How to Make Ironic Baremetal Deployment More Secure and Reliable?

Wan-yen Hsu
Shivanand Tendulker
Ramakrishnan G
HP
June 6, 2015
Agenda

• Ironic Baremetal Provisioning Security and Reliability Challenges
• Features in Ironic Kilo and iLO Drivers to Mitigate Security Risks and Enhance Reliability
• Demo
Ironic Baremetal Provisioning Security and Reliability Challenges (Pre-Juno)

• No Ironic mechanism existed to prepare a node for a clean start of initial deployment or re-deployment
• Manual hardware property configuration is cumbersome and error-prone
• Lack of segregation of management network from data network
• tftp protocol used by PXE driver is neither secure nor reliable
• No mechanism exists to prevent loading of un-authorized option ROMs, bootloader, and kernel
Features in Ironic Kilo and iLO Driver to Enhance Security and Reliability

• Node Cleaning
• Hardware Property Introspection
• PXE-less iLO Virtual Media Deployment
• Secure Boot
  - Secure Boot Overview
  - Secure Boot in iLO Drivers
Ironic iLO drivers

- iLO drivers are Ironic plug-in drivers for HP ProLiant servers
- Integrate iLO and ProLiant platform capabilities with Ironic to optimize OpenStack baremetal provisioning for ProLiant servers

Features
- PXE-less Virtual Media Deployment (Juno & Kilo)
- UEFI Boot and Boot Mode Management (Juno & Kilo)
- Node Cleaning (kilo)
- Hardware and Firmware Property Introspection (Kilo)
- Secure Boot (Kilo)
- Boot Options (Kilo)

Document:
- http://docs.openstack.org/developer/ironic/drivers/ilo.html
Kilo Node Cleaning and ILO Drivers

• **Kilo Node Cleaning**
  • Preparing a node for a clean start of initial deployment or re-deployment
  • Pluggable Ironic node cleaning framework

• **iLO Driver Node Cleaning Features**
  • Reset system ROM firmware settings to baseline
  • Reset secure boot keys to manufacturer’s defaults
  • Clear secure boot keys
  • Reset iLO credential
  • Reset iLO
  • Disk erase

• **Node cleaning Configuration**
  • Enabled (default) or disabled via ironic.conf: `[conductor] clean_nodes=[true| false]`
  • Cleaning step priority is configurable- default priorities, change of priority, or setting a cleaning step priority to zero to skip the step.
Kilo Hardware Introspection and iLO Driver Value-adds

• Kilo Hardware Introspection
  ▪ Standard Ironic API and CLI to discover baremetal node’s properties (CPU architecture, Number of CPUs, Memory and disk sizes)
  ▪ Ease of use and less error prone. Automatic configuration of the discovered properties for bare-metal nodes.

• iLO Driver Introspection Features
  ▪ Discovery of extra hardware and firmware capabilities
    • Secure Boot
    • System ROM Firmware Version
    • iLO Firmware Version
  ▪ Automatic creation of node capabilities for the discovered properties
    For instance, automatic configuration of node capabilities='secure_boot:true'
  ▪ Users can create flavors based on node capabilities discovered by iLO driver to place a workload on a bare-metal node that matches its security, reliability, and resource needs (e.g., flavors for secure_boot, firmware version with hot fixes, 10G link speed)
    nova flavor-key secure-deploy set capabilities:secure_boot="true"
    nova boot --flavor secure-deploy --image test-image instance-1
  ▪ Out-of-band introspection: Faster and introspection anytime independent of OS running
PXE-less iLO Virtual Media Deployment

- Deploy ramdisk and kernel are booted via iLO virtual media over a separate management network
- No need to set up TFTP server and PXE environment
- User image is retrieved over data network to speed up large image transfer
- Users need to enable Swift over HTTPS or SSL proxy

iLO PXE-less Deploy Drivers

- iLO virtual media iSCSI driver (iscsi_ilo driver)
  Uses iLO virtual media to boot deploy kernel and ramdisk and then provisions user image from Ironic conductor to bare metal node via iSCSI.

- iLO virtual media IPA driver (agent_ilo driver)
  Uses iLO virtual media to boot Ironic Python Agent ramdisk. IPA then deploys user image from bare-metal node.

- Automatic ISO image creation from user partitioned ramdisk and kernel images.
- DIB and IPA tools to build ISO images for deploy ramdisk and kernel.
UEFI Secure Boot

• Industry standard
• Supported in Window 2012+ and many Linux distributions
• No TPM dependency
• Chain of Trust and Validation
  - Each piece of boot component is signed.
  - Signature databases (db and dbx) are embedded in the system.
  - Signature of each component is validated by the previous component in the chain before allowing it to execute.
• Prohibits loading of unauthorized option ROMs, boot loaders, kernel and kernel modules.
• Mitigates security risks including malicious firmware and OS attacks

An Example of Linux Secure Boot Chain of Trust and Validation

[Diagram showing the chain of trust starting from Firmware, through Shim, Grub2, and finally OS, with validation points at each step]
Secure Boot

Signing – by the creator:

Driver or Program -> Hash Function (SHA256) -> Hash -> Encrypt Hash Using Signer’s Private Key -> Signature

Signature -> Attach to Program

= Digitally Signed Program

Verification – In the system:

Program -> Hash Function

Digitally Signed Driver or Program

Signature

Signature with Signer’s Public Key

10101 10110

= ?

Check local databases for certificate. If certificate found and not revoked, run UEFI Executable.
UEFI Secure Boot in iLO Drivers (Kilo)

- Automatic or manual creation of node capability for secure_boot
  iLO drivers can be configured to deploy a node in secure boot by either of two ways:
  - Performing hardware introspection on the Gen9 ProLiant node to populate secure_boot node capabilities: (recommended)
  - or
  - Using Ironic CLI to add ‘secure_boot’ to node properties/capabilities’

- Create Nova flavor for secure_boot to place workload on a ProLiant server with secure_boot capability
  `nova flavor-key SECBOOT_FLAVOR set capabilities:secure_boot="true"`

- Nova boot with secure_boot flavor
  `nova boot --flavor SECBOOT_FLAVOR --key-name KEY_NAME INSTANCE_NAME`

- No manual configuration of secure boot enablement is required.
  UEFI boot mode and secure boot enablement are done automatically in the iLO drivers as needed.

- Added Disk Image Builder (DIB) options to create digitally signed deploy and user images
Apply the following Kilo features to improve security and reliability for ironic bare-metal provisioning:

- Hardware introspection to prevent node configuration errors
- Creating flavors based on node capabilities discovered by iLO driver to place a workload on a bare-metal node that matches its security and reliability needs
  - e.g., flavors for secure_boot, firmware version with hot fixes, 10G link speed
- Node cleaning to prepare a node for a clean start of deployment or re-deployment
  - Set ProLiant firmware settings to baseline
  - Reset iLO credential
  - Scrub disks
- PXE-less iLO virtual media deployment to segregate management and data networks
- iLO driver secure boot to mitigate risk of malicious firmware and software attacks.
- Happy Ironic bare-metal deployment 😊
iLO Driver Demo

• Hardware Introspection
• PXE-less iLO Virtual Media Deployment with Secure Boot
• Node Cleaning