ORACLE
SOLARIS

An Oracle White Paper
February 2011

Exploring Network Virtualization With Oracle Solaris Zones on Oracle Solaris 11 Express

ORACLE

Disclaimer

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Executive Overview

Oracle Solaris 11 Express provides powerful new networking features for network resource management and network virtualization. This paper provides hands on guidance for exploring some of these features.

Introduction

Before we investigate using network resource management and network virtualization facilities we want to briefly address why these new features are important. In brief:

- Network virtualization allows partitioning a physical NIC for consolidation purposes, i.e sharing a physical NIC between multiple zones or virtual machines running on the same system.

- Network virtualization takes server virtualization to the next level - the ability to virtualize entire network topologies of servers, routers, switches, and firewalls all running on a single platform and requiring no additional investment in networking hardware. Network virtualization can be used for a variety of purposes, from prototyping, to developing and testing, to service deployment.

- Network resource management adds the capability to specify for network interfaces
    - Bandwidth limits
    - CPU assignments for handling network traffic. Resource management properties can be assigned to either physical or virtual network interfaces with no extra performance overhead.

Network Virtualization Concepts

In this paper we will cover using the facilities within Oracle Solaris 11 for both network virtualization and network resource management. For network virtualization we mean the ability to virtualize physical network components, in other words, to duplicate a physical network topology within a single system where the basic building blocks are virtual constructs, not physical devices. We start with the fundamental building blocks within Oracle Solaris 11 for building virtual networks:

- VNICs- Virtual Network Interface Controllers (VNICs), the fundamental building block of network virtualization. VNICs are created and assigned IP addresses as communication end points.
- Virtual Switching- the capability for VNICs to communicate with each other as if they were connected through a physical switch.
- Etherstubs- a special type of data link that can be used instead of a physical NIC to create VNICs and the virtual switches that connect them.
- Flows: Per link (virtual or physical) management to attain Quality of Service (QoS) goals by setting bandwidth limits for a subset of the traffic flowing through the datalink.

VNIC: Virtual Network Interface

VNICs are created on top of physical interfaces or on top of etherstubs, and from the application's point of view, VNICs appear exactly like physical interfaces. You configure VNICs as you configure any physical port, using the same commands with the same syntax. Tools such as ifconfig, ipadm, snoop, etc., all work on VNICs. The dladm(1M) command is used to control data link configurations and much of the functionality of the dladm command is focused on managing VNICs. Also, each VNIC has its own statistics for purposes of monitoring and those statistics can be accessed through the new dlstat(1M) command.

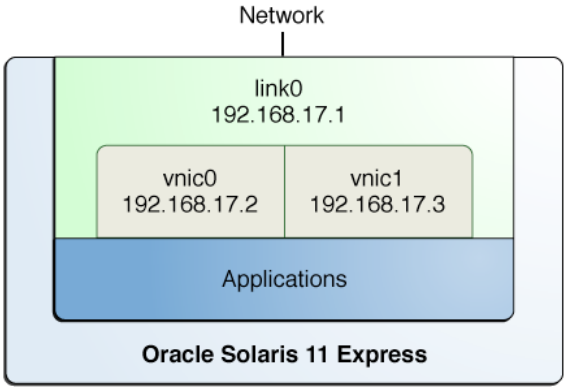

Figure 1 - Virtual Network Interfaces

Etherstub: Build a Virtual Network

Whenever you create two or more VNICs on the same physical port, a virtual switch will be created at the MAC layer. In the example below we highlight the virtual switch that is enabled by the configuration.

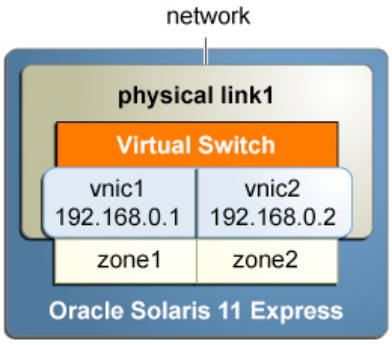

Figure 2 - Virtual network switch created automatically on VNICs sharing same physical port.

The effect of the creation of the virtual switch is that traffic between 192.168.0.1 and 192.168.0.2 is switched at the MAC layer. It does not need to leave via the physical NIC to be switched by some external piece of hardware. As long as the VNICs share the same physical NIC, and are on the same VLAN, this MAC layer virtual switch can be employed.

What about the more general case where you need to create complex networks within a system? You don't want to be constrained by the availability of physical ports on a system or that all VNICs are created on one physical link.

In the general case, a new software construct can be employed, the Etherstub. Etherstubs allow creating VNICs that communicate with each other through a virtual switch which is independent from a physical NIC. The general use for this is for consolidating physical network topology into a virtual network and also simulating a physical network for planning purpose, for experimentation, for debugging, or for developing distributing applications.

We will go into detail about implementing this particular scenario in the second part of this guide. For now, note the physical scenario and one possible virtual equivalent.

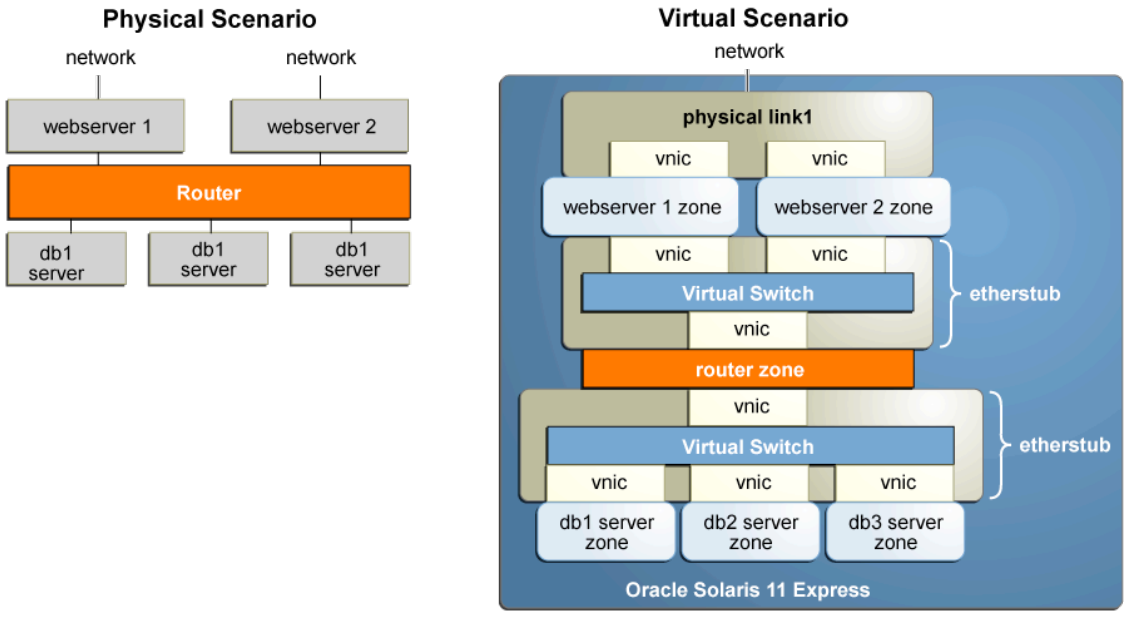

Figure 3 - Virtual switching enabled by use of Etherstubs

The solution is a combination of virtual switches enabled by creating Etherstubs, and a router zone, which can route packets out the correct virtual interface. From this simple example you could see how you could build very complex network topologies within a single server, with applications running in zones and communicating through networks that potentially could be restricted to within the system.

Example 2 will discuss in great detail constructing this particular virtualization example.

Flow: Managed Network Traffic Policy

A flow is created on top of an interface – physical or virtual – and describes network traffic in terms of application (port), protocol, source, and destination, and provides a handle that can be used to implement resource policies for bandwidth.

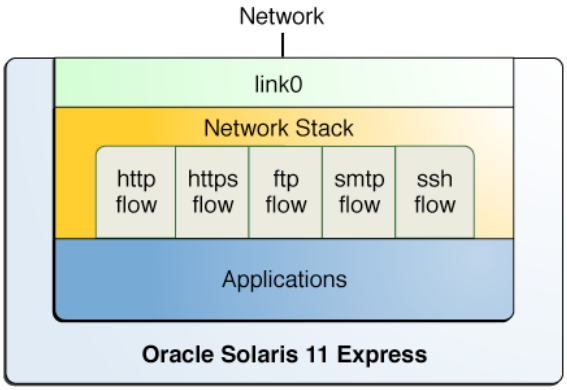

Figure 4 - Flows

The diagram shows an interface with several TCP flows: web, secure web, file transfer, email, and secure login. Any traffic that does not match any of the flows passes through the network stack without any policies imposed (other than policies applied to the interface itself). Traffic that does match one of the flows will be limited by the bandwidth assigned to that flow.

Example 1: VNICs and Flows on a Physical Interface

This example shows how to

- Create a VNIC
- Create a flow
- Apply bandwidth limits to VNICs and flows

The use of a flow is illustrated below. We are going to set up not only the bandwidth of general traffic through the virtual NIC, but also bandwidth of particular types of traffic through the same VNIC. To accomplish the latter, we use the "flow" construct from Oracle Solaris 11. In our example, we will identify the traffic type based on what TCP port is being used- specifically http traffic on port 80 and https traffic on port 443. Other attributes we could have used to identify flows are the source and destination IP addresses, or protocol (e.g. UDP, TCP). The network configuration looks like this:

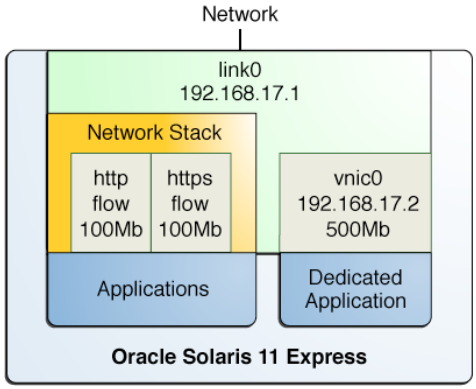

Figure 5 - Two flows and one VNIC have bandwidth limits.

Step 1: Disable nwamd

By default, Oracle Solaris 11 Express uses nwamd, the network automagic (configuration) daemon, to automatically configure the network. For experimenting with network virtualization and network resource management, we recommend disabling nwamd.

```
# svcadm disable network/physical:nwam
# svcadm enable network/physical:default
```

Then manually configure your network using the new ipadm command (or the ifconfig command). Depending on your network configuration you may have to edit /etc/defaultrouter,

/etc/nsswitch.conf, and /etc/resolv.conf. For example, if you were using DNS, /etc/nsswitch.conf would be based on /etc/nsswitch.dns.

Run these commands from the console rather than over the network- for example, over an ssh connection- to avoid losing connectivity to the system during reconfiguration.

Step 2: Create the VNIC

First determine the physical NICs available on the system- in this case **bge0**. Then create a virtual NIC, vnic0 on it. Bring up the interface with the address 192.168.17.2. The '/24' is a shorthand way to designate a netmask of 255.255.255.0. We illustrate enabling the network using the new ipadm command.

```
# dladm show-phys
LINK         MEDIA       STATE   SPEED  DUPLEX  DEVICE
bge0         Ethernet    up      1000   full    bge0
# dladm create-vnic -l bge0 vnic0
# dladm show-vnic vnic0
LINK    OVER    SPEED  MACADDRESS       MACADDRTYPE  VID
vnic0   bge0    1000   2:8:20:5e:69:f2  random       0
# ipadm create-addr -T static -a 192.168.17.2/24 vnic0/v4addr
# ipadm show-addr
ADDROBJ       TYPE     STATE      ADDR
<other output>
vnic0/v4ddr   static   ok         192.168.17.2/24
```

Step 3: Create the Flows

We'd like to assign different bandwidths to the different types of traffic, and to do that sort of filtering we employing the 'flow' mechanism of Oracle Solaris 11. In this case we will set up two filters, one for http traffic and one for https traffic. The first will be called 'http-flow', the second, 'https-flow'.

```
# flowadm add-flow -l bge0 -a transport=TCP,local_port=80 http-flow
# flowadm add-flow -l bge0 -a transport=TCP,local_port=443 https-flow
```

Step 4: Set Bandwidth on Flows

Once we have defined a flow that characterizes the type of traffic, we can then apply resource management controls to that flow. In this case we set a bandwidth limit of 100Mb/s on http traffic and https traffic through link0 (but not on other types of traffic through link0), and 500Mb/s for the

```
# dladm set-linkprop -p maxbw=500M vnic0
# flowadm set-flowprop -p maxbw=100M http-flow
# flowadm set-flowprop -p maxbw=100M https-flow
```

traffic through vinic0.

Step 5: Review the Configuration

Now review the settings for vnic0 as well as the special treatment for http and https traffic through the two flows.

```
# dladm show-linkprop -p maxbw vnic0
LINK       PROPERTY   PERM  VALUE  DEFAULT POSSIBLE
vnic0      maxbw      rw    500     --     --
# flowadm show-flowprop http-flow
FLOW       PROPERTY       VALUE   DEFAULT POSSIBLE
http-flow  maxbw          100     --      100M
http-flow  priority       --      --      ?
# flowadm show-flowprop https-flow
FLOW       PROPERTY       VALUE   DEFAULT POSSIBLE
https-flow maxbw          100     --      100m
http-flow  priority       --      --      ?
```

Although not shown it is also possible to specify the properties at the time of creation of a VNIC.

Step 6: Clean up

```
# dladm show-vnic
LINK         OVER     SPEED  MACADDRESS        MACADDRTYPE         VID
vnic0        nge0     100    2:8:20:67:9f:89   random              0
# ifconfig vnic0 down
# ifconfig vnic0 unplumb
# dladm delete-vnic vnic0
# flowadm show-flow
FLOW        LINK        IPADDR      PROTO  LPORT   RPORT   DSFLD
http-flow   nge0        --           tcp    80      --      --
https-flow  nge0        --           tcp    443     --      --
# flowadm remove-flow http-flow
# flowadm remove-flow https-flow
```

Example 2: Network in a Box

This example is a multi-tiered application. There are several web servers running in dedicated Solaris Zones that are exposed to the physical network. There are several database servers running in dedicated Solaris Zones that are not exposed to the physical network. The web servers share a back-end etherstub (virtual switch) called webswitch0 in the following diagram. The database servers also share a backend etherstub, labeled dbswitch0 in the diagram. The network will be configured so that the database zones will have no direct access to the physical network, which securely insulates them from outside communications. The network and zone configuration looks like this:

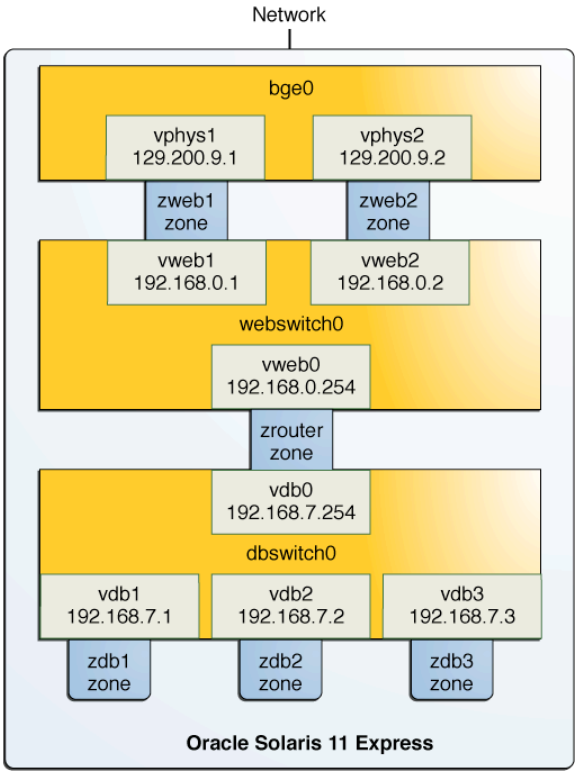

Figure 6 – Network-in-a-box Example

Step 1: Disable NWAM and set up network

See Example 1 above.

Step 2: Create the Etherstubs and VNICs

In this step, create the two virtual switches (etherstubs) and the VNICs used in the six zones, as pictured in Diagram 6. First find out what your physical NIC name is:

```
# dladm show-phys
LINK    MEDIA       STATE     SPEED      DUPLEX    DEVICE
bge0    Ethernet     up       1000       full      bge0
```

For this system, there is only one physical interface, 'bge0'. We will start by creating two virtual NICs on the physical interface, bge0.

Then create the virtual switches (etherstubs) and VNICs for the rest of the interfaces shown in Figure 6.

```
# dladm create-etherstub webswitch0
# dladm create-etherstub dbswitch0
# dladm create-vnic -l webswitch0 vweb0
# dladm create-vnic -l webswitch0 vweb1
# dladm create-vnic -l webswitch0 vweb2
# dladm create-vnic -l dbswitch0 vdb0
# dladm create-vnic -l dbswitch0 vdb1
# dladm create-vnic -l dbswitch0 vdb2
# dladm create-vnic -l dbswitch0 vdb3
```

Check the setup:

```
# dladm show-vnic
LINK        OVER        SPEED  MACADDRESS      MACADDRTYPE  VID
vphys1      bge0        100    2:8:20:d1:b6:26 random       0
vphys2      bge0        100    2:8:20:ae:69:f7 random       0
vweb0       webswitch0  0      2:8:20:ae:89:e0 random       0
vweb1       webswitch0  0      2:8:20:77:c6:1  random       0
vweb2       webswitch0  0      2:8:20:51:35:ec random       0
vdb0        dbswitch0   0      2:8:20:b1:85:b1 random       0
vdb1        dbswitch0   0      2:8:20:d7:6f:df random       0
vdb2        dbswitch0   0      2:8:20:22:38:2e random       0
vdb3        dbswitch0   0      2:8:20:f5:b5:ce random       0
```

Step 3: Create the ZFS Pool for Storing the Zones

We are going to create a single Solaris Zone and then, to reduce the time to create additional ones, we will clone the first. Cloned zones require sharing a ZFS dataset, so creating a ZFS dataset is the first step.

```
# zfs create -o mountpoint=/zonefs rpool/zonefs
# chmod 700 /zonefs
```

Step 4: Create a Zone for Cloning

In general it is much faster to clone a zone than to create a zone from scratch under Oracle Solaris 11, so we will use the cloning technique in this example to first create one zone and then clone it 6 times.

Configure the clone zone:

```
# zonecfg -z zclone
zclone: No such zone configured
Use 'create' to begin configuring a new zone.
zonecfg:zclone> create
zonecfg:zclone> set zonepath=/zonefs/zclone
zonecfg:zclone> set ip-type=exclusive
zonecfg:zclone> verify
zonecfg:zclone> commit
zonecfg:zclone> exit
# zoneadm -z zclone install
zoneadm: zclone: No such zone configured
```

Now install the zone

```
# zoneadm -z zclone install
A ZFS file system has been created for this zone.
   Publisher: Using solaris
(http://ipkg.sfbay.sun.com/dev/solaris/ ).
       Image: Preparing at /zonefs/zclone/root.
       Cache: Using /var/pkg/download.

<lots of output including Oracle Technical Network license>

Done: Installation completed in 136.245 seconds.

  Next Steps: Boot the zone, then log into the zone console
              (zlogin -C) to complete the configuration process.
```

The install should take approximately 2 minutes assuming that you connect to pkg.oracle.com over a high speed internet connection. A faster alternative would be to build a repository on your own local area network.

Now boot the clone zone:

```
# zoneadm -z zclone boot; zlogin -C zclone
[Connected to zone 'zclone' console]
Loading smf(5) service descriptions: 100/100
Hostname: zclone
Loading smf(5) service descriptions: 3/3



What type of terminal are you using?
 1) ANSI Standard CRT
 2) DEC VT100
 3) PC Console
 4) Sun Command Tool
 5) Sun Workstation
 6) X Terminal Emulator (xterms)
 7) Other
Type the number of your choice and press Return:
```

We put the zlogin on the same line so that the command will be immediately executed and you will see the zclone zone boot including the initial configuration of the SMF services. You want this step to complete so that when you clone this zone you won't have to wait for each clone to configure SMF services.

You know it is safe to halt the clone when you are prompted to select a terminal type. Do not select one. Instead enter ~. (tilde, dot) to escape to the global zone,

```
Type the number of your choice and press Return: ~.
```

and then halt the creation of zclone.

```
# zoneadm -z zclone halt
```

To review, we have created a clone has gone through the initial install and first boot, which configured the SMF services. At that point we stop the process, because we don't want to further configure the zone yet. As you will see, creating this clone will save us the time associated with initially provisioning a zone and the time for the first. All we will need to do to use this clone is to give it a network identity and provide some other initialization information.

Step 5: Create an Example Zone

In Diagram 6 we see 6 zones. To simplify the typing, the approach we will take is to walk through setting up one zone as an example and then suggest you use some scripts that are provided in the appendix to automate the process.

We are going to first create one of the database zones, zdb1, i.e. one of the zones which would contain a database in our example network. We aren't actually going to provision with a database server, since this example is only about setting up the networking infrastructure.

First we get the configuration information from the clone zone:

```
# zonecfg -z zclone export | zonecfg -z zdb1 -f -
```

Then designate where to set up the zone files, add the VNIC, (vdb1) and create the zdb1 zone:

```
# zonecfg -z zdb1 "set zonepath=/zonefs/zdb1"
# zonecfg -z zdb1 "add net;set physical=vdb1;end"
# zoneadm -z zdb1 clone zclone
```

Note how fast it is possible to clone a zone, compared to installing a new one from scratch.

For ZFS-based zones, you now have to 'ready' the zone to see the zone's file system from the global zone. Explore /zonefs/zdb1 before and after executing the following command.

```
# zoneadm -z zdb1 ready
```

Now we are ready to configure zdb1 and give it a network personality. As you saw above when you created the zclone zone, and then zlogin'ed into it, you were asked to specify a terminal type. If you had answered, you would have gone through the entire sysidtool configuration process (see sysidtool(1M) man page for details). In other words you would have been asked for hostname, IP address, name services, time zone location, i.e. all the information for which you are prompted after doing a sys-unconfig on a Solaris 10 system.

You can set up a configuration file, sysidcfg, in advance, and copy it into the zone so that you won't be asked any questions. This however is a transition solution and will not be the approach used for Oracle Solaris 11. Also for Solaris 11 Express, not all the features of the Oracle Solaris 10 version of the sysidcfg files are supported. One example is the root_password property. Oracle Solaris 11 Express uses a different password encryption mechanism that sysidtool does not recognize. Fortunately Oracle Solaris 11 Express can process passwords generated on earlier versions of Oracle Solaris 10. So the work around is to generate the root_password encryption on an Oracle Solaris 10 system, and cut and paste that into the input for the configuration file. **In this case the password-encrypted string corresponds to the plain text password "crossbow".**

In this next step we will create the sysidcg file for pre-configuring the zdb1 zone so that when we login into the zone the first time, we won't be prompted to enter any configuration information.

You will want to create the file /zonefs/zdb1/root/etc/sysidcfg which contains the following lines.

```
terminal=vt100
system_locale=C
timezone=US/Pacific
nfs4_domain=dynamic
security_policy=NONE
root_password=rJmv5LUXM1OcU
network_interface=PRIMARY {
hostname=zdb1
ip_address=192.168.7.1
netmask=255.255.255.0
protocol_ipv6=no default_route=none
}
name_service=NONE
```

NOTE: we are using the IP addresses/netmask as specified in Figure 6.

Boot the zone zdb1 and check that it is running.

```
# zoneadm -z zdb1 boot
# zoneadm list -cv zdb1
  ID  NAME    STATUS      PATH           BRAND    IP
  0   global running      /              native   shared
  78  zdb1    running      /zonefs/zdb1   ipkg     excl
  -   zclone installed    /zonefs/zclone ipkg   excl
```

Step 6: Create the Rest of the Zones

As you can see from the above, generating the configuration takes quite a bit of typing. To simplify creating the zones, there are scripts for both creating and cleaning up zones in the Appendix. You should copy and paste those scripts into the appropriate file names. See the Appendix for more instructions.

Assuming you put the scripts in /opt/crossbowhowto, then you should see them all with:

```
# cd /opt/crossbowhowto/scripts
# ls
cleanupallzones.sh    cleanupzone.sh       createzone-2vnic.sh
cleanupdladm.sh       createzone-1vnic.sh
```

To start, you already created zdb1 in Step 5 so now create the other two.

Create zdb2 and zdb3 zones:

```
# sh createzone-1vnic.sh zdb2 vdb2 zdb2 192.168.7.2
# sh createzone-1vnic.sh zdb3 vdb3 zdb3 192.168.7.3
```

Now create the router zone. To keep the example simple we are going to use the default Oracle Solaris router for handling routing tasks. If the routing had been more complex or we wanted to use other routing protocols, we could have used the open source quagga project, which can be found in the Oracle Solaris 11 Express package repository. See quagga(8).

Note the router zone has two VNICs so we use a different script. You will create the zrouter zone here, and configure the routing in it later.

```
# sh createzone-2vnics.sh zrouter vweb0 zrouter 192.168.0.254
vdb0 zrouter-vdb0 192.168.7.254
```

And finally, create the two zones in which the web servers would be run.

Create zweb1 and zweb2 zones. You will need to replace the 129.200.9 addresses with addresses on your subnet, since the vphys1 and vphys2 addresses are seen by external systems.

```
# sh -x createzone-2vnics.sh zweb1 vphys1 zweb1 129.200.9.1 vweb1
zweb1-vweb 192.168.0.1
# sh -x createzone-2vnics.sh zweb2 vphys2 zweb2 129.200.9.2 vweb2
zweb2-vweb 192.168.0.2
```

We won't actually install a web server. We only illustrate in this example how to set up the networking for this arrangement.

Finally verify the setup:

```
# zoneadm list -cv
ID NAME    STATUS      PATH              BRAND    IP
0  global  running     /                 native   shared
78 zdb1    running     /zonefs/zdb1      ipkg     excl
81 zdb2    running     /zonefs/zdb2      ipkg     excl
82 zdb3    running     /zonefs/zdb3      ipkg     excl
83 zrouter running     /zonefs/zrouter   ipkg     excl
84 zweb1   running     /zonefs/zweb1     ipkg     excl
85 zweb2   running     /zonefs/zweb2     ipkg     excl
- zclone   installed   /zonefs/zclone    ipkg     excl
```

Step 7: Set up Routing for the Zones

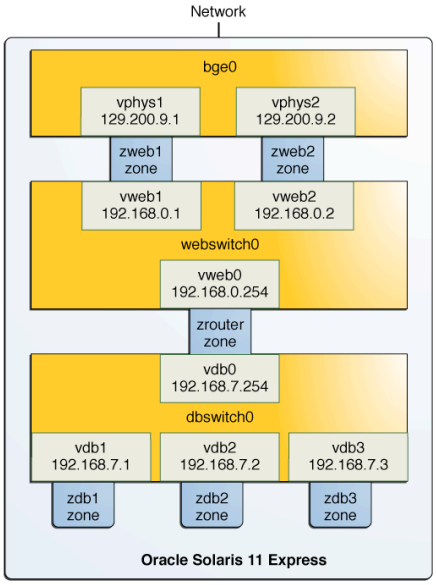

Figure 7 - The example again

Now you must log in and specify some default routing information for each zone. If you just finished running the script, you may want to wait a few minutes because your zlogin below may not immediately succeed due to some characteristics specific to this release.

Always log into a zone the first time using the -C flag. This will invoke sysidtool to prompt you for input. You provided this information prior to booting via the sysidcfg file, so you shouldn't be prompted. But if you made any changes to the sysidcfg file contents, you may trigger the prompting.

```
# zlogin –C zweb1
[Connected to zone 'zweb1' console]
```

If you don't see the following login prompt, press the return key:

```
zweb1 console login: root
Password: crossbow

Oracle Corporation      SunOS 5.11      snv_151a    November 2010
root@zweb1:~#
root@zweb1:~# route -p add net 192.168.7.0/24 192.168.0.254
root@zweb1:~# logout
zweb1 console login:~.
```

Again, we use "~." to exit zlogin and return to the global zone.

Next zlogin in to zweb2 and specify the same routing

```
root@zweb2:~# route -p add net 192.168.7.0/24 192.168.0.254
```

and exit that zone. Now zlogin to each of the database zones and provide the route to the 192.168.0.0 network:

```
root@zdb1:~# route -p add net 192.168.0.0/24 192.168.7.254
```

```
root@zdb2:~# route -p add net 192.168.0.0/24 192.168.7.254
```

```
root@zdb3:~# route -p add net 192.168.0.0/24 192.168.7.254
```

Finally configure zrouter, which routes packets between the webserver zones and the database zones. However before doing this, try zlogin to zweb1 and

```
# zlogin -C zweb1

root@zweb1:~# ping 192.168.7.254
```

which should work and

```
root@zweb1:~# ping 192.168.7.1
```

which should not work because you haven't told the zrouter zone to forward packets between the 192.168.7 and 192.168.0 networks. You can easily fix the latter by setting up ip forwarding in the zrouter zone. Get back to the global zone and

```
# zlogin -C zrouter
.
.
root@zrouter:~# svcadm enable network/ipv4-forwarding:default
```

You can now try various experiments by using the ping or traceroute commands between various zones:

- Try to reach zweb1 or zweb2 from an external system on the same subnet as those addresses (129.200.1 in this example). You should succeed.
- Try to reach one of the IP addresses on the 192.168.0 or 192.168.7 networks from an external system and you should not be successful because IP forwarding is not on for zweb1 or zweb2.
- Try to reach one of the database zones, zdb1, zdb2, zd3 (192.168.7.1, 2, or 3) from zweb1 or zweb2, and because IP forwarding is turned on for zrouter, you should be successful.
- And try the opposite- from zdb1, zdb2, or zdb 3, try to reach 192.168.0.1 or 2. Again because of the zrouter configuration you should succeed.

Clean Up

Finally, see the appendix for three scripts to clean up the zones and the VNICs and restore the system to its original state.

```
# sh cleanupallzones.sh
# sh cleanupdladm.sh
# zfs destroy -r rpool/zonefs
```

Conclusion

We hope this step by step guide will give you some ideas for future experimentation. With Oracle Solaris 11 Express capabilities you can easily set up fairly complex environments. And using additional facilities not discussed here but available through the Oracle Solaris 11 Express repository, like IP Filter firewall, Quagga routing package, and new load balancer, you can address very complex networking requirements.

For More Information

See the Oracle White Paper - Oracle Solaris 11 Express Network Virtualization and Network Resource Management for an overview of Project Crossbow.

For details on using network virtualization and network resource management capabilities, see the product documentation, in the System Administrative Guide: Network Interfaces and Network Virtualization manual.

An engineering paper on Project Crossbow awarded Best Paper by the Large Installation System Administration (LISA) 2009 conference provides a technical overview: Crossbow Virtual Wire: Network in a Box.

Appendix

You will find four scripts in this appendix. The recommendation is to cut and paste their contents into the suggested file names using gedit to create the files on Oracle Solaris 11 Express.

Following is the first script, createzone-1vnic.sh for creating a zone with one VNIC.

```
#!/bin/sh
#
# FILENAME:    createzone-1vnic.sh
#

if [ $# != 4 ]
then
echo "Usage: createzone-1vnic.sh <zonename> <vnic1> <hostname for vnic1>"
echo "                            <ip addr for vnic1>"
exit 1
fi
ZONENAME=$1
VNIC1=$2
VNIC1HOSTNAME=$3
VNIC1IP=$4

echo "Create zone $ZONENAME" with
echo " Interface $VNIC1, hostname $VNIC1HOSTNAME at IP Address $VNIC1IP"

zonecfg -z zclone export | zonecfg -z $ZONENAME -f -
zonecfg -z $ZONENAME "set zonepath=/zonefs/$ZONENAME"
# add the VNIC
zonecfg -z $ZONENAME "add net;set physical=$VNIC1;end"
# now create the new zone from the clone
zoneadm -z $ZONENAME clone zclone
# For ZFS based zones, you have to 'ready' them to see the zone's file
# system from the global zone.
#
zoneadm -z $ZONENAME ready

# Note, you can not generate a root password under Oracle Solaris 11
# and use that encrypted string for the root_password property.
# The work around is to generate a password on Oracle Solaris 10 instead,
# and use that encryption (from /etc/shadow) for the root_password
# setting.

cat > /zonefs/$ZONENAME/root/etc/sysidcfg << _EOF_
terminal=vt100
system_locale=C
timezone=US/Pacific
nfs4_domain=dynamic
security_policy=NONE
root_password=rJmv5LUXM1OcU
network_interface=PRIMARY {
hostname=$VNIC1HOSTNAME
ip_address=$VNIC1IP
netmask=255.255.255.0
protocol_ipv6=no
default_route=none
}
name_service=NONE
_EOF_
zoneadm -z $ZONENAME boot
#END FILE createzone-1vnic.sh
```

Following is the second script, createzone-2vnics.sh, for creating a zone with 2 VNICs.

```
#!/bin/sh
#
# FILENAME: createzone-2vnics.sh
#

if [ $# != 7 ]
then
       echo "usage: createzone-2vnics.sh <zonename> <vnic1> <hostname for vnic1>"
       echo "                            <ip address for vnic1> <vnic2> <hostname for vnic2>"
       echo "                            <ip address for vnic2>"
       exit 1
fi
ZONENAME=$1
VNIC1=$2
VNIC1HOSTNAME=$3
VNIC1IP=$4
VNIC2=$5
VNIC2HOSTNAME=$6
VNIC2IP=$7
echo "Create zone $ZONENAME" with
echo " Interface $VNIC1, hostname $VNIC1HOSTNAME at IP Address $VNIC1IP"
echo " Interface $VNIC2, hostname $VNIC2HOSTNAME at IP Address $VNIC2IP"
zonecfg -z zclone export | zonecfg -z $ZONENAME -f -
zonecfg -z $ZONENAME "set zonepath=/zonefs/$ZONENAME"
# add the two VNICs
zonecfg -z $ZONENAME "add net;set physical=$VNIC1;end"
zonecfg -z $ZONENAME "add net;set physical=$VNIC2;end"
# now create the new zone from the clone
zoneadm -z $ZONENAME clone zclone
# For ZFS based zones, you have to 'ready' them to see the zone's
# file system from the global zone.
zoneadm -z $ZONENAME ready

# Note, you can not generate a root password under Oracle Solaris 11
# and use that encrypted string for the root_password property.
# The work around is to generate a password on Oracle Solaris 10 instead,
# and use that encryption (from /etc/shadow) for the root_password
# setting.
cat > /zonefs/$ZONENAME/root/etc/sysidcfg << _EOF_
terminal=vt100
system_locale=C
timezone=US/Pacific
nfs4_domain=dynamic
security_policy=NONE
root_password=rJmv5LUXM1OcU
network_interface=$VNIC2 {
hostname=$VNIC2HOSTNAME
ip_address=$VNIC2IP
netmask=255.255.255.0
protocol_ipv6=no
default_route=none
}
network_interface=$VNIC1 {
primary hostname=$VNIC1HOSTNAME
ip_address=$VNIC1IP
netmask=255.255.255.0
protocol_ipv6=no
default_route=none
}
name_service=NONE
_EOF_
zoneadm -z $ZONENAME boot
# END FILE createzone-2vnics.sh
```

Following is the third script, cleanupzone.sh. This can be used to completely remove a zone.

```
#!/bin/sh
#
# FILENAME: cleanupzone.sh
#
# Usage: cleanupzone.sh <zone name>
#
# This will completely remove a zone from the system
#
if [ $# != 1 ]
then
      echo "Usage: cleanupzone <zone name>"
      exit 1
fi
echo 'zoneadm -z '$1' halt'
zoneadm -z $1 halt
echo 'zoneadm -z '$1' uninstall -F'
zoneadm -z $1 uninstall -F
echo 'zonecfg -z '$1' delete -F'
zonecfg -z $1 delete -F
#
# END FILE cleanupzone.sh
```

This fourth script can be used to remove all the zones.

```
#!/bin/sh
#
# FILENAME: cleanupallzones.sh
# Usage: cleanupallzones.sh
#
# This will completely remove all the Crossbow demo zones
sh cleanupzone.sh zweb1
sh cleanupzone.sh zweb2
sh cleanupzone.sh zrouter
sh cleanupzone.sh zdb1
sh cleanupzone.sh zdb2
sh cleanupzone.sh zdb3
sh cleanupzone.sh zclone
#
# END FILE cleanupallzones.sh
```

The fifth script is for after removing all the zones from a system. Use this script to remove all the VNICs and etherstubs that you created.

```
#!/bin/sh
#
# FILENAME: cleanupdladm.sh
#
# Usage: cleanupdladm
#
# This will completely remove the crossbow network-in-a-box
# VNICs and etherstubs
#
dladm delete-vnic
vphys1 dladm delete-vnic
vphys2 dladm delete-vnic
vweb0 dladm delete-vnic
vweb1 dladm delete-vnic
vweb2 dladm delete-vnic vdb0
dladm delete-vnic vdb1
dladm delete-vnic vdb2
dladm delete-vnic vdb3
dladm delete-etherstub webswitch0
dladm delete-etherstub dbswitch0
#
# END FILE cleanupdladm.sh
```

Oracle is committed to developing practices and products that help protect the environment

Exploring Network Virtualization With Oracle Solaris Zones on Oracle Solaris 11 Express
Feb 2011
Author: Jeff McMeekin
Contributing Authors: Nicolas Droux, Duncan Hardie

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
oracle.com

SOFTWARE. HARDWARE. COMPLETE.