OCTAVE ORGERON

an introduction to logical domains

PART 3: ADVANCED NETWORKING AND STORAGE

Octave Orgeron is a Solaris Systems Engineer and an OpenSolaris Community Leader. Currently working in the financial services industry, he also has experience in the e-commerce, Web hosting, marketing services, and IT technology markets. He specializes in virtualization, provisioning, grid computing, and high availability.

unixconsole@yahoo.com

IN THE OCTOBER 2007 ISSUE OF *;login:*, I walked you through the installation and configuration of the Logical Domain (LDom) technology. In this article, I'll talk about advanced topics concerning configuration networking and storage. The discussion here will give you a deeper understanding of the two most challenging resources to administer with LDoms.

Is It a Guest Domain?

When one is remotely logged in, it is difficult to determine whether the environment is in a guest domain. The Solaris operating system will behave and function like a normal standalone server. The most obvious place to see the difference is in the device tree:

```
ldom1:~ $ cat /etc/path_to_inst
#
# Caution! This file contains critical kernel state
#
"/iscsi" 0 "iscsi"
"/pseudo" 0 "pseudo"
"/scsi_vhci" 0 "scsi_vhci"
"/options" 0 "options"
"/virtual-devices@100" 0 "vnex"
"/virtual-devices@100/channel-devices@200" 0 "cnex"
"/virtual-devices@100/channel-devices@200/network@0" 0 "vnet"
"/virtual-devices@100/channel-devices@200/network@1" 1 "vnet"
"/virtual-devices@100/channel-devices@200/disk@0" 0 "vdc"
"/virtual-devices@100/channel-devices@200/disk@1" 1 "vdc"
"/virtual-devices@100/console@1" 0 "qcn"
"/virtual-devices@100/ncp@6" 0 "ncp"
"/virtual-devices@100/random-number-generator@e" 0 "n2rng"
"/virtual-devices@100/n2cp@7" 0 "n2cp"
```

Notice how short the device tree is and that many of the devices are under the "/virtual-devices@100" nexus. All of the networking and storage devices are virtualized, a clear indicator that the system you are logged into is a guest domain. Beyond these hardware subtleties, it is difficult for a user or developer to tell the difference. However, from an administrative point of view, networking and storage are critical aspects of any environment.

Advanced Networking

Networking with LDoms can be complicated, depending on the goals you wish to accomplish. As such, it is important to have a clear understanding of what is possible where networking is concerned.

Virtual switches, or VSWs, are virtual devices that provide the functions of a basic layer 2 network switch and packet demultiplexer. The VSW enables network communications between guest domains on the same VSW and the external network to which the VSW is connected. For internal traffic, the VSW classifies incoming packets based on the target VNET MAC address and switches the packets to the destination VNET device. For external traffic, it acts like a forwarding agent between VNET interfaces and external clients.

The MAC address of a VSW or a VNET can be automatically assigned or specified during creation. Normally, the automatic assignment of MAC addresses should not cause conflicts with other MAC addresses on your networks. However, if a conflict does arise, you can specify the MAC address. This can also be handy if you are going to use DHCP for assigning IPs or have a specific application requirement. For example:

```
primary:~ # ldm add-vsw mac-addr=8:20:4f:ab:cd:ef net-dev=e1000g6 primary-vsw6 primary
primary:~ # ldm add-vnet mac-addr=00:14:4f:f9:c6:96 ldom1-vnet2 primary-vsw6 ldom1
```

This may also be useful if you want the MAC address for a VSW to be the same as the physical network port to which it is connected:

```
primary~ # ifconfig e1000g0
e1000g0: flags=201000802<UP,BROADCAST,MULTICAST,IPv4,CoS> mtu 1500 index 2
      inet 192.168.2.12 netmask ffffff00 broadcast 192.168.2.255
      ether 0:14:4f:96:f6:72

primary:~ # ldm add-vsw mac-addr=0:14:4f:96:f6:72 net-dev=e1000g0 primary-vsw0 primary
```

By default, when a VSW is created, the primary domain is incapable of communicating directly with the guest domains on that VSW. For the primary domain to be able to communicate with the guest domains directly, it must be connected to the VSW. This can be done by either plumbing the VSW in addition to the physical network device or only plumbing the VSW. The second option consumes fewer IPs and can prevent confusion:

```
primary:~ # ifconfig e1000g0 down unplumb
primary:~ # mv /etc/hostname.e1000g0 /etc/hostname.vsw0
primary:~ # svcadm restart network/physical
primary:~ # ifconfig vsw0
vsw0: flags=201000802<UP,BROADCAST,MULTICAST,IPv4,CoS> mtu 1500 index 2
      inet 192.168.2.12 netmask ffffff00 broadcast 192.168.2.255
      ether 0:14:4f:96:f6:72
```

This plumbing enables the primary domain to communicate directly with the guest domains that are connected to the VSW without having to transverse the physical network. However, this can expose your primary domain to insecure networks. The primary domain should be protected from public access because of its importance. If the primary domain does not need to be connected to the same network your guest domains are on, the underlying physical network device or VSW can be left unplumbed.

VSWs can also be configured for private networking. Such private networks can be useful for configuring communications only among LDoms. This is accomplished by not specifying a physical network device to bind with the VSW:

```
primary:~ # ldm add-vsw primary-vsw4 primary
primary:~ # ldm list-bindings primary
...
NAME            MAC               NET-DEV   DEVICE      MODE
primary-vsw3    00:14:4f:96:f6:75 e1000g3   switch@3    prog,promisc
```

```
    primary-vsw4     00:14:4f:f8:d4:a6              switch@4 routed
...
```

Notice the differences between these VSWs. When a physical network device is specified, the VSW is enabled with layer 2 switching with the underlying hardware in programmed and promiscuous mode. If a physical network device is not specified, the VSW is enabled with layer 3 IP routing in nonpromiscuous mode.

If a physical network port were to go offline, this could potentially cause a major outage for your guest domains. One of the ways you can reduce the risks of such an outage is to configure IPMP (IP Multi-Pathing), thereby enabling multiple network paths to handle the fail-over of IPs in the event of a NIC or connection failure. IPMP also load-balances outgoing traffic. The easiest option is to configure it within the guest domain. This is accomplished by configuring at least two VNETs into a guest domain that are connected to separate VSWs on the same physical network, then configuring IPMP with IP-probe-based error detection in the guest domain:

```
primary:~ # ldm add-vnet ldom1-vnet0 primary-vsw0 ldom1
primary:~ # ldm add-vnet ldom1-vnet1 primary-vsw2 ldom1

ldom1:~ # cat /etc/hostname.vnet0
192.168.2.101/24 broadcast + group net1 -failover deprecated up
addif ldom1/24 broadcast + up

ldom1:~ # cat /etc/hostname.vnet1
192.168.2.102/24 broadcast + group net1 -failover deprecated up

ldom1:~ # ifconfig -a

...
vnet0: flags=209040843<UP,BROADCAST,RUNNING,MULTICAST,DEPRECATED,IPv4,
NOFAILOVER,CoS> mtu 1500 index 2
        inet 192.168.2.101 netmask ffffff00 broadcast 192.168.2.255
        groupname net1
        ether 0:14:4f:fb:49:89
vnet0:1: flags=201000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4,CoS> mtu 1500 index 2
        inet 192.168.2.100 netmask ffffff00 broadcast 192.168.2.255
vnet1:
flags=209040843<UP,BROADCAST,RUNNING,MULTICAST,DEPRECATED,IPv4,NOFAILOVER,
CoS> mtu 1500 index 3
        inet 192.168.2.102 netmask ffffff00 broadcast 192.168.2.255
        groupname net1
        ether 0:14:4f:fb:f3:12
```

IP-based probing configures the interfaces to ping each other to determine a failure. This determination is required with guest domains because the link status information from the physical network devices is not propagated up through the VSWs. Although such a procedure will provide IP fail-over for your guest domain, it does consume additional IPs for each guest domain you configure with IPMP. Other methods are discussed in the LDom administrator guide [1].

Networking features such as bridging, VLAN tagging, and link aggregation are not yet supported with LDoms. However, they are being worked on in the OpenSolaris community [2].

Advanced Storage

As was demonstrated in the previous article, storage can be virtualized in interesting ways to support LDoms. The difficult part is figuring out which storage option to use. The chart in Table 1 can be used as a quick reference.

Storage Option	Jumpstart	Virtualized for Guest Domains	Non-Virtualized for Guest Domains
DASD	Yes	Yes	No
SAN	Yes	Yes	No
iSCSI	Yes	Yes	Yes
ZFS volumes	Yes*	Yes	Yes
SVM meta-devices	Yes*	Yes	Yes
Virtual disk images	Yes	Yes	No
NAS	No	No	Yes

* Currently, only works with Solaris Express Build 75 and higher.

TABLE 1: STORAGE OPTIONS

DASD, SAN, iSCSI, virtual disk images, ZFS volumes, and SVM meta-devices can be virtualized as either boot or data storage for guest domains. Let's start out and take a look at DASD:

```
primary:~ # ldm add-vdsdev /dev/dsk/c3t2d0s2 ldom4-vdsk0@primary-vds0
primary:~ # ldm add-vdsdev /dev/dsk/c3t3d0s2 ldom4-vdsk1@primary-vds0
primary:~ # ldm add-vdisk ldom4-vdsk0 ldom4-vdsk0@primary-vds0 ldom4
primary:~ # ldm add-vdisk ldom4-vdsk1 ldom4-vdsk1@primary-vds0 ldom4

ldom4:~ # format
...
       0. c0d0 <SUN72G cyl 14087 alt 2 hd 24 sec 424>
          /virtual-devices@100/channel-devices@200/disk@0
       1. c0d1 <SUN72G cyl 14087 alt 2 hd 24 sec 424>
          /virtual-devices@100/channel-devices@200/disk@1
...
```

DASD storage is simple and consumes little overhead. However, you are limited by the amount that can be installed internally or attached externally to your server. In the event of a system failure, that storage would have to be manually migrated to a standby server.

SAN storage has many benefits, including performance, reliability, and portability among servers. Solaris 10 and above include support for SAN connectivity. Although third-party drivers and FC HBAs may work on standalone servers, they may not work with LDoms. This is also true with SAN multi-pathing software. STMS (a.k.a. MPXIO) should be utilized. STMS can only be utilized in a service domain that has direct access to the FC HBAs, such as the primary domain. The SAN infrastructure and multi-pathing are completely transparent to guest domains. This situation will change in the future when FC HBAs can be virtualized using NPIV, which provides guest domains with a virtualized FC HBA instance [3]. Here is an example of virtualizing SAN storage:

```
primary:~ # ldm add-vdsdev /dev/dsk/c6t60060160B5681200B2CE5AD37981DB11d0s2 ldom5-vdsk0@primary-vds0
primary:~ # ldm add-vdsdev /dev/dsk/sc6t60060160B5681200B3CE5AD37981DB11d02 ldom5-vdsk1@primary-vds0
primary:~ # ldm add-vdisk ldom5-vdsk0 ldom5-vdsk0@primary-vds0 ldom5
primary:~ # ldm add-vdisk ldom5-vdsk1 ldom5-vdsk1@primary-vds0 ldom5

ldom5:~ # format
...
       0. c0d0 <DGC-RAID5-0219 cyl 32766 alt 2 hd 64 sec 10>
          /virtual-devices@100/channel-devices@200/disk@0
       1. c0d1 <DGC-RAID5-0219 cyl 32766 alt 2 hd 64 sec 10>
          /virtual-devices@100/channel-devices@200/disk@1
...
```

iSCSI has some interesting advantages over SAN storage owing to its low entry costs and the ubiquity of Ethernet. Combined with dedicated networking or 10Gb Ethernet, it can be a viable solution for virtualization. iSCSI targets can be virtualized for guest domain storage and they can be used directly by guest domains. However, OpenBoot does not support iSCSI, so neither standalone servers nor guest domains can boot directly from it at this point. Hopefully, this will change in the future.

For iSCSI to be utilized as boot storage for a guest domain, it must first be virtualized:

```
primary:~ # iscsiadm list target
Target: iqn.1986-03.com.sun:02:8a59994a-b4df-4968-d1e1-a63e4229bd2c
        Alias: storage/ldom3-vdsk0
        TPGT: 1
        ISID: 4000002a0000
        Connections: 1
Target: iqn.1986-03.com.sun:02:6eae2782-e453-ea54-c39b-d87bca8636de
        Alias: storage/ldom3-vdsk1
        TPGT: 1
        ISID: 4000002a0000
        Connections: 1

primary:~ # ldm add-vdsdev /dev/dsk/c2t01000003BA16E64B00002A004747D0F9d0s2 ldom3-vdsk0@primary-vds0
primary:~ # ldm add-vdsdev /dev/dsk/c2t01000003BA16E64B00002A004747D0FBd0s2 ldom3-vdsk1@primary-vds0

primary:~ # ldm add-vdisk ldom3-vdsk0 ldom3-vdsk0@primary-vds0 ldom3
primary:~ # ldm add-vdisk ldom3-vdsk1 ldom3-vdsk1@primary-vds0 ldom3

ldom3:~ # format
...
        0. c0d0 <SUN-SOLARIS-1 cyl 32766 alt 2 hd 4 sec 160>
           /virtual-devices@100/channel-devices@200/disk@0
        1. c0d1 <SUN-SOLARIS-1 cyl 32766 alt 2 hd 4 sec 160>
           /virtual-devices@100/channel-devices@200/disk@1
...
```

Guest domains can also make use of iSCSI for additional storage directly:

```
ldom1:~ # iscsiadm list target
Target: iqn.1986-03.com.sun:02:aa8c6f76-52b5-e3f3-9725-f8c3bbb18fbf
    Alias: storage/ldom1-vdsk2
    TPGT: 1
    ISID: 4000002a0000
    Connections: 1

ldom1:~ # format
...
    2. c1t01000003BA16E64B00002A004747B609d0 <SUN-SOLARIS-1 cyl 32766 alt 2 hd 4 sec 80>
       /scsi_vhci/ssd@g01000003ba16e64b00002a004747b609
...
```

ZFS and SVM can be used to create volumes or meta-devices that can be virtualized for LDoms, allowing a very granular level of control over how storage is utilized. These volumes will appear as local disks in a guest domain. However, in Solaris 10 these volumes are crippled in the sense that they are not virtualized as whole disks and are presented with only slice 0. This prevents them from being utilized for Jumpstart or for booting. Luckily, this has already been fixed in OpenSolaris as of build 75 and will be back-ported to Solaris 10 at some point in the future. Here is a demonstration of how ZFS volumes can be utilized with LDoms:

```
primary:~ # zfs create -V 10gb ldoms/ldom2-vdsk0
primary:~ # zfs create -V 10gb ldoms/ldom2-vdsk1

primary:~ # ldm add-vdsdev /dev/zvol/dsk/ldoms/ldom2-vdsk0 ldom2-vdsk0@primary-vds0
primary:~ # ldm add-vdsdev /dev/zvol/dsk/ldoms/ldom2-vdsk1 ldom2-vdsk1@primary-vds0

primary:~ # ldm add-vdisk ldom2-vdsk0 ldom2-vdsk0@primary-vds0 ldom2
primary:~ # ldm add-vdisk ldom2-vdsk1 ldom2-vdsk1@primary-vds0 ldom2

ldom2:~ # format
...
          0. c0d0 <SUN-DiskImage-10GB cyl 34950 alt 2 hd 1 sec 600>
             /virtual-devices@100/channel-devices@200/disk@0
          1. c0d1 <SUN-DiskImage-10GB cyl 34950 alt 2 hd 1 sec 600>
             /virtual-devices@100/channel-devices@200/disk@1
...
```

Virtual disk images provide a wide range of flexibility as they can be stored on DASD, SAN, iSCSI, and NAS. When combined with the use of advanced file systems, such as ZFS, LDoms can be quickly replicated and provisioned. Here is a demonstration of this type of flexibility:

```
primary :~ # ldm list-bindings primary | grep ldom1 | grep img
   primary-vds0    ldom1-vdsk0      /ldoms/local/ldom1/ldom1-vdsk0.img
                   ldom1-vdsk1      /ldoms/local/ldom1/ldom1-vdsk1.img

primary :~ # zfs list /ldoms/local/ldom1
NAME             USED   AVAIL   REFER   MOUNTPOINT
ldoms/ldom1      25.0G  72.0G   25.0G   /ldoms/local/ldom1

primary :~ # ldm list ldom1
NAME   STATE   FLAGS   CONS   VCPU   MEMORY        UTIL     UPTIME
ldom1  active  -n—-   5000   4      4G            0.2%     16h 11m
```

As we can see, ldom1 is using virtual disk images that are on a ZFS file system. We can utilize ldom1 as a template for future LDoms by logging into it to perform a sys-unconfig to remove the host configuration and bring ldom1 to a halt. Then we can take a ZFS snapshot and clone:

```
primary :~ # ldm list ldom1
NAME   STATE   FLAGS   CONS   VCPU   MEMORY        UTIL     UPTIME
ldom1  bound   ——--   5000   4      4G

primary :~ # zfs snapshot ldoms/ldom1@copy1
primary :~ # zfs clone ldoms/ldom1@copy1 ldoms/ldom6
primary :~ # zfs set mountpoint=/ldoms/local/ldom6 ldoms/ldom6

primary :~ # zfs list ldoms/ldom6
NAME             USED   AVAIL   REFER   MOUNTPOINT
ldoms/ldom6      16K    72.0G   25.0G   /ldoms/local/ldom6

primary:~ # ls /ldoms/local/ldom6
ldom1-vdsk0.img ldom1-vdsk1.img ldom1-vdsk2.img
```

The ZFS snapshot is a point in time copy that does not use any additional space. When we take a ZFS clone, only the data that is changed will consume space. Now let's rename the virtual disk images and add them to our new LDom:

```
primary :/ldoms/local/ldom6 # mv primary 1-vdsk0.img ldom6-vdsk0.img
primary :/ldoms/local/ldom6 # mv primary 1-vdsk1.img ldom6-vdsk1.img

primary :~ # ldm add-vdsdev /ldoms/local/ldom6/ldom6-vdsk0.img ldom6-vdsk0@primary-vds0
```

```
primary :~ # ldm add-vdsdev /ldoms/local/ldom6/ldom6-vdsk1.img ldom6-vdsk1@primary-vds0

primary :~ # ldm add-vdisk ldom6-vdsk0 ldom6-vdsk0@primary-vds0 ldom6
primary :~ # ldm add-vdisk ldom6-vdsk1 ldom6-vdsk1@primary-vds0 ldom6

primary :~ # ldm bind ldom6
primary :~ # ldm start ldom6
LDom ldom6 started

primary :~ # ldm list ldom6
NAME    STATE    FLAGS   CONS    VCPU    MEMORY   UTIL    UPTIME
ldom6   active   t---    5003    4       2G       25%     25s
```

Once we connect to the console for ldom6 and boot it, we can configure it for usage by following the standard host configuration screens. Once that is completed, we can start using ldom6:

```
ldom6:~ # uname -a
SunOS ldom6 5.11 snv_75 sun4v sparc SUNW,SPARC-Enterprise-T5120
ldom6:~ # format
...
       0. c0d0 <SUN-DiskImage-10GB cyl 34950 alt 2 hd 1 sec 600>
          /virtual-devices@100/channel-devices@200/disk@0
       1. c0d1 <SUN-DiskImage-10GB cyl 34950 alt 2 hd 1 sec 600>
          /virtual-devices@100/channel-devices@200/disk@1
...

primary:~ # zfs list ldoms/ldom6
NAME            USED   AVAIL   REFER   MOUNTPOINT
ldoms/ldom6     382M   71.7G   25.0G   /ldoms/local/ldom6

primary:~ # du -sh /ldoms/local/ldom6
  25G /ldoms/local/ldom6
```

Notice that our cloned LDom is actually only using about 382 MB of space in the ZFS storage pool. Using this method can save considerable amounts of storage for each LDom. Since virtual disk images are just sparse files, they can be easily migrated among different storage types and even burned to DVD disks for portability.

NAS storage cannot be used for Jumpstart, but it can be utilized for storing virtual disk images. This enables NAS storage to consolidate virtual disk images for LDoms and make them easily accessible for a farm of LDom-capable servers. Here's how:

```
primary:~ # df -h /ldoms/nas/ldom7
Filesystem              size    used   avail  capacity  Mounted on
192.168.2.2:/export/ldoms/ldom7
                        92G     11G    82G    12%       /ldoms/nas/ldom7

primary:~ # mkfile 10g /ldoms/nas/ldom7/ldom7-vdsk0.img
primary:~ # mkfile 10g /ldoms/nas/ldom7/ldom7-vdsk1.img

primary:~ # ldm add-vdsdev /ldoms/nas/ldom7/ldom7-vdsk0.img ldom7-vdsk0@primary-vds0
primary:~ # ldm add-vdsdev /ldoms/nas/ldom7/ldom7-vdsk1.img ldom7-vdsk1@primary-vds0
primary:~ # ldm add-vdisk ldom7-vdsk0 ldom7-vdsk0@primary-vds0 ldom7
primary:~ # ldm add-vdisk ldom7-vdsk1 ldom7-vdsk1@primary-vds0 ldom7

ldom7:~ # format
```

```
...
       0. c0d0 <SUN-DiskImage-10GB cyl 34950 alt 2 hd 1 sec 600>
          /virtual-devices@100/channel-devices@200/disk@0
       1. c0d1 <SUN-DiskImage-10GB cyl 34950 alt 2 hd 1 sec 600>
          /virtual-devices@100/channel-devices@200/disk@1
...
```

Within a guest domain, NAS storage can be accessed via standard NFS and can be utilized for sharing common storage such as home directories, application binaries, and application data.

One of the things you may have noticed is that all of the storage examples shown here are configured with at least two virtual disks for each guest domain. This is because the virtualized boot disks for the guest domains are mirrored. The mirroring is handled by SVM and in the future ZFS. This increases the reliability of your guest domains and enables features such as Live Upgrade, which is used for Solaris release upgrades.

```
ldom1:~ # metastat -p
d20 -m /dev/md/rdsk/d21 /dev/md/rdsk/d22 1
d21 1 1 /dev/rdsk/c0d0s1
d22 1 1 /dev/rdsk/c0d1s1
d10 -m /dev/md/rdsk/d11 /dev/md/rdsk/d12 1
d11 1 1 /dev/rdsk/c0d0s0
d12 1 1 /dev/rdsk/c0d1s0

ldom1:~ # df -h /
Filesystem             size   used  avail capacity  Mounted on
/dev/md/dsk/d10        7.9G   4.5G   3.3G    58%    /
ldom1:~ # swap -l

swapfile             dev  swaplo blocks     free
/dev/md/dsk/d20      85,5       16 2097584  2097584
```

The mirroring of the virtual boot disks in a guest domain is only the beginning. SVM and ZFS can be utilized to manage any additional storage that is virtualized into a guest domain or connected through other means such as iSCSI, enabling a wide range of configurations and possibilities for managing storage.

There are some key things to keep in mind with LDoms and storage:

- Virtualized storage cannot be shared among guest domains. This can prevent certain storage applications, such as clustered file systems, from functioning.
- When storage is virtualized, low-level SCSI system calls are not implemented. This can affect certain third-party volume managers or applications that expect such low-level access to SCSI targets.
- Using SVM or ZFS in the service domain for managing guest domain storage can increase the I/O overhead. As such, at least 4 GB of memory should be allocated to your primary domain.
- Backups should be performed from your guest domains. If you have a guest domain that is using virtual disk images, it should be shut down before being backed up. This will prevent the backups of the images files from being "fuzzy."

Summary

This article has demonstrated many of the advanced networking and storage configurations. This background should enable you to make informed decisions when configuring LDoms in conjunction with your networking and

storage infrastructures. In the next article, I will demonstrate other advanced topics that center around hardware design, resource management, and manageability.

REFERENCES

[1] LDoms Administration Guide 1.0: http://docs-pdf.sun.com/819-6428-11/819-6428-11.pdf.

[2] OpenSolaris LDoms Community: http://opensolaris.org/os/community/ldoms/.

[3] OpenSolaris NPIV Project: http://www.opensolaris.org/os/project/npiv/.