presented by

UEFI Security Enhancements

UEFI Fall Plugfest – October 24-27, 2011

Presented by Kevin Davis (Insyde Software)

Agenda

- Introduction
- Authenticated Variables
- Driver Signing
- System Defined Variables
- Secure Boot value
- Demo
- Questions

Introduction: Why UEFI Secure Boot?

- Current OSs have improved virus resistance
- Microsoft Windows 8 improves even more
- Legacy BIOS has become the latest malware target

> *Called Mebromi, the malware is reminiscent of the IceLord proof of concept BIOS rootkit in 2007, was a late 1990s virus that was able to erase the motherboard software. This new rootkit is a different caliber as it is appears to be one of the most persistent malware programs we have heard so far. – Tom's Hardware (Sept 15, 2011)*
>
> *Are BIOS rootkits a real threat? Yes, we can consider Mebromi the first real BIOS rootkit incident discovered in the wild – let's consider IceLord BIOS rootkit more a proof of concept. -- webroot threat blog (Sept. 13, 2011)*

- UEFI 2.3.1 Secure Boot
  - Software identity checking at every step of boot – Platform Firmware, Option Cards, and OS Bootloader

- Secure Boot is a Windows 8 requirement!

UEFI 2.3.1 Specification Update

Security
- Authenticated Variable Update Changes
- Key Management Service (KMS)
- Storage Security Command Protocol for encrypted HDD

Network
- Netboot6 client use DUID-UUID to report platform identifier

Interoperability
- New FC and SAS Device Path
- FAT32 data region alignment
- HII clarification & update
- HII Modal Form

Performance
- Non-blocking interface for BLOCK oriented devices

Technology
- USB 3.0

Maintenance
- User Identifier, etc.

UEFI 2.3.1 Enabling More Security Support

Secure Boot compared to Measured Boot

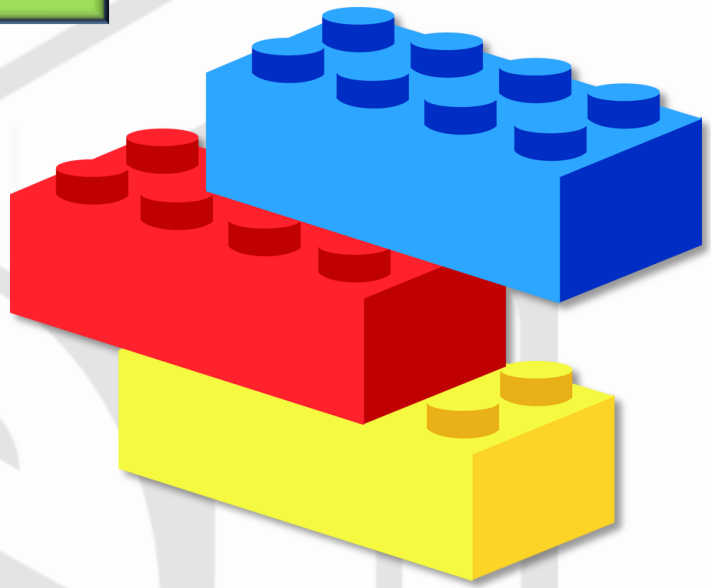
	Secure Boot	Measured Boot (TCG - TPM)
Security Function	Help BIOS verify OS is OK	Help OS verify BIOS and OS Boot paths are unchanged
Scan boot path and hash all BIOS code	No	Yes reduce BIOS attack surface
Check OS boot loader for unauthorized replacement or modification?	Yes reduce OS boot attack surface	Yes
Easy for End User to update system	Yes New version must be signed by someone in KEK or db	No Measured boot must be manually turned off to update
TPM Required?	No	Yes store measurements in TPM PCRs

UEFI Secure Boot Overview

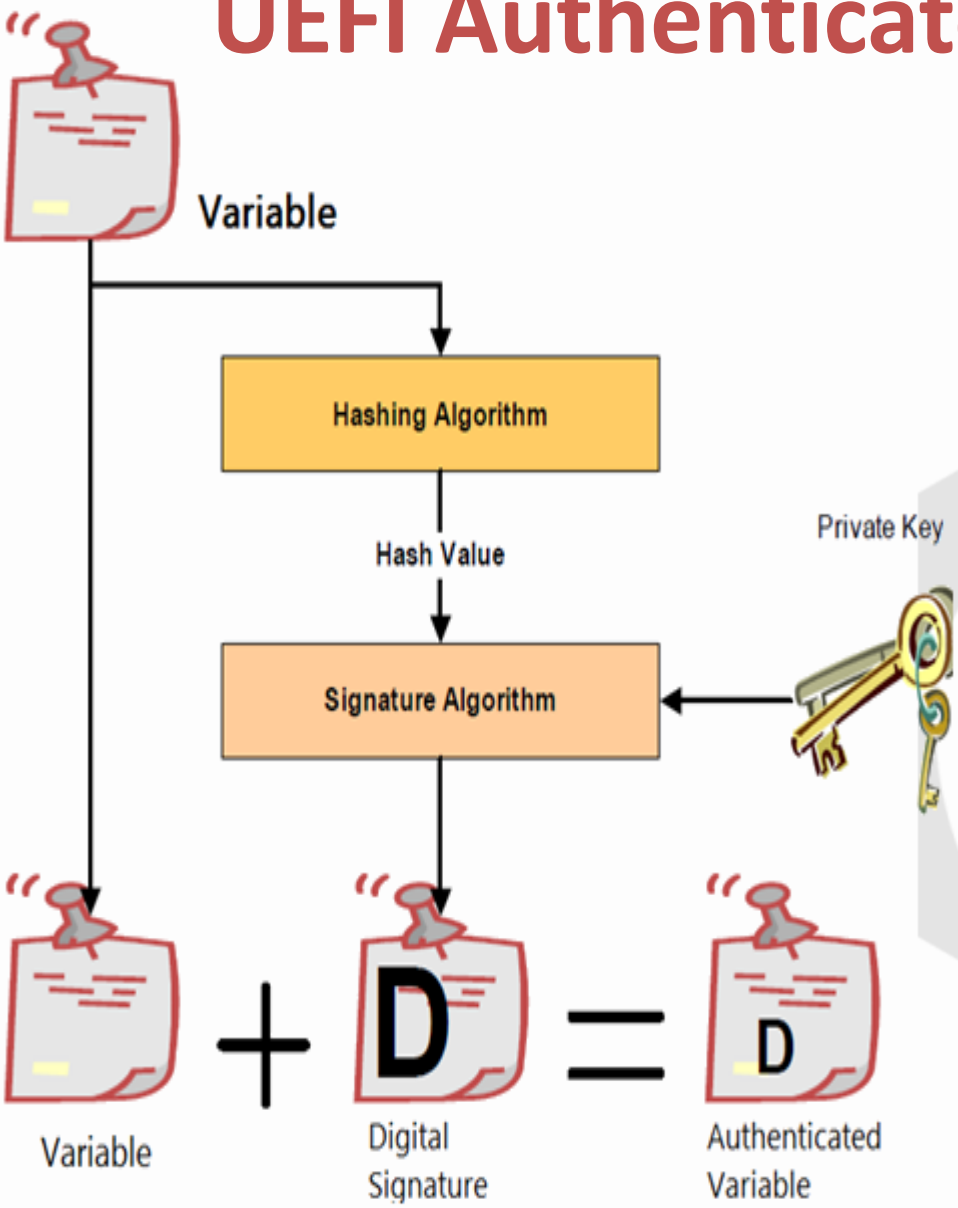
- System Firmware Store is the ‘root of trust’
  - Firmware is hardware-protected
  - All Firmware Updates must be a secure process
- UEFI 2.3.1 provides Building Blocks
- BIOS implements Secure Boot using the Building Blocks in the UEFI Specification

Building Blocks

1. Authenticated Variables
2. Driver Signing
3. System Defined Variables



UEFI Authenticated Variables



- Uses standard UEFI Variable Functions
- Available Pre-boot and also Runtime
- Typically stored in Flash
- Variable Creator signs Variable Hash with Private Key (PKCS-7 format)
- Signature & Variable passed together for Create, Replace, Extend, or Delete
- Several System defined variables for Secure Boot

Extensible Integrity Architecture

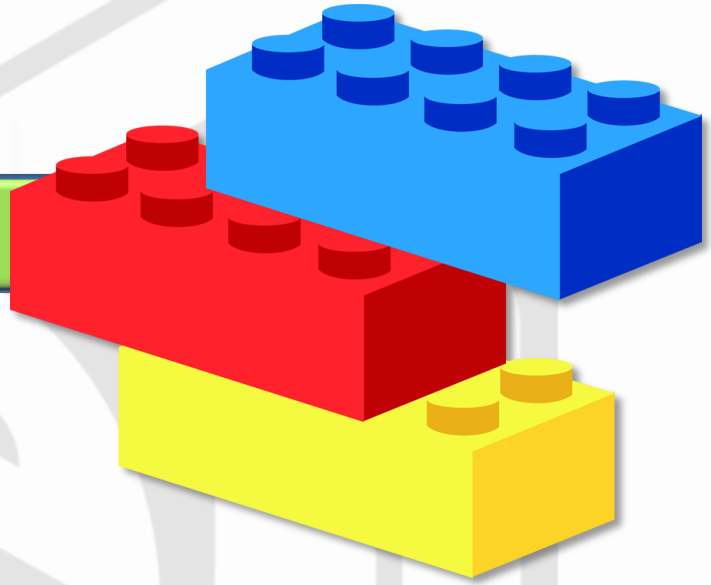
Updating Authenticated Variables

- Support for Append added (UEFI 2.3.1)
- Counter-based authenticated variables (UEFI 2.3)
  - Uses monotonic count to protect against suspicious replay attack
  - Hashing algorithm – SHA256
  - Signature algorithm – RSA-2048
- Time-based authenticated variable (UEFI 2.3.1)
  - Uses timestamp as rollback protection mechanism
  - Hashing algorithm – SHA256
  - Signature algorithm – X.509 certificate chains
    - Complete X.509 certificate chain
    - Intermediate certificate support (non-root certificate as trusted certificate)

Protected Variables that can be Securely Updated

Building Blocks

1. Authenticated Variables
2. Driver Signing
3. System Defined Variables

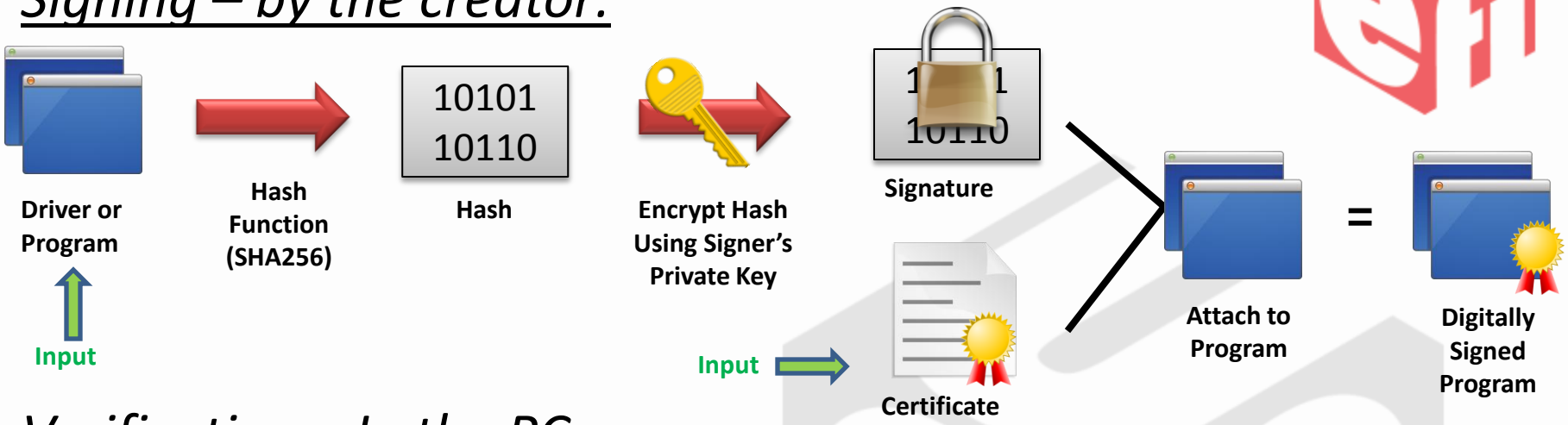


UEFI Driver Signing

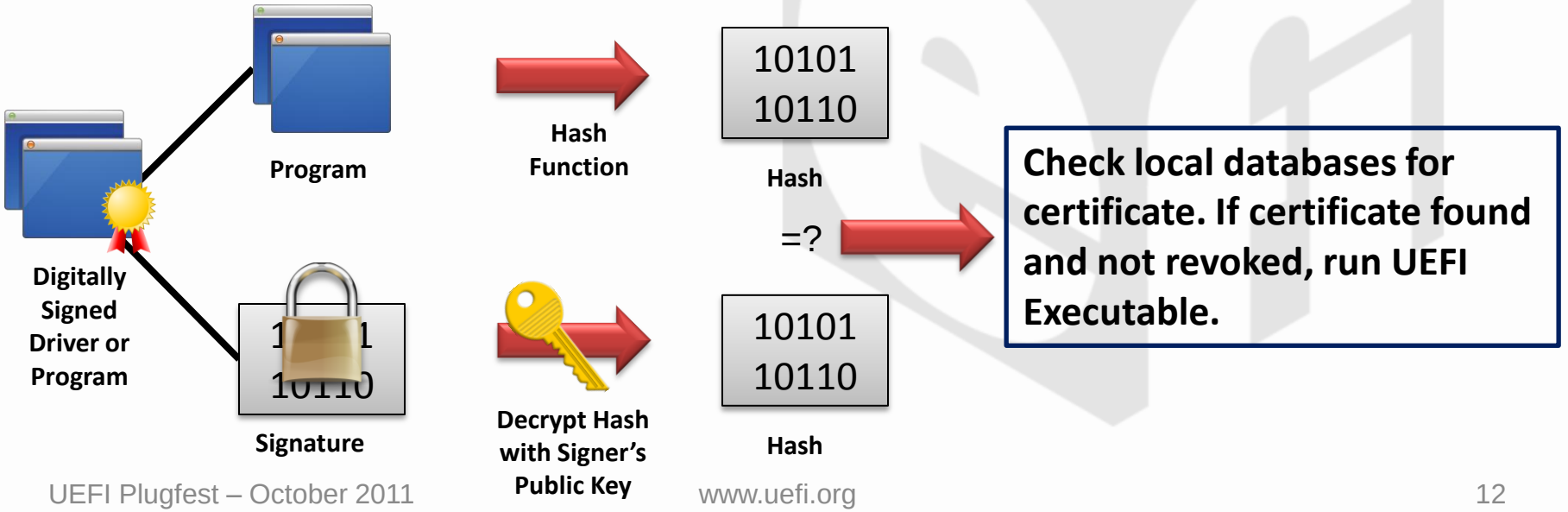
- UEFI Driver Signing utilizes Microsoft Authenticode Technology to sign UEFI executable
- Secure Boot should check these signatures …
  - UEFI Drivers loaded from PCI-Express cards
  - Drivers loaded from mass storage and USB
  - UEFI Shell apps (example: BIOS update utilities)
  - UEFI OS Boot loaders
- UEFI Signing is not required for …
  - Drivers in the factory BIOS
  - Legacy components used only during legacy boots

UEFI Driver Signing Process

Signing – by the creator:



Verification – In the PC:

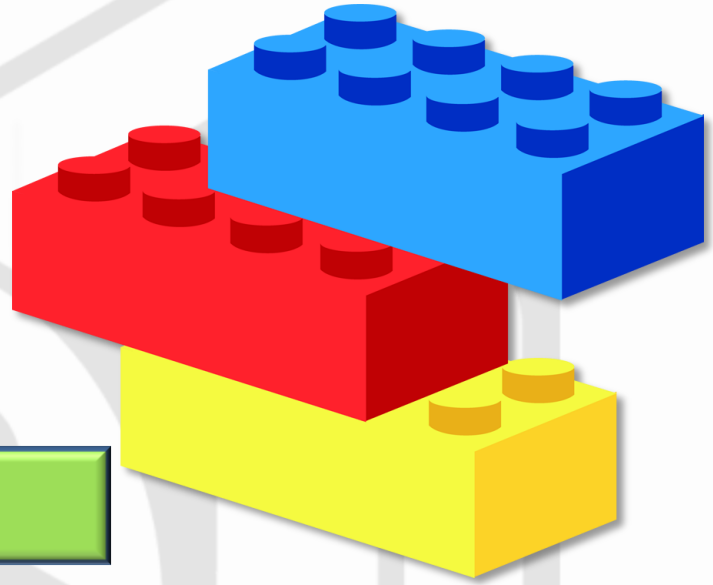


Building Blocks

1. Authenticated Variables

2. Driver Signing

3. System defined Variables



Secure Boot Authenticated Variables

Notes:

- Owner of certificate in KEK can update db, dbx
- Owner of certificate in PK can update KEK

PK	**P**latform **K**ey – Root key set to enable Secure Boot
KEK	**K**ey **E**xchange **K**ey List of Cert. Owners with db, dbx update privilege
db	List of Allowed Driver or App. Signers (or hashes)
dbx	List of Revoked Signers (or hashes)
SetupMode	1= in Setup Mode, 0 = PK is Set (User Mode)
SecureBoot	1 = Secure Boot in force

Building Blocks

1. Authenticated Variables

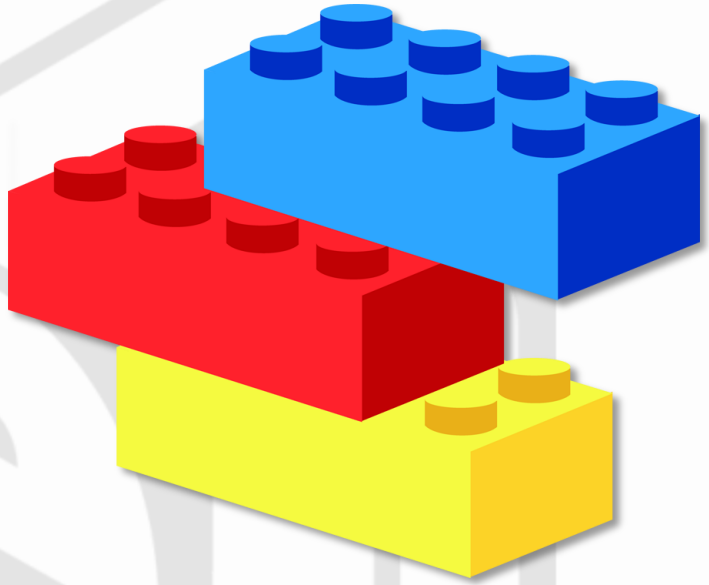
+

2. Driver Signing

+

3. System Defined Variables

=



InsydeH2O Secure Boot

Thanks for attending the
UEFI Fall Plugfest 2011

For more information on the Unified EFI Forum and UEFI Specifications, visit
http://www.uefi.org

presented by

insyde™

But wait, there's more …

T 

~~**Welcoming Remarks –** Aven Chuang, Insyde Software~~
~~**UEFI Forum Updates –** Dong Wei, VP of the UEFI Forum~~

T 

~~**Tips for UEFI Driver Compatibility –** Stefano Righi, American Megatrends, Inc.~~
~~**Understanding Platform Requirements for UEFI HII –** Brian Richardson, Intel Corporation~~

W 

~~**UEFI Security Enhancements –** Kevin Davis, Insyde Software~~
How to Protect the Pre-OS Environment with UEFI – Tony Mangefeste, Microsoft

Th 

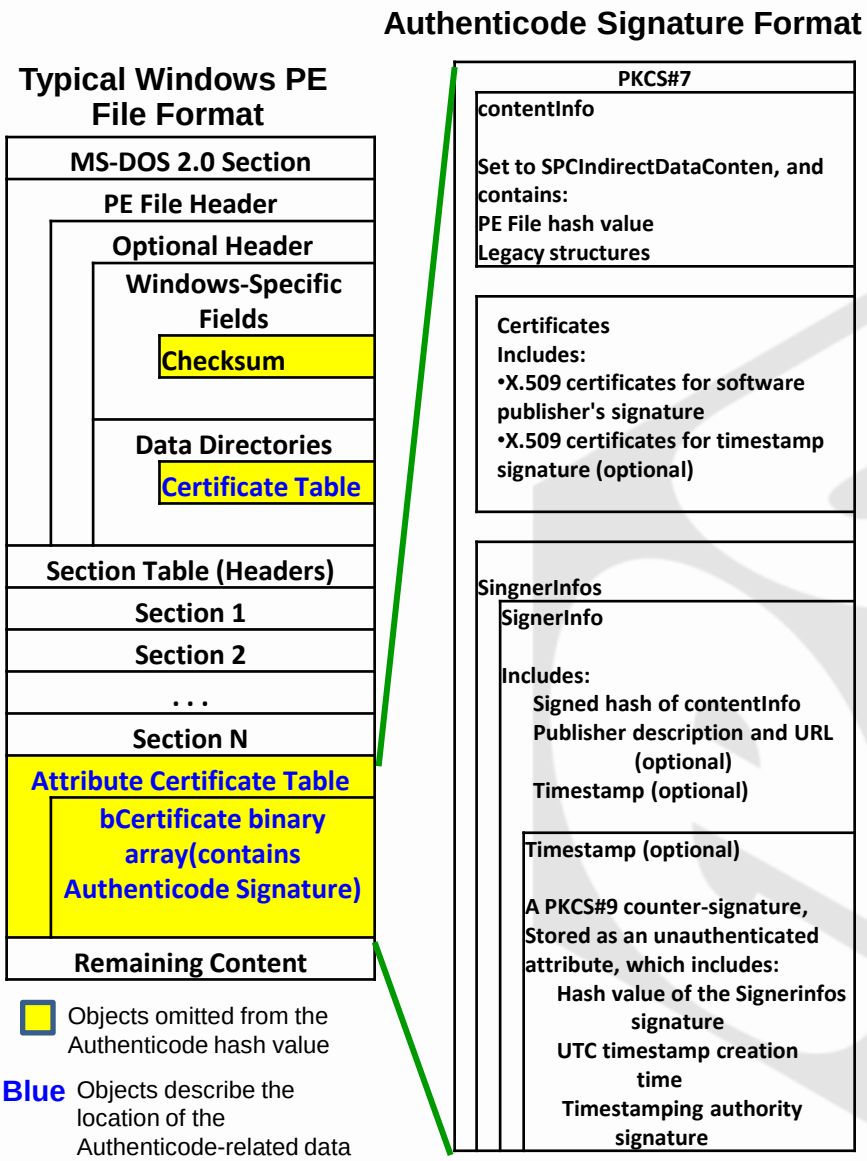
Pre-OS Display Switching using GOP – James Huang, AMD
Debug Methodology Under UEFI – Jack Wang, Phoenix Technologies

Download presentations after the plugfest at www.uefi.org

UEFI Plugfest – October 2011 - UEFI Security Enhancements

Backup Materials

Authenticode Format



Secure Boot begins at the Factory

Pre-production — **Production** — **User**

Certificate Generating Station @ OEM

OEM collects certificates provided by OSVs, Partners, and OEM's own keys.

"DB Generator" creates the Initial Security Load for new computers.

Initial Security Load is installed onto each computer at the factory, enabling Secure Boot.
1) Initial db and dbx
2) KEK with allowed updaters
3) Platform Key (PK)

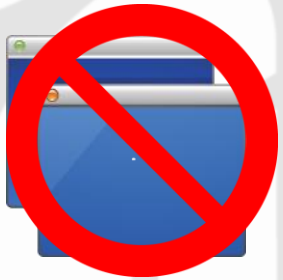
OEM is responsible for Initializing Secure Boot

Secure Boot protects the End User

User attempts to boot a compromised system

OS Boot-loader image checked against pre-loaded database

Root-kit fails checks, user protected by Secure Boot



Secure Boot Tests Signatures to Reject Potential Threats