



New Features

**Solaris 10™ OS 10/08
(Solaris 10 Update 6)**



Stephen Mohr

SL: Sidifen Yiyuan
Senior Instructor
Collier Computing

Based on slides from:
Kristi Herd
Sun Microsystems

Solaris 10, Update 6 – 10/08

- New features at a glance:
 - > **Boot procedure**
 - > SPARC and x86 use same: miniroot/ramdisk, universal
 - > **ZFS**
 - > Bootable, gzip compression, history and upgrade commands, migrate (UFS to ZFS), refquota, delegation
 - > **Zones**
 - > Package updates, default router, ZFS zonepath
 - > Intel support
 - > Support for new processors, peripherals, fault management
 - > Virtualization
 - > xVM
- This presentation will cover **ZFS**, **Boot**, and **Zones**

Boot

New Solaris Boot Design – SPARC

- Solaris SPARC bootstrap has been redesigned to be inline with Solaris x86.
 - > Solaris boot archives are now available on the SPARC. There are two types on each platform:
 - > Primary boot archive
 - > Failsafe boot archive
 - > A ramdisk miniroot is supported on the SPARC platform.
 - > With this simplified network boot architecture, other non-UFS file systems such as ZFS are easily supported as root file systems.

New Solaris Boot Design – SPARC

Features

- The `bootadm` command now works on the SPARC platform. It functions the same as on x86. It can be used manually to manage the boot archives:

```
# bootadm update-archive
```

- The `bootadm` command automatically updates the GRUB menu (x86 only) during an installation or upgrade.
- The boot archive service is controlled with the `svcadm` command (SMF):

```
svc:/system/boot-archive:default
```
- Supports booting a ZFS file system or a single miniroot for installation, as well as booting from DVD, NFS, or HTTP.

New Solaris Boot Design – SPARC

Architecture

- SPARC continues to use the Open Boot Prom (OBP)
- x86 uses the BIOS and Grand Unified Bootloader (GRUB) menu.
- On the SPARC and x86 platforms, there is one primary boot archive and one failsafe boot archive.

New Solaris Boot Design – SPARC

The four phases of new booting:

1. Boot Loader – loads the Solaris root file system archive from the media (disk or network interface) to memory. The SPARC boot loader is still the OBP.
2. Booter – reads the ramdisk directly and executes it.
3. Ramdisk – a boot archive (same as x86) containing either kernel modules or an install miniroot. The ramdisk extracts the kernel image from the boot archive and executes it.
4. Kernel – extracts the rest of the primary modules from the boot archive, initializes itself, mounts the real root file system, and discards the boot archive (same as x86).

ZFS

ZFS Install/Boot

- You can now boot using the ZFS file system, in one of the following ways:
 - > Select ZFS as the root file system during an initial installation.
 - > Migrate a UFS root file system to ZFS using Solaris Live Upgrade. A ZFS pool must exist before the Live Upgrade migration operation.
 - > Create a ZFS storage pool and designate a bootable ZFS file system from a custom Jumpstart profile.

ZFS Install/Boot

Features – SPARC

- To support booting from ZFS on the Solaris SPARC platform, two new OBP options have been added:
 - > The `boot -L` option displays a list of available bootable datasets within a ZFS pool.
 - > The `boot -Z dataset` option boots the root file system for the specified ZFS bootable dataset.

ZFS Install/Boot

Example – list bootable datasets

ok **boot -L** <- Lists the datasets at OBP, then prints instructions on the screen

...

```
Boot device: /pci@1f,0/pci@1/scsi@8/disk@0,0 File  
and args: -L
```

```
zfs-file-system
```

```
Loading: /platformsun4u/boot1st
```

```
1.s10s_nbu6wos
```

```
2 zfs2BE
```

```
Select environment to boot: [ 1 - 2 ]: 2
```

```
to boot the selected entry, invoke:
```

```
boot [<root-device>] -Z rpool/ROOT/zfs2BE
```

ZFS Install/Boot

Features

- Also included with the new ZFS Install:
 - > Interactive text installer for a UFS (default) or ZFS root file system.
 - > Custom JumpStart features to set up a profile to create a ZFS storage pool and designate a bootable ZFS file system.
 - > The `lucreate` and `luactivate` commands are enhanced to support ZFS pools and file systems.
 - > Create mirrored ZFS root pools by selecting two disks during installation, or add additional disks after installation.

ZFS Install/Boot

Migrating a UFS File System to a ZFS File System

- Migrating to a ZFS root file system requires an existing ZFS storage pool, designated with the `-p` option.
- Only zones that are in a non-shared file system are migrated from UFS to ZFS. If the zone is in a shared UFS file system, it must be migrated as in previous Solaris releases.
- If a zone is cloned when upgrading within the same ZFS pool, it is migrated.
- Live Upgrade uses the ZFS snapshot and clone features when creating a ZFS Bootable Environment in the same pool, so BE creation is much faster.

ZFS Install/Boot

Requirements, Dependencies, and Limitations

- 786 Mbytes of memory is required to install a ZFS root file system.
- 1 Gbyte of memory is recommended for overall ZFS performance.
- At least 16 Gbytes of disk space is recommended.

ZFS Roll Back a Dataset Without Unmounting

- The ability to rollback a dataset without unmounting it first.
- This enhancement means that the `-f` option to force an unmount operation is no longer needed or supported, and is ignored if specified:

```
# zfs rollback -f
```

ZFS send Command

- The ZFS `send` command has been updated:
 - > Send all incremental streams from one snapshot to a cumulative snapshot.
 - > Send an incremental stream from the original snapshot to create a clone.
 - > Send a replication stream of all descendent file systems, up to the named snapshots, preserving all properties, snapshots, clones, and descendent file systems.
 - > Send an incremental replication stream.

ZFS Quotas and Reservations for File System Data Only

- Quotas and reservations in ZFS previously included the space consumed by descendents such as snapshots and clones.
- In Solaris 10™ OS 10/08, ZFS additionally has filesystem-only quotas and reservations:
 - > `refquota` – limits the amount of space a dataset can consume. This property enforces a hard limit on the amount of space that can be used. This hard limit does not include space used by descendents, such as snapshots and clones.
 - > `refreservation` – sets the minimum amount of space that is guaranteed to a dataset, not including its descendents.

ZFS Quotas and Reservations for File System Data Only

Example

- The following example sets a 10 Gbyte hard limit of referenced space for student A using `refquota`. For additional flexibility, a 20 Gbyte `quota` is set for management of student A's snapshots.

```
# zfs set refquota=10g tank/studentA
```

```
# zfs set quota=20g tank/studentA
```

- For more information and detailed examples, see the ZFS Admin Guide:

ZFS Quotas and Reservations

ZFS Storage Pool Property

- The following property has been added to ZFS Storage Pools:
 - > `failmode` – determines the behaviour of a pool failure due to a loss of device connectivity, or the failure of all devices in the pool.
 - > **wait** (default behaviour) – blocks all I/O access until the device connectivity is restored by reconnecting the device or replacing a failed device, and then it clears the error with the `zpool clear` command.
 - > **Continue** – returns EIO to any new write I/O requests, but allows reads to any of the remaining operable devices.
 - > **panic** – prints out a message to the console and generates a system crash dump.

ZFS `zpool history` Enhancements

- The following enhancements have been made to the ZFS `zpool history` command:
 - > `zpool history users` – displays file system event information.
 - > `zpool history -l users` – displays the system event information in long format.
 - > `zpool history -i users` – displays the system event information with internal event information, handy for diagnostic purposes.

Upgrading ZFS File Systems

- `zfs upgrade` command is included to provide future ZFS file system enhancements to existing file systems
- `zpool upgrade` provides pool enhancements to existing storage pools is also available
- File systems that are upgraded and any streams created from those upgraded file systems by the `zfs send` command are not accessible on systems that are running older software releases.

ZFS Delegated Administration

- Fine-grained permissions to perform ZFS administration tasks can be delegated to non-privileged users with the `zfs allow` and `zfs unallow` commands.
- Modify the ability to use delegated administration with the pool's `delegation` property.
- By default, the `delegation` property is `on`.

Separate ZFS Logging Devices

Option to set up a separate ZFS logging device

- The ZFS Intent Log (ZIL) is provided to satisfy POSIX (Portable Operating System for UNIX) requirements for synchronous transactions.
- By default, the ZIL is allocated from blocks within the main storage pool. However, better performance might be possible using separate intent log devices in the ZFS storage pool such as with Non-Volatile RAM (NVRAM), or a dedicated disk (speed).

Creating Intermediate ZFS Datasets

- With Solaris 10™ OS 10/08, the `zfs` sub-commands `create`, `clone`, and `rename` support a new `-p` option which quickly creates an intermediate dataset, if one doesn't already exist (similar to `mkdir -p`).
- For example, to create ZFS datasets (`users/area51`) in the `datab` storage pool (`users` does not exist):

```
# zfs list
NAME                                USED  AVAIL  REFER  MOUNTPOINT
datab                                106K  16.5G   18K    /datab
# zfs create -p -o compression=on datab/users/area51
#
```

ZFS Hotplugging Enhancements

- ZFS now responds more effectively to devices that are removed
- Provides the ability to automatically identify devices that are inserted:
 - > If `autoreplace` property is set to `off` (default), device replacement must be initiated using the `zpool replace` command.
 - > If the property is set to `on`, any new device found in the same physical location as a device the previously belonged to the pool is automatically formatted and replaced.

ZFS Hotplugging Enhancements

- > The storage pool state `REMOVED` is provided when a device or hot spare is removed, if the device was physically removed while the system was running. A hot spare device is substituted for the removed device, if available.
- > If a device is removed and then inserted, the device is placed online. If a hot spare was activated when the device is re-inserted, the spare is removed when the online operation completes.
- > Hot spares are checked periodically to make sure they are online and available.

Recursive Rename of ZFS Snapshots

- `zfs rename -r` command recursively renames all descendent ZFS snapshots.
- For example, to recursively rename all snapshots called 'today' to 'yesterday':

```
# zfs rename -r users/home@today @yesterday
```

GZIP Compression In ZFS

- `gzip` compression for ZFS file systems is available in addition to the default `lzjb`.
- The compression can be specified as `lzjb`, `gzip`, or `gzip-N` where `N` equals `1-9`.
For example:

```
# zfs create -o compression=gzip-9  
users/home/oldfiles  
  
# zfs get compression users/home/oldfiles  
NAME                PROPERTY            VALUE  
SOURCE  
users/home/oldfiles  compression         gzip-9  
local
```

Multiple Copies of ZFS User Data

- ZFS file system metadata is automatically stored multiple times across different disks, if possible, and is known as ditto blocks.
- With Solaris 10™ OS 10/08, specifying either 1 (default), 2, or 3 copies of user data is also available
- This feature:
 - > Improves data retention by allowing recovery from unrecoverable block read faults, such as media faults.
 - > Provides data protection even when only a single disk is available.
 - > Allows selection of data protection policies on a per-file system basis, beyond the capabilities of the storage pool.

Zones

New `-u` Option to `zoneadm`

- This feature examines the zone that is being migrated and determines which packages must be updated to match the new host. Only those packages are updated. The rest of the packages, and their associated patches, can vary from zone to zone.

```
host2# zoneadm -z myzone attach -u
```

- If a new host has the same or later versions of the zone-dependent packages and their associated patches, it updates those packages within the zone to match the new host.
- The `-u` option also enables automatic migration between machine classes, such as from `sun4u` to `sun4v`.

Default Router in a Shared IP Zone

- An optional `defrouter` property is added to the `net` resource in the `zonecfg` utility for non-global zones that share an IP address. The default router for the network interface is set through this property.
- For example, to set the `net` resource for a shared-IP zone, adding address `192.168.0.1`, physical interface `bge0`, and a default router setting, type the following:

```
zonecfg:myzone> add net
zonecfg:myzone:net> set address=192.168.0.1
zonecfg:myzone:net> set physical=bge0
zonecfg:my-zone:net> set defrouter=10.0.0.1
zonecfg:my-zone:net> end
```

ZFS Zone Path Permitted

- `zonepath` of a non-global zone can be on ZFS, and Solaris Live Upgrade can be used to upgrade the zone.
- Solaris Live Upgrade feature will migrate zones to a ZFS root file system.
- A zone in a non-shared file system is automatically migrated when the UFS root file system is migrated to a ZFS root file system.
- If the zone is in a shared UFS file system, then the zone must be upgraded as in previous Solaris releases.

Summary:

- New features covered:
 - > **Boot procedure**
 - > SPARC and x86 use same: miniroot/ramdisk, universal
 - > **ZFS**
 - > Bootable, gzip compression, history and upgrade commands, migrate (UFS to ZFS), refquota, delegation,
 - > **Zones**
 - > Package updates, default router, ZFS zonepath

For More Information:

- www.sun.com/solaris
- www.docs.sun.com
- System Administration Guide
- ZFS Administration Guide
- Solaris 10/08 Release Notes
- BigAdmin
- Man pages



Thank you for attending!!

What's New in Solaris 10™ OS 10/08
(Update 6)

THE END