
BE Utility for Snap Upgrade

Design Specifications Revision 0.3

1. Introduction

1.1 Overview

The BE utility is the user interface for managing Boot Environments and is the replacement for the Live Upgrade commands (luupgrade(1M), lucreate(1M)...) This utility is intended to be used by System Administrators who want to manage multiple Solaris Instances on a single system. The BE utility will be implemented with ZFS support only, however a migration path from UFS to ZFS will also be supported but not in the Spring release.

1.2 Definitions

BE - Boot Environment

This is an instance of a bootable Solaris environment consisting of a set of mount points, file systems, ZFS data sets and Zones. The file systems that make up a BE, which are supported by beadm, will only be ZFS file systems. The currently active UFS BE's will be migrated to a ZFS BE.

2. Requirements

The BE utility will support the following operations:

- Create a new BE, based on the active BE.
- Create a new BE, based on a BE other than the active BE.
- Mount a BE.
- Unmount a BE.
- Destroy a BE.
- Upgrade the active or non-active BE to a later version of a Solaris Image
A Solaris Image can be any valid Solaris installation medium.
- List all BE's in a human readable format.
- Activate any ZFS BE and the source UFS BE if migration to ZFS fails.
- Convert separate File Systems on the active BE into separate datasets
on an alternate BE.

2.1 Beadm tools requirements

Currently the LU tools require the user to download and install the latest LU patches and packages from the release they are upgrading to in order to successfully upgrade a BE. beadm will automatically download and install the latest beadm and dependent packages from a pkg(5) image server before attempting an upgrade operation. See 3.2.7.5 for more details.

2.1.2 pkg(5)

The pkg(5) command will be the actual command that upgrades the bits during an upgrade. pkg(5) must be able to upgrade the following BE configurations:

- Active BE
- Active BE with Zones
- Active BE with Zones but only upgrade the active BE
- Active BE with Zones but only upgrade a subset of the Zones

- Non-active BE

- Non-active BE with Zones
- Non-active BE with Zones but only upgrade the active BE
- Non-active BE with Zones but only upgrade a subset of the Zones

2.1.4 UFS -> ZFS migration

beadm will support migrating the active UFS BE to a root zpool(1M) and ZFS file systems however this requirement will be slated for a release later than the Spring release.

2.1.5 Boundary conditions for the upgrade subcommand

The beadm upgrade feature will also be available in pkg(5) as image-update. Decisions need to be made if beadm upgrade and image-update should coexist or the feature only exists in one or the other or have pkg(5) contain all the beadm subcommands in one form or another. Whatever CLI manifestations occur the features described here, in one form or another, should be implemented.

3. Command line interface

Name:

/usr/sbin/beadm - Manage Boot Environments

Synopsis:

beadm [-?]
Display usage.

beadm create [-a] [-f] [-z] [-e non-activeBeName | beSnapshot]
[-o property=value] ... [-p zpool] [-u] beName

beadm create beSnapshot

- a - Activate the newly created BE. Default is to not activate the newly created BE.
- e - non-activeBeName - Create a new BE from a non-active BE. Default is to create the BE from the active BE.
- e - beSnapshot - Create a new BE from a previously created snapshot. beSnapshot must be of the form BE@<description>
- f - Force moving of Zones to the supported namespace if they reside outside the default namespace.
- p zpool - Create a new BE in the specified root zpool. Default is to create the BE in the active zpool.
- o - property=value - Create a new BE with ZFS properties. Multiple -o options can be specified. See zfs(1M) for more information on the -o option.
- u - Create private zone datasets if they exist in the source BE. Default is to NOT create private zone datasets.
- z - Do not create Zones on the target BE if they exist in the source BE. Default is to create the Zones. -z and -u are mutually exclusive.

beName - Name of the BE to create.

beSnapshot - Name of the snapshot to create.

beadm destroy [-f] beSnapshot | beName
-f - Don't prompt if interaction is needed.
beSnapshot - must be of the form beName@description

beadm upgrade [-d] [-Z] [-u] [-z zoneName[,zoneName ...]]
[-n beName[,beName...]] pathToSolarisImage
-d - dryrun. Verify the BE configuration and Solaris image.

-u - Don't display a message asking to download beadm software.

-Z - Do not upgrade any Zones. Mutually exclusive with
the -z option.

-z zoneName - Upgrade one or more non-global Zones.
Not specifying the -z option defaults to upgrading beName
plus all the Zones that are part of that BE. zoneName
is a comma separated list of Zone names. A Zone
can only be upgraded if it's parent BE is being upgraded.

-n beName - Optional beName to upgrade. Default is to upgrade the
active BE. The keyword all means to upgrade all known BE's
on the system.

beadm rename beName newBeName

beadm list [[-a] | [-d] [-s] [-z]] [-H] [beName]
-a - List snapshots, containers and datasets.
-d - List the datasets.
-H - Don't list any header information.
-s - List the snapshots.
-z - List the containers.
beName - If beName is not provided then list all BE's.
The default is to list BE's without any additional
information. If beName resides in a non-active zpool
then the list subcommand will attempt to find it.

beadm mount beName [mountpoint]

beadm unmount beName

beadm activate beName
Make beName the active BE upon next reboot.

Returns

0 - Success
>0 - Failure

3.1 Log interface

/var/log/beadm/<beName>/create.log.<yyyymmdd_hhmmss>
Log used for capturing "beadm create" output
/var/log/beadm/<beName>/upgrade.log.<yyyymmdd_hhmmss>
Log used for capturing "beadm upgrade" output

yyyymmdd_hhmmss - 20071130_140558
yy - year; 2007
mm - month; 11
dd - day; 30

hh - hour; 14
mm - minute; 05
ss - second; 58

/var/log is chosen since it will probably be shared across all BE's.

The actual information contained in the log is yet to be determined.

3.2 Command line interface details

3.2.1 beadm create ...

The create option will create a boot environment on a ZFS enabled system from the active BE or any non-active BE. Each BE will be its own dataset in a reserved namespace. Each BE could also contain separate datasets for traditional file systems (/usr, /var...) and separate datasets for any Zones that are configured with zonecfg(1M). The following shows datasets for some example BE's and Zones:

```
rpool1/ROOT/BE1
rpool1/ROOT/BE1/var
rpool1/ROOT/BE1/usr
rpool1/ROOT/BE1/zone1
rpool1/ROOT/BE1/zone2

rpool2/ROOT/BE2
rpool2/ROOT/BE2/zone3
rpool2/ROOT/BE2/zone4
```

rpool1 and rpool2 are zpool's. Note that the zpool names are just an example and the pools will already exist on the system, previously setup by initial install or upgrade. BE is the default name given to the directory containing the boot environments. All BE's will be located within the default name space. BE1 and BE2 are the names of the BE's. zone{1,2,3,4} are Zones within BE's BE1 and BE2.

If the -o option is provided then the behavior of the -o option is exactly as described in zfs(1M). The properties will be passed to zfs when creating the Boot Environment. If an invalid property is provided and zfs(1M) fails to add the property to the dataset, then the creation of the Boot Environment will fail and appropriate message will indicate the error.

If a snapshot is provided as the -e argument then the source of the target BE is a previously create snapshot of a BE. If a snapshot is not a previously created snapshot of a BE then beadm will exit with an error and display an appropriate message.

If a snapshot is the target of the create subcommand, the second form in section 3, then a snapshot will be attempted to be created insted of a BE. A snapshot will only be taken from the active BE.

3.2.1.1 Creating Zones

By default, Zones on the source BE, if they exist, will automatically be created on the target BE depending upon there location on the source BE. The following sections describe those different scenarios and how beadm create will handle them.

3.2.1.2 Zone in BE root name space

In this case, all the datasets will be cloned and the only change will be the path of the resultant datasets. The path will now contain the name of the target BE.

Existing source BE:

```
rpool/ROOT/BE1/  
rpool/ROOT/BE1/usr  
rpool/ROOT/BE1/var  
rpool/ROOT/BE1/opt  
rpool/ROOT/BE1/zones/zone1
```

Target BE after creation:

```
rpool/ROOT/BE2/  
rpool/ROOT/BE2/usr  
rpool/ROOT/BE2/var  
rpool/ROOT/BE2/opt  
rpool/ROOT/BE2/zones/zone1
```

3.2.1.3 Zone in a shared dataset

When a Zone resides in a shared file system it needs to be cloned and renamed. A SMF service will run before local file systems are mounted and will modify the mountpoints of the zone datasets so that the zone dataset corresponding to the BE being booted gets mounted at the right place.

Existing source BE:

```
rpool/ROOT/BE1/  
rpool/ROOT/BE1/usr  
rpool/ROOT/BE1/var  
rpool/ROOT/BE1/opt  
rpool/export  
rpool/export/zone1
```

Target BE after creation:

```
rpool/ROOT/BE2/  
rpool/ROOT/BE2/usr  
rpool/ROOT/BE2/var  
rpool/ROOT/BE2/opt  
rpool/export  
rpool/export/zone1          (mounted as /export/zone1-BE1)  
rpool/export/zone1-BE2     (mounted as /export/zone1)
```

rpool/export is a shared dataset which contains the Zone zone1. zone1 must be cloned and will be named <zoneName-beName> as shown above. During boot, a SMF service will mount the cloned zone at /export/zone1. The shared zone that was cloned from BE1 will be renamed as described above.

3.2.1.4 Zone with private file systems

A Zone can be configured with its own dedicated file system (via 'add fs' in zonecfg(1M)). When a Zone contains a dedicated file system, copying the zone to a target BE needs to be handled via two different methods. The private file system needs to be setup as a private or shared dataset in the target BE. For example, a Zone has a dedicated /var and /export file system. The /var file system shouldn't be shared, but the /export file system should be. Note that this is determined by an Administrator during beadm create. The -u option described in section 3.0 is intended for admins to use to specify that all existing private Zone datasets will be private after creating the target BE. By default, if the -u option is not provided beadm create will create all private datasets as shared. Also note that there isn't an option for creating private zone UFS file systems as private in the target BE. They will be created as

shared.

For the Zones private file systems that are not shared, the dataset gets cloned and named something that is specific to the target BE. The Zones configuration data in the target BE gets modified to denote which file system to use as the private file system. The following shows an example of the above scenario:

Existing source BE:

```
rpool/ROOT/BE1/  
rpool/ROOT/BE1/usr  
rpool/ROOT/BE1/var  
rpool/ROOT/BE1/opt  
rpool/export/zone1  
rpool/zone1_var           (mounted as /var for zone1)  
rpool/zone1_export       (mounted as /export for zone1)
```

zone1 is a Zone in a shared dataset. zone1_var and zone1_export are private datasets for zone1.

Target BE after creation:

```
rpool/ROOT/BE2/  
rpool/ROOT/BE2/usr  
rpool/ROOT/BE2/var  
rpool/ROOT/BE2/opt  
rpool/export/zone1  
rpool/export/zone1-BE2  
rpool/zone1_var           (mounted as /var for zone1 in BE1)  
rpool/zone1_var-BE2      (mounted as /var for zone1 in BE2)  
rpool/zone1_export       (mounted as /export for zone1  
                           in both BE1 and BE2)
```

The example assumes -u was provided to beadm create.

3.2.1.5 Zones on UFS, migrating to ZFS (not supported in Spring 08)

Zones in system file systems (/var, /usr...) will automatically get migrated to ZFS since all system file systems get migrated to ZFS. However, since beadm won't be migrating non-system filesystems (/export) to ZFS, Zones that live there have to be handled differently. Zones that live on those non-system UFS file systems must be migrated to a ZFS file system. The plan for this is that beadm will migrate them to a dedicated dataset area for the target BE.

For example, consider the following UFS based BE with a separate UFS /export with zone1 in it:

Existing source BE:

```
/  
/export  
/export/zone1
```

Target BE after creation:

```
rpool/ROOT/BE1/  
rpool/ROOT/BE1/usr  
rpool/ROOT/BE1/var  
rpool/ROOT/BE1/opt  
rpool/BE_ZONES/BE1/export/zone1 (mounted at /export/zone1)  
/export (existing UFS slice)      (mounted legacy at /export)
```

3.2.2 beadm list ...

Will list all the BE's created by beadm create and any other BE that resides in the reserved namespace.

The following will be the default output:

Boot Environment	Active	Active on reboot	Status	Dataset	Mountpoint	Space Used GB
BE1	yes	no	active	pool1/ROOT/BE1	/	5.4
BE2	no	yes	mounted	pool2/ROOT/BE2	/pool2/ROOT/BE2	6.2

The old tool for listing BE's is lustatus(1). It listed the following attributes that will not be listed by beadm:

- Is Complete
- Can Delete
- Copy Status

beadm create will never create a partial BE. It will either have succeeded or failed. There will never be an in between state.

The "Can Delete" attribute is redundant since all BE's, except the active BE can be destroyed. "destroy" subcommand will not allow the active BE to be destroyed. See "beadm destroy" for more information.

The "Copy Status" attribute is not necessary. The "create" subcommand will either successfully create a new BE based off an existing BE or it will not. beadm will not provide a mechanism for managing read/write control.

As displayed above there will be six attributes describing a BE with the "list" subcommand. All are self explanatory except the "Status" column. The Status column can show one or more of the following states:

- mounted
- unmounted

3.2.2.1 beadm list -d

If the -d option is provided beadm list will display the datasets that make up the BE's. The status of those datasets will either be mounted or unmounted as follows:

Boot Environment	Active	Active on reboot	Status	Dataset	Mountpoint	Space Used	
BE1	yes	no	active	pool1/ROOT/BE1	/	5.4G	
				Datasets	Status	Mountpoint	Space Used
				BE1/var	mounted	/var	55M
				BE1/usr	unmounted	/usr	2.6G
BE2	no	yes	mounted	/pool2/ROOT/BE2	/pool2/ROOT/BE2	6.2G	

Datasets	Status	Mountpoint	Space Used
BE2/var	mounted	/var	65M
BE2/var/log	mounted	/var/log	1M

3.2.2.2 beadm list -z

If the -z option is provided beadm list will display the Zones, if any, that are contained within BE's. The status of those Zones will be the same list that zoneadm(1M) list displays. The following is an example of the output of the subcommand:

Boot Environment	Active	Active on reboot	Status	Root Location	Mountpoint	Space Used
BE1	yes	no	active	/pool1/ROOT/BE1	/	5.4G

Zones	Status	Mountpoint	Space Used
zone1	active	/zones/zone1	220M
zone2	configured	-	-

BE2 no yes mounted /pool2/ROOT/BE2 /pool2/ROOT/BE2

3.2.2.3 beadm list -s

If the -s option is provided beadm list will display the Snapshots, if any, that exist for any BE. The status of those Snapshots will be the same list that zfs(1M) list displays. The Operation column is a one word description of what caused the snapshot to occur. The quantified list is yet to be determined. The following is an example of the output of the subcommand:

Boot Environment	Active	Active on reboot	Status	Root Location	Mountpoint	Space Used
BE1	yes	no	active	/pool1/ROOT/BE1	/	5.4G

Snapshots	Policy	Date
pool1/ROOT/BE1@<snapshotName>	static	2007-10-20
pool1/ROOT/BE1@<snapshotName>	dynamic	2007-11-13

BE2 no yes mounted /pool2/ROOT/BE2 6.2G

3.2.2.4 beadm list -a

If the -a option is provided beadm list will display Datasets, Zones and Snapshots that exist for any BE. The following is an example of the output of the subcommand:

Boot Environment	Active	Active on reboot	Status	Root Location	Mountpoint	Space Used
BE1	yes	no	active	/pool1/ROOT/BE1	/	5.4G

Datasets	Status	Mountpoint	Space Used
----------	--------	------------	------------

```
BE1/var    mounted    /var        55M
BE1/usr    unmounted  /usr        2.6G
```

```
Zones      Status      Mountpoint   Space Used
-----
zone1      active      /zones/zone1 220M
zone2      configured  /zones/zone2  -
```

```
Snapshots          Policy      Date
-----
pool1/ROOT/BE1@<snapshotName> static      2007-10-20
pool1/ROOT/BE1@<snapshotName> dynamic     2007-11-13
```

```
BE2          no          yes          mounted    /pool2/ROOT/BE2          6.2G
```

3.2.2.5 beadm list -H

If the -H option is provided, beadm list will display the BE's and associated fields without header information. The list -H subcommand and option will display BE's, Datasets, Zones and Snapshots, The following is an example of what will be output on stdout for the BE's listed in section 3.2.2.4:

```
BE1:yes:no:active:/pool1/ROOT/BE1:5.4G;pool1/ROOT/BE1/var:mounted:pool1/ROOT/BE1/var:/var:55M,
pool1/ROOT/BE1/usr:unmounted:/pool1/ROOT/BE1/usr:/usr:2.6G;zone1:active:/pool/ROOT/BE1/zone1:/
zones/zone1:220M,zone2:configured:/pool2/ROOT/BE1/zone2:/zones/zone2:-;pool1/ROOT/BE1@initial:
upgrade:2007-10-20,pool1/ROOT/BE1@SUNWbe:SUNWbe-tools:2007-11-13
BE2:no:yes:mounted:/pool1/ROOT/BE2:6.2G;;;
```

The ';' delimits BE's, Datasets, Zones and Snapshots. The ':' delimits attributes for BE's, Datasets, Zones and Snapshots. The ',' delimits multiple Datasets, Zones and Snapshots. Multiple BE's are delimited with a carriage return.

The -H option can be combined with the other options. When the other options are provided along with -H then a subset of the fields shown in the example above will be displayed. Note that the output above is an example of using list -H without any additional options and is the default listing. Invoking the list subcommand with -H and -a will produce the same results as the example output.

3.2.3 beadm destroy ...

beadm destroy will only destroy a non-active BE. If the active BE is attempted to be destroyed, beadm will not allow the destruction.

beadm destroy will prompt the user to continue with the destruction unless the -f option is provided. If the -f option is provided the BE will be destroyed without prompting.

3.2.4 beadm rename ...

beadm rename will allow the user to rename a BE to another name. Since the name of a BE is a part of a ZFS dataset as described above, the dataset name will need to be changed. Once changed the ZFS list command will show the new name in the listing as well as will the beadm list command.

If the BE to be renamed contains Zones then those Zones root paths will automatically be changed since those Zones datasets are inherited from the BE being renamed.

If a dataset exists and is named the same as <newBeName> then the attempt to rename the BE will

fail. The user will have to do a "zfs destroy" and "zfs rename" if they want to change names to an existing dataset.

The active BE will not be allowed to be renamed since the active BE cannot be unmounted.

A rename will not allow a BE to be renamed if <beName> and <newBeName> reside in different zpools.

3.2.5 beadm mount & unmount

3.2.5.1 mount

beadm mount will allow a user to mount a non-active BE to a mount point. If that mount point doesn't exist beadm mount will attempt to create it. If the BE is already mounted at an existing mount point it will be unmounted and mounted at the new mount point. All mount points must be within the namespace mentioned in 3.2.2. If the mount point doesn't reside within the supported namespace then beadm will halt and provide an appropriate message to stderr. If a <beName> is not provided on the command line then beadm will halt and provide an appropriate message to stderr.

3.2.5.1.1 mount & Zones

If Zones exist within the non-active BE to be mounted then mount points will be created if the Zones configuration calls for them. If the -f is provided, mounting the Zones file systems will be done automatically. If -f is not provided and there are additional mount points to create then the user will be notified with an appropriate message and to use the -f option if they want the mounts to be done automatically.

3.2.5.2 unmount

A non-active BE can be unmounted if it is mounted. If that mounted BE contains Zones and those Zones have separate mountable file systems then those file systems will be unmounted.

3.2.6 beadm activate

beadm activate will, with no arguments, display the name of the BE that will be active upon the next reboot of the system. When beName is provided, beadm activate activates the specified BE associated with beName.

beadm activate activates a BE by setting the zpool bootable property, bootfs, to true. The bootfs property of the zpool will be set for the root dataset of the BE that's being activated. beadm activate will then create a grub entry in the BE's menu.lst file. The title displayed to the user will be the name of the BE and the dataset that corresponds to the name of the BE.
e.g:

```
Solaris Boot Environment "BE2" pool/ROOT/BE1
```

Once a BE has been activated it will continue to be an option upon boot. This means that if there are 4 BE's configured on a system and 2 of the 4 have been activated, those 2 will be selectable during boot. The default BE will be the last BE that has been activated. So the boot menu could look as follows:

```
Solaris Boot Environment "BE2" pool/ROOT/BE1  
Solaris Boot Environment "BE1" pool/ROOT/BE2  
OpenSolaris Preview ...  
...
```

If a BE fails to boot, information on how to recover will be displayed at boot time and not at activation time. For most failed boot attempts the instructions will be to simply boot from the

previously booted BE. If the first BE created fails to boot then either the old UFS BE will still be available for booting if the BE has been migrated from UFS or the last ZFS BE will be available.

3.2.7 beadm upgrade

3.2.7.1 Boundary conditions

Note that this feature will also be available in pkg(5) as image-update. Decisions need to be made if beadm upgrade and image-update should coexist or the feature only exists in one or the other or have pkg(5) contain all the beadm subcommands in one form or another. Whatever CLI manifestations occur the features described here, in one form or another, should be implemented.

3.2.7.2 Operational behavior

beadm upgrade will upgrade one or more BE's to the image provided by the user on the command line via the <pathToSolarisImage> argument. By default, the upgrade subcommand will update the entire content of a BE to the new version of Solaris. All the Zones within that BE will be upgraded.

If beName is not provided then the active BE will be upgraded. The active BE will be cloned and upgraded as if were a non-active BE. A snapshot will be taken after the upgrade has been completed and will be the initial snapshot in case that BE ever needs to be rolled back to it's initial post-upgrade state. If the BE fails to upgrade then the cloned BE and its snapshot will be deleted. If the upgrade succeeds then the cloned BE will be promoted and the previous BE will be deleted.

3.2.7.2.1 Operational behavior Zones

All Zones within the target BE will be upgraded unless the -z option is provided. If the -z option is provided then only the list of Zone arguments will be upgraded. If -n is provided then no Zones will be upgraded. The -Z and -z options are mutually exclusive.

The -n option assumes that the user does not want to upgrade the Zones until a later time or that the Zones will never be upgraded and eventually destroyed. Since the global Zone and the non-global Zones must run at the same Solaris version level, any Zone not upgraded at the same time the global Zone is upgraded will be left in or brought to the detached state. If attempted to be brought online after the global Zone is upgraded, zoneadm(1M) must state that the Zones must be upgraded in order to be booted. After the global Zone is upgraded the user can upgrade the Zones at a later time to the same Solaris Image that the global container was upgraded to via the -z option. If a Zone is attempted to be upgraded before the global Zone is upgraded, then beadm will halt the operation and provide an appropriate message to stderr and the log file. Once the upgrade of the Zones is complete they will be attempted to be booted. If unable to boot then an appropriate message will be sent to stderr and the log file.

Non-Native Zones will not be upgraded and their File Systems will be untouched and mounted at the same location after the upgrade.

3.2.7.3 Non-active BE's

If a non-active BE is the target of the upgrade then a snapshot will be taken prior to the upgrade and the BE will be upgraded at it's current mount point. It will not be cloned before the upgrade takes place. If the upgrade fails the snapshot will be cloned and promoted and the original BE will be renamed to <beName>_damaged. It will then be renamed to the original name and mounted at the original mount point. The space available to do the above needs to be verified before the non-active BE is upgraded.

3.2.7.3.1 Non-active BE's Zones

If a target non-active BE contains Zones then those Zones will also be upgraded depending on the options provided. A snapshot for each Zone being upgraded will be created before they are upgraded. If one or more Zones fails to be upgraded an appropriate message will be sent to stderr and the upgrade log file. The snapshot will be deleted and the upgrade will continue if more Zones are to be upgraded.

The Zones can be upgraded after the global Zone is upgraded even if the BE is not the active BE. This implies that beadm needs to compare the "upgrading to" image with the inactive BE image.

The pkg(5) command will need to be able to support upgrading Zones on the non-active BE. See 2.1.3 pkg(5) for more information.

3.2.7.4 Upgrade support

beadm upgrade will only support upgrading from an official Solaris Indiana release that contains the FCS or later version of beadm. If the software finds itself on any other unsupported release, then it will not do the upgrade and display an appropriate message. If the user tries to upgrade to a Solaris version released before the official Indiana release then beadm upgrade will detect this and display an appropriate message to the user.

The release image used as the source of the upgrade will be responsible for determining the upgradability of the BE being upgraded.

3.2.7.5 Download behavior

Each attempt to upgrade a BE will result in beadm detecting if the latest beadm software from a pkg(5) install image is currently installed. If it is not then beadm will automatically download and install the latest beadm and dependent software before performing the upgrade. If new software is installed then beadm will install the software and reinvoke itself to use the new software.

3.2.7.6 Handling existing BE and Zone configurations

If the BE being upgraded is not in the namespace created by beadm create then beadm will NOT attempt to convert it to the supported namespace mentioned in section 3.2.1 and appropriate message will be displayed.

4.0 Snapshot maintenance

Snapshots will be named and managed by beadm. Snapshots will be created before each pkg(1) transaction. All snapshots created by beadm can be displayed with the list subcommand. The main reason for showing the snapshots beadm produces is so that users can reconcile snapshots created by beadm and others created by ZFS. Their use will be as described throughout this document.

There could be a useful option added to retain snapshots for any failed operation for debugging purposes, however that will not be implemented in this release.

5.0 References

Snap Upgrade project page

http://www.opensolaris.org/os/project/caiman/Snap_Upgrade/

6.0 Issues

The design calls out features that depend on other projects, see Section 2. If those dependencies can't be met then either the features listed need to be re-worked or the project

will deliver a subset of the features described in this document.