it must associate with a key or hash value in db, and not associate with a key or hash value in dbx. Any attempts to update the contents of db or dbx must be signed by a private PK or KEK. Any attempts to update the contents of PK or KEK must be signed by a private PK.
3 Configuring and Enabling Secure Boot

3.1 Standard and Custom Policies
Dell divides Secure Boot policies into two categories: standard policies and custom policies. Dell provides standard policies, while system administrators define custom policies.

Each Dell server ships with one standard policy embedded in the system BIOS. The standard policy includes a Dell-generated PK and one or more keys from operating system vendors in KEK. Dell secures the private PK, while private KEKs are protected by their respective parties. Db contains keys which authorize the firmware on integrated devices, and dbx holds compromised keys or hash values.

Many device vendors sign their firmware through the UEFI certificate authority, and the standard policy’s db contains a certificate that authorizes such firmware. Therefore, expansion card firmware which did not ship with the Dell server still may execute under the standard policy. The section, “Using the standard policy” in this document, explains how to enable Secure Boot with Dell’s standard key set.

Dell anticipates that advanced platform owners will want to create and maintain their own keys. Platform owners who do not want to rely on Dell’s standard policy may replace it with a custom policy. In this case, the owner must create keys or certificates and import them into the system. A custom policy may include none of the standard policy components, or may include both standard and custom components. The section, “Creating a custom policy,” outlines this process.

3.2 Viewing policy components
While configuring Secure Boot, system administrators may want to examine the contents of the Secure Boot policy. Dell offers two methods for viewing policy components.

The first method provides a summary of all components in the system:

1. Power on the system and enter System Setup (press F2 when the system BIOS prompts “F2 = System Setup”).
2. When the System Setup screen appears, select System BIOS > System Security > Secure Boot Policy Summary. The summary page gives basic information about each entry in PK, KEK, db, and dbx. See Figure 2.
The second method offers more details about each component. Using this method, the system administrator can export individual components to a filesystem (such as a removable USB drive or network share). Then, programs on other systems (such as certificate viewers) can display the component contents. To export a component:


**Note:** If Secure Boot Custom Policy Settings is grayed, temporarily change Secure Boot Policy to Custom. The Secure Boot Custom Policy Settings page is enabled, but displays the standard policy components. The custom policy components match the standard policy components until you add or remove components.

2. Select a policy component (PK, KEK, db, or dbx).
3. When exporting KEK, db, or dbx components, select the individual entry to export (such as a certificate or hash value). When exporting PK, move on to step 4 since PK contains only one entry.
4. Click the Export link.
5. Select a filesystem from the list.

**Note:** System Setup detects FAT32 filesystems only. Other types of filesystems will not appear in the list.

6. Browse through the filesystem to find a directory for the export. To indicate that the current directory should hold the export, select “Select Current Directory.”
7. Enter a file name and click “Export.”

### 3.3 Using the standard policy

Dell’s standard policy offers a simple way to guard against firmware rootkits and bootkits. This policy suits administrators who deploy the server with a mainstream operating system and do not change the hardware configuration (that is, they do not add expansion cards after receiving the system from Dell).

Operating systems that support Secure Boot include Microsoft™ Windows™ Server 2012, SUSE™ Linux Enterprise Server 11, and Red Hat™ Enterprise Linux 7. Earlier versions of these operating systems (such as Windows Server 2008) do not support Secure Boot. Enabling Secure Boot may cause the system not to boot to earlier OS versions.

To use the standard policy:

1. Power on the system and enter System Setup (press F2 when the system BIOS prompts “F2 = System Setup”).
2. When the System Setup screen appears, select System BIOS > Boot Settings.
3. Make sure that Boot Mode is set to UEFI. Secure Boot cannot be enabled when Boot Mode is set to BIOS.
4. Press Esc to exit the Boot Settings menu, then select Miscellaneous Settings.
5. Make sure that Load Legacy Video Option ROM is set to Disabled. Secure Boot cannot be enabled when Load Legacy Video Option ROM is set to Enabled.
6. Press Esc to exit Miscellaneous Settings, then select System Security.
7. Set Secure Boot Policy to Standard.
8. Set Secure Boot to Enabled.
9. Make sure a Setup Password is installed. The Setup Password field is in the same menu (System Security). If a Setup Password is not installed, unauthorized users could disable or bypass Secure Boot.
10. Exit System Setup, saving the changes.
11. Reset the system. In this configuration, the system BIOS will apply the Standard Policy objects to each pre-boot module, including device firmware and operating system loaders.

The system BIOS will display an error message if any device firmware or operating system loader fails to satisfy the Secure Boot policy. The Lifecycle Controller log also will contain such error messages. If error messages appear, consider creating a custom policy to suit the server’s specific configuration.
3.4 Creating a custom policy in System Setup

A custom Secure Boot policy incorporates keys, certificates, and hash values not included in the standard policy. The system administrator starts with a copy of the standard policy objects then deletes and imports policy objects as needed.

To create a custom policy:

1. Power on the system and enter System Setup (press F2 when the system BIOS prompts “F2 = System Setup”).
2. When the System Setup screen appears, select System BIOS > Boot Settings.
3. Make sure that Boot Mode is set to UEFI. Secure Boot cannot be enabled when Boot Mode is set to BIOS.
4. Press Esc to exit the Boot Settings menu, then select Miscellaneous Settings.
5. Make sure that Load Legacy Video Option ROM is set to Disabled. Secure Boot cannot be enabled when Load Legacy Video Option ROM is set to Enabled.
6. Press Esc to exit Miscellaneous Settings, then select System Security.
7. Set Secure Boot Policy to Custom.
8. Use the Secure Boot Custom Policy Settings menu to modify the policy.
   a. To remove entries, refer to “Deleting policy objects” below.
   b. To add entries, see “Importing policy objects” below.
   c. To restore standard entries after deleting them, see “Restoring standard policy objects” below.
9. Set Secure Boot to Enabled.
10. Make sure a Setup Password is installed. The Setup Password field is in the same menu (System Security). If a Setup Password is not installed, unauthorized users could disable or bypass Secure Boot.
11. Exit System Setup, saving the changes.
12. Reset the system. In this configuration, the system BIOS will apply the Standard Policy objects to each pre-boot module, including device firmware and operating system loaders.

3.4.1 Deleting policy objects in System Setup

The Secure Boot Custom Policy Settings menu in System Setup provides mechanisms for deleting any object in the Secure Boot policy. To delete all keys, certificates, and hash values in the policy, select “Delete All Policy Entries” from the main page of this menu. To delete individual objects, enter the appropriate submenu and select the “Delete” option.

For example, to delete an entry in KEK:
1. From the Secure Boot Custom Policy Settings menu in System Setup, enter the Key Exchange Key Database (KEK) submenu.
2. Select the KEK entry to delete.
3. Select “Delete KEK Entry.”
4. Choose “Yes” at the confirmation prompt.

### 3.4.2 Importing policy objects in System Setup

Administrators can add objects to the custom policy by importing files from available filesystems.

To import a policy object:

1. From the Secure Boot Custom Policy Settings menu, select the policy component (PK, KEK, db, or dbx) that will hold the imported object.
2. Select the Import option.

**Note:** When importing PK, the Import option is available only when a Platform Key is not installed. To import a new Platform Key, first delete the existing Platform Key.

3. Choose the filesystem that contains the file to import.

**Note:** System Setup detects FAT32 filesystems only. Other types of filesystems will not appear in the list.

4. Browse the directories in the filesystem and select the file to import. Note that each policy component accepts specific file formats, as shown in Table 1 below. To add an image digest to db or dbx, select an EFI image. System Setup calculates the digest (hash) of the selected image.

5. [Optional] Enter a Globally Unique Identifier (GUID) for the entry. Software uses the GUID to associate an entry with an owner. Unless you are aware of software that needs to identify this entry with its owner, leave this GUID as all zeroes (00000000-0000-0000-0000-000000000000).

6. Select the Import link to complete the import operation.

7. If the import does not complete successfully, make sure that the file matches an acceptable file format and file extension for the policy component. See Table 1.

<table>
<thead>
<tr>
<th>Policy Component</th>
<th>Acceptable File Formats</th>
<th>Acceptable File Extensions</th>
</tr>
</thead>
</table>
| PK               | • X.509 Certificate (binary DER format only) | • .cer  
|                  |                         | • .der  
|                  |                         | • .crt  |
| KEK              | • X.509 Certificate (binary DER format only) | • .cer  
|                  | • Public Key Store      | • .der  
|                  |                         | • .crt  
|                  |                         | • .pbk  |
### 3.4.3 Restoring standard policy objects in System Setup

Creators of custom policies may decide to include some of the standard policy objects after deleting them. For example, an administrator may replace all standard db certificates with custom certificates, and then realize later that an integrated device requires a certificate from the standard policy.

To restore a single policy component:

1. From the Secure Boot Custom Policy Settings menu, select the policy component (PK, KEK, db, or dbx) to restore.
2. Select the Restore link, then choose Yes at the confirmation prompt.

Restoration will delete all imported entries in the component; the restored component matches the standard policy component. The administrator must re-import any custom entries after performing a restoration.
**Terminology**

**Administrator**: Computer user with elevated privileges. Administrators possess control over aspects such as system configuration, software installation, server management, and security mechanisms.

**Basic Input/Output System (BIOS)**: Firmware that initializes system components (such as memory, processors, and input/output devices), launches device firmware (such as storage controllers, network controllers, and video controllers), and boots operating systems. Think of the BIOS as a small operating system for the pre-boot environment.

**Firmware**: Software that interacts directly with hardware components. In computer systems, firmware often acts as an interface between hardware and higher-level software.

**Malware**: Software or firmware with malicious intent. Attackers use malware to intercept sensitive data, destroy system components, or otherwise inconvenience users.

**Operating System (OS)**: Software that provides services for user applications. Examples of operating system services include memory management, user interface primitives, access to input/output devices, and filesystem management.

**Pre-boot Environment**: The combination of hardware resources and firmware services available to software that executes before the operating system launches. The timeframe of this environment begins at system power-on and ends when the BIOS passes control to the operating system.

**Unified Extensible Firmware Interface**: A set of industry-standard software interfaces provided by the system BIOS. Pre-boot software uses these interfaces to access hardware components (such as timers), system memory (volatile and non-volatile), input/output devices (such as video, network, storage, and serial communications), and filesystems. The system BIOS also provides a subset of these interfaces to operating systems.
B Additional Resources

Find documentation for Dell products at dell.com/support.

To view an alternate method for configuring Secure Boot on Dell PowerEdge servers, visit http://www.delltechcenter.com (search for “Using Lifecycle Controller to Configure UEFI Secure Boot and OS Deployment”)

References that provide a high-level overview of asymmetric cryptography:

- Wikipedia article: http://en.wikipedia.org/wiki/Asymmetric_cryptography

References that address asymmetric cryptography in detail:


References that discuss Secure Boot and pre-boot attacks in detail:

- Unified Extensible Firmware Interface Specification, version 2.4, sections 27.2 through 27.6: http://www.uefi.org/specifications