Securing Linux VMs in a Hosted Environment

mikbras@microsoft.com

Goal – attacks from the outside

To protect Linux VMs from **outside attacks** (from processes running on the host).

- Injecting code into the boot chain.
- Stealing data from private disk volumes.
- Spying on a VM’s memory.
- Hacking a VM’s persistent state data.
- Attaching a debugger to a VM.

Non-Goal – attacks from within

Improving protection of Linux from **inside attacks** (from Linux root processes) is not a goal of this effort.

Our operating environment

Windows Server 2016 introduces **virtualization security** features.

- Hyper-V adds **VM shielding.**
  - Protects VM resources
  - Utilizes UEFI and virtual TPM 2.0 support
- Windows OS enhanced to utilize **VM shielding.**

What is VM shielding?

Hyper-V defines a new “shielded mode” for running VMs.

- VM memory is protected from privileged processes.
- VM configuration is encrypted.
- VM persistent execution state is encrypted.
- Console access is disabled.
- Debugger attachment is disabled.
- Hyper-V verifies the first-stage bootloader via UEFI.
- Windows OS secures boot chain and encrypts volumes.

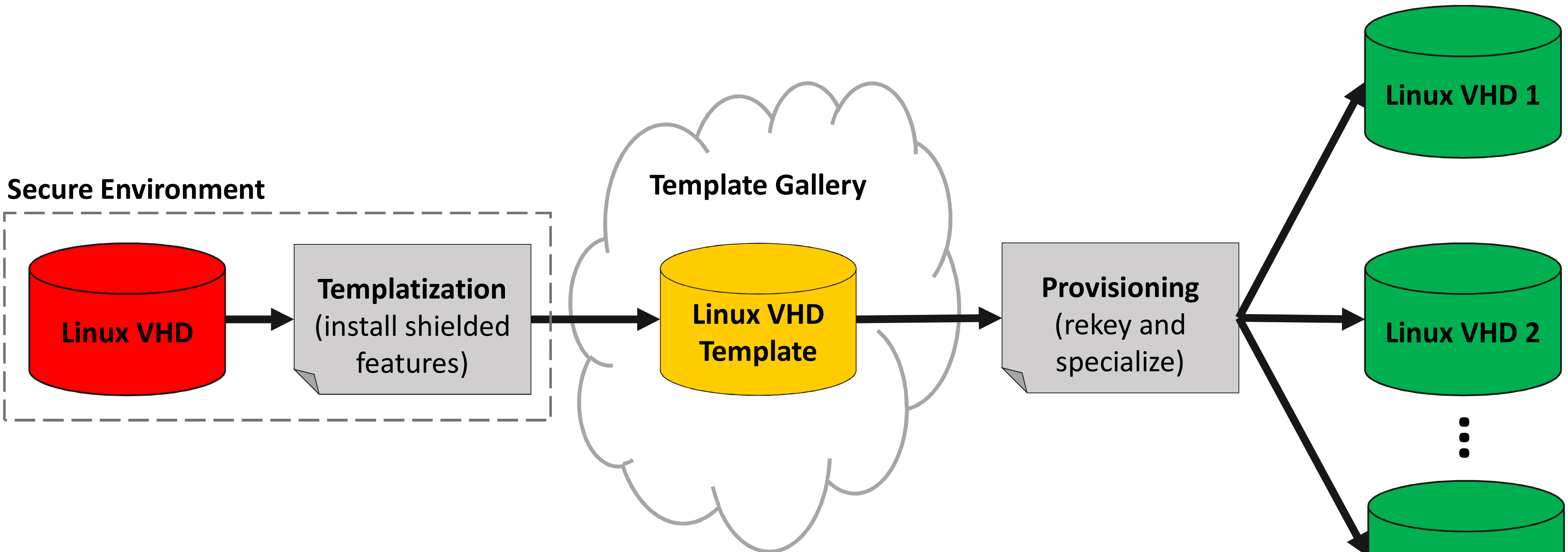
Main objective – Linux Shielded VMs (LSVM)

To enhance Linux security so that Linux VMs may execute in **shielded mode** (rather than **normal mode** as they do today).

Recap of Mechanisms for establishing trust

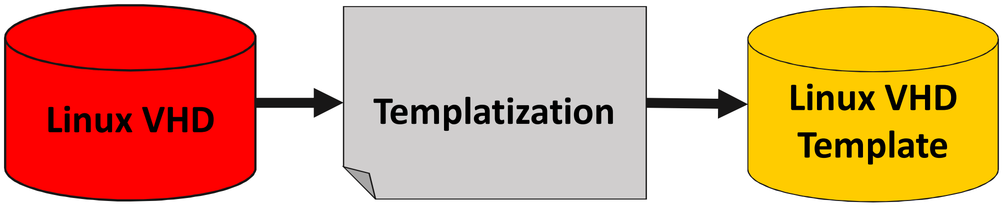
1. UEFI Secure Boot
2. Linux Secure Boot
3. TPM Measurements
4. Disk Encryption

Lifecycle of a Linux VM



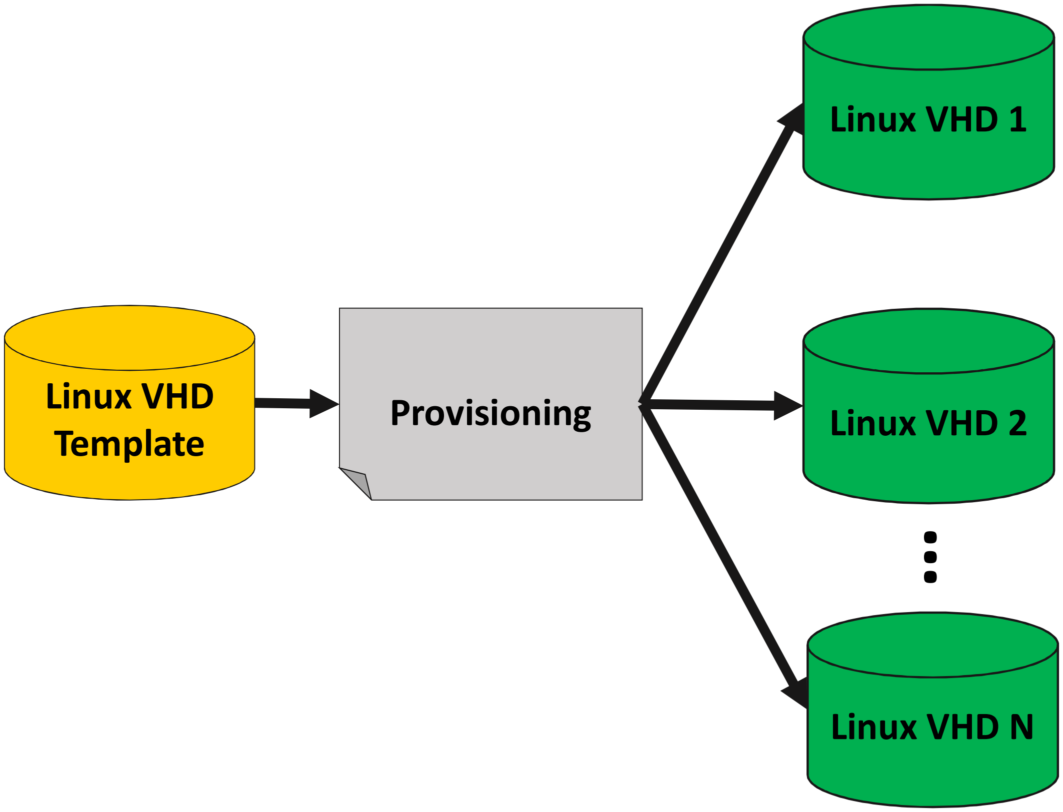
Templatization

1. Install first-stage boot loader to ESP.
2. Encrypt boot partition with well-known key.
3. Encrypt root partition with well-known key.
4. Update initial ramdisk with unseal utility.
5. Generate partition signatures.
6. Install mini provisioning OS to ESP.
7. Publish template

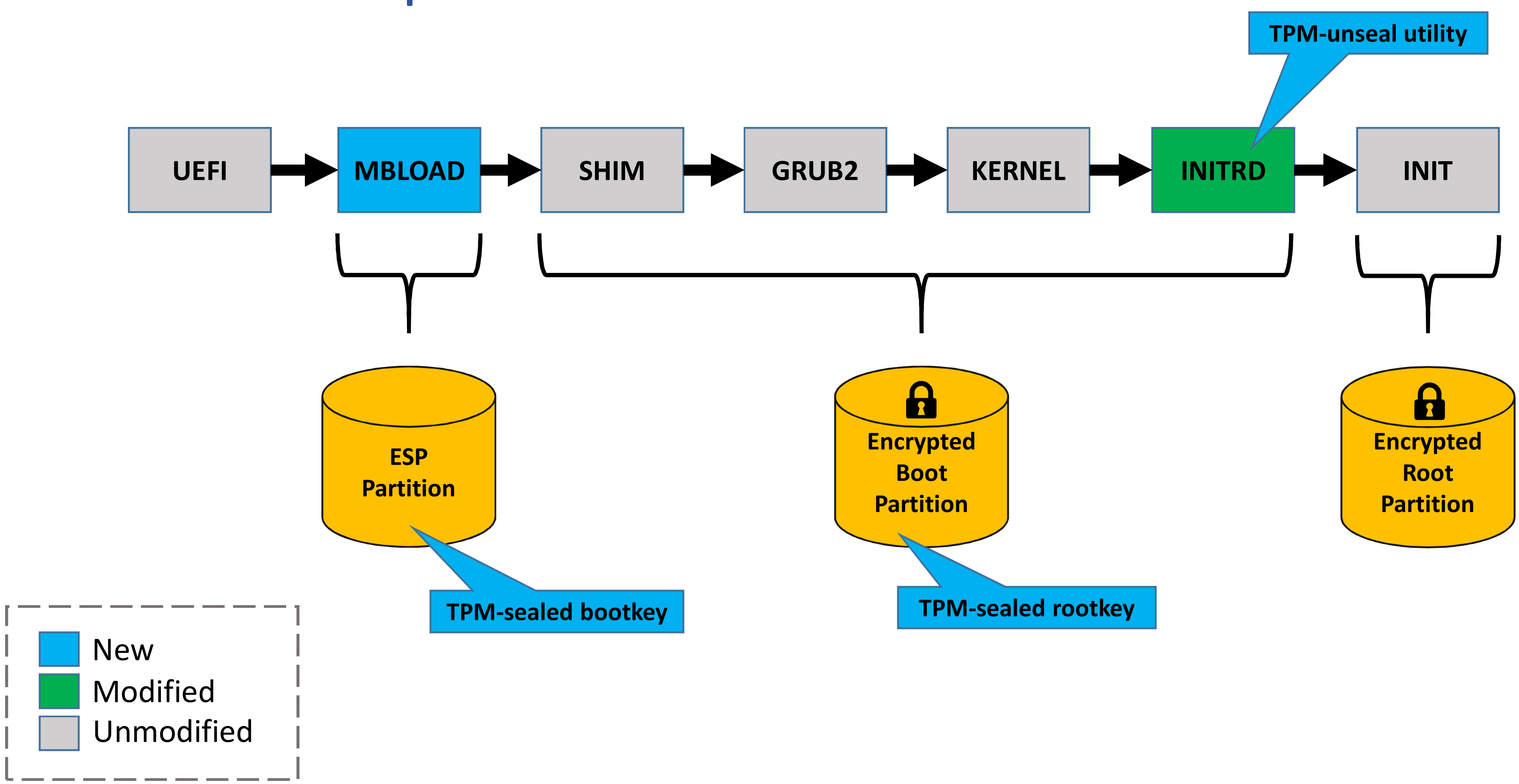


Provisioning

1. Boot into mini provisioning OS.
2. Contact central provisioning service.
3. Verify the publisher.
4. Verify partition signatures.
5. Re-encrypt root partition with owner key.
6. Re-encrypt root partition with owner key.
7. Seal boot and root keys with TPM.
8. Encrypt specialization data to ESP.
9. Remove mini provisioning OS from ESP.
10. Boot into Linux VM for first time.
11. Perform specialization.



The boot path



Boot steps

1. UEFI verifies and loads MBLOAD
2. MBLOAD unseals the bootkey (used to decrypt boot partition).
3. MBLOAD installs EFI I/O hooks (to capture all shim and GRUB I/O).
4. MBLOAD verifies and executes the shim (Microsoft UEFI CA).
5. The shim verifies and executes GRUB2.
6. GRUB2 executes the kernel with the initial ramdisk.
7. Initial ramdisk uses unseal utility to unseal root key and boot key..
8. The initial ramdisk mounts these partitions.
9. System boot continues normally.

Implications of last two slides

1. The Shim and GRUB2 are used as-is without modification.
2. Linux secure boot are preserved intact.
3. The Shim and GRUB2 are copied to the encrypted boot partition during templatization.
4. An unseal utility is copied into the initial ramdisk during templatization.
5. The initial ramdisk script that performs “luksOpen” is modified to use the rootkey, making boot non-interactive.
6. MBLOAD transparently redirects EFI I/O to the encrypted boot partition.
7. TPM 2.0 is used to unseal the boot and root keys.
8. No dependency on measurements of the kernel or initrd (simplifies kernel update).

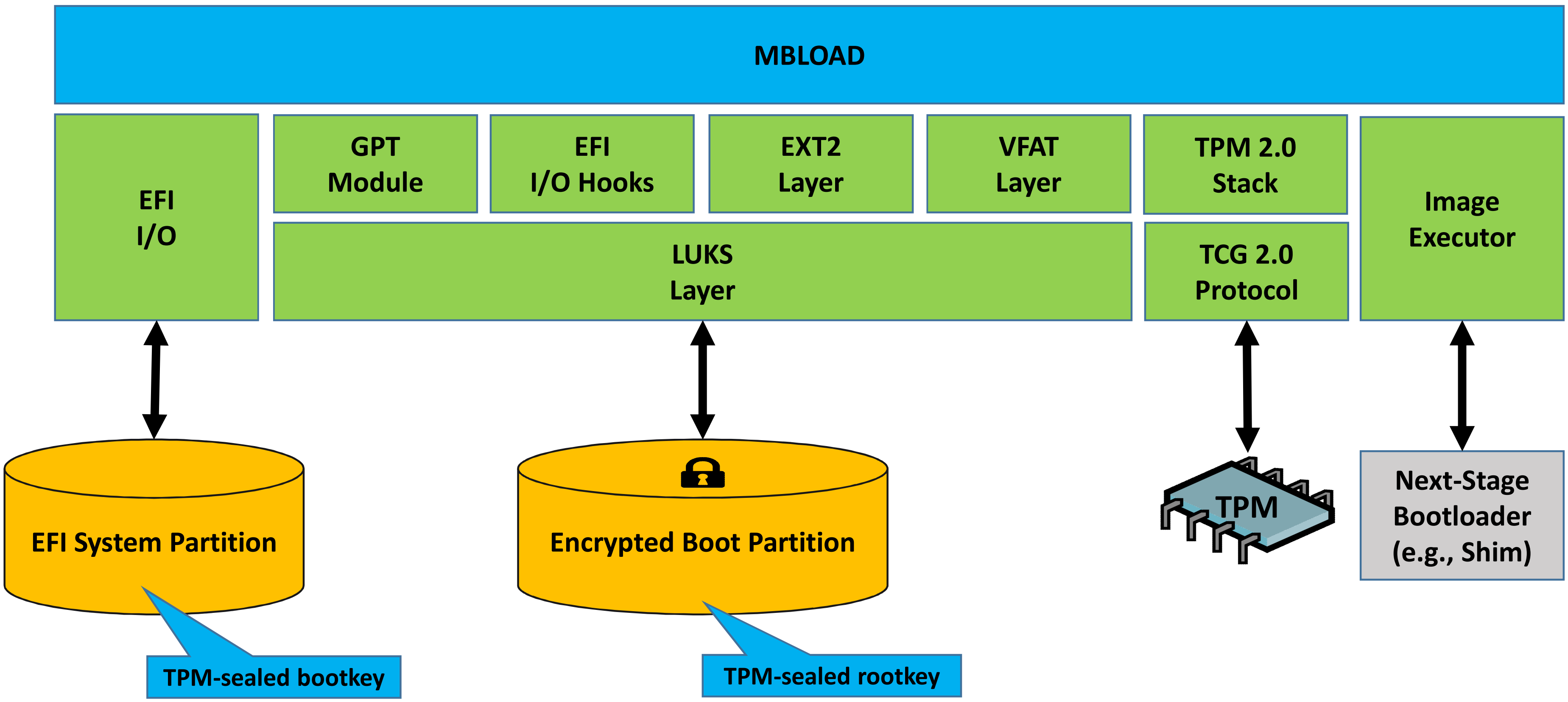
Why MBLOAD?

- MBLOAD is the root of trust for Hyper-V (special certificate).
- MBLOAD must protect Hyper-V as much as it must protect Linux.
- MBLOAD must refuse to run in non-Hyper-V environments.
- MBLOAD must leave the TPM PCRs in a state that cannot be used later to load and attack Windows.
- MBLOAD's basis of trust is encryption of the boot drive.

Alternative configurations

1. MBLOAD may execute GRUB2 directly.
2. MBLOAD may execute the kernel + initrd directly.
3. Considering allowing Shim and GRUB2 to load from the ESP (rather than encrypted boot partition) depending on future Shim/GRUB2 features.

MBLOAD Elements



Known Limitations

1. MBLOAD currently only supports EXT2 boot partitions.

Open-source licensing

- Preparing to open-source current work.
- Sources will be available on Github.
- The license will probably be MIT.