presented by

Implementing a Secure Boot Path with UEFI 2.3.1

UEFI Summer Plugfest – July 6-9, 2011

Presented by Jeff Bobzin, Insyde Software

Agenda

- **New Security Features of UEFI 2.3.1**
- UEFI Secure Boot
- Implementing a Secure Boot Path with UEFI 2.3.1
- Next Steps
- Q&A

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

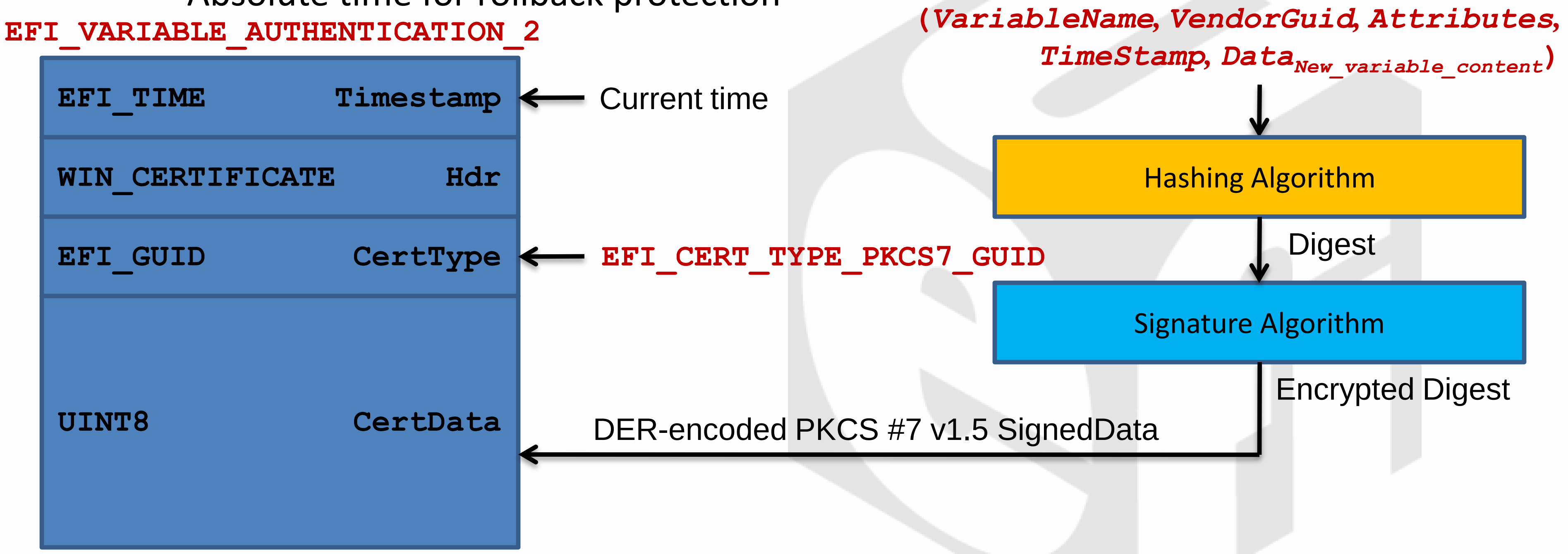
Technology
- USB 3.0

Maintenance
- User Identifier, etc.

UEFI 2.3.1 Enabling More Security Support

2.3.1 Auth. Variable Update

- Append operation for Auth. Variables incl. Signature Databases
- Time-based authenticated Variable
  - Certificate chaining infrastructure
  - Absolute time for rollback protection



Better support servicing of UEFI Secure Boot in a large ecosystem with many actors

More 2.3.1 Security Updates

- Key Management Service (KMS)
  - Services to generate, store, retrieve, and manage cryptographic keys
  - Based on remote key server, or local Hardware Security Module (HSM), or software
- Storage Security Command Protocol
  - Send/receive security protocol defined data to/from mass storage devices
  - Supports Self-encrypting drives – OPAL or eDRIVE
  - Used by OS bootloader to unlock encrypted partition(s)
  - Supported command set
    - **TRUSTED SEND/RECEIVE** (ATA8-ACS)
    - **SECURITY PROTOCOL IN/OUT** (SPC-4)

System Security is a Cooperative Endeavor

Agenda

- New Security Features of UEFI 2.3.1
- **UEFI Secure Boot**
- Implementing a Secure Boot Path with UEFI 2.3.1
- Next Steps
- Q&A

Why Implement UEFI Secure Boot

- As OS becomes more resistant to attack the threat targets the weakest element in the chain
- And 16-bit Legacy Boot is not secure!

*It should be no surprise that a TDL Gang botnet climbed into the **number one** position in the Damballa Threat Report – Top 10 Botnets of 2010. "RudeWarlockMob" … applied effective behaviors of old viruses and kits. It combined techniques that have been effective since the days of 16-bit operating systems, like Master Boot Record (MBR) infection … with newer malware techniques.*

(from http://blog.damballa.com)

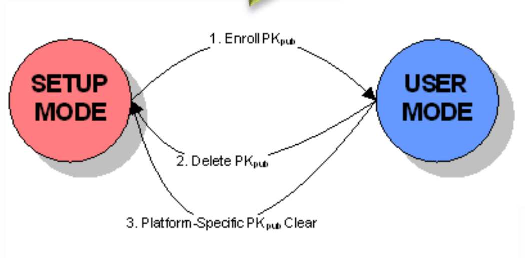
- Secure Boot based on UEFI 2.3.1 removes the Legacy Threat and provides software identity checking at every step of boot – Platform Firmware, Option Cards, and OS Bootloader

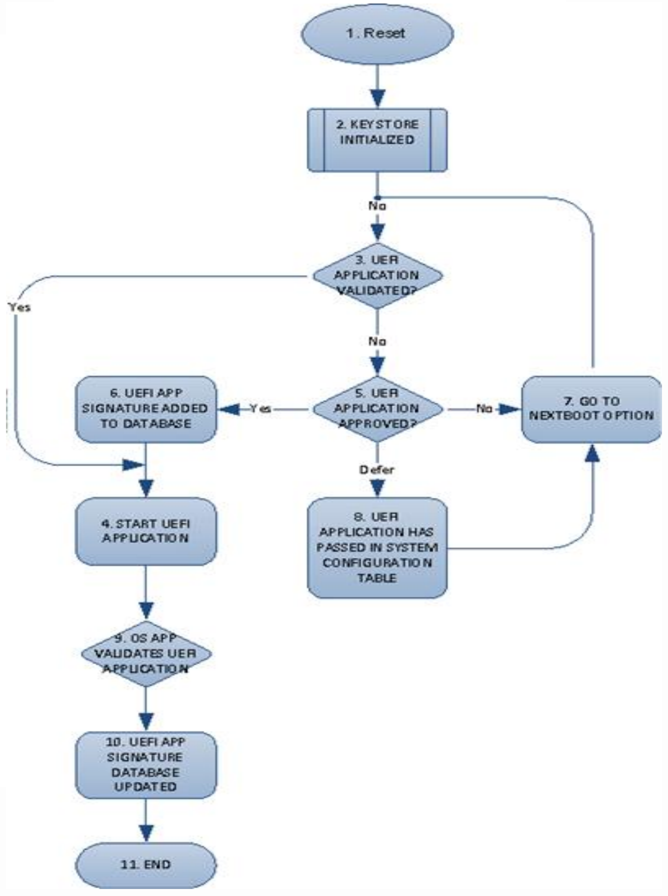
2.3.1

UEFI Secure Boot

Extensive Improvement to UEFI 2.3.1

- Platform security and integrity
  - Allows firmware to authenticate UEFI images, such as OS loader
  - Ensures firmware drivers are loaded in an owner-authorized fashion
- Technology includes:
  - Driver signing, a means of embedding a digital signature of a UEFI executable, and verifying the signature from an authorized source
  - Authenticated variable service
  - New Global defined variables

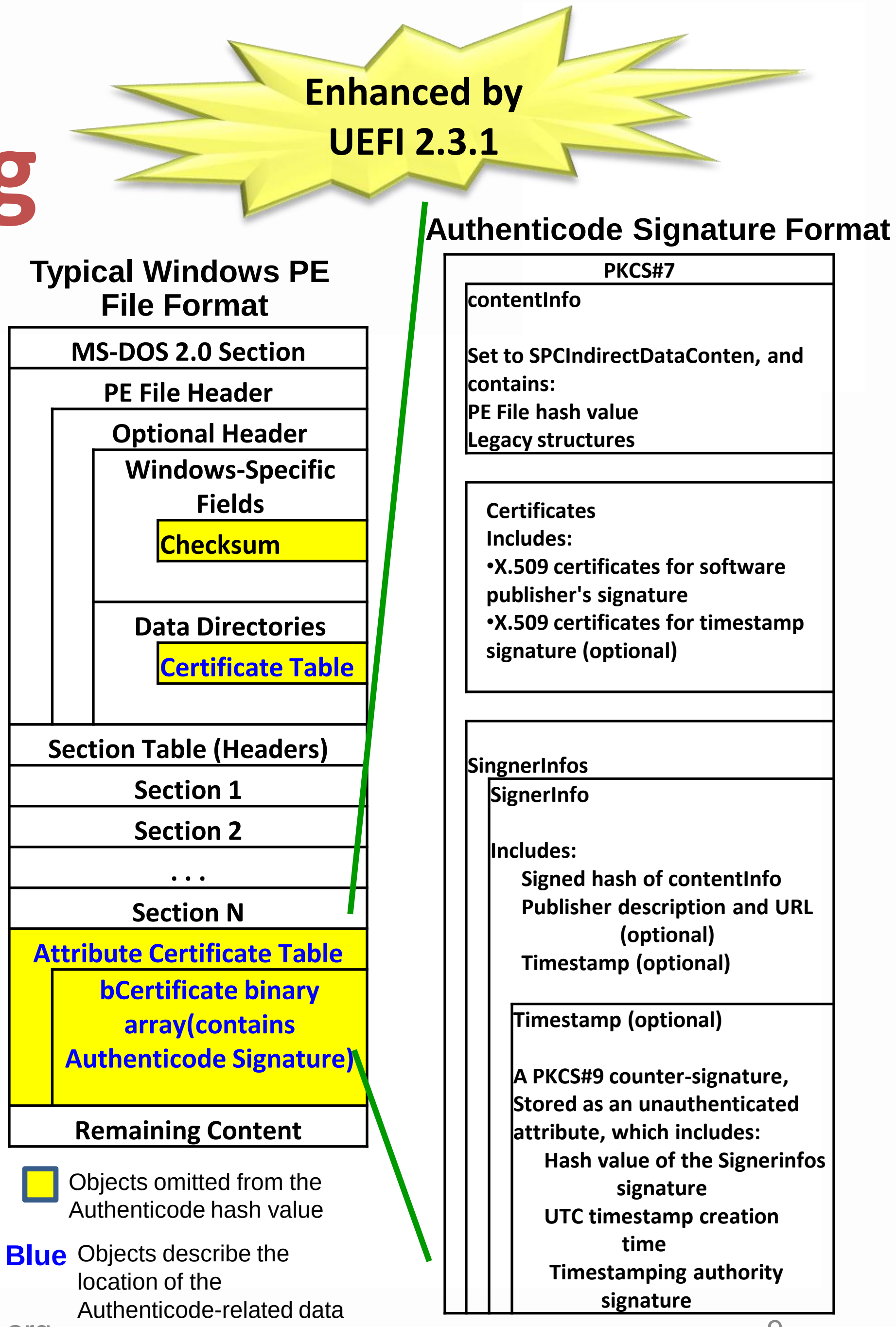



1. Reset
2. KEYSTORE INITIALIZED
No
3. UEFI APPLICATION VALIDATED?
Yes
No
5. UEFI APPLICATION APPROVED?
Yes
6. UEFI APP SIGNATURE ADDED TO DATABASE
No
7. GO TO NEXT BOOT OPTION
Defer
8. UEFI APPLICATION HAS PASSED IN SYSTEM CONFIGURATION TABLE
4. START UEFI APPLICATION
9. OS APP VALIDATES UEFI APPLICATION
10. UEFI APP SIGNATURE DATABASE UPDATED
11. END

UEFI Driver Signing

- Adds policy around UEFI and its 3rd party image extensibility
  - Security for OS loaders, apps, and 3rd party drivers in system
  - Gives IT control around these executables
  - Detects/prevents malware
- Technology includes
  - Supports "known-good" and "known-bad" signature databases
  - Policy-based updates to list
  - Authenticode* signature types (Windows Authenticode Portable Executable Signature Format)

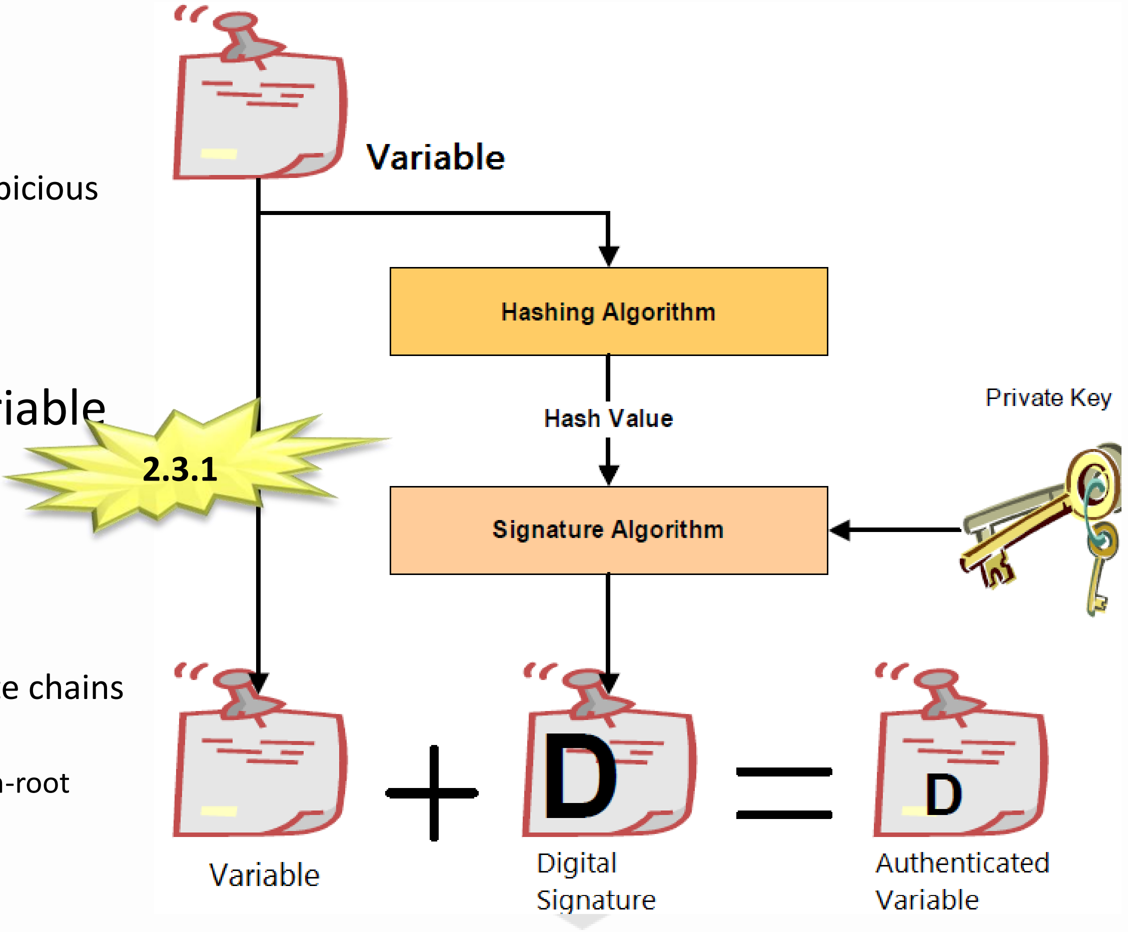
Extensible integrity architecture



Enhanced by UEFI 2.3.1
Typical Windows PE File Format
MS-DOS 2.0 Section
PE File Header
Optional Header
Windows-Specific Fields
Checksum
Data Directories
Certificate Table
Section Table (Headers)
Section 1
Section 2
. . .
Section N
Attribute Certificate Table
bCertificate binary array(contains Authenticode Signature)
Remaining Content
Objects omitted from the Authenticode hash value
Blue Objects describe the location of the Authenticode-related data
Authenticode Signature Format
PKCS#7
contentInfo
Set to SPCIndirectDataConten, and contains:
PE File hash value
Legacy structures
Certificates
Includes:
•X.509 certificates for software publisher's signature
•X.509 certificates for timestamp signature (optional)
SingnerInfos
SignerInfo
Includes:
Signed hash of contentInfo
Publisher description and URL (optional)
Timestamp (optional)
Timestamp (optional)
A PKCS#9 counter-signature, Stored as an unauthenticated attribute, which includes:
Hash value of the SignerInfos signature
UTC timestamp creation time
Timestamping authority signature

UEFI Authenticated Variables

- Counter-based authenticated variable (UEFI 2.3)
  - Uses monotonic count to against suspicious replay attack
  - Hashing algorithm – SHA256
  - Signature algorithm – RSA-2048
- Time-based authenticated variable (UEFI 2.3.1)
  - Use EFI_TIME 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.



Secure Boot – Global Variables

- **PK** - setting this variable moves the machine from non-secure ‘setup mode’ to secure boot ‘user mode’. Variable is self-signed.
- **KEK** - an authenticated variable containing the public certificates of all entities with permission to update the allowed and forbidden databases

Global Variables cont.

- **db** - (Allowed) - an authenticated variable containing public certificates of software signers. Modules signed by these signers are allowed in secure boot mode (user mode). Also can contain hashes of allowed module hashes.
- **dbx** - (Forbidden) - an authenticated variable containing revoked signing certificates and also hashes of forbidden modules

Global Variables cont.

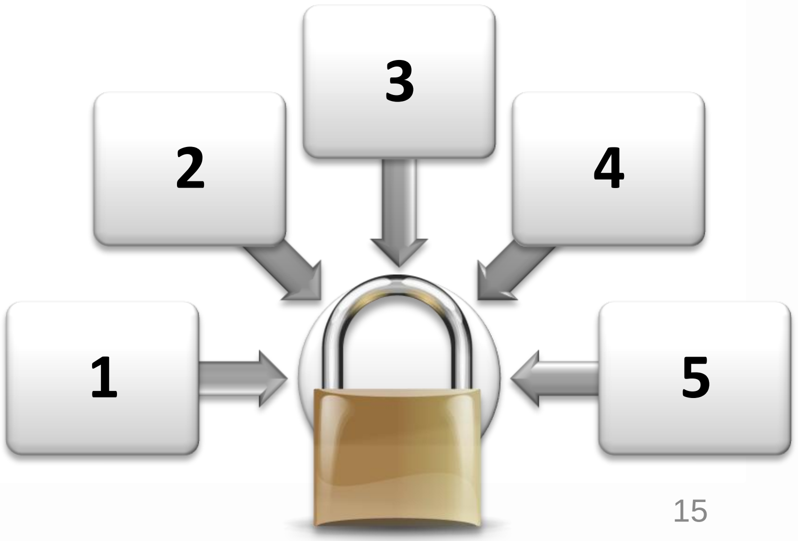
- **SetupMode** - a read-only runtime variable generated by the firmware, means PK is set, 1 = setup, 0 = user mode (PK set, secure boot on)
- **SecureBoot** - a read-only runtime variable generated by the firmware, 1 = PK set plus all conditions for Secure boot were met. On some systems user can choose to suspend SecureBoot checking, for example for recovery, but stay in user mode. This flag informs the OS.

Agenda

- New Security Features of UEFI 2.3.1
- UEFI Secure Boot
- **Implementing a Secure Boot Path with UEFI 2.3.1**
- Next Steps
- Q&A

OEM/IHV Guide to UEFI 2.3.1 Secure Boot

- <u>The Five Elements of Secure Boot Strategy:</u>
  1. UEFI Platform Firmware with 2.3.1 implemented and backed by Strong Firmware Security Policies
  2. Hardware protection of critical security data
  3. Coordination from IBV, IHV and ISV partners
  4. UEFI Factory Provisioning and Field Support Tools
  5. Secure Firmware Update

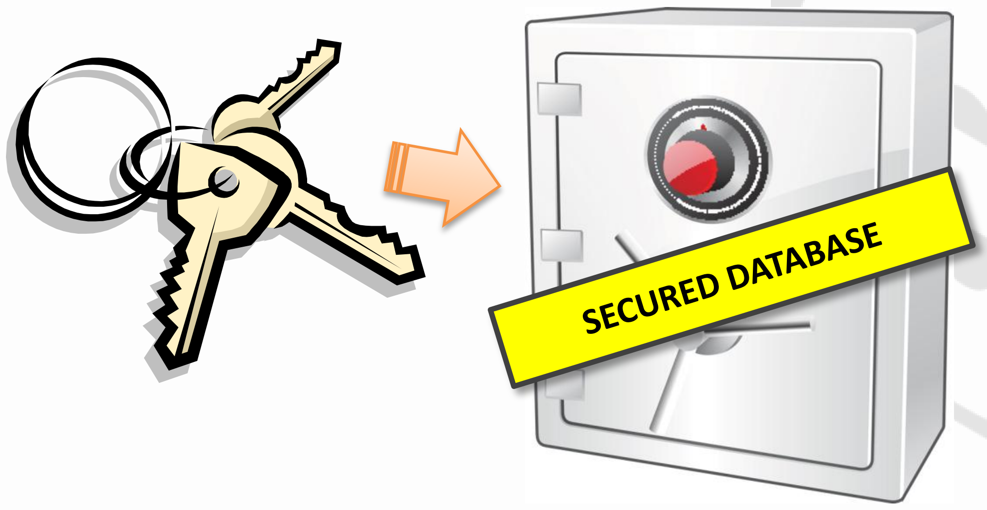


Element #1: UEFI Platform Firmware with 2.3.1 And Strong Firmware Security Policies

- UEFI 2.3.1 is an architectural specification
- But real security strength is in the policy enforcement
- **OEM-ACTION**$\longrightarrow$ Policy must lock-out un-trusted code including all legacy 16-bit code
- But User Experience is key to acceptance:
  - *We ship locked-down secure systems but how much freedom should I give users to reconfigure?*
  - *How does my UI design minimize confusion from users used to "less secure" systems?*

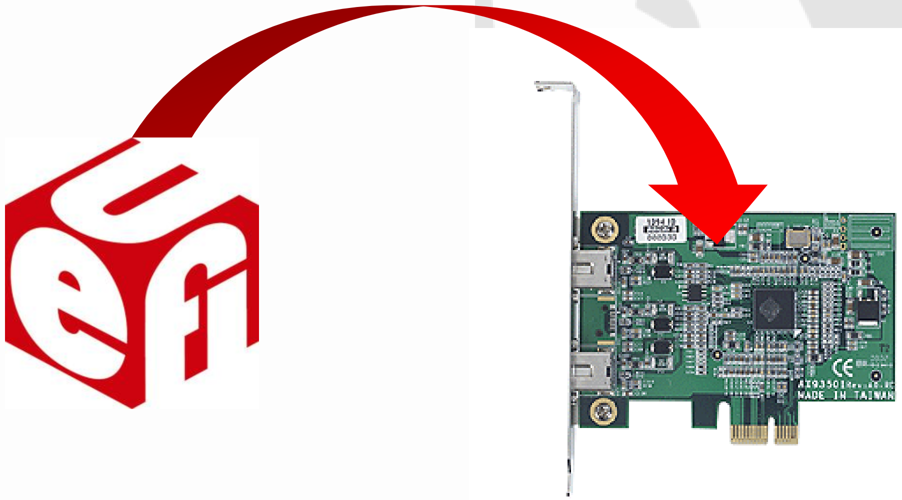
Element #2: Hardware Protection of Critical Data

- Hardware protection of the key database is integral to a secure implementation
- **OEM-ACTION**→ Work with your chipset provider and IBV to implement strong protection of critical data



Element #3: Support from IBV, IHV & ISV Partners

- **OEM-ACTION**→ System ROM will need to contain UEFI drivers for all onboard devices (and no legacy drivers)
- **IHV-ACTION**→ Expansion cards will need Signed UEFI drivers
- **ISV-ACTION**→ Pre-boot software tools, for example bootable recovery disk, will need to be Signed



Element #4: Factory Provisioning

- Several new steps at the end of the factory flow will be required

- **OEM-ACTION**→ Provision with:
  - OS Partner Key
  - OEM Support and Update Key
  - Install Platform Key to lock system

Element #4:... And Field Support Tools

- Any field support tools should be:
  - Signed UEFI executable (using UEFI Shell, not DOS)
  - Shipped pre-signed by the OEM key

- **OEM-ACTION** $\rightarrow$ Examine field support flow, for example
  - Consider what users will do to reinitialize replacement motherboards?

- Support the future - Enterprise Administrator install of Enterprise key
  - Can Enterprise buyer unlock new system and re-provision using your tools?

Element #5: Secure Firmware Update

- Security level of the Firmware Update must match system goals for security

OEM-ACTION→

1. Sign all Firmware Updates images
2. Firmware Update process must occur under control of secure firmware (not in OS)
3. H/W Flash Protection must reject any flash writes from unauthorized sources

Summary

- Industry transition from Legacy to UEFI will impact all industry segments this year
- UEFI 2.3.1 spec update adds significant new value allowing improved protection of the UEFI systems
- Driver signing and authenticated variables are key tools for constructing UEFI Secure Boot
- OEMs need to implement UEFI Secure Boot as part of an integrated strategy in concert with IHV and ISV partners

Implementing a Secure Boot Path with UEFI 2.3.1

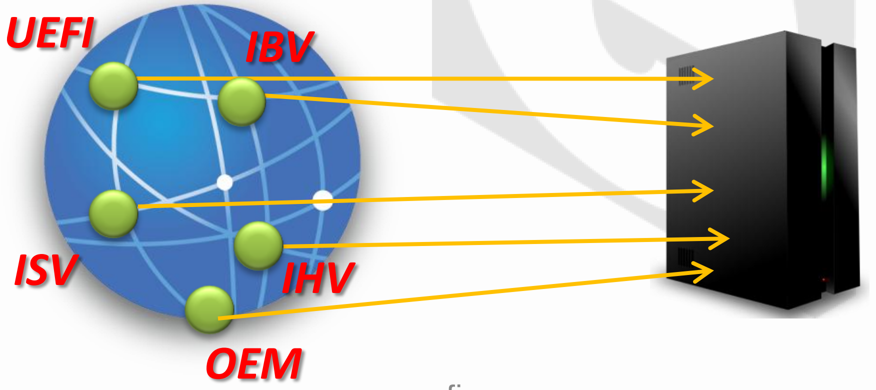
DEMO : Signing Test Tool

Agenda

- New Security Features of UEFI 2.3.1
- UEFI Secure Boot
- Implementing a Secure Boot Path with UEFI 2.3.1
- **Next Steps**
- Q&A

Next Steps

- Download the new UEFI 2.3.1 Spec from www.uefi.org
- OEMs need to implement UEFI boot and use UEFI 2.3.1 security features to harden their systems
- OEMs must work with IBV, IHV and ISV partners in coordinated approach



Q&A

- Questions?

Thanks for attending the UEFI Summer Plugfest 2011

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

presented by

But wait, there's more …

Wed
(July 6)

- UEFI State of the Union (10:30am, Intel)
- Implementing a Secure Boot Path with UEFI 2.3.1 (1:00pm, Insyde)
- UEFI SCT Overview (2:30pm, HP/Intel)

Thu
(July 7)

- Replacing VGA: GOP Implementation in UEFI (10:30am, AMD)
- UEFI prototyping using a Windows-hosted UEFI environment (1:00pm, Phoenix)
- EFI Shell Lab (2:00-4:00pm, "Thunder", Intel)
- GOP Enabling & Testing Lab (4:30—5:30pm, "Thunder", Intel)

Fri
(July 8)

- Best Practices for UEFI Option ROM Developers (10:30am, AMI)

Download presentations after the plugfest at www.uefi.org