Using Amanda Enterprise 3.0 on Sun Fire™ X4500 Server With OpenSolaris™ 2008.11 OS

Caryl Takvorian

May 2009

Sun Microsystems, Inc.

This article was published by BigAdmin at:
http://www.sun.com/bigadmin/features/articles/amanda_opensolaris.jsp
To keep track of the latest content published by BigAdmin, subscribe to the BigAdmin newsletter: http://www.sun.com/bigadmin/newsletter/.

Table of Contents

Introduction

The objective of this paper is to describe how to deploy Amanda Enterprise 3.0 from Zmanda on OpenSolaris™ 2008.11 OS on a Sun Fire™ X4500 server using ZFS™ file systems, and to show how the Amanda software performs. In particular, the article demonstrates how Amanda Enterprise takes advantage of ZFS snapshot technology to improve overall backup performance.

Note: In this scenario, the Sun Fire X4500 server acted as the backup server, while a Sun Fire X4200 server acted as the backup client.

This document is intended to be read in conjunction with the article *Sun Fire X4540 Server as Backup Server for Zmanda's Amanda Enterprise 2.6 Software* by Thomas Hanvey (Sun Microsystems), Dmitri Joukovski, and Ken Crandall (Zmanda), September 2008, which is available at http://www.sun.com/bigadmin/features/articles/zmanda_sfx4540.jsp.

For more information on Zmanda Recovery Manager (ZRM) Enterprise, see http://www.zmanda.com/amanda-enterprise-edition.html.

Amanda Enterprise is available for download from the Zmanda Network: http://network.zmanda.com/.

Architecture Overview

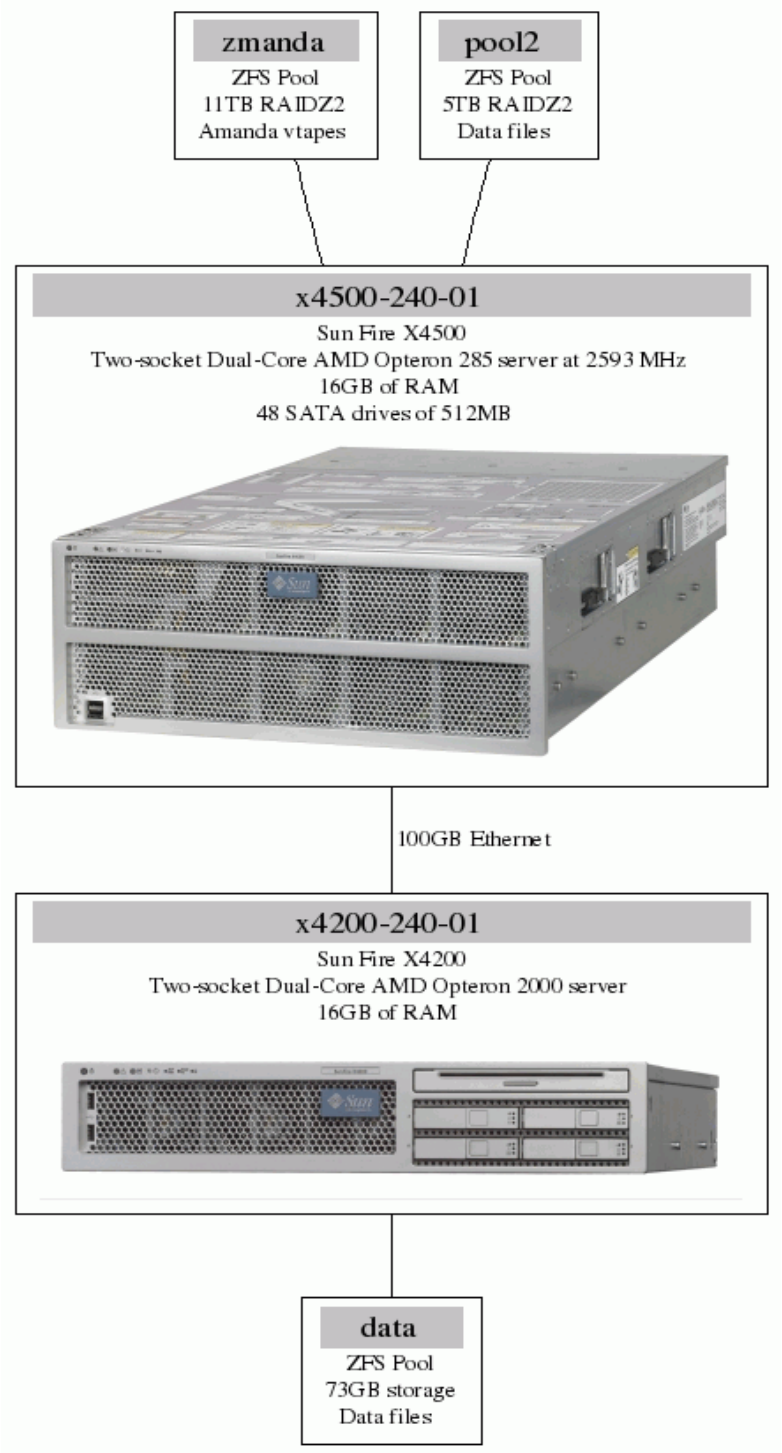


Figure 1: Architecture Overview

OpenSolaris OS Configuration on Sun Fire X4500 Server

For this guide, Amanda Enterprise was deployed on a Sun Fire X4500 server running the OpenSolaris 2008.11 OS release.

The system used during this test is a server with two dual-core AMD Opteron™ 285 processors running at 2593 MHz with 16 Gbyte of RAM and 48 SATA drives of 512 Gbyte each, for a total of 24 Tbyte of storage capacity.

The two boot drives were mirrored using ZFS volume management capabilities and the remaining 46 drives were available for backup data.

For more information on the OpenSolaris OS and the Sun Fire X4500 server, see the "Product Information" section at the end of this article.

ZFS File System Configuration

For more information on ZFS, see the "Product Information" section at the end of this article.

Zpool Configuration

For the purposes of this guide, the storage available on the Sun Fire X4500 server was split into two pools:

- A backup pool made of 24 drives and two spares in a RAIDZ2 configuration. This provided 11 Tbyte of capacity for backups with a very high level of protection against drive failure and performance sufficient to handle 300 to 350 Mbyte/sec of backup traffic over the network.
- A data pool made of 12 drives and two spares in a RAIDZ2 configuration was used to store the data to be backed up during local backups.

First, we created the RAIDZ2 pool `zmanda` with six drives. Then, we added the other drives by groups of six and verified at the end that we ended up with a pool with 10.9 Tbyte of physical space.

Here are the zpool creation commands for pool `zmanda`:

```
# zpool create -f zmanda raidz2 c3t1d0 c4t1d0 c6t1d0 c7t1d0 c8t1d0 c9t1d0
# zpool add zmanda raidz2 c3t2d0 c4t2d0 c6t2d0 c7t2d0 c8t2d0 c9t2d0
# zpool add zmanda raidz2 c3t5d0 c4t5d0 c6t5d0 c7t5d0 c8t5d0 c9t5d0
# zpool add zmanda raidz2 c3t6d0 c4t6d0 c6t6d0 c7t6d0 c8t6d0 c9t6d0
# zpool add zmanda spare c8t0d0 c9t0d0
```

We used a similar command to create the pool `pool2`.

Here are the zpool creation commands for pool `pool2`:

```
# zpool create -f pool2 raidz2 c3t3d0 c4t3d0 c6t3d0 c7t3d0 c8t3d0 c9t3d0
# zpool add -f pool2 raidz2 c3t7d0 c4t7d0 c6t7d0 c7t7d0 c8t7d0 c9t7d0
# zpool add -f pool2 spare c8t4d0 c9t4d0
```

Finally, we checked the ZFS pools that were created:

```
# zpool list
NAME     SIZE   USED  AVAIL    CAP  HEALTH  ALTROOT
pool2   5.44T   225K  5.44T     0%  ONLINE  -
rpool    464G  13.3G   451G     2%  ONLINE  -
zmanda  10.9T   252K  10.9T     0%  ONLINE  -
```

ZFS Configuration

Next, we need to create the ZFS file systems, which will be used by Amanda Enterprise and to store the data.

We need three different ZFS file systems:

- `zmanda/vtapes` to hold the backup sets or virtual tapes
- `zmanda/logs` to hold the log files
- `pool2` to hold the data files to be backed up

So, first let's create the ZFS file system where Amanda will store the backup images, and then mount it in the default location expected by Amanda Enterprise:

```
# zfs create -o mountpoint=/var/lib/amanda/vtapes zmanda/vtapes
```

Then, create the ZFS file system where the data files will be stored and the third file system to store the log files:

```
# zfs create pool2
# zfs create zmanda/logs
```

Finally, we can list the file systems available to us:

```
# zfs list
NAME                       USED  AVAIL  REFER  MOUNTPOINT
pool2                     1.35T  2.22T  1023G  /pool2
rpool                      185G   272G    60K  /rpool
rpool/ROOT                 150G   272G    18K  legacy
rpool/ROOT/OpenSolaris-1   150G   272G   121G  legacy
rpool/dump                7.95G   272G  7.95G  -
rpool/export              19.1G   272G    19K  /export
rpool/export/home         19.1G   272G  18.9G  /export/home
rpool/swap                8.16G   280G   216M  -
zmanda                    50.0G  7.08T  38.0K  /zmanda
zmanda/logs                815K  7.08T   815K  /zmanda/logs
zmanda/vtapes             87.9K  7.08T  87.9K  /var/lib/amanda/vtapes
```

Amanda Enterprise Configuration

The Sun Fire X4500 server was configured as an Amanda Enterprise backup server with Zmanda Management Console (ZMC). For more information on Amanda Enterprise, see the “Product Information” section at the end of this article.

To install Amanda Enterprise and ZMC, follow these overall steps, which are explained in more detail in subsequent sections.

1. From the Zmanda Network web site (which requires registration), download the following:

- Amanda Rapid Installer binary for Solaris™ 10 OS systems with Intel® or AMD Opteron processors: `amanda-enterprise-3.0-installer-intel.bin`
- Prerequisite packages provided by Zmanda: `Solaris-10-amanda-dependency-intel.tar.gz`

2. Install the required dependencies. There are two ways to install the required dependencies:

- Use `pkgadd` to install the prerequisite packages included in the Zmanda `Solaris-10-amanda-dependency-intel.tar.gz` archive file.
- Download and install individual packages using the OpenSolaris Image Packaging System (IPS).

3. Extract the binary package and run the Zmanda Rapid Installer to install Amanda Enterprise.

4. After the installation is complete, the ZMC Welcome page is displayed, and you can begin configuring backup sets.

Installing Required Dependencies Using `pkgadd`

The first option is to download the dependency packages provided by Zmanda from the Zmanda Network at http://network.zmanda.com. The required dependency packages are provided in a compressed tar archive file, `Solaris-10-amanda-dependency-intel.tar.gz`.

Extract all the packages:

```
# gtar zxvf Solaris-10-amanda-dependency-intel.tar.gz
```

The dependency packages provided are as follows:

```
CSWbdb4              berkeleydb4 - Embedded database libraries and utilities
CSWbzip2             bzip2 - a high-quality block-sorting file compressor
CSWcommon            common - common files and dirs for CSW packages
CSWcurl              curl - Library and tool for common Internet protocols
CSWcurlrt            curlrt - Library for common Internet protocols Runtime
CSWexpat             expat -  XML Parser Toolkit
CSWfindutils         findutils - A set of utilities for searching a file system
CSWgcc3corert        gcc3corert - GNU C Compiler Run Time
CSWgcc3g++rt         gcc3g++rt - GNU C++ Compiler Run Time
CSWgfile             fileutils - GNU file utilities
CSWggettext          ggettext - GNU gettext
CSWglib2             glib2 - the low-level core library for GTK+ and GNOME
CSWgtar              gtar - GNU tape archiver
CSWiconv             libiconv - GNU iconv library
CSWlibidn            libidn - Implementation of the Stringprep, Punycode and IDNA
CSWlibnet            libnet - the libnet packet construction library
CSWmtx               mtx - SCSI Media Changer and Backup Device Control
CSWoldaprt           openldap_rt - OpenLDAP runtime libraries (oldaprt)
CSWossl              openssl - Openssl meta package
CSWossldevel         openssl_devel - Openssl development support
CSWosslrt            openssl_rt - Openssl runtime libraries
CSWosslutils         openssl_utils - Openssl binaries and related tools
CSWreadline          readline - library to enable interactive line editing
CSWsasl              sasl - Simple Authentication and Security Layer
CSWsunmath           libsunmath - Sun maths library
CSWzlib              zlib - Zlib Data Compression Library
```

These packages are available from Blastwave at http://blastwave.org and also from Sunfreeware at http://sunfreeware.org.

Install each package using `pkgadd`:

```
# pkgadd -d <pkg_name>
```

Installing Required Dependencies Using OpenSolaris IPS

The second option is to use the OpenSolaris Image Packaging System. The most popular Solaris packaging repositories, such as http://blastwave.org and http://sunfreeware.org, have already converted the software they offer to the new IPS package format. You can set the OpenSolaris OS to use such a repository in just one step.

For example, to add the Blastwave repository to the list of authorized package repositories, use this command:

```
# pkg set-authority -O http://blastwave.network.com:10000 blastwave
```

Let's make sure that everything is OK:

```
# pkg authority -H
AUTHORITY                       URL
blastwave                       http://blastwave.network.com:10000/
OpenSolaris.org (preferred)     http://pkg.OpenSolaris.org:80/
```

Now we can start installing the community software (CSW) packages required by Amanda Enterprise.

Note that the names of the Blastwave packages available for OpenSolaris IPS start with `IPS` instead of the traditional `CSW`, for example, `IPScommon` instead of `CSWcommon`.

```
# pkg install IPScommon
DOWNLOAD                                  PKGS       FILES     XFER (MB)
Completed                                  1/1         1/1     0.00/0.00
PHASE                                      ACTIONS
Install Phase                              146/146
PHASE                                        ITEMS
Reading Existing Index                       9/9
Indexing Packages                            1/1
```

To show the ease of use of OpenSolaris IPS, we'll install the complete list of required packages in a single command:

```
# pkg install IPScurl IPSbzip2 IPSbdb4 IPScurlrt IPSexpat IPSfindutils IPSgfile IPSggettext
IPSgtar IPSiconv IPSlibidn IPSlibnet IPSoldaprt IPSossl IPSossldevel IPSosslrt IPSosslutils
IPSreadline IPSsasl IPSsunmath IPSzlib IPSglib2 IPSgcc3g++rt IPSmtx
```

Once finished, let's check that we have everything. (Once installed, the Blastwave packages use their old CSW names.)

```
# pkginfo | grep CSW
application CSWbdb4              berkeleydb4 - Embedded database libraries and utilities
application CSWbzip2             bzip2 - a high-quality block-sorting file compressor
system      CSWcommon            common - common files and dirs for CSW packages
application CSWcurl              curl - Library and tool for common Internet protocols
application CSWcurlrt            curlrt - Library for common Internet protocols Runtime
application CSWexpat             expat -  XML Parser Toolkit
system      CSWfindutils         findutils - A set of utilities for searching a file system

application CSWgcc3corert        gcc3corert - GNU C Compiler Run Time
application CSWgcc3g++rt         gcc3g++rt - GNU C++ Compiler Run Time
system      CSWgfile             fileutils - GNU file utilities
system      CSWggettext          ggettext - GNU gettext
application CSWglib2             glib2 - the low-level core library for GTK+ and GNOME
application CSWgtar              gtar - GNU tape archiver
```

```
system      CSWiconv            libiconv - GNU iconv library
application CSWlibidn           libidn - Implementation of the Stringprep, Punycode and
IDNA
system      CSWlibnet           libnet - the libnet packet construction library
application CSWmtx              mtx - SCSI Media Changer and Backup Device Control
application CSWoldaprt          openldap_rt - OpenLDAP runtime libraries (oldaprt)
application CSWossl             openssl - Openssl meta package
application CSWosssldevel       openssl_devel - Openssl development support
application CSWosslrt           openssl_rt - Openssl runtime libraries
application CSWosslutils        openssl_utils - Openssl binaries and related tools
system      CSWreadline         readline - library to enable interactive line editing
application CSWsasl             sasl - Simple Authentication and Security Layer
system      CSWsunmath          libsunmath - Sun maths library
application CSWzlib             zlib - Zlib Data Compression Library
```

Installing Amanda Enterprise

Installing Zmanda Amanda Enterprise 3.0 is a simple task of running the Zmanda Rapid Installer, which contains the Amanda backup server, the Amanda backup client, as well as the Zmanda Management Console (ZMC) web user interface.

Because we are connected to the Sun Fire X4500 server through an SSH connection, we'll start the Zmanda Rapid Installer in text mode:

```
# ./amanda-enterprise-3.0-installer-intel.bin --mode text
```

Once completed, the following software components are installed:

- Amanda Enterprise 3.0 backup server software
- Amanda Enterprise 3.0 client software
- Zmanda Management Console with MySQL™, PHP, Apache, PHPmyadmin, Perl and various Perl modules and their dependencies (these are installed under the `/opt/zmanda` directory)

We can in the same way install the Amanda backup client for Solaris software:

```
# pkgadd -G -d AmandaEnterpriseBackupClient-3.0-Sun-10-intel.pkg
```

There are a few additional steps that need to be done to complete the installation:

On each of the clients (including the Amanda backup server, if it is also to act as a client), add the following line to the `/var/lib/amanda/.amandahosts` file, where *host* is the host name of the Amanda backup server:

```
<host> amandabackup amdump
```

Then, add the following lines to `/etc/amanda/amanda-client.conf`, where `10.6.240.138` is the IP address of the backup server:

```
index_server "10.6.240.138"     # your amindexd server
tape_server  "10.6.240.138"     # your amidxtaped server
```

We can check that everything is OK by running the following command:

```
# inetadm | grep svc:/network/amanda/tcp:default
enabled      on-line      svc:/network/amanda/tcp:default
```

We can also check that all the relevant services are up and running:

```
# /etc/init.d/zmc status
apache already running
mysql already running
Checking ZMC Services status...
EventServer is running PID : 7290.
DbLogger is running PID : 7291.
Parser is running PID : 7292.
Backup_monitor is running PID : 13077.
Rss2event is not running
```

To start or stop the ZMC application, simply run the following command:

```
# /etc/init.d/zmc start | stop
```

Finally, run the "Verify" tests presented by the ZMC console and make a dry-run backup and restore cycle to make sure everything is working as it should.

Performance Test Setup

System Setup

This section describes the system architecture and configuration implemented for the performance test.

The hardware used during this performance test was:

- Sun Fire X4500 server, 2-socket dual-core AMD Opteron 285 CPUs running at 2593 MHz with 16 Gbyte of RAM and 48 SATA drives of 512 Gbyte each, acting as the backup server
- Sun Fire X4200 server, 2-socket dual-core AMD Opteron 2000 CPU with 16 Gbyte of RAM, acting as the backup client

The two boot drives of the backup server were mirrored using ZFS volume management capabilities and the remaining 46 drives were available for backup data.

As a general rule, no specific OS tuning was done on the systems, nor were the binaries of Amanda Enterprise and the various command-line tools optimized in any way. The objective of the tests was to deploy the out-of-the-box solution.

The only OS tuning performed was to limit the amount of memory available to the ZFS Adaptive Replacement Cache (ARC) to 4 Gbyte by adding the following line to `/etc/system`:

```
set zfs:zfs_arc_max = 4294967296
```

Amanda Enterprise Software Release

The performance tests were carried out using an early-access version of Amanda Enterprise 3.0 for the Solaris 10 OS for x64 platforms.

The following software components were installed:

- The Amanda Enterprise 3.0 software
- Zmanda Management Console with MySQL, PHP, Apache, PHPmyadmin, Perl and various Perl modules and their dependencies; these are installed under the `/opt/zmanda/amanda` and `/opt/zmanda/common` directories

OpenSolaris Operating System Release

The following version of the OpenSolaris OS was used:

```
                  OpenSolaris 2008.11 snv_101b_rc2 X86
        Copyright 2008 Sun Microsystems, Inc.  All Rights Reserved.
                    Use is subject to license terms.
                       Assembled 19 November 2008
```

Zpool Configuration

As mentioned previously, the storage available on the Sun Fire X4500 server was split into two pools:

- A backup pool made of 24 drives and two spares in a RAIDZ2 configuration. This provided 11 Tbytes of capacity for backups with a very high level of protection against drive failure and performance sufficient to handle 300 to 350 Mbyte/sec of backup traffic over the network.

- A data pool made of 12 drives and two spares in a RAIDZ2 configuration, which was used to store the data files used during the local tests.

The Sun Fire X4200 system was used as the backup client and contained a data pool made of a single 73-Gbyte drive, which was used to hold the data to be backed up during the network tests.

ZFS Configuration

Four ZFS file systems were created on the backup server:

- `zmanda/vtapes` to hold the backup sets
- `pool2/10gb` to hold the data files for the 10-Gbyte tests
- `pool2/50gb` to hold the data files for the 50-Gbyte tests
- `pool2/1tb` to hold the data files for the 1-Tbyte tests

It's easy to make sure the data sets are of the right size by using ZFS `quota`:

```
# zfs create -o quota=10gb pool2/10gb
# zfs create -o quota=50gb pool2/50gb
# zfs create -o quota=1tb pool2/1tb

# zfs list
NAME                      USED  AVAIL  REFER  MOUNTPOINT
pool2                    2.21T  1.35T  2.02T  /pool2
pool2/1tb                36.0K  1024G  36.0K  /pool2/1tb
pool2/50gb               50.0G  37.1M  50.0G  /pool2/50gb
pool2/10gb               10.0G  1.39M  10.0G  /pool2/10gb
```

Similarly, we created a 50-Gbyte ZFS file system on the client server.

In order to optimize the disk throughput, we made sure the ZFS record size was set to 128 Kbyte:

```
# zfs set recordsize=128k zmanda/vtapes
```

Note that this is the default value for ZFS file systems.

Then, let's make sure that the default Amanda user, `amandabackup`, has all the necessary credentials:

```
# pfexec zfs allow amandabackup snapshot,rollback,mount,destroy pool2
# pfexec zfs allow amandabackup snapshot,rollback,mount,destroy pool2/1tb
```

```
# pfexec zfs allow amandabackup snapshot,rollback,mount,destroy pool2/50gb
# pfexec zfs allow amandabackup snapshot,rollback,mount,destroy pool2/10gb
# pfexec zfs allow amandabackup snapshot,rollback,mount,destroy zmanda/vtapes
```

Finally, add the ZFS management role to the `amandabackup` user profile:

```
# usermod -P "ZFS File System Management,ZFS Storage Management" amandabackup
```

Note: This needs to be done on both the backup server and the client server.

Generating the Data Files

A simple `dd` command helped to generate a 1-Gbyte data file, which was then copied 10, 50, and 1000 times to create the expected data-set sizes.

```
# dd if=/dev/rdsk/c0t0d0s0 of=bin.data bs=131072 count=8192
```

This provided three data sets of various sizes: 10 Gbyte, 50 Gbyte, and 1 Tbyte.

Because of a lack of sufficient storage capacity on the client server, the network tests were performed only on 10 Gbyte and 50 Gbyte.

Note: Using the `mkfile(1M)` command is not appropriate to generate the data files because Amanda Enterprise uses archivers that support sparse files, meaning that a file created with `mkfile` will be backed up using only a few bytes.

Testing Methodology

The performance of Amanda Enterprise 3.0 was tested by conducting backups of several sizes of data sets, both local to the backup server and located on a client.

Each backup was performed as a level-0 backup with no compression and no indexing.

Backups were performed directly to disk, as virtual tapes, and Amanda Enterprise was configured to use a “holding disk,” which is disk storage used as a buffer pool before writing the backup to the vtapes.

Each backup set (10 Gbyte, 50 Gbyte, and 1 Tbyte) was processed using five different archivers:

- The GNU tar utility, called `gtar`
- The tar utility called `star` from Joerg Schilling (Schily); the `star` project page is `http://developer.berlios.de/projects/star` and the binaries can be downloaded from from numerous places on the Internet, such as `http://sunfreeware.com` or `http://www.blastwave.org`
- The default Sun tar utility, called `tar`
- The ZFS snapshot method
- The ZFS send method

What was measured was the overall time taken by Amanda Enterprise to complete the backup, as reported by the `amreport` command.

Each test was carried out three times in a row (except for the tests with the 1-Tbyte data set) and both the slowest and fastest results were discarded.

The following actions were taken before each measured run:

- Delete all previous backups from `/var/lib/amanda/vtapes`.
- Delete all previous ZFS snapshots created by Amanda Enterprise.
- Unmount and mount the ZFS file systems, in order to remove any caching effect from ZFS from consecutive runs.

Although the Zmanda Management Console is very useful for day-to-day operations, we used the command line to prepare, start the backup jobs, and report the results. This enabled us to prepare whole series of backup scenarios as batch jobs.

The following command starts a backup for backup-set `10gb`:

```
$ amdump 10gb
```

The following command generates a report where `log.last` is the corresponding log file:

```
$ amreport 10gb -i -l log.last
```

From the dump summary produced, the `TAPER STATS` throughput is the one of interest.

```
DUMP SUMMARY:
                                   DUMPER STATS                TAPER STATS
HOSTNAME     DISK        L ORIG-MB  OUT-MB  COMP%  MMM:SS   KB/s MMM:SS   KB/s
-------------------------- ------------------------------------- -------------
10.6.240.138 -l2/10g-bin 0   10227   10227    --     3:15 53715.1   0:50 208859.7
```

Amanda Enterprise Configuration

A lot of time was spent in testing and tuning the various parameters available to configure Amanda Enterprise. In the end, the following parameters were used:

- `netusage 524288 kbps` (for local tests) / `122880 kbps` (for network tests)
- `device_output_buffer_size 128000k`
- `blocksize 128KB` (for `gtar`, `suntar`) / blocksize of 10 Kbyte for `star` with holding disk
- `inparallel 10`
- `maxdumps 4`
- `autoflush yes`
- `dtimeout 1800`
- `ctimeout 30`
- `etimeout 300`

The virtual tapes were created as a chunk of 1 Tbyte and their block size was set to match the ZFS record size:

```
#define tapetype HARDDISK {
       length 1048576 mb
       blocksize 128 kb
}
```

The holding disk was created on the `pool2` ZFS pool to ensure the best possible performance when flushing the holding disk to the virtual tapes.

```
holdingdisk hdisk {
        comment "Holding Disk"
        directory "/pool2/holding"
         chunksize 20 gb
}
```

Performance Test Results

1. Baseline Throughput

Before starting to measure the performance of the backup server itself, let's review the raw throughput sustainable on the Sun Fire X4500 server with the current ZFS and disk configuration.

The following results were gathered by copying a 100-Gbyte data file from the `pool2/1tb` file system to the `zmanda/vtapes` file system using various tools and parameters.

Table 1: Baseline Results

Utility	Block Size (KB)	Sparse File Support	Throughput (MB/s)
`dd`	128	n/a	273
`cp`	n/a	n/a	162
`gtar`	10 (default)	no	189
`gtar`	128	no	215
`gtar`	128	yes	113
`star`	128	no	234
`star`	128	yes	226
`sun-tar`	128	no	221
`pax*`	128	yes	231
* Not currently supported by Amanda Enterprise			

Clearly, Table 1 shows that `gtar` is at a disadvantage when dealing with sparse file support, which is the mode used by Amanda Enterprise. The reason for this was not investigated in detail, but a first analysis showed that `gtar` was reading the input file twice, thereby halving the overall throughput.

Table 1 also shows that setting the block size to match the ZFS record size yields the best overall throughput. Unfortunately, Amanda Enterprise doesn't allow you to specify block-size parameters to the archive utility when using `gtar` and `star`.

So, to work around this limitation, we created a simple interposing script to each archiver. As an example, a script called `gtar`—calling the real `gtar.bin` binary and placed in the default `amandabackup` user path—would allow us to change the default options passed by Amanda Enterprise to `gtar`.

```
#!/bin/sh

        blocksize=256   # 128KB
        cmd=/opt/csw/bin/gtar.bin

        $cmd $@ --blocking-factor=$blocksize
```

2. Local Backups

Local backup tests were performed by running backups of various data-set sizes located on the `pool2` ZFS pool on the Amanda backup server itself. All backups were configured to use a holding disk located on the same pool as the data set, but separate from the virtual tapes.

The results in Table 2 and Figure 2 show the throughput obtained using several archivers.

Table 2: Overall Throughput for Local Backups in Mbyte/sec

Archiver	10-GB Data Set	50-GB Data Set	1-TB Data Set
`star`	198.6	196.5	n/a
`gtar`	218.6	194.6	97.4
`tar`	218.0	194.2	n/a
`zfs-sendrecv`	214.4	196.9	n/a
`zfs-snapshot`	224.9	199.9	96.2

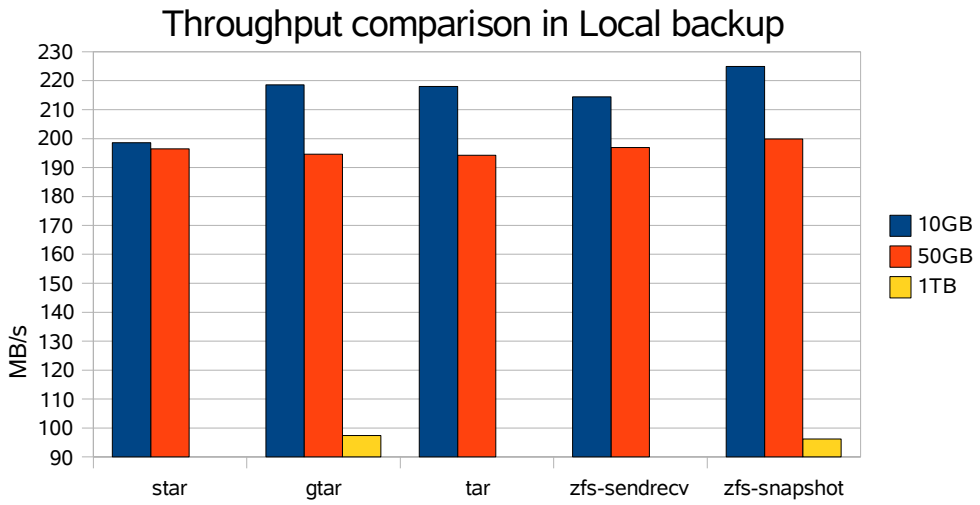


Figure 2: Throughput Comparisons for Local Backup

The results from `star` are slightly lower than for the other archivers due to the fact that the block size was left at the default of 10 Kbyte, whereas all other archivers had a block size set at 128 Kbyte.

However, block size and differences in algorithms become negligible when the size of the data set increases to 50 Gbyte.

3. Network Backups

Network backups tests were performed by running backups of various data-set sizes located on a separate client machine, linked to the Amanda backup server by a 100-Gbyte Ethernet link. All backups were configured to use a holding disk located on a separate ZFS pool on the backup server rather than on the same ZFS pool used by the virtual tapes.

The results in Table 3 and Figure 3 show the throughput obtained using several archivers.

Table 3: Overall Throughput for Network Backups in Mbyte/sec

Archiver	10-GB Data Set	50-GB Data Set
`star`	198.6	196.5
`gtar`	218.6	194.6
`tar`	218.0	194.2
`zfs-sendrecv`	214.4	196.9
`zfs-snapshot`	224.9	199.9

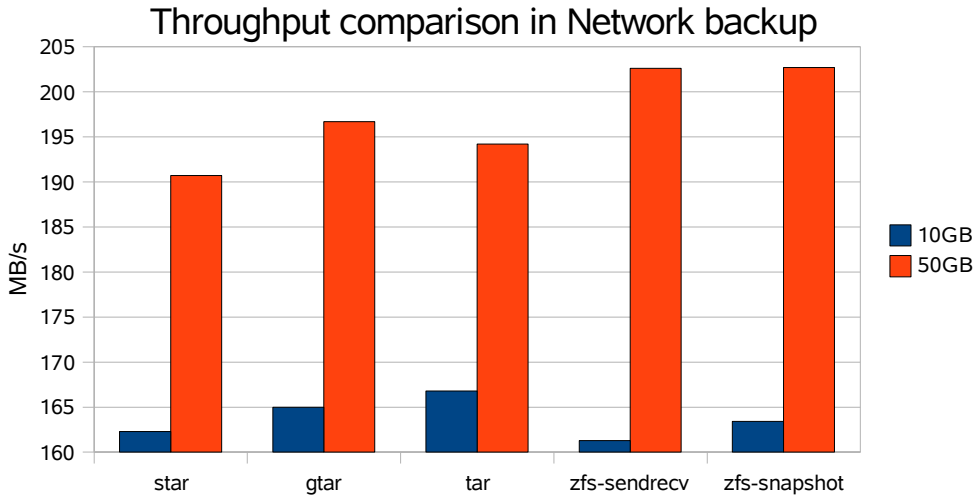


Figure 3: Throughput Comparisons for Network Backup

Product Information

OpenSolaris Operating System

OpenSolaris 2008.11 OS supports Dynamic Tracing (DTrace), ZFS, cryptographic infrastructures, IP filter and User and Process Rights Management, and Predictive Self-Healing, which can further enhance system reliability.

ZFS File System

ZFS is a 128-bit general-purpose file system that spans the desktop to the data center, where space is shared dynamically between multiple file systems from a single storage pool. Physical storage can be added to or removed from storage pools dynamically, without interrupting services. This capability provides new levels of flexibility, availability, and performance.

One of the most interesting features of ZFS when combined with backup technologies is its snapshot capabilities. A snapshot is a read-only copy of a file system or volume that can be created very quickly. Thanks to ZFS copy-on-write characteristics, snapshots are essentially a free operation, which initially consume no additional space within the pool. However, as data within the active data set changes, the snapshot consumes space by continuing to reference the old data and so prevents the data from being freed back to the pool.

ZFS snapshots provide the following advantages:

- ZFS snapshots persist across system reboots.
- ZFS snapshots use no separate backing store. Snapshots consume disk space directly from the same storage pool as the file system from which they were created.
- Recursive snapshots are created quickly as one atomic operation. The snapshots are created together (all at once) or not created at all. The benefit of atomic snapshots operations is that the snapshot data is always taken at one consistent time, even across descendent file systems.

Sun Fire X4500 Server

The Sun Fire X4500 server can deliver the performance of a four-way x64 server and provides 48 SATA drives with a total capacity of 48 Tbyte in 4U of rack space. Its bigger brother, the Sun Fire X4540 server, is powered by two quad-core AMD Opteron CPUs operating at 2300 MHz and provides the same amount of storage capacity. (The Sun Fire X4500 server has reached end-of-life and is no longer orderable. It is superceded by the next-generation Sun Fire X4540 Server.)

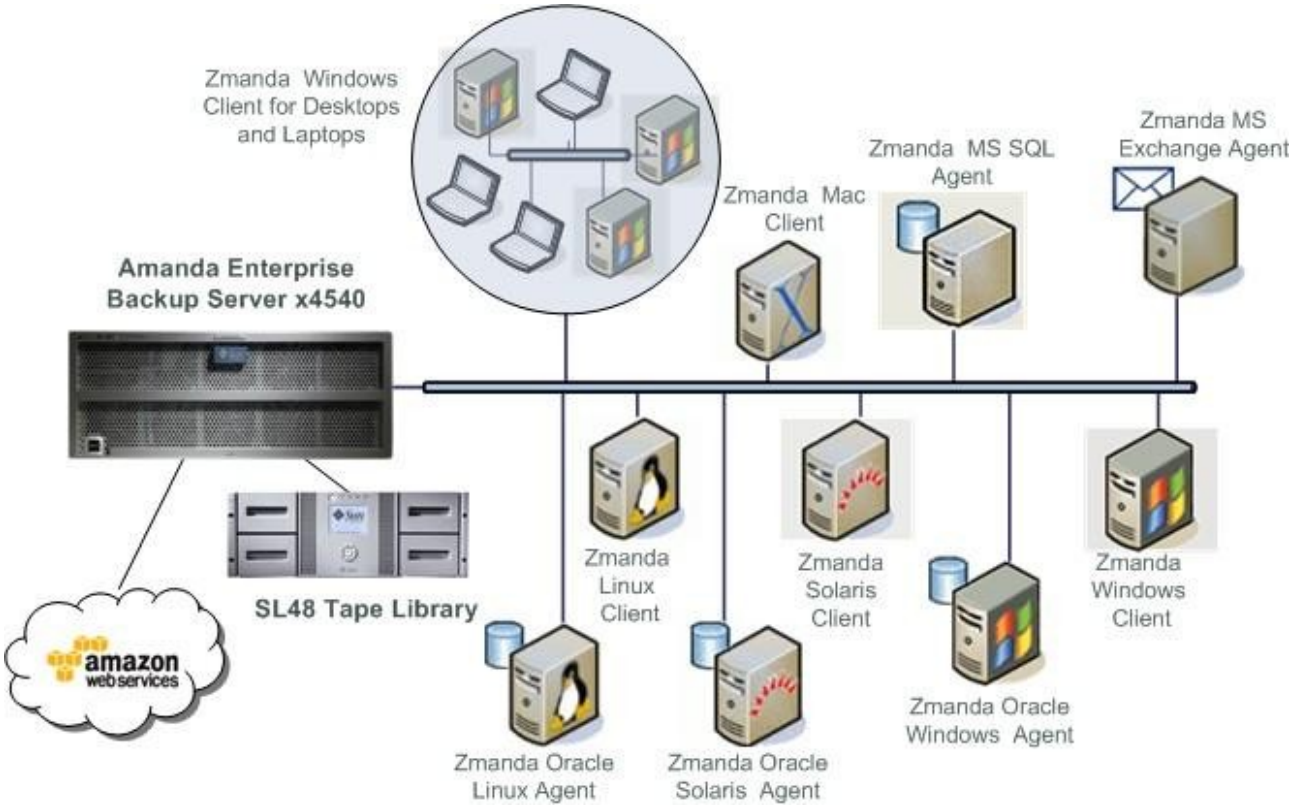
Zmanda Amanda Enterprise 3.0

Explosive data growth, combined with demanding requirements for data availability, has placed a tremendous burden on IT operations staff at businesses of all sizes. Yet, many organizations do not have the staff or budget to purchase and manage complex and expensive backup and recovery software products.

Amanda Enterprise was designed to address these challenges—providing a backup and recovery solution that combines fast installation, simplified management, enterprise-class functionality, and low-cost subscription fees. As an open source product, Amanda Enterprise uses only standard formats and tools—effectively freeing you from being locked into a vendor to recover your archived data.

Centralized Management for Heterogeneous Environments

Back up and recover a few, dozens, or even hundreds of Linux, Solaris, Microsoft Windows, and Mac computers, as well as your databases and applications, using one web-based management console. Data can be stored on tape, disk, optical devices, or online storage.



Notes: SL48 here refers to Sun StorageTek™ SL48 Tape Library. X4540 refers to Sun Fire X4540 Server. MS refers to Microsoft. Mac refers to Macintosh systems from Apple. All figures reprinted with permission from Zmanda.

Figure 4: Centralized Management for Heterogeneous Environments

Summary

This guide was a quick introduction to configuring the Sun Fire X4500 server as a backup server for Zmanda Amanda Enterprise 3.0. We have seen how to deploy Amanda Enterprise 3.0 from Zmanda on a system running OpenSolaris 2008.11 and successively learned how to:

- Create a ZFS storage pool using RAIDZ2
- Create a ZFS file system
- Install packages using the new OpenSolaris Image Packaging System
- Deploy Amanda Enterprise in this environment

We then demonstrated the benefits of using Amanda's ZFS snapshot integration in improving the performance of an enterprise backup solution.

Configuration Files

Example of Amanda Configuration File Used for the 10-GB Backup Set

amanda.conf:

```
org "10gb"                                    # your organization name for reports

infofile "/etc/amanda/10gb/curinfo"           # database DIRECTORY
logdir   "/etc/amanda/10gb"                   # log directory
indexdir "/etc/amanda/10gb/index"             # index directory
tapedev "file://var/lib/amanda/vtapes/10gb"   # the no-rewind tape device to be used

debug_holding 1
debug_dumper 1
debug_chunker 1
debug_planner 1
debug_taper 1

inparallel 10             # maximum dumpers that will run in parallel (max 63)
maxdumps 4

#netusage  524288 Kbps    # maximum net bandwidth for Amanda, in KB per sec
netusage  122880 Kbps     # maximum net bandwidth for Amanda, in KB per sec

device_output_buffer_size 128000 k      # amount of buffer space to use when
                                        # writing to devices
                                        # Default: 1280k. Controls the amount of
                                        # memory used by Amanda to hold data as it
                                        # is read from the network or disk before
                                        # it is written to the output device.
                                        # Higher values may be useful on fast tape
                                        # drives and optical media.

##################### Default settings from the installer ##########################

dumpuser "amandabackup"          # the user to run dumps under
mailto "amandabackup@127.0.0.1"  # space separated list of operators at your site

displayunit "m"                  # Possible values: "k|m|g|t"
                                 # Default: k.
                                 # The unit used to print many numbers.
                                 # k=kilo, m=mega, g=giga, t=tera
runspercycle 1                   # the number of amdump runs in dumpcycle days
                                # (1 week * 5 amdump runs per week -- just weekdays)

tapecycle 7                      # the number of tapes in rotation
                                 # 1 week (dumpcycle) times 5 tapes per week (just
                                 # the weekdays) plus a few to handle errors that
                                 # need amflush and so we do not overwrite the full
                                  # backups performed at the beginning of the
                                 # previous cycle

runtapes 1                # number of tapes to be used in a single run of amdump

changerfile "/etc/amanda/changer.conf"
tpchanger "chg-disk"                    # the tape-changer glue script
changerdev "/dev/null"

tapetype HARDDISK                       # what kind of tape it is

dumporder "sssS"                        # specify the priority order of each dumper
                                        #   s -> smallest size
                                        #   S -> biggest size
                                        #   t -> smallest time
                                        #   T -> biggest time
```

```
                                        #   b -> smallest bandwidth
                                        #   B -> biggest bandwidth
                                        # try "BTBTBTBTBTBT" if you are not holding
                                        # disk constrained

taperalgo first           # The algorithm used to choose which dump image to send
                          # to the taper.

bumpsize 20 Mb            # minimum savings (threshold) to bump level 1 -> 2
bumppercent 20            # minimum savings (threshold) to bump level 1 -> 2
bumpdays 1                # minimum days at each level

usetimestamps yes

maxdumpsize -1            # Maximum number of bytes the planner will schedule
                          # for a run (default: runtapes * tape_length).

amrecover_do_fsf yes      # amrecover will call amrestore with the
                          # -f flag for faster positioning of the tape.

amrecover_check_label yes # amrecover will call amrestore with the
                          # -l flag to check the label.

bumpmult 4                # threshold = bumpsize * bumpmult^(level-1)


amrecover_changer "changer"      # amrecover will use the changer if you restore
                                 # from this device. It could be a string like
                                 # 'changer' and amrecover will use your changer
                                 # if you set your tape to 'changer' with
                                 # 'setdevice changer' or via 'tapedev "changer"'
                                 # in amanda-client.conf

autoflush yes                    # if autoflush is set to yes, then amdump will
                                 # schedule all dump on holding disks to be flush
                                 # to tape during the run.


###################### Common Settings adapted to the tests ########################

labelstr "^amanda-tests-[0-9][0-9]*$"   # label constraint regex: all tapes must
                                        # match label_new_tapes "amanda-tests-%%%"

reserve 30                       # percent
                                 # This means save at least 30% of the
                                 # holding disk space for degraded mode backups.

dtimeout 1800                    # number of idle seconds before a dump is aborted.
ctimeout 30                      # maximum number of seconds that amcheck waits
etimeout 300                     # number of seconds per file system for estimates.

dumpcycle 0               # the number of days in the normal dump cycle


define tapetype HARDDISK {
       length 1048576 mb
       blocksize 128 kb    #32kb is default. Same as BLOCK_SIZE device property
}

# network interfaces
define interface local {
    comment "a local disk"
    use 524288 kbps
}

holdingdisk hdisk {
       comment "Holding Disk"
       directory "/pool2/holding"
```

```
       chunksize 20 Gb
}

define application-tool app_amgtar {
  comment "amgtar"
  plugin "amgtar"
  property "ATIME-PRESERVE" "NO"
}

define application-tool app_amsuntar {
  comment "amsuntar"
  plugin "amsuntar"
  property "BLOCK-SIZE" "128"
#  property "EXTENDED-ATTRIBUTES" "NO"
#  property "EXTENDED-HEADERS" "NO"
}

define application-tool app_zfs_sendrecv {
  comment "amzfs-sendrecv"
  plugin "amzfs-sendrecv"
  property "DF-PATH"  "/usr/sbin/df"
  property "ZFS-PATH" "/usr/sbin/zfs"
  property "PFEXEC-PATH" "/usr/bin/pfexec"
  property "PFEXEC" "YES"
}

define script-tool script_amzfs_snapshot {
  comment "amzfs-snapshot"
  plugin "amzfs-snapshot"
  execute-where client
  execute-on pre-dle-amcheck, post-dle-amcheck, pre-dle-estimate, post-dle-estimate, pre-
dle-backup, post-dle-backup
  property "DF-PATH"  "/usr/sbin/df"
  property "ZFS-PATH" "/usr/sbin/zfs"
}


define dumptype global {
        comment "Global definitions"
        auth "bsdtcp"
       strategy noinc
       compress none
        index no
       encrypt none
       estimate server
}

define dumptype user-gtar {
       global
       program "APPLICATION"
        application "app_amgtar"
        comment "dumptype using gtar"
       index no
       holdingdisk never
}

define dumptype hdisk-gtar {
       global
       program "APPLICATION"
        application "app_amgtar"
        comment "dumptype using gtar and holding disk"
       index no
       holdingdisk required
}

define dumptype user-star {
       global
       program "STAR"
```

```
       comment "dumptype using star"
       index  no
       holdingdisk never
}

define dumptype hdisk-star {
       global
       program "STAR"
       comment "dumptype using star and holding disk"
       index no
       holdingdisk required
}

define dumptype user-suntar {
       global
       program "APPLICATION"
       application "app_amsuntar"
       comment "dumptype using suntar"
       index no
       holdingdisk never
}

define dumptype hdisk-suntar {
       global
       program "APPLICATION"
       application "app_amsuntar"
       comment "dumptype using suntar and holding disk"
       index no
       holdingdisk required
}

define dumptype zfs-sendrecv {
       global
       program "APPLICATION"
       application "app_zfs_sendrecv"
       comment "dumptype using zfs send/receive"
       index no
       holdingdisk never
}

define dumptype hdisk-zfs-sendrecv {
       global
       program "APPLICATION"
       application "app_zfs_sendrecv"
       comment "dumptype using zfs send/receive and holding disk"
       index no
       holdingdisk required
}

define dumptype zfs-snapshot {
       global
       program "APPLICATION"
       application "app_amgtar"
       comment "dumptype using zfs snapshots"
       script "script_amzfs_snapshot"
       index no
       holdingdisk never
}

define dumptype hdisk-zfs-snapshot {
       global
       program "APPLICATION"
       application "app_amgtar"
       comment "dumptype using zfs snapshots and holding disk"
       script "script_amzfs_snapshot"
       index no
       holdingdisk required
}
```

Example of Disklist Entry (DLE) File Defined for a Network Backup Using `zfs-sendrecv`

```
disklist:
10.6.240.138 “/pool2/10gb” hdisk-zfs-sendrecv
```

About the Author

Caryl Takvorian is currently a Staff Engineer at Sun Microsystems. He joined Sun in 1998 in what is now known as ISV Engineering where he assists independent software vendors (ISVs) in making the best use of Sun technologies. His technical skills are mainly in the Solaris OS and multithreaded applications and development tools. Caryl also leads the Telecommunications segment for ISV Engineering.

For More Information

Here are additional resources.

Zmanda Resources

- Zmanda Amanda Enterprise web site: http://amanda.zmanda.com/
- Amanda wiki: http://wiki.zmanda.com/index.php/Main_Page
- Zmanda forums: http://forums.zmanda.com/
- Resources on Zmanda Network (requires registration):
    - Download for Amanda Enterprise: http://network.zmanda.com/
    - Documentation for Amanda Enterprise: http://network.zmanda.com/documentationEnterprise.php

Sun Fire Server Resources

- BigAdmin article: *Sun Fire X4540 Server as Backup Server for Zmanda's Amanda Enterprise 2.6 Software*: http://www.sun.com/bigadmin/features/articles/zmanda_sfx4540.jsp
- Sun Fire X4500 Server document collection on docs.sun.com: http://docs.sun.com/app/docs/prod/sf.x4500#hic

OpenSolaris Resources

- *OpenSolaris 2009.06 Image Packing System Guide* (PDF): http://opensolaris.org/os/project/pkg/files/ips.pdf
- OpenSolaris web site: http://opensolaris.org

ZFS Documentation

- *Solaris ZFS Administration Guide* (part of the Solaris 10 System Administration Collection): http://docs.sun.com/app/docs/doc/819-5461
- *ZFS Best Practices Guide* on the Solaris Internals web site: http://www.solarisinternals.com/wiki/index.php/ZFS_Best_Practices_Guide

Other Sun Resources

- Sun documents at http://docs.sun.com
- Sun training courses at http://www.sun.com/training/
- Discussions:
  - Sun Storage forums: http://forum.java.sun.com/category.jspa?categoryID=66
  - Sun Hardware - Servers forums: http://forums.sun.com/forum.jspa?forumID=830
- Wikis:
  - Sun BluePrints™ wiki (http://wikis.sun.com/display/BluePrints/Main), especially the Storage BluePrints page: http://wikis.sun.com/display/BluePrints/Storage+BluePrints
  - Sun Storage Administration wiki: http://wikis.sun.com/display/StorageAdmin/Home
  - BigAdmin Storage Tech Tips wiki: http://wikis.sun.com/display/BigAdmin/Storage+Tech+Tips
- Other storage resources:
  - BigAdmin Storage Resource Collection (includes community content): http://www.sun.com/bigadmin/collections/storage.html
  - Sun Storage web site: http://www.sun.com/storagetek/index.jsp
  - Storage Stop Blog: http://blogs.sun.com/storage
- Support:
  - Register your Sun gear: https://inventory.sun.com/inventory/
  - Services: http://www.sun.com/service/index.jsp
  - SunSolve℠ Online: http://sunsolve.sun.com
- Events of interest to users of Sun products:
  - Worldwide developer events and Sun Tech Days: http://developers.sun.com/events/
  - Current events: http://www.sun.com/events/index.jsp

Licensing Information

Unless otherwise specified, the use of this software is authorized pursuant to the terms of the license found at http://www.sun.com/bigadmin/common/berkeley_license.html.