



Logical Domains (LDoms) Concepts and Examples

Rich Reynolds /co Jeff Savit
Enterprise Architect
Sun Microsystems, Inc.



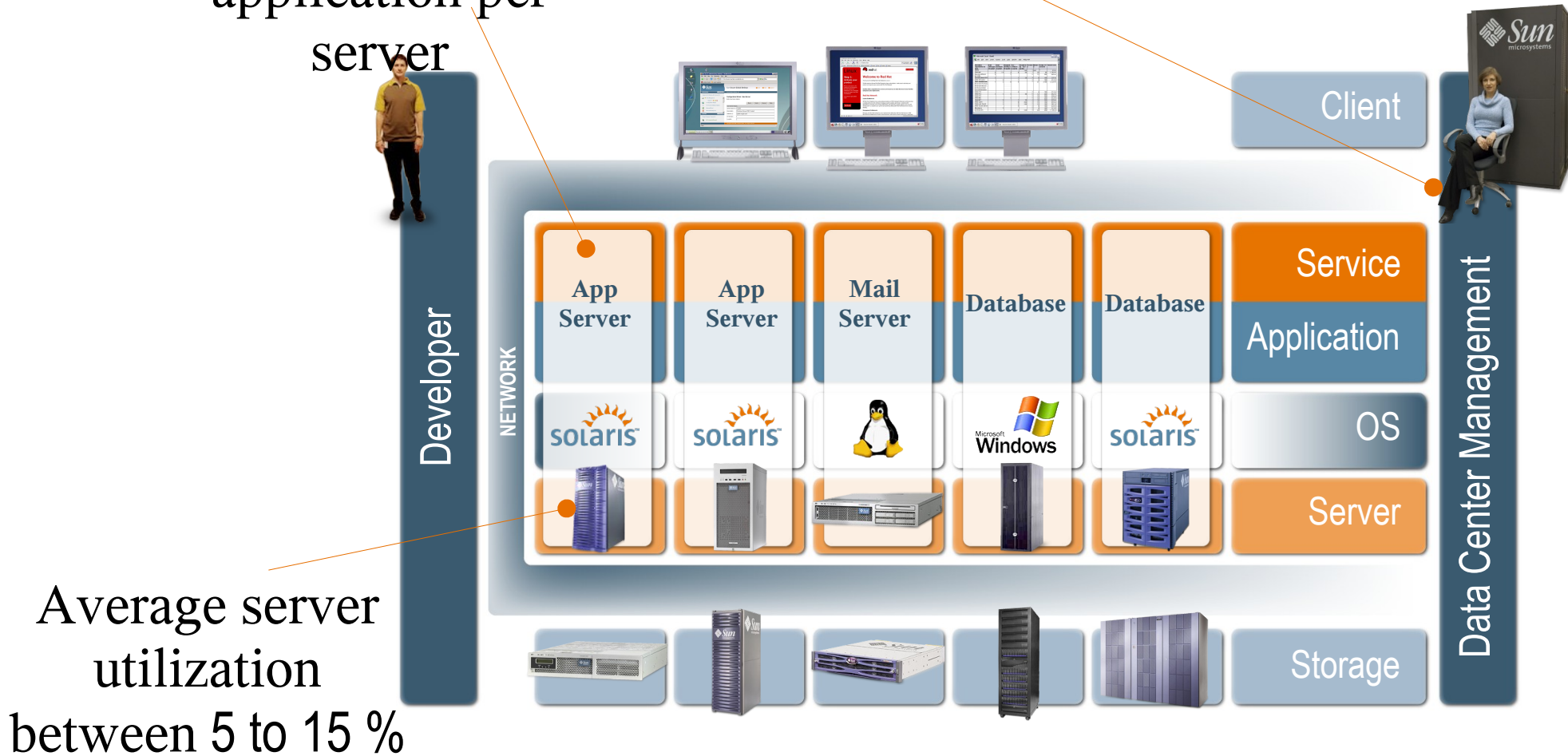
Agenda

- What is virtualization – and why now?
- Sun's new virtualization technology: Logical Domains (LDoms)
- LDoms concepts and features
- LDoms implementation and examples
- Best practices
- References and links

The Data Center Today

Single application per server

Server sprawl is hard to manage.
Deployment takes too much time



Virtualization: the illusion of a dedicated computer for multiple OS instances

- Virtualization has been around since the 1960s
 - > Now a mainstream technology available on multiple platforms
 - > Renewed emphasis due to changed economics and need: server sprawl, energy costs, need for rapid deployment, etc...
- Different styles with different benefits and limitations
 - > Domaining/partitioning: hardware or firmware that divides a server's CPU and RAM for use by separate OS instances
 - > Virtual Machines: host OS (“hypervisor”) software runs “guest machines” that act like separate copies of that server's architecture
 - > Differences: presence of architectural limit on number of OS instances, resource granularity, ability to overcommit CPU and RAM, amount of overhead
- **Sun innovations:**
 - > Solaris Containers: light-weight OS-level virtualization
 - > ***Logical domains: a new flexible domaining technology for T1-based servers***

What the customer wants

- Reduce datacenter server sprawl
- Ease infrastructure management
- Boost utilization for better return on investment
- Reduce burgeoning costs for power and cooling
- Improve reliability
- Increase service deployment speed
- Improve security
- Reduce, rationalize software licensing costs
- Reduce overall datacenter operational expense

Primary motivating factor - consolidation

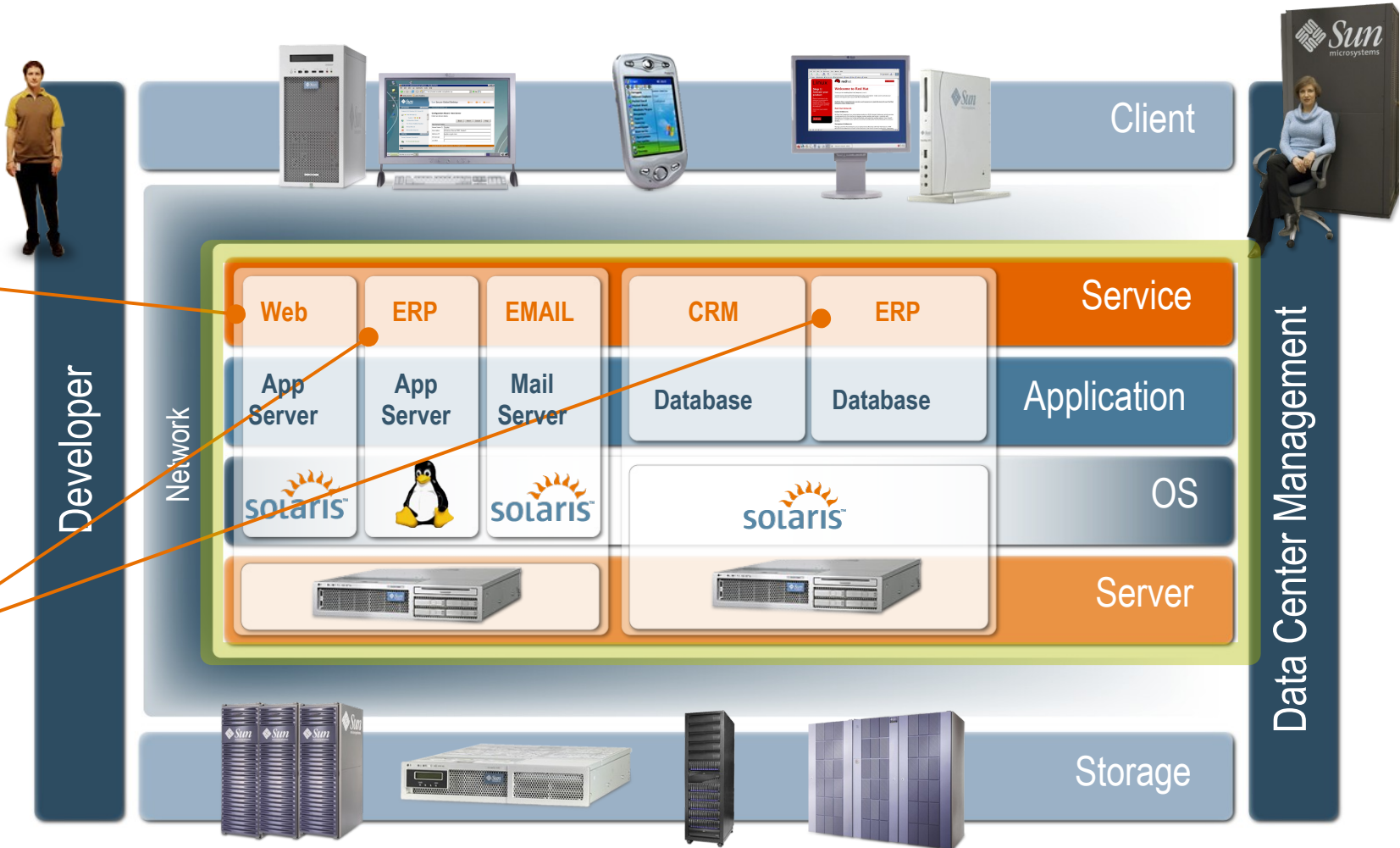
- “Customers are interested in improving the utilization of their computing resources”
 - > Translation: they want better return on the money they spend
- One way to do this is server consolidation
- Consolidation requires (at least) the following
 - > **1. Non-interference of independent workloads: the need driving server virtualization.** Consolidation also requires...
 - > 2. Resource management (to ensure service levels are met)
 - > 3. Resource accounting (to pay for shared resources)
 - > 4. RAS services (since eggs now in fewer baskets...)

Virtualized Data Center

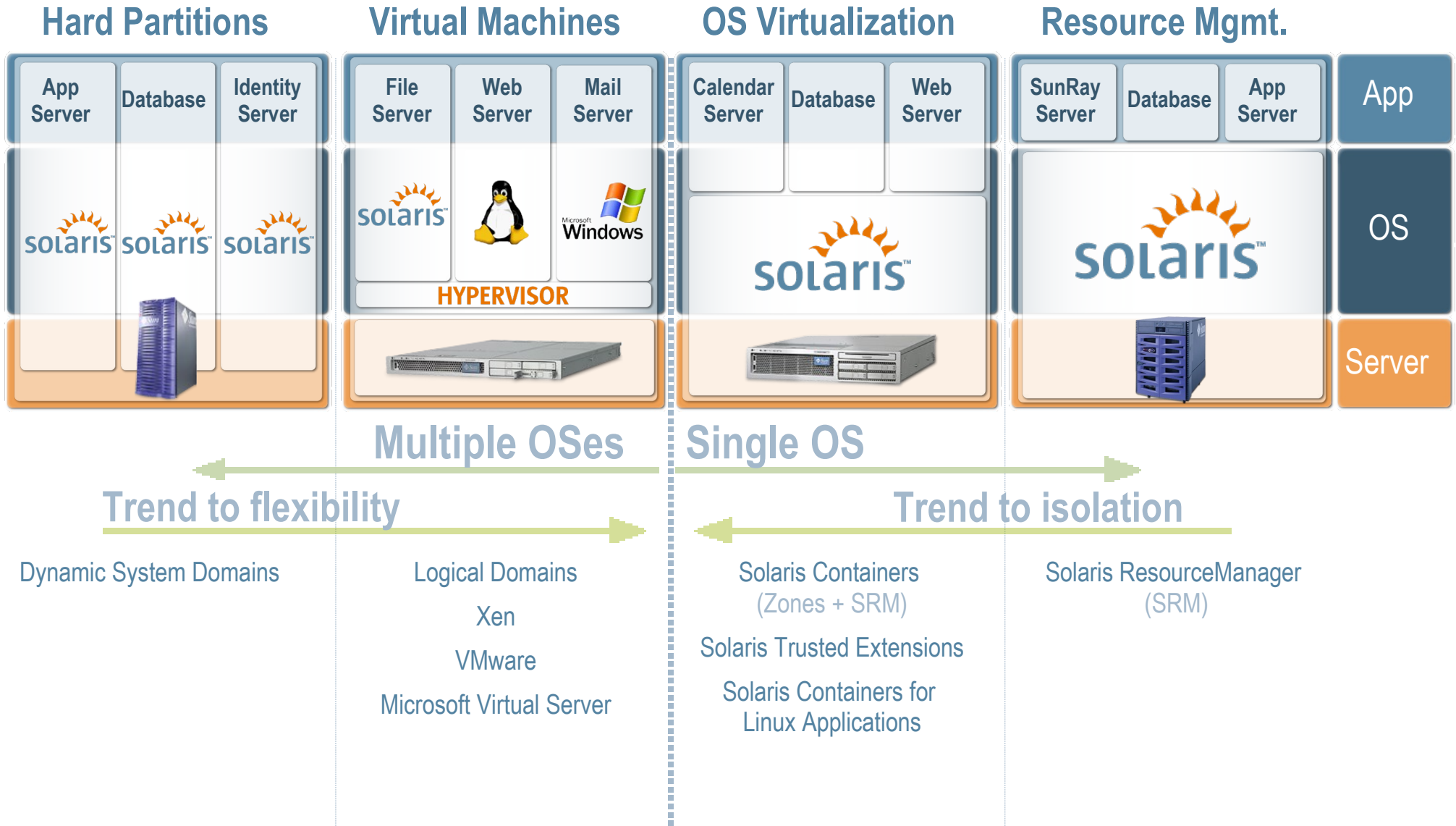
Maximize system resource use across entire Data Center

Ease management and cost of ownership through commonality of hardware

Improve reliability through application of Virtualization RAS features



Sun's Virtualization Offerings



Review of Sun Virtualization Offerings

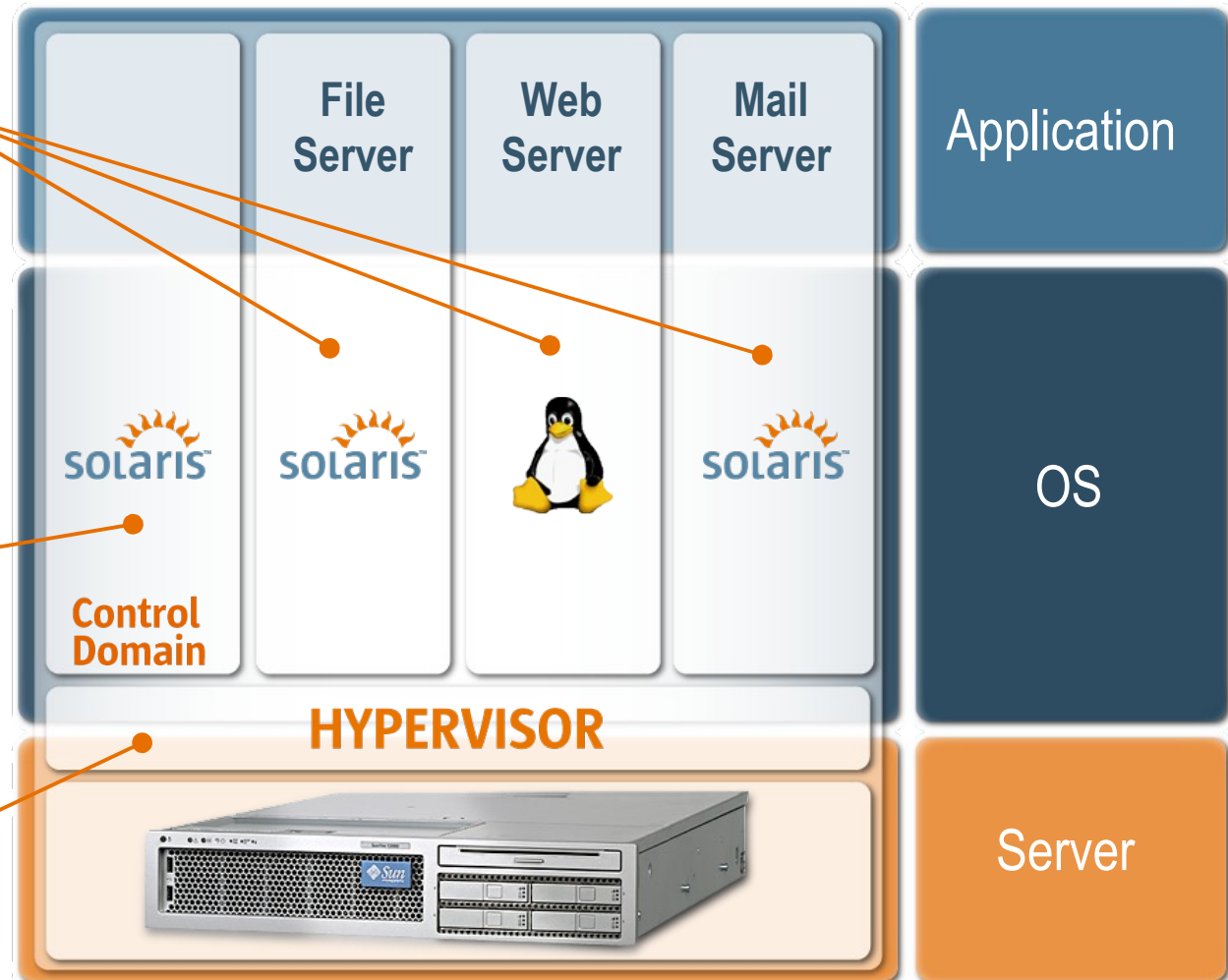
- Physical partitioning with Dynamic System Domains
 - > Electrically isolated, no single points of failure,
 - > Resource granularity at board level, and now improved granularity in the APL product line (to socket level)
- Virtual machines on x64
 - > Solaris unique among Unix systems for running under VMware
 - > Solaris support for open-source Xen in development for 2007
 - > Sun x64 servers excellent for VM-based consolidation
- Solaris containers, both SPARC and x64
 - > Low overhead, superior management and observability
- **Logical domains – now available on T[1-2] based servers**

Logical Domains

Solaris or Linux
guest domains

Solaris Control
Domain

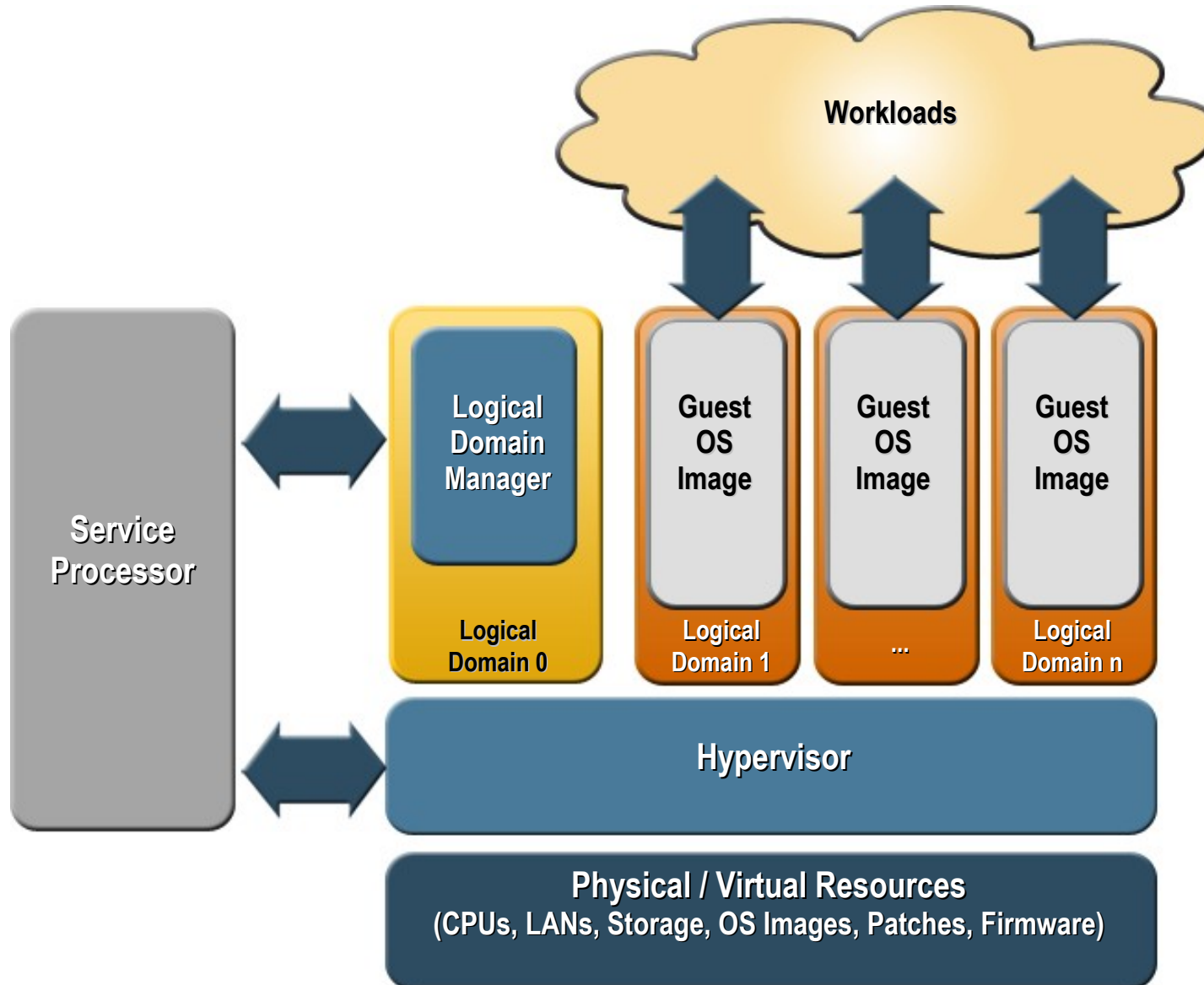
Ultra lightweight
Hypervisor in the
firmware



Concepts of LDom

- Each Logical Domain ...
 - > Appears as a fully independent server
 - > Has a unique OS install and configuration
 - > Has configurable CPU, memory, network, and I/O
 - > Has the ability to stop, start and reboot independently
- Which requires
 - > Fast, non-blocking, secure, robust I/O
 - > Efficient CPU virtualization
 - > Rigorously enforced security
 - > High reliability, availability, and serviceability
- A domain can be for system control, I/O, or general use

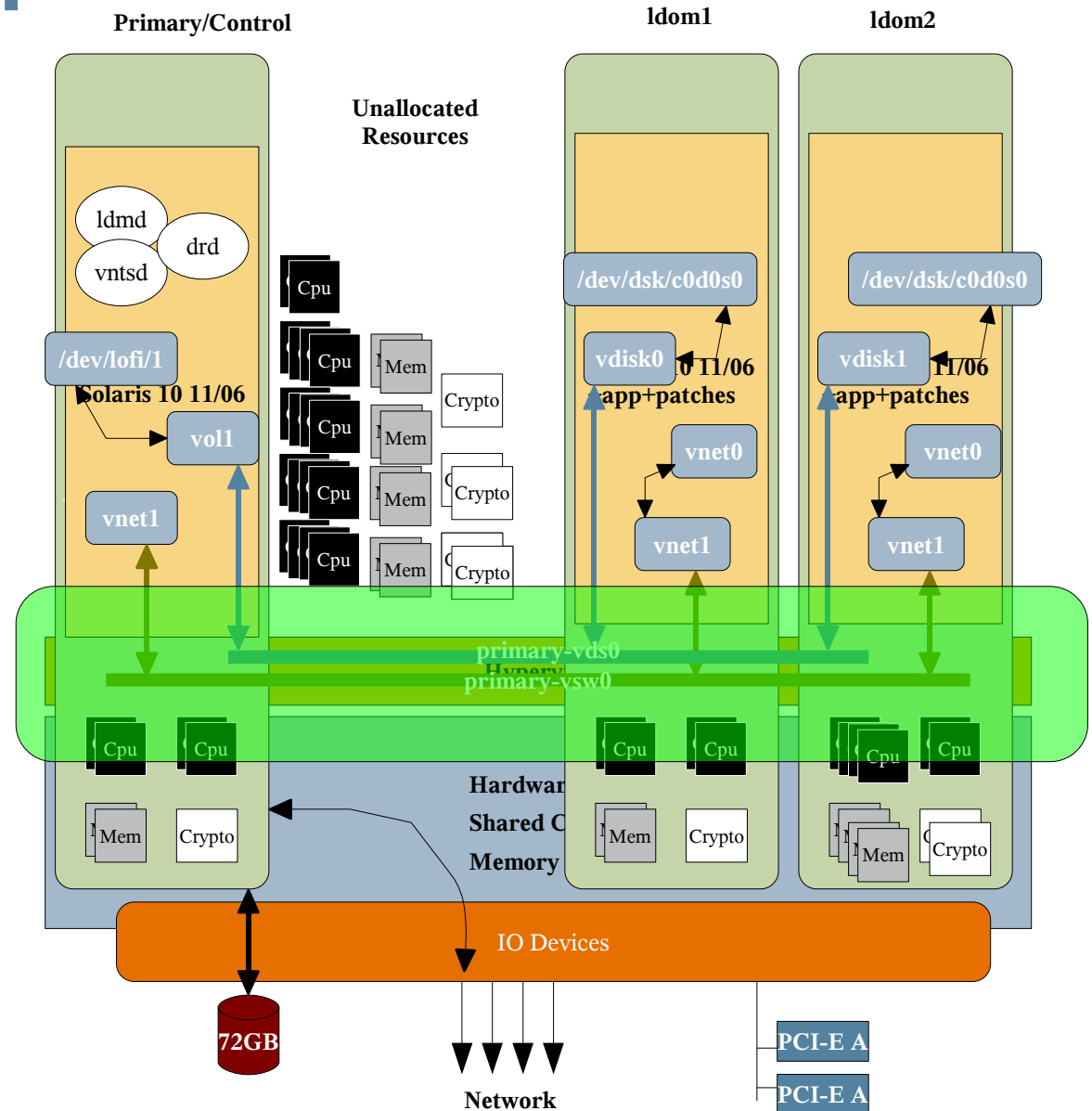
Server Virtualization: Logical Domains



- A hardware virtualization solution for T1000, T2000 and Netra T2000 and future CMT processors
- Partitions a single physical system into one or more fully isolated “logical domains”
- Enables fine-grained “physical to virtual” resource mapping and physical resource sharing
- Physical resource can be dynamically reassigned without impact on running OS images
- Exploits CMT properties for effective, efficient partitioning

Key LDoms components

- The Hypervisor
- The Control Domain
- The Service Domain
- The I/O Domain
- Multiple Guest Domains
- Virtualized devices



Hypervisor Support

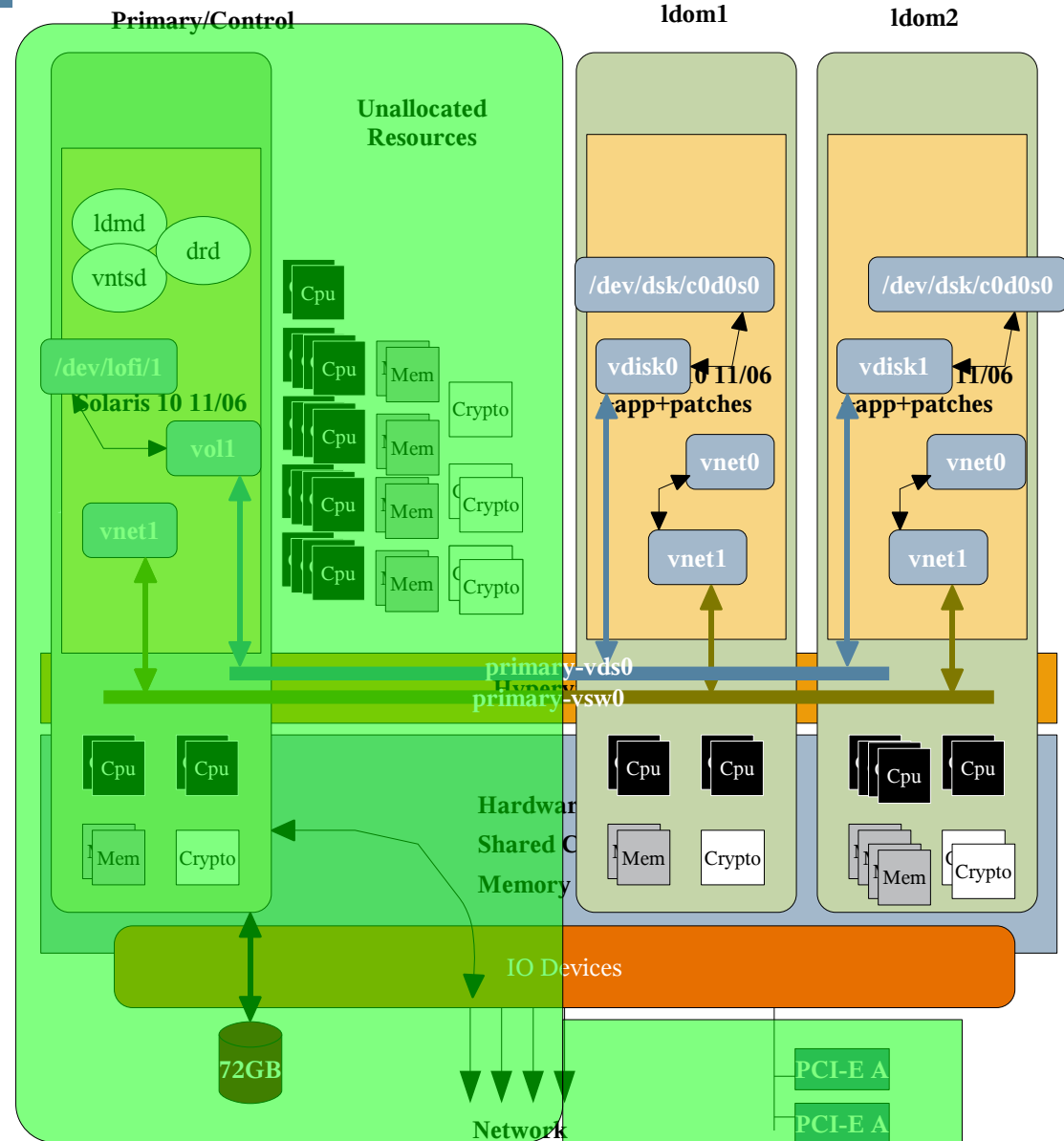
- Hypervisor software/firmware responsible for maintaining separation (eg: visible hardware parts) between domains
 - > Using extensions built into a sun4v CPU
- Provides Logical Domain Channels (LDCs) so domains can communicate with each other
 - > Mechanism by which domains can be virtually networked with each other, or provide services to each other
 - > MMU maps RAM into domains' address spaces, and a protocol lets hypervisor and domains queue and dequeue messages
- Concept of a “service domain” using these channels and owning I/O resources for bridged access
 - > Physical I/O for service domains, virtual I/O for guest domains

Roles of Domains

- All logical domains are the same except for the roles that you specify for them. There are multiple roles that logical domains can perform such as
 - > control domain
 - > service domain
 - > I/O domain
 - > guest domain

Key LDoms components

- The Hypervisor
- The Control Domain
- The Service Domain
- The I/O Domain
- Multiple Guest Domains
- Virtualized devices

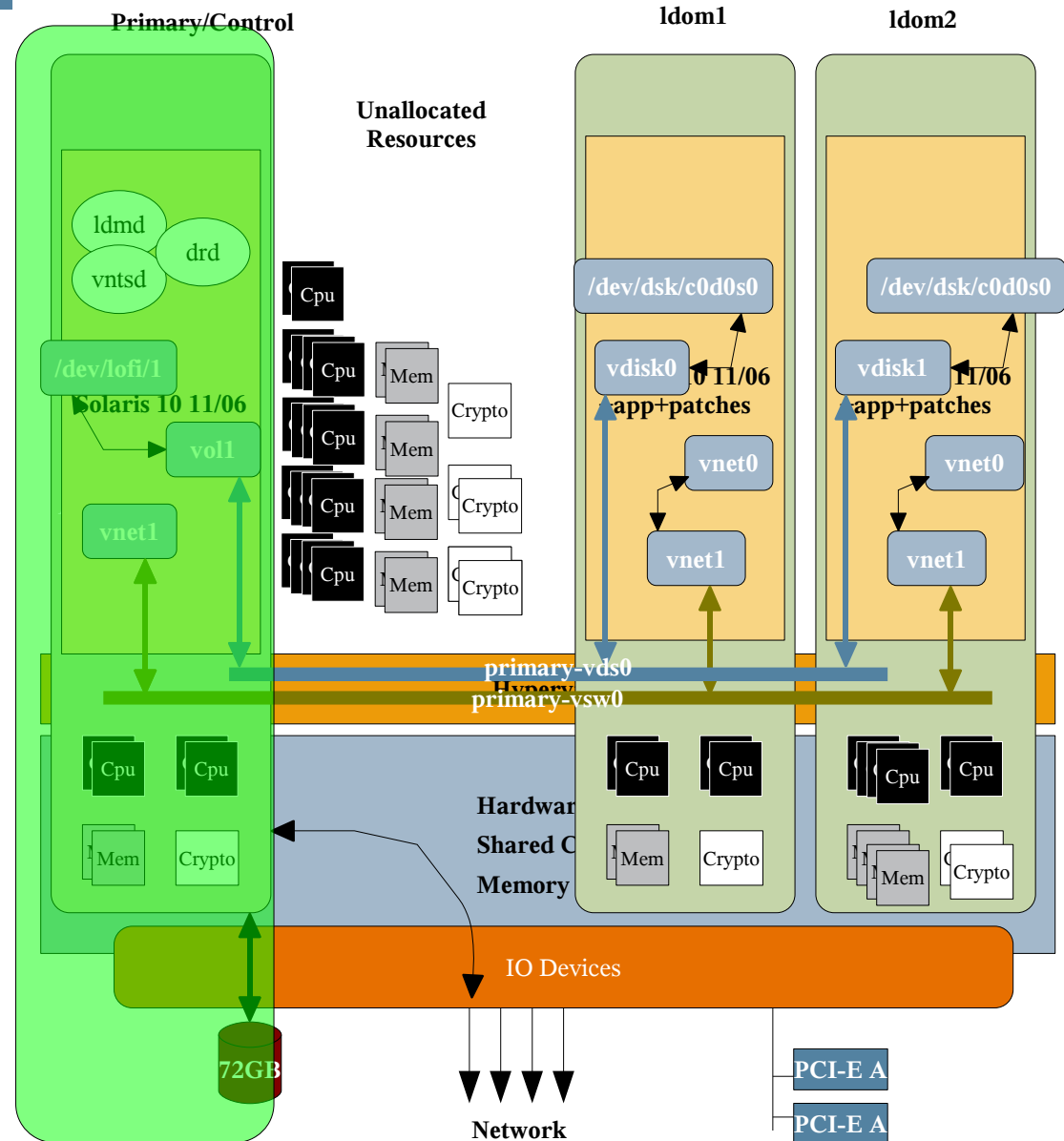


'Control' Domain

- Configuration platform for managing server and domains
- Allows monitoring and reconfiguration of domains
- Interfaces with the hypervisor to set up access rule sets
- Runs the LDom Manager software ('ldm' command and ldmd daemon)
- Administers the constraints engine and resource mapping
- Recommendation: make this domain as secure as possible, and don't add applications that could affect its performance or stability

Key LDom components

- The Hypervisor
- The Control Domain
- The Service Domain
- The I/O Domain
- Multiple Guest Domains
- Virtualized devices

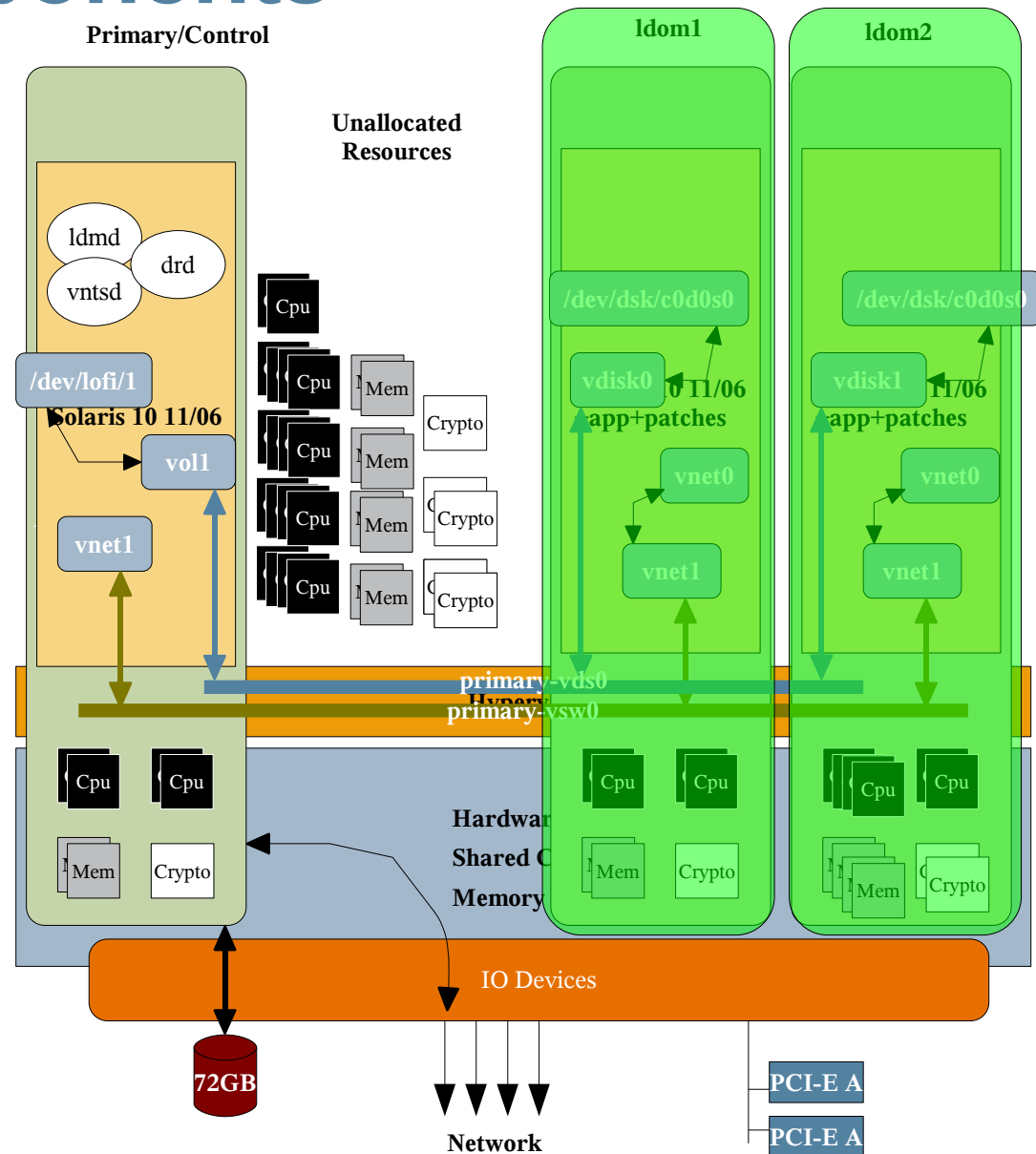


'Service' and 'I/O' Domain

- Service Domain provides virtual device services to other domains such as a virtual switch, a virtual console concentrator, and a virtual disk server
 - > Multiple service domains can exist with shared or sole access to system facilities
- I/O Domain owns the physical I/O and provides I/O facilities to itself and other guest domains
 - > Allows I/O load separation and redundancy within domains deployed on a platform
 - > I/O Domain is usually a Service Domain
- may also act as a 'control' or 'guest' Domain
 - > In current release, control domain is also a service domain and I/O domain
 - > In general, not recommended to run guest applications in the service domain for load and security reasons

Key LDom components

- The Hypervisor
- The Control Domain
- The Service Domain
- The I/O Domain
- Multiple Guest Domains
- Virtualized devices

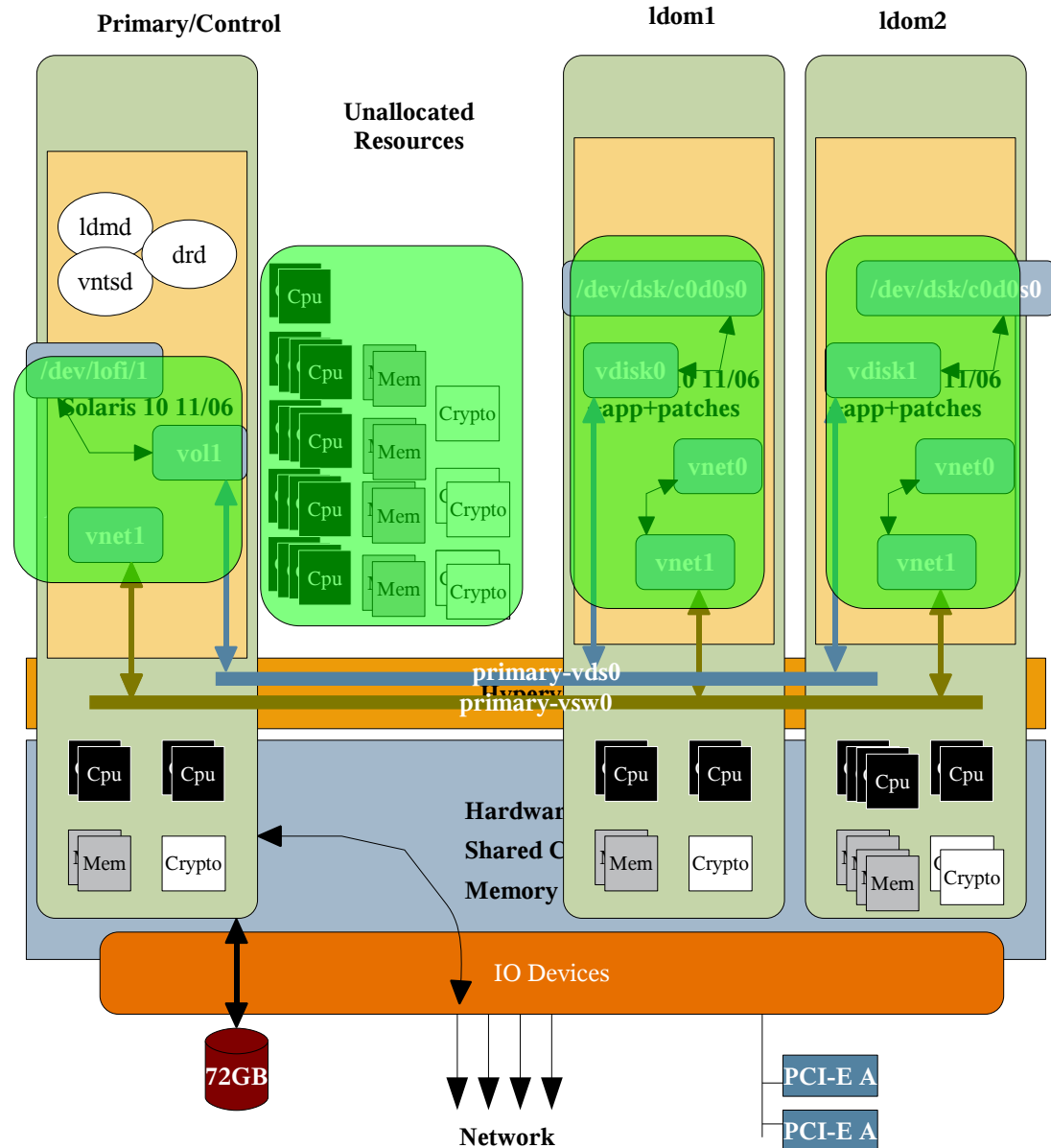


'Guest' Domains

- Where the applications are hosted
- Multiple guest domains can exist
- May use one or more service domains to obtain I/O
- Can be remotely and interactively reconfigured by the primary domain
- Can be independently 'powered up' and rebooted without affecting other domains

Key LDoms components

- The Hypervisor
- The Control Domain
- The Service Domain
- The I/O Domain
- Multiple Guest Domains
- Virtualized devices



Virtual Subsystems

- Virtual devices abstract physical devices
- Inter-Domain I/O via Logical Domain Communication Channels (LDCCs) configured in the Control domain through the hypervisor
- Virtual devices are :-
 - > CPU's
 - > Memory
 - > Crypto cores
 - > Network switches
 - > NICs
 - > Disk servers
 - > Disks
 - > Consoles
 - > A Virtual Terminal Server (vntsd)

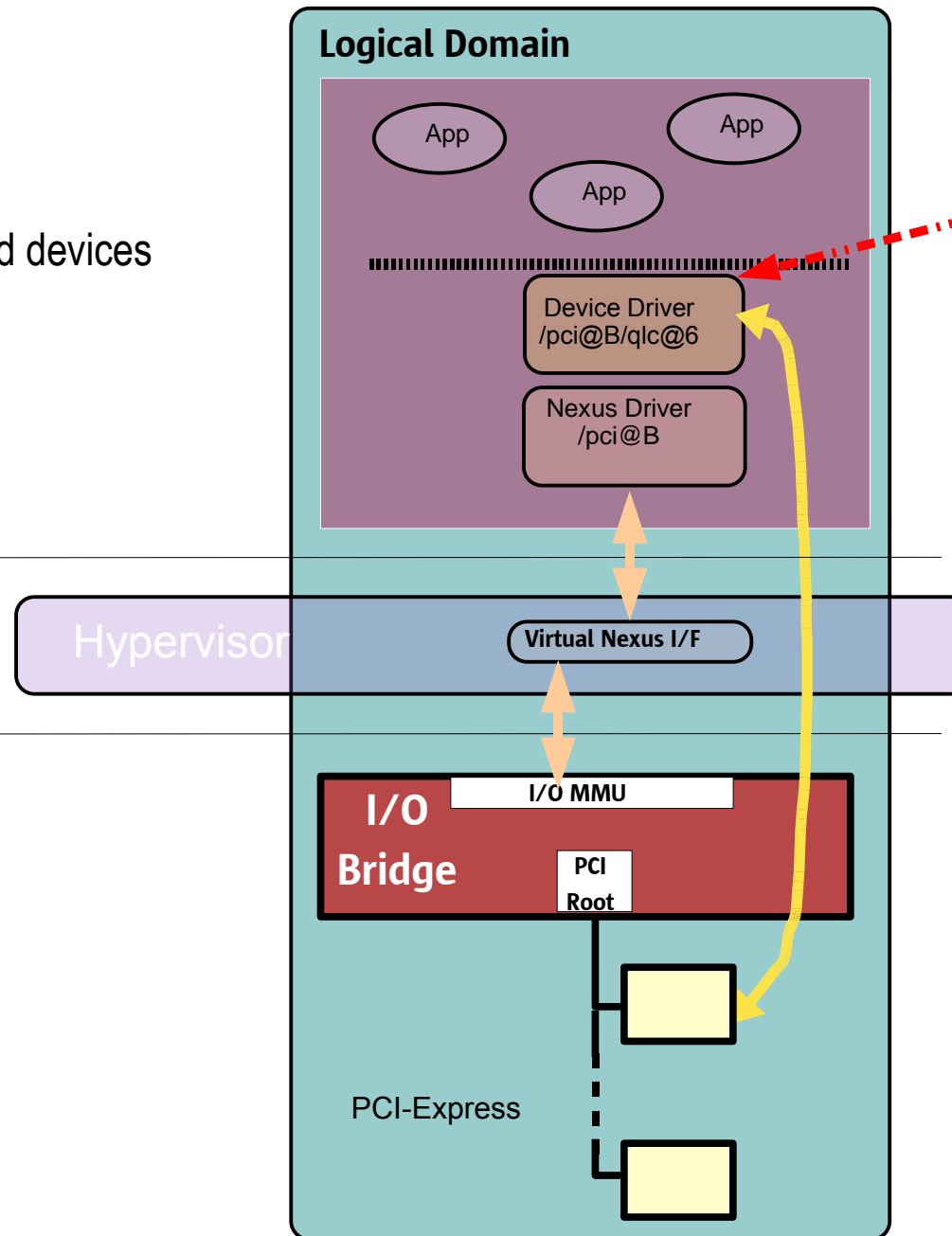
Direct I/O

- Traditional model
 - > Existing drivers and devices continue to work

Privileged

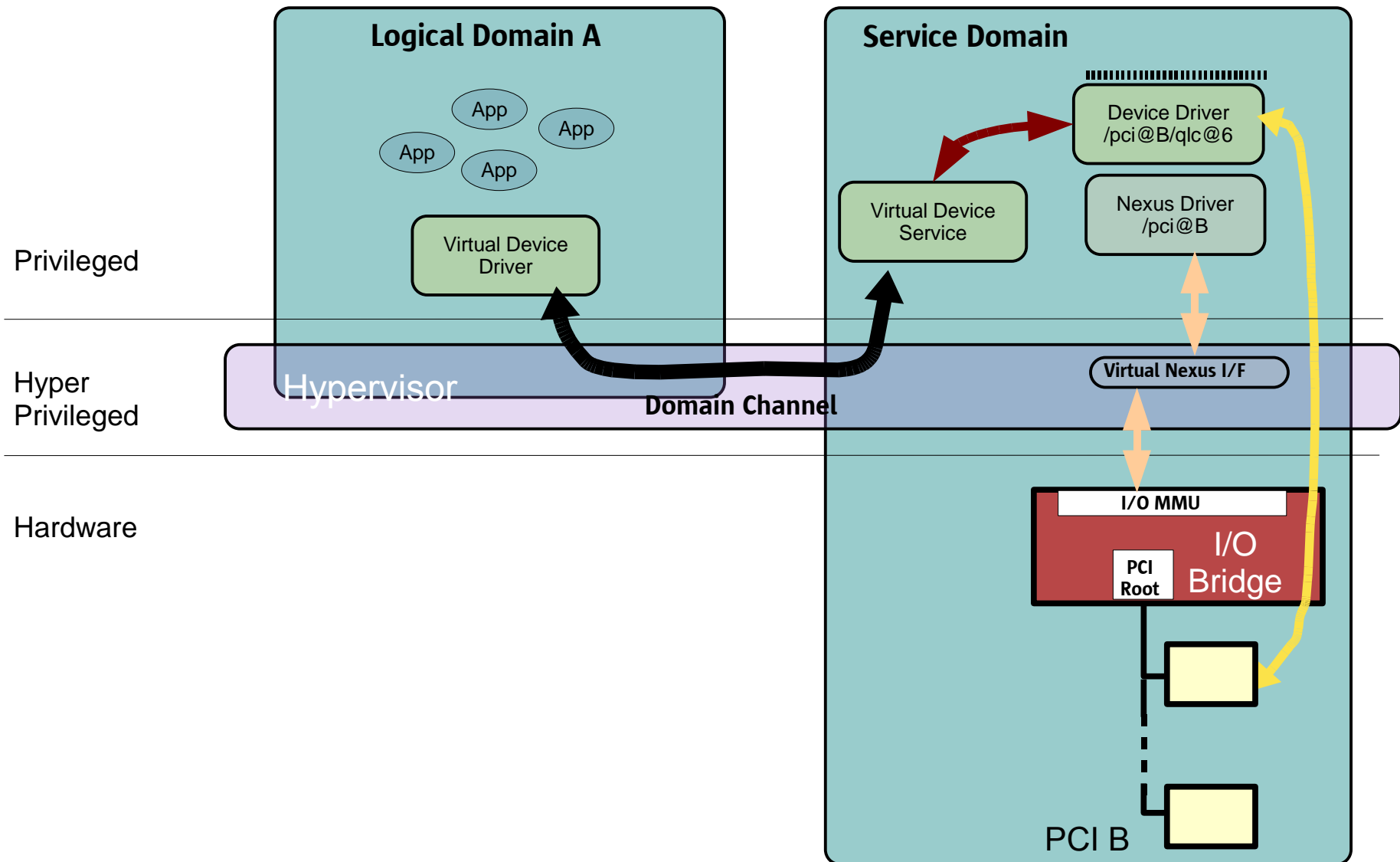
Hyper Privileged

Hardware



Logical Domain owns PCI root and tree

Virtualized I/O



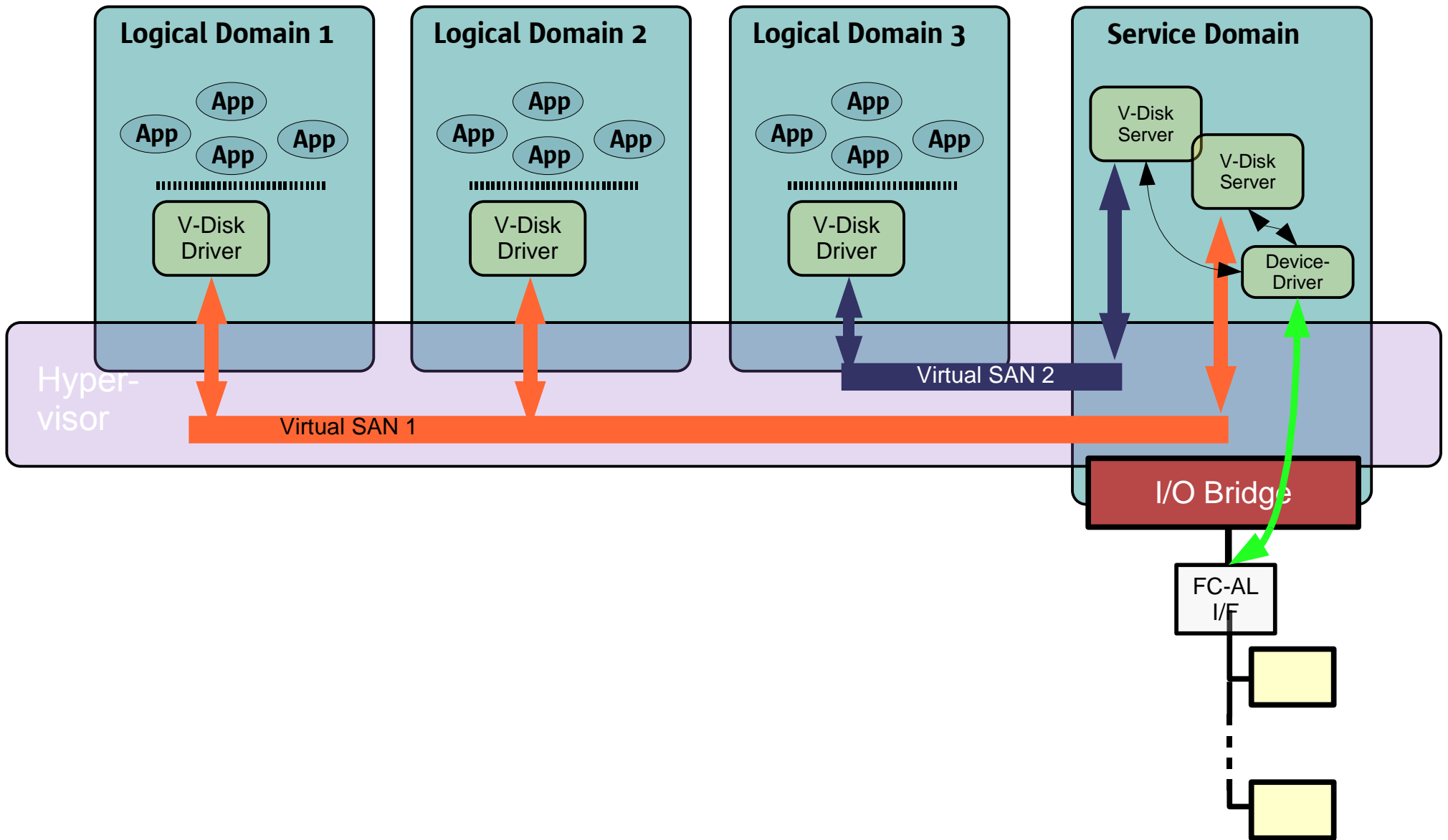
Virtual Disk Server device (vds)

- vDisk vds server virtualizes and exports block devices to virtual disk clients (vdc)
 - > The exported devices can be ...
 - > An entire physical block device (disk/LUN)
 - > A full slice of a physical device or LUN
 - > A disk image file residing in UFS or ZFS
 - > A ZFS volume
 - > A CDROM device (future)
- Using disk image files can enable extremely efficient reuse of disk images and boot environments
 - > Leverage ZFS snapshots and clones to great advantage
- Whole physical devices will perform better

Virtual Disk Client (vdc)

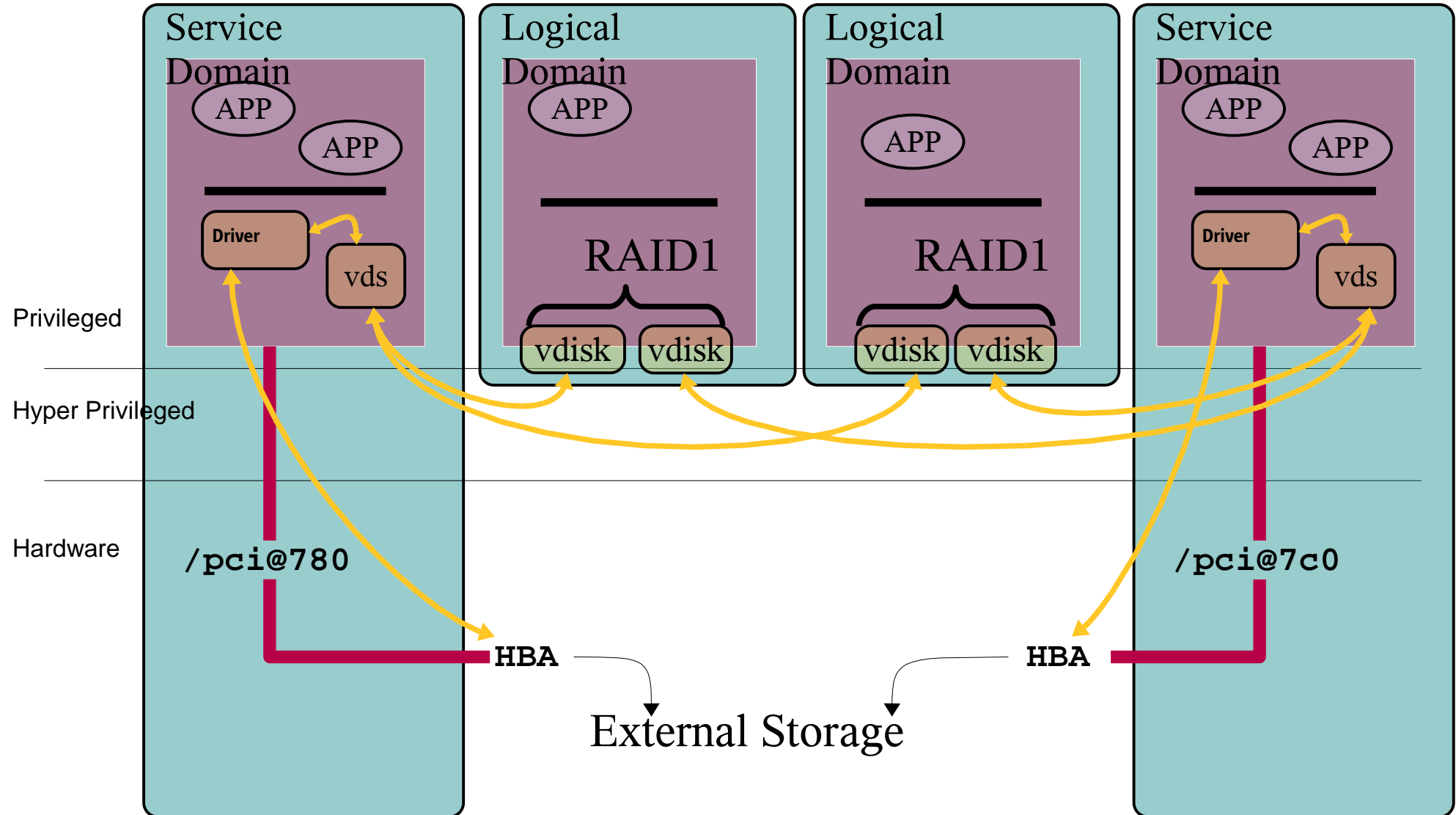
- vdc's give guest domains access to data via a virtual disk server running on a service domain
- 'vdisk's are the objects seen by OBP on guest systems
- Guest domain Solaris sees normal SCSI devices
- Domain administrators may setup devaliases or use raw vdisk devices.
- A future release will provide virtualized access to DVD/CD-ROM in service domains

Virtual (Block) Disk device



Virtualized I/O

Redundant Boot disks for LDom on T2000



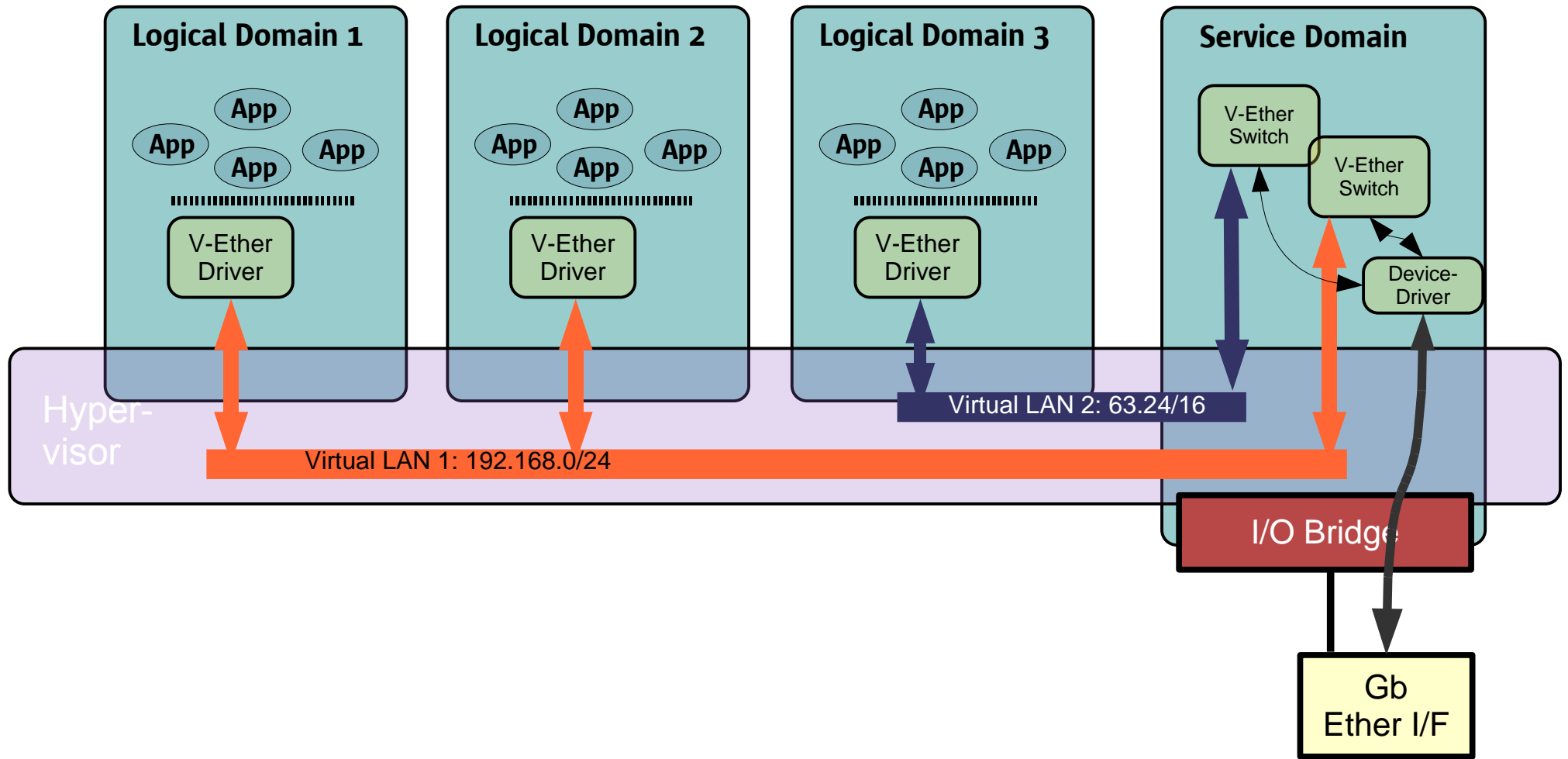
Virtual Network Switch services (vswitch)

- Layer 2 switch - handles unicast and broadcast packets
- configured on service domains as a switch and configuration agent for all domain network traffic
 - > Can be implemented as a deviceless in-memory only switch to communicate between domains
 - > Can be tied to a physical network device, a NEMO compliant device driver for external network connection
- Provides a NEMO compliant driver interface
 - > Use service-domain's kernel for Layer-3
 - > routing, iptable filtering, NAT, firewalling
- Behavior dependant on the config
 - > Physical adapter presents MAC of each vnet device using it.
 - > Routed Mode (no physical NIC specified)

Virtual Network Device (vnet)

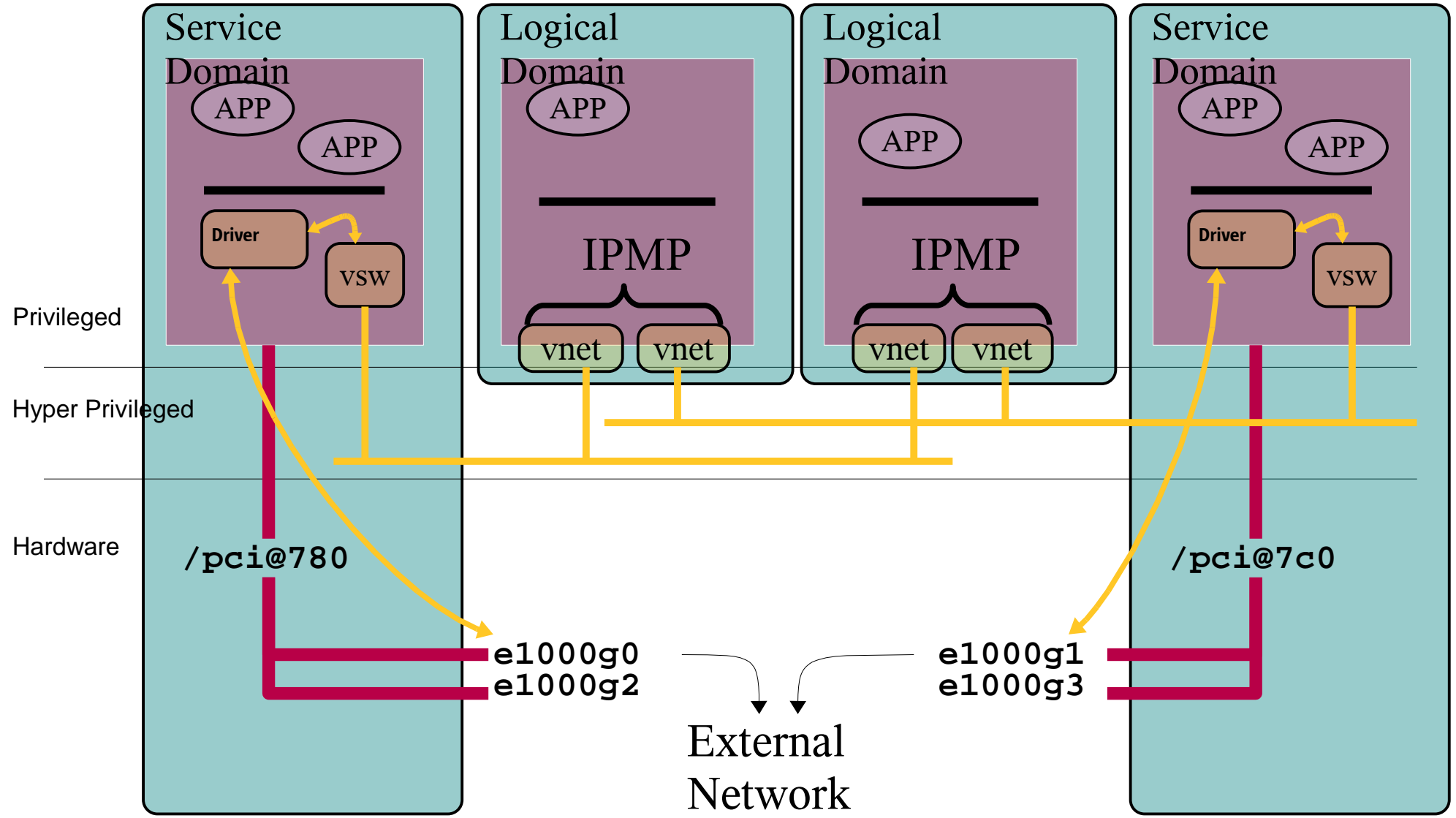
- Implements an ethernet device in a domain
 - > Communicates with other vnets over the vswitch devices
- If the vSwitch is configured correctly packets can be routed out of the server.
- vnet looks like a NEMO driver
 - > Exports NEMO mac interface
 - > Uses multi-protocol transports for send and receive
 - > Uses a distributed switching infrastructure to send unicast packets directly to other vnets rather than through a central switch

Virtual Ethernet device



Virtualized I/O

Redundant Network on T2000



Virtual Console Concentrator (vcc)

- Service domain VCC driver communicates with all guest console drivers over the Hypervisor
- Makes each console available as a tty device on the Control/Service domain
- Solaris and OBP Console clients use existing hooks to send and receive console communications over the Hypervisor

Virtual Network Terminal Server daemon (vntsd)

- Operates as a daemon service on the Control/Service domain
- Provides a standard `telnet <ip> <port>` interface to domain serial consoles
 - > Accessible once a domain is configured and bound
 - > Attach prior to domain start to watch domain OBP boot sequence
- Only one user at a time can view a serial console using this mechanism
- Not visible outside the control/service domain by default

Memory

- Memory is configured through the Control Domain
- Minimum allocatable unit is 8KB
 - > Minimum size is 12MB (for OBP)
 - > Though most OS deployments will need > 512MB
- This is a delayed configuration property (requires domain boot to take effect)

Crypto Units

- Each T2 physical CPU core has a crypto unit
 - > “MAU” - Modular Arithmetic Unit, accelerates crypto
 - > 8 in total on a 8 core system
- MAU can only be allocated to domains that have at least one vcpu (thread) on the same physical core as the unit
- Probably best to allocate all four threads on a core to a domain that wants to use the MAU
 - > Threads on one core can only use a MAU on their own core
- Re-configurations are delayed until domain reboot

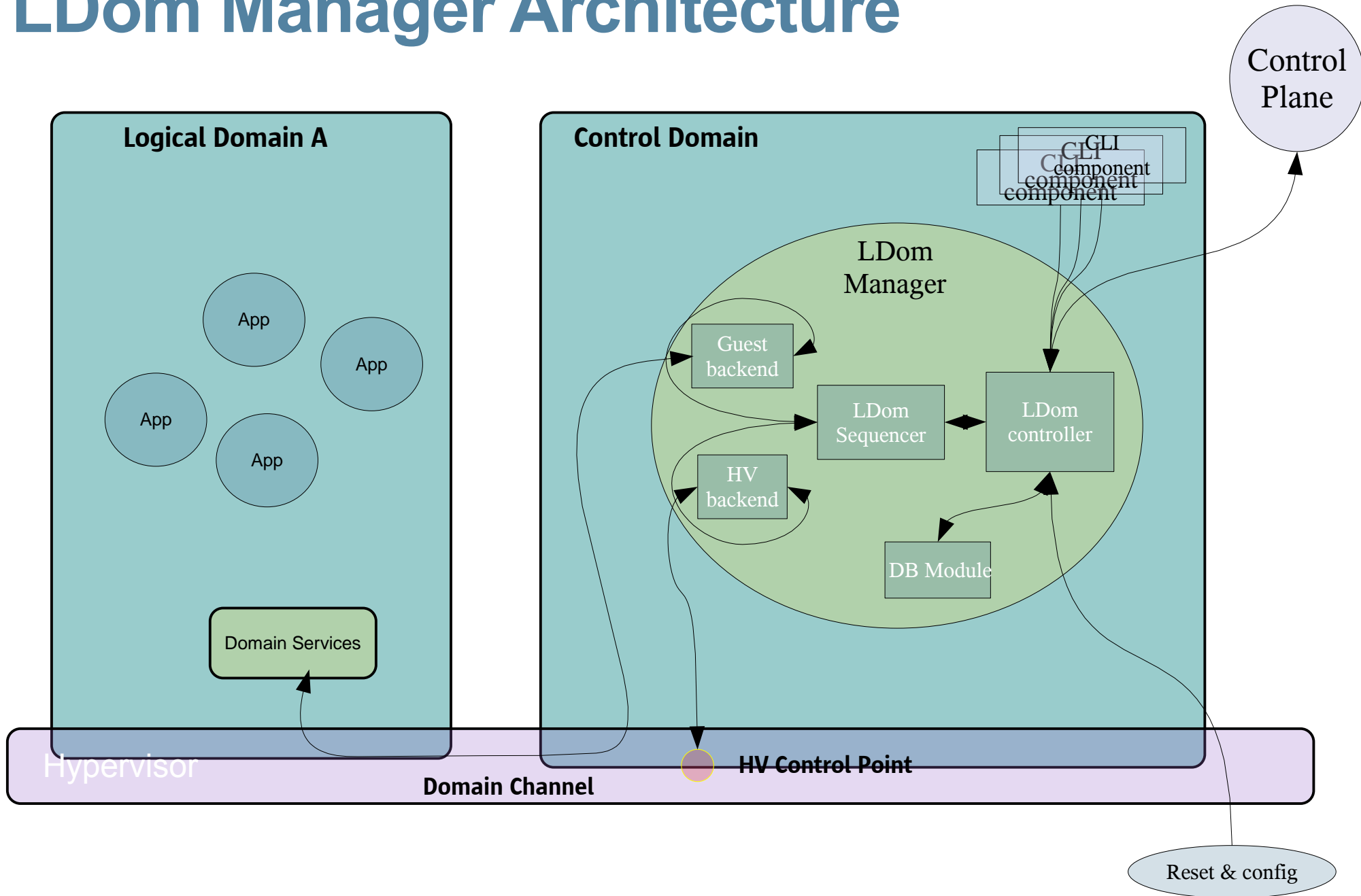
vCPU's

- Each UltraSPARC T1 has up to 8 physical cores with 4 threads each
 - > Each thread is a vCPU, so up to 32 vCPUs
- Granularity is 1 vCPU per domain (a domain can have between 1 and 32 vCPU's)
- vCPU's are allocated to one Domain at a time.
 - > So up to 32 domains on UltraSPARC T1 systems
- Can be dynamically allocated with the domain running, **adding or removing a vcpu to or from a running domain**

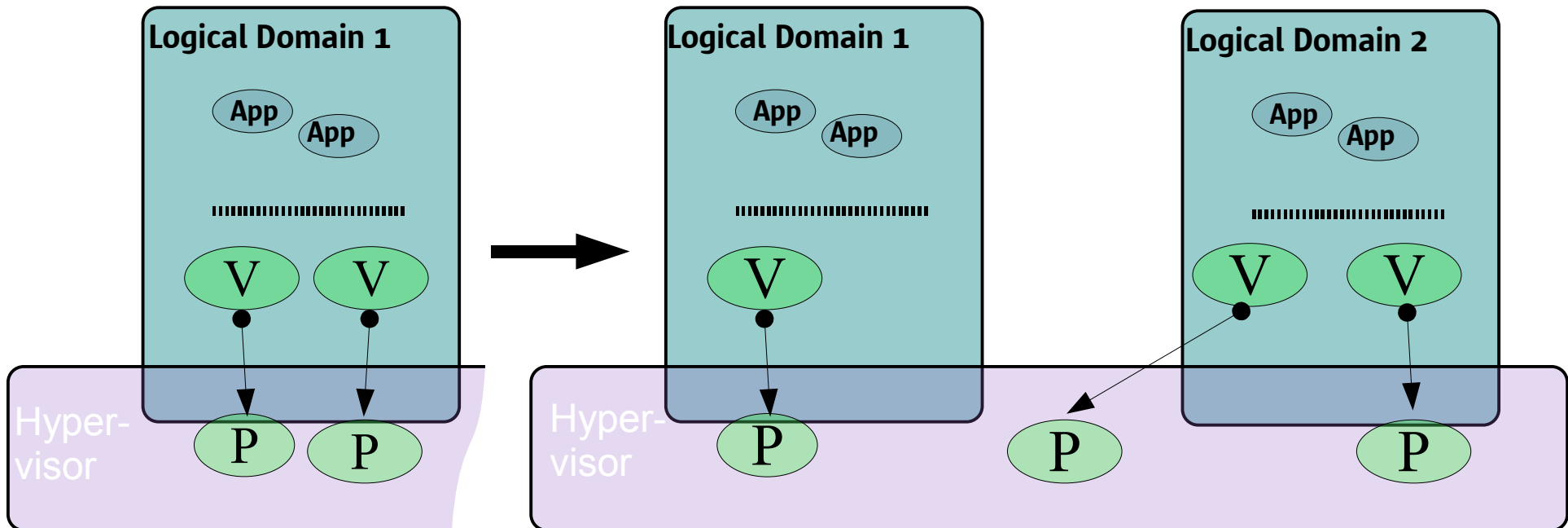
LDom Manager

- One Manager per machine
 - > Can be run in any domain, but only 1 domain at a time
 - > The “Control Domain”
 - > Controls Hypervisor and all the LDomS
- Exposes CLI to administrator
- Maps Logical Domains to physical resources
 - > Constraint engine
 - > Heuristic binding of LDomS to resources
 - > Assists with performance optimization
 - > Assists in event of failures / blacklisting

LDom Manager Architecture



Example: vCPU reconfiguration



Example command line operations:

```
# ldm remove-vcpu 1 domain1
# ldm add-vcpu 1 domain2
```

Requirements for LDom

- UltraSPARC T1 processor based systems
 - > Actually, CMT processors: so T2, Rock etc, in the future
- LDom package SUNWldm installed
- Server configured with good I/O diversity to allow flexibility/performance of the LDom config
- Use all means to secure control/service domains
- Choose suitable server redundancy mechanisms
 - > Duplication, snapshots, SAN boot images, clustering
- Make well planned LDom configurations

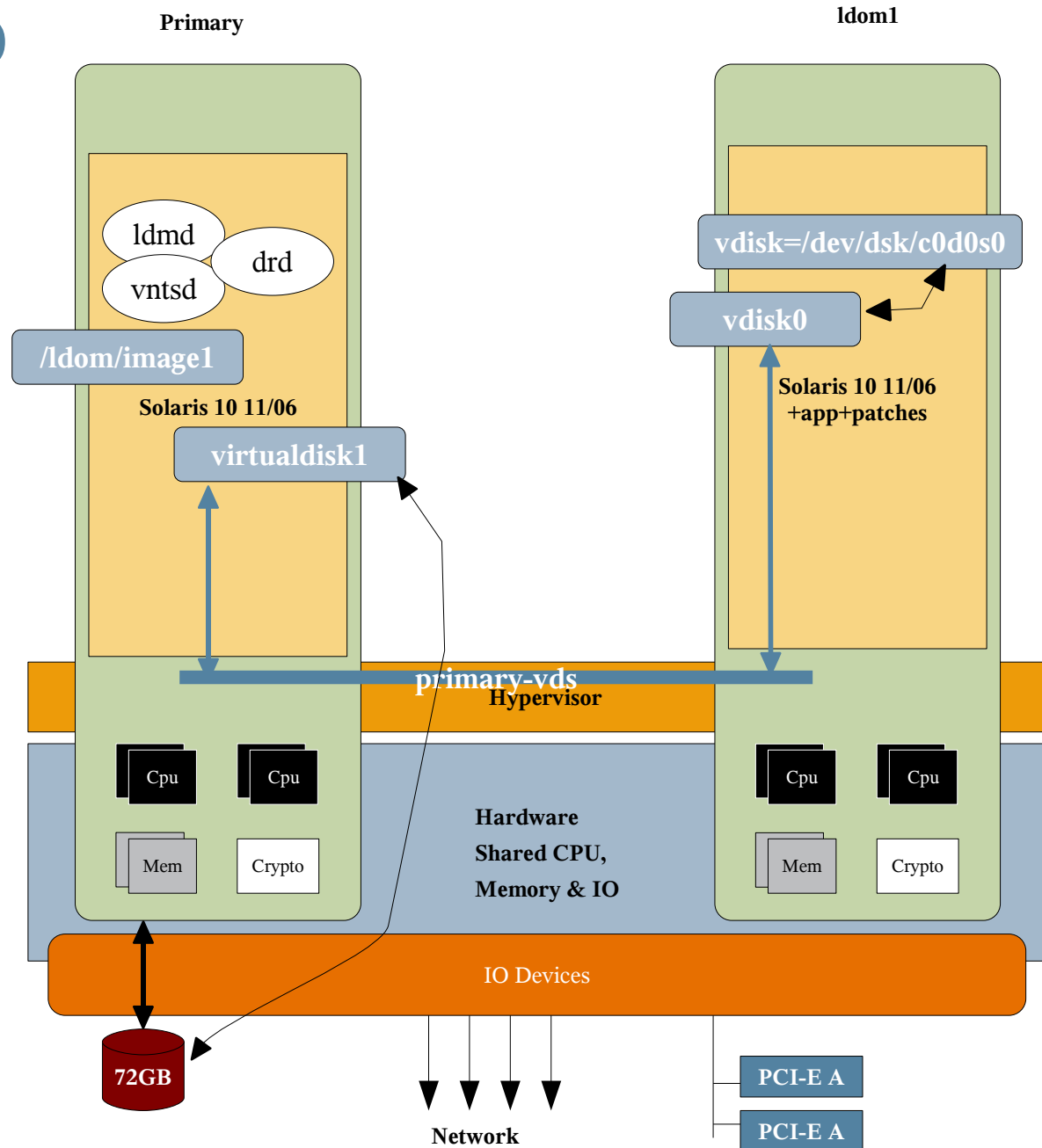
Naming Conventions

..or How to confuse yourself...

- Choose LDom component names carefully
 - > Lots of things begin with the letter “v”... read names carefully
 - > Bad choices can be very confusing later on...
 - > Only number devices where you have more than one...
- You need names for ...
 - > Disk Servers
 - > Network Virtual Switches
 - > Domains
 - > **Not for devices**, these are automatically enumerated in the guest domains...
- Service and device names are only known to the control and service domains
 - > Guest domains just see virtual devices.

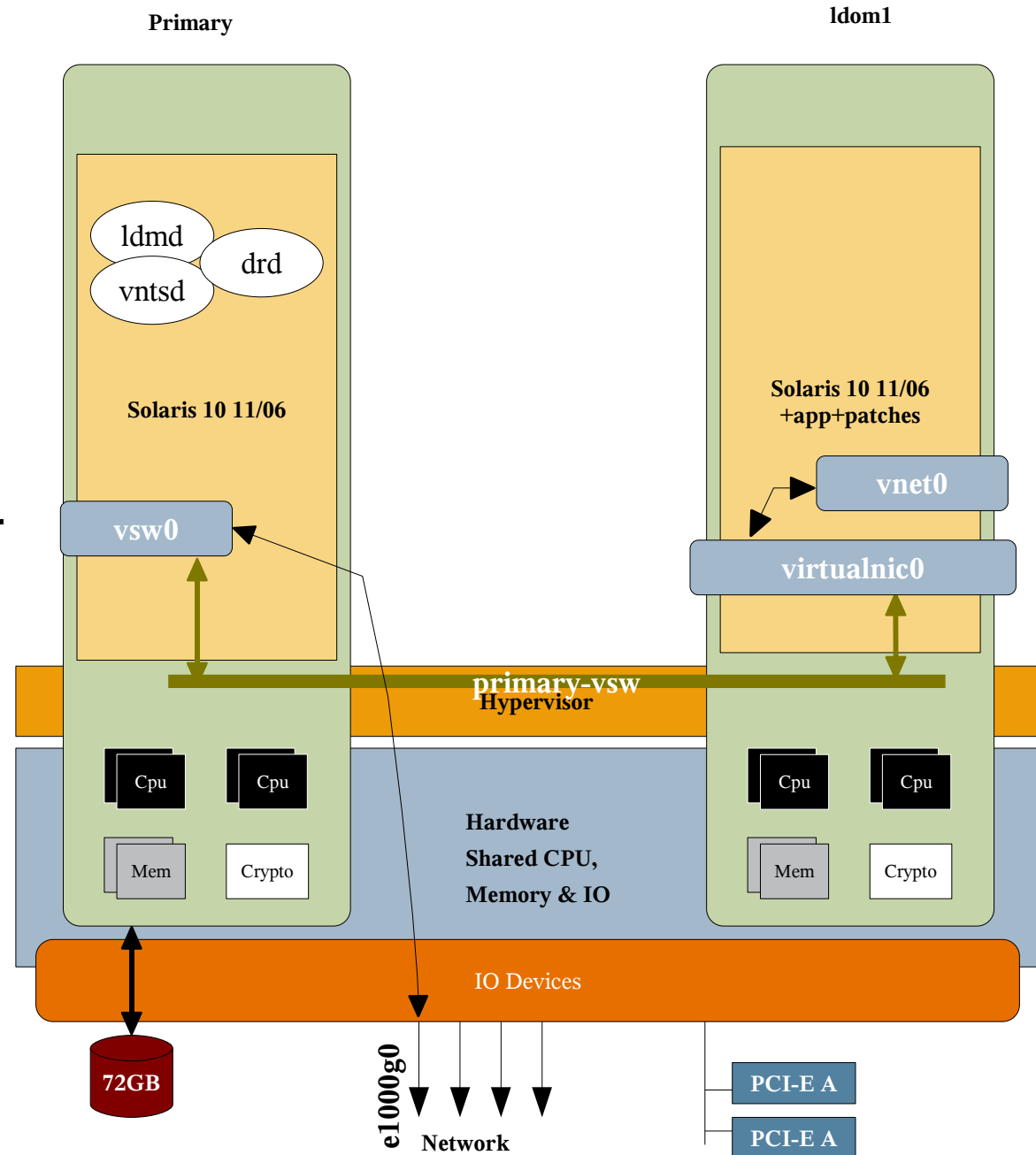
Disk Service Setup

- Establish a Virtual Disk Service
 - > 'primary-vds'
- Associate it with some form of media.
 - > A disk image at '/ldom/image'
 - > (or /dev/dsk/c0d0s0 for example)
- Create disk server device that can be exported to guest domains
 - > 'vol1@primary-vds'
- Give the server device a name and associate it with a guest domain.
 - > 'virtualdisk1'



Network Setup

- Establish a Virtual Network Switch Services
 - > 'primary-vsw'
 - > Automatically associated with a vsw device instance
 - > 'vsw0'
- Associate it with some form of media.
 - > 'e1000g0' a real NIC
 - > or no NIC .. in memory
- Create a network device instance to provide to guest domains
 - > 'virtualnic0'
- The guest domain will enumerate the device automatically
 - > 'vnet0'



Example Primary LDom setup

Setting up the basic services and config

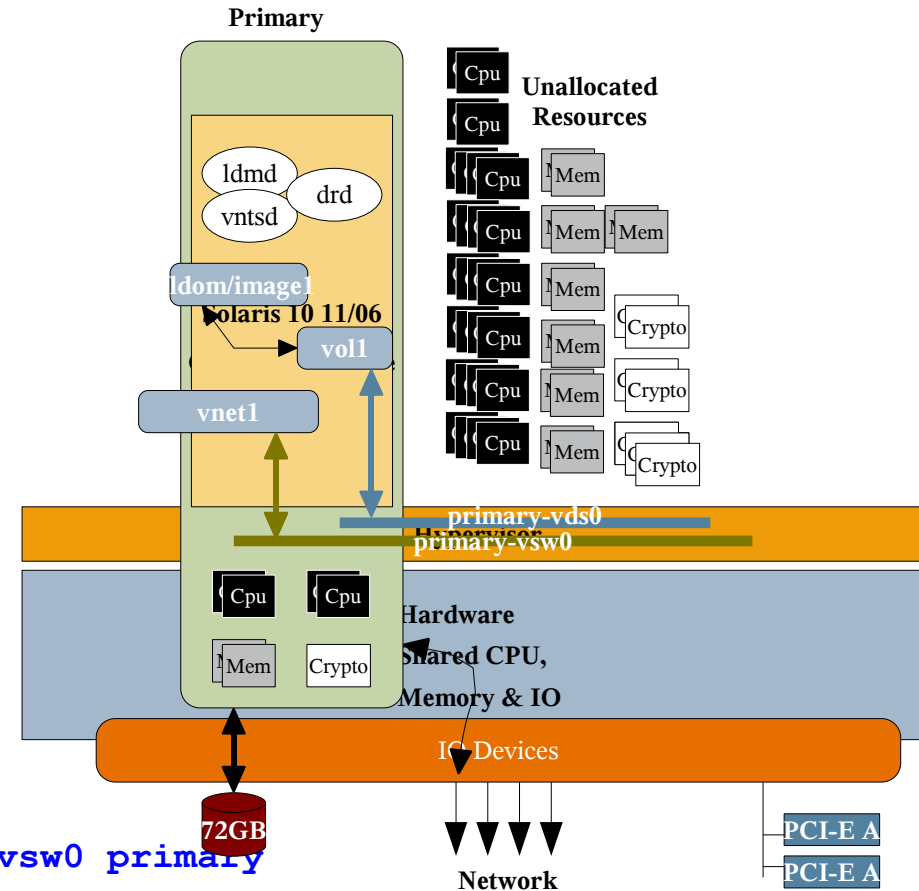
- **Control Domain – 'primary'**
 - > 4 x vCPU's
 - > 1 x Crypto units
 - > 1 GB memory
 - > 1 x vsw vSwitch attached to e1000g0
 - > Using e1000g0 network device natively
 - > Installed on, and booting from physical disk0
 - > 1 x vds to provide disk services to other domains
 - > vcc and vNTSd to access domain consoles
 - > Owns all IO devices

28 vCPU's, 7 Crypto units, 31GB, 3 x NICs not used

How to set that up...

- On our 'Primary' Domain do the following ...
- Ensure that the system is properly installed first
- In this case we will combine the Control and Service
- Set up the basic services needed..

- > #. <upgrade the ALOM/OBP/Hypervisor>
 - > flashupdate -s <ftp_ip> -f <firmware.bin>
- > pkgadd SUNWldm
- > svcadm ldmd start
- > ldm list
- > ldm add-vdiskserver primary-vds0 primary
- > ldm add-vswitch net-dev=e1000g0 primary-vsw0 primary
- > ldm add-vconscon port-range=5000-5100 primary-vcc0 primary
- > ldm set-crypto 1 primary
- > ldm set-vcpu 4 primary
- > ldm set-mem 1g primary
- > ldm add-config initial
- > shutdown -i6 -g0 -y
- > svcadm enable vntsd



Example LDOM Configurations

Add a single guest domain

- **Control and Service Domain**

- > 4 x vCPU's
- > 1 x Crypto units
- > 1 GB memory
- > 1 x vsw0 vSwitch attached to e1000g0
- > e1000g0 plumbed and active
- > Physical disk0 to boot from
- > 1 x vds0 configured to share a LOFI disk image as vol1
- > vcc and vNTSd ports 5000 upwards
- > Owns all IO devices

- **Guest Domain**

- > 4 x vCPU's
- > 1 x Crypto unit
- > 1 GB memory
- > 1 x vnet0 over vsw0
- > 1 x vdisk0 (via vds0 device vol1)
- > Installed/config'd application
- > S10/ Patches applied

24 vCPU's, 6 Crypto units, 30GB, 3 x NICs not used

How to set that up...

- On the Primary Domain do the following ...
- Ensure that the system is properly installed first
- We will use a disk image on the primary domain..

```
> /ldoms_images/ldom1.disk1
```

- Create our new domain description

```
> ldm create ldom1
> ldm set-vcpu 4 ldom1
> ldm set-mem 1g ldom1
> ldm set-crypto 1 ldom1
> ldm add-vnet vnet1 primary-vsw0 ldom1
> ldm add-vdiskserverdevice /ldoms_images/ldom1.disk1 voll@primary-vds0
> ldm add-vdisk vdisk0 voll@primary-vds0 ldom1
> ldm bind ldom1
```

- Now just need to start the domain...

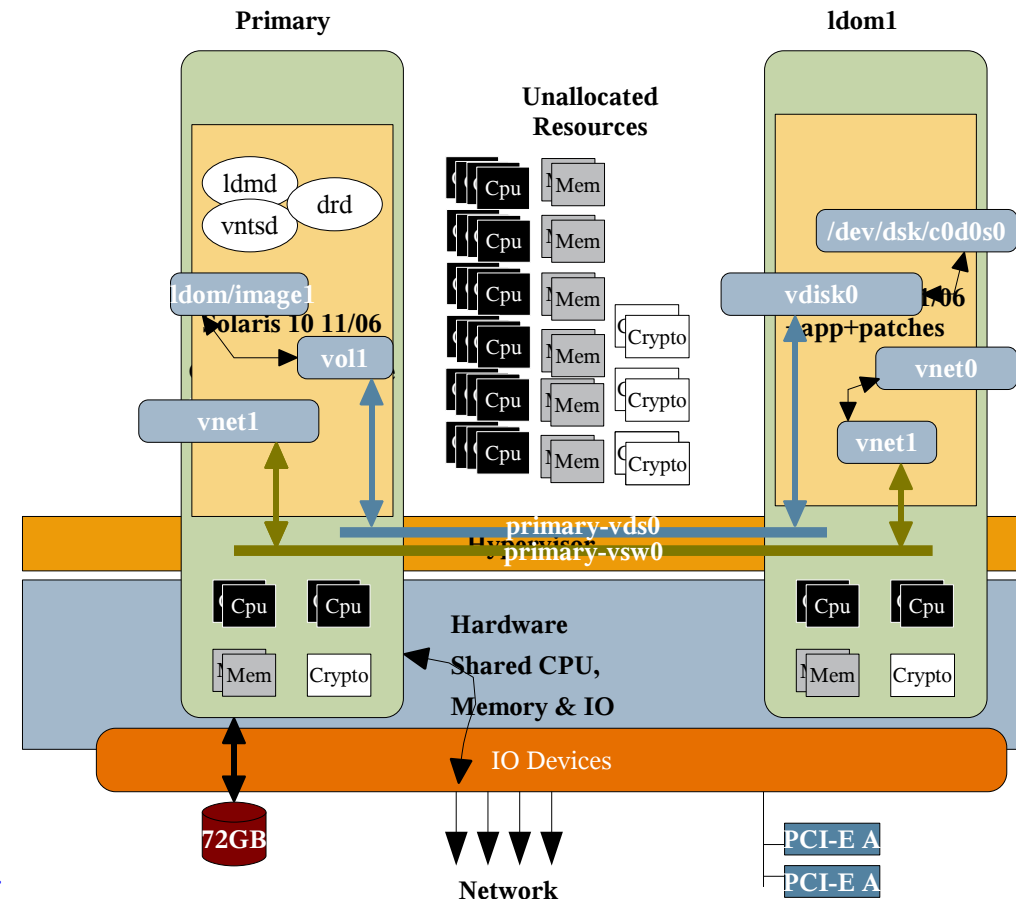
```
> ldm start ldom1
```

- Create an ldom set name 'domain-set0'

```
> ldm add-config domain-set0
```

- Watch the console of ldom1 using ...

```
> telnet localhost 5000
```



Example LDOM Configurations

Adding a second, bigger domain

- **Control & Service Domain**

- > 4 x vCPU's
- > 1 x Crypto units
- > 1 GB memory
- > 1 x vsw0 vSwitch attached to e1000g0
- > 1 x Physical disk0
- > 1 x vds0
- > vcc and vNTSd ports 5000+
- > S10/JASS applied

- **Guest Domain 1**

- > 4 x vCPU's
- > 1 x Crypto units
- > 1 GB memory
- > 1 x vnet0 over vsw0
- > 1 x vdisk0 over vol1 (LOFI device)
- > S10 and app installed

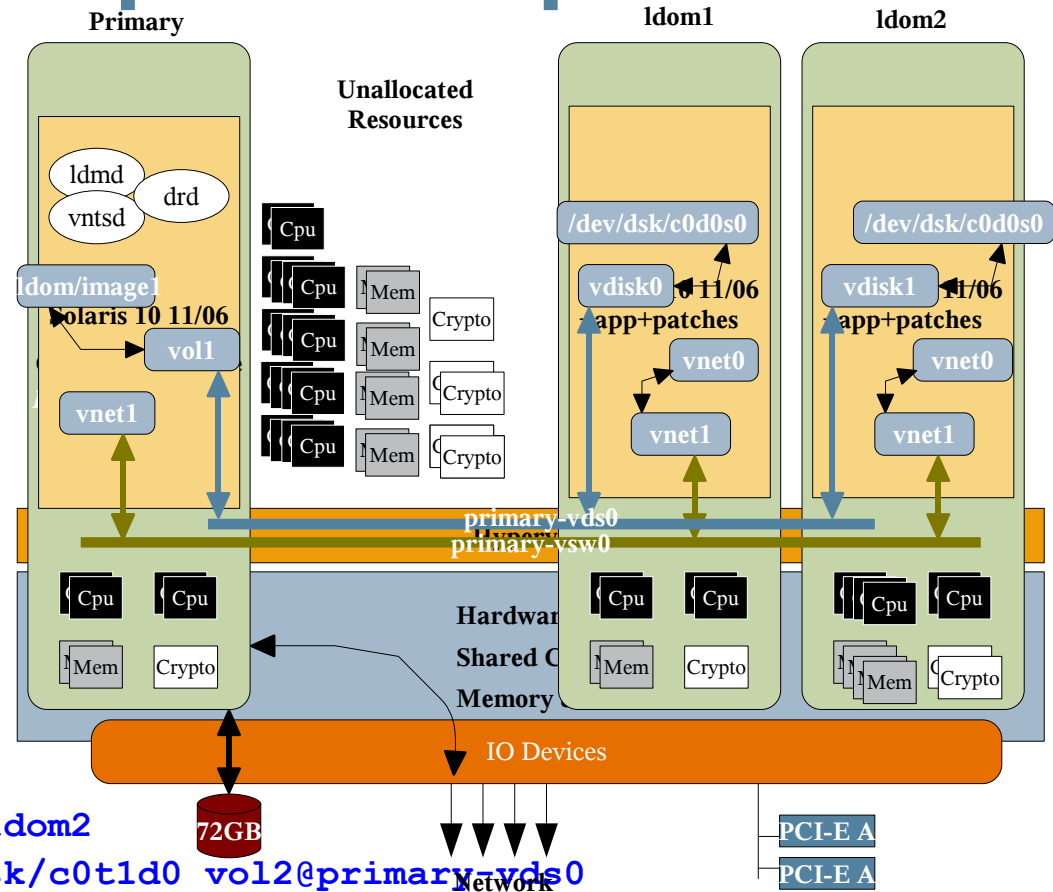
- **Guest Domain 2**

- > 6 x vCPU's
- > 2 x Crypto units
- > 8 GB memory
- > 1 x vnet0 over vsw0
- > 1 x vdisk0 (real disk via vds0)
- > S10 app, patches

18 vCPU's, 4 Crypto units, 22GB, 3 x NICs not used

A slightly more complex setup..

- Add a second Domain ...
- ..with more vCPU's (6 this time)
- ..and more crypto units (might as well)
- ..and more memory
- Provide it with a real disk this time
- Call this domain ldom2
 - > `ldm create ldom2`
 - > `ldm set-vcpu 6 ldom2`
 - > `ldm set-mem 8g ldom2`
 - > `ldm set-crypto 2 ldom2`
 - > `ldm add-vnet vnet1 primary-vsw0 ldom2`
 - > `ldm add-vdiskserverdevice /dev/dsk/c0t1d0 vol2@primary-vds0`
 - > `ldm add-vdisk vdisk0 vol2@primary-vds0 ldom2`
 - > `ldm bind ldom2`
- Watch the console of ldom2 using ...
 - > `telnet localhost 5001`



Getting started with LDoms – up close

- First, get a T1000 or T2000
- Install Solaris 10 Update 3, Build 8 or later
 - > Sun Download Center
- Update firmware to LDoms level (6.4 or later)
- Install necessary patches and packages
- Ensure services are running and define initial config
- Reboot to activate that configuration
- Create and boot your first logical domain

System Identification and Update

- Check the Service Processor of your system for firmware levels

```

sc> showhost
Sun-Fire-T2000 System Firmware 6.4.2 2007/04/02 18:07

Host flash versions:
Hypervisor 1.4.1 2007/04/02 16:37
OBP 4.26.1 2007/04/02 16:26
POST 4.26.0 2007/03/26 16:45
sc>

```

Check SC Firmware
version 6.4.x

- Upgrade your system firmware if needed...

```

sc> poweroff -yf
sc> flashupdate -s <ftp ip>
-f /<path_to>/Sun_System_Firmware-6_4_2-Sun_Fire_T2000.bin
sc> resetsc -y
sc> poweron
sc> console -f
{0} ok boot

```

System Identification and Update

- Check the OS version and patch level of your system

```
-bash-3.00# uname -a
SunOS t21domdemo 5.11 snv 61 sun4v sparc SUNW,Sun-Fire-T200
-bash-3.00# pkginfo -x |grep ldm
SUNWldm Logical Domains Manager
(sparc.sun4v) 1.0,REV=2007.04.02.21.17
-bash-3.00# pkginfo-x |grep jass
-bash-3.00#
```

Check OS version is
LDOMS aware..
Nevada Build 61

or

Solaris 10 10/06 +
patch 124921 + patch
125043

=> KU 118833-36

Check for the
SUNWldm package
and

Recommend applying
the SUNWjass security
patches/minimisation

- Upgrade the OS level or add patches/packages

```
-bash-3.00# patchadd 124921-xx
-bash-3.00# patchadd 125043-xx
-bash-3.00# pkgadd -d SUNWldm
-bash-3.00# pkgadd -d SUNWjass

or

-bash-3.00# gunzip -c Ldoms_manager_1.0.tar.gz |tar xvf -
-bash-3.00# ./LDoms_manager_1.0/Install/install_ldm
...
```

Activation and Control domain setup

- Ensure that the install packages and services are running

```
-bash-3.00# svcadm enable ldmd
-bash-3.00# svcadm enable vntsd
-bash-3.00# /opt/SUNWldm/bin/ldm list
Name      State   Flags  Cons  VCPU  Memory  Util  Uptime
primary   active -n-cv  SP    32    32G     0.5%  6d 3h 29m
-bash-3.00#
```

Ensure both the `ldmd` and `vntsd` daemons are running

- Now LDoms-capable with a single domain owning all resources: set up the initial config of your control domain

```
-bash-3.00# psrinfo -vp
The physical processor has 32 virtual processors (0-31)
UltraSPARC-T1 (cpuid 0 clock 1200 MHz)
-bash-3.00# ldm add-vdiskserver primary-vds0 primary
-bash-3.00# ldm add-vconscon port-range=5000-5100 \
primary-vcc0 primary
-bash-3.00# ldm add-vswitch net-dev=e1000g0 primary-vsw0 primary
-bash-3.00# ldm list-services

..
-bash-3.00# ldm set-mau 1 primary
-bash-3.00# ldm set-vcpu 4 primary
-bash-3.00# ldm set-memory lg primary
-bash-3.00# ldm add-config initial
-bash-3.00# ldm list-config
factory-default [current]
initial [next]

-bash-3.00# shutdown -y -g0 -i6
```

Configure a control domain and set the configuration for use in subsequent reboots. Once all the changes have been queued up you must reboot the control domain to make the config active.

Checking the initial configuration

- Following reboot check out the system configuration

```

-bash-3.00# psrinfo -vp
The physical processor has 4 virtual processors (0-3)
  UltraSPARC-T1 (clock 1200 MHz)
-bash-3.00# ldm list-bindings primary
Name:      primary
State:     active
Flags:     normal,control,vio service
OS:        Solaris running
Util:      0.6%
Uptime:    6d 3h 45m
Vcpu:      4
           vid      pid      util  strand
           0         0        13%   100%
           1         1        13%   100%
           2         2        13%   100%
           3         3        12%   100%
Memory:    1G
           real-addr      phys-addr      size
           0x8000000      0x8000000      1G
..
..
..
Vcc:       primary-vcc0
           [LDC: 0xf]
           [LDom ldom1, group: ldom1, port: 5000]
           port-range=5000-5100
Vsw:       primary-vsw0
           mac-addr=0:14:4f:fa:85:bb
           net-dev=e1000g0
           [LDC: 0xc]
           [LDom ldom1, name: vnet1, mac-addr: 0:14:4f:f8:f:12]
Vcons:     SP
-bash-3.00#

```

There is a large amount of output from this command...
 Notice in particular

- No. of vcpu's
- Amount of memory
- vds configurations
- vsw configurations
- vcc configurations
- IO configurations (pci busses controlled by this control/service domain)
- etc...

Creating your first guest domain

- Decide on a configuration and build it...

```
-bash-3.00# ldm add-domain myldom1
-bash-3.00# ldm set-vcpu 4 myldom1
-bash-3.00# ldm set-mau 1 myldom1
-bash-3.00# ldm set-memory 1g myldom1
-bash-3.00# ldm add-vnet vnet1 primary-vsw0 myldom1
-bash-3.00# ldm add-vdiskserverdevice /images/ldom1.disk \
voll@primary-vds0
-bash-3.00# ldm add-vdisk vdisk1 voll@primary-vds0 myldom1
-bash-3.00# ldm set-variable autoboot\?=false myldom1
-bash-3.00# ldm set-variable boot-device=/virtual-devices@100\
/channel-devices@200/disk@0 myldom1
-bash-3.00# ldm bind-domain myldom1
-bash-3.00# ldm start-domain myldom1
```

- Lets also watch the new domain boot ...

```
-bash-3.00# telnet localhost 5000
Trying 127.0.0.1...
Connected to localhost....
Escape character is '^}i'.
Connecting to console "myldom1" in group "myldom1" ....
Press ~? for control options ..

{0} ok
```

The following items must be configured for your new domain..

- No. of vcpus
- Amount of memory
- Network devices
- A disk device
- ...and the corresponding disk server in the primary domain

Once the config is complete, it is bound together and can be started

Before you start the domain, attach to its console port and watch it boot
Notice, domains boot very quickly from disk images

Take a look around your Ldom

- Take a look around OBP ...

```
{0} ok show-disks
a) /virtual-devices@100/channel-devices@200/disk@0
q) NO SELECTION
Enter Selection, q to quit:q

{0} ok show-nets
a) /virtual-devices@100/channel-devices@200/network@0
q) NO SELECTION
Enter Selection, q to quit:q

{0} ok banner
Sun Fire T200, No Keyboard
Copyright 2007 Sun Microsystems, Inc. All rights reserved.
OpenBoot 4.26.1, 1024 MB memory available, Serial #66588434.
Ethernet address 0:14:4f:f8:f:12, Host ID: 83f80f12.

{0} ok boot vdisk
...
```

OBP behaves just like OBP on previous SPARC systems, you can use your favorite commands to explore the system

Useful to learn paths to disk/net devices and perhaps create an alias

Also note how quickly Solaris Boots on such a configuration of LDOM

Take a look around your domain

- Watch the new domain boot Solaris

```
{0} ok boot vdisk
Boot device: /virtual-devices/disk@0  File and args:
SunOS Release 5.10 Version Generic_118833-36 64-bit
Copyright 1983-2006 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.
Hostname: ldom1
checking ufs filesystems
/dev/rdisk/c0d0s7: is logging.

ldom1 console login: root
Password:xxxxxx
Apr 25 10:38:17 ldom1 login: ROOT LOGIN /dev/console
Last login: Tue Apr 24 20:27:36 on console
Sun Microsystems Inc.  SunOS 5.10  Generic January 2005
You have new mail.
-bash-3.00# psrinfo
0          on-line    since 04/25/2007 10:36:44
1          on-line    since 04/25/2007 10:36:45
2          on-line    since 04/25/2007 10:36:45
3          on-line    since 04/25/2007 10:36:45
-bash-3.00# ifconfig -a
lo0: flags=2001000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4,VIRTUAL> mtu 8232
    index 1 inet 127.0.0.1 netmask ff000000
vnet0: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
    inet 10.6.160.70 netmask fffffc00 broadcast 10.6.163.255
    ether 0:14:4f:f8:f:12
-bash-3.00# hostid
83f80f12
-bash-3.00#
```

Solaris boots quickly, you will see the normal output

Look around, psrinfo reports the appropriate number of CPU's
The IP address is as we expected on a vnet0 device
the hostid is derived from the MAC address of the 'systems' first network port

Some exercises..

- Look at the guest domain's configuration

```

-bash-3.00# ldm list -l myldom1
Name:    myldom1
State:   active
Flags:   transition
OS:
Util:    0.0%
Uptime:  4d 17h 13m
Vcpu:    4
      vid    pid    util  strand
      0      4     0.3%  100%
      1      5     0.0%  100%
      2      6     1.2%  100%
      3      7     0.0%  100%
Memory:  1G
      real-addr      phys-addr      size
      0x8000000      0x48000000      1G
Vars:    nvramrc=devalias vnet0 /virtual-devices@100/channel-devices@200/network@0
devalias vdisk0 /virtual-devices@100/channel-devices@200/disk@0
boot-device=vdisk
use-nvramrc?=true
Vnet:    vnet1
mac-addr=0:14:4f:f8:f:12
service: primary-vsw0 @ primary
Vdisk:   vdisk1 voll@primary-vds0
service: primary-vds0 @ primary
Vcons:   ldom1@primary-vcc0 [port:5000]

```

Some exercises..

- Look at the control domain's configuration

```
-bash-3.00# ldm list -l primary
Name:    primary
State:   active
Flags:   normal,control,vio service
OS:      Solaris running
Util:    0.5%
Uptime:  16d 17h 39m
Vcpu:    4
      vid    pid    util  strand
      0      0      0.8%  100%
      1      1      0.2%  100%
      2      2      0.4%  100%
      3      3      0.4%  100%
Memory:  1G
      real-addr      phys-addr      size
      0x8000000      0x8000000      1G
Vars:    reboot-command=boot
IO:      pci@780 (bus_a)
         pci@7c0 (bus_b)
Vldc:    primary-vldc0 [num_clients=5]
Vldc:    primary-vldc3 [num_clients=7]
Vds:     primary-vds0  [num_clients=2]
         vdsdev: vol1  device=/ldoms/ldom1/disk_ldom1.img
         vdsdev: vol2  device=/ldoms/ldom2/diskimage
Vcc:     primary-vcc0  [num_clients=2]
         port-range=5000-5100
Vsw:     primary-vsw0  [num_clients=2]
         mac-addr=0:14:4f:fa:85:bb
         net-dev=e1000g0
         mode=prog,promisc
Vcons:   SP
```

Some exercises..

- See what is bound to the guest domain

```
-bash-3.00# ldm list-bindings myldom1
```

```
Name:    myldom1
State:   active
Flags:   transition
OS:
Util:    0.0%
Uptime:  4d 17h 4m
Vcpu:    4
```

vid	pid	util	strand
0	4	0.3%	100%
1	5	0.2%	100%
2	6	0.1%	100%
3	7	0.0%	100%

```
Memory: 1G
```

real-addr	phys-addr	size
0x8000000	0x48000000	1G

```
Vars:  nvramrc=devalias vnet0 /virtual-devices@100/channel-devices@200/network@0
devalias vdisk0 /virtual-devices@100/channel-devices@200/disk@0
boot-device=vdisk
use-nvramrc?=true
```

```
Vldcc: vldcc0 [Domain Services]
service: primary-vldc0 @ primary
[LDC: 0x0]
```

```
Vnet:  vnet1 [LDC: 0x4]
[Peer LDom: ldom2, mac-addr: 0:14:4f:f9:b6:f3]
mac-addr=0:14:4f:f8:f:12
service: primary-vsw0 @ primary
[LDC: 0x1]
```

```
Vdisk: vdisk1 voll@primary-vds0
service: primary-vds0 @ primary
[LDC: 0x2]
```

```
Vcons: [via LDC:3]
ldom1@primary-vcc0 [port:5000]
```

Some exercises..

- See what is bound to the control domain (slide 1 of 2)

```
-bash-3.00# ldm list-bindings primary
Name:    primary
State:   active
Flags:   normal,control,vio service
OS:      Solaris running
Util:    0.2%
Uptime:  16d 17h 44m
Vcpu:    4
      vid  pid  util strand
      0    0    0.5%  100%
      1    1    0.2%  100%
      2    2    0.2%  100%
      3    3    0.2%  100%
Memory:  1G
      real-addr      phys-addr      size
      0x8000000      0x8000000      1G
Vars:    reboot-command=boot
IO:      pci@780 (bus_a)
         pci@7c0 (bus_b)
Vldc:    primary-vldc0
         [LDC: 0x0]
         [(HV Control channel)]
         [LDC: 0x1]
         [LDom primary   (Domain Services channel)]
         [LDC: 0x3]
         [LDom primary   (FMA Services channel)]
         [LDC: 0xb]
         [LDom ldom1     (Domain Services channel)]
         [LDC: 0x10]
         [LDom ldom2     (Domain Services channel)]
... (more) ...
```

Some exercises..

- See what is bound to the control domain (slide 2 of 2)

```

... (more) ...
Vds:    primary-vds0
        vdsdev: vol1    device=/ldoms/ldom1/disk_ldom1.img
        [LDom ldom1, dev-name: vol1]
        [LDC: 0xe]
Vcc:    primary-vcc0
        [LDC: 0xf]
        [LDom ldom1, group: ldom1, port: 5000]
        port-range=5000-5100
Vsw:    primary-vsw0
        mac-addr=0:14:4f:fa:85:bb
        net-dev=e1000g0
        [LDC: 0xc]
        [LDom ldom1, name: vnet1, mac-addr: 0:14:4f:f8:f:12]
Vldcc:  vldcc1 [FMA Services]
        service: ldmfma
        service: primary-vldc0 @ primary
        [LDC: 0x4]
Vldcc:  vldcc2 [SP channel]
        service: spfma
        [LDC: 0x5]
Vldcc:  vldcc0 [Domain Services]
        service: primary-vldc0 @ primary
        [LDC: 0x2]
Vldcc:  hvctl1 [Hypervisor Control]
        service: primary-vldc0 @ primary
        [LDC: 0x0]
Vcons:  SP

```

Some exercises..

- List constraints (control and guest domain)

```
-bash-3.00# ldm list-constraints primary
```

```
Name: primary
```

```
Vcpu: 4
```

```
Memory: 1G
```

```
Vars: reboot-command=boot
```

```
IO: pci@780
```

```
pci@7c0
```

```
Vldc: primary-vldc0
```

```
Vldc: primary-vldc3
```

```
Vds: primary-vds0
```

```
        vdsdev: vol1      device=/ldoms/ldom1/disk_ldom1.img
```

```
        vdsdev: vol2      device=/ldoms/ldom2/diskimage
```

```
Vcc: primary-vcc0
```

```
        port-range=5000-5100
```

```
Vsw: primary-vsw0
```

```
        mac-addr=0:14:4f:fa:85:bb
```

```
        net-dev=e1000g0
```

```
-bash-3.00# ldm list-constraints myldom1
```

```
Name: myldom1
```

```
Vcpu: 4
```

```
Memory: 1G
```

```
Vars: nvramrc=devalias vnet0 /virtual-devices@100/channel-devices@200/network@0
```

```
devalias vdisk0 /virtual-devices@100/channel-devices@200/disk@0
```

```
        boot-device=vdisk
```

```
        use-nvramrc?=true
```

```
Vnet: vnet1
```

```
        mac-addr=0:14:4f:f8:f:12
```

```
        service: primary-vsw0
```

```
Vdisk: vdisk1 vol1@primary-vds0
```

```
        service: primary-vds0
```

Some exercises..

- List free devices; list services

```
-bash-3.00# ldm list-devices cpu
```

```
vCPU:
```

```

vCPUID  %FREE
  20      100%
  21      100%
... (snip) ...
  31      100%
```

```
-bash-3.00# ldm list-devices mem
```

```
Memory: Available mblocks:
```

```

PADDR          SIZE
0x188000000    26496M (0x678000000)
```

```
-bash-3.00# ldm list-services
```

```
Vldc: primary-vldc0
```

```
Vldc: primary-vldc3
```

```
Vds: primary-vds0
```

```

vdsdev: vol1      device=/ldoms/ldom1/disk_ldom1.img
```

```
Vcc: primary-vcc0
```

```

port-range=5000-5100
```

```
Vsw: primary-vsw0
```

```

mac-addr=0:14:4f:fa:85:bb
```

```

net-dev=e1000g0
```

```

mode=prog,promisc
```

```
-bash-3.00# ldm list-services primary
```

```
Vldc: primary-vldc0
```

```
Vldc: primary-vldc3
```

```
Vds: primary-vds0
```

```

vdsdev: vol1      device=/ldoms/ldom1/disk_ldom1.img
```

```
Vcc: primary-vcc0
```

```

port-range=5000-5100
```

```
Vsw: primary-vsw0
```

```

mac-addr=0:14:4f:fa:85:bb
```

```

net-dev=e1000g0
```

```

mode=prog,promisc
```

Some exercises..

- Behavior of the vsw and vnets...

```
-bash-3.00# ifconfig -a
lo0: flags=2001000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4,VIRTUAL> mtu 8232
      index 1 inet 127.0.0.1 netmask ff000000
vnet0: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
      inet 10.6.160.70 netmask fffffffc00 broadcast 10.6.163.255
      ether 0:14:4f:f8:f:12
-bash-3.00# hostid
83f80f12
-bash-3.00#
```

```
-bash-3.00# ifconfig -a
lo0: flags=2001000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4,VIRTUAL> mtu 8232
      index 1 inet 127.0.0.1 netmask ff000000
e1000g0: flags=201000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4,CoS> mtu 1500 index 3
      inet 10.6.160.125 netmask fffffffc00 broadcast 10.6.163.255
      ether 0:14:4f:fa:85:bb
-bash-3.00# hostid
83f80f12
-bash-3.00# ping 10.6.160.70
no answer from 10.6.160.70
-bash-3.00#
```

By default the control domain cannot communicate with the guest domains..., even if both domains on same subnet and can ping an external gateway... You must unplumb the e1000g0 device and plumb the vsw0 device.

Some exercises..

- Plumb vsw0 device to allow communications with your guest domains...

```

-bash-3.00# ifconfig -a
lo0: flags=2001000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4,VIRTUAL> mtu 8232
      index 1 inet 127.0.0.1 netmask ff000000
e1000g0: flags=201000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4,CoS> mtu 1500 index 3
      inet 10.6.160.125 netmask fffffc00 broadcast 10.6.163.255
      ether 0:14:4f:2b:65:4e
-bash-3.00# hostid
842b654e
-bash-3.00# ifconfig e1000g0 down           (See important note below!)
-bash-3.00# ifconfig e1000g0 unplumb
-bash-3.00# ifconfig vsw0 plumb
-bash-3.00# ifconfig vsw0 10.6.160.125 netmask 255.255.252.0 up
-bash-3.00# ifconfig -a
lo0: flags=2001000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4,VIRTUAL> mtu 8232 index 1
      inet 127.0.0.1 netmask ff000000
vsw0: flags=201000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4,CoS> mtu 1500 index 5
      inet 10.6.160.125 netmask fffffc00 broadcast 10.6.163.255
      ether 0:14:4f:fa:85:bb
-bash-3.00# ping 10.6.160.70
10.6.160.70 is alive
-bash-3.00# hostid
842b654e
-bash-3.00#

```

Important: don't be logged on via the interface you're about to unplumb!

Note that the HostID is based of the e1000g0 port MAC address
Even after changing the ethernet device, the HostID persists.

Some exercises..

- Setting up to jumpstart a domain...

```
{0} ok show-nets
a) /virtual-devices@100/channel-devices@200/network@0
q) NO SELECTION
Enter Selection, q to quit: a
/virtual-devices@100/channel-devices@200/network@0 has been selected.
Type ^Y ( Control-Y ) to insert it in the command line.
e.g. ok nvalias mydev ^Y
      for creating devalias mydev for /virtual-devices@100/channel-devices@200/network@0
{0} ok nvalias vnet0 ^Y/virtual-devices@100/channel-devices@200/network@0
{0} ok devalias
vnet0          /virtual-devices@100/channel-devices@200/network@0
vdisk0         /virtual-devices@100/channel-devices@200/disk@0
virtual-console /virtual-devices/console@1
net            /pci@780/network@1,1
disk           /pci@7c0/scsi@1/disk@0
scsi           /pci@7c0/scsi@1
ttyb           /virtual-devices/console@4
nvram          /virtual-devices/nvram@3
ttya           /virtual-devices/console@1
vdisk          /virtual-devices/disk@0
name           aliases
{0} ok boot vnet0 - install
```

Ensure that the image you intend to jumpstart is suitably patched and of the right versions to be LDoms aware..

No default network device alias exists so create one ...

You may want to set boot-device and auto-boot NVRAM variables at this stage too...

Some exercises..

- Using ZFS to host a Solaris image

```

bash# ls -l /ldoms/ldom1/*
-rw-r--r--  1 root      root          10737418240 May  6 04:32 /ldoms/ldom1/disk_ldom1.img

bash# zfs snapshot mydisk/ldom1@initial
bash# zfs clone mydisk/ldom1@initial mydisk/myldom1
bash# zfs set mountpoint=/ldoms/myldom1 mydisk/myldom1
bash# zfs list
NAME                                USED  AVAIL  REFER  MOUNTPOINT
mydisk                              10.0G  124G   24.5K  /mydisk
mydisk@initial                       23.5K    -    24.5K  -
mydisk/ldom1                         10.0G  124G   10.0G  /ldoms/ldom1
mydisk/ldom1@initial                   0      -   10.0G  -
mydisk/myldom1                         0     124G   10.0G  /mydisk/myldom1
bash# ls -l /ldoms/myldom1/*
-rw-r--r--  1 root      root          10737418240 May  8 06:31 /ldoms/myldom1/disk_ldom1.img

```

ZFS makes it possible to snapshot a disk image and quickly clone it for domains – saving disk space and time. Here, we make a clone in /ldoms/myldom1 based on a snapshot of a Solaris image in /ldoms/ldom1 **Note:** Make sure the control domain has 4GB of memory.

LDom alignment with CMT

- Traditional partitioning and VM technologies based on assumption “CPUs are scarce, so overcommit and time-slice them”. Hence:
 - > Overhead for time-slicing different contexts
 - > Intercept for “privileged operations”
 - > Latency servicing every interrupt
- VM systems also often overcommit RAM, causing overhead, requiring complex memory management
- CMT systems are “thread-rich” - so can dedicate CPU threads to each domain for lower overhead
 - > Reduces latency, overhead, and complexity
 - > Don't adjust weights or shares, just reassign threads where needed

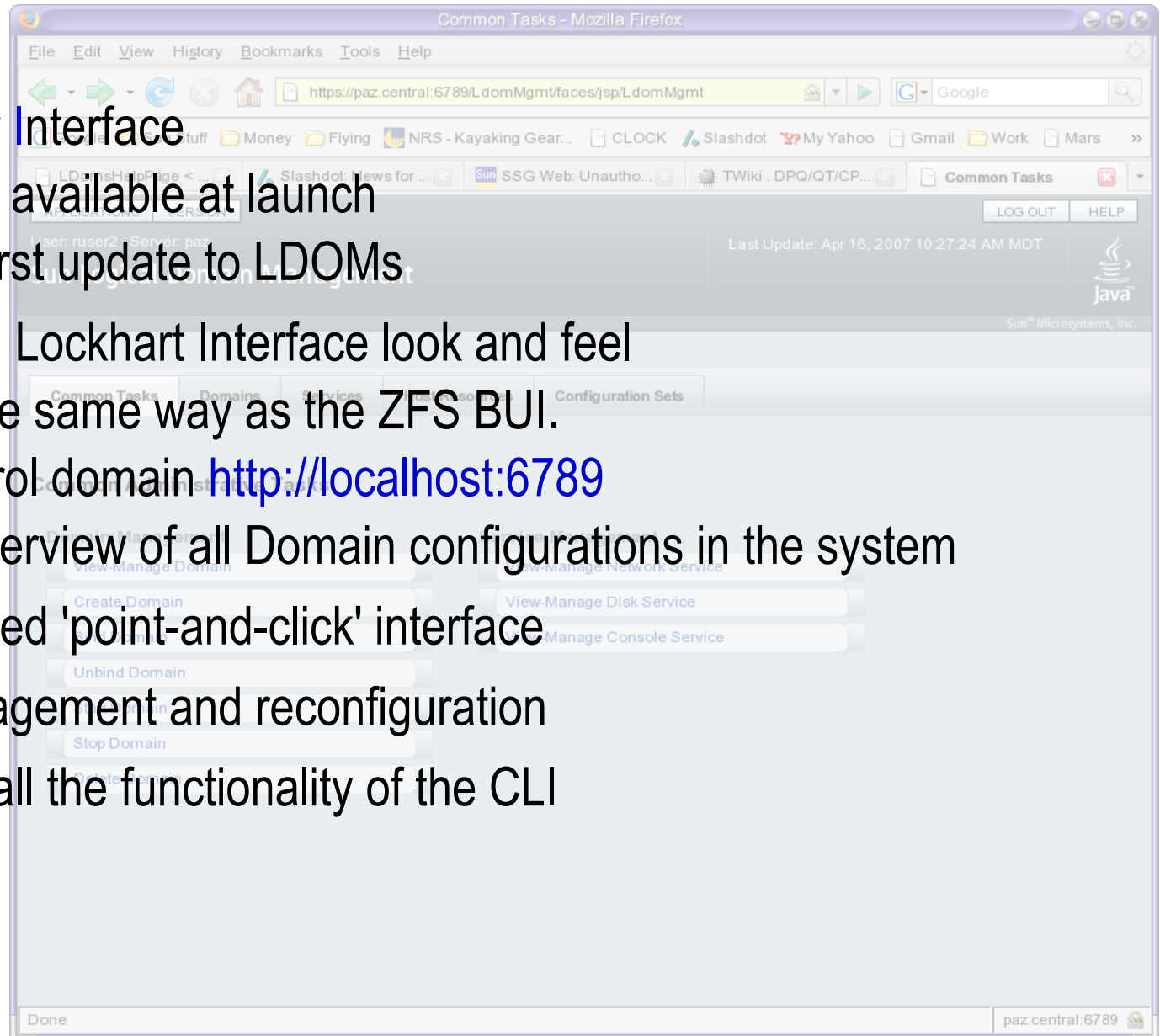
Logical Domains Features

- LDom 1.0 (4/26/2007)
 - > Niagara support
 - > Up to 32 LDom per system, guest domain may be rebooted independently
 - > Virtualized console, ethernet, disk & cryptographic acceleration
 - > Live re-configuration of virtual CPUs
 - > FMA diagnosis for each domain
 - > Control domain hardening
- LDom 1.0.1 (Q3CY07)
 - > Niagara2 support
 - > I/O domain reboot support
 - > Control domain minimization
 - > SNMP MIB
 - > Web management tool (freeware/unsupported)

* Requiring new Solaris 10 update

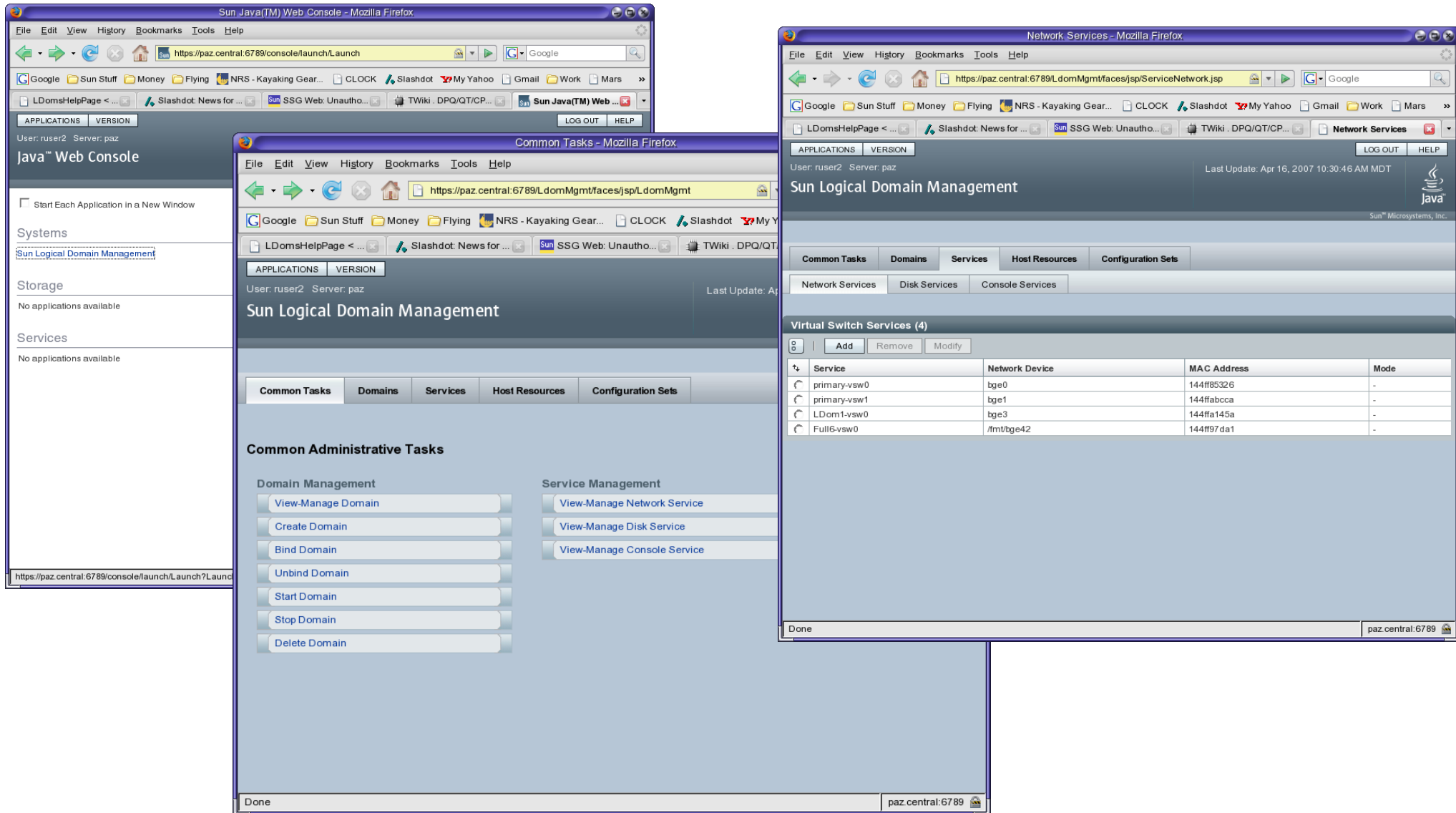
The easy option ... the BUI

- The **B**rowser **U**ser **I**nterface
 - > This will not be available at launch
- Coming with the first update to LDOMs
- Uses the standard Lockhart Interface look and feel
 - > Launched in the same way as the ZFS BUI.
 - > From the Control domain <http://localhost:6789>
 - > Provides an overview of all Domain configurations in the system
- Provides a simplified 'point-and-click' interface
- Allows basic management and reconfiguration
- Does not provide all the functionality of the CLI



The easy option ... the BUI

- The Browser User Interface



The image displays three overlapping browser windows showing the Sun Java(TM) Web Console interface. The top-left window shows the 'Sun Java(TM) Web Console' with a sidebar for 'Systems', 'Storage', and 'Services'. The middle window shows 'Common Tasks - Mozilla Firefox' with a 'Sun Logical Domain Management' page featuring 'Common Administrative Tasks' for Domain and Service management. The right window shows 'Network Services - Mozilla Firefox' with a 'Sun Logical Domain Management' page displaying 'Virtual Switch Services (4)' in a table.

Service	Network Device	MAC Address	Mode
primary-vsw0	bge0	144f85326	-
primary-vsw1	bge1	144fabcca	-
LDom1-vsw0	bge3	144ffa145a	-
Full6-vsw0	/fmbt/bge42	144f97da1	-

LDoms Best Practices

- Plan your configuration carefully. Reconfiguration can be awkward
- Use easy to understand names
 - > Don't overload vds, vsw, ldom, vdisk, vnic, etc...
- Where possible consider separating control, service and guest domains
- For high speed inter-domain communications use deviceless/in-memory VSW LDC channels
- For high performance, allocate a whole real device via a dedicated, properly sized service domain
- Use ZFS and other Solaris technologies to efficiently use of the server
- Look at the server physical device architecture to ensure you get the bandwidth you expect
- Secure and minimize the installations on the control and service domains as highly as possible

References for further information

- <http://www.sun.com/ldoms>
- <http://www.opensolaris.org/os/community/ldoms>
- Interesting Blog Entries:
 - > Linux support - http://blogs.sun.com/barton808/entry/linux_2_6_23_is
 - >
- SDLC Release of LDOMs
 - > <http://www.sun.com/download/products.xml?id=462e6bd6>
- Official Documentation for the SDLC release
 - http://www.sun.com/products-n-solutions/hardware/docs/Software/enterprise_computing/systems_management/ldoms/index.html

Increasing Customer Choice

Virtualization Technologies

- **Logical Domains**
 - > Multiple Solaris instances on same T[12] SPARC server
- **Solaris Containers**
 - > Zones & Resource Management
 - > Low-overhead OS virtualization
- **Xen**
 - > Open Source – Sun a full contributor, Solaris a full player
 - > Porting Solaris on x86 – has already booted
- **VMware**
 - > Reselling / partner agreement / excellent on x4_00
 - > Solaris on x64/86 as client OS

High Level Functional Comparison

Solaris Containers

- Available everywhere Solaris runs: USII and up, x86/64
- Far smaller CPU, disk, memory footprint than full OS images
- Extremely efficient – suitable for “large N” deployments
- SRM or pool managed sub-CPU resource management
- Some restrictions (like NFS in a zone, ISV issues)
- Single kernel, with implications for single-point-of-failure, patching

Logical Domains

- Available on CMT SPARC only
 - > As many domains as there are CPU threads
 - > For workloads applicable to T1
- Flexible resource allocation
- Each domain runs an independent OS: can be different levels and independently patched
- Efficient implementation, with less overhead than other virtual machines, but still an entire OS instance

Logical Domains (LDoms) Concepts and Examples

Rich Reynolds

odd1@sun.com