ZFS Boot Environments

Sławomir Wojciech Wojtczak
vermaden@interia.pl
vermaden.wordpress.com
twitter.com/vermaden

https://is.gd/BEADM
What is ZFS Boot Environment?

Its bootable clone/snapshot of the working system.
What is ZFS Boot Environment?

Its bootable clone/snapshot of the working system.

- In ZFS terminology its clone of the snapshot.
  
  ZFS dataset → ZFS dataset@snapshot → ZFS clone (origin=dataset@snapshot)
What is ZFS Boot Environment?

Its bootable clone/snapshot of the working system.

- In ZFS terminology its clone of the snapshot.
  
  ZFS dataset → ZFS dataset@snapshot → ZFS clone (origin=dataset@snapshot)

- In ZFS (as everywhere) snapshot is read only.
What is ZFS Boot Environment?

Its bootable clone/snapshot of the working system.

- In ZFS terminology its clone of the snapshot.  
  ZFS dataset → ZFS dataset@snapshot → ZFS clone (origin=dataset@snapshot)

- In ZFS (as everywhere) snapshot is read only.

- In ZFS clone can be mounted read write (and you can boot from it).
What is ZFS Boot Environment?

Its bootable clone/snapshot of the working system.

- In ZFS terminology its clone of the snapshot.
  
  ZFS dataset → ZFS dataset@snapshot → ZFS clone (origin=dataset@snapshot)

- In ZFS (as everywhere) snapshot is read only.

- In ZFS clone can be mounted read write (and you can boot from it).

- The BEs are placed in the pool/ROOT ZFS dataset path.
  
  sys/ROOT/default
  sys/ROOT/pre-upgrade
  (...
Use cases?

Allows bulletproof upgrades/changes to the system.
Use cases?

Allows **bulletproof** upgrades/changes to the system.

- Create **safe failback** ZFS Boot Environment before upgrade or major changes to system.
Use cases?

Allows **bulletproof** upgrades/changes to the system.

- Create **safe failback** ZFS Boot Environment before upgrade or major changes to system.
- Update system **inside** new ZFS Boot Environment without touching running system.
Use cases?

Allows **bulletproof** upgrades/changes to the system.

- Create **safe failback** ZFS Boot Environment before upgrade or major changes to system.
- Update system **inside** new ZFS Boot Environment without touching running system.
- Perform upgrade and test the results **inside FreeBSD Jail**.
Use cases?

Allows **bulletproof** upgrades/changes to the system.

- Create **safe failback** ZFS Boot Environment before upgrade or major changes to system.

- Update system **inside** new ZFS Boot Environment without touching running system.

- Perform upgrade and test the results **inside FreeBSD Jail**.

- **Copy/move** ZFS Boot Environment into another machine.
Use cases?

Allows **bulletproof** upgrades/changes to the system.

- Create **safe failback** ZFS Boot Environment before upgrade or major changes to system.
- Update system **inside** new ZFS Boot Environment without touching running system.
- Perform upgrade and test the results **inside FreeBSD Jail**.
- **Copy/move** ZFS Boot Environment into another machine.
- **Major reconfiguration** (Bareos/Postfix/...).
Can I test and break ZFS BEs without consequence?

Yes you can! Over and over again.

Groundhog Day (1993)
How it was before BEs?

Vendors used *split mirror* or *copying files* to the other/second disk.

**IBM AIX**
- alt_disk_copy
- alt_disk_install
- nimadm
- unmirrorvg
  

**HP-UX**
- lvsplit
- lvmerge
- vgchange
- vgcfgrestore
  

**SUN Solaris Live Upgrade**
- lucreate
- luactivate
- luupgrade
- ludelete
  

**GNU/Linux**
- mdadm
- mirrorlv
- lvconvert
  

---

**Dark ages**

IBM AIX

alt_disk_copy

alt_disk_install

nimadm

unmirrorvg

SUN Solaris

Live Upgrade

lucreate

luactivate

luupgrade

ludelete

HP-UX

lvsplit

lvmerge

vgchange

vgcfgrestore

GNU/Linux

mdadm

mirrorlv

lvconvert

---

**ZFS Boot Environments**

**Polish BSD User Group**

2018/07/30
Mistyped command?

Felling lucky?

Raiders of the Lost Ark (1981)
The beadm command

One simple command – beadm – to create/activate/destroy ZFS Boot Environments.

# beadm
usage:
  beadm activate <beName>
  beadm create [-e nonActiveBe | -e beName@snapshot] <beName>
  beadm create <beName@snapshot>
  beadm destroy [-F] <beName | beName@snapshot>
  beadm list [-a] [-s] [-D] [-H]
  beadm rename <origBeName> <newBeName>
  beadm mount <beName> [mountpoint]
  beadm { umount | unmount } [-f] <beName>
  beadm version
The `beadm` is written in POSIX `/bin/sh`

```bash
#!/bin/sh

# ZFS Boot Environments
# Polish BSD User Group
# 2018/07/30

beadm code

513 (activate) # ------------------------------------------
514 if [ $# -ne 2 ]
515 then
516   _usage
517 fi
518 _be_exist ${POOL}/${BEDS}/${Z}
519 if [ "$BOOTFS" != "${POOL}/${BEDS}/${Z}" ]
520 then
521   echo 'Already activated'
522   exit 0
523 else
524   if _be_mouted ${POOL}/${BEDS}/${Z}
525      then
526        MNT=$( mount | grep -E '^[PPOOL]/^[BEDS]/^[Z] ' | awk '{print $3}' )
527        if [ "${MNT}" != "" ]
528           then
529             # boot environment is not current root and its mounted
530             echo 'Attempt to umount boot environment ${Z} mounted at ${MNT}';
531             umount ${MNT} 1>/dev/null 2>/dev/null
532             then
533               echo 'ERROR: Unable to umount boot environment ${Z} mounted at ${MNT}';
534               echo 'ERROR: Cannot activate manually mounted boot environment ${Z}';
535               exit 1
536           fi
537           echo 'Gracefully umounted boot environment ${Z} from ${MNT} mount point''
538           fi
539       fi
540 # do not change root () mounted boot environment mountpoint
541 HAVE_ZFSBE=0
542 if [ "$ROOTFS" != "$POOL"/$BEDS/${Z} ]
543 then
544   TMPMNT=$( mktemp -d -t BE-)
545   mkdir -p $TMPMNT 1>/dev/null
546   then
547     echo 'ERROR: Cannot create $TMPMNT' directory'
548     exit 1
549   fi
550 MOUNT=0
551 while read FS MNT TYPE OPTS DUMP FSCK;
552 do
553   if [ "$FS" == "$POOL"/$BEDS/${Z} ]
```
Example beadm usage (1/5)

List current BEs and create new one named newbe.

```bash
# beadm list
BE         Active Mountpoint  Space         Created
11.2-RELEASE NR     /            6.3G 2018-05-21 16:01

# beadm create newbe
Created successfully

# beadm list
BE         Active Mountpoint  Space         Created
11.2-RELEASE NR     /            6.3G 2018-05-21 16:01
newbe      -            -          296.0K 2018-07-18 10:04
```
Example `beadm usage (2/5)`

Verify which `snapshot` is used for this `clone` used as `newbe` BE.

```
# beadm list -s
```

<table>
<thead>
<tr>
<th>BE/Dataset/Snapshot</th>
<th>Active</th>
<th>Mountpoint</th>
<th>Space</th>
<th>Created</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.2-RELEASE</td>
<td></td>
<td>/</td>
<td>6.3G</td>
<td>2018-05-21 16:01</td>
</tr>
<tr>
<td>sys/ROOT/11.2-RELEASE</td>
<td>NR</td>
<td>/</td>
<td>6.3G</td>
<td>2018-05-21 16:01</td>
</tr>
<tr>
<td>sys/ROOT/11.2-RELEASE@2018-07-18-10:04:22</td>
<td>-</td>
<td>-</td>
<td>288.0K</td>
<td>2018-07-18 10:04</td>
</tr>
<tr>
<td>newbe</td>
<td></td>
<td></td>
<td>8.0K</td>
<td>2018-07-18 10:04</td>
</tr>
<tr>
<td>sys/ROOT/newbe</td>
<td></td>
<td></td>
<td>8.0K</td>
<td>2018-07-18 10:04</td>
</tr>
<tr>
<td>11.2-RELEASE@2018-07-18-10:04:22</td>
<td>-</td>
<td>-</td>
<td>288.0K</td>
<td>2018-07-18 10:04</td>
</tr>
</tbody>
</table>

```
# zfs get origin sys/ROOT/newbe
```

<table>
<thead>
<tr>
<th>NAME</th>
<th>PROPERTY</th>
<th>VALUE</th>
<th>SOURCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys/ROOT/newbe</td>
<td>origin</td>
<td>sys/ROOT/11.2-RELEASE@2018-07-18-10:04:22</td>
<td>-</td>
</tr>
</tbody>
</table>
Example beadm usage (3/5)

Rename snapshot used for this clone.

```bash
# zfs rename sys/ROOT/11.2-RELEASE@2018-07-18-10:04:22 sys/ROOT/11.2-RELEASE@newbe

# zfs get origin sys/ROOT/newbe
NAME            PROPERTY   VALUE                        SOURCE
sys/ROOT/newbe  origin     sys/ROOT/11.2-RELEASE@newbe -

# beadm list -s
BE/Dataset/Snapshot           Active Mountpoint  Space Created

11.2-RELEASE
sys/ROOT/11.2-RELEASE         NR     /            6.3G 2018-05-21 16:01
sys/ROOT/11.2-RELEASE@newbe  -      -          516.0K 2018-07-18 10:04
newbe
sys/ROOT/newbe                -      -            8.0K 2018-07-18 10:04
11.2-RELEASE@newbe            -      -          516.0K 2018-07-18 10:04
```
Activate the **newbe** BE to be booted after the restart.

```
# beadm list
<table>
<thead>
<tr>
<th>BE</th>
<th>Active</th>
<th>Mountpoint</th>
<th>Space</th>
<th>Created</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.2-RELEASE</td>
<td>NR</td>
<td>/</td>
<td>6.4G</td>
<td>2018-05-21 16:01</td>
</tr>
<tr>
<td>newbe</td>
<td>-</td>
<td>-</td>
<td>68.8M</td>
<td>2018-07-18 10:04</td>
</tr>
</tbody>
</table>
```

```
# beadm activate newbe
Activated successfully
```

```
# beadm list
<table>
<thead>
<tr>
<th>BE</th>
<th>Active</th>
<th>Mountpoint</th>
<th>Space</th>
<th>Created</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.2-RELEASE</td>
<td>N</td>
<td>/</td>
<td>187.5M</td>
<td>2018-05-21 16:01</td>
</tr>
<tr>
<td>newbe</td>
<td>R</td>
<td>-</td>
<td>6.3G</td>
<td>2018-07-18 10:04</td>
</tr>
</tbody>
</table>
```
Remove newbe. It will ask for additional confirmation as we renamed snapshot.

```bash
# beadm list
BE          Active Mountpoint  Space     Created
11.2-RELEASE NR /       6.4G 2018-05-21 16:01
newbe       -         -    68.8M 2018-07-18 10:04

# beadm destroy newbe
Are you sure you want to destroy 'newbe'? Yes
This action cannot be undone (y/[n]): y
Boot environment 'newbe' was created from existing snapshot
Destroy '11.2-RELEASE@newbe' snapshot? (y/[n]): y
Destroyed successfully

# beadm list
BE          Active Mountpoint  Space     Created
11.2-RELEASE NR /       6.4G 2018-05-21 16:01
```
FreeBSD loader integration

Selection of BE at boot is integrated into the FreeBSD loader.
FreeBSD loader integration

The **test** BE is selected to boot instead of the **default** one.
Not just FreeBSD loader ...

Its integrated into other operating systems as well.

- BSDs
  - FreeBSD
  - HardenedBSD (rolling FreeBSD fork)
- Illumos
- OpenIndiana
- OmniOS
**Original not so original ...**

SUN Solaris and Oracle Solaris use GNU GRUB for the BE selection at boot.
What about Linux?

Instructions are fragmented and complicated.
What about Linux?

Instructions are **fragmented and complicated**.

- Only ONE distribution allows root on ZFS install.
  
  Antergos has ZFS option in installer.

  Ubuntu comes with ZFS support but not for root.
What about Linux?

Instructions are **fragmented and complicated**.

- Only ONE distribution allows root on ZFS install.
  - Antergos has ZFS option in installer.
  - Ubuntu comes with ZFS support but not for root.

- Howtos do not use `beadm` command integration.
What about Linux?

Instructions are **fragmented and complicated**.

- Only ONE distribution allows root on ZFS install.
  - Antergos has ZFS option in installer.
  - Ubuntu comes with ZFS support but not for root.

- Howtos do not use `beadm` command integration.

- Howtos are complicated and **VERY** long.
What about Linux?

Instructions are fragmented and complicated.

- Only ONE distribution allows root on ZFS install.
  Antergos has ZFS option in installer.
  Ubuntu comes with ZFS support but not for root.

- Howtos do not use `beadm` command integration.

- Howtos are complicated and VERY long.

- BTRFS alternative with `snapper` on openSUSE/SUSE.
  Red Hat deprecated BTRFS recently.
  Red Hat does not have BTRFS developers.
  Red Hat has lots of XFS developers.
  Fedora and CentOS will follow Red Hat.
What about Linux?

Instructions are fragmented and complicated.

- Only ONE distribution allows root on ZFS install.
  Antergos has ZFS option in installer.
  Ubuntu comes with ZFS support but not for root.

- Howtos do not use `beadm` command integration.

- Howtos are complicated and VERY long.

- BTRFS alternative with `snapper` on openSUSE/SUSE.
  Red Hat deprecated BTRFS recently.
  Red Hat does not have BTRFS developers.
  Red Hat has lots of XFS developers.
  Fedora and CentOS will follow Red Hat.
What about BTRFS?

Can BTRFS Snapshots provide same functionality as ZFS Boot Environments?
What about BTRFS?

Can BTRFS Snapshots provide same functionality as ZFS Boot Environments?

Nope.
What about BTRFS?

Can BTRFS Snapshots provide same functionality as ZFS Boot Environments?

Nope.

Cite from System Recovery and Snapshot Management with Snapper for openSUSE Leap 15 Linux.

- Limitations
  
  A complete system rollback, restoring the complete system to the identical state as it was in when a snapshot was taken, is not possible.
What about Linux?

Too busy rewriting `ip` for `ifconfig` or `ss` for `netstat`?
What about Linux?

Too busy rewriting \texttt{ip} for \texttt{ifconfig} or \texttt{ss} for \texttt{netstat}? ... or \texttt{systemd} for \texttt{init}?
What about Linux?

Too busy rewriting `ip` for `ifconfig` or `ss` for `netstat`? ... or `systemd` for `init`?
Default FreeBSD layout supports ZFS BEs

Default Auto (ZFS) `bsdinstall` option supports ZFS BEs.

```
# zfs list
NAME               USED  AVAIL  REFER  MOUNTPOINT
zroot              339M  8.87G  88K    /zroot
zroot/ROOT         337M  8.87G  88K    none
zroot/ROOT/default 337M  8.87G  337M   /
zroot/tmp          88K   8.87G  88K    /tmp
zroot/usr          352K  8.87G  88K    /usr
zroot/usr/home     88K   8.87G  88K    /usr/home
zroot/usr/ports    88K   8.87G  88K    /usr/ports
zroot/usr/src      88K   8.87G  88K    /usr/src
zroot/var          596K  8.87G  88K    /var
zroot/var/audit    88K   8.87G  88K    /var/audit
zroot/var/crash    88K   8.87G  88K    /var/crash
zroot/var/log      152K  8.87G  152K   /var/log
zroot/var/mail     92K   8.87G  92K    /var/mail
zroot/var/tmp      88K   8.87G  88K    /var/tmp
```
Default FreeBSD layout supports ZFS BEs

The `/usr` and `/var` filesystems have `canmount` property set to `off`.

```
# zfs get -r canmount zroot
NAME                PROPERTY  VALUE     SOURCE
zroot               canmount  on        default
zroot/ROOT          canmount  on        default
zroot/ROOT/default  canmount  noauto    local
zroot/tmp           canmount  on        default
zroot/usr           canmount  off       local
zroot/usr/home      canmount  on        default
zroot/usr/ports     canmount  on        default
zroot/usr/src       canmount  on        default
zroot/var           canmount  off       local
zroot/var/audit     canmount  on        default
zroot/var/crash     canmount  on        default
zroot/var/log       canmount  on        default
zroot/var/mail      canmount  on        default
zroot/var/tmp       canmount  on        default
```
Default FreeBSD layout supports ZFS BEs

This way /usr and /var are placed on the / dataset the zroot/ROOT/default BE.

```
# df -g
Filesystem         1G-blocks Used Avail Capacity Mounted on
zroot/ROOT/default  9    0     8     4%    /
devfs                      0    0     0   100%    /dev
zroot/tmp                  8    0     8     0%    /tmp
zroot/usr/home             8    0     8     0%    /usr/home
zroot/usr/ports            8    0     8     0%    /usr/ports
zroot/usr/src              8    0     8     0%    /usr/src
zroot/var/audit            8    0     8     0%    /var/audit
zroot/var/crash            8    0     8     0%    /var/crash
zroot/var/log              8    0     8     0%    /var/log
zroot/var/mail             8    0     8     0%    /var/mail
zroot/var/tmp              8    0     8     0%    /var/tmp
zroot                      8    0     8     0%    /zroot
```

\[\text{\textleft\Rightarrow\textright} /\text{usr} \& /\text{var}\]
Add beadm to FreeBSD

Just add `beadm` package or install `sysutils/beadm` port ... or download it.

- **Package.**
  
  ```
  # pkg install -y beadm
  ```

- **Port.**
  
  ```
  # make -C /usr/ports/sysutils/beadm install clean
  ```

- **Manual.**
  
  ```
  # fetch https://raw.githubusercontent.com/vermaden/beadm/master/beadm
  # chmod +x beadm
  # ./beadm list
  ```

<table>
<thead>
<tr>
<th>BE</th>
<th>Active Mountpoint</th>
<th>Space</th>
<th>Created</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.2-RELEASE</td>
<td>NR</td>
<td>6.4G</td>
<td>2018-05-21 16:01</td>
</tr>
<tr>
<td>newbe</td>
<td>-</td>
<td>80.2M</td>
<td>2018-07-18 10:04</td>
</tr>
</tbody>
</table>
Using update/upgrade tools with BEs

These tools on FreeBSD are `freebsd-update(8)` and `pkg(8)`.
Using update/upgrade tools with BEs

These tools on FreeBSD are `freebsd-update(8)` and `pkg(8)`.

- On FreeBSD by default these tools **operate on running system**.
Using update/upgrade tools with BEs

These tools on FreeBSD are `freebsd-update(8)` and `pkg(8)`.

- On FreeBSD by default these tools operate on running system.
- By contrast on Solaris/Illumos by default they operate on newly created BE and require reboot into that BE.
Using update/upgrade tools with BEs

These tools on FreeBSD are `freebsd-update(8)` and `pkg(8)`.

- On FreeBSD by default these tools operate on running system.
- By contrast on Solaris/Illumos by default they operate on newly created BE and require reboot into that BE.

**PKG(8)** - [https://man.freebsd.org/pkg](https://man.freebsd.org/pkg)

- `-c` (chroot path), `--chroot` (chroot path)
  pkg will chroot in the (chroot path) environment.
- `-r` (root directory), `--rootdir` (root directory)
  pkg will install all packages within the specified (root directory).

**FREEBSD-UPDATE(8)** - [https://man.freebsd.org/freebsd-update](https://man.freebsd.org/freebsd-update)

- `-b` basedir Operate on a system mounted at basedir. (default: /)
- `-d` workdir Store working files in workdir. (default: /var/db/freebsd-update)
Emulate Solaris/Illumos behaviour on FreeBSD

Example upgrade of packages in the newly created BE for that purpose.

```bash
# beadm create safe
Created successfully

# beadm mount safe
Mounted successfully on '/tmp/BE-safe.ostSai22'

# pkg -r /tmp/BE-safe.ostSai22 update -f
(...)

# pkg -r /tmp/BE-safe.ostSai22 upgrade
(...)

# pkg -r /tmp/BE-safe.ostSai22 info -s feh
feh-2.27.1 438KiB

# pkg -r / info -s feh
feh-2.27 438KiB

# pkg info -s feh
feh-2.27 438KiB
```
Emulate Solaris/Illumos behaviour on FreeBSD

Example fetch security updates in the newly created BE for that purpose.

```bash
# beadm create safe
Created successfully

# beadm mount safe /tmp/safe
Mounted successfully on '/tmp/safe'

# rm -rf /var/db/freebsd-update

# freebsd-update -b /tmp/safe fetch
freebsd-update: Directory does not exist or is not writable: /var/db/freebsd-update

# freebsd-update -b /tmp/safe -d /tmp/safe/var/db/freebsd-update fetch
Looking up update.FreeBSD.org mirrors... 3 mirrors found.
Fetching metadata signature for 11.2-RELEASE from update4.freebsd.org... done.
Fetching metadata index... done.
Inspecting system... done.
Preparing to download files... done.

No updates needed to update system to 11.2-RELEASE-p0.
```
Case where FreeBSD ISO or MEMSTICK boot is required

Deleting `/boot` directory from the BE that is currently set as `bootfs` in your zpool.

```bash
# beadm create test
Created successfully

# beadm list
BE   Active Mountpoint  Space    Created
safe NR     /            6.2G 2018-05-21 16:01
test -      -          220.0K 2018-07-22 01:17

# rm -rf /boot

# reboot
```
Case where FreeBSD ISO or MEMSTICK boot is required

Now **`loader(8)`** does not even show us *Boot Menu* but only following error message.

```
\Can’t find /boot/zfsloader

FreeBSD/x86 boot
Default: zroot/ROOT/safe:/boot/kernel/kernel
boot: zroot/ROOT/test:/boot/kernel/kernel/
int=00000006 err=00000000 efl=00010056 eip=00000003
eax=fd310000 ebx=00310000 ecx=a0100001 edx=00026948
esi=0001e5f0 edi=00094768 ebp=000949d0 esp=00310000

cs=0008 ds=0010 es=0010 fs=0010 gs=0010 ss=0010

cs:eip=f0 53 ff 00 f0 53 ff 00-f0 53 ff 00 f0 53 ff 00
  f0 53 ff 00 f0 cc e9 00-f0 53 ff 00 f0 a5 fe 00

ss:esp=00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00
  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

BTX halted
```

- Bug already submitted - [https://bugs.freebsd.org/bugzilla/show_bug.cgi?id=229926](https://bugs.freebsd.org/bugzilla/show_bug.cgi?id=229926)
Case where FreeBSD ISO or MEMSTICK boot is required

Workaround that brings system back to normal functional state.

- Boot broken system from FreeBSD ISO or MEMSTICK image and select `<Live CD>` option.
  
  ```
  login: root
  (...)
  root@:~ #
  ```

- Forcefully import the ZFS pool.
  
  ```
  root@:~ # zpool import -f zroot
  ```

- Set `bootfs` manurally for the ZFS pool to created backup BE - `zroot/ROOT/test` in our case.
  
  ```
  root@:~ # zpool set bootfs=zroot/ROOT/test zroot
  ```

- Reboot system without FreeBSD ISO or MEMSTICK image and it will boot as usual with `Boot Menu`.
  
  ```
  root@:~ # reboot
  ```
First one was `manageBE` script which had some problems and complicated syntax.

- Create new BE.
  ```
  # manageBE create -n 9_20120321 -s 9_20120317 -p zroot
  manageBE: cannot create /zroot/ROOT/9_20120321/boot/loader.conf: No such file or directory
  manageBE: cannot create /zroot/ROOT/9_20120321/etc/fstab: No such file or directory
  The new Boot-Environment is ready to be updated and/or activated.
  ```

- Listing existing BEs.
  ```
  # manageBE list
  Poolname: zroot
  BE                        Active Active Mountpoint                      Space  Used by BE snapshots: 1.99G
  Name                      Now    Reboot -                               Used
  ----                      ------ ------ ----------                      -----  
  9_20120321                no     no     /ROOT/9_20120321                 145M
  9_20120317                yes    yes    /                               1.59G
  ```
Current upstream `beadm` source and alternatives/forks.
Current upstream `beadm` source and alternatives/forks.

- The `manageBE` source - [https://outpost.h3q.com/patches/manageBE/manageBE](https://outpost.h3q.com/patches/manageBE/manageBE)
Current upstream **beadm** source and alternatives/forks.

- The **manageBE** source - [https://outpost.h3q.com/patches/manageBE/manageBE](https://outpost.h3q.com/patches/manageBE/manageBE)

- Current **beadm** implementation - [https://github.com/vermaden/beadm](https://github.com/vermaden/beadm) ➞ source for **beadm** package
  
  - Fork with separate boot pool support - [https://bitbucket.org/aasoft/beadm](https://bitbucket.org/aasoft/beadm) ➞ fork of **vermaden/beadm**
  
  - Fork with support for Linux system - [https://github.com/b333z/beadm](https://github.com/b333z/beadm) ➞ fork of **vermaden/beadm**

- Original **HOWTO: FreeBSD ZFS Madness** thread - [https://forums.freebsd.org/threads/31662/](https://forums.freebsd.org/threads/31662/)
History/Mods/Forks/Alternatives

Current upstream **beadm** source and alternatives/forks.

- The **manageBE** source - [https://outpost.h3q.com/patches/manageBE/manageBE](https://outpost.h3q.com/patches/manageBE/manageBE)

- Current **beadm** implementation - [https://github.com/vermaden/beadm](https://github.com/vermaden/beadm) ➞ source for **beadm** package
  - Fork with separate boot pool support - [https://bitbucket.org/aasoft/beadm](https://bitbucket.org/aasoft/beadm) ➞ fork of **vermaden/beadm**
  - Fork with support for Linux system - [https://github.com/b333z/beadm](https://github.com/b333z/beadm) ➞ fork of **vermaden/beadm**
  - Original **HOWTO: FreeBSD ZFS Madness** thread - [https://forums.freebsd.org/threads/31662/](https://forums.freebsd.org/threads/31662/)

- The **zedenv** in Python 3.6 with support for FreeBSD and Linux - [https://github.com/johnramsden/zedenv](https://github.com/johnramsden/zedenv)
  - Currently at alpha stage of development (experimental) - not production ready.
  - Needs **python36** and **py36-setuptools** packages to work.
  - Supports plugins but currently comparable with **beadm** features or its forks.
History/Mods/Forks/Alternatives

Current upstream `beadm` source and alternatives/forks.

- The `manageBE` source - [https://outpost.h3q.com/patches/manageBE/manageBE](https://outpost.h3q.com/patches/manageBE/manageBE)

- Current `beadm` implementation - [https://github.com/vermaden/beadm](https://github.com/vermaden/beadm) ==> source for `beadm` package
  - Fork with separate boot pool support - [https://bitbucket.org/aasoft/beadm](https://bitbucket.org/aasoft/beadm) ==> fork of `vermaden/beadm`
  - Fork with support for Linux system - [https://github.com/b333z/beadm](https://github.com/b333z/beadm) ==> fork of `vermaden/beadm`
  - Original HOWTO: FreeBSD ZFS Madness thread - [https://forums.freebsd.org/threads/31662/](https://forums.freebsd.org/threads/31662/)

- The `zedenv` in Python 3.6 with support for FreeBSD and Linux - [https://github.com/johnramsden/zedenv](https://github.com/johnramsden/zedenv)
  - Currently at alpha stage of development (experimental) - not production ready.
  - Needs `python36` and `py36-setuptools` packages to work.
  - Supports plugins but currently comparable with `beadm` features or its forks.

- Ansible `beadm` module - [https://docs.ansible.com/ansible/latest/modules/beadm_module.html](https://docs.ansible.com/ansible/latest/modules/beadm_module.html)
Questions?

Sławomir Wojciech Wojtczak
vermaden@interia.pl
vermaden.wordpress.com
twitter.com/vermaden

https://is.gd/BEADM
Thank You!

Sławomir Wojciech Wojtczak
vermaden@interia.pl
vermaden.wordpress.com
twitter.com/vermaden

https://is.gd/BEADM