Advances in Pre-OS Networking in UEFI 2.4

UEFI Summerfest – July 15-19, 2013
Presented by
Samer El-Haj-Mahmoud
(Hewlett-Packard Company)
Agenda

• Adapter Information Protocol
• Network Media Detection
• iSCSI / FCoE Boot Clarifications
• FCoE SAN MAC Address
• > 256 NICs support
• Summary
Adapter Information Protocol
Adapter Information Protocol

• **New UEFI driver-model API to retrieve and set adapter information**

• **Light weight.** Can be used for dynamically changing data
  – Must never be blocking
  – Quickly return dynamic information, not cached information.
  – Quickly set dynamic settings, not initialize the adapter.

• **Extensible:** New data payloads can be defined (using new GUIDs) without changing the protocol interface.
  – Useful for OEM/IHV/ISV extensions

• **Opaque:** Callers can pass data payloads from/to the adapters without understanding the content
Adapter Information Protocol (AIP)

Protocol Interface Structure

typedef struct _EFI_ADAPTER_INFORMATION_PROTOCOL {
    EFI_ADAPTER_INFO_GET_INFO        GetInformation;
    EFI_ADAPTER_INFO_SET_INFO        SetInformation;
    EFI_ADAPTER_INFO_GET_SUPPORTED_TYPES GetSupportedTypes;
} EFI_ADAPTER_INFORMATION_PROTOCOL;

- **GetInformation**: Gets device state information from adapter. Input is a GUID and a data buffer.
- **SetInformation**: Sets device information for adapter. Input is a GUID, output is a data buffer.
- **GetSupportedTypes**: Gets a list of supported information types for this instance of the protocol.
Network Media Detection
Network Media Detection

• UNDI/SNP Media detection Issues seen in the field with today’s model
  – UNDI Initialize with DETECT_CABLE PXE OP flag sets ‘MediaPresent’ with current cable status
  – Not all IHVs support this (MediaPresentSupported)
  – UNDI with DETECT_CABLE flag may cost ~5 sec
  – UNDI Initialize called multiple times to build the network stack
  – Dynamic media changes (e.g. wireless or cable plug/unplug) are not easy to detect
UEFI 2.3.1 Errata for SNP MediaPresent

• UEFI 2.3.1 Errata D addressed one issue
  – Update description of ‘MediaPresent’ in EFI_SIMPLE_NETWORK_MODE:
    "MediaPresent TRUE if media are connected to the network interface; otherwise FALSE. This field is only valid immediately after calling Initialize()."

  – Updates to UDK2010 SNP implementation to minimize UNDI Initialize calls with CABLE_DETECT
UEFI 2.4 : Network Stack return EFI_NO_MEDIA

• UEFI 2.4 Updates to add EFI_NO_MEDIA Return Status to network protocols
  – Allows media status to propagate from UNDI to SNP to the upper layers of the stack
  – All transmit/receive functions in MNP, UDP, IP, TCP, DHCP, MTFTP (both IPv4 and IPv6)
  – Consumers of the network stack can decide to retry the same network interface, or try another one.
  – Gives flexibility to preform error recovery after a network disconnect.
UEFI 2.4 : AIP : Network Media State

- Dynamic and light weight media state check
- Can be used by callers in polling loops
- Distinguishes no media (EFI_NO_MEDIA) from media that is bouncing (EFI_NOT_READY).
  - Useful for wireless NICs where media state can fluctuate depending on signal strength

```c
typedef struct {
    EFI_STATUS MediaState;
} EFI_ADAPTER_INFO_MEDIA_STATE;
```
iSCSI / FCoE Boot Clarifications
iSCSI / FCoE Current Status

• Lack of standard iSCSI and FCoE Boot Protocols in UEFI Specification caused inconsistent implementations:
  – **UDK2010**: Generic software iSCSI initiator on top of the UEFI TCP/IP stack
  – **NIC IHVs**: iSCSI/FCoE boot using Block I/O directly on the PCI device handle

• Two different methods / UIs to configure iSCSI (and in the future FCoE) boot
UEFI IP4 Stack
- EFI_IP4_CONFIG_PROTOCOL
- EFI_TCP4_PROTOCOL
- EFI_ARP_PROTOCOL
- EFI_IP4_PROTOCOL

EFI_IPSEC_CONFIG_PROTOCOL

UEFI IP6 Stack
- EFI_TCP6_PROTOCOL
- EFI_IP6_CONFIG_PROTOCOL
- EFI_IP6_PROTOCOL

MNP (IPv4 & IPv6 ready)

SNP

UNDI

PCIo

DHCP4

iSCSI (S/W Initiator)

FC/SCSI

FCoE Protocol

DHCP6

System FW

IHV NIC FW

SCSI

UEFI Summerfest – July 2013

www.uefi.org
## iSCSI SW/HW Initiators

<table>
<thead>
<tr>
<th><strong>iSCSI Software Boot Initiator</strong></th>
<th><strong>IHV Block I/O iSCSI Boot</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Generic. Works on top of any UNDI/SNP</td>
<td>IHV / adapter specific</td>
</tr>
<tr>
<td>Does not use any hardware specific features</td>
<td>May use card specific hardware features (acceleration, multi-path I/O, TOE/offloading engine, crypto engine, etc..).</td>
</tr>
<tr>
<td>Consistent configuration UIs</td>
<td>Different configuration UIs for different adapters</td>
</tr>
<tr>
<td>Varies with IHV implementations</td>
<td></td>
</tr>
</tbody>
</table>
# iSCSI SW/HW Initiators

<table>
<thead>
<tr>
<th>iSCSI Software Boot Initiator</th>
<th>Vendor Block I/O iSCSI Boot</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Device Path nodes:</strong></td>
<td><strong>Device Path nodes:</strong></td>
</tr>
<tr>
<td>- Controller (Type 1, Subtype 5)</td>
<td>- Controller (Type 1, Subtype 5)</td>
</tr>
<tr>
<td>- MAC (Type 3, Subtype 11)</td>
<td>- SCSI (Type 3, Subtype 2)</td>
</tr>
<tr>
<td>- IPv4 (Type 3, Subtype 12)</td>
<td>Does not describe MAC address.</td>
</tr>
<tr>
<td>- IPv6 (Type 3, Subtype 13)</td>
<td>Does not describe any of the iSCSI parameters. Relies on IBFT.</td>
</tr>
<tr>
<td>- iSCSI (Type 3, Subtype 19)</td>
<td>Some IHVs:</td>
</tr>
<tr>
<td></td>
<td>- iSCSI (Type 3, Subtype 19)</td>
</tr>
</tbody>
</table>

Describes IP address and iSCSI parameters (initiator name, login, etc...)

Supports both IPv6 and IPv4

Can support IPv6/IPv4 if IHV implements it
iSCSI Clarifications

• UEFI 2.3.1 hints to this issue, but doesn't offer methods for detecting the different iSCSI drivers and their capabilities.

• UEFI 2.4 clarifies this further

15.1.1 iSCSI UEFI Driver Layering

iSCSI UEFI Drivers may exist in two different forms:

• iSCSI UEFI Driver on a NIC:
  The driver will be layered on top of the networking layers. It will use the DHCP, IP, and TCP and packet level interface protocols of the UEFI networking stack. The driver will use an iSCSI software initiator.

• iSCSI UEFI Driver on a Host Bus Adapter (HBA) that may use an offloading engine such as TOE (or any other TCP offload card):
  The driver will be layered on top of the TOE TCP interfaces. It will use the DHCP, IP, TCP protocols of the TOE. The driver will present itself as a SCSI device driver using interfaces such as `EFI_EXT_SCSI_PASS_THRU_PROTOCOL`.

To help in detecting iSCSI UEFI Drivers and their capabilities, the iSCSI UEFI driver handle must include an instance of the `EFI_ADAPTER_INFORMATION_PROTOCOL` with a `EFI_ADAPTER_INFO_NETWORK_BOOT` structure.
AIP : iSCSI/FCoE Boot Info

• UEFI 2.4 defines a new AIP instance to detect NIC adapter iSCSI/FCoE boot capabilities
• Platform policy (or user configuration) can select the proper SW / HW iSCSI/FCoE boot Initiator
AIP : iSCSI/FCoE Boot Info

```c
typedef struct {
    BOOLEAN iScciIpv4BootCapability;
    BOOLEAN iScciIpv6BootCapability;
    BOOLEAN FCoeBootCapability;
    BOOLEAN OffloadCapability;
    BOOLEAN iScsiMpioCapability
    BOOLEAN iScsiIpv4Boot;
    BOOLEAN iScsiIpv6Boot;
    BOOLEAN FCoeBoot;
} EFI_ADAPTER_INFO_NETWORK_BOOT;
```

- **Boot Capabilities:**
  - iSCSI IPv4/IPv6 and FCoE boot support
  - Offload engine
  - Multi Path I/O

- **Configured Boot Mode:**
  - iSCSI IPv4/IPv6
  - FCoE
  - None (System Firmware can still enable PXE IPv4/IPv6)
FCoE SAN MAC Address
FCoE SAN MAC Address

- FCoE SW Initiator running on top of NIC stack
  - The Fibre Channel WWID and the SAN MAC address are sufficient to create a well-formed FCoE endpoint address
  - Ethernet MAC Address is still used
  - UEFI Device Path describes Ethernet MAC and FC WWID, but not FC SAN MAC.
AIP : FCoE SAN MAC Address

typedef struct {
    EFI_MAC_ADDRESS SanMacAddress;
} EFI_ADAPTER_INFO_SAN_MAC_ADDRESS;

SanMacAddress Returns the SAN MAC address for the adapter.
For adapters that support today’s 802.3 ethernet networking and Fibre-Channel Over Ethernet (FCOE), this conveys the FCOE SAN MAC address from the adapter

Note: A future UEFI 2.4 errata may update this to add iSCSI MAC address as well (since iSCSI HW Initiator may not install UNDI or MAC Device Path node)
> 256 NICs Support
> 256 NICs Support

• **UEFI 2.3.1**
  – Limit to 256 UNDI instances, per managing UNDI driver.
  – Some highly scalable servers can hit this limitation

• **UEFI 2.4**
  – Expand UNDI support to 65K NICs per UNDI driver.
  – New revision of `EFI_NETWORK_INTERFACE_IDENTIFIER_PROTOCOL` and `!PXE` structure
  – Fix inconsistent definitions of IFcnt and IfNum
Summary

- UEFI 2.4 Adds multiple enhancements for UEFI networking capabilities (iSCSI/FCoE, Media Detection, etc...)
- NIC IHVs are encouraged to start adopting
- UDK2010 implementation need to catch up to 2.4 spec
References

• UEFI 2.4 Specification
  – http://www.uefi.org/specs/

• “Advances in pre-OS Networking in UEFI 2.4” White Paper
  – Vincent Zimmer (Intel), Samer El-Haj-Mahmoud (HP), Fiona Jensen (Apple)
  – To be published on http://uefidx.com
Thanks for attending the UEFI Summerfest 2013

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

presented by