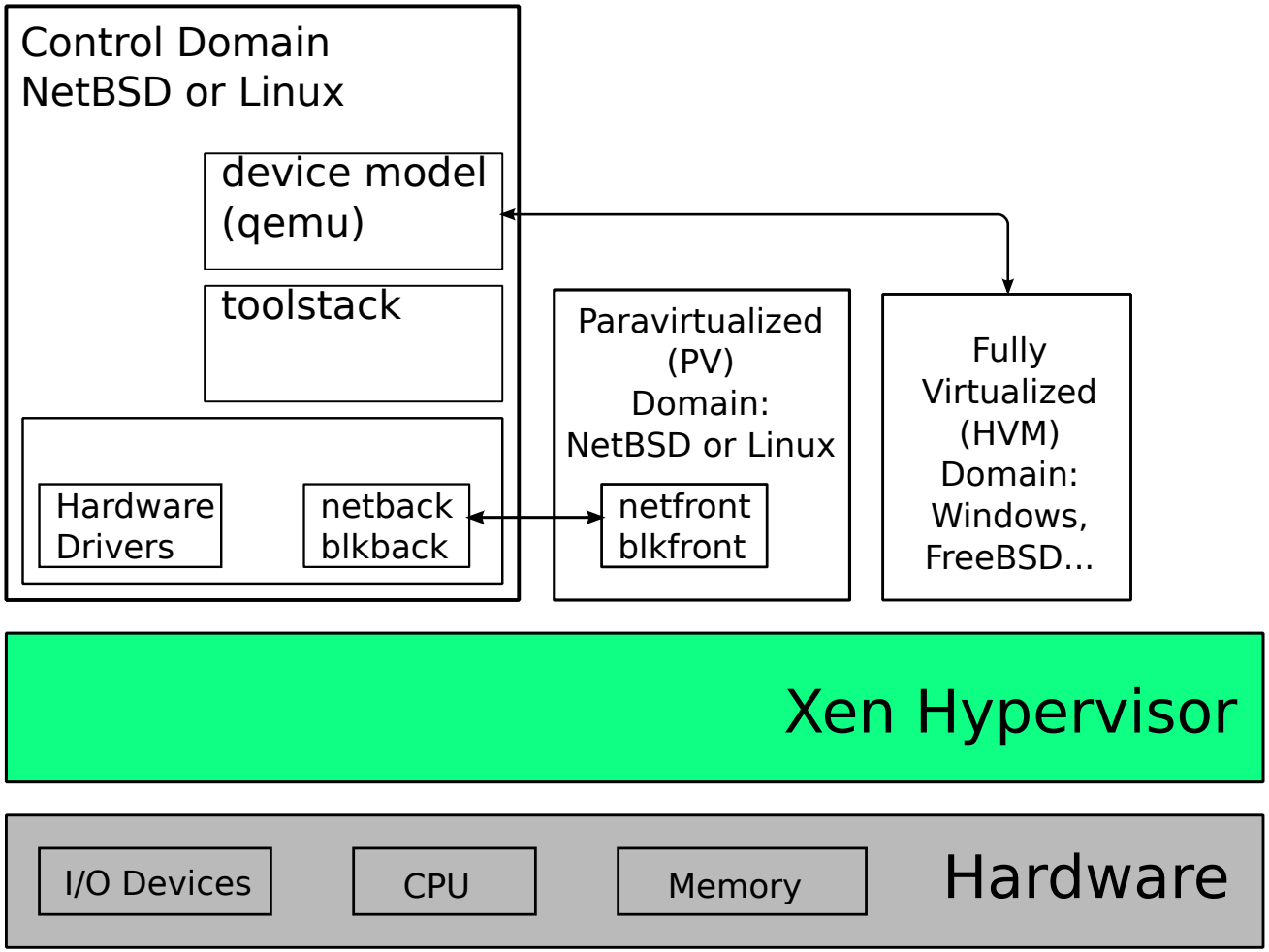
Performance tuning Xen

Roger Pau Monné
roger.pau@citrix.com

Madrid – 8th of November, 2013

Xen.org

Xen Architecture



Paravirtualization

- Virtualization technique 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 paravirtualiztion

Paravirtualization

- All this changes lead to the following interfaces being paravirtualized:
  - Disk and network interfaces
  - Interrupts and timers
  - Boot directly in the mode the kernel wishes to run (32 or 64bits)
  - Page tables
  - Privileged instructions

Full virtualization

- 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.

The full virtualization spectrum

VS	Software virtualization
VH	Hardware virtualization
PV	Paravirtualized

Poor performance

Room for improvement

Optimal performance

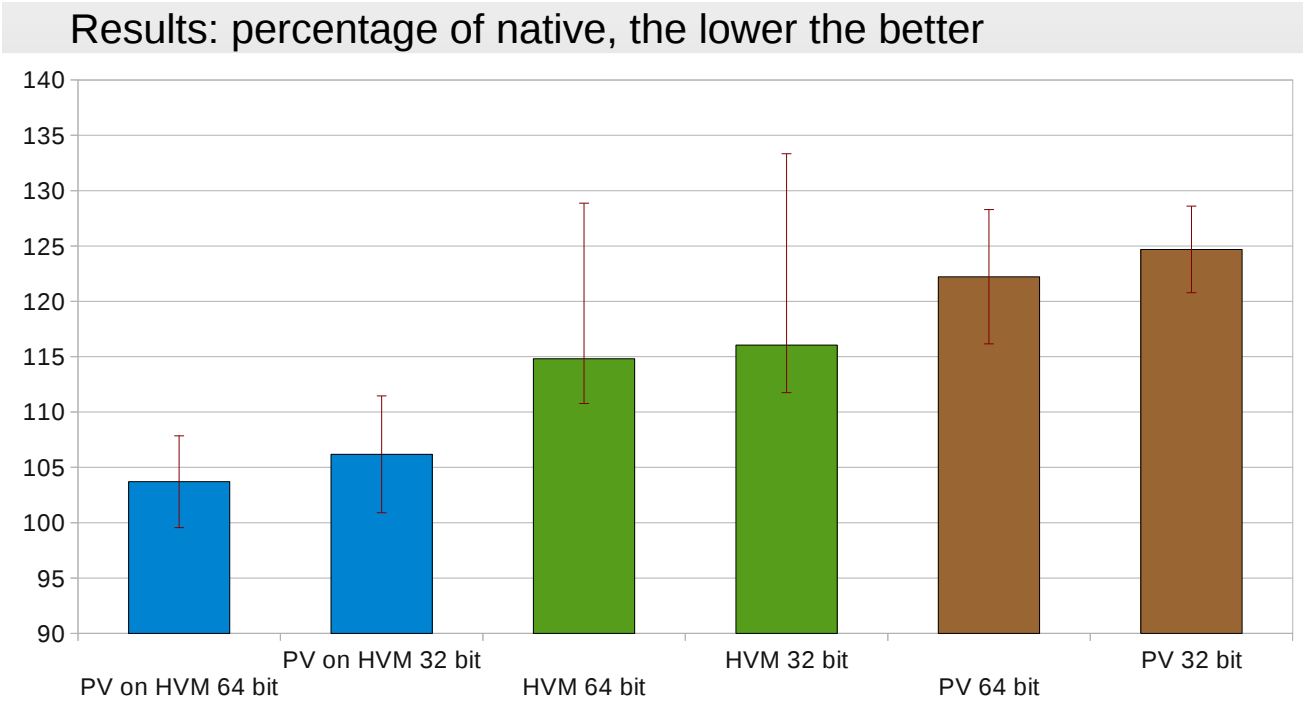
	Disk and network	Interrupts and timers	Emulated motherboard	Privileged instructions and page tables
HVM	VS	VS	VS	VH
HVM with PV drivers	PV	VS	VS	VH
PVHVM	PV	PV	VS	VH
PV	PV	PV	PV	PV

Guest support

- List of OSes and virtualization support:

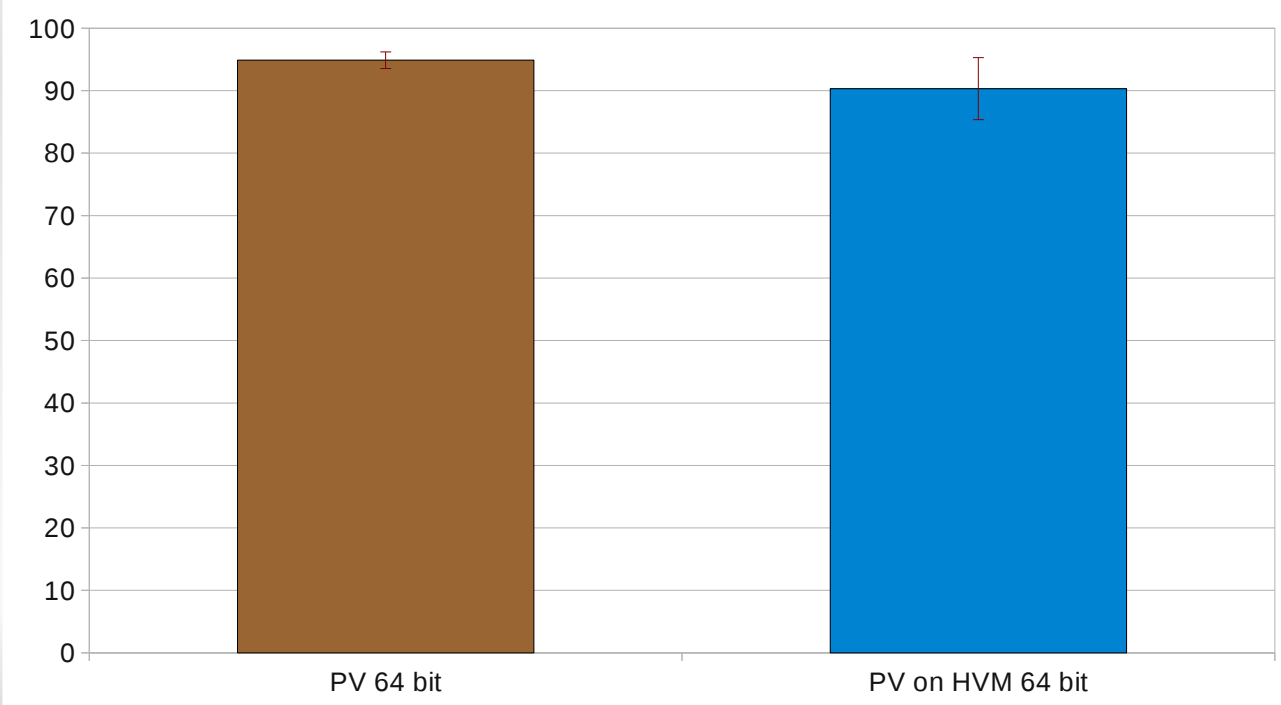
	PV	PVHVM	HVM with PV drivers	HVM
Linux (PVOPS)	YES	YES	YES	YES
Windows	NO	NO	YES	YES
NetBSD	YES	NO	NO	YES
FreeBSD	NO	YES	YES	YES
OpenBSD	NO	NO	NO	YES
DragonflyBSD	NO	NO	NO	YES

Kernbench


Results: percentage of native, the lower the better
140
135
130
125
120
115
110
105
100
95
90
PV on HVM 64 bit
PV on HVM 32 bit
HVM 64 bit
HVM 32 bit
PV 64 bit
PV 32 bit

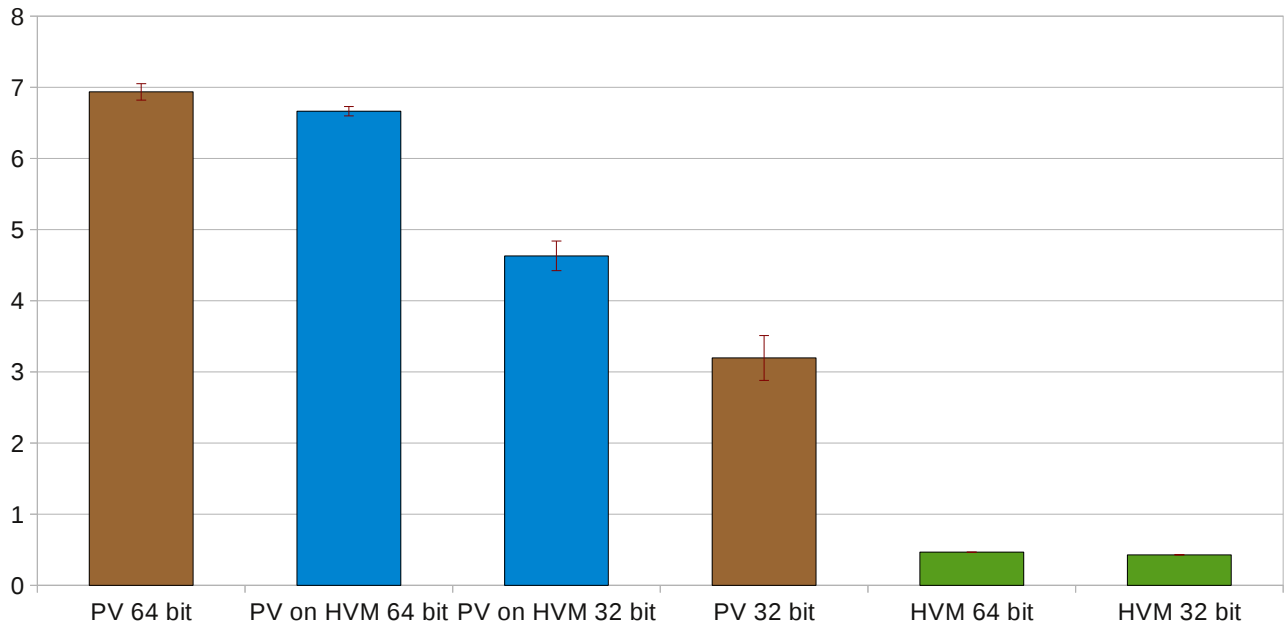
Specjbb2005

Results: percentage of native, the higher the better



Iperf

Results: gbit/sec, the higher the better



What virtualization mode should I choose?

- Linux supports several virtualization modes, which one is better?
  - Depends on the workload.
- Generally PV mode will provide better performance for IO, but when using 64bit guests PV can be slower.
- There isn't a fixed rule here, the best way to find out is to evaluate the workload on the different kind of guests.

Dom0

- Dom0 is the most important guest in the Xen infraestructure.
- It can become a bottleneck easily if not configured correctly.
- Dom0 is in charge of creating the guests, but usually also provides the backends and device models for guests.
- Xen provides some options to tune performance of Dom0

dom0_mem boot option

- dom0_mem tell Xen how much memory can be used by the Dom0.
- If not set all memory will be assigned to the Dom0, and ballooning will be used when launching new guests, reducing the memory used by the Dom0.
- The value should be set depending on the usage, HVM guests consume more memory in the Dom0 because they need a Qemu instance.
- If dom0_mem is set make sure to disable ballooning in the toolstack by setting **autoballoon=0** in **/etc/xen/xl.conf**.

dom0_max_vcpus and dom0_vcpus_pin

- dom0_max_vcpus: maximum number of CPUs the Dom0 will see, also depends on the utilization of the Dom0 and the type of guests.
- dom0_vcpus_pin: pinning Dom0 vcpus to physical CPUs is a good idea for systems running IO intensive guests.
- Setting up the serial cable: although not important for performance, setting up a serial cable is really important when debugging. For more info:
  `http://wiki.xen.org/wiki/Xen_Serial_Console`

Dom0 Boot tuning options

- For example if I had to set up a Dom0 on a machine with 8 CPUs and 8GB of RAM I would use the following boot line: **com1=115200,8n1 console=com1 dom0_mem=1024M dom0_max_vcpus=2 dom0_vcpus_pin**.
- More info about boot parameters can be found at: `http://xenbits.xen.org/docs/unstable/misc/xen-command-line.html`.

General performance notes

- Always try to use physical disks as backends. Xen has mainly two ways of connecting disks to the guest depending on the format of the image, if it's a block device it will be attached using blkback, which is inside the Linux kernel and it's faster.
- Take into account the number of CPUs your physical box has and avoid using more VCPUS than PCPUS if running performance intensive applications.

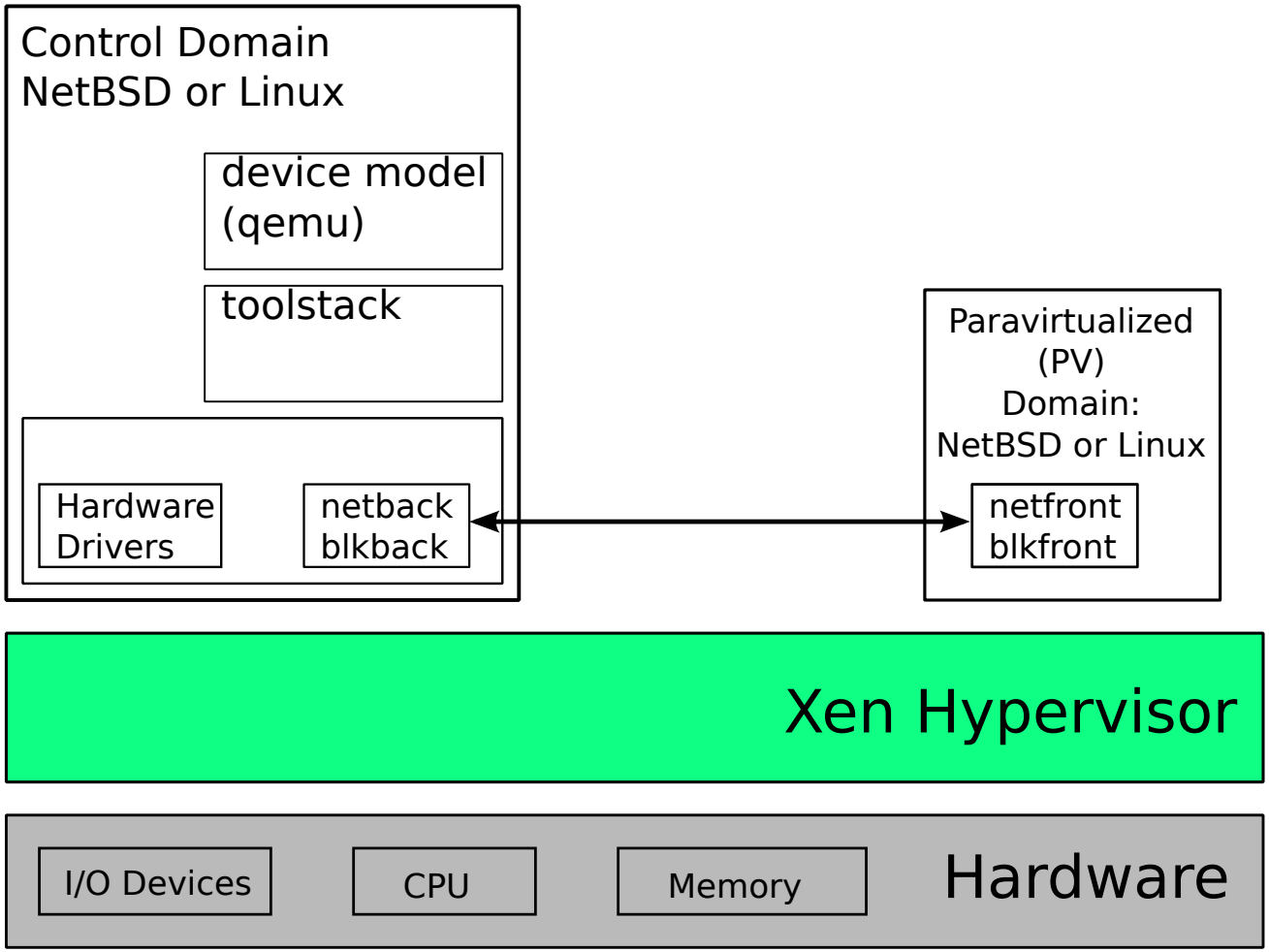
Pinning CPUs

- You can pin VCPUs to PCPUs in order to obtain better performance or to distribute the workload across your CPUs to suit your needs. For example low latency VMs can be exclusively pinned to different PCPUs.
- cpus: allows to select in which CPUs the guest can run. The list can also contain specific CPUs where the guest is not allowed to run. Specifying "0-3,5,^1" allows the guest to run on CPUs 0,2,3,5.
- If Dom0 is pinned to certain PCPUs avoid running guests on those PCPUs to obtain better performance. If Dom0 is pinned to CPU 0, use the following CPU mask in order to prevent other guests from running on CPU 0: "^0".

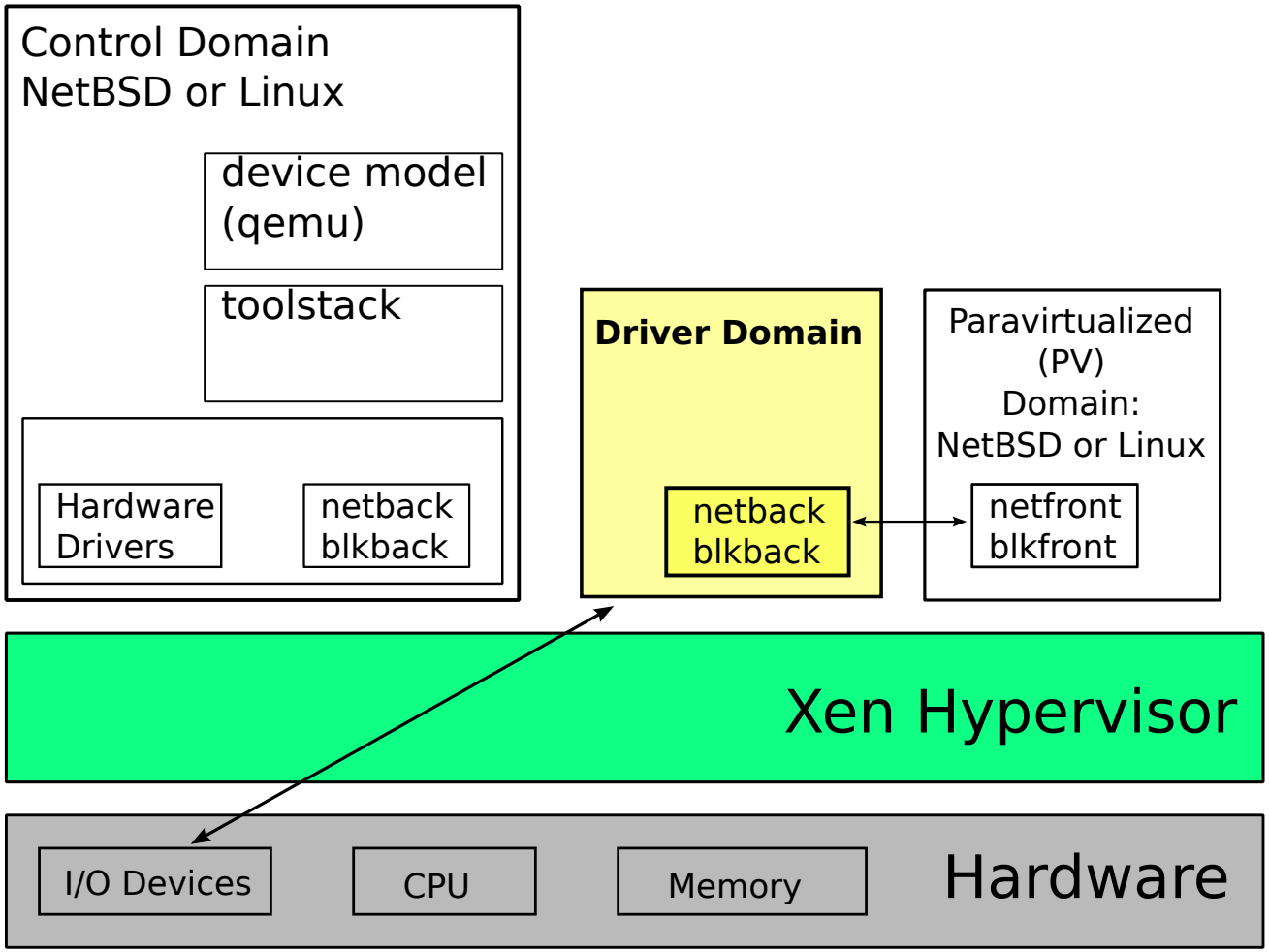
Scheduler options

- The Xen scheduler has several options that can also be tunned from the guest configuration file, in order to give a certain guest more share from the processor or to schedule it more frequently.
- cpu_weight: weight of the domain in terms of CPU utilization. For example a domain with a weight of 512 will get twice as much CPU than a domain with a weight of 256. Values range from 1 to 65535.
- cap: fixes the maximum amount of CPU a domain is able to consume. Expressed in percentage of one physical CPU. 100 is one CPU, 50 half a CPU, 400 four CPUs.
- More info can be found at `http://xenbits.xen.org/docs/unstable/man/xl.cfg.5.html`

Driver Domains I



Driver Domains II



Driver Domains III

- Driver domains allow to offload work normally done in Dom0 to other domains.
- It also provides better security, less surface for exploits in Dom0.
- This is a current work-in-process.

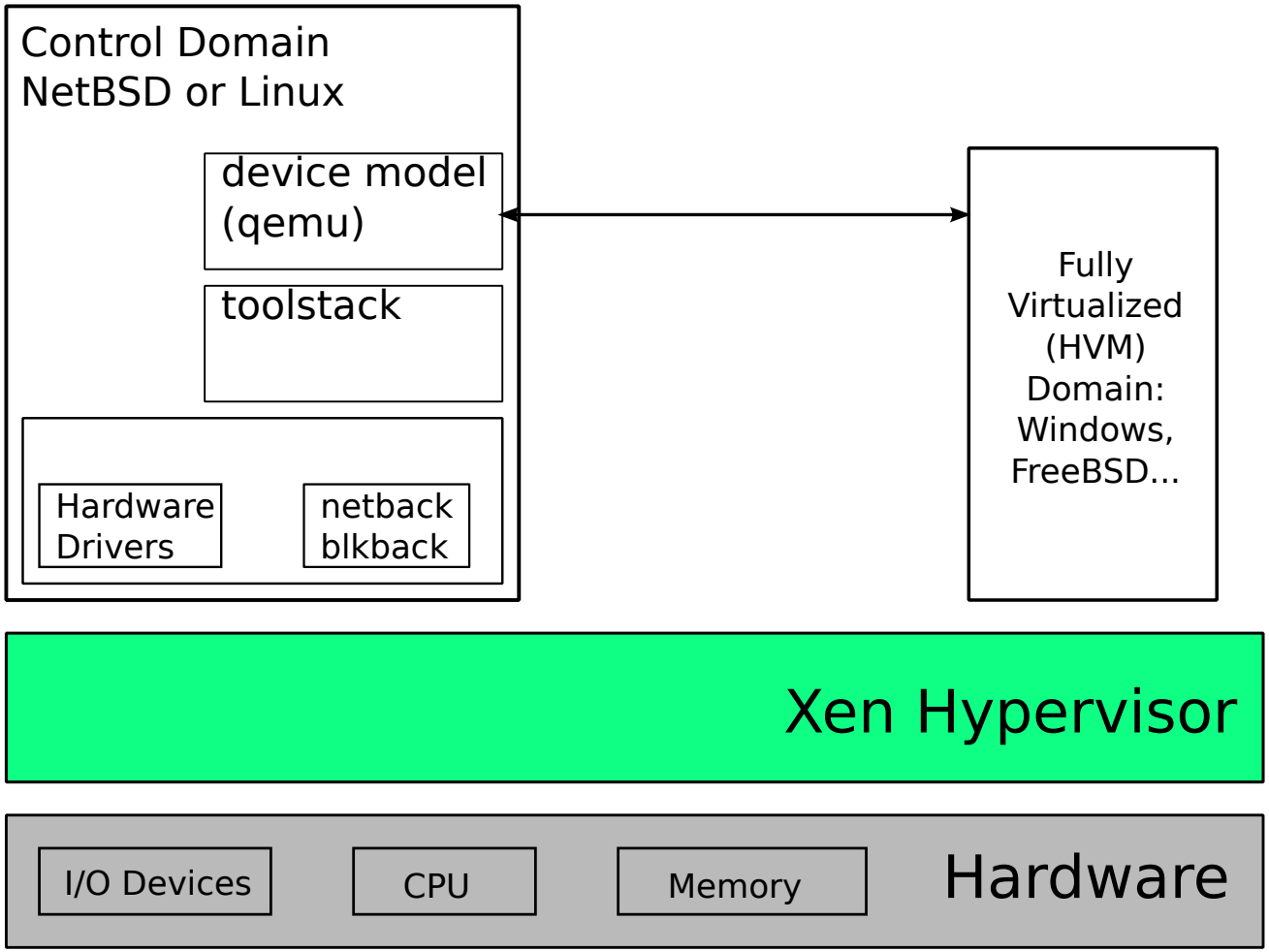
HVM specific I

- HVM guest require the usage of assisted paging, in order for the guest to see the memory area as contiguous when it's not.
  - HAP: (Hardware Assisted Paging) is used by default since it tends to perform better under most workloads
  - shadow: was introduced before HAP, and can provide better performance under certain workloads that have low TLB locality (for example databases or java applications).
- Again, the best way to know is to try the workload by yourself.

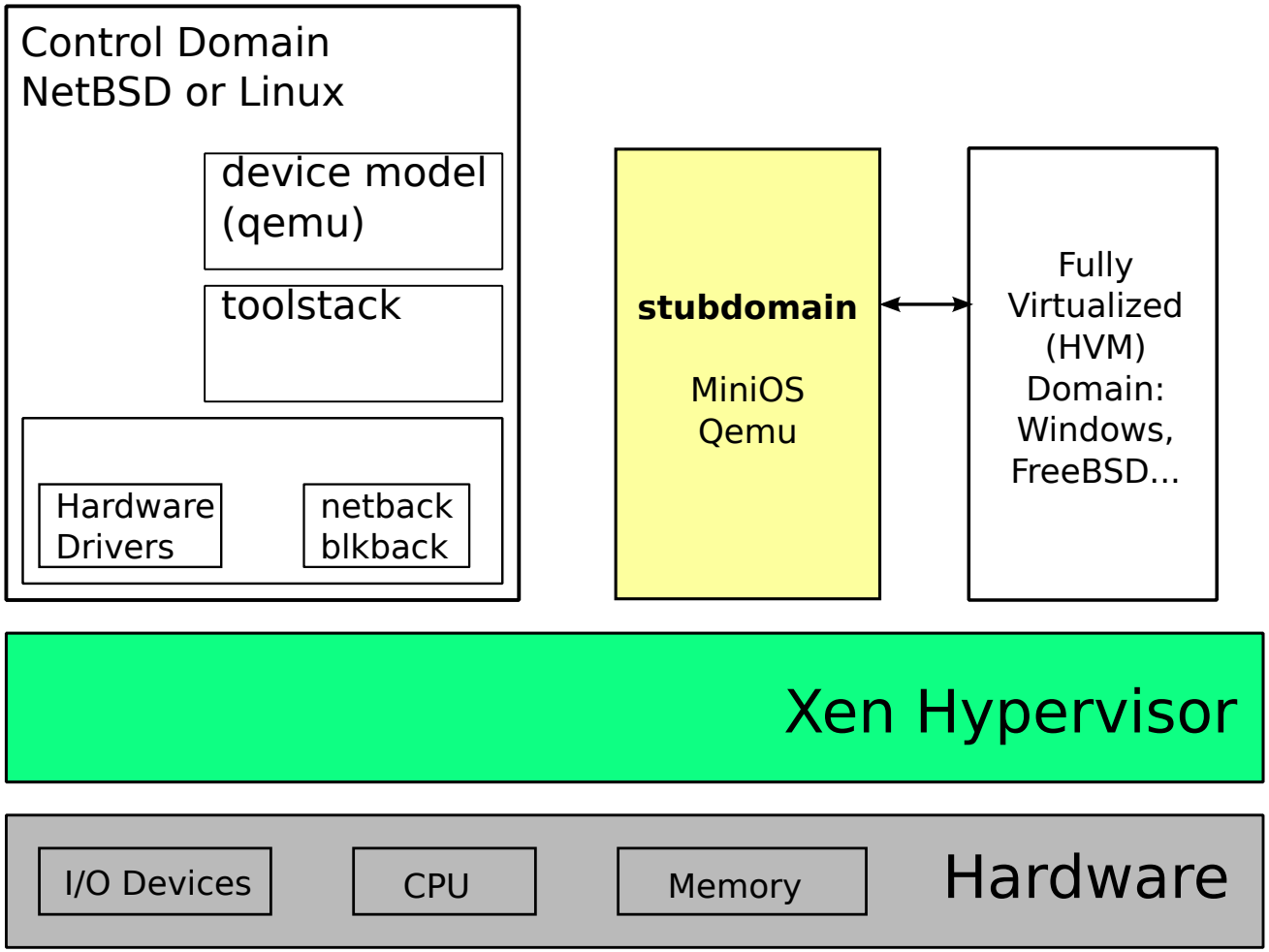
HVM specific II

- HVM domains require a Qemu instance in Dom0 to perform the necessary device emulation.
- This might be a bottleneck if running a lot of HVM domains in the same node, since each one requires a Qemu instance running in Dom0 that uses both Dom0 CPU and Memory.
- To avoid this, we can launch the Qemu process in a different domain called "Stubdomain".
- This allows to offload work from Dom0.

HVM specific III



HVM specific IV



Conclusions

- Xen offers a wide variety of virtualization modes.
- The best way to know which mode will bring better performance is to try it, although there are several tips that apply to all guests.
- We are constantly working on performance improvements, so keep updated in order to get the best performance.

Q&A

Thanks

Questions?

http://wiki.xen.org/wiki/Xen_Best_Practices
http://wiki.xen.org/wiki/Xen_Common_Problems