ZFS 101 (aka ZFS is Cool and Why You Should be Using It

Dru Lavigne
Documentation Lead, iXsystems
SCALE, February 23, 2014

Outline

Discuss ZFS features and describe the available management utilities for the following FreeBSD-based operating systems:

- FreeNAS 9.2.1: open source NAS (Network Attached Storage)

- PC-BSD 10.0: open source desktop (GUI) or server (CLI)

Latest versions of these operating systems are on par with the latest OpenZFS “feature flags”

History of ZFS

Modern filesystem specifically designed to add features not available in traditional filesystems

Originally developed at Sun with the intent to open source

After the Oracle acquisition, open source development continued and the original engineers founded OpenZFS (open-zfs.org) which is under active development

OpenZFS uses feature flags instead of versions

What is ZFS?

128-bit COW (Copy on Write) filesystem and logical volume manager with a maximum pool/file size of 16 exabytes

In a traditional Unix filesystem, you need to define the partition size and mount point at filesystem creation time

In ZFS, you instead feed disks to a “pool” and create filesystems from the pool as needed

Pool

Root (parent) volume which can be logically sub-divided as needed

The number of disks added at a time is known as a “vdev”

To optimize performance and resilvering time, number of disks per vdev is limited

As more capacity is needed, add identical vdevs--these will be striped into the pool

RAIDZ

RAIDZ* levels designed to overcome hardware RAID limitations such as the write-hole and corrupt data written over time before the controller provides an alert

Designed for commodity disks so no RAID controller is needed

Can also be used with a RAID controller, but it typically should be put into JBOD mode

RAIDZ1

Parity blocks are distributed across all disks

Up to one disk can fail per vdev without losing pool

Pool can be lost if second disk in a vdev fails before resilver completes

Optimized for vdev of 3, 5, or 9 disks

RAIDZ2

Double-parity solution similar to RAID6

Parity blocks are distributed across all disks

Up to two disks can fail per vdev without losing pool, with no restrictions on which disks can fail

Optimized for vdev of 4, 6, or 10 disks

RAIDZ3

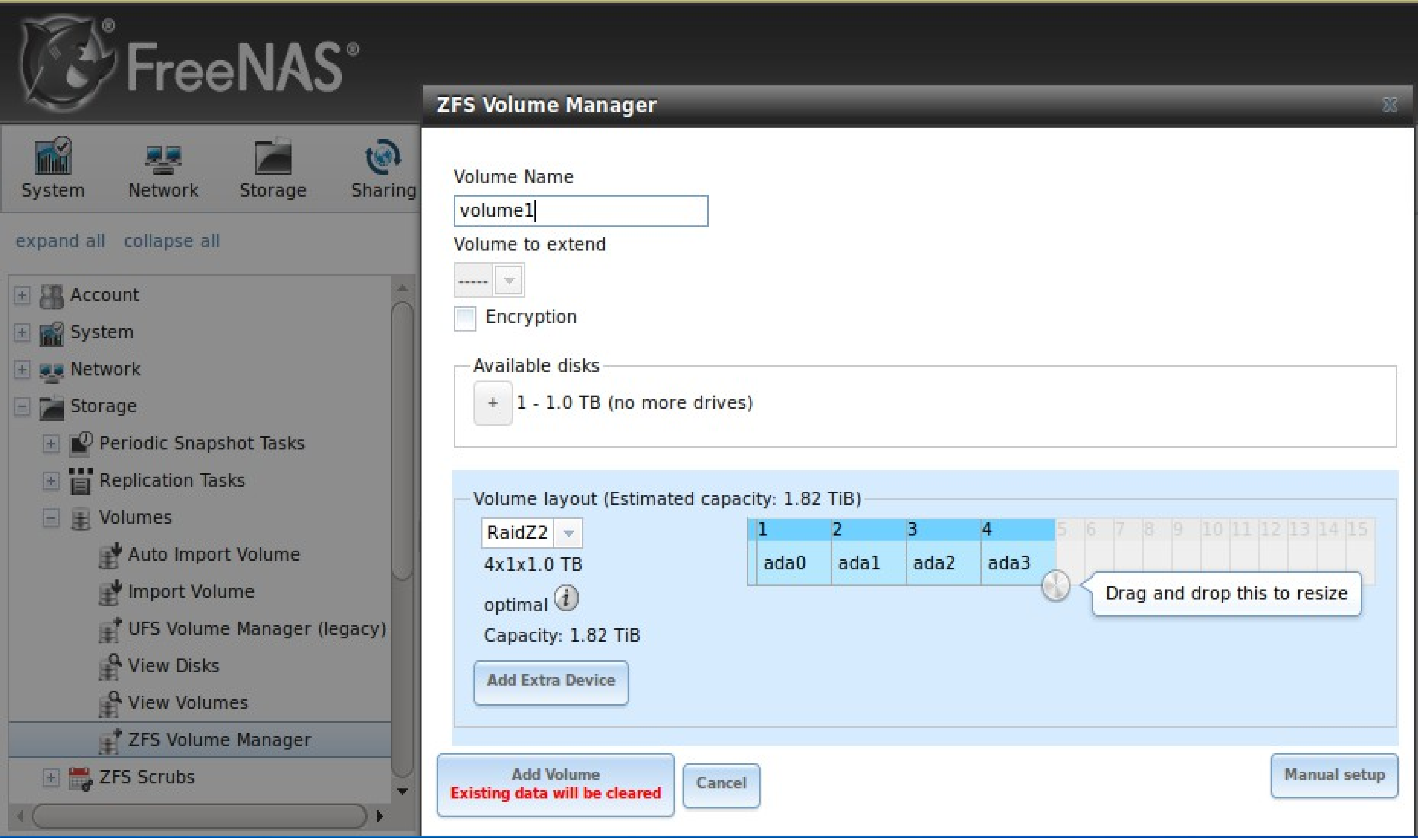
Triple-parity solution

Parity blocks are distributed across all disks

Up to three disks can fail per vdev without losing pool, with no restrictions on which disks can fail

Optimized for vdev of 5, 7, or 11 disks

Create Pool on FreeNAS



ZFS Volume Manager

Volume Name: volume1

Volume to extend: -----

□ Encryption

Available disks

+ 1 - 1.0 TB (no more drives)

Volume layout (Estimated capacity: 1.82 TiB)

RaidZ2

4x1x1.0 TB

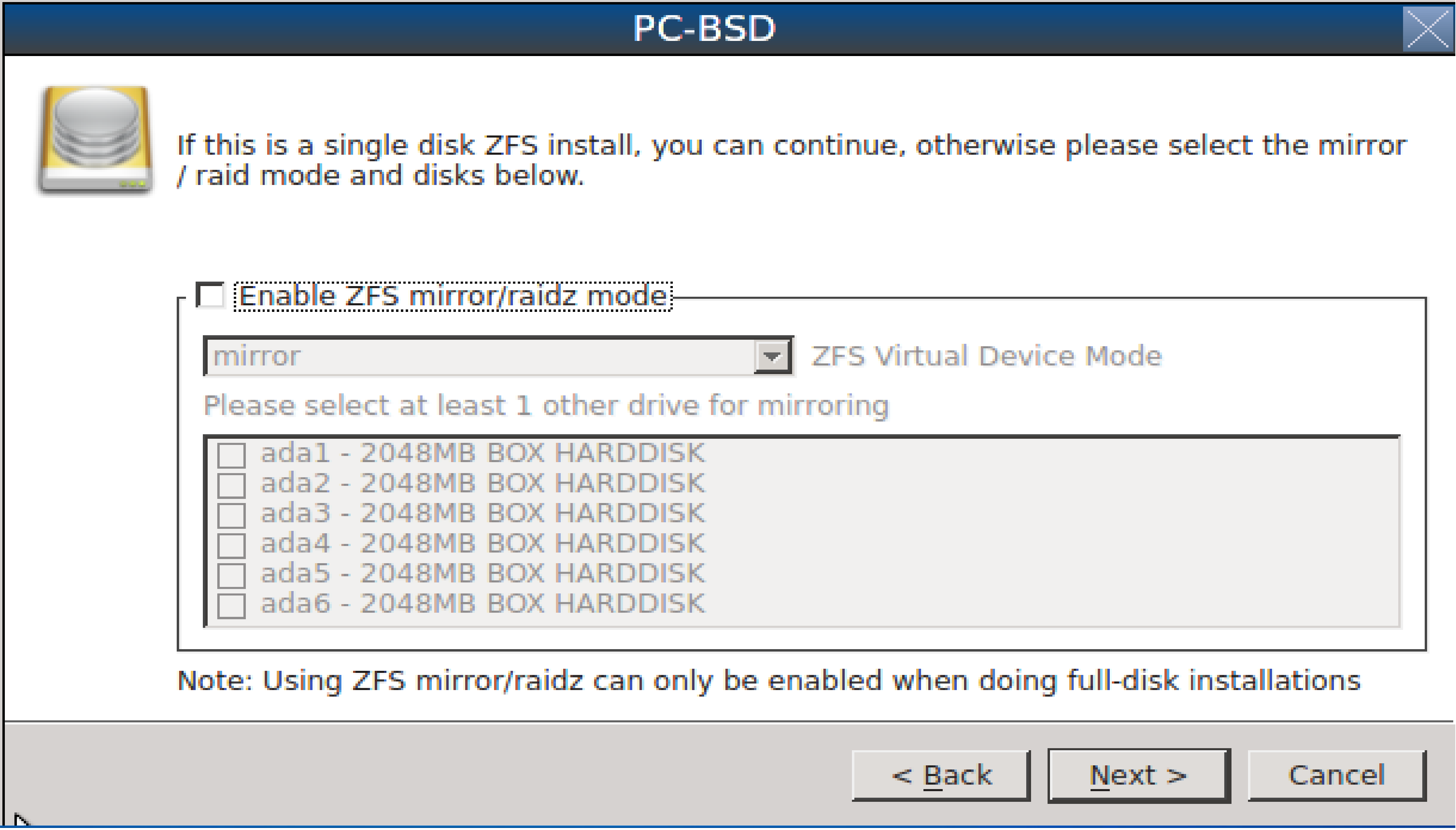
optimal

Capacity: 1.82 TiB

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
ada0	ada1	ada2	ada3											

Drag and drop this to resize

Create Pool on PC-BSD



If this is a single disk ZFS install, you can continue, otherwise please select the mirror / raid mode and disks below.

□ Enable ZFS mirror/raidz mode

mirror ▾ ZFS Virtual Device Mode

Please select at least 1 other drive for mirroring

□ ada1 - 2048MB BOX HARDDISK
□ ada2 - 2048MB BOX HARDDISK
□ ada3 - 2048MB BOX HARDDISK
□ ada4 - 2048MB BOX HARDDISK
□ ada5 - 2048MB BOX HARDDISK
□ ada6 - 2048MB BOX HARDDISK

Note: Using ZFS mirror/raidz can only be enabled when doing full-disk installations

ZIL

ZFS Intent Log

Effectively a filesystem journal that stores sync writes until they are committed to the pool

A dedicated SSD as a secondary log device (SLOG) can increase synchronous write performance, will have no effect on asynchronous writes

FreeNAS includes the zilstat CLI utility to help determine if system would benefit from a SLOG

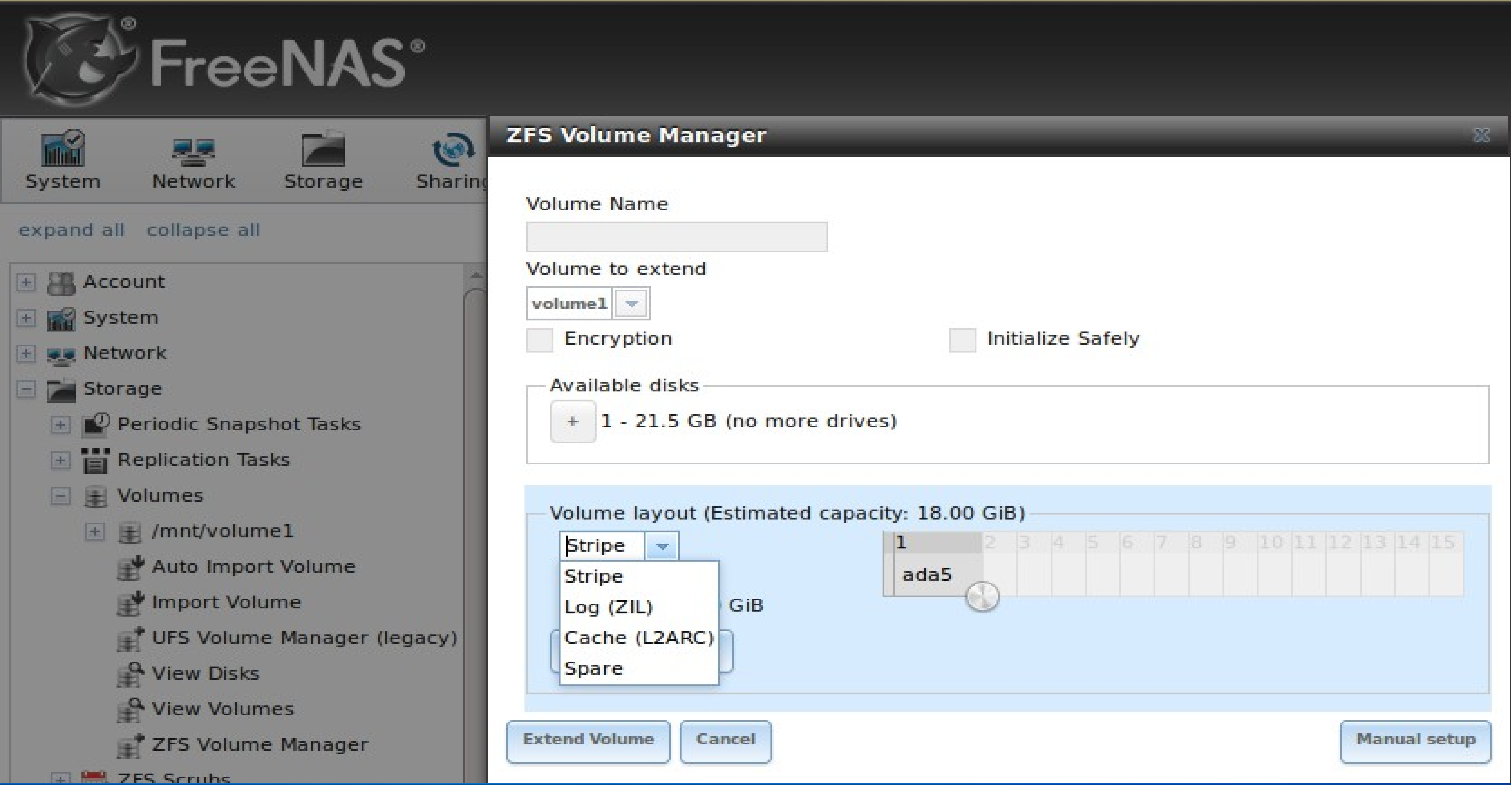
ARC and L2ARC

ARC refers to read cache in RAM. Takes time for ARC to populate with hits; if high misses continue for cached reads, the system needs to be tuned.

Freenas adds ARC stats to top(1) and includes arc_summary.py and arcstat.py tools for ARC monitoring

Optional, secondary ARC can be installed on SSD or disk in order to increase random read performance. Always add as much RAM as possible first.

Adding SLOG/L2ARC on FreeNAS



ZFS Volume Manager

Volume Name:

Volume to extend: volume1

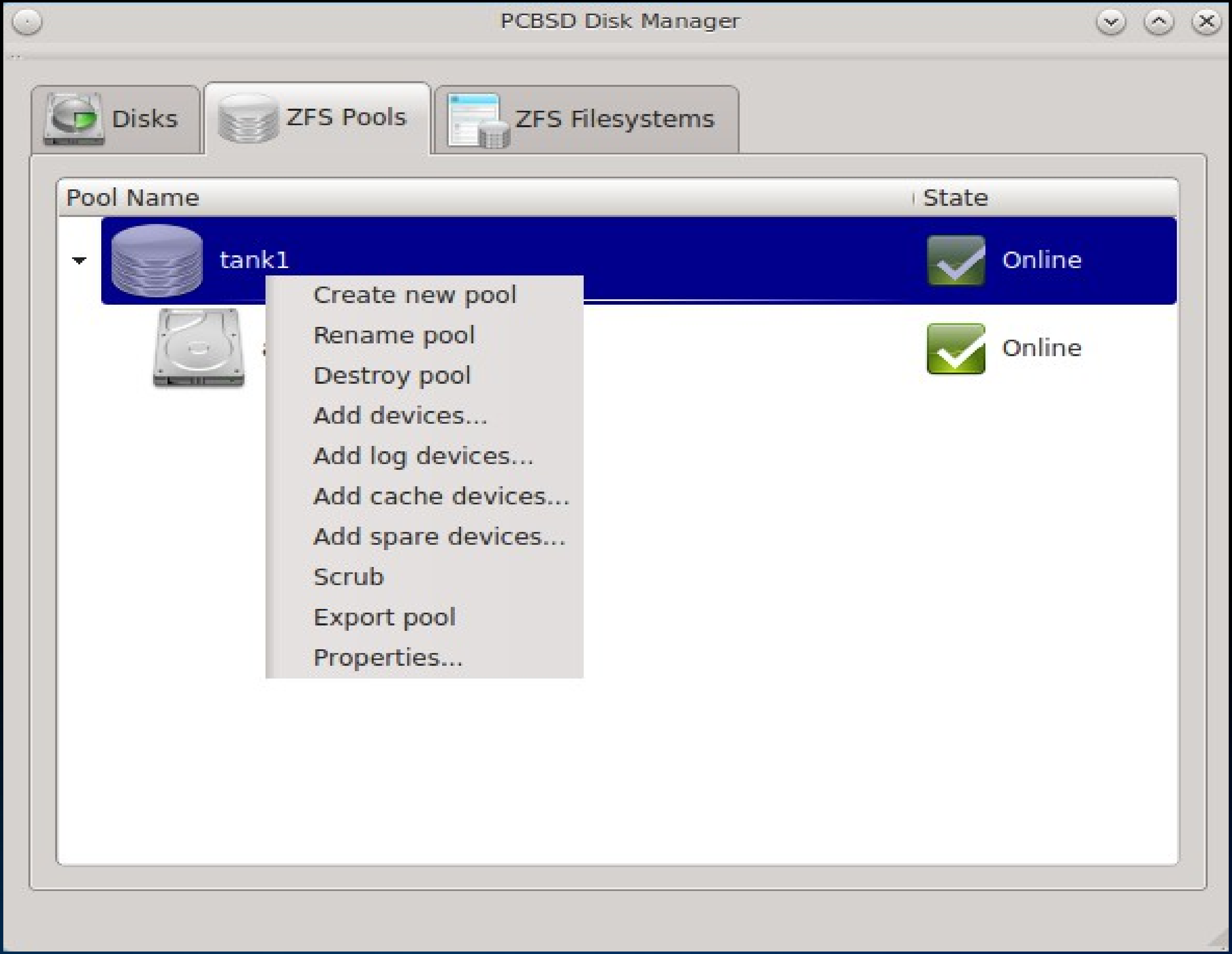
□ Encryption

□ Initialize Safely

Available disks: 1 - 21.5 GB (no more drives)

Volume layout (Estimated capacity: 18.00 GiB): Stripe

Adding SLOG/L2ARC on PC-BSD



Datasets

As needed, pool can be divided into additional, dynamically sized filesystems known as datasets

Permissions and properties such as quotas and compression can be set on a per-dataset level

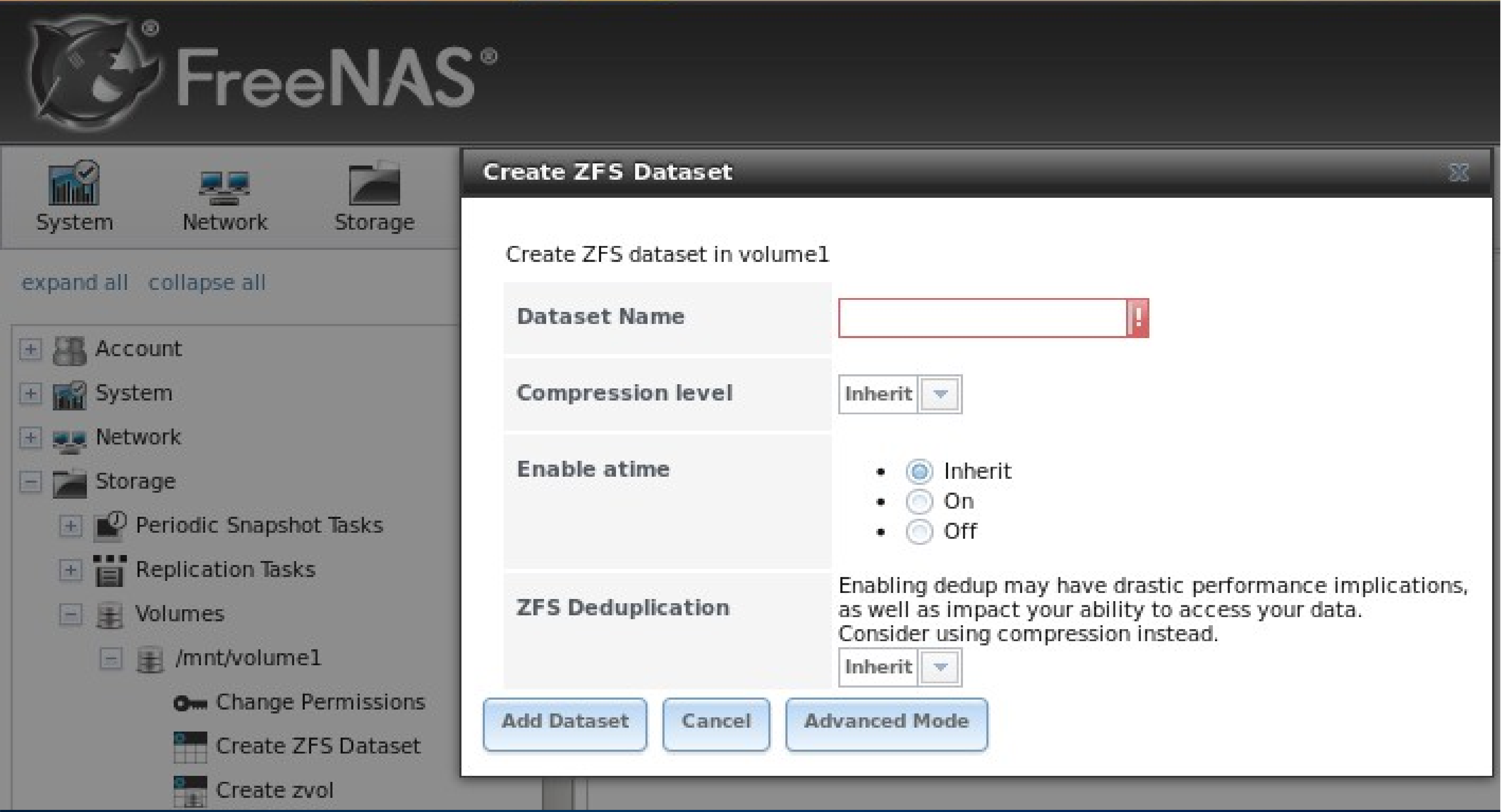
A well thought out design can optimize storage for the type of data being stored

Properties

Dozens of configurable properties such as: atime (access time), canmount, compression, copies, dedup, exec, quota, userquota, groupquota, readonly, recordsize, reservation, setuid, etc.

Descriptions can be found at http://www.freebsd.org/cgi/man.cgi?query=zfs

Adding Dataset on FreeNAS

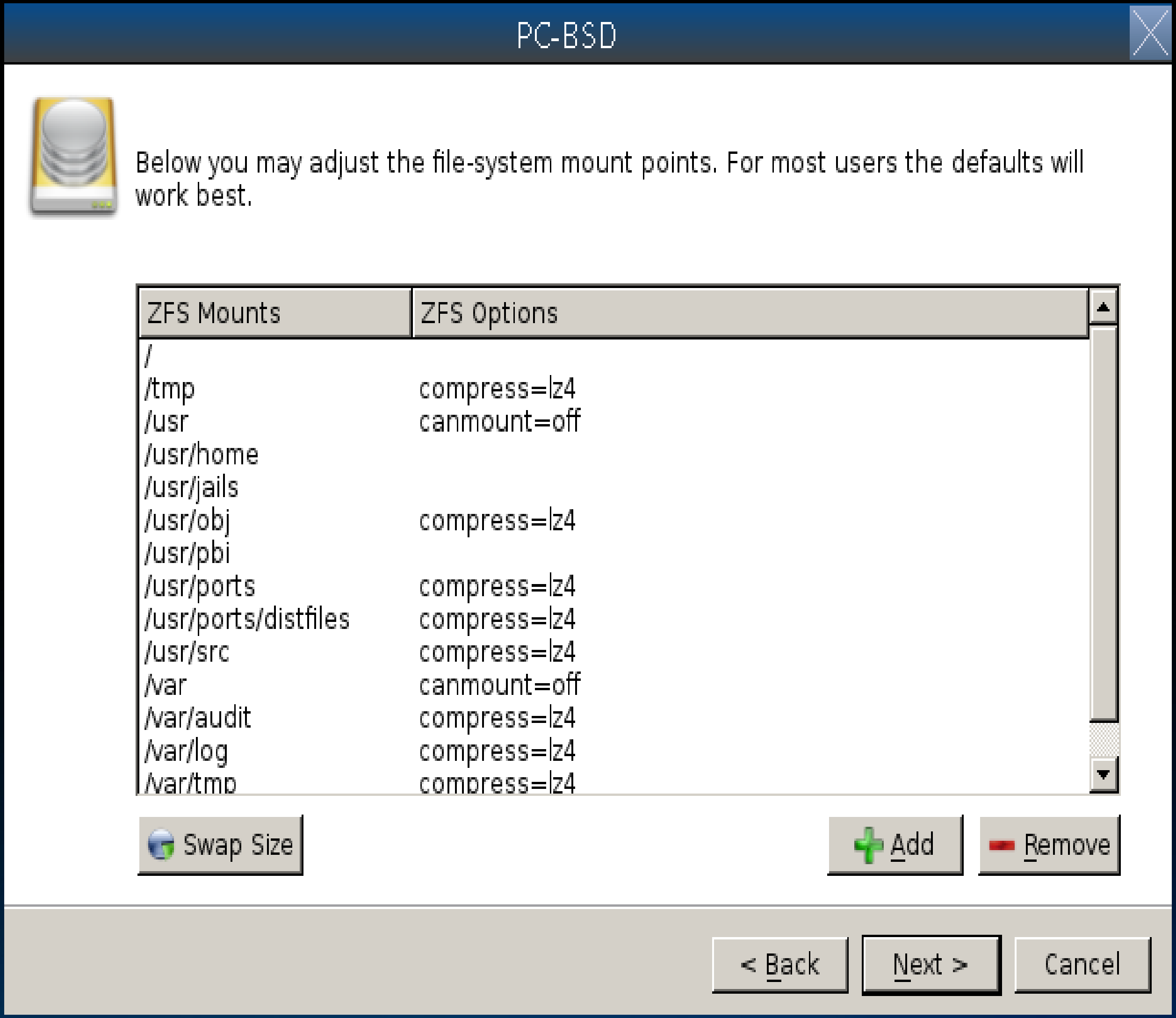


Create ZFS Dataset

Create ZFS dataset in volume1

Dataset Name	
Compression level	Inherit
Enable atime	● Inherit ○ On ○ Off
ZFS Deduplication	Enabling dedup may have drastic performance implications, as well as impact your ability to access your data. Consider using compression instead. Inherit

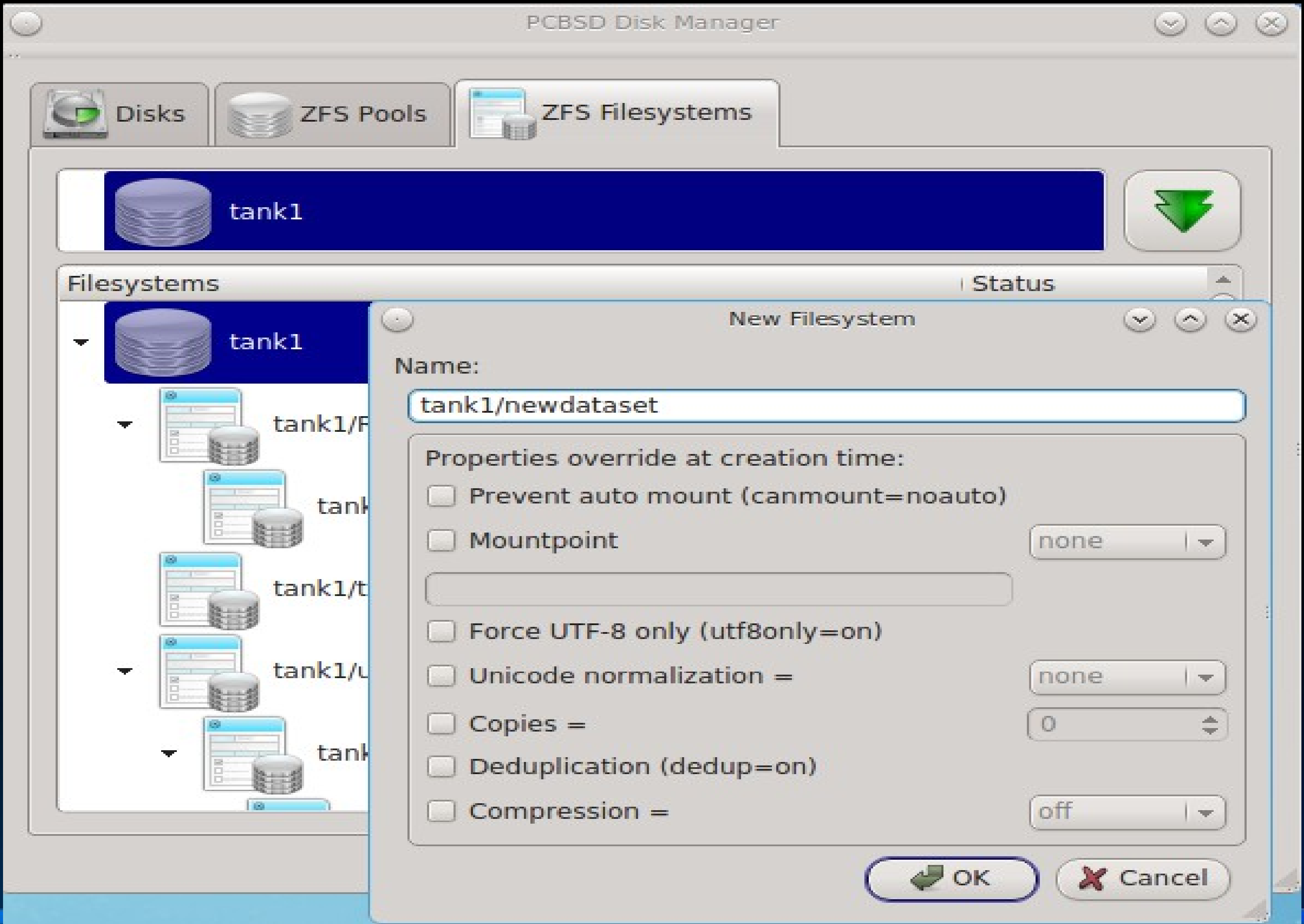
Adding Dataset During PC-BSD Installation



Below you may adjust the file-system mount points. For most users the defaults will work best.

ZFS Mounts	ZFS Options
/	
/tmp	compress=lz4
/usr	canmount=off
/usr/home	
/usr/jails	
/usr/obj	compress=lz4
/usr/pbi	
/usr/ports	compress=lz4
/usr/ports/distfiles	compress=lz4
/usr/src	compress=lz4
/var	canmount=off
/var/audit	compress=lz4
/var/log	compress=lz4
/var/tmp	compress=lz4

Adding Dataset Using PC-BSD Disk Manager



New Filesystem

Name:

tank1/newdataset

Properties override at creation time:

- □ Prevent auto mount (canmount=noauto)
- □ Mountpoint — none
- □ Force UTF-8 only (utf8only=on)
- □ Unicode normalization = none
- □ Copies = 0
- □ Deduplication (dedup=on)
- □ Compression = off

Zvols

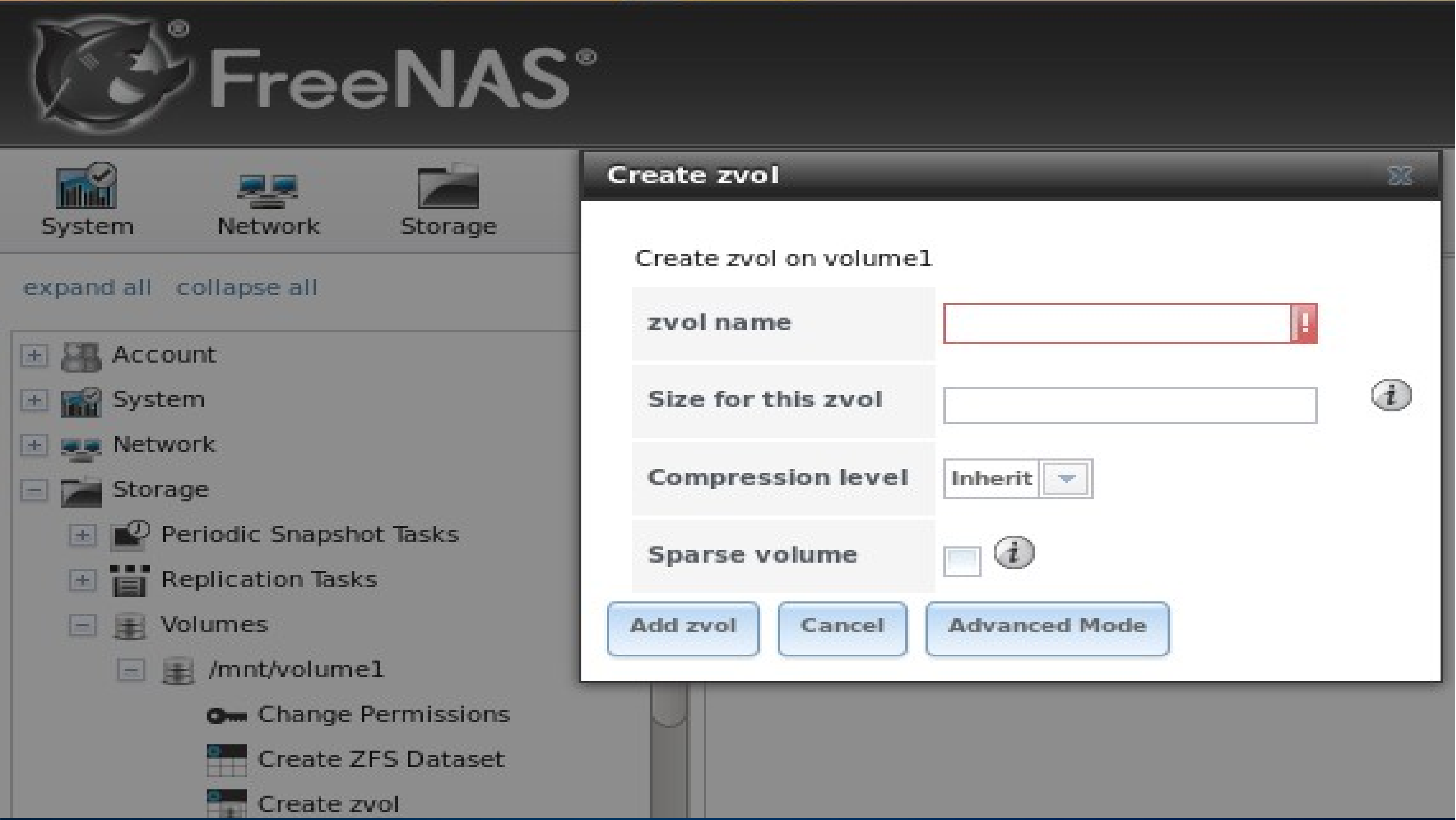
Pool can also be divided into zvols

Essentially, a virtual, raw block device

Ideal for iSCSI device extents or for hosting foreign file systems

Regardless of the filesytem the zvol is formatted with by the iSCSI initiator, the underlying disk blocks still benefit from all of the features provided by ZFS

Creating Zvols on FreeNAS



Create zvol

Create zvol on volume1

zvol name	
Size for this zvol	
Compression level	Inherit
Sparse volume	□

Snapshots

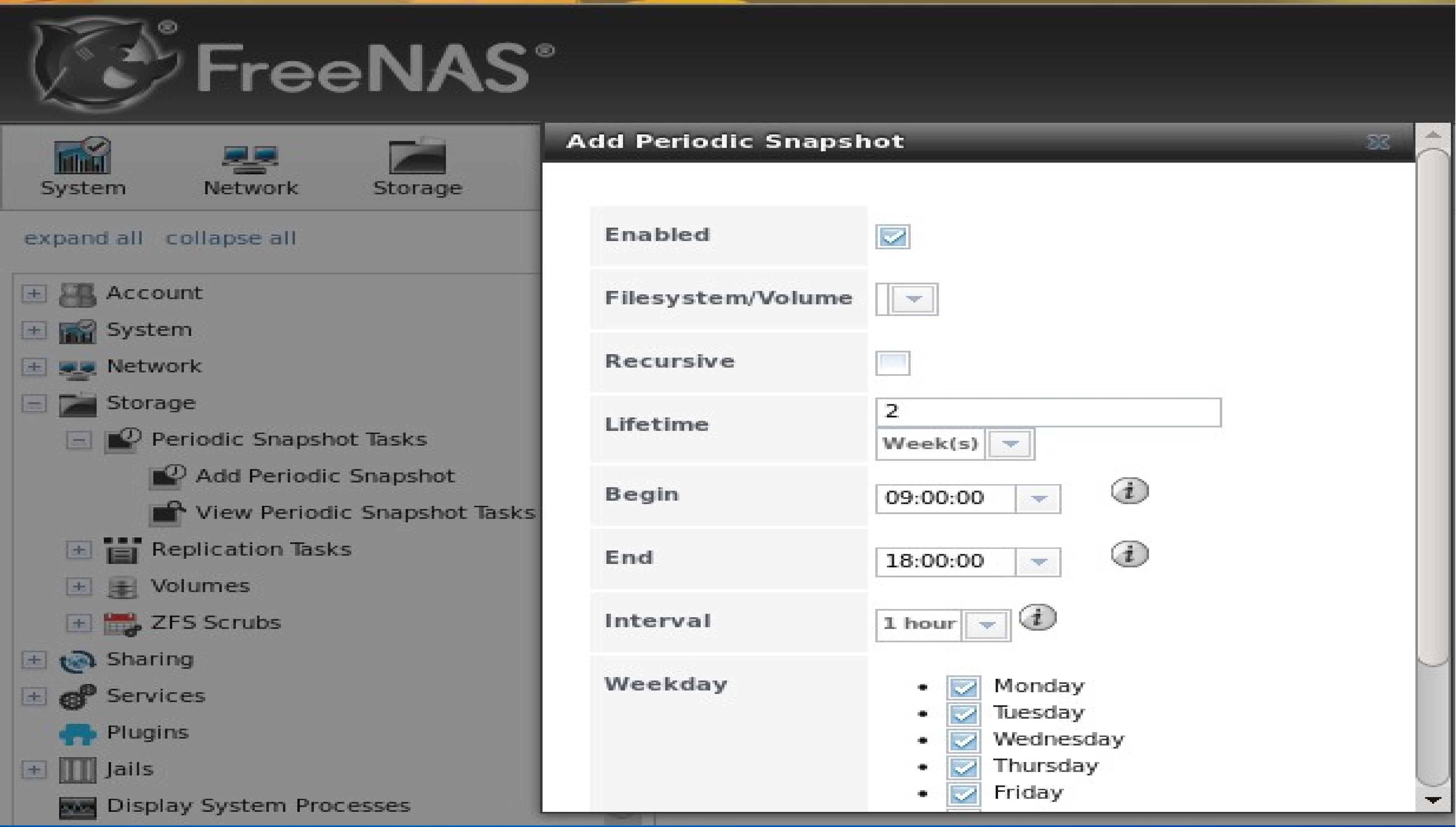
Provide low cost, instantaneous, read-only, point-in-time image of the specified pool, dataset, or zvol

Snapshots can be recursive (atomic inclusion of all child datasets)

Initial size is 0 bytes as COW, snapshot increases in size as changes are written to disk

Can be replicated to another system

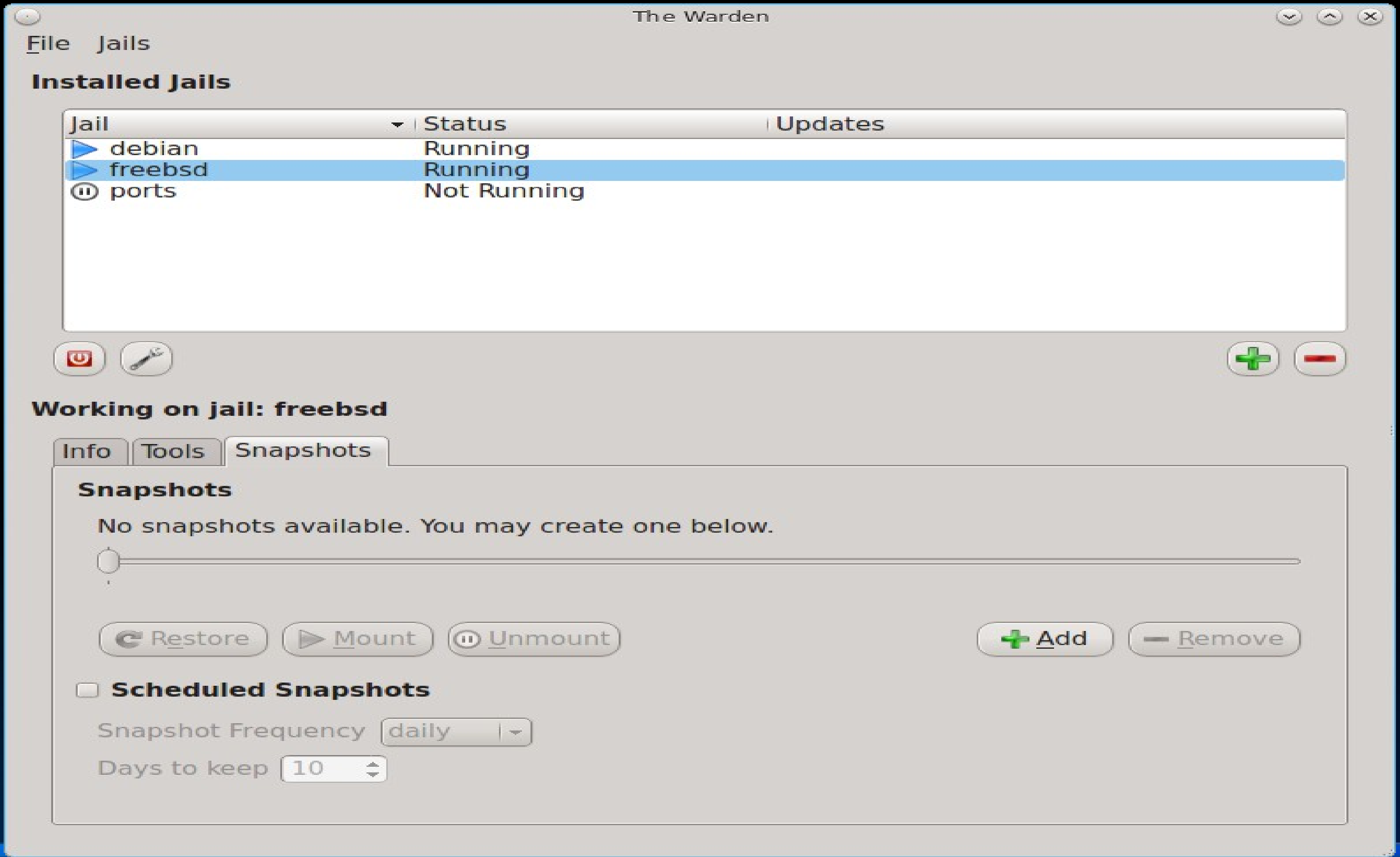
Create Snapshot on FreeNAS



Add Periodic Snapshot

Enabled	☑
Filesystem/Volume	
Recursive	□
Lifetime	2 Week(s)
Begin	09:00:00
End	18:00:00
Interval	1 hour
Weekday	☑ Monday ☑ Tuesday ☑ Wednesday ☑ Thursday ☑ Friday

Create Snapshot on PC-BSD Using Warden



Installed Jails

Jail	Status	Updates
debian	Running	
freebsd	Running	
ports	Not Running	

Working on jail: freebsd

Snapshots

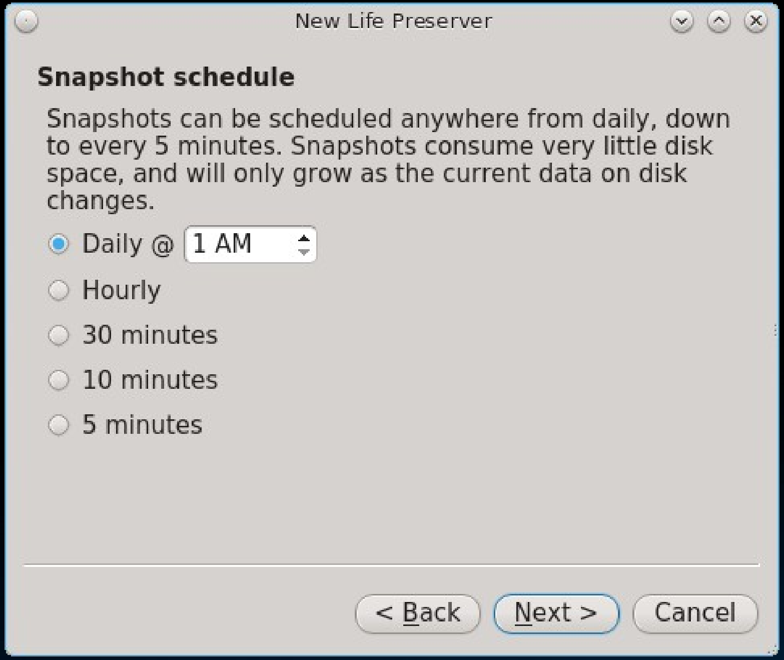
No snapshots available. You may create one below.

□ **Scheduled Snapshots**

Snapshot Frequency: daily

Days to keep: 10

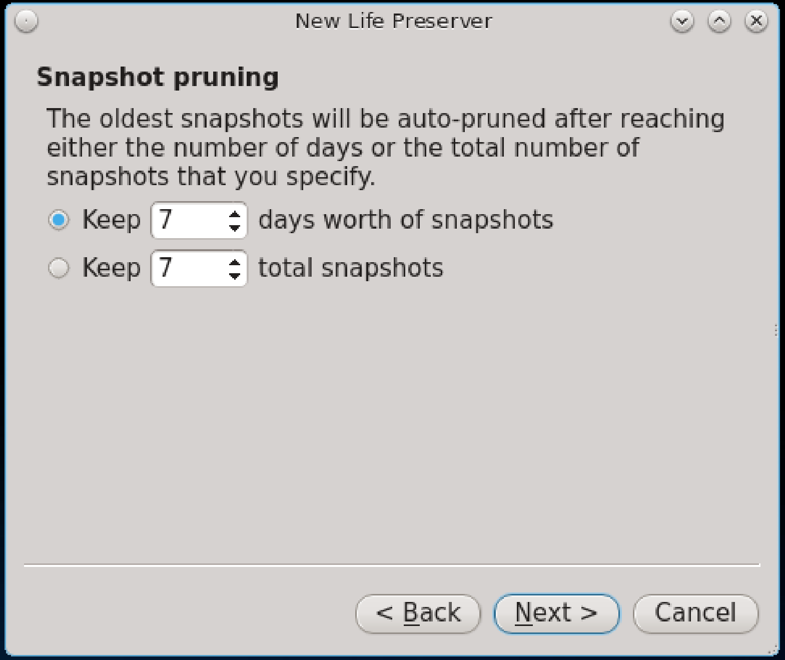
Automating Snapshots on PC-BSD Using Life Preserver



Snapshot schedule

Snapshots can be scheduled anywhere from daily, down to every 5 minutes. Snapshots consume very little disk space, and will only grow as the current data on disk changes.

- ● Daily @ 1 AM
- ○ Hourly
- ○ 30 minutes
- ○ 10 minutes
- ○ 5 minutes



Snapshot pruning

The oldest snapshots will be auto-pruned after reaching either the number of days or the total number of snapshots that you specify.

- ● Keep 7 days worth of snapshots
- ○ Keep 7 total snapshots

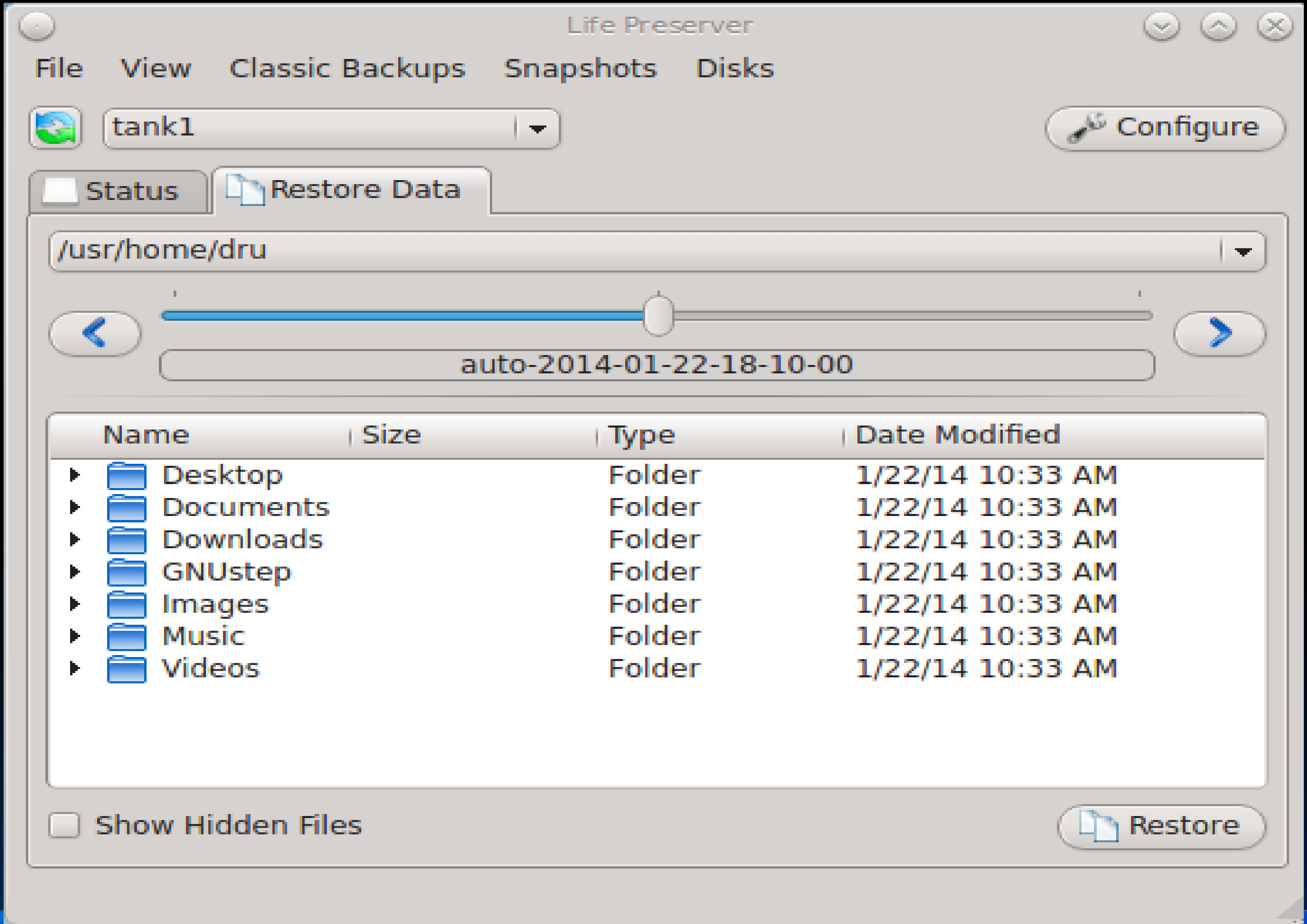
Snapshot Restore

In PC-BSD, the Life Preserver utility provides a snapshot browser for finding and restoring copies of earlier versions of files

It can also automate the replication of local snapshots to another system or to a FreeNAS system over SSH

A remote snapshot can be used to perform an operating system restore from a PC-BSD install media, should the system become unusable

Restoring Data from a PC-BSD Snapshot



tank1

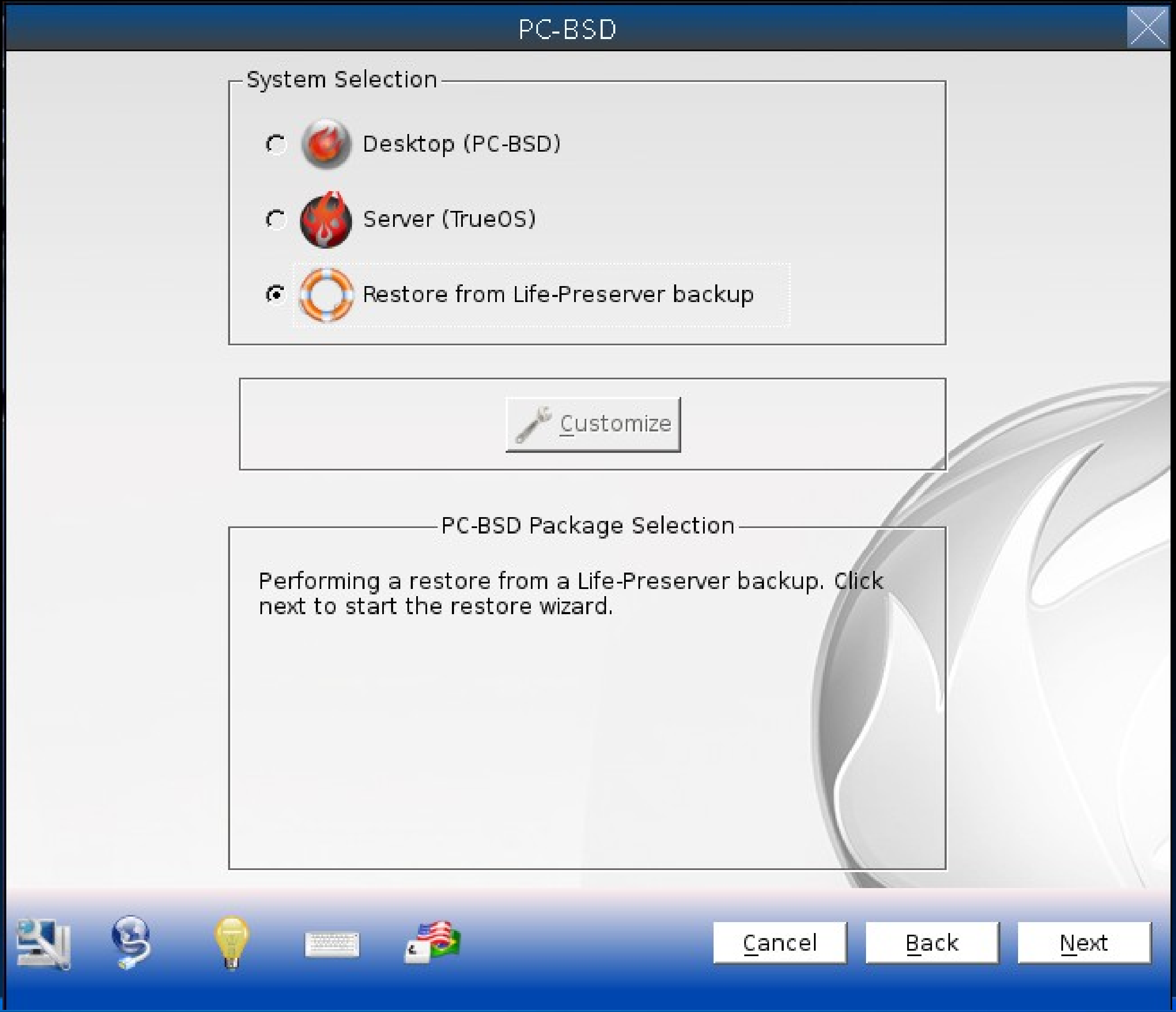
/usr/home/dru

auto-2014-01-22-18-10-00

Name	Size	Type	Date Modified
Desktop		Folder	1/22/14 10:33 AM
Documents		Folder	1/22/14 10:33 AM
Downloads		Folder	1/22/14 10:33 AM
GNUstep		Folder	1/22/14 10:33 AM
Images		Folder	1/22/14 10:33 AM
Music		Folder	1/22/14 10:33 AM
Videos		Folder	1/22/14 10:33 AM

□ Show Hidden Files

Restoring the OS From a Remote Snapshot



PC-BSD

System Selection

○ Desktop (PC-BSD)

○ Server (TrueOS)

● Restore from Life-Preserver backup

PC-BSD Package Selection

Performing a restore from a Life-Preserver backup. Click next to start the restore wizard.

Scrubs

ZFS was designed to be self-healing; it creates and verifies checksums as data is written to disk

A scrub verifies the checksum in each disk block and attempts to correct data as necessary

I/O intensive, so should be scheduled appropriately

Reading the scrub results can provide an early indication of possible disk failure

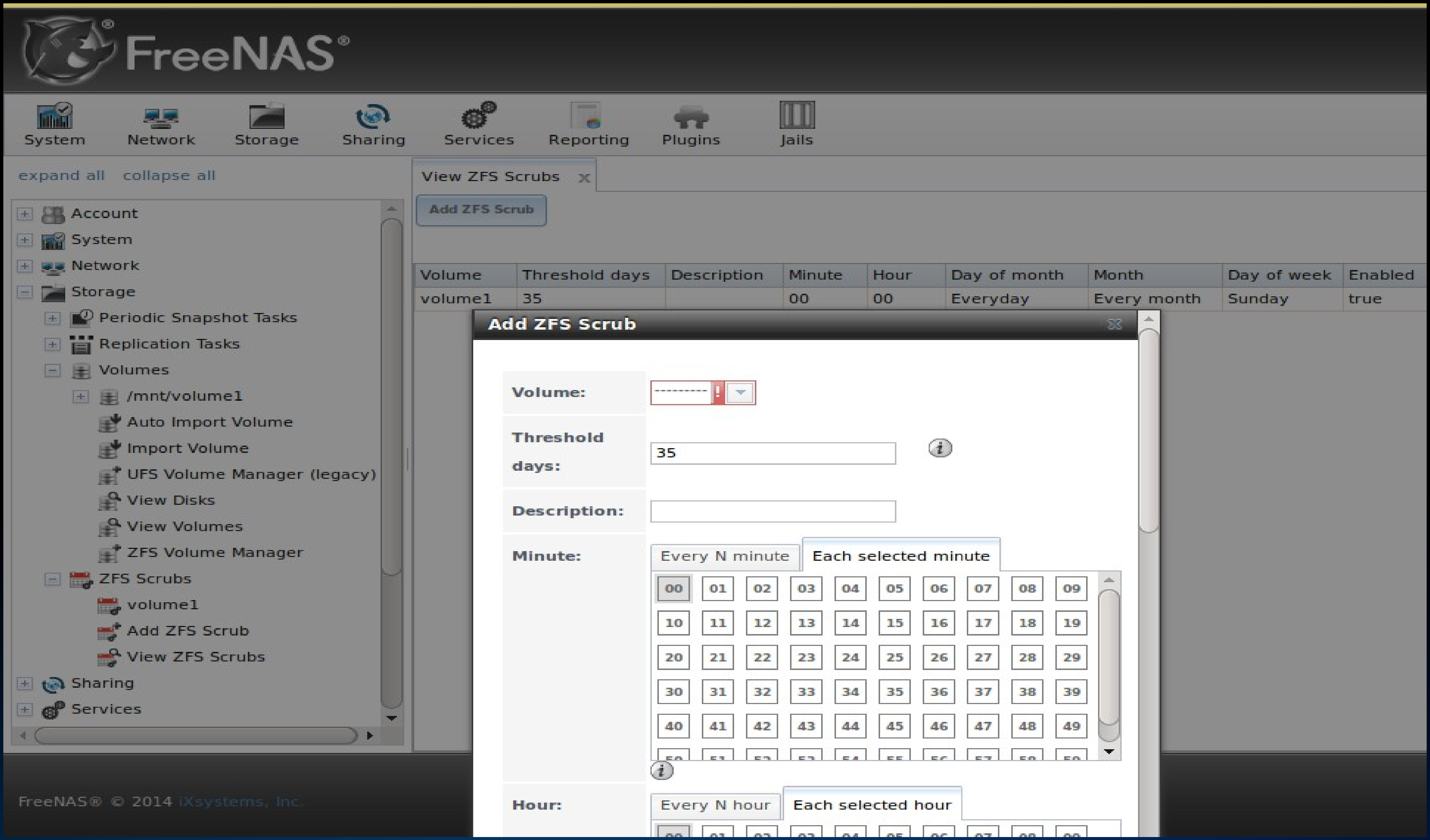
Scrubs

In FreeNAS, a scrub is automatically scheduled to run every Sunday at midnight whenever a pool/volume is created (this can be edited)

The results of the last scrub can be viewed from Volume Status or by typing “zpool status”, and a scrub can be started now from View Volumes

In PC-BSD, a scrub can be started from Disk Manager or Life Preserver

Scheduling Scrubs on FreeNAS



Volume	Threshold days	Description	Minute	Hour	Day of month	Month	Day of week	Enabled
volume1	35		00	00	Everyday	Every month	Sunday	true

Add ZFS Scrub

Volume: ---------

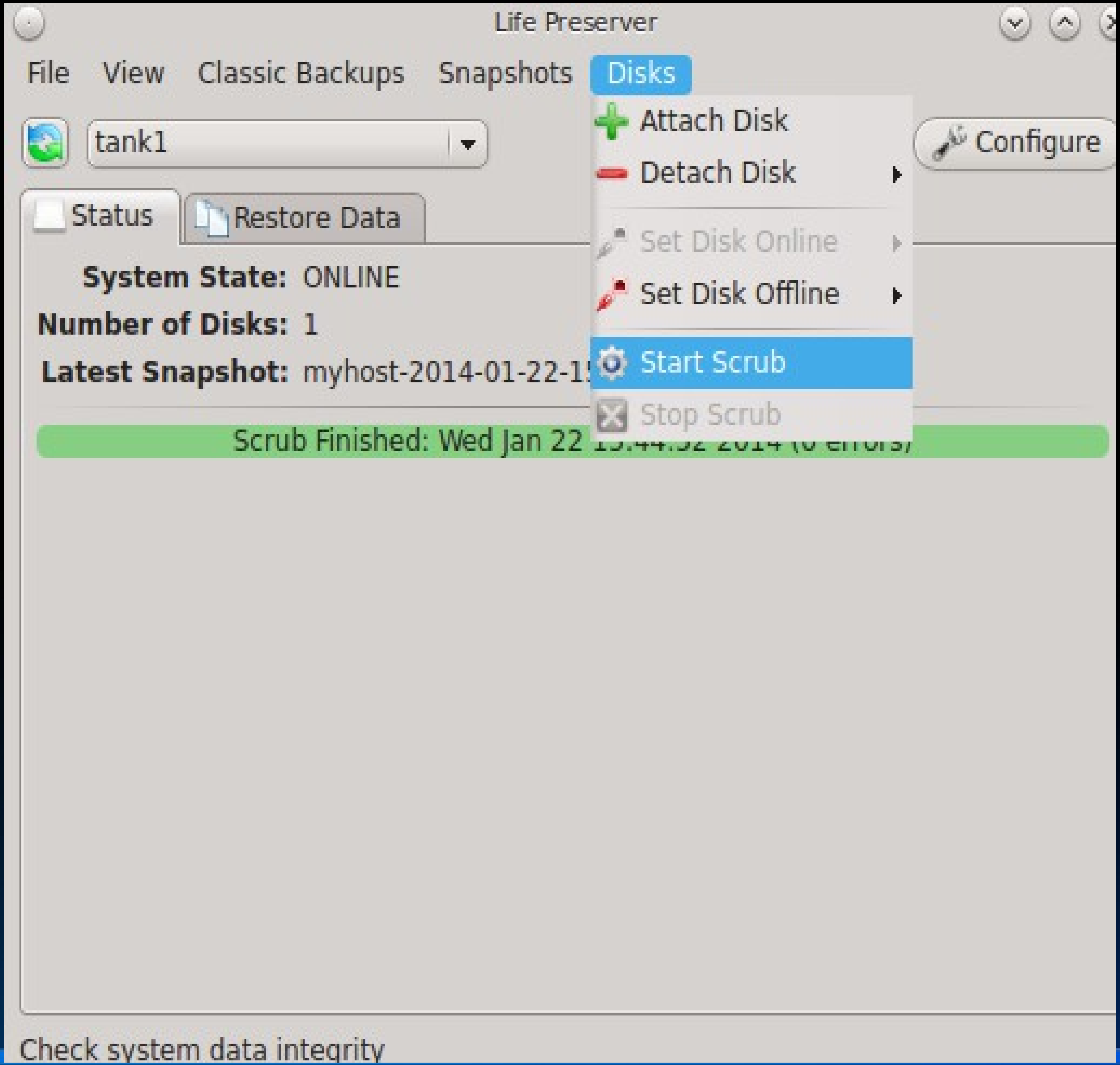
Threshold days: 35

Description:

Minute: Every N minute | Each selected minute

Hour: Every N hour | Each selected hour

Starting a Scrub on PC-BSD



System State: ONLINE

Number of Disks: 1

Latest Snapshot: myhost-2014-01-22-1

Scrub Finished: Wed Jan 22 15:44:52 2014 (0 errors)

Check system data integrity

Deduplication

ZFS property which avoids writing duplicate data

Can improve storage efficiency at the price of performance—compression is often the better choice

Dedup tables must fit into L2ARC, rule of thumb is at least 5 GB RAM/L2ARC per TB of storage to be deduplicated

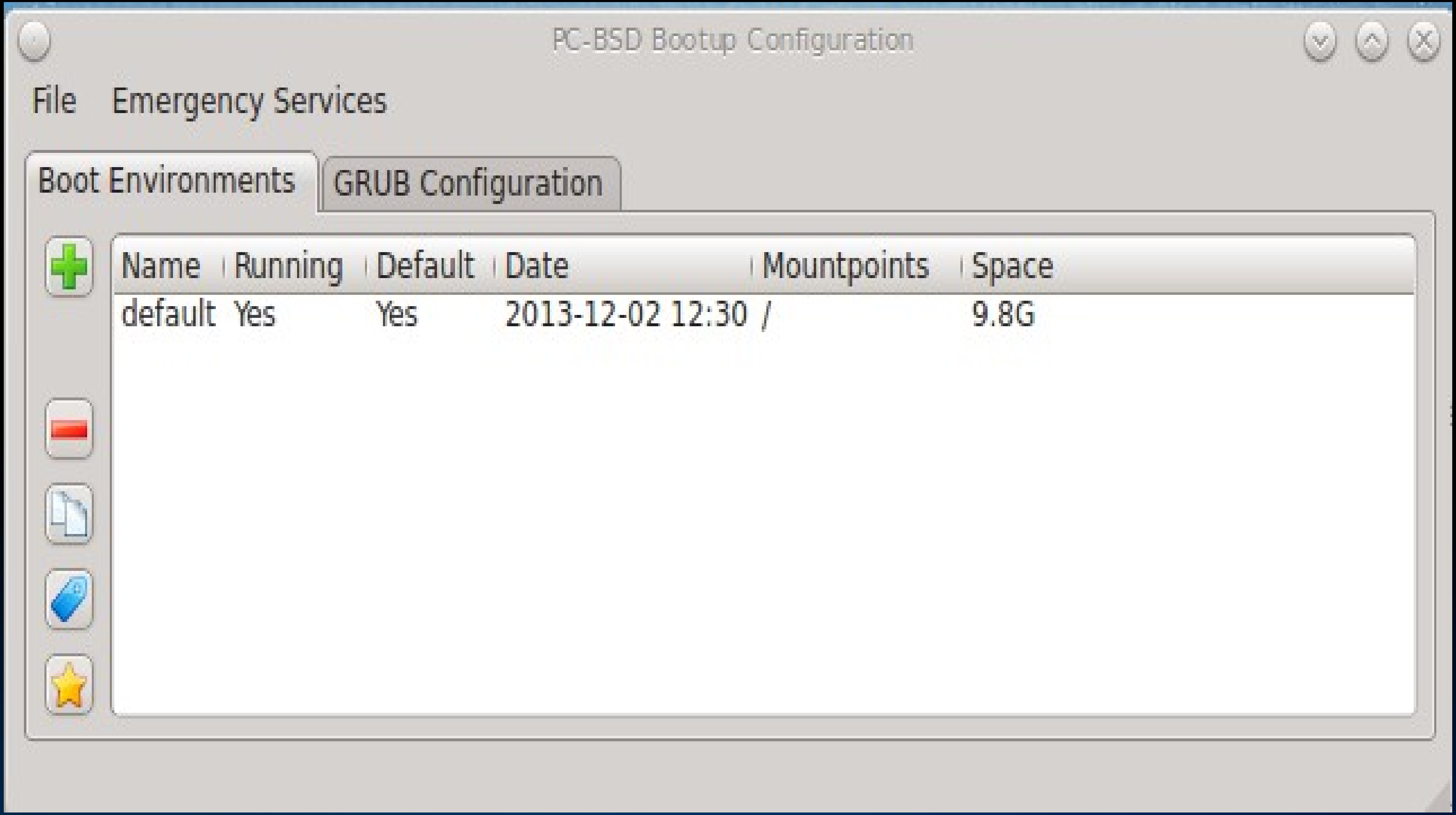
PC-BSD Boot Environments

A snapshot of the dataset the operating system resides on can be taken before an upgrade or a system configuration change

This saved “boot environment” is automatically added to the GRUB boot manager

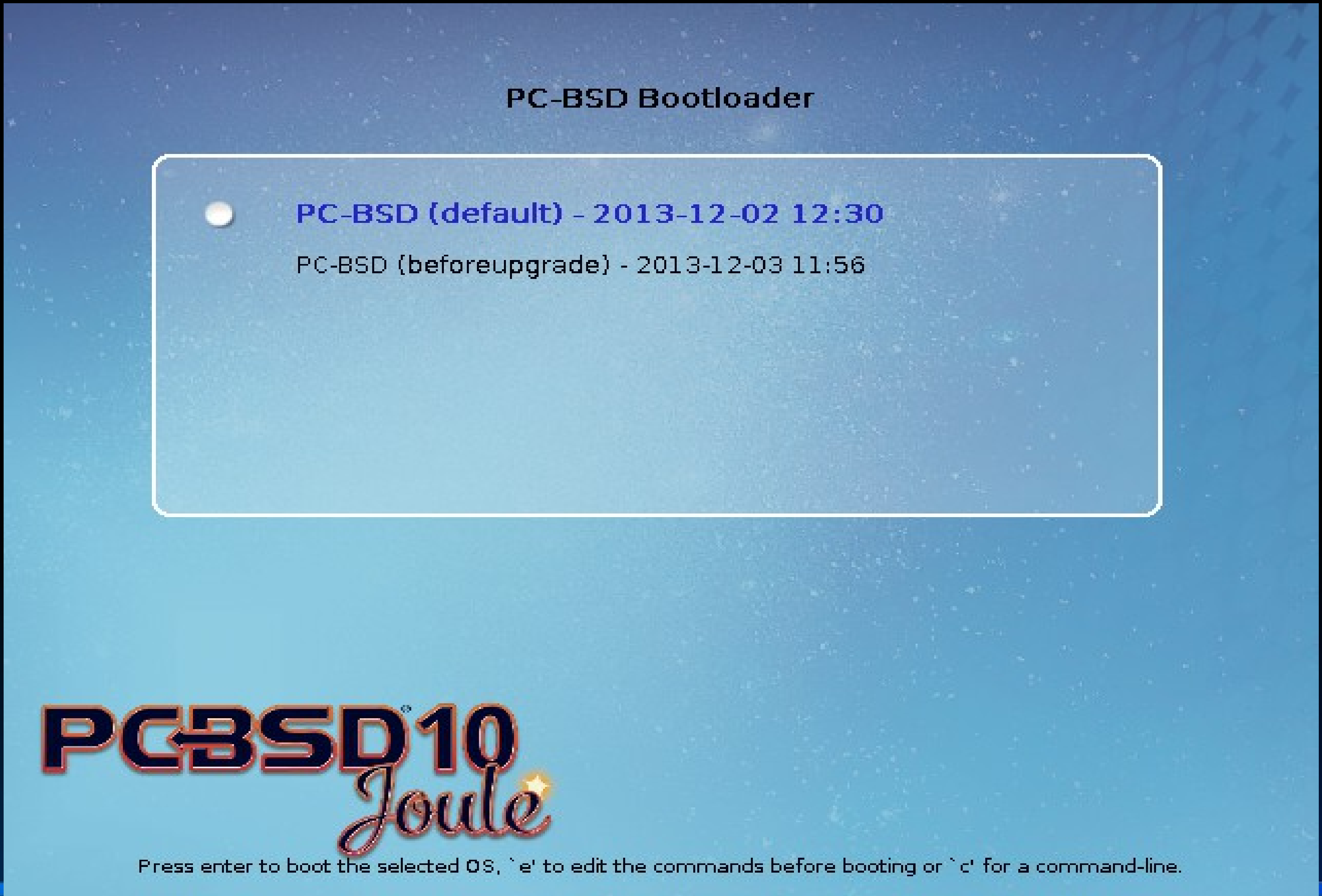
Should the upgrade or configuration change fail, simply reboot and select the previous boot environment from the boot menu

Managing PC-BSD Boot Environments



Name	Running	Default	Date	Mountpoints	Space
default	Yes	Yes	2013-12-02 12:30	/	9.8G

Managing PC-BSD Boot Environments



PC-BSD Bootloader

○ PC-BSD (default) - 2013-12-02 12:30

PC-BSD (beforeupgrade) - 2013-12-03 11:56

Press enter to boot the selected OS, `e' to edit the commands before booting or `c' for a command-line.

Additional Resources

PC-BSD Users Handbook: wiki.pcbsd.org

FreeNAS User Guide: doc.freenas.org

ZFS Best Practices Guide: http://ow.ly/oHtP3

Becoming a ZFS Ninja:
https://blogs.oracle.com/video/entry/becoming_a_zfs_ninja

Questions

Contact:

dru@freebsd.org

URL to Slides:

http://slideshare.net/dlavigne/scale2014