The Way of the Panda:

Getting Started with Xen

Lars Kurth

Community Manger, Xen Project
Chairman, Xen Project Advisory Board
Director, Open Source, Citrix

IRC lars_kurth

George Dunlap

Committer, Xen Project
Senior Software Engineer, Citrix

IRC gwd

Contributors

Andrew Cooper. Committer, Xen Project □ Roger Pau Monné, Maintainer, Xen Project □ Wei Liu, Committer, Xen Project

Session Goals

Virtualization Concepts

Overview of Xen’s Basic concepts and use-cases

- With exercises built in

How to get help from the community

A peek view into Xen’s more advanced features

Important Note: Usually, you will use Xen indirectly as part of a commercial product or part of a bigger SW stack, or have scripts to automate much of what is covered in this session. However, by following this session you will learn how Xen and virtualization works under the hood.

What is Xen and Xen Project?

Versatile Virtualization Platform

Designed to be a component in a SW stack
Ease of use for end-users **not** a design goal

Xen Hypervisor = “Engine”

Taken by integrators to build a product, service, …
Analogy: Xen integrators build a “Car”

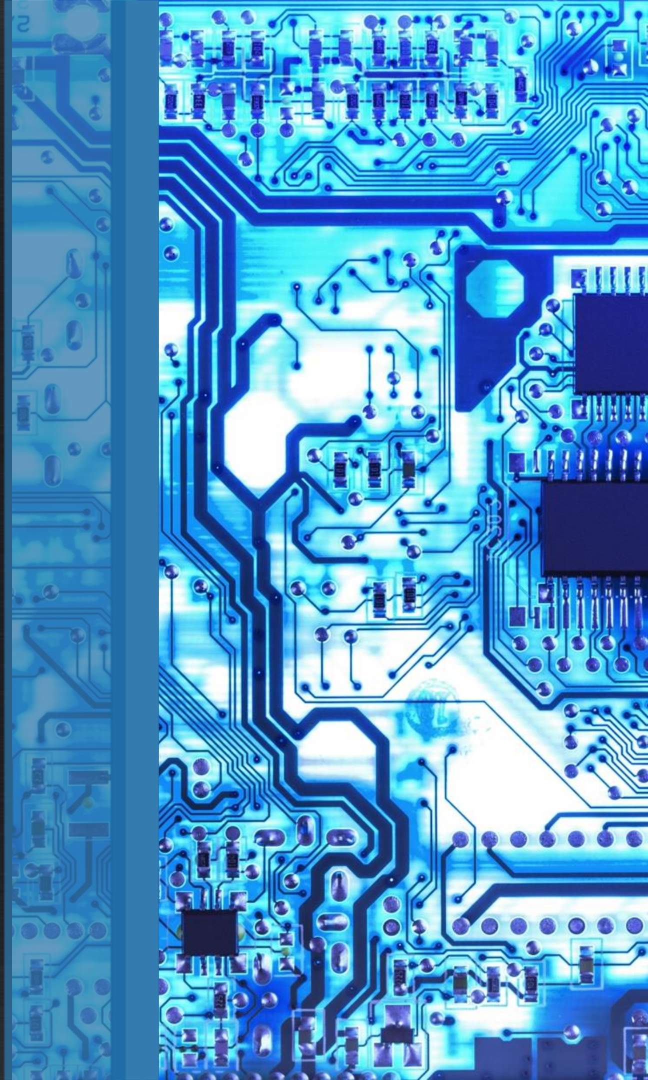
Examples at the end

Xen Project

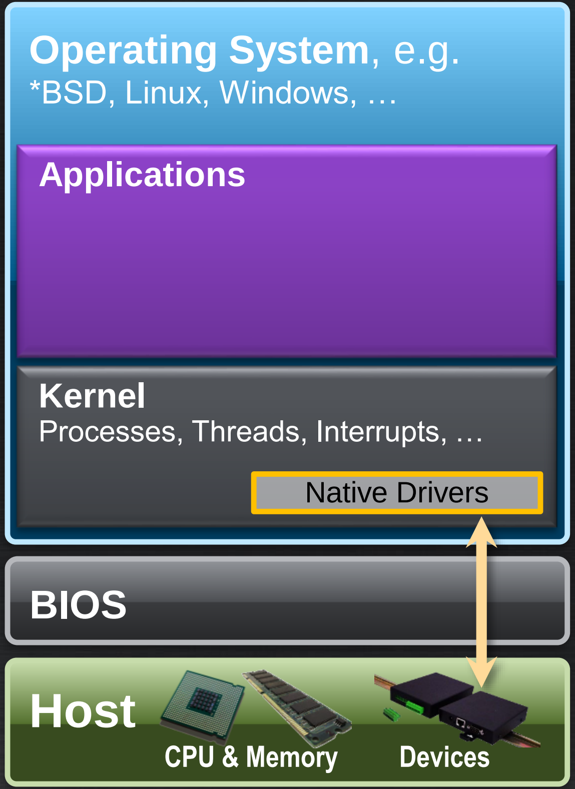
Development community with several sub projects that develop technologies related to Xen

- Hypervisor
- PV Drivers
- Unikernel related projects: MirageOS, Unikraft

Virtualization Concepts



Virtualization



Hypervisor

separates a computer’s operating system and applications from the underlying physical hardware → Virtual Machine

Creates an illusion that the Virtual Machine owns a set of CPUs and Memory memory within the host

This is done via CPU virtualization, where the Hypervisor

- Temporally manages CPU resources via a scheduler and takes control of interrupts and timers
- Spatially manages memory resources and ensures that a VM can only access the memory it is supposed to

I/O Virtualization

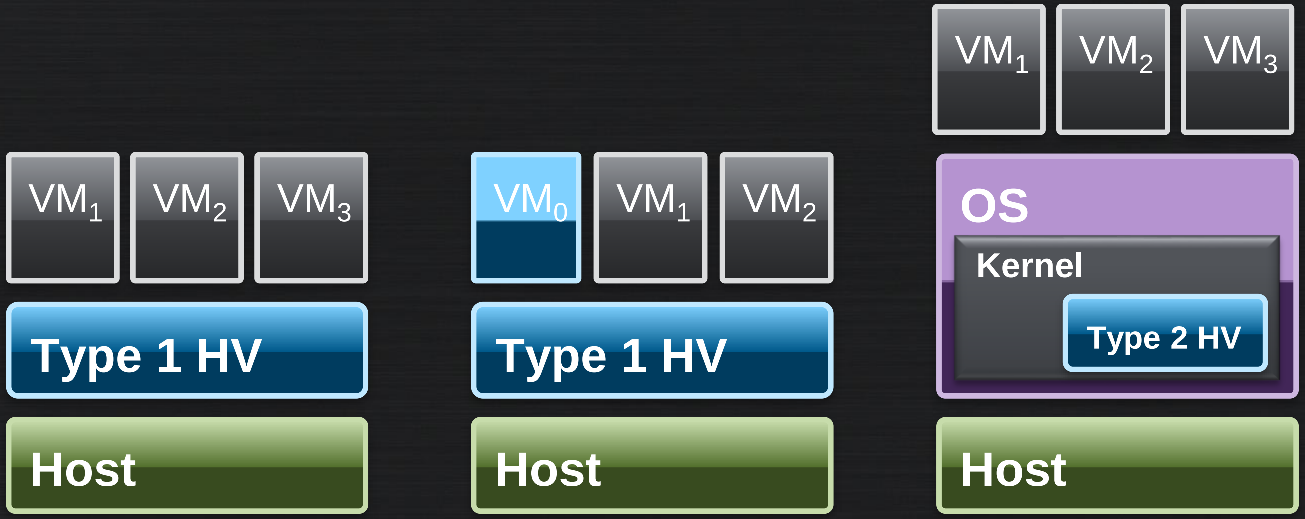
Multiplexes I/O devices across different virtual machines such that they can be shared across different VMs.

- There are a number of different ways of how to do this

Assign devices to specific Virtual Machines → Passthrough

Hypervisor Architectures

Very simplified



ESX Server

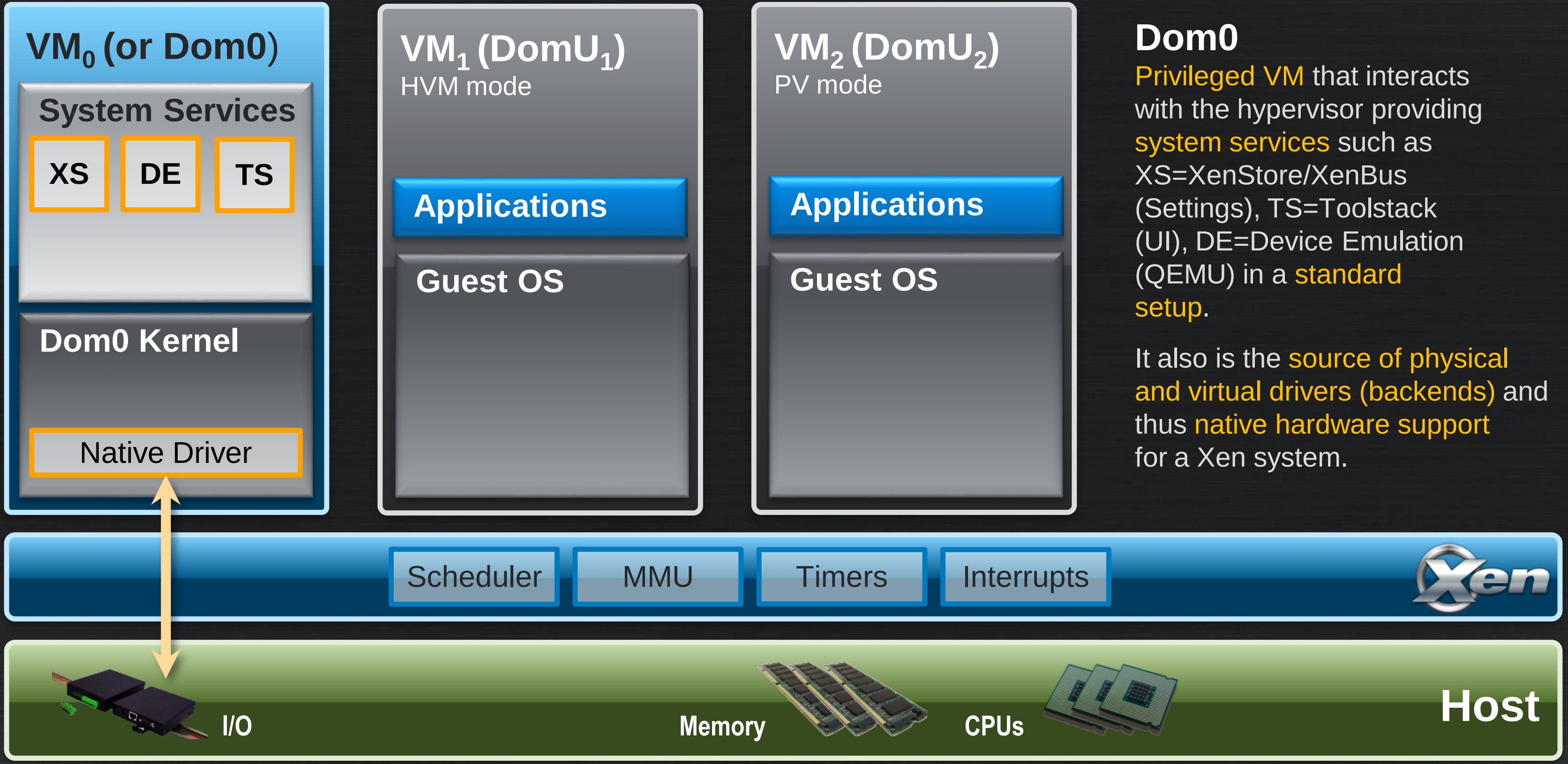
Xen
Hyper-V

KVM
VirtualBox

Xen, a type-1 Hypervisor with a twist

Introduction of key concepts

Xen Architecture

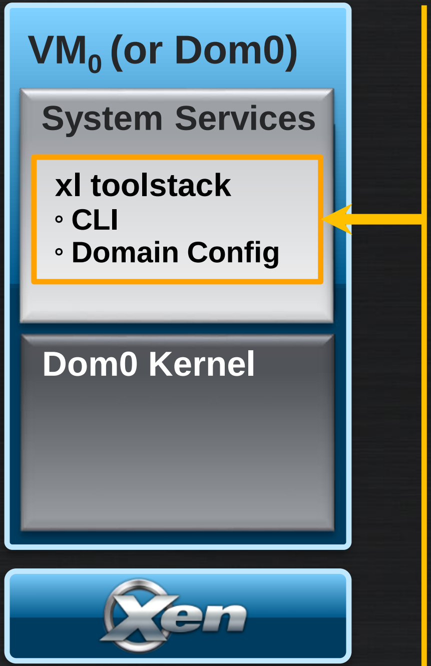


Dom0

Privileged VM that interacts with the hypervisor providing system services such as XS=XenStore/XenBus (Settings), TS=Toolstack (UI), DE=Device Emulation (QEMU) in a standard setup.

It also is the source of physical and virtual drivers (backends) and thus native hardware support for a Xen system.

xl & domain configuration files



VM$_0$ (or Dom0)

System Services

xl toolstack
- CLI
- Domain Config

Dom0 Kernel

xl is the built-in toolstack for Xen

– Virsh / virt-manager can also be used

– XAPI is the toolstack for XenServer and XCP-ng

xl can be used
https://xenbits.xen.org/docs/unstable/man/xl.1.html

– normally run as root in Dom0

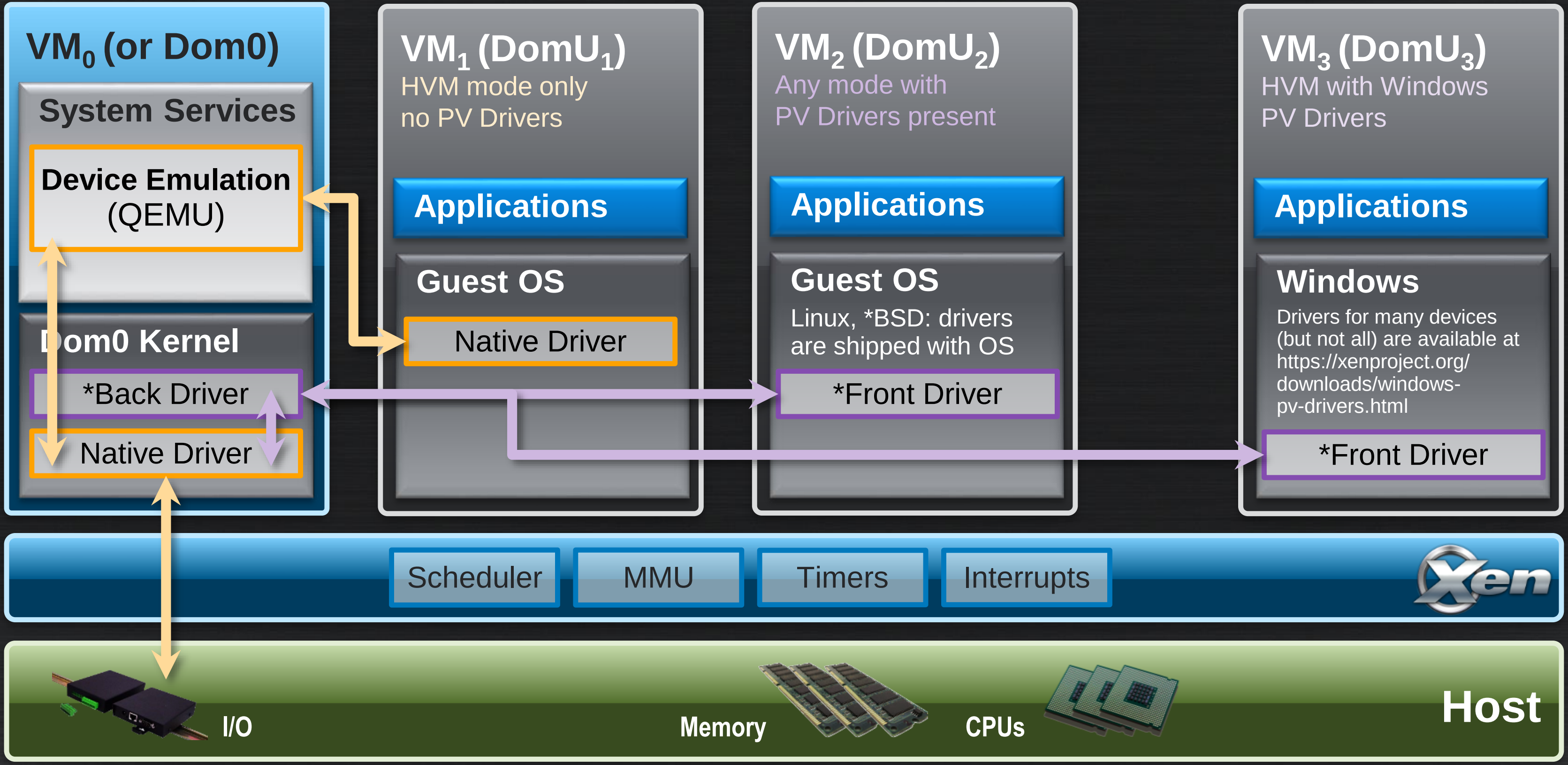
– to create, pause, and shutdown domains

– to list current domains, enable or pin VCPUs, and attach or detach virtual block devices

Domain configuration files (/etc/xen/<domain>.cfg)
https://xenbits.xen.org/docs/unstable/man/xl.cfg.5.html

– describe per domain/VM configuration in Dom0 filesystem

I/O Virtualization in Xen



VM$_0$ (or Dom0)

System Services

Device Emulation (QEMU)

Dom0 Kernel

*Back Driver

Native Driver

VM$_1$ (DomU$_1$)

HVM mode only
no PV Drivers

Applications

Guest OS

Native Driver

VM$_2$ (DomU$_2$)

Any mode with
PV Drivers present

Applications

Guest OS

Linux, *BSD: drivers are shipped with OS

*Front Driver

VM$_3$ (DomU$_3$)

HVM with Windows
PV Drivers

Applications

Windows

Drivers for many devices (but not all) are available at https://xenproject.org/downloads/windows-pv-drivers.html

*Front Driver

Scheduler

MMU

Timers

Interrupts

Xen

I/O

Memory

CPUs

Host

I/O Virtualization in Xen

PV Drivers

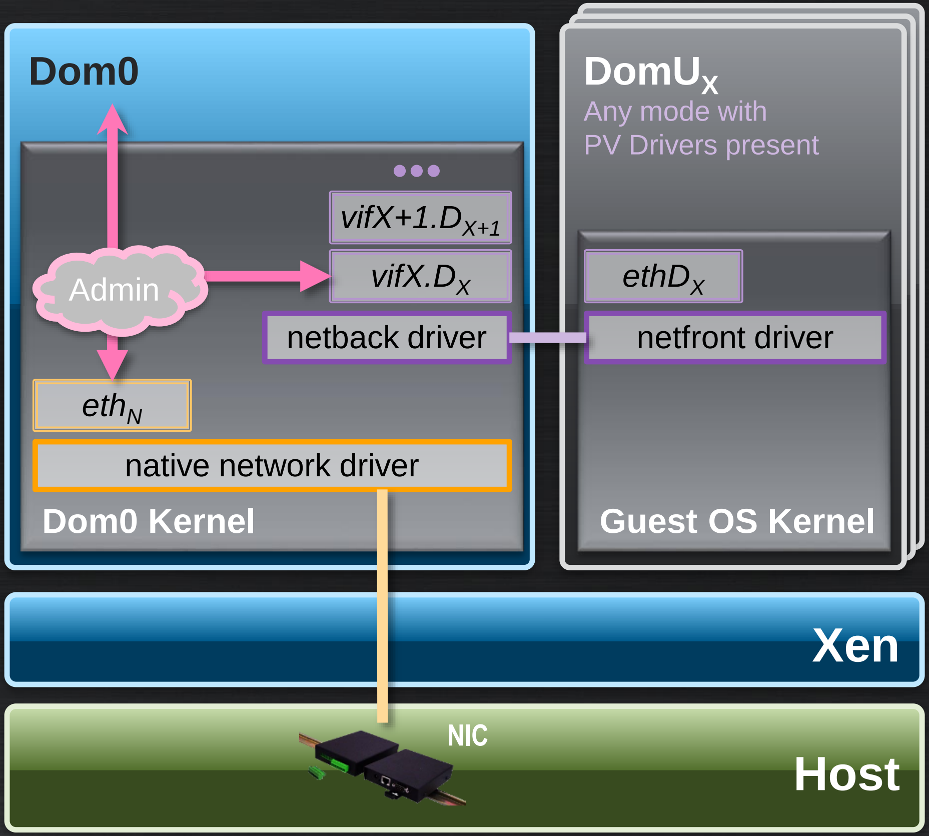
Originally developed for disk and network I/O
But there are a host of PV drivers for DRM, Touchscreen, Audio, … for non-server use of Xen

Device Emulation …

is normally only used during system bootstrap or installation
and for low-bandwidth devices

A few PV backends (e.g. support for QCOW2 images) can also run in userspace within QEMU

Networking in Xen



With **xl**, the host networking configuration is **not configured** by the toolstack

The host administrator needs to **setup an appropriate network configuration in** Dom0 using native Linux/BSD tools using a number of different networking styles

Post Xen Install File Locations

Xen follows FHS: www.pathname.com/fhs/pub/fhs-2.3.html

/etc/xen : scripts, config file examples, your config files

/var/log/xen : log files

/usr/lib64/xen/bin : xen binaries
/usr/lib64/xen/boot : xen firmware and boot related binaries

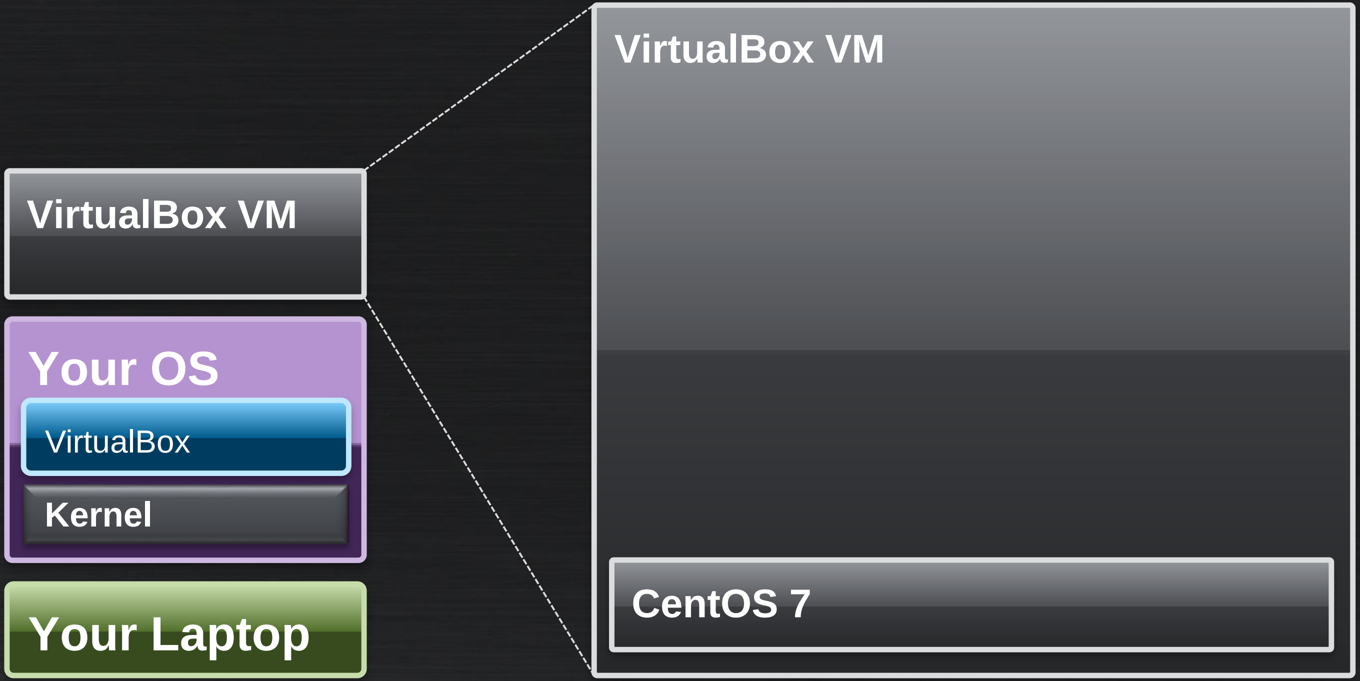
/boot : boot and install images

Exercises: Setup

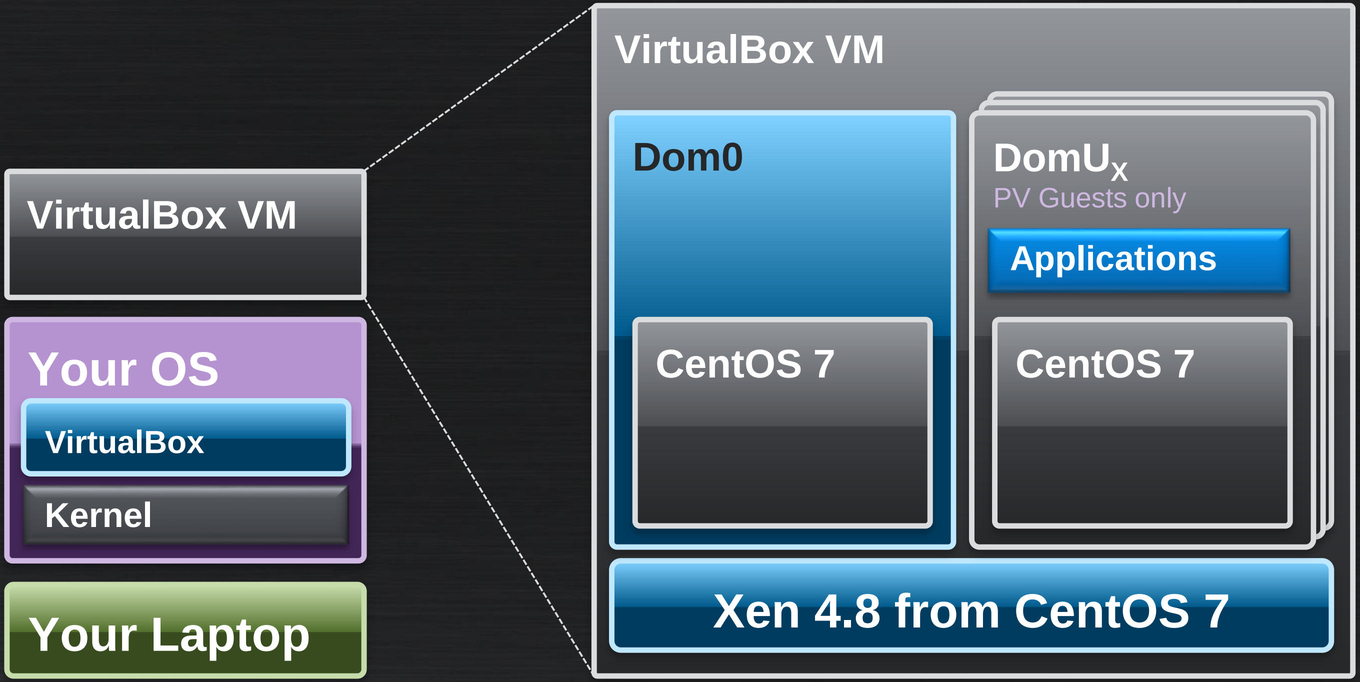
Section 1 of **session guide**

Duration VB Install : <2 minutes
Duration rest of Install : <6 minutes

Training Setup



Training Setup: Post Xen Install



Training Setup: Why the strange setup?

Xen takes over the entire host

Not really what you want after a training session

People have different environments

This makes it hard to run an effective training session

Can show almost everything

Xen PV guests can run fairly fast within any other Hypervisor

To use HVM or PVH you will need a dedicated host

Why Xen 4.8 from CentOS 7?

Has a lot of functionality up to Xen 4.10 backported

For other distros, you will need the equivalent of Xen 4.10

Let’s get started

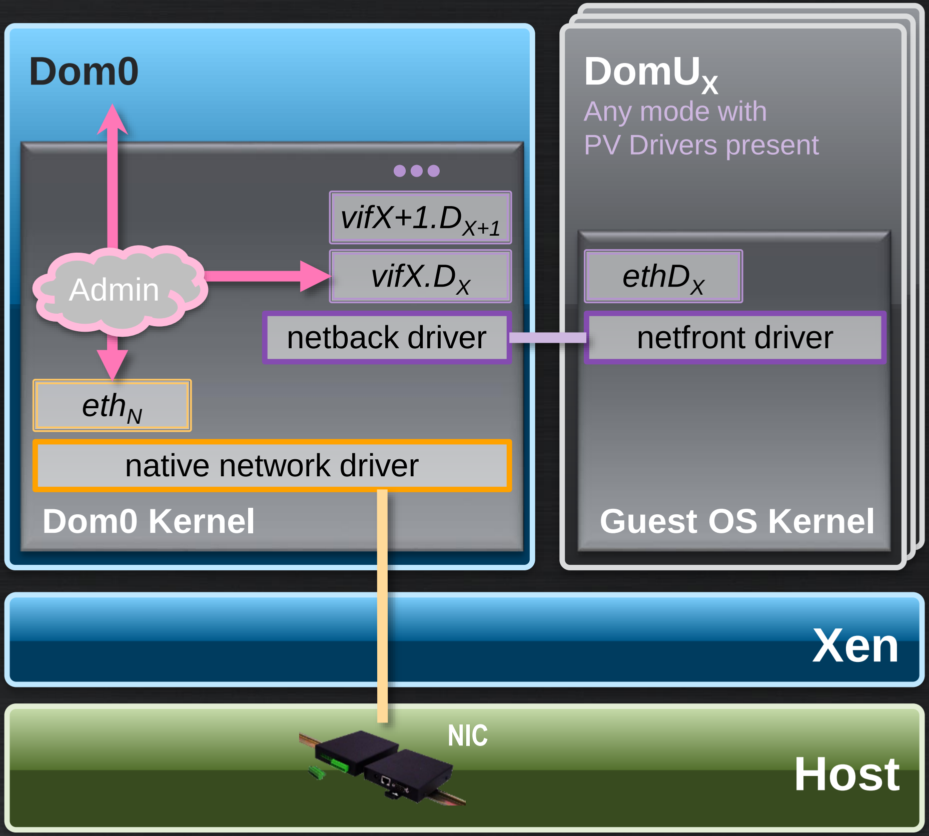
Install and configure Virtual Box
See section 1.1 of the **session guide**
Hopefully you have already done this

Import CentOS 7 Virtual Box Image
See section 1.2 of the **session guide**

Install Xen in Virtual Box VM
See section 1.3 of the **session guide**

Networking revisited,
Guest Types,
Storage Options,
Connecting to VMs &
Basic xl commands

Networking in Xen : Revisited



With **xl**, the host networking configuration is **not configured** by the toolstack

The host administrator needs to **setup an appropriate network configuration in** Dom0 using native Linux/BSD tools using one of the following networking styles:

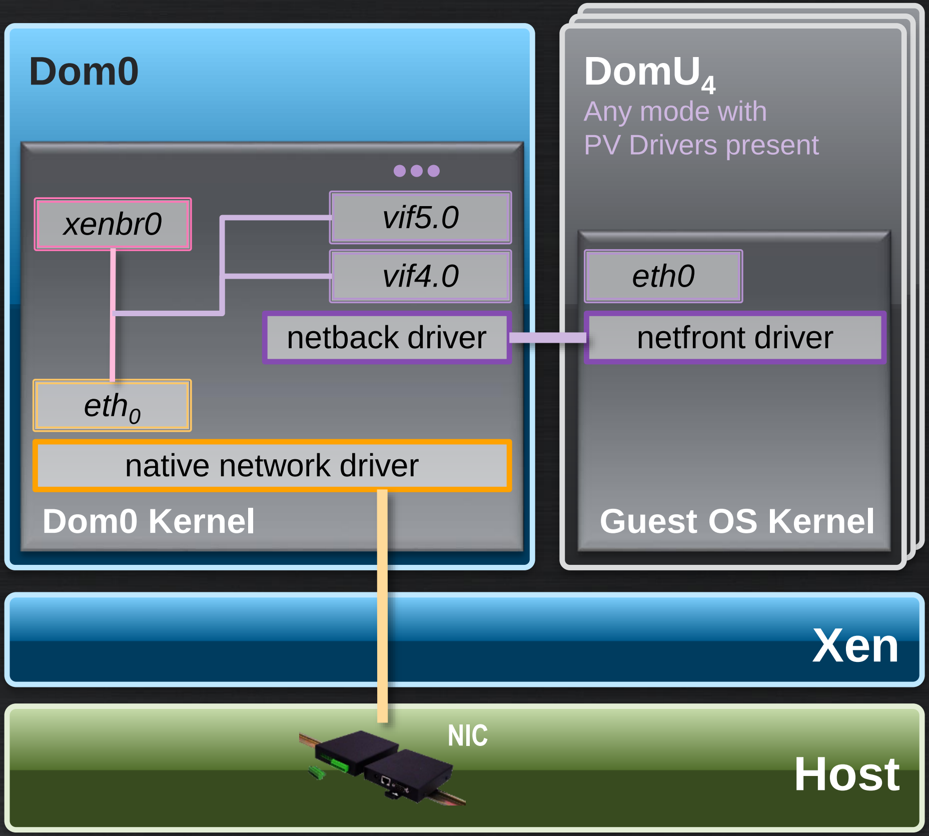
- Bridging (most common)
- Open vSwitch
- Routing
- NAT

Documentation @ wiki.xenproject.org/wiki/

- Network_Configuration_Examples_(Xen_4.1%2B)
  Dom0: Examples for enabling different networking styles in various distros
- Xen_Networking
  Xen configuration examples for different networking styles

  vif=[...]

Networking in Xen: Bridging



Step 1: install bridging software packages, if not present ✓

Step 2: set up a network bridge (xenbr0) in Dom0 ✓

Step 3: connect DomU's to network bridge

DomU$_4$

vif = ['mac=…, bridge=xenbr0']

DomU$_5$

vif = ['mac=… '] # xenbr0 is the default

…

Note on MAC addresses:
MAC addresses will be assigned automatically by xl, unless specified
➜ may change on host reboot

Evolution: Guest Types & Variants

2003

PV

Requires no HW support

But requires PV support in guest operating systems.

From 2011 (Linux 3.0) linux supports Xen PV out of the box.

2005/6

HVM

Requires Intel VT-x or AMD SVM

2010 to 16

HVM Optimizations

Changes to HVM: instead of Device Emulation, use HW acceleration when available (e.g. Local APIC and Posted Interrupts).

On PV capable hosts and guests use PV extension where faster, including on Windows (marketing term: PVHVM)

2013

Xen/Arm

Added Arm32 and later 64 support

Re-think the historical split between PV / HVM modes
➜ one virtualization mode on Arm

2017 to now

PVH (lightweight HVM)

Re-architecting of HVM to avoid use of QEMU.

Goals: Windows guests without QEMU, reduce code size, increase security, enable PVH Dom0.

Requires PVH support in guest OSes.

Backwards compatibility mode for PV ➜ capability to build an HVM only version of Xen

Evolution: Paravirtualization (PV)

Virtualization technique called ring de-privileging developed in the late 90s.

Designed by:

- XenoServer research project at Cambridge University
- Intel
- Microsoft labs

x86 instructions behave differently in kernel or user mode: options for virtualization were full software emulation or binary translation.

- Design a new interface for virtualization
- Allow guests to collaborate in virtualization
- Provide new interfaces for virtualized guests that allow to reduce the overhead of virtualization

The result of this work is what we know today as paravirtualization, with Linux, *BSD and Windows implementing some or all PV interfaces.

Evolution: Full Virtualization (HVM)

With the introduction of hardware virtualization extensions Xen is able to run unmodified guests

- This requires emulated devices, which are handled by Qemu
- Makes use of nested page tables when available
- Allows to use PV interfaces if guest has support for them

Over time, HVM guests have been changed to automatically…

- use additional Hardware Acceleration support, such as Local APIC and Posted Interrupts, if available
- make use of guest PV interfaces where they are faster (this capability has been dubbed PVHVM or PV-on-HVM for marketing reasons)

Evolution: PVH (or Lightweight HVM)

Combine the best of PV and HVM mode

- Next-generation paravirtualization mode
- Takes advantage of hardware virtualization support
- No need for emulated BIOS or emulated devices
- Lower performance overhead than PV
- Lower memory overhead than HVM
- More secure than either PV or HVM mode

More Information:

- https://www.slideshare.net/xen_com_mgr/lcc18-xen-project-after-15-years-whats-next-george-dunlap-citrix
- https://www.youtube.com/watch?v=10KsJ1UxUMY

Guest Types: PV vs. HVM vs. PVH

PV mode: type="pv"

Primarily of use for legacy HW and legacy guest images
And in special scenarios, e.g. special guest types, special workloads (e.g. Unikernels), running Xen within another hypervisor without using nested virtualization, as container host, guest limits (more PV guests than HVM guests), …

HVM mode: type="hvm"

Typically the best performing option on for Linux, Windows, *BSDs
Adapts to hardware and software environment for performance
Guests look exactly like a "PC or Server"

PVH mode: type="pvh"

Lightweight version of HVM → promise of better performance and security
Needs Linux ≥ 4.15 and FreeBSD ≥ 12 (later in 2018)
Guest looks like a simpler abstraction of a "PC or Server"
Relatively new (Xen 4.10)

Storage Options & Disk Specifications

DomU$_x$

disk = […]

disk = ['format=…,	Disk format, e.g. raw, qcow, qcow2, vhd, qed
vdev=…,	Virtual device name as seen by the guest See xen-vbd-interface(7) man page
access=…,	Read write access: r, w
devtype=cdrom,	If you want to use an ISO
target=…, ']\	Block device or image file path (must be last)

Local Host

Use LVM to carve up your physical disk into multiple block devices

Store guest disk images as files on a local filesystem

Before installing a Dom0, consider where you are intending to store the guest OS disk images ➜ you have to manage the disk space available and partition the disk accordingly using LVM volumes

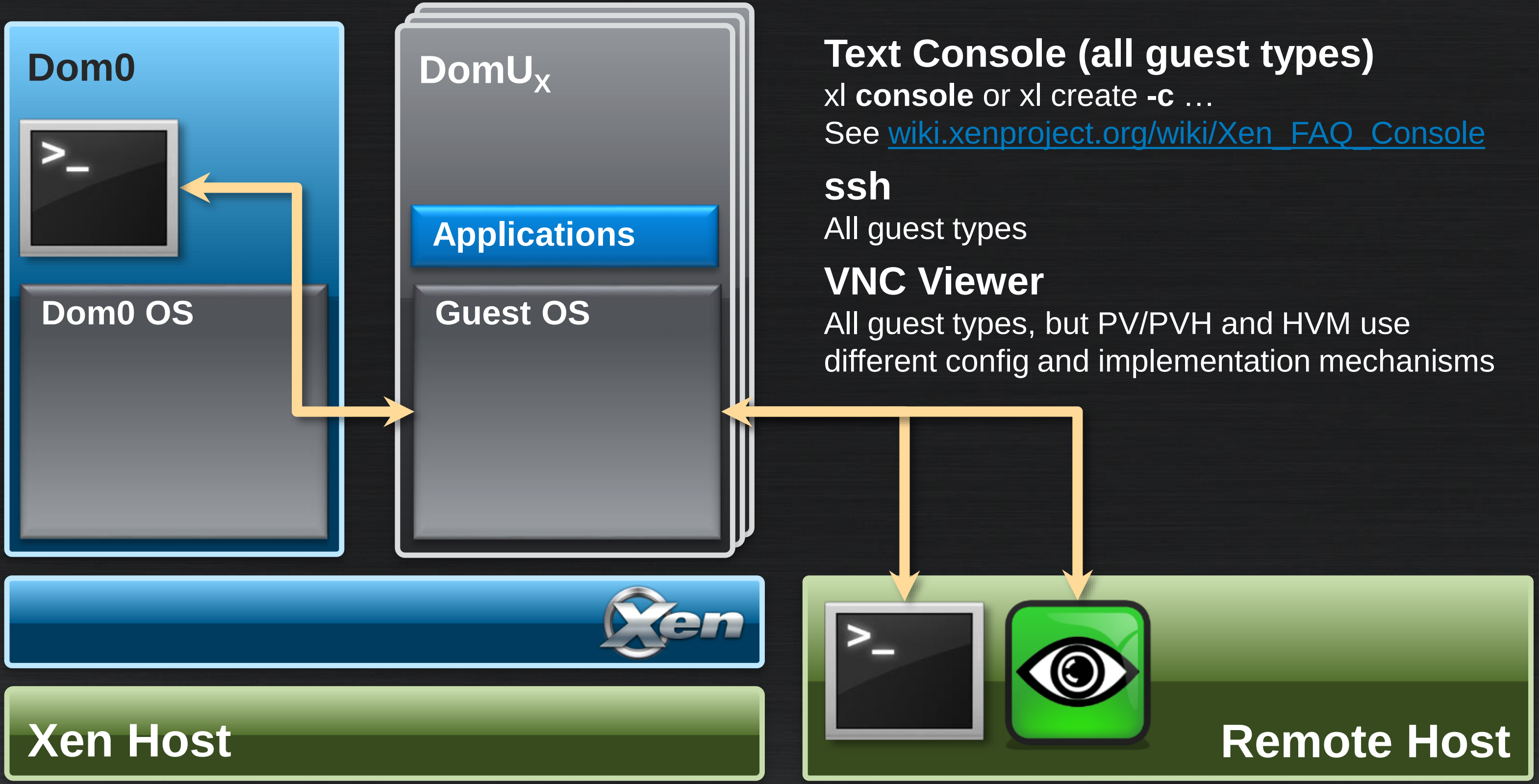
For the exercises we will store these in the root filesystem of the Dom0 guest OS

Remote Storage

For example RBD, NBD, NFS, DRDB or iSCSI

A remote storage set-up is the normal set-up when Xen is used at scale, either on premise or in a cloud computing set-up.

Connecting to a VM



Text Console (all guest types)
xl **console** or xl create **-c** …
See wiki.xenproject.org/wiki/Xen_FAQ_Console

ssh
All guest types

VNC Viewer
All guest types, but PV/PVH and HVM use different config and implementation mechanisms

Basic xl commands

VM control

xl **create** [*configfile*] [*OPTIONS*] | **shutdown** [*OPTIONS*] *-a|domain-id*

destroy [*OPTIONS*] *domain-id*

xl **pause** *domain-id* | **unpause** *domain-id*

Information

xl **info** [*OPTIONS*]

xl **list** [*OPTIONS*] [*domain-id* ...]

xl **top**

xl **uptime**

Debug

xl **dmesg** [*OPTIONS*]

xl **-v** … logs from /var/log/xen/xl-${DOMNAME}.log, /var/log/xen/qemu-dm-${DOMNAME}.log, …

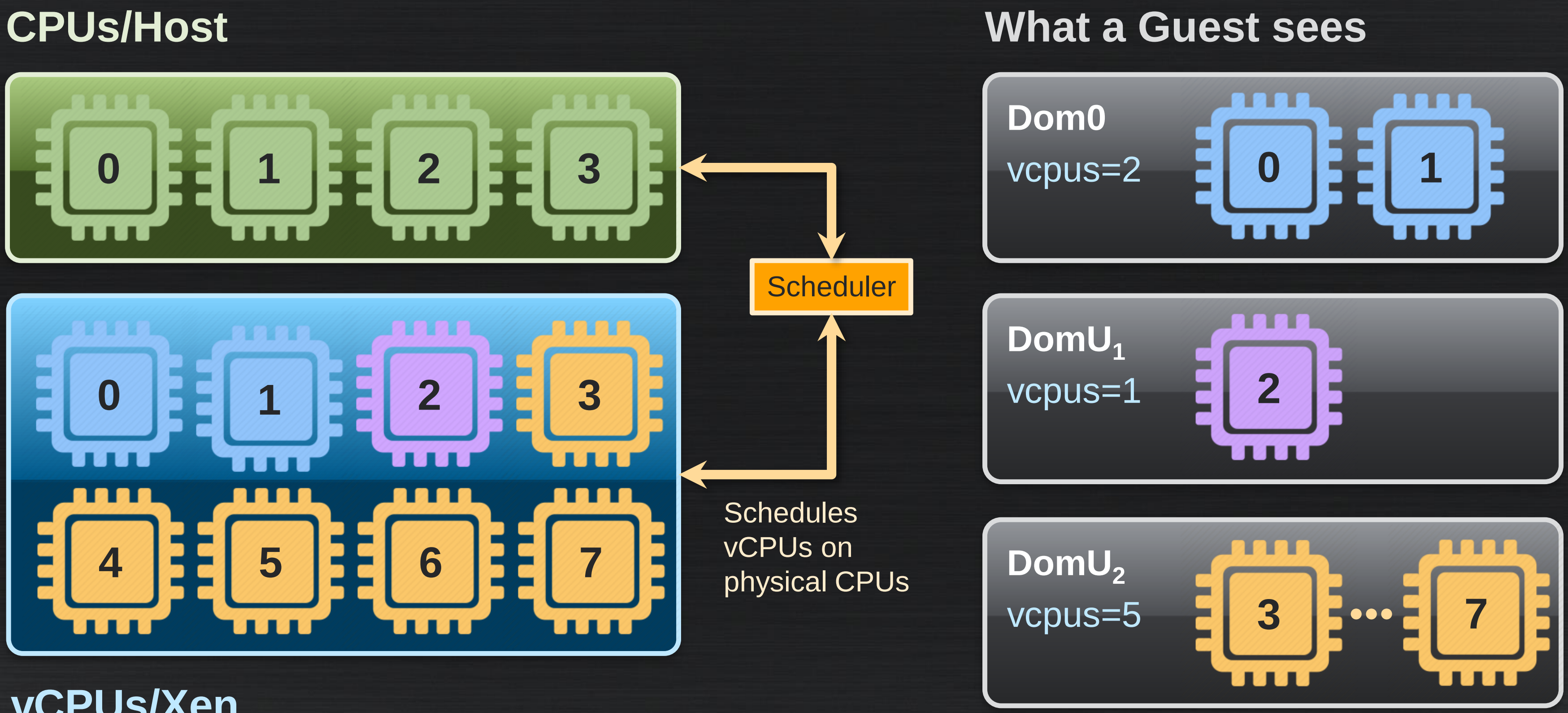
Exercises: Setup

Section 2 of **session guide**

Duration: <10 minutes

Guest Types, Storage Options, Connecting to VMs & Basic xl commands

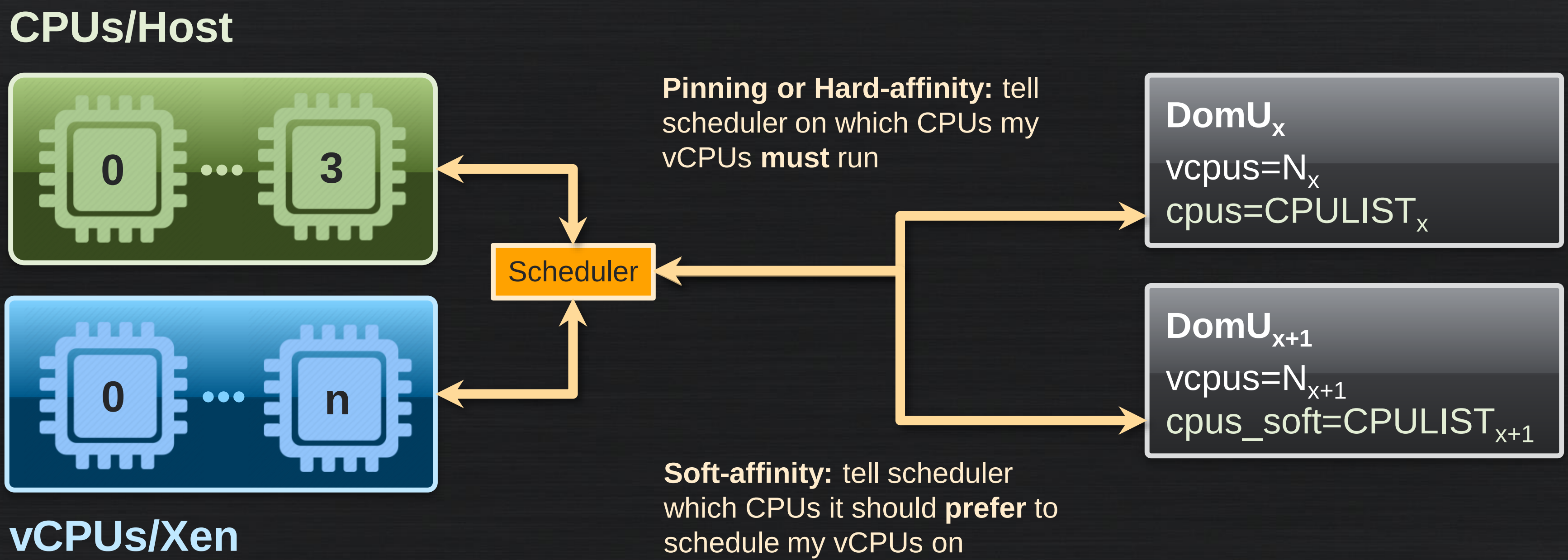
vCPUs, CPUs and Guests



vCPUs/Xen

Created on demand based on user supplied information

CPUs: slightly more Advanced Topics



CPUs/Host

0 ... 3

Scheduler

0 ... n

vCPUs/Xen

DomU$_x$

vcpus=N$_x$

cpus=CPULIST$_x$

DomU$_{x+1}$

vcpus=N$_{x+1}$

cpus_soft=CPULIST$_{x+1}$

Pinning or Hard-affinity: tell scheduler on which CPUs my vCPUs **must** run

Soft-affinity: tell scheduler which CPUs it should **prefer** to schedule my vCPUs on

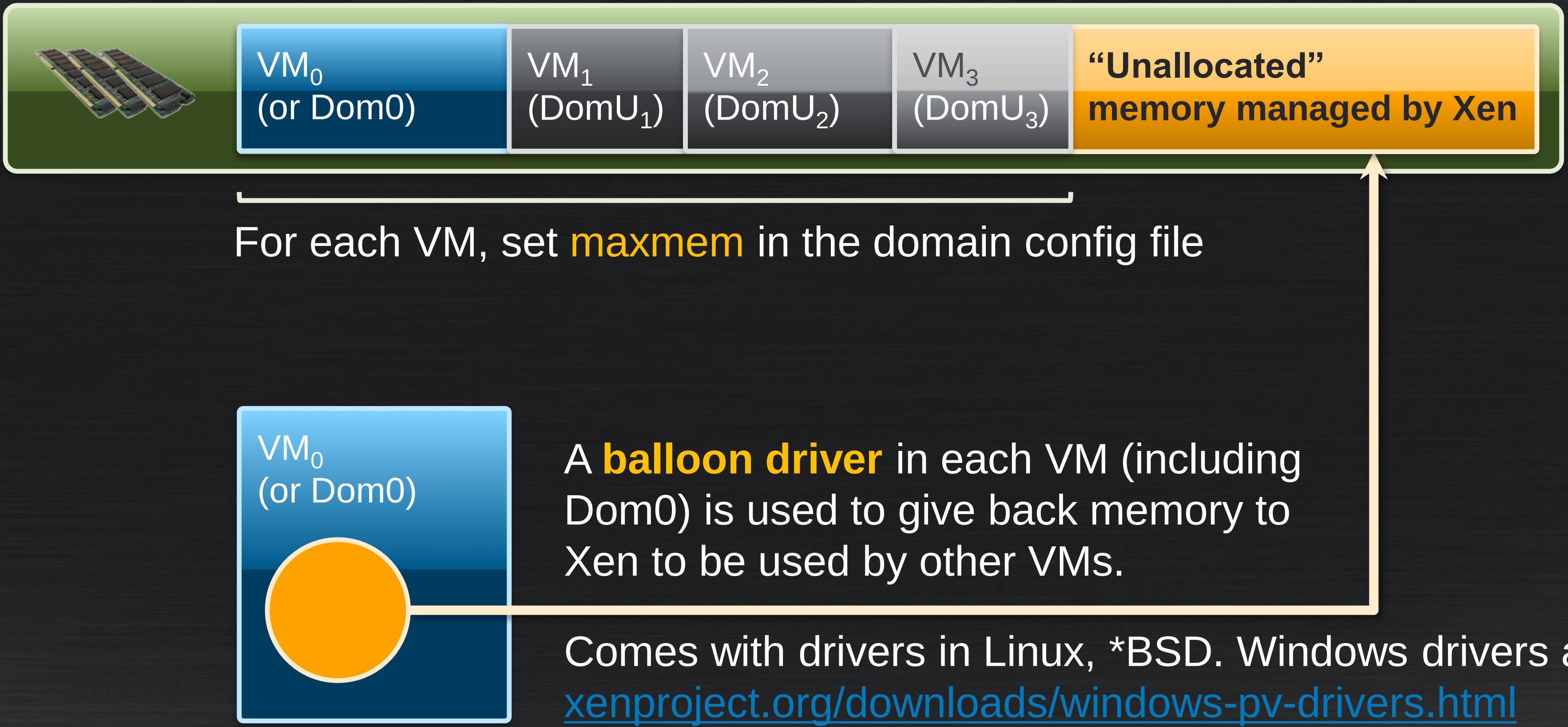
Related xl commands:

vcpu-list [*domain-id*]

vcpu-pin [*-f*|*--force*] *domain-id vcpu cpus hard cpus soft*

Also see CPUPOOLS

Xen, Memory and Ballooning



For each VM, set maxmem in the domain config file

A **balloon driver** in each VM (including Dom0) is used to give back memory to Xen to be used by other VMs.

Comes with drivers in Linux, *BSD. Windows drivers at xenproject.org/downloads/windows-pv-drivers.html

Xen, Memory and Ballooning

Config file	**xl … *domain-id mem***
maxmem=MBYTES	
memory=MBYTES	**mem-set** … sets the balloon size

Important Notes:

From within the guest, the balloon is reported as used memory
If you have a guest that started at 2GiB and you ballooned down to 1GiB, it will look like there's a memory hog driver that's grabbing 1GiB of RAM.

OS’es have to use memory to track memory even if it's ballooned out
Setting maxmem=16GiB memory=1GiB you'll have a lot less free memory than maxmem=2GiB memory=1GiB

Changing vCPUs and memory and of a guest

Section 3 of **session guide**

Duration: <15 minutes

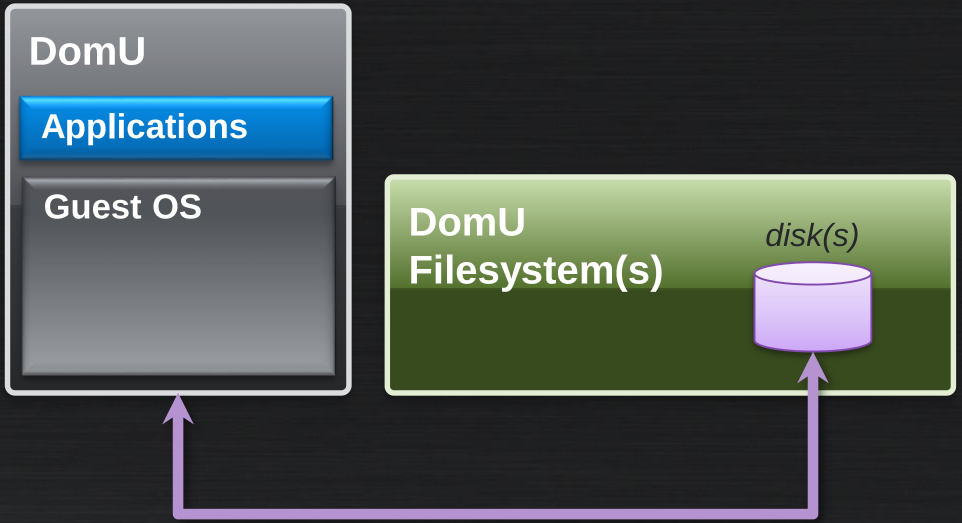
Save, Restore, Migrate

Save/Restore are building blocks that enable moving VMs from one host to another without downtime

Maintenance, Replacing Hosts, Building Block for High Availability/Disaster Recovery, …

Shutdown & Restart

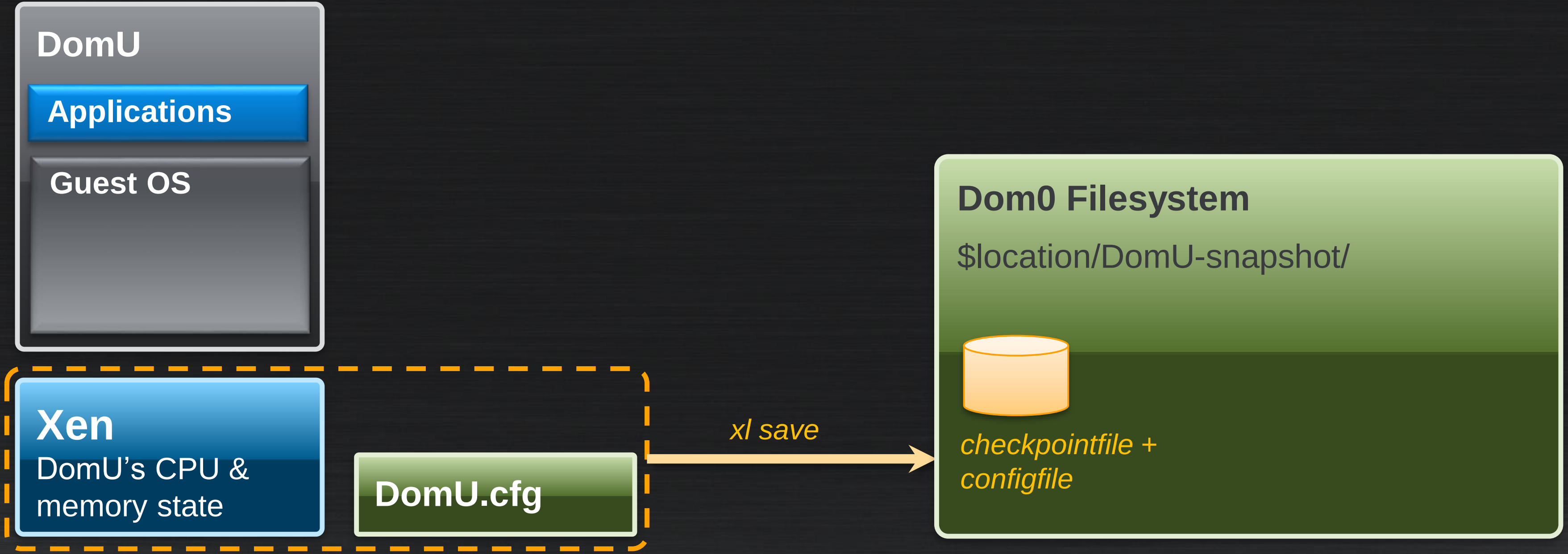
xl **shutdown|create** domain-id



When shutdown, copying guest disks and config files allows you to clone a VM (or move them to another host)

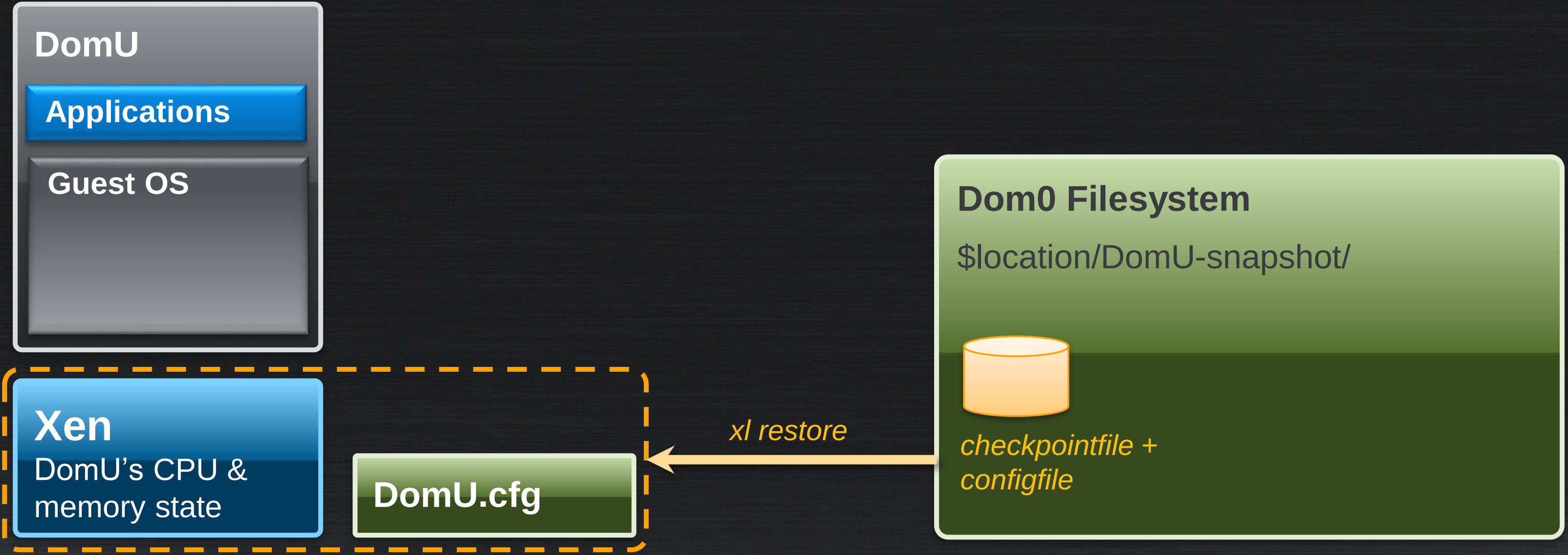
Save & Restore

xl **save** [*OPTIONS*] domain-id checkpointfile [configfile]



Save & Restore

xl **restore** [*OPTIONS*] [*configfile*] *checkpointfile*



Save and Restore of a guest

Section 4 of **session guide**

Duration: <5 minutes

Migrate

xl **migrate** [*OPTIONS*] *domain-id host*

Migrate a VM from one host to another (uses save/restore as building blocks).

For this to work, you need

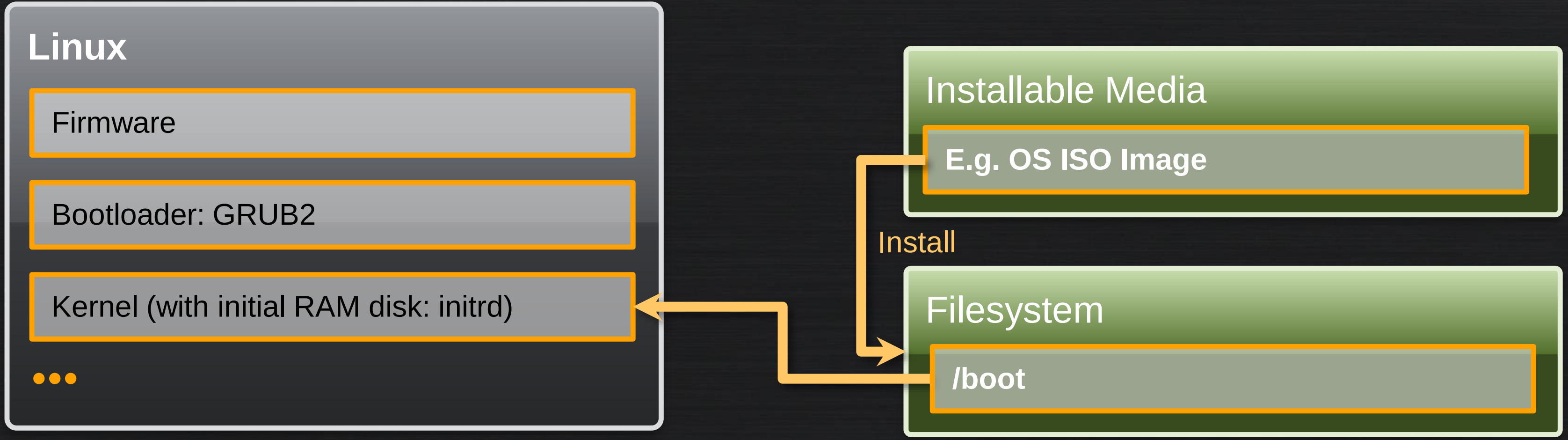
- Shared network storage between the two hosts
- Identical host network setups, ssh keys for the root users, …
- Compatible host models

  A VM can only be migrated safely from one host to another if both hosts offer the set of CPU features which the VM expects. If this is not the case, CPU features may appear or disappear as the VM is migrated, causing it to crash.
- Compatible Xen versions

  A VM build on an older Xen version can be migrated to a newer Xen version, but not vice versa
  Restricted by the Xen compatibility policy

Bootloaders in Xen

Boot & Install Process



Linux

Firmware

Bootloader: GRUB2

Kernel (with initial RAM disk: initrd)

•••

Installable Media

E.g. OS ISO Image

Install

Filesystem

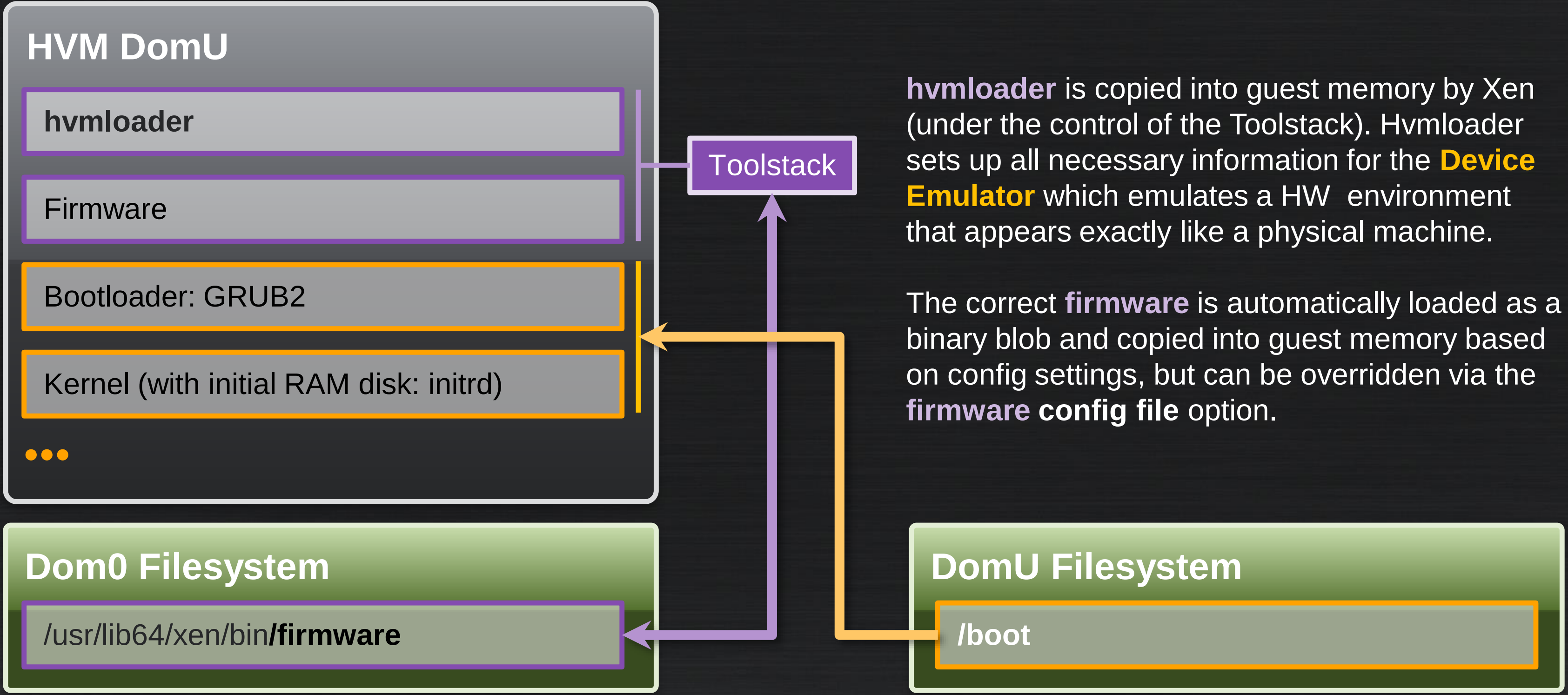
/boot

For reference:

Linux: more information see https://opensource.com/article/17/2/linux-boot-and-startup

Other operating systems follow a similar pattern
They diverge after the Bootloader step

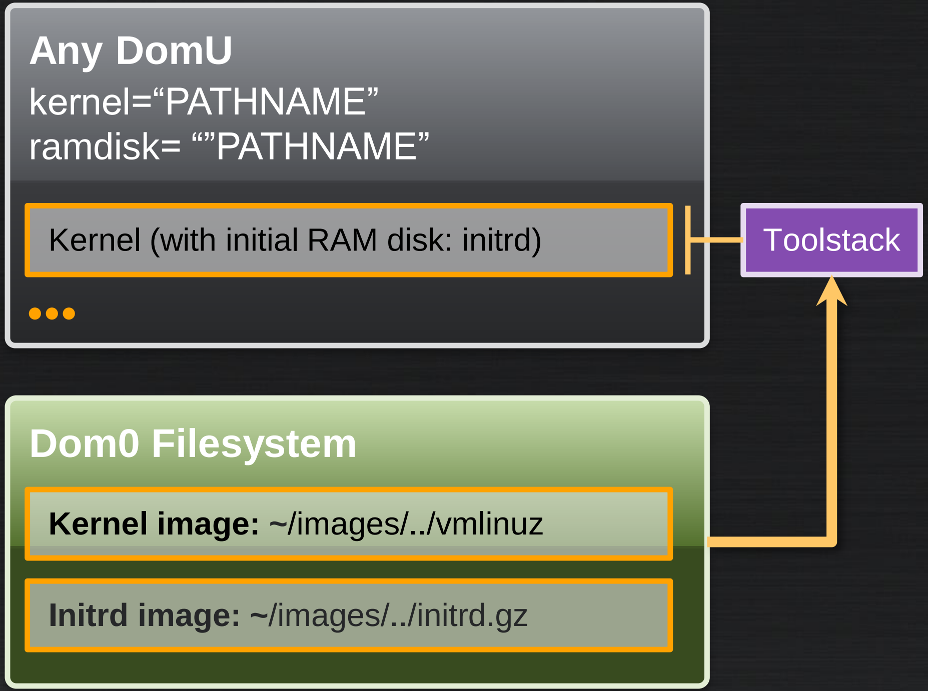
Xen: HVM Guest Boot Process



hvmloader is copied into guest memory by Xen (under the control of the Toolstack). Hvmloader sets up all necessary information for the **Device Emulator** which emulates a HW environment that appears exactly like a physical machine.

The correct **firmware** is automatically loaded as a binary blob and copied into guest memory based on config settings, but can be overridden via the **firmware config file** option.

Xen: Direct Kernel Boot



Any DomU

kernel="PATHNAME"
ramdisk= ""PATHNAME"

Kernel (with initial RAM disk: initrd)

•••

Toolstack

Dom0 Filesystem

Kernel image: ~/images/../vmlinuz

Initrd image: ~/images/../initrd.gz

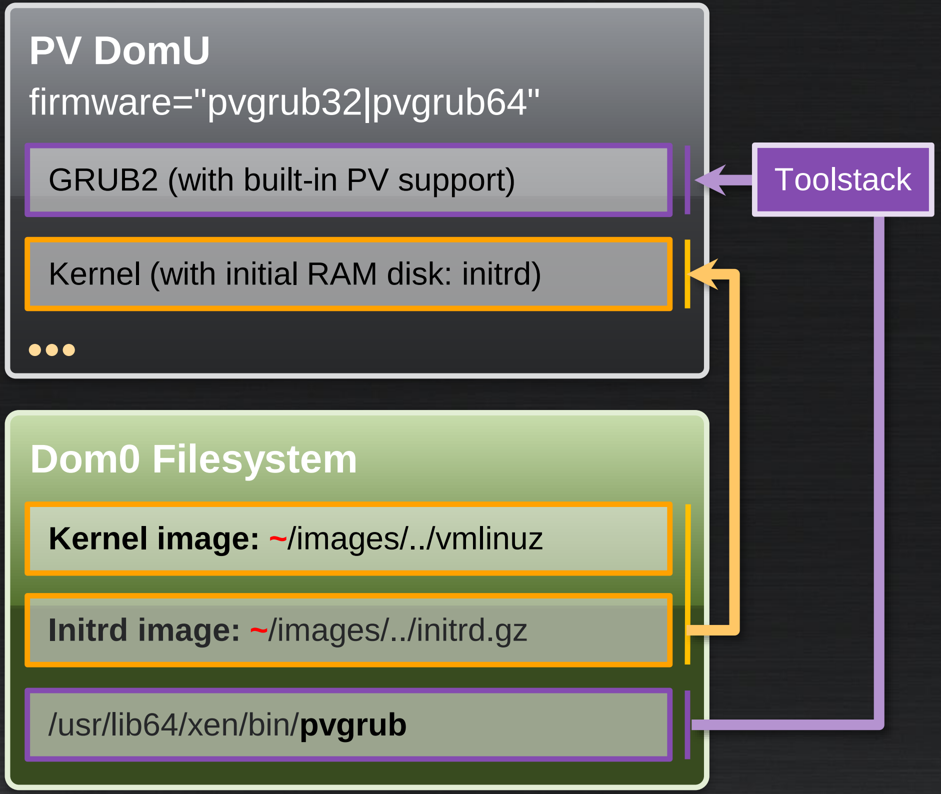
Works for all guest types

Non standard way of installing/booting

Need to be a **host admins** to configure (need access to Dom0).

Useful for netboot, see wiki.xenproject.org/wiki/Xenpvnetboot

Xen: PVGrub


PV DomU
firmware="pvgrub32|pvgrub64"
GRUB2 (with built-in PV support)
Kernel (with initial RAM disk: initrd)
•••
Toolstack
Dom0 Filesystem
Kernel image: ~/images/../vmlinuz
Initrd image: ~/images/../initrd.gz
/usr/lib64/xen/bin/**pvgrub**

Works for PV guest types

Non standard way of installing/booting, with a standard bootloader UI.

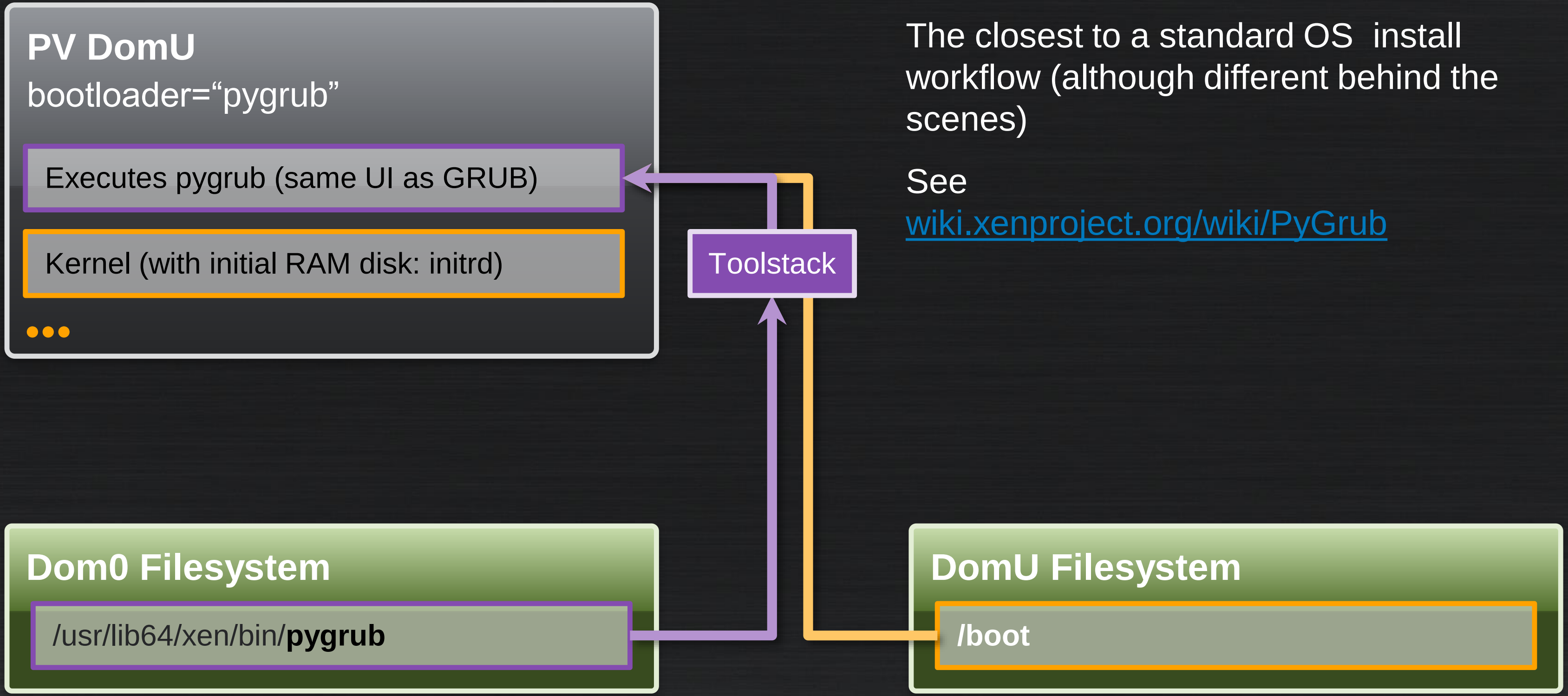
Allows **host admins** to configure what guests and kernel versions a **guest admin** can install.

Also used for PXE booting

Requires a PV capable GRUB2 (you may need to build from source or install an appropriate distro package)

Also see
wiki.xenproject.org/wiki/PvGrub2

Xen: PyGrub


PV DomU
bootloader="pygrub"
Executes pygrub (same UI as GRUB)
Kernel (with initial RAM disk: initrd)
•••
Toolstack
Dom0 Filesystem
/usr/lib64/xen/bin/**pygrub**
DomU Filesystem
/boot

The closest to a standard OS install workflow (although different behind the scenes)

See
wiki.xenproject.org/wiki/PyGrub

Xen: Boot Options – Discussion

In most real-life scenarios you will use HVM guests

- Guest install workflow as on a native system
- That does not scale across a large number of hosts

In Xen based products install complexity is usually hidden

- Via templates, pre-baked guest images and other means

Exercises: will use PV with PyGrub

- Using a prepared VirtualBox image that contains Dom0 and Guest OS
- Avoid downloads of guest distros

Summary: What’s in Guest Config?

```
# Guest name and type, Memory Size and VCPUs
name = “myguestname”
type = “TYPE”
memory = MMM
vcpus = VVV

# Boot related information, unless type='hvm’ … one of the following
# Netboot/Direct Kernel Boot/PV GRUB
kernel = "/…/vmlinuz”
ramdisk = "/…/initrd.gz”
extra = …
# To use PVGrub (if installed)
firmware="pvgrub32|pvgrub64

# Boot from disk
bootloader=“pygrub”

# Disk specifications
disk = [' ']

# Network specifications
vif = [' ']
```

Create Guests from Scratch

Section 5 of **session guide**

Duration: <10 minutes

Exercise Summary: Key Steps

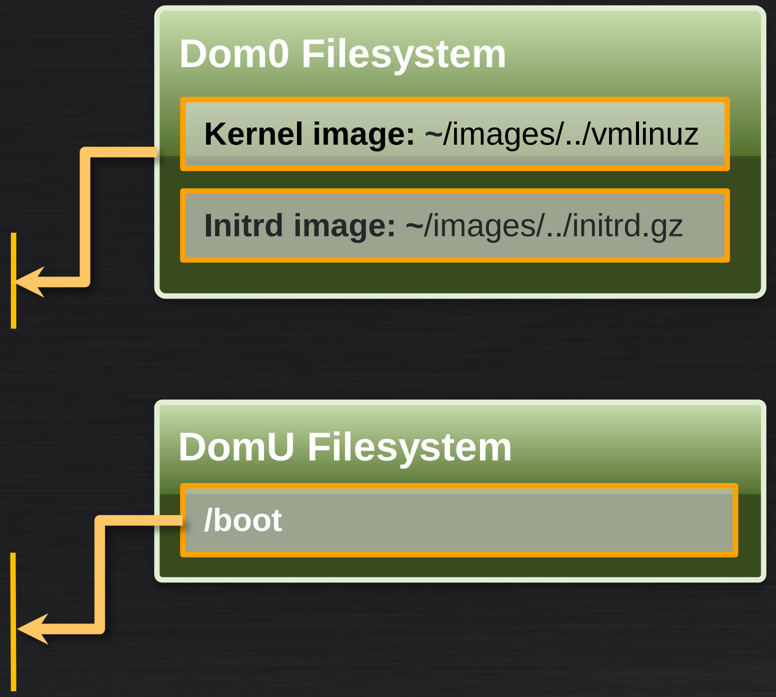
Step 1: Get **vmlinuz & initrd.gz**
In this case from Debian

Step 2: Create DomU filesystem

Step 2: Set up config for **Direct Kernel Boot** □ Start guest

Step 3: Perform **Install**
Fix any loose ends that the installer didn’t handle

Step 4: Change config to use **pygrub** □ Shut down and restart guest



Dom0 Filesystem

Kernel image: ~/images/../vmlinuz

Initrd image: ~/images/../initrd.gz

DomU Filesystem

/boot

Getting Help from the Xen Community

Getting Help

Channels

IRC@freenode: #xen … xenproject.org/help/irc.html
Lists: xen-users@lists.xenproject.org … lists.xenproject.org
FAQs: wiki.xenproject.org/wiki/Category:FAQ

Preparing information

Xen: Log files (/etc/log/xen), xl dmesg output, xl info output
Dom0: OS Info, System Configs (networking, …), dmesg output
DomU: OS Info, xl configuration files

Netiquette

wiki.xenproject.org/wiki/Xen_Users_Netiquette
wiki.xenproject.org/wiki/Reporting_Bugs_against_Xen_Project

Advanced Xen Features which may be worth looking at

Security

Live Patching, Virtual Machine Introspection and Vulnerability Management

A Primer and Practical Guide – Lars Kurth

Presentation: goo.gl/MLMu5b

Demo Videos: goo.gl/wuQLPh & goo.gl/dEGfDS

Virtual Machine Introspection

@ 31c3 - Tamas K Lengyel, Thomas Kittel

Presentation: goo.gl/khq92r

Video: www.youtube.com/watch?v=MhEIyzfLa6U

Current Hot Topics

Xen on x86, 15 years later
Recent development, future direction - George Dunlap
Presentation: goo.gl/8Djm7w
Video: www.youtube.com/watch?v=10KsJ1UxUMY

Speculation and response
Spectre, Meltdown, XPTI, and Panopticon - George Dunlap
Presentation: goo.gl/xnoj8J
Video: www.youtube.com/watch?v=36jta61XTw8

Embedded, Automotive, …

Securing embedded Systems using Virtualization
@ FOSDEM18 - Lars Kurth
Presentation: goo.gl/dEGfDS
Video: goo.gl/V6DA6P

Xen and the Art of Embedded Systems Virtualization
@ ELC18 - Stefano Stabellini
Presentation: goo.gl/WdbtzN
Video: www.youtube.com/watch?v=GYb-Qn3KAUM

Unikernels / Unikraft

Unleashing the Power of Unikernels with Unikraft
@ XPDDS18 – Florian Schmidt
Presentation: goo.gl/ky7Jr9
Video: www.youtube.com/watch?v=OYgTWhYjD0o

Unikraft: An easy way of crafting Unikernels on Arm
@ XPDDS18 – Kaly Xin
Presentation: goo.gl/162aAq
Video: www.youtube.com/watch?v=_ocRiTtYdfQ

Questions

lars.kurth@xenproject.org
george.dunlap@citrix.com

Picture by Lars Kurth