System Administration

Contents

1 Accounts **4**
- 1.1 User accounts . . . 4
- 1.2 System (or Service) accounts . . . 5
  - 1.2.1 Superuser account . . . 5
  - 1.2.2 Roles (and Role-Based Access Control) . . . 9
- 1.3 Groups . . . 10
- 1.4 Managing accounts . . . 10
  - 1.4.1 Examples . . . 11

2 Software management **12**
- 2.1 Example . . . 13
- 2.2 Common Idioms . . . 15
  - 2.2.1 Find the name of a package to which a file on the system belongs . . . 15
- 2.3 Publishers . . . 15
- 2.4 References . . . 15
- 2.5 Active Directory Integration . . . 15
  - 2.5.1 Introduction . . . 15
  - 2.5.2 Native AD integration . . . 16
  - 2.5.3 Kerberos and LDAP . . . 16
  - 2.5.4 winbind . . . 16

3 Management of System Resources **16**
- 3.1 Basic system information . . . 16
  - 3.1.1 System processes . . . 16
  - 3.1.2 Disk usage . . . 17
  - 3.1.3 Largest files in a directory . . . 17
  - 3.1.4 Who is logged on to the system . . . 17
  - 3.1.5 List all software packages installed on the system . . . 17
- 3.2 System shutdown, reboot, ... 17

4 Updating All Packages (Upgrading Your System) **18**

5 Service Management Facility **18**
- 5.1 SMF Basic Commands . . . 19
  - 5.1.1 svcs . . . 19
  - 5.1.2 svcadm . . . 21
  - 5.1.3 svccfg . . . 22

5.1.4 svcprop . . . 22
5.1.5 inetadm . . . 22
5.1.6 inetconv . . . 22
5.2 Troubleshooting . . . 22
5.2.1 Basic Strategy . . . 22
5.2.2 Corrupt Service Configuration: Revert to an Old Configuration . . . 24
5.2.3 Boot Failure . . . 24
5.2.4 Maintenance Mode . . . 24

6 Configuring and Tuning 24
6.1 Configuring a UPS . . . 25
6.2 Fault management (FMA) . . . 25
6.3 Virtual Terminals/Consoles (VT) . . . 25
6.4 Service management (SMF) . . . 26
6.5 Systems logging and monitoring . . . 26

7 Illumos boot process 27

8 Security 27

9 Zones 27
9.1 Zone networking model . . . 28
9.2 Quick Setup Example . . . 28
9.3 System repository configuration . . . 30
9.4 Troubleshooting . . . 31
9.4.1 Fixing zone installation issues . . . 31

10 Storage 31
10.1 Mounting file systems . . . 31
10.1.1 Mounting and Unmounting ISO images . . . 32
10.1.2 Mounting NTFS Volumes - 3rd party support . . . 32
10.2 Configuring OpenIndiana as an ISCSI Target Server - (COMSTAR) . . . 33
10.3 System backups . . . 33
10.4 ZFS . . . 33
10.4.1 Importing ZFS disks . . . 33
10.4.2 How does one mirror their root zpool? . . . 34
10.4.3 How does one create additional zpools? . . . 34
10.4.4 Modifying zpool settings and attributes . . . 34
10.4.5 Modifying zfs file system settings and attributes . . . 34
10.4.6 How does one create additional zfs datasets? . . . 34
10.4.7 Configuring system swap . . . 34

11 Virtualization 35
11.1 OpenIndiana as a virtualization host server . . . 35

12 Localization 35

13 Dtrace 35

14 Configuring Networking 35
14.1 Automatic Configuration (NWAM) . . . 35

14.2 Changing from NWAM to Manual Configuration 36
14.3 More on Automatic Configuration (NWAM) 37
14.3.1 Using NWAM configuration tools 37
14.3.2 Network automagic online help 41
14.3.3 Troubleshooting NWAM 41

15 Clustering with Open HA Cluster 41

Once OpenIndiana has been installed, the system will require monitoring to ensure smooth operation. Software will periodically have to be updated, redundant software removed, new users added to the system, etc. All these activities and many more are referred to as system administration.

Basic system administration can be reduced to a number of common tasks:

- Accounts
- Software management
- System monitoring
- Virtualisation
- Network configuration
- Network file systems (NFS)

While it is certainly possible to add more to this list or select alternative items, this small selection is readily absorbed and is convenient to illustrate a number of essential concepts central to OpenIndiana system administration.

1 Accounts

There are different types of accounts provided by OpenIndiana:

- User accounts
- System (or Service) accounts
- Superuser account
- Roles (Role-Based Access Control)

It is also possible to *group* users together:

- Groups

1.1 User accounts

OpenIndiana supports multiple users to work on the same computer at the same time. While only one person can sit in front of the monitor and keyboard, many users can be remotely logged onto the machine and all work simultaneously on it. To use the system, a user requires an account, which are known as *user accounts*.

A user account facilitates interactive access to the system and is primarily used for day-to-day tasks. Each user requiring access to the system should have a unique account specifically assigned to that user. System administrators can thereby monitor user activity and establish system dignostics to optimise system performance. Moreover permission to perform various system activities can be separately assigned to each user on an individual basis.

When the system is initially installed, usually only one user account is created. Additional accounts can be added later.

Each user account has a number of attributes associated with it:

- **UID** (user ID) a unique number which is the account ID on the system. The UID is limited to a number less than 2147483647. However, for compatibility reasons, a number above 65535 should not be used.
- **GID** (group ID) a unique number designating the primary group of the account.

- **username** the name of the account, which is frequently displayed on the login screen, depending on the login display manager in use. The username format is briefly described in passwd(4)
- **password** the password required to access the account. *Passwords are used for security concerns. Accounts without a password should are a security hazard.*
- **comment** (also referred to as gecos) - account description This is often the user's real name.
- **home directory** the path to the directory which is *home* to the user and where the user is placed by default after logging onto the system.
- **login shell** user's initial shell program.
- **password last change time** time at which the the user password was last changed.
- password aging information - several shadow(4) fields determining when password should be changed
- **min** the minimum number of days required between password changes;
- **max** the maximum number of days the password is valid;
- **warn** the number of days before password expires that the user is warned.

1.2 System (or Service) accounts

System accounts, including the root account, are designed to be used for administrative purposes. While all such tasks can be performed using the root account, system accounts all have limited powers with respect to root, thereby delegating only those powers necessary to carry out an administrative task to a particular system account. System accounts often supply a specific function such as managing mail accounts or services to the network such as DNS, SMTP or WWW.

All system accounts have a `UID` less than 100.

All system accounts – the one notable exception being root – are supplied with either `LK` or `NP` in the second column of the `/etc/shadow` file. In other words, it is possible to obtain a list of all system accounts:

```
# more /etc/shadow | grep -e NP -e LK | cut -d: -f1
```

Service account	Purpose
`nobody`	for services that require no privileges
`pkg5srv`	for the `pkg(5)` server
`webservd`	for web servers
…	

1.2.1 Superuser account

Command	Purpose
su(1M)	switch user

Although the superuser account is a system account, its unique features and importance justify a section solely devoted to it.

Every UNIX-like system has one superuser account, named `root`, used for system administrative tasks. This account has UID 0.

It is highly inadvisable to use this account for day-to-day usage such as browsing the web, reading emails or to watch movies. The root account is not limited to any restrictions and any vulnerability while operating as root can cause serious damage to the system.

If a user is created during the OI installation process, then this user is assigned the root role, while the root account is not created by the system. This means that it is not possible to directly log into a system using the root account as it does not exist. It is, however, possible to log in as the user created during the installation and then switch to root using su(1M).

However, if a user is not created during the OI installation, then the root account is created as a regular account and it is possible to directly log in to the root account. Note, that even when root is created as a role, it is possible to use this account directly to log into the system when it is booted in single-user mode.

1.2.1.1 SUperuser DO: sudo(1m)

Tool	Purpose
`sudo(1m)`	execute a command as superuser

The `sudo` command, i.e., superuser do, permits a regular user to execute a specified set of commands with superuser privileges without having to become the superuser. While `sudo` actually allows one user to execute a command as another user, it is predominantly used to allow a user to execute commands requiring elevated privileges as the superuser. It is not always feasible for one user to perform all administrative tasks. It would be more flexible if some tasks could be performed by some, say, experienced users. To enable some users to carry out a command with root privileges, or to *do* an administrative command `sudo(1m)` can be used. `sudo` can be configured to provide a user with elevated privileges for *all* commands requiring heightened privileges or configured to make only a select number of commands available to a user. Furthermore, in addition to being able to equip users with permission to perform some actions, `sudo` can be applied to a `group`. The group can be assigned to users who are then elevated to the powers available to that group.

One common strategy is to to define a minimum set of commands to perform some task, e.g., manage a mailserver, and then assign these commands to a group. Thereby, mailserver management can be assigned to a user by adding the group membership to a user.

While elevated user privileges for `sudo` are configured in `/etc/sudoers`, user configuration is preferably performed by adding files to the `/etc/sudoers.d/` directory. All files in this directory will be automatically added for use by `sudo`. At a glance, any experienced administrator can view system changes to an installation; and system re-installation or migration becomes less ardent.

1.2.1.1.1 Sudo configuration

Tool	Purpose
`sudoers(4)`	sudo security policy

Tool	Purpose
`/etc/sudoers`	system file containing a list of users and groups allowed to use sudo
`/etc/sudoers.d/`	directory containing list of users and groups allowed to use sudo as added by system administrator
`visudo(1m)`	wrapper framework used to edit sudo configuration files

Elevated user privileges for `sudo` are configured in `/etc/sudoers`. This file contains a list of users and groups allowed to use `sudo`, and the commands each user, or group, are allowed to use with `sudo`.

However, it is well advised to preserve this file in its pristine state upon installation. Modifications intended for this file are preferably carried out by creating a new file and making the changes in this new file. Add the new file to the `/etc/sudoers.d/` directory. All files in this directory will be automatically assigned for use by `sudo`.

To edit any `sudo` configuration files is subject to subtle syntax pitfalls. It is also dangerous for more than one user to edit this file at the same time. To prevent such mishaps, `visudo` was designed to assist in avoiding such issues. `visudo` works in conjunction with an editor, by default `vi`.

A standard installation of OpenIndian supplies an easy to use editor called `pluma`. To start `visudo` with it using `pluma`, do the following:

```
$ sudo EDITOR=pluma visudo /etc/sudoers.d/whatever
```

This will start `pluma` to edit `whatever` with the `visudo` wrapper.

It is also possible to set the default editor in your login profile, e.g., for BASH: add `export EDITOR=pluma` to `$HOME/.bashrc` and start `visudo` as follows:

```
$ sudo -E visudo /etc/sudoers.d/whatever
```

The `-E` option permits passing environment variables from the user to the `sudo` environment, which is not always guaranteed. The `EDITOR` environment variable can be set in your init profile, e.g., `.bashrc` if you are using the BASH shell.

Note: this depends on how `sudo` was compiled. Currently, the version of `sudo` packaged with OpenIndiana supports the above mechanism. Details can be found in the manpage.

1.2.1.1.2 Sudoers file format

```
[user] [host]([run-as-user]:[run-as-group]) [command1, command2]
```

`user` and `host` are separated by a tab.

- **user:** is either a username or the name of a group. the name of a group is preceded an '%'.
- **host:** the name of one or more machines from which the commands are permitted to run.

- **run-as-user:** the name of a user by which the commands are permitted to be run.
- **run-as-group:** the name of a group which must be assigned to a user by which the commands are permitted to be run.
- **commands:** list of commands separated by commas permitted to be executed

The sudoers file supports a rich collection of features beyond this cursory treatment. Consult the `sudoers(1)` manpage, in particular the section titled 'Sudoers File Format' for details.

1.2.1.1.3 Example of a Sudoers file

```
# Sample sudo configuration file
#
# User root is assigned all privileges
root  ALL=(ALL:ALL) ALL
# Assign all privileges to group admin. Any user assigned to admin has all privileges
%admin ALL=(ALL) ALL
...
```

To illustrate, we provide the following example.

1.2.1.1.4 Example: *Problem* To shutdown the system, elevated privileges are required. If a standard user issues the `shutdown` command, the following occurs:

```
$ sudo shutdown -i5 -g0 -y
/usr/sbin/shutdown:    Only root can run /usr/sbin/shutdown
```

In this example, we would like to provide one or more users with the possibility of doing the following:

- the ability to run `/usr/sbin/shutdown` including the ability to specify command line arguments
- the ability to run only `/usr/sbin/reboot`, without being allowed to specify any command line arguments
- the ability to run all commands in `/opt/groups/offclub/bin`, a directory specifically created for this group by an administrator.

Solution Strategy:

- create a group called `offclub`
- create a file called `/etc/sudoers.d/offclub`
- edit `/etc/sudoers.d/offclub` and allow any member of group `offclub` to issue `/usr/sbin/shutdown` and `/usr/sbin/reboot`
- create `/opt/groups/offclub/bin`.

We are going to create a group, `offclub` and all members of the offclub group to issue the required commands with relevant privileges. Moreover, for demonstrative purposes rather less than for security concerns, we will create a directory and all members of this group will be allowed to run any command in this directory.

Generate the group. The group name is restricted to a maximum of 8 lower case characters and create the directory which contains command that can be executed by members of the `offclub` group:

```
# groupadd offclub
# sudo mkdir -p /opt/groups/offclub/bin
```

```
$ visudo /etc/sudoers.d/offclub
```

Add the following line:

```
%offclub    ALL=(root)    /opt/groups/offclub/bin, /usr/sbin/reboot,
 ↪  /usr/sbin/shutdown
```

The `%` indicates that `offclub` is a group. You can, off course, add users instead of groups here. The final three indicate the commands, or directory; while ALL=(root) indicates that root privileges are bestowed for these commands.

There are a number of ways of enabling permission to execute files in a directory. One way is to assign a file, say *myapp* to the group, and provide the group permission to execute *myapp':

```
# chown offclub /opt/groups/offgroup/bin/myapp
# chmod g+x /opt/groups/offgroup/bin/myapp
```

All members of the `offclub` can now perform the commands. Example, add user *whoever* to the group:

```
usermod -G offclub whoever
```

1.2.2 Roles (and Role-Based Access Control)

Tool	Purpose
roles(1)	print roles assigned to a user
auths(1)	print authorisations assigned to a user
privileges(5)	
ppriv(1)	manage process privileges
profiles(1)	print user profiles

A role is a basic unit of Role-Based access control (RBAC) or set of privileges(5) a user account can assume. A role can be thought of as a no-login account. It has most of the attributes of a normal user account and is identified as a normal user; but it is permissible to log into such an account directly. One should login using a regular account and use su(1M) to assume the role.

1.2.2.1 Role-Based Access Control (RBAC) The *all-or-nothing* power assigned to the root user has its obvious limitations. While sudo is an improvement by limiting root privileges for only several commands, more granular control is often desired.

One improvement on the above systems would be one in which privileges could be assigned on a more fine-grained and selective basis; whereby the focus is less on commands and more on performing actions to accomplish some task.

Imagine a user assigned the task of administrating some particular hardware, for example, printers attached to the system. A more desirable system would be one in which this user had the ability to permit users to use a printing device, remove print jobs from the print spool

or add new printers to the system. Moreover, it would be advantageous if it were possible to assign privileges to perform only these actions and none other.

RBAC was developed to accomplish this.

1.2.2.1.1 ProFile EXECute: pfexec(1)

Tool	Purpose
pfexec(1)	

1.3 Groups

Command	Operand	Purpose
groups(1)	[user]	list groups assigned to a user

A group is a collection of users which is predominantly used to assign privileges to users.

To obtain a list of groups assigned to a user:

```
groups [user]
```

Get a list of all groups on the system:

```
$ getent group | cut -d: -f1
```

1.4 Managing accounts

Tool	Description
useradd(1M)	create a new account on the system
userdel(1M)	delete an account from the system
usermod(1M)	modify account information
groupadd(1M)	create a group on the system
groupdel(1M)	delete a group from the system
groupmod(1M)	modify group information on the system
roleadd(1M)	create a new role on the system
roledel(1M)	delete a role from the system
rolemod(1M)	modify role information in the system
roles(1)	print roles assigned to a user
passwd(1)	change login password and password attributes
id(8)	print user and groups IDs

- **useradd** creates a user account on the system. Attributes can also be set while an account is created such as a comment, group membership, home directory path, UID number of the account or the login shell.

- **userdel** is used to remove an account from the system.
- **usermod** is used to modify properties of an existing account.
- **groupadd** creates a new group on the system by adding appropriate information to the /etc/group file.
- **groupdel** deletes a group from the system.
- **groupmod** is used to modify group attributes.
- **roleadd** is used create roles in the system.
- **roledel** deletes a role from the system.
- **rolemod** modifies role information on the system.
- **roles**
- **passwd** is used to change the password of a user account. An unprivileged user may only change the password of that user's account, whereas the superuser may change the password for any account. A privileged user can also use `paasswd` to change login password attributes (such as expiration date) or can lock an account.

1.4.1 Examples

1.4.1.1 Use sudo to enable a user to shutdown the system The root user or a user with sudo enabled can shutdown the system.

1.4.1.2 Use RBAC to enable a user to shutdown the system We can use RBAC to enable a user to be able to shutdown the system. However, we can create a role that allows only the privilege to shutdown the system, and no additional privileges. We can then assign this role to one or several users.

- assign a privilege to a role to shutdown the system

```
# roleadd shutdown
```

- Assign a password

```
# passwd shutdown
```

- Assign this role to a user

```
# usermod -R shutdown whoever
```

- Create a SHUTDOWN profile

```
# echo "SHUTDOWN:::profile to shutdown:help=shutdown.html" >>
 ↪  /etc/security/prof_attr
```

- Okay, now assign the role profile SHUTDOWN to the role shutdown

```
# rolemod -P SHUTDOWN shutdown
```

- Assign some administrative command to profile

```
# echo "SHUTDOWN:suser:cmd:::/usr/sbin/shutdown:uid=0" >> /etc/security/exec_attr
```

- Use it

```
$ su shutdown
# shutdown -i5 -g0 -y
```

Now user *whoever* can shutdown the system.

The `pfexec` command is more flexible in the number of privileges that can be assigned to a user.

2 Software management

OpenIndiana uses the Image Packaging System, *IPS*, to manage software packages. IPS supports installing, updating and searching OpenIndiana software repositories. It also allows users to create and publish software packages to provide new software for other users. Its features are vast and we only present the *essentials* here.

Software packages are *published* to a publisher. An IPS software package consists of files, directories, links and dependencies to other packages.

The user interface to install, update, query, generate, etc packages on a publisher is `pkg`:

pkg(1M) is the client for **I**mage **P**acking **S**ystem.

Usage:

```
pkg [options] command [cmd_options] [operands]
```

Command	cmd_options	operands	Purpose
`help`			list all commands
		command	command usage
`publisher`			list package repositories (known as *publisher* on OI)
		a_publisher	list details on *a_publisher*
`set-publisher`			add a publisher
`unset-publisher`			Remove a publisher
`refresh`			update local package catalogue with new packages
`update`			update all packages to the latest versions
		a_package	only update *a_package* to the latest version
`list`			list packages installed on the system
	`-a`		also list non-installed packages from publishers
`contents`			list files in a package installed on the system
	`-r`		list files on remote (publisher) package
`info`			get information on all packages on the system
		a_package	get information on a particular package on the system
	`-r`	*a_package*	get information on a remote package
`search`			**search for a package on the system**
	`-r`		**search for a package not on the system**

Command	cmd_options	operands	Purpose
`install`		*a_package*	install *a_package* onto the current boot environment
	`--be-name` *name*	*a_package*	install on a new boot environment called *name*.The current be remains unchanged.
`uninstall`		*a_package*	remove a package from the system
`revert`			restore files to their original installed states
		/the/path/file	restore *file* located in /the/path/ to its original state
`verify`			verify all packages. Only report with a message on error
		a_package	verify only *a_package*
`fix`			fix any errors detected by `verify`
		a_package	fix any errors detected by `verify` for *a_package*

- `help`: Without any commands will provide a list of all commands supported by `pkg`. When followed by a command, usage on that command will be provided.
- `publisher`: a package repository is known as a *publisher* on OI. A publisher contains one or more packages that can be queried or installed by `pkg` using an appropriate pkg command. A list of all publishers configured to be used by `pkg` can be obtained by using this command without any options. By supplying a publisher, details of the publisher will be provided.
- `set-publisher`: add a new publisher to the user's list of publishers
- `refresh`: the local system has a catalogue of all packages provided by all publishers. Using this command without any operands will update the catalogue for all publishers configured by the user. By supplying a publisher, only packages provided by that publisher will be updated.
- `list`: by default will list installed packages. Use `-a` to also list packages in publishers.
- `contents`: this command lists files in a package that is currently installed. Use `-r` to list files in packages not installed on a *r*emote publisher.
- `info`: by default provides information on installed packages. Add the `-r` option for information from publishers.
- `install`: by default, a package is installed to the current boot environment. It is possible to install to a boot environment, leaving the current boot environment untouched. For example, to create a clone of the current boot environment, and install the package to the new boot environment, use the `--be-name` option. This is the recommended way to install new packages.
- `search`: by default queries only installed packages, `-r` extends to publishers. Support for wildcards *?* and * in the query.
- `revert`: this will adjust a software package to that state it originally had after installation, including configuration files and file permissions.

2.1 Example

To illustrate, look for, install and validate a package called `ffmpeg`, which is a video and audio processing application.

```
$ pkg refresh && pkg update
```

```
$ pkg list -a | grep -i ffmpeg
$
```

The result of running this command is empty: nothing found. Post a question to the OI community or ask on IRC About OpenIndiana, How can I contact OpenIndiana community. Assuming someone informs you that the package does exit and can be found on the publisher *hipster-encumbered* with address *http://pkg.openindiana.org/hipster/* then you can continue.

Add the *hipster-encumbered* publisher and try again:

```
# sudo pkg set-publisher -G'*' -g http://pkg.openindiana.org/hipster
 ↪  hipster-encumbered
$ pkg publisher
PUBLISHER                    TYPE     STATUS P LOCATION
userland                     origin   online F
 ↪  file:///export/home/benn/devl/oi-userland/i386/repo/
openindiana.org (non-sticky) origin   online F http://pkg.openindiana.org/hipster/
hipster-encumbered              origin   online F
 ↪  https://pkg.openindiana.org/hipster-encumbered/
```

So *hipster-encumbered* has been added as the last publisher to our list of publishers. Obtain information on the publisher:

```
$  pkg publisher hipster-encumbered


        Publisher: hipster-encumbered
            Alias:
       Origin URI: https://pkg.openindiana.org/hipster-encumbered/
    Origin Status: Online
          SSL Key: None
         SSL Cert: None
      Client UUID: 7501a944-6bce-11ec-893c-880027010518
  Catalog Updated: December 21, 2021 at 10:58:33 PM
          Enabled: Yes
```

So the publisher is online, i.e., is up and ready to accept requests.

Search again for out package:

```
$ pkg list -a | grep -i ffmpeg
video/ffmpeg (hipster-encumbered)                4.4.1-2020.0.1.0          ---
$ pkg info -r video/ffmpeg
Name: video/ffmpeg
          Summary: A very fast video and audio converter
         Category: System/Multimedia Libraries
            State: Installed
        Publisher: hipster-encumbered
          Version: 4.4.1
           Branch: 2020.0.1.0
   Packaging Date: November  9, 2021 at 01:39:02 PM
Last Install Time: January  2, 2022 at 01:34:22 PM
             Size: 58.88 MB
```

```
            FMRI: pkg://hipster-encumbered/video/ffmpeg@4.4.1-2020.0.1.0:20211109T
             ↪  133902Z
       Source URL: https://www.ffmpeg.org/releases/ffmpeg-4.4.1.tar.bz2
      Project URL: https://www.ffmpeg.org/
```

We can list all files in the package with `pkg contents -r video/ffpmeg`. Finally, install:

```
# pkg install video/ffmpeg
$ pkg verify pkg://hipster-encumbered/video/ffmpeg@4.4.1-2020.0.1.0:20211109T133902Z
```

2.2 Common Idioms

2.2.1 Find the name of a package to which a file on the system belongs

```
$ pkg search -o pkg.name -l path:*ffmpeg
PKG.NAME
video/ffmpeg
```

2.3 Publishers

Publisher	Location	Notes
`openindiana.org`	`http://pkg.openindiana.org/hipster/`	Primary OI publisher
`hipster-encumbered`	`https://pkg.openindiana.org/hipster-encumbered/`	many multimedia packages
`localhostoih`	`https://sfe.opencsw.org/localhostoih`	experimental, be careful with version conflicts

2.4 References

This section provides the fundamentals of `pkg`. More detailed information is available here:

- Getting Started, section: Image Package System (IPS) for details.
- OpenSolaris 2009.06 Image Packaging System Guide is a thorough treatment.

2.5 Active Directory Integration

2.5.1 Introduction

There are at least three different possible approaches for Active Directory authentication and each has its pros and cons.

1. Use the new native AD integration with **idmap**, **nss_ad** and **kclient**, this will work with CIFS and NFS out of the box.
2. Use Kerberos and LDAP (**kclient**, **ldapclient**, **pam_krb5** and **nss_ldap**).
3. Use **windbind** (**pam_winbind** and **nss_winbind**).

2.5.2 Native AD integration

- Pro: Fully integrated and native tools only
- Cons: Doesn't work for UNIX services other than CIFS and NFS. The ephemeral id mapping strategy supposedly wasn't designed for other UNIX services. As a result several problems arrise, one of them is that the mappings aren't constant over the lifetime of UNIX processes which severly breaks UNIX semantics. Depending on your UNIX service you will see unexpected results or even process crashing.

In order to use this approach with any UNIX service (eg **FTP**) you need to enable *directory-based name mapping* and install **IDMU** (Identity Management for UNIX) on the AD server.

Refer to the original documentation from Oracle for getting this working: nss_ad, CIFS.

2.5.3 Kerberos and LDAP

- Pro: Fully integrated and native tools only
- Cons: Requires installation of additional role services (IDMU, Identity Management for UNIX) on the Active Directory side

> ⓘ **NOTE:**
> On the Wiki the 'Kerberos and LDAP' page was a separate detailed article on configuration of Windows Server 2008 Active Directory to work with OI. As of 2021, Windows Server 2008 is EoL. This section should not be migrated until it has been checked and updated for recent Windows. Ideally this information should be on a separate page (potentially community contributions) as it's large and contains a lot of Windows information.

2.5.4 winbind

- Pro: Easy setup, no AD modification
- Cons: Depends on 3rd party software (Samba), group membership resolution didn't work

> ⓘ **NOTE:**
> As above note, the 'winbind' page was a separate detailed article which needs to be checked and updated before it's migrated.

3 Management of System Resources

3.1 Basic system information

3.1.1 System processes

```
$ prstat
```

This command provides a host of information on all processes running on the system. Some of the information provided is as follows:

- percentage of CPU used by each process
- amount of memory consumed by each process
- unique id of each process (which can, for example, be used to stop the process)

3.1.2 Disk usage

```
# df -h
```

Provides information on disk size, amount of space used and available free space for all attached storage devices. The `-h` option reports this information in human readable format.

3.1.3 Largest files in a directory

Go to the directory using the `cd` command and issue the following command:

```
$ du | sort -n
```

This will list the size of each file in the current directory and all sub-directories, starting with the smallest up to the largest files.

3.1.4 Who is logged on to the system

```
$ listusers
```

3.1.5 List all software packages installed on the system

```
$ pkg list
```

3.2 System shutdown, reboot, …

OpenIndiana defines a number of different system states known as run-levels. You can change from one system state to another by using the `shutdown` command and specify the run-level using the `i` option. You can always determine the run-level via `who -r`.

You must be root or have root privileges (e.g., using `sudo`) to send the system into a different state, i.e., turn off, reboot, etc. Shutdown and turn off all hardware (if supported by the hardware) now:

```
# shutdown -i5 -g0 -y
```

Changing the run-level of the system can be disruptive to other users currently using the system. Thus, it is always wise to establish who is currently logged onto the system before changing the run-level.

- `-i [run-level]` is used to specify the run-level. This is either a digit or a single letter. Here are some run-levels available:
- `5` stop all system services, and turns off hardware devices, etc.
- `6` reboot the system.
- `1` single-user mode. Primarily used for system maintenance.
- `S` single-user mode where only a command line terminal is available.
- `-g [seconds]` is used to specify the number of seconds after which to commence shutting down services. `0` immediately intiates shutting down all services.

- `-y` automatically answers all system questions with 'yes'. The shutdown process is not interrupted by system prompts requiring user-interactive intervention.

4 Updating All Packages (Upgrading Your System)

OpenIndiana Hipster is a rolling release distribution, so to upgrade your system, you simply run `pkg update` all packages will be updated to the latest revision:

```
pfexec pkg update
```

(Note: for `pfexec` to succeed you'll need to have the `Software Installation` profile enabled for your user account.)

New installation images are beneficial for those with newer hardware that illumos may have added support for, but the above command should suffice for existing installations.

5 Service Management Facility

Services are an extension to processes. The administration of services is carried out on OpenIndian using *Service Management Facility*, commonly known as *SMF*.

As application become more complex, their demands on system resources and peripheral resource increase. Modern applications often require more resource to provide more services. Resources such as an internet connection, a web server running in the background, a databank up and running in an appropriate state or mode of operation are increasingly found to be the prerequisite of modern applications. Although the application may provide users with a single service, the application might require many processes before it can itself run.

To provide some system functionality, more than one process is required and some of these processes might depend on other processes. A modern application that provides some functionality might require a group of interrelated processes. We would like to start, stop, analyse, diagnose, in other words, manage the group as a single unit. We refer to such a unit as a *service* and the mechanism of providing this on OpenIndiana as *Service Management Facility*.

SMF was originally developed to improve limitations of the old startup mechanism of the Solaris operating system using scripts known as `init.d`.

Service Management Framework on OI	Purpose
smf(7)	Service Management Facility

A service is a definition of how a service should be started while a service instance is a precise configuration. Some services can have multiple instances whereas others are defined to be a singleton.

Each service instance can be identified by a FMRI which uniquely defines each instance, which is why the SMF commands use FMRIs to identify services. A SMF FMRI is composed of three sections, the sections are delimited by a colon:

```
service resource:service name:service instance
```

FMRIs are found in various areas: software packages, ZFS, etc., and the first part denotes this area or *scheme*. For software packages the scheme is `pkg:`, while for services the scheme is `svc`.

Many commands use the FMRI to identify service instances, but it is not the only way of specifying a service instance to the commands. In this manual, nomenclature is simplified as `[service]` as a unique identification of a service.

- Opensolaris documentation

5.1 SMF Basic Commands

Command	Purpose
svcs(1)	display information about services
svcadm(7)	manage services (enable/disable, …)
svccfg(8)	configure services
svcprop(1)	display service configuration properties
inetadm(8)	initialise service managed by `inetd`
inetconv(8)	convert `inetd` service to smf services

Use of the SMF commands require relevant privileges. The following RBAC profiles can be made available to all a user to become a SMF administrator:

RBAC profile	Purpose
Service Management	manage services. A service manager can manipulate any service in any way
Service Operator	administer services. A service operator can enable or disable any service.

Further details: smf_security(7)

5.1.1 svcs

This command is used to display services and their relevant status. It is also used to provide a list of services that depend on a service, and it can also display a list of services that a service requires to operate.

All services generate log files and the location of a log file associated with a service can be obtained using this command. A list of processes requirred by a service can be displayed using this command.

Usage: `svcs [options] [FMRI]`

Option	Information to Display
	enabled services
`?`	detailed help
`a`	all services
`d [service]`	services that this service requires

Option	Information to Display
`D [service]`	services that require this service
`l [service]`	verbose information on service
`L [service]`	log file associated with service
`o column,...`	various information as defined by *columns*
`p [service]`	processes associated with service
`x`	details on services that are enabled but not running or prevent other services from running
`v`	more verbose output for some options
`z [zone]`	services in `zone`

- **service:** to specify a service use the FMRI, an FRMI instance or a FMRI pattern. Obtain a list of service FMRIs via `-o FMRI`.
  - **column:** comma separated column names
- **desc:**: description of service
- **fmri:**: Fault Management Resource Identifier
- **inst:**: service instance
- **svc:**: name of service
- **state:**: service state: uninitialised, offline, online, maintenance, disabled, degraded

5.1.1.1 Examples List services and their status

```
svcs
svcs -a
svcs -o svc,state,desc
```

Get printer information

```
> svcs -o svcs,desc | grep -i print
application/print/service-selector print service selector
application/cups/scheduler CUPS Print Spooler
> svcs -l scheduler
fmri         svc:/system/scheduler:default
name         default scheduling class configuration
enabled      true
state        online
next_state   none
state_time   March 27, 2023 at 08:03:54 AM CEST
logfile      /var/svc/log/system-scheduler:default.log
restarter    svc:/system/svc/restarter:default
dependency   require_all/none svc:/system/filesystem/root (online)

fmri         svc:/application/cups/scheduler:default
name         CUPS Print Spooler
enabled      true
state        online
next_state   none
state_time   March 27, 2023 at 08:03:57 AM CEST
logfile      /var/svc/log/application-cups-scheduler:default.log
```

```
restarter    svc:/system/svc/restarter:default
contract_id  42
dependency   require_all/none file://localhost/etc/cups/cupsd.conf (online)
dependency   require_all/none svc:/system/filesystem/minimal (online)
dependency   optional_all/error svc:/network/loopback (online)
dependency   optional_all/error svc:/milestone/network (online)
```

5.1.2 svcadm

Use this command to change the status of a service: enable, disable, etc.

Usage: `svcadm [options] [subcommand] [subcommand options] [service]`

options	purpose
`v`	verbose
`z [zone]`	command applies to the zone specified

subsommand	subcommand option	Purpose
`enable`		enable `service` to `online` state
	`r`	enable `service`, but also recursively enable its dependencies
	`t`	enabling a service as persistent, even after rebooting. This option enables the service *only temporarily* until the next system boot
`disable`		disable `service` to `offline` state
	`c`	add a comment which can be viewed by `svcs`
	`t`	disable service *temporarily*. The service will be enabled on ensuing boots
`restart`		restart `service`
`refresh`		re-read service configuration, so any configuration changes will take effect after a service `restart`
`clear`		for a service in a *maintenance* state, signal the service as repaired. for a service in a *degraded* state, attempt to bring the service online

5.1.2.1 Examples Restart a service

```
$ date
XXXXX, 2023 at 03:26:50 PM CEST
# svcadm restart application/cups/scheduler
# svcs -l application/cups/scheduler | grep state_time
XXXXX, 2023 at 03:27:02 PM CEST
```

First, we check the system time which is 3:26:30 PM. We then restart the CUPS printer scheduler, then check the status of the service using `-l` and extract the service property `stat_time` to obtain the time at which the service became online.

5.1.3 svccfg

This command is used to manage service configurations.

Usage: `svccfg [options] -s [FMRI]`

5.1.4 svcprop

This is used to retrieve service instance properties. It is also used to snapshots. It can also be used to validate the existence of a property.

options	purpose
`[service]`	display all properties of `service`
`v`	verbose
`z [zone]`	command applies to the zone specified

Refer to the manual page for details: scvprop(1)

5.1.5 inetadm

5.1.6 inetconv

5.2 Troubleshooting

5.2.1 Basic Strategy

Here is general methodology intended as an introduction in how to resolve a service that fails.

```
svcs -xv
```

This command provides the following information: - a possible reason for the failure - location of the log file associated with the service - references for further reading

Analysis during this step will frequently suffice to resolve your issue. However, this is not alway the case.

```
svcs -d FMRI
```

This command provides a list of services that the failed service requires. The failed service might be failing due to a service that it depends on not operating appropriately, although its status is online and it *appears* to function correctly.

```
svcs -l FMRI
```

While this command provides similar information provided by the first command, this command will also provide information on services that are online and but might exhibit dubious behaviour. This command will provide information similar to the first command.

```
svcs -d FMRI
```

Should analysis of the previous commands not be sufficient to resolve the issue, examination of the processes associated with suspicioous service might be necessary. Use this command to list all processes associated with the service. Use the tools from the previous section, System Monitoring, to scrutinise each process listed.

5.2.1.1 Examples To illustrate troubleshooting, we manipulate a service by removing a dependency:

- move a dependency file `/etc/cups/cupsd.conf`
- `refresh` and `restart` the service
- Use SMF commands in an attempt to reveal the cause of the failure

```
# svcs -l svc:/application/cups/scheduler:default
fmri         svc:/application/cups/scheduler:default
name         CUPS Print Spooler
enabled      true
state        online
next_state   none
state_time   XXXXX XX, 2023 at 09:18:13 AM CEST
logfile      /var/svc/log/application-cups-scheduler:default.log
restarter    svc:/system/svc/restarter:default
contract_id  42
dependency   require_all/none file://localhost/etc/cups/cupsd.conf (online)
dependency   require_all/none svc:/system/filesystem/minimal (online)
dependency   optional_all/error svc:/network/loopback (online)
dependency   optional_all/error svc:/milestone/network (online)
```

The service is `online` and is operating satisfactorily.

Rename `/etc/cups/cupsd.conf`, refresh the configuration and restart the service, will cause the service to fail:

```
# mv /etc/cups/cupsd.conf /etc/cups/cupsd.conf.orig
# svcadm refresh svc:/application/cups/scheduler:default
# svcadm restart svc:/application/cups/scheduler:default
```

Obtain a list of failed services:

```
# svcs -xv
svc:/application/cups/scheduler:default (CUPS Print Spooler)
 State: offline since March 29, 2023 at 07:25:48 PM CEST
Reason: Dependency file://localhost/etc/cups/cupsd.conf is absent.
   See: http://illumos.org/msg/SMF-8000-E2
   See: man -M /usr/share/man -s 8 cupsd
   See: /var/svc/log/application-cups-scheduler:default.log
Impact: This service is not running.
```

We always obtain a reason which in our case pinpoints the error precisely. This is not always the case.

5.2.2 Corrupt Service Configuration: Revert to an Old Configuration

SMF periodically takes snapshots of the service configuration so that it is alway possible to revert to a previous configuration.

```
# List all cups services:
$ svcs | grep -i cups
legacy_run     16:33:35 lrc:/etc/rc3_d/S99cups-browsed
online         14:33:25 svc:/application/cups/scheduler:default
# list snapshots associated with the cups service
$ svccfg -s svc:/application/cups/scheduler:default listsnap
initial
last-import
running
start
# There are four snapshots available one of which is currently running.
# Lets suppose we had a corrupt config and would live to revert to the initial
# cups service configuration:
$ svccfg -s svc:/application/cups/scheduler:default revert initial
# re-read configuration
svcadm refresh svc:/application/cups/scheduler:default
svcadm restart svc:/application/cups/scheduler:default
```

5.2.3 Boot Failure

- On the boot menu, select =3= to load boot prompt

```
3. Escape to loader prompt
```

- Now boot without services by entering: `boot -m milestone=none`:

```
OK boot -m milestone=none
```

- Login as root and enable all services:

```
svcadm milestone all
```

- Now the system will boot and fail. So use `svcs` to determine which services are failing:

```
svcs -a
```

5.2.4 Maintenance Mode

6 Configuring and Tuning

There are a few tools to configure and tune an OpenIndiana system. One of the tools is sysding, a tool that is used by the current OpenIndiana installer to setup static IP addresses when NWAM (automatic network configuration) is not used.

For more information on sysding, see the manpage and example config file :

```
# man sysding
```

The configuration file is :

`/etc/sysding.conf`

There is also a logfile :

`/var/log/sysding.log`

For example a simple /etc/sysding.conf configuration file for a static IP is :

```
setup_interface e1000g0 v4 10.0.2.16
setup_route default 10.0.2.2
```

This service is meant to be run only once and sets a boolean flag config/finished. In case you want to run sysding again, you'll have to force the finished flag to false :

```
svccfg -s sysding:system
setprop config/finished = false
refresh
```

6.1 Configuring a UPS

- NUT?

6.2 Fault management (FMA)

< place holder >

6.3 Virtual Terminals/Consoles (VT)

Virtual Terminals/Virtual Consoles (VT) are used to switch between terminals and using system in text mode with *Ctrl+Alt+F1,F2..F8* key combinations.

> ⓘ **NOTE:**
> Switching to VTs using *Ctrl+Alt+Fn* during current desktop session (at **VT7**) can result in **X server** and **lightdm** (or **gdm**) restarting and stopping all applications running within it. Currently, there could be problems with getting back to gdm/Xorg session if switching to VTs after gdm restart. Use `svcadm restart ligthdm` (or `svcadm restart gdm`) command again, to have lightdm/gdm Xorg session restarted at **VT8** (*Ctrl+Alt+F8*).

At fresh Openindiana install, VT/Consoles are **not enabled by default**. One needs to set up vtdaemon and console-login:vt2 (till :vt6) and enable **vtdaemon** options/hotkeys property:

```
pfexec svcadm enable vtdaemon
pfexec svcadm enable console-login:vt2
pfexec svcadm enable console-login:vt3
pfexec svcadm enable console-login:vt4
pfexec svcadm enable console-login:vt5
pfexec svcadm enable console-login:vt6
```

Or do that in a one-liner Bash script: `for i in 2 3 4 5 6 ; do pfexec svcadm enable console-login:vt$i; done;`

Then, enable options/hotkeys property (*Ctrl+Alt+Fn*) to switch VTs and refresh and restart **vtdaemon** service:

```
pfexec svccfg -s vtdaemon setprop options/hotkeys=true
pfexec svcadm refresh vtdaemon
pfexec svcadm restart vtdaemon
```

Optionally, you can also disable VT consoles auto screen locking (recommended for personal use, not recommended on server): `pfexec svccfg -s vtdaemon setprop options/secure=false`

The above was inspired by this blog by Danx on Virtual Consoles.

6.4 Service management (SMF)

> ⓘ **NOTE:**
> ITEMS TO WRITE ABOUT: provide more detailed explanations.

List services:

```
$ svcs # list (permanently) enabled services
$ svcs -a # list all services
$ svcs -vx # list faulty services
```

Get information about a service:

```
$ svcs <service name> # one-line status
$ svcs -x <service name> # important information
$ svcs -d <service name> # check the service's dependencies
$ svcs -l <service name> # all the available information
```

Start a service:

```
# svcadm enable <service name> # permanently enable/start
# svcadm enable -t <service name> # temporary start (won't survive a reboot)
# svcadm enable -r <service name> # permanently enable/start service along with its
 ↪  dependencies
```

Restart / reload a service:

```
# svcadm refresh <service name> # reload the service's configuration
# svcadm restart <service name> # restart the service
```

6.5 Systems logging and monitoring

> ⓘ **NOTE:**
> ITEMS TO WRITE ABOUT:
> - Where to find the logs (`/var/log`, `/var/svc/log`).

7 Illumos boot process

< place holder >

8 Security

< place holder >

9 Zones

> ⓘ **NOTE:**
> ITEMS TO WRITE ABOUT:
> - Need to mention some of the changes to zone management…e.g..
>   - sys-unconfig gone.
>   - sysding replaced syscfg
>   - now have to have DNS, root password, etc. all configured inside the zone before being able to logon using `zlogin -C <zonename>`, otherwise have to do `zlogin <zonename>`.
>
> So a fair amount of stuff has changed there.

Zones are an OpenIndiana feature that provides operating system-level virtualization. Each zone is managed as a completely separate OpenIndiana machine. Zones have very low overhead and are one of the most efficient forms of OS virtualization.

The global zone (GZ) is the operating system itself, which has hardware access. From the global zone, non-global zones (NGZ) are created and booted. Boot time for non-global zones is very fast, often a few seconds. The CPU, network, and memory resources for each zone can be controlled from the global zone, ensuring fair access to system resources. Disk space access is usually controlled by ZFS (with quotas and reservations if needed), as well as mounting of filesystem resources with NFS or lofs. As with other forms of virtualization, each zone is isolated from the other zones – zones cannot see processes or resources used in other zones. The low marginal cost of a zone allows large systems have tens or even hundreds of zones without significant overhead. The theoretical limit to the number of zones on a single platform is 8,192.

Different releases of (Open)Solaris used different packaging distribution method for the global zone. OpenIndiana zones use two basic brands - “ipkg” and “nlipkg”, which are based on IPS Packaging. The brand determines how zone is initialized and how zone’s processes are treated by kernel. Both type of zones represent a PKG image. “ipkg”-branded zones are tightly coupled with GZ. Image pakaging system (IPS) knows about ipkg-branded zones and can perform several actions simultaneously in GZ and NGZ. For example, you can update all your zones and GZ with a single “pkg update -r” command. IPS can ensure some depenencies between packages in GZ and NGZ. To allow this it cheks that NGZ’s publishers are a superset of GZ’s publishers and their properties are the same (for example, stickiness or repository location). As this is not always suitable for development zones, “nlipkg”-branded zones were introduced. “nlipkg”-branded zone behave like completely independent instance and IPS ignores them during operations in GZ.

An easy way to implement zones is to use a separate ZFS file system as the zone root's backing store. File systems are easy to create in ZFS and zones can take advantage of the ZFS snapshot and clone features. Due to the strong isolation between zones, sharing a file system must be done with traditional file sharing methods (eg NFS).

When each zone is created it comes with a minimal set of packages, and from there you can add and use most packages and applications as required.

9.1 Zone networking model

OpenIndiana zones can use one of two networking models: a shared IP stack and an exclusive IP stack. There are pros and cons to each of them.

Low-level system networking components, such as the ipfilter firewall and the kernel IP routing tables attach to an "IP stack", and are thus either unique to a zone or shared by all zones with the one shared stack. There are also some other nuances, such as that the zones with the shared stack can communicate over IP directly, regardless of their subnetting and, to some extent, default firewall packet filtering (that has to be specially configured), while exclusive-IP zones with addresses in different subnets have to communicate over external routers and are subject to common firewall filtering.

The global zone defines which physical networks and VLANs the NGZ has access to, and hands down the predefined networking interfaces (the NGZ can not use or create other interfaces). Also, while shared networking allows to configure and attach (or detach) network interfaces from GZ to the NGZ "on the fly", changes in exclusive networking require reboot of the zone to propagate device delegation.

A zone with an exclusive IP stack can have all the benefits of dedicated hardware networking, including a firewall, access to promiscuous sniffing, routing, configuration of its own IP address (including use of DHCP and static network addressing), etc. This requires a fully dedicated NIC. Usually this is a VNICs - a virtual adapter with own MAC address, that operate as if the VNIC was plugged directly into local LAN in a particular (or default) VLAN. Note also that the local zones with an exclusive IP stack are not subject to the host's shared-stack firewall.

> ⓘ **NOTE:**
> Note that if you create zone in VM, the hypervisor can require you to provide some information about its mac addresses. For example, if you configure bridged networking on VirtualBox running on OpenIndiana, you must set secondary-macs property of the VNIC, delegated to VM, to the set of mac addresses of VNICs created inside VMs and enable promiscous mode on VirtualBox network adapter.

9.2 Quick Setup Example

Zone creation consists of two steps - creating zone configuration and zone installation or cloning. Zone configuration determines basic parameters, such as zone's root location and provided resources. Zone configuration is performed using zonecfg tool, zone administration (for example, installation) is performed using zoneadm tool.

For example, we create a simple zone with shared networking model:

```
# zonecfg -z example
example: No such zone configured
Use 'create' to begin configuring a new zone.
zonecfg:example> create
zonecfg:example> add net
zonecfg:example:net> set physical=e1000g0
zonecfg:example:net> set address=192.168.0.10/24
zonecfg:example:net> end
zonecfg:example> set zonepath=/zones/example
zonecfg:example> verify
zonecfg:example> commit
zonecfg:example> exit
```

Here `create` puts you inside the zone configuration program where you can change and update settings particular to the zone specified with -z. `zonecfg` break different resource groups of data, you add a new resource with add. The next block adds ressource “net”, configuring network in default `shared` ip-type mode. It allows zone to share IP stack with GZ. If you want to get dedicated nic in NGZ, you have to use `set ip-type=exclusive`. In exclusive mode zone has complete control over network interface and you can’t assigned address in zonecfg prompt. After network configuration `zonepath` is set. It’s a location for zone’s root file system, which should be a ZFS filesystem. The `verify` command checks that no mistakes were made. Finally changes are committed (saved to zone configuration file).

If you want to use dedicated networking for your zone, you should create VNIC and delegate it to the zone.

```
# dladm create-vnic -l e1000g0 vnic0
```

If you want to specify VLAN id for VNIC, use `-v dladm create-vnic` option:

```
# dladm create-vnic -l e1000g0 -v 123 vnic0
```

Now you can create your zone and delegate VNIC to it (note that we set zone’s `ip-type` to `exclusive`):

```
# zonecfg -z example
example: No such zone configured
Use 'create' to begin configuring a new zone.
zonecfg:example> create
zonecfg:example> set ip-type=exclusive
zonecfg:example> add net
zonecfg:example:net> set physical=vnic0
zonecfg:example:net> end
zonecfg:example> set zonepath=/zones/example
zonecfg:example> verify
zonecfg:example> commit
zonecfg:example> exit
```

After configuring a zone you can install it with `zoneadm install` subcommand:

```
# zoneadm -z example install
```

During installation pkg image rooted at $zonepath/root is created and minimal set of packages is installed to the image. When installation finishes, zone can be booted with `zoneadm -z`

`example boot` command. If you want your zone to boot automatically during system startup, you should set autoboot parameter to true during zone configuration:

```
zonecfg:example> set autoboot=true
```

Once zone is booted you can log in locally with `zlogin example`. If you created zone with dedicated network adapter, you should configure it inside zone.

> ⓘ **NOTE:**
> Note, that on first zone boot sysding(1M) will set root's password to NP. Before this happened you will not be able to login to zone with zlogin, so this command will not work on early startup stage.

9.3 System repository configuration

On OpenIndiana it is possible to allow NGZs to access configured publishers via GZ proxy service (so-called zone proxy daemon). This can be useful for sharing pkg cache between zones or to provide network access for performing updates to otherwise restricted zone environment (i.e. to zone without Internet access).

The functionality is provided by series of services in GZ and NGZs. In GZ two services are running: system repository service and zones proxy daemon (see `pkg.sysrepo(1M)`). In NGZ zones proxy client communicates with GZ's zone proxy daemon. System repository service `svc:/application/pkg/system-repository` is responsible for providing access to the package repositories configured in a reference image through a centralized proxy. Zones proxy daemon service `svc:/application/pkg/zones-proxyd` starts on system boot and registers door in each running ipkg-branded zone (the door is created at `/var/tmp/zoneproxy_door` path). Later, on zone startup or shutdown /usr/lib/zones/zoneproxy-adm is used to notify zones-proxyd, so that it could create the door for the zone or to cleanup it. Zones proxy daemon client `svc:/application/pkg/zones-proxy-client:default` runs in NGZ and talks to GZ's zones-proxyd via created door.

> ⓘ **NOTE:**
> Note, you can't use system repository with nlipkg-branded zones.

IPS determines if it should use zones proxy client in zone based on image's `use-system-repo` property (defaults to False).

To configure your system to use system repository, perform the following actions.

1) In global zone:

```
# pkg install pkg:/package/pkg/system-repository pkg:/package/pkg/zones-proxy
# svcadm enable svc:/application/pkg/system-repository:default
# svcadm enable svc:/application/pkg/zones-proxyd:default
```

2) In non-global zone:

```
# pkg install pkg:/package/pkg/zones-proxy
# svcadm enable svc:/application/pkg/zones-proxy-client:default
# pkg set-property use-system-repo True
```

After this in NGZ's publisher description you'll see system-repository location:

```
# pkg publisher
PUBLISHER                    TYPE     STATUS P LOCATION
openindiana.org (non-sticky, syspub) origin   online T <system-repository>
hipster-encumbered (syspub)     origin   online T <system-repository>
```

You can check if your configuration works by issuing `pkg refresh` command in the zone. `pkg(1M)` should contact repository indirectly via zones-proxy-client.

To revert your zone to proxy-less configuration, run

```
# pkg set-property use-system-repo False
```

9.4 Troubleshooting

Zone configuration and management operations can fail for a number of reasons, sometimes obscure. This section contains information on how to solve different well-known issues.

9.4.1 Fixing zone installation issues

Zones on OpenIndiana use ZFS features to manage boot environments. Zone's root and its parent directory should be a ZFS dataset. It's not enough for zone's root (or its parent directory) to be just a directory on ZFS filesystem. If you try to create a zone which root doesn't satisfy this requirement, you can get the following error:

```
# zoneadm -z example install
Sanity Check: Looking for 'entire' incorporation.
ERROR: the zonepath must be a ZFS dataset.
The parent directory of the zonepath must be a ZFS dataset so that the
zonepath ZFS dataset can be created properly.
```

To fix this, you can just uninstall the zone and create its root dataset manually:

```
# zoneadm -z example uninstall
Are you sure you want to uninstall zone example (y/[n])? y
# zfs create rpool/zones/example
# chmod 700 /zones/example
# zoneadm -z example install
```

10 Storage

< place holder >

10.1 Mounting file systems

> ⓘ **NOTE:**
> ITEMS TO WRITE ABOUT:
> - Need a walkthrough of mounting options for other filesystems…FAT, UFS, etc.

10.1.1 Mounting and Unmounting ISO images

One can use lofiadm(1M) to mount ISO images by attaching them to a block device.

```
pfexec lofiadm -a /path/to/foo.iso /dev/lofi/1
```

When the above line is repeated for several ISO images, issue the `lofiadm` command to list which ISO images are attached to which block devices.

```
pfexec lofiadm
Block Device File Options
/dev/lofi/1 /home/scarcry/foo.iso -
/dev/lofi/2 /home/scarcry/bar.iso -
```

Use the mount(1M) command to mount an image:

```
pfexec mount -F hsfs -o ro /dev/lofi/1 /mnt
```

Check the mounted image by issuing `ls` on the mount point.

```
ls /mnt
```

To unmount and detach the image(s):

```
pfexec umount /mnt
pfexec lofiadm -d /dev/lofi/1
```

10.1.2 Mounting NTFS Volumes - 3rd party support

> ⓘ **NOTE:**
> For removable storage devices, first make sure your external disk drive is connected and powered on.
> To list attached removable storage devices: `rmformat -l`

Verify the pX partition number that contains the NTFS filesystem, typically “p1”, using fdisk. Even though it may seem counterintuitive, include the partition number “p0” as shown by rmformat in fdisk inquiries.

```
pfexec fdisk /dev/rdsk/c6t0d0p0
```

10.1.2.1 Additional software installation and configuration NTFS and FUSE support is provided by Tuxera’s NTFS-3G project that aims at providing a stable NTFS driver for several operating systems.

It is made possible thanks to Jean-Pierre André, on Openindiana page: http://jp-andre.pagesperso-orange.fr/openindiana-ntfs-3g.html Follow instructions on this page to install. Please install 64-bit package on 64-bit installations.

10.1.2.2 Mounting the NTFS filesystem Now mount the NTFS partition using:

```
pfexec ntfs-3g /dev/dsk/c6t0d0p1 /mnt/backup/
```

You can now also add a vfstab entry like so:

```
/dev/dsk/c6t0d0p1 /dev/rdsk/c6t0d0p1 /mnt/backup ntfs-3g - no -
```

10.2 Configuring OpenIndiana as an ISCSI Target Server - (COMSTAR)

< Place holder for content >

10.3 System backups

OpenIndiana offers several backup solutions. Here are just a few of them:

- Borg Backup
- Bacula
- Time-Slider
- rdiff-backup
- Rsync
- Zetaback
- ZFS exports
- cpio
- tar, zip, etc.

10.4 ZFS

> ⓘ **NOTE:**
> ITEMS TO WRITE ABOUT:
> Gotcha's such as the following:

```
<e^ipi> don't suppose there's any solution to this:
<e^ipi> cannot replace 15092805280450214 72 with
 ↪  /dev/dsk/c0t5000C5009204EB9Bd0s0: devices have different sector alignment
<tsoome> thats 512 versus non-512 sector issue
<tsoome> you need to build new pool based on larger sector
<tsoome> if its mirror, you can attach 512B disk to 4k pool, but not vice
 ↪  versa...
<e^ipi> well, damn.
<tsoome> that error message is too confusing, should be replaced by more clear
 ↪  one;)
<e^ipi> I swear this pool is already mix & match, freebsd complained about it
<e^ipi> (but still used it)
<tsoome> there is that thing that ashift is vdev property;)
<tsoome> not pool property (one reason why that linux zpool create ashift=
 ↪  option is bad)
<tsoome> or sort of bad anyhow
```

10.4.1 Importing ZFS disks

> ⓘ **NOTE:**
> ITEMS TO WRITE ABOUT:
> - Talk about the ZFS import command.

10.4.2 How does one mirror their root zpool?

> ⓘ **NOTE:**
> ITEMS TO WRITE ABOUT:
> - Adding a 2nd disk to the root pool

10.4.3 How does one create additional zpools?

> ⓘ **NOTE:**
> ITEMS TO WRITE ABOUT:
> - zpool create command
>   - Mirrors
>   - Raidz

10.4.4 Modifying zpool settings and attributes

> ⓘ **NOTE:**
> ITEMS TO WRITE ABOUT:
> - zpool get/set commands

10.4.5 Modifying zfs file system settings and attributes

> ⓘ **NOTE:**
> ITEMS TO WRITE ABOUT:
> - zfs get/set commands

10.4.6 How does one create additional zfs datasets?

> ⓘ **NOTE:**
> ITEMS TO WRITE ABOUT:
> - zfs create command

10.4.7 Configuring system swap

> ⓘ **NOTE:**
> ITEMS TO WRITE ABOUT:
> - zfs set command
> - swap -l

11 Virtualization

< Place holder >

11.1 OpenIndiana as a virtualization host server

ⓘ **NOTE:**
ITEMS TO WRITE ABOUT:

- Qemu-KVM (KVM) walkthrough
  - illumos KVM port does not support AMD processors.
  - Intel processors require EPT support.
- Virtualbox walkthrough
  - There is no package for this yet, but folks do have it working, see the wiki for details.

ⓘ **NOTE:**
ITEMS TO WRITE ABOUT:
In a nutshell, most modern Intel processors such as i3, i5, i7, and Xeon provide EPT support. Most older processors such as Core2duo and Core2Quad lack EPT support, and a few of them lack virtualization support at all. You can check your processor for EPT support via the following link: http://ark.intel.com/Products/VirtualizationTechnology

12 Localization

ⓘ **NOTE:**
ITEMS TO WRITE ABOUT:
Possible resources to help write this section:

- https://wiki.openindiana.org/oi/4.4+Localization
- https://docs.oracle.com/cd/E23824_01/html/E26033/glmen.html

13 Dtrace

< Place Holder >

14 Configuring Networking

14.1 Automatic Configuration (NWAM)

During installation of OpenIndiana, there are three options to configure networking : Automatic, Manual (static IP), and None.

Option Autamatic enables NWAM and configures all interfaces automatically.

Network Auto-Magic (NWAM) manages network interfaces as they are dynamically added or removed to or from the system, using network profiles, and is able to change network settings on the fly.

NWAM is suitable for servers, laptops, desktops and workstations alike, to automatically configure all wired or wireless network interfaces.

Option Manual disables NWAM and creates a /etc/sysding.conf file to setup static IP addresses.

Option None can be used to configure networking manually, without NWAM, after installation.

14.2 Changing from NWAM to Manual Configuration

If during install you enabled NWAM, and if you want to disable NWAM you can proceed as follows :

```
# svcadm disable physical:nwam
```

Define your IP/hostname in `/etc/hosts`. For example:

```
192.168.1.22 hostname hostname.local localhost loghost
# Subsittude 192.168.1.22 for YOUR IP
```

Enable the default physical service with `svcadm` and configure the `interface`:

```
# svcadm enable physical:default
```

There are multiple ways to configure the interface : using sysding or directly with ipadm.

For more information on sysding, see the section on Configuring and Tuning.

To configure an interface with ipadm:

```
# ipadm create-addr -T static -a local=192.168.1.22/24 bge0/v4static
```

or

```
# ipadm create-addr -T dhcp bge0/dhclient
```

The previous example sets up DHCP without using NWAM. NWAM is more than DHCP, because NWAM automatically configures any new interfaces, while the above way to manually setup DHCP is fixed for a specific interface.

If you do not know what the interface name is (bge0 in this case); then type in

```
$ dladm show-link
```

or:

```
$ kstat -c net | grep net

# look for hme0, bge0, e1000g0 or soemthing that resembles the driver in use.
```

Add gateway

```
# route -p add default 192.168.1.121
```

or

```
# nano /etc/defaultrouter

# Enter in your gateways IP
```

or use the /etc/sysding.conf file to configure a default router and DNS servers.

Set DNS server(s)

```
# nano /etc/resolv.conf

# Enter in the DNS server IP(s)
nameserver 192.168.1.121
```

or

```
# echo "nameserver 192.168.1.121" >> /etc/resolv.conf
```

Restart

```
# reboot
```

> ⓘ **NOTE:**
> IF you cannot ping an external IP (e.g. google.com) run this command and try again.

```
# cp /etc/nsswitch.dns /etc/nsswitch.conf
```

credit for this section of the docs go to /u/127b

14.3 More on Automatic Configuration (NWAM)

The following is a more in depth discussion of NWAM, which normally works without any user interaction, but NWAM can be customized using profiles by the user.

14.3.1 Using NWAM configuration tools

Usually NWAM configuration is done via GUI `nwam-manager` applet and `nwam-manager-properties` program.

From CLI all configuration can be done using two tools, `nwamcfg` to configure network profiles and `nwamadm` to manage them.

On its backend, NWAM relies on `nwamd`, the NWAM policy engine daemon and `netcfgd`, the NWAM repository daemon.

Access to the command line and GUI tools is controlled via security profiles ‘Network Auto-conf Admin’ and ‘Network Autoconf user’.

To create NWAM profile use `nwamcfg` tool.

```
# nwamcfg
nwamcfg> create ncp Abroad
nwamcfg:ncp:Abroad> end
```

Now you can see a new profile in the list of NCPs:

```
nwamcfg> list
NCPs:
Abroad
Automatic
Locations:
Automatic
NoNet
User
nwamcfg>
```

The network interface to be managed under the new profile can be set and configured with `create ncu` command:

```
# nwamcfg
nwamcfg> select ncp Abroad
nwamcfg:ncp:Abroad> create ncu phys e1000g0
Created ncu 'e1000g0'. Walking properties ...
activation-mode (manual) [manual|prioritized]> prioritized
enabled (true) [true|false]>
priority-group> 0
priority-mode [exclusive|shared|all]> exclusive
link-mac-addr>
link-autopush>
link-mtu>
nwamcfg:ncp:Abroad:ncu:e1000g0>
```

Most important here is the `activation-mode`, which states if the profile is to be automatically set based on certain policies or if it is to be manually set using nwamadm. The latter should be most likely the case in a server environment. The `priority-group` and `priority-mode` here are set to (0 / exclusive). This says that the profile is for wired access and will always be the only active one at a time.

If after listing the changes you are satisfied with the configuration, permanently store it using `commit` command:

```
nwamcfg:ncp:Abroad:ncu:e1000g0> list
ncu:e1000g0
 type link
 class phys
 parent "Abroad"
 activation-mode prioritized
 enabled true
 priority-group 0
 priority-mode exclusive
nwamcfg:ncp:Abroad:ncu:e1000g0> commit
Committed changes
nwamcfg:ncp:Abroad:ncu:e1000g0> end
```

After an interface has been assigned to the profile, its IP configuration has to be defined:

```
nwamcfg:ncp:Abroad> create ncu ip e1000g0
Created ncu 'e1000g0'. Walking properties ...
enabled (true) [true|false]>
```

```
ip-version (ipv4,ipv6) [ipv4|ipv6]> ipv4
ipv4-addrsrc (dhcp) [dhcp|static]> static
ipv4-addr> 192.168.100.100
ipv4-default-route> 192.168.100.1
nwamcfg:ncp:Abroad:ncu:e1000g0> list
ncu:e1000g0
 type interface
 class ip
 parent "Abroad"
 enabled true
 ip-version ipv4
 ipv4-addrsrc static
 ipv4-addr "192.168.100.100"
 ipv4-default-route "192.168.100.1"
 ipv6-addrsrc dhcp,autoconf
nwamcfg:ncp:Abroad:ncu:e1000g0>
```

As you can see, the profile uses a static IP address and, for simplicity reasons, only provides IPv4 networking. Again, if you're happy with the results, commit them.

```
nwamcfg:ncp:Abroad:ncu:e1000g0> commit
Committed changes
nwamcfg:ncp:Abroad:ncu:e1000g0>
```

As it stands, now you have defined a pysical and ip layer attached to a network profile:

```
nwamcfg:ncp:Abroad:ncu:e1000g0> end
nwamcfg:ncp:Abroad> list
NCUs:
 phys e1000g0
 ip e1000g0
nwamcfg:ncp:Abroad>exit
```

To activate the profile, you use nwamadm. First of all, check for the current situation:

```
# nwamadm list
TYPE     PROFILE    STATE
ncp      Abroad     disabled
ncp      Automatic  online
ncu:phys e1000g0    online
ncu:ip   e1000g0    online
loc      Automatic  online
loc      NoNet      offline
loc      User       disabled
```

As you can see, Automatic is the active profile while Abroad is currently disabled. We can change that easily, but be aware that you might lock yourself out if you are connected via SSH when changing the profile!

```
# nwamadm enable -p ncp Abroad
Enabling ncp 'Abroad'
```

To check if the change has worked, you can use nwam again, and also ifconfig should show that interface is up:

```
# nwamadm list
TYPE PROFILE STATE
ncp Abroad online
ncu:phys e1000g0 online
ncu:ip e1000g0 online
ncp Automatic disabled
loc Automatic online
loc NoNet offline
loc User disabled
```

```
# ifconfig e1000g0
e1000g0: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 8
inet 192.168.100.100 netmask ffffff00 broadcast 192.168.100.255
ether 0:c:29:56:46:73
```

If you want to revert to default `Automatic` ncp, use the following command:

```
# nwamadm enable -p ncp Automatic
```

Although it is possible to directly modify profiles using an editor, it is not advisable and will be hardly necessary, anyway. One of the coolest features of NWAM tools is that they can be completely scripted. All steps above could also be put in one line:

```
# nwamcfg "create ncp Abroad;create ncu phys e1000g0;set activation-mode=manual;set
 ↪  enabled=true;set priority-group=0;set priority-mode=exclusive;end;create ncu ip
 ↪  e1000g0;set enabled=true;set ip-version=ipv4;set ipv4-addrsrc=static;set
 ↪  ipv4-addr=192.168.100.100;set ipv4-default-route=192.168.100.1;commit"
```

Or, more nicely formatted, commented and stored in a file:

```
#
# Profile for use on the road
#
# Created on 13/01/2013 by S. Mueller-Wilken
#
create ncp Abroad

# Create physical interface definition for 1st network card
create ncu phys e1000g0
set activation-mode=manual
set enabled=true
set priority-group=0
set priority-mode=exclusive
end

# Create IP configuration for first network card
create ncu ip e1000g0
set enabled=true
set ip-version=ipv4
set ipv4-addrsrc=static
set ipv4-addr=192.168.100.100
set ipv4-default-route=192.168.100.1
```

```
# Commit the settings
commit
```

This file can then be read by nwamcfg:

```
# nwamcfg -f abroad.cfg
Configuration read.
```

While there is no longer a need to fiddle around in /etc/nwam, the configuration is still completely there, as can be easily verified:

```
# ls /etc/nwam
loc loc.conf ncp-Abroad.conf ncp-Automatic.conf
```

All configuration is placed in readable ASCII files so that configuration from a global zone is possible:

```
# cat /etc/nwam/ncp-Abroad.conf
link:e1000g0 type=uint64,0;class=uint64,0;parent=string,Abroad;enabled=boolean,tru⌋
 ↪  e;activation-mode=uint64,4;priority-group=uint64,0;priority-mode=uint64,0;
interface:e1000g0 type=uint64,1;class=uint64,1;parent=string,Abroad;enabled=boolea⌋
 ↪  n,true;ipv6-addrsrc=uint64,0,1;ip-version=uint64,4;ipv4-addrsrc=uint64,2;ipv4-⌋
 ↪  default-route=string,192.168.100.1;ipv4-addr=string,192.168.100.100;
```

14.3.2 Network automagic online help

Comprehensive and fully illustrated online help for using NWAM is available by right clicking the NWAM tray icon and selecting *Help*. This opens help browser.

14.3.3 Troubleshooting NWAM

If NWAM is already configured and fails to connect to a wireless network try restarting the service.

For example:

```
# svcadm restart nwam
```

Sometimes the location gets set to *NoNet* and it's nessessary to manually change the location to *Automatic*.

When the location setting is configured to *Switch Locations Automatically*, it's not possible to change the location. This is resolved by reconfiguring the location to allow manual switching. To perform this task, do the following:

Right click the NWAM tray icon and select ***Location > Switch Locations Manually***. Right click the NWAM tray icon and select ***Location > Automatic***.

15 Clustering with Open HA Cluster

ITEMS TO WRITE ABOUT:

See old sun docs

- http://docs.oracle.com/cd/E19735-01/

Also see:

- http://zfs-create.blogspot.nl/