System Prep Applications
A Powerful New Feature in UEFI 2.5

UEFI Spring Plugfest – May 18-22, 2015
Presented by Kevin Davis

NEW!
Insyde Software
Agenda

- Introduction
- Current Methods
- Current Example
- Solution
- Call to action
- Questions?
SysPrep Purpose

• Some UEFI Drivers and Applications need to run before the OS to Prepare the System
• Possible examples:
  – Encrypted Storage Unlock
  – User identification and verification
  – System provisioning and maintenance from central management with Network Boot
  – Firmware version check up-to-date
  – Early Disability Accessibility UI provider
Current Methods

• DRIVER# works great!
  – Pros:
    • Very early
  – Cons:
    • Limited devices available
    • Driver Model Start requires driver to be attached to a device
    • No console rights available for user interactions

• BOOT# works great!
  – Pros:
    • Console available for user interactions
  – Cons:
    • OSs and OEMs modify Boot Order for to be first or value-add
    • Too late in the process
    • Lots of issues trying to understand all of the other items in the BOOT#
Example - Disk Encryption

• Protect data at rest on the desk
  – User enter PIN or uses biometric device
    • Receive unlock permission
  – Unlock the encrypted OS and Data device
  – Perform recovery actions if invalid permission
  – Currently use LOAD_OPTIONCATEGORY_BOOT to manage load of application
Example - Disk Encryption (con’t)

• Currently App needs to integrate into BootOrder
  – App Runtime needs to start before the OS loader
  – OS installers tend to change the BootOrder during initial installs or re-installs
    • Can cause app to fail and OS disk to remain encrypted

• But App Runtime needs to stay before OS
  – Need to review BootOrder on every boot?
  – How?
SysPrep#### is the solution

- Similar handing as Boot#### and Driver####
- Loaded after Driver####
- Loaded before Boot####
- Must be EFI_IMAGE_SUBSYSTEM_EFI_APPLICATION!
- Attributes sub-field LOAD_OPTION_CATEGORY is ignored!
Resources Available

• Same console rights as Boot### target - and can ask for console if not previously started by platform policy
• Same network rights as Boot### target - and can ask for network if not previously started by platform policy
Boot Manager Policy Protocol

• Protocol published by the UEFI Boot Manager

• Can be used by Sysprep or other EFI Applications
  – Purpose: Connection requests for any required devices using platform policy.
typedef struct

_UEFI_BOOT_MANAGER_POLICY_PROTOCOL  EFI_BOOT_MANAGER_POLICY_PROTOCOL;

struct _UEFI_BOOT_MANAGER_POLICY_PROTOCOL {

    UINT64 Revision;

    EFI_BOOT_MANAGER_POLICY_CONNECTDEVICEPATH
        ConnectDevicePath;

    EFI_BOOT_MANAGER_POLICY_CONNECTDEVICECLASS
        ConnectDeviceClass;

};

// Classes for ConnectDeviceClass
#define EFI_BOOT_MANAGER_POLICY_CONSOLE_GUID  
    {0xCAB0E94C,0xE15F,0x11E3,{0x91,0x8D,0xB8,0xE8,0x56,0x2C,0xBA,0xFA } } }  
#define EFI_BOOT_MANAGER_POLICY_NETWORK_GUID   
    {0xD04159DC,0xE15F,0x11E3,{0xB2,0x61,0xB8,0xE8,0x56,0x2C,0xBA,0xFA } } }  
#define EFI_BOOT_MANAGER_POLICY_CONNECT_ALL_GUID  
    {0x113B2126,0xFC8A,0x11E3,{0xBD,0x6C,0xB8,0xE8,0x56,0x2C,0xBA,0xFA } } }
Security

• For systems implementing UEFI SecureBoot, App MUST be signed!
  – Could be UEFI CA
  – Could be OEM / ODM / IBV CA
  – OEM could include ISV’s Cert in db
Call to action

• ISVs, IHVs
  – Work with a partner to validate your solution
• Other BIOS companies
  – Implement!
  – Work with a partner to validate your solution
• Be ready by the next plugfest
Thanks for attending the UEFI Spring Plugfest 2015

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

presented by

insyde®