

Storage and Virtualization

Aaron Dailey
***Sun China Engineering and
Research Institute***

Agenda

- What is server virtualization
 - Solaris virtualization
- Problems with SANs and Virtualization
- N_Port_ID Virtualization
- The Solaris Implementation
 - Driver and Management
 - Xen
 - Logical Domains
 - Containers

Server Virtualization 1.0



Server Virtualization

- Server virtualization abstracts hardware, using software and/or hardware
- Create multiple instances of a platform, so you can run multiple instances of operating systems
- One server can run different operating systems concurrently
- *Migration – a guest operating system can move from one host operating system, and therefore one server to another*

Vocabulary

- Guest operating system
 - OS runs on virtual hardware
 - In Xen DOMU
 - In Logical domains LDOM A
- Host operating system
 - OS touches the hardware, and supplies services to guest
 - In Xen DOM0
 - In LDOM, Service Domain

How does it work?

- Two significant methods today:
- Paravirtualization
 - Each guest operating system has special device drivers to access the host operating system
- Full Virtualization
 - All hardware is simulated by the host operating systems, guest operating system can be unchanged

Paravirtualization

- Operating systems are modified with special device drivers between guest and host
- Advantages:
 - Fast
- Disadvantages
 - Must write special device drivers for each guest and host operating system
- Examples
 - Traditional Xen, Logical Domains

Full Virtualization

- Host operating system can simulate common hardware
- Guest operating systems run unmodified
- Advantages
 - Guest operating systems run unmodified
- Disadvantages
 - Possibly slower
 - Must write device drivers to simulate hardware
- Example: VMWare, Xen with HVM

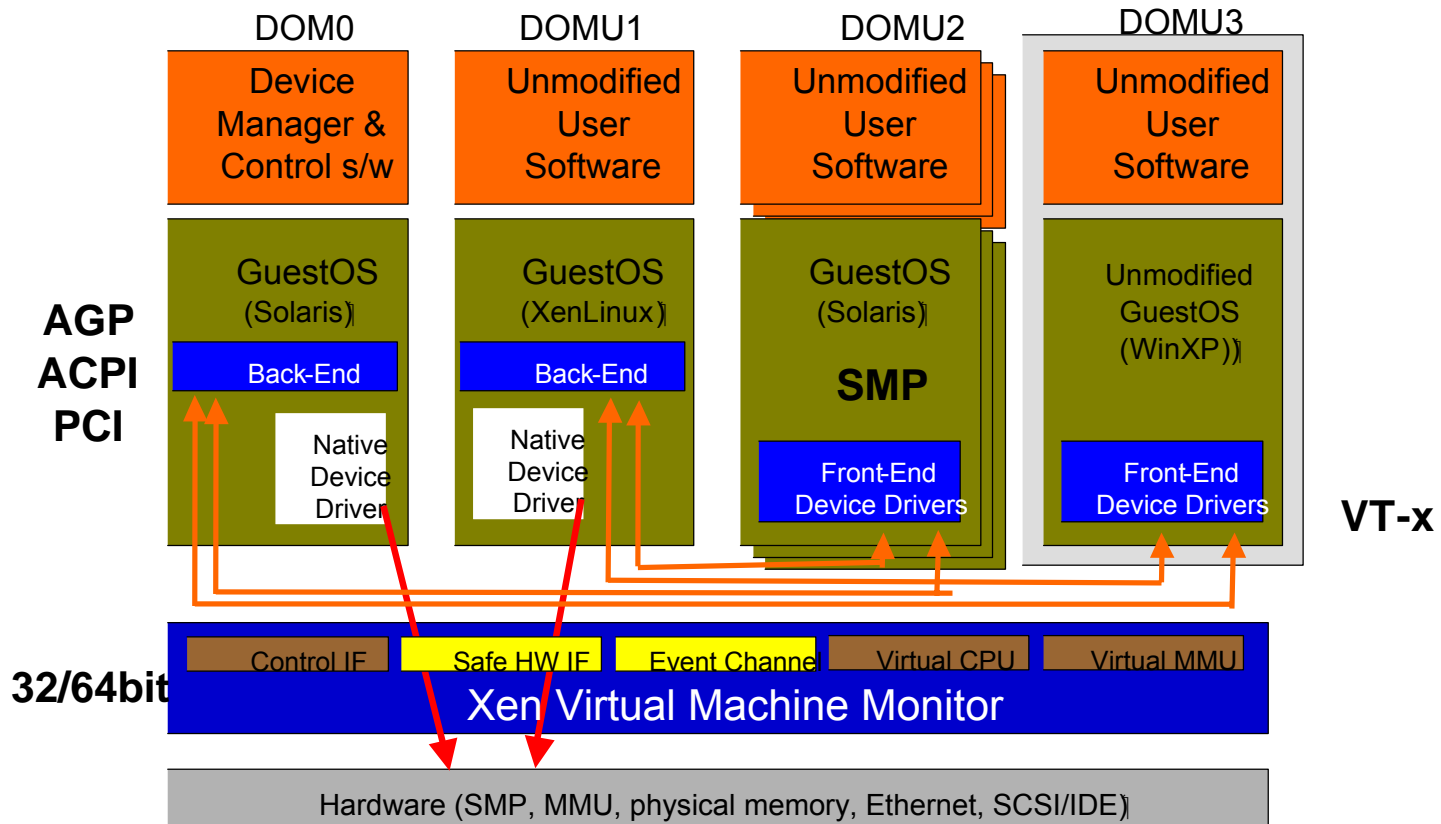
Solaris and Virtualization

- Several implementations
 - Xen
 - X86 architecture only
 - Logical Domains
 - SPARC sun4v architecture only
 - Containers (Zones)
 - Both x86 and SPARC
 - Not really virtualization – only one operating system instance, but simulated separation

Storage in Xen and LDOM environment

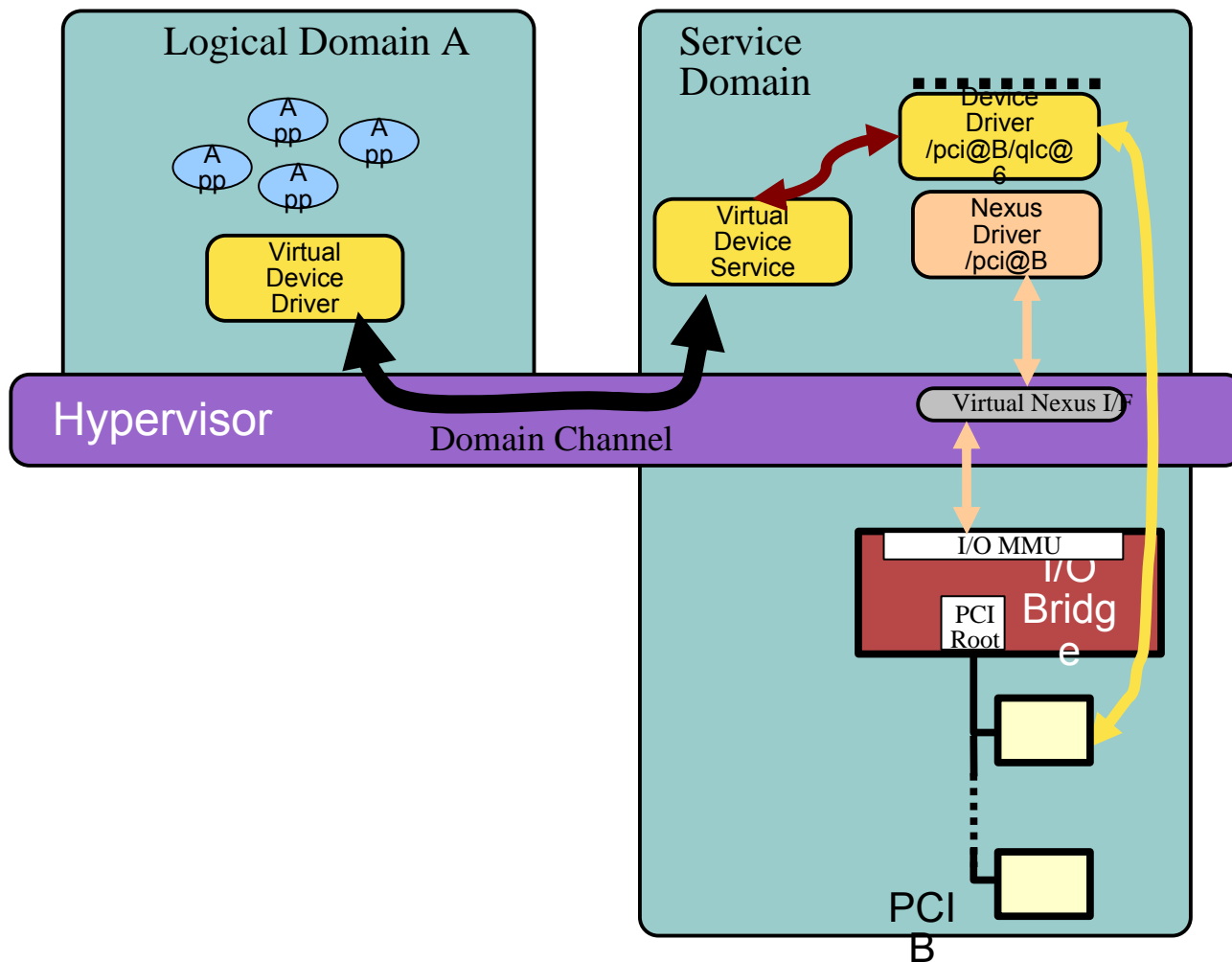
- Today, block devices are presented
 - No SCSI passthrough access (but this is changing ...)
- Access block devices via split block driver
- Specific block devices are presented to specific DOMUs
- `lofi(7D)` allows files from DOM0 to be accessed as block devices in DOMU

Xen 3.0 Architecture





Solaris Logical Domains



FibreChannel in SANs

- Background
 - System administrators use World Wide Names (WWN) from FibreChannel HBAs for access control
 - Used in switches to create zones
 - Used in SCSI targets for LUN masking, to create a list of hosts which can see a certain LUN

FibreChannel in Virtualized Servers

- Today, every guest OS shares an HBA's WWN on the SAN
 - LUN Masking and Zoning isn't possible between guest OS instances
- Migration makes things worse ...
 - Now every HBA in pool of migration machines must be in the same zone, and be in the same LUN mask list

One solution - dedicate HBA to guest OS

- Why not just dedicate an HBA to a DOMU?
 - Most virtualization systems *can* do this
- It *can* be useful to dedicate hardware to a guest operating system
 - If interfaces don't allow full access to HW
- But, in general
 - It's expensive
 - Machines don't have enough slots

Solution!

- Create virtual FibreChannel HBA ports
 - Called N_Port_ID Virtualization (NPIV)
- One physical port can have multiple WWNs
- For each HBA dedicate one WWN per guest operating system per HBA
- Zoning and LUN Masking work as they always have



N_Port_ID Virtualization

- Background
 - Devices have one or more physical ports
 - Ports for devices that connect to a switch or one other device are call N_Ports
 - There are also are other types, but we won't consider here
 - N_Ports have a unique world wide name (WWN)
 - But, N_Ports are addressed by a 24 bit N_Port_ID, which is assigned by the switch



N_Port_ID Virtualization

- How does it work?
- First, do N_Port does normal FLOGI ELS for permanent WWN
 - Same as always, get an N_Port_ID
- Next, send FDISC ELS with new Port_Name (WWN) and S_ID=0
 - Fabric returns new virtual N_Port_ID
- This N_Port_ID can be used to send SCSI commands, or anything a permanent WWN based N_Port_ID could be used for

Infrastructure

- Generally, NPIV requires
 - New switch firmware
 - New HBA firmware
- Targets just work, and don't know they're using NPIV initiator
- Generally 4Gb equipment will support NPIV
 - Sometimes need updated firmware
- Some 2Gb equipment with support NPIV with new firmware

The Solaris Implementation

- Device Driver Changes
- Management Changes
- Xen Changes
- Logical Domain Changes
- Container Changes

Driver Architecture Changes

- Solaris FC driver architecture is modular
 - FCA control hardware specific portions, for example delivering a command to HBA
 - fp controls a single port, for example querying nameserver, responding to requests
 - fcp knows about SCSI to FC mapping
- Change relationship between FCA and fp instances from one to one to one to many
- Management changes
 - New ioctls to manage creating and deleting NPIV ports

Solaris FC Driver Architecture

today

One NPIV Enabled HBA, multi ports

fcp
(scsi)

fcp
(scsi)

fp
(port)

fp
(port)

fp
(port)

fp
(port)

fp
(port)

FCA

FCA

FCA



Note: only relevant drivers are shown



Management Changes

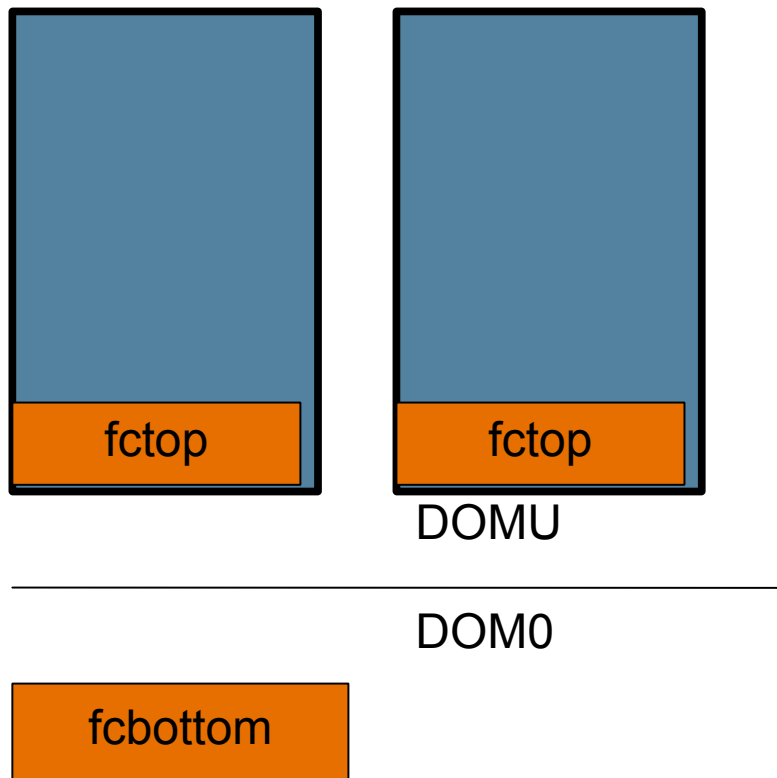
- First, need a way to talk to driver
 - Use additions to T11 HBA API
 - T11 HBA API standardization is in progress, but incomplete
- Then, modify existing HBA API based utility to configure NPIV

Xen Approach 1

- The first approach:
 - Create a new split driver at the FC Port layer
 - Run FibreChannel protocol in DOMU for an NPIV WWN
 - Run HBA driver in DOM0
 - Split driver communicates between DOM0 and DOMU
- Advantages
 - Architecturally clean
 - Possible to run FibreChannel protocol in each DOMU
 - Very flexible and powerful



Xen Approach 1



- Each guest operating system contains top half FC driver
- Host operating system contains bottom half FC driver
- NPIV address routes between top and bottom
- Communicate via Hypervisor primitives

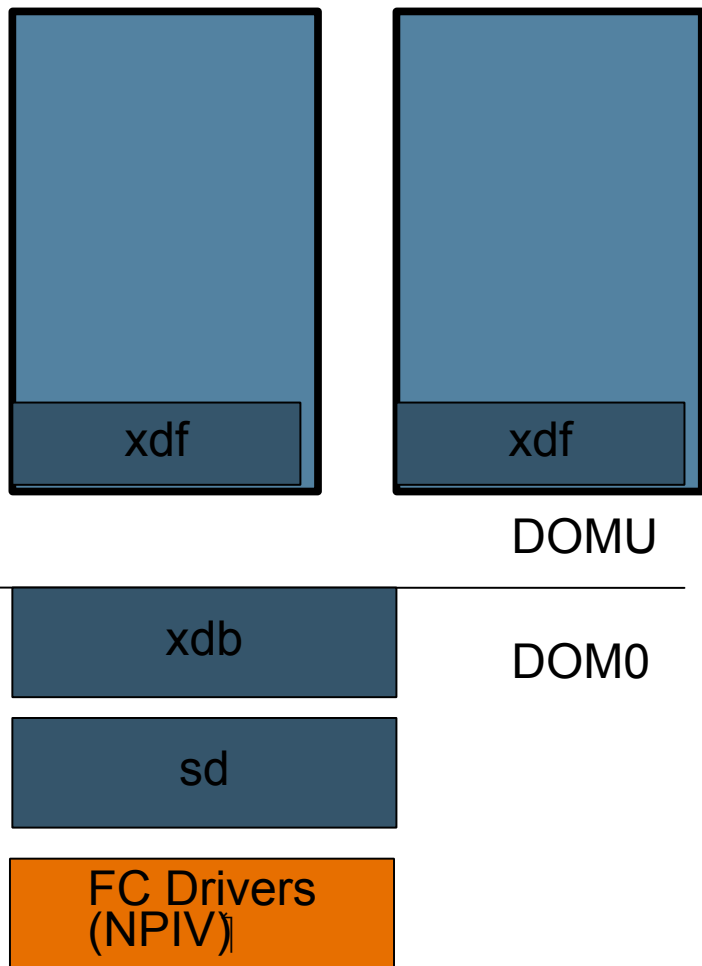
Xen Approach 1

- Ultimately, rejected
- Too complicated
 - Split drivers are hard to write, even for a fairly well abstracted interface found in Solaris FC driver –
 - interface between top and bottom is ring queue
 - No direct calls
 - Modify every guest OS
 - While elegant, not necessary for taking advantage of NPIV

Xen Approach 2

- Entire Implementation in DOM0
- Modify block split driver to determine origin of block device
 - Each DOMU has a WWN bound to it
- Optionally, automatically expose a block device to a DOMU if bound to that DOMU
 - It's how standalone systems work
- By default, use existing block device map to map between block devices and DOMU

Xen Approach 2



- NPIV is implemented in FC driver
- FC Driver Resides in DOM0
- Existing block split driver manages communication between DOM0/DOMU
- Block split driver filters based on NPIV

Xen Sample (initial state)

```
-bash-3.00# xm npiv-list 1  
NPIV wasn't configured for this domain
```

```
-bash-3.00# fcinfo hba-port  
HBA Port WWN: 3c00000000000000  
  OS Device Name: /dev/cfg/c0  
  Manufacturer: Emulex  
  Model: LP11000-M4  
  Firmware Version: 2.70a5  
  FCode/BIOS Version: none  
  Type: N-port  
  State: online  
  Supported Speeds: 1Gb 2Gb 4Gb  
  Current Speed: 2Gb  
  Node WWN: 0100000000000000  
  NPIV port list :
```

Xen Sample (Enable NPIV)

```
-bash-3.00# xm npiv-add 1 10000000c94cc1af 10000000c94cc1a0
```

```
-bash-3.00# xm npiv-list 1
```

```
NPIV info for domain dom0:
```

```
Virtual Port WWN:      10000000c94cc1af
```

```
Virtual Node WWN:     10000000c94cc1a0
```

```
No binding info.
```

```
-bash-3.00# xm npiv-bind 1 3c00000000000000 0
```

```
-bash-3.00# xm npiv-list 1
```

```
NPIV info for domain dom0:
```

```
Virtual Port WWN:      10000000c94cc1af
```

```
Virtual Node WWN:     10000000c94cc1a0
```

```
Physical Port WWN:    3c00000000000000
```

```
Autoattach:          0
```

```
No devices found.
```



Xen Sample (Results)

```
-bash-3.00# fcinfo hba-port
HBA Port WWN: 3c00000000000000
  OS Device Name: /dev/cfg/c0
  Manufacturer: Emulex
  Model: LP11000-M4
  Firmware Version: 2.70a5
  FCode/BIOS Version: none
  Type: N-port
  State: online
  Supported Speeds: 1Gb 2Gb 4Gb
  Current Speed: 2Gb
  Node WWN: 0100000000000000
  NPIV port list :
    Virtual Port1:
      Node WWN:10000000c94cc1a0
      Port WWN:10000000c94cc1af
```

Logical Domains

- From NPIV perspective, logical domain solution should be similar to Xen
- Modify Service Domain, similar to Xen DOM0
- Bind NPIV WWN to domains
- LDOM also uses split block driver



Containers

- Solaris way to partition operating system, although one operating system instance
- Conceptually, NPIV can work here, by partitioning information
- Zones currently contains facilities for exporting certain devices to certain zones
- We will use these facilities to export *sd instances to certain zones based on NPIV WWN*

Related Work

- Fujitsu's SCSI Split driver for Xen
 - Allows SCSI passthrough
 - Solves a problem with Xen – no good way to access SCSI devices, especially tape, from from DOMU
- See XenSource.com, Xen Summit Spring 2007
- Also, alternative implementation of NPIV

References/Further Reading

- NPIV Project on OpenSolaris
- My blog
 - <http://blogs.sun.com/aarondailey>