UNIX - From New User to Technical Expert

IBM's AIX on RS/6000's

Stewart Watkiss

Contents

List of figures

Introduction

About this book

This book was originally started as a revision guide, written whilst I was revising to take my AIX Advanced Technical Expert Certification exams. As the document got longer and longer, I decided to add a bit of structure and before I knew it, I had the starting of a book on AIX. I rearranged some of the sections, expanded a few of the sections where the information was a little brief and the revision guide started to take the form of a book suitable for both beginners to UNIX and as a reference for experienced AIX users and support personnel. This certainly helped with my revision. The first complete (or closest it came to complete) version took 18 months to write which is the same length of time it took me to complete the certification process.

The target audience of this book would typically be someone with little or no experience in UNIX, but maybe a bit of background using other operating systems (e.g. Windows 95/98). However all sections are explained in full and therefore no previous experience of any other environments are necessary. As the book is not just a beginners book, it takes in most of the requirements for System Support and is therefore suitable for someone with experience administering or supporting an AIX or other UNIX environment.

The book is based on AIX and some of the examples will only work on the AIX operating system, however most of the principles are the same for all different UNIX platforms and therefore as a theoretical guide this is also valid for other UNIX operating systems. You should refer to the documentation provided with your operating system where commands specific to AIX cannot be used.

UNIX the Computers Operating System

UNIX is an operating system, that is a layer of software that provides a path for applications to control and communicate with the hardware.

Hardware refers to the bit’s you can actually see and touch - Disk drives, Monitors, Keyboards, CPU etc.

Software refers to programs or pieces of code that make the computer perform a required function. In fact software is what differentiates a computer from other electronic devices. A computer has the ability through using different software to perform different functions. Compare this to an electronic typewriter that is only capable of doing what the hardware is coded to do (type letters directly onto paper). Nowadays it is harder to distinguish the difference as Word Processors can perform the function of a typewriter but can have the ability to do other things, such as transfer the files to a computer or save to disk.

Operating Systems are what ties the Software Applications and Hardware together.

To explain this a little clearer I'll use an analogy to a taxi. The taxi itself would be the hardware part of the computer, the people travelling in the taxi would be the applications and the taxi driver would be the operating system.
Imagine then that you are requested by your boss to attend a meeting at a different location and that you should take a taxi. This would be like a user requesting that an application performs a certain function. You would then request a taxi and get into it. You would not be allowed to drive the taxi yourself so would make a request to the driver (the operating system) to take you to a certain destination. The driver would drive the taxi (control the hardware by using the pedals, steering wheel etc.) to the destination requested by you. Once you reached your destination you would leave the taxi allowing it to transport other people.

You can see that this would shield you from having to know how to drive the taxi, you would not need to know whether it was a diesel taxi or a petrol taxi, or whether it was a manual or automatic. All you need to be able to do is to talk with the driver and explain where you wanted to go.

In this way if an application needs to write to a disk, it does not need to know the type and size of the disk, it does not need to know how to move the head backwards and forwards. All it needs to do it to send a request to the operating system asking it to write the required data to disk and the operating system will worry about all the details.
With multitasking operating systems such as UNIX, several programs can be running at once. The operating system is also required to control the applications so that they don't interfere with each other. It does this by controlling which applications should be running in the processor, which memory has been allocated to it and when it needs to step back and let another program run in the processor.

The following list are some of the operating systems that you may have heard of.
UNIX; MS-DOS; Windows 3.1 / 98 / NT and Apple Macintosh.

It is worth noting that whilst many of the operating systems come with bundled applications for example word processors, basic graphics editors / viewers, calculator etc. the applications are not actually part of the operating systems. If these applications were removed the operating system would still function properly and other applications could still communicate with the hardware via the operating system.

More about UNIX

UNIX was originally developed during the early 1970's at AT&T's Bell Laboratories. The main developers were Ken Thomas and Dennis Ritchie.

UNIX was developed as one of the earliest multi-user multitasking (time-sharing) operating systems. Earlier computers were only able to handle a single user running a single program at any one time. These new generation of computers had to be able to run more than one program by more than one user. This was done using time-sharing where each program (more accurately a thread) would run in the processor for a certain length of time, before being removed from the processor to let another program run. When done fast enough this gave the illusion that several programs were running simultaneously. With multiple processors it's possible to run a true multitasking environment where multiple threads can be running at simultaneously in the separate processors.

Where UNIX was different from other operating systems at the time was that it was designed as an open standard. Compared with proprietary operating systems which could only run on a certain computer, UNIX could be modified to run on a variety of different computers from different manufacturers. This no longer tied the operating system to any particular manufacturer of computer and was ultimately one of the reason for it's success.
The operating system became an open free operating system that was developed further by Universities and companies. The most notable of these was Berkeley University, in fact many of the features in UNIX are referred to as to BSD standard, referring to Berkeley Software Distribution.

Unfortunately this open standard also allowed companies to go there own way and develop different "improvements" to the operating system. This in turn brought about incompatibilities between the different versions. A list of some of the better known UNIX operating systems are listed:

Company / Developer	**UNIX Operating System**
Sun	SunOS / Solaris
IBM	AIX
Hewlett-Packard	HP-UX
Linus Torvalds	Linux

Main versions of UNIX available

So why is AIX different

AIX is a version of UNIX developed by IBM. It is developed to run on the RS6K computer (also referred to as: Risc System 6000; or RS/6000), which comes in many different versions.

AIX is an acronym for **A**dvanced **I**nteractive E**x**ecutive and is a combination of many of the popular flavours of UNIX.

Although many of the standard rules apply to AIX there are a few differences between the different versions of UNIX and where they might differ this book has provided the version for AIX. This does not mean that the book does not apply to different UNIX operating systems. Some of the commands will be different when not on AIX or may not have the same tools to make them easier to use / configure.

A few items that are different in AIX from some of the other UNIX Operating Systems are:

SMIT - Software Management and Installation Tool
This is product exclusive to IBM which simplifies a lot of the complicated instructions required for installing software and configuring the system.
ODM - Object Data Manager
This database provides a replacement for some of the flat files normally used to configure UNIX. This allows the operating system to be faster by combining the configuration details into a Data Repository. There are more reasons for using an ODM as opposed to flat files which I will not go into here.
Disk Storage - Allowing scalable servers
The disk storage is organised to allow different filesystems to be grown, expanded or moved as appropriate. This is different to methods implemented by other companies versions of UNIX.

There are also a lot of common things introduced to make some different UNIX Operating Systems look and feel the same despite being from different companies. One of these is the Common Desktop Environment (CDE) This is an X Windows, Window Manager developed jointly between IBM, HP, Sun and SCO designed to give a common theme and common set of basic applications across different platforms.

Getting Started

This section includes some of the basic information a new user needs when first logging in and getting around in a UNIX environment.
This assumes that you have a pre-installed system and that you have been given a username and password to login to the system.

UNIX machines are often used by lots of different people this differs from PC's which are often used solely by one person. To ensure that only those that are authorised to use the computer, and to ensure that files can only be read by people who they are intended to it is necessary to "login" to the computer. The most common method for logging into a computer is to give a username and password, until the correct userid and password is entered then no commands can be issued or files accessed. It is important that the password is kept secret as this is the basis of all security regarding your identity and files. Further details on the importance of usernames and passwords are provided in the Security section of this book.

There are a number of ways of logging in to a UNIX system. The two most common methods are mentioned, logging in directly onto a local machine or by connecting over a network.

Logging Directly into a local machine

When logging into a local machine, you will have a screen and keyboard attached directly to the UNIX computer. The Computer will normally be either text based or a Graphical Workstation. The login prompt should be already on the screen. This will prompt initially for your username which you will have been provided with, this is case sensitive and therefore “stewart” is not the same as “Stewart” (this will typically be all in lower case i.e. “stewart”).

```
AIX Version 4
(C) Copyrights by IBM and by others 1982, 1996.
login:
```

Type in your username
For a graphics login then hit the “TAB” key to move to the password field. For text logins hit the “ENTER” key to be prompted for your password.
You should then enter your password and hit “Enter”. Passwords will not show on the screen and are case sensitive.
You may be required to change the password (see the Security section for hints on choosing a new password).

When logging onto a UNIX system care should be taken when entering the password as it does not display and is very difficult to correct if you make a mistake. Sometimes you may see an ‘X’ on the screen for every key entered but not always. If you do make a mistake you could try pressing the “DELETE” key but this does not always produce the desired effect.

Depending upon the security settings of the machine you are trying to log onto you may only a few attempts at entering the correct password. More than the maximum attempts and your password will be revoked preventing you from logging in in future. If this happens you should contact the system administrator to get your password and login count reset.

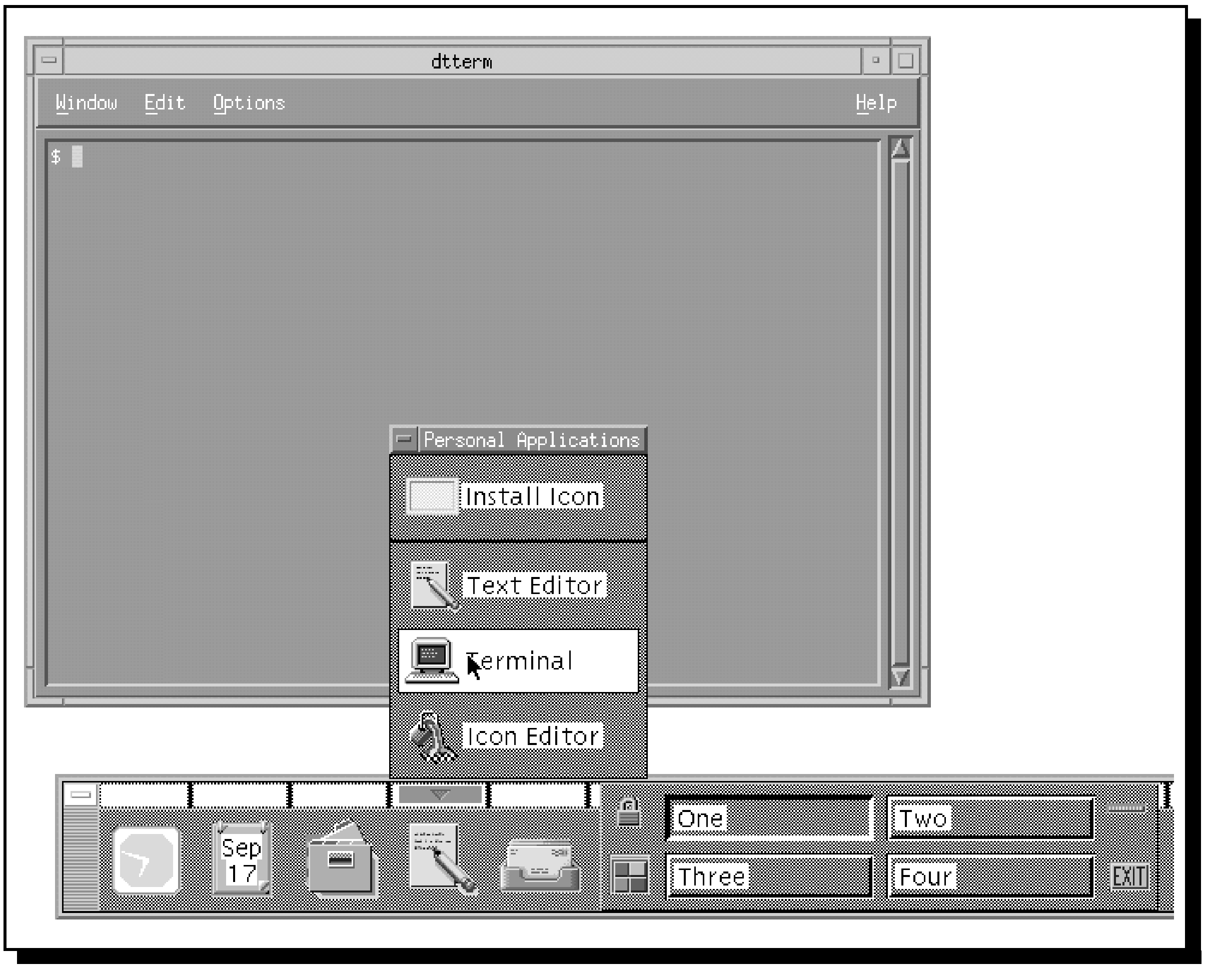
If you enter the username or password wrong little information will be given to the actual cause of the error. For example it may say “You entered an invalid login name or password” and prompt you to retry. This is deliberate as it makes it harder for any unauthorised people to try and login by guessing passwords, however this also makes it harder to see what you entered incorrectly. If this does happen then re-enter your username and password slowly and carefully to ensure that you enter them correctly.

```
AIX Version 4
(C) Copyrights by IBM and by others 1982, 1996.
login: wrong
wrong's Password:
3004-007 You entered an invalid login name or password.
login:
```

You should then get either a text screen for a text based screen or a Graphical Windows system if the system is set to use a graphical login.
If you are at a text screen then you are ready to proceed to the next section. If you have a graphical screen then we need to have an active “Command Window” to continue. If you already have a text window within the graphical display then this will probably be what is required if not then you need to start a “Command Window”.

```
Last login: Mon Aug 23 08:35:58 GMT 1999 on /dev/pts/0 from pc.mynet.com

$
```

The command window can be accessed by either right clicking the mouse button on the main part of the screen and selecting one of: Terminal; Command; Prompt; Term etc... or if you are running CDE it’s one of the ICONS available in the same group as the text editor. The Command Window should have a prompt on the screen this may be the ‘$’ sign as shown above or a ‘>’ or anything that the system administrator has set for you.
Once you have a text “Command Window” you can proceed to the next section.



```
$
```

Launching of Terminal window from CDE (terminal is open)

Logging in via the network

There are several ways of connecting to a UNIX computer over a network. I am assuming the TCP/IP protocol is running on the UNIX machine and that TCP/IP is running on a PC or similar from what you will be logging in from.

Depending upon the type of computer that you are connecting from it is possible to do different things. The standard available across all platforms is the ability to get a text command screen that allows commands to be run and text output to be seen. It is possible if using another UNIX type environment to redirect graphics applications from the remote machine to the computer that your connecting from. With Windows and OS/2 this can also be achieved by using emulation software such as Exceed which is developed by Hummingbird.

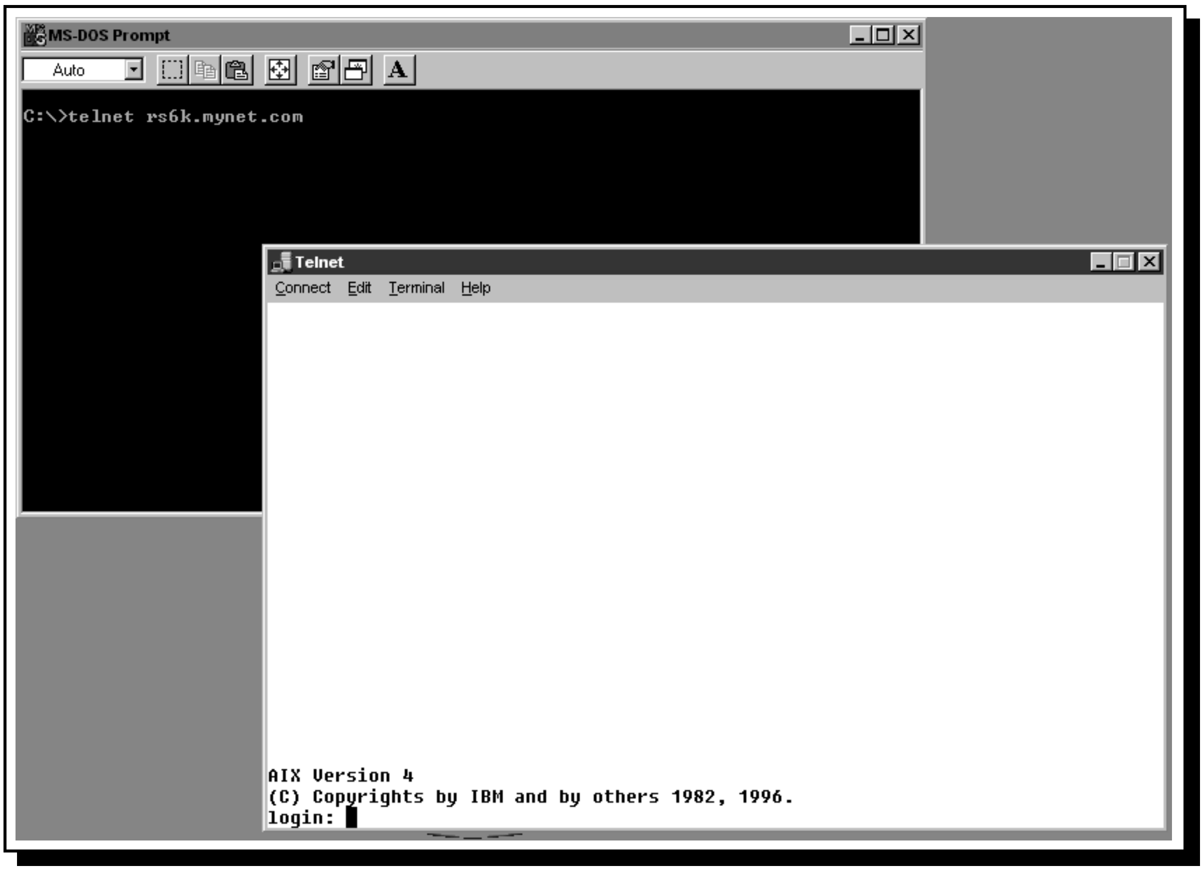
Whilst there are a few ways of connecting (e.g. rlogin, rsh, rexec, telnet) I will be using telnet which is probably the most widely available of all the methods.

Telnet is provided as standard on the following platforms: All types of UNIX, Linux, Windows 95 and later, OS/2 and even some "dumb terminals".

Telnet is started by issuing the command

```
      telnet machine
```

from a command prompt (MS-DOS prompt for Windows 95/98). Where machine is either the IP address or the hostname for the computer.



```
C:\>telnet rs6k.mynet.com
```

```
AIX Version 4
(C) Copyrights by IBM and by others 1982, 1996.
login:
```

Telnet from Windows 95

e.g. IP address 10.12.142.7 might have a hostname of rs6k.mynet.com . The hostname or address will be provided by the system administrator. The hostname will generally be the computer name followed by the network domain that it is connected in. See the Networking section for a further explanation of the TCP/IP protocol.

You should then get a login screen to the UNIX machine. This should show the name of the computer followed by a login prompt requesting a username.
You will have been provided with your username, this is case sensitive and therefore "stewart" is not the same as "Stewart" (this will typically be all in lower case i.e. "stewart").

Type in your username
Then hit the “ENTER” key to be prompted for your password.
You should then enter your password and hit “Enter”. Passwords will not show on the screen and are case sensitive.
You may be required to change the password (see the Security section for hints on choosing a new password).

When logging onto a UNIX system care should be taken when entering the password as it does not display and is very difficult to correct if you make a mistake. Sometimes you may see an ‘X’ on the screen for every key entered but not always. If you do make a mistake you could try pressing the “DELETE” key but this does not always produce the desired effect.

Depending upon the security settings of the machine you are trying to log onto you may only a few attempts at entering the correct password. More than the maximum attempts and your password will be revoked preventing you from logging in in future. If this happens you should contact the system administrator to get your password and login count reset.

If you enter the username or password wrong little information will be given to the actual cause of the error. For example it may just say “Authorisation Failed” and prompt you to retry. This is deliberate as it makes it harder for any unauthorised people to try and login by guessing passwords, however this also makes it harder to see what you entered incorrectly. If this does happen then re-enter your userid and password slowly and carefully to ensure that you enter them correctly.

```
AIX Version 4
(C) Copyrights by IBM and by others 1982, 1996.
login: wrong
wrong's Password:
3004-007 You entered an invalid login name or password.
login:
```

After entering your username and password correctly you will end up at a command prompt.

```
Last login: Mon Aug 23 08:35:58 GMT 1999 on /dev/pts/0 from pc.mynet.com

$
```

Exiting from UNIX

Just as it is required to login to UNIX it is important that after you have finished you should logout of the computer. This prevents anyone else from coming up to your terminal after you have left pretending to the computer that it is you that is telling it what to do.

This is done by entering

```
logout
```

from the command line. For a user at a graphical screen it may be necessary to click on the icon marked “EXIT” or one marked with an ‘ X’.

Another way of logging out of a logged in session / or of cancelling an application requiring further input is to use CTRL-D

Important Points for new users

There are a few oddities of the UNIX operating system that may catch a new user unawares.

Probably the most prevalent of these is that UNIX is case sensitive. With most other operating systems entering “telnet” would have the same effect as if you entered “TELNET” or indeed “TeLnEt”. This is not the case with UNIX where almost everything is case sensitive. It is most common for directories and files to be all in lower case. Unless you have a good reason to do it differently I’d suggest that it is a good idea to use lower case throughout.

A second difference is that the UNIX directory path separator is the ‘/’ (forward slash) as opposed to DOS, Windows and OS/2 which all use the ‘\’ (backward slash). This can be a constant point of frustration for someone that spends a lot of a time switching between UNIX and other operating systems in remembering to use the correct command directory path separator.

Depending upon you method of connecting to the computer some of the keys on the keyboard may not work as expected (or indeed at all). Some of the keys to be aware of are “BACKSPACE”, “DELETE”, and the arrow keys. If the “BACKSPACE” key doesn’t work then try the “DELETE” key and visa-versa. The arrow keys are not really needed on a UNIX system as most programs are written to accept normal keys as cursor keys.
These problems can sometimes be resolved by changing the terminal type or by re-mapping of the keyboard, which are described later.

Shells

If you’ve followed the instructions above you’ll be in a UNIX shell now. Whilst this is what most people see as being the UNIX operating system it is in fact a program that is running on top of the operating system. To take a basic view of how UNIX is built up see the diagram below:

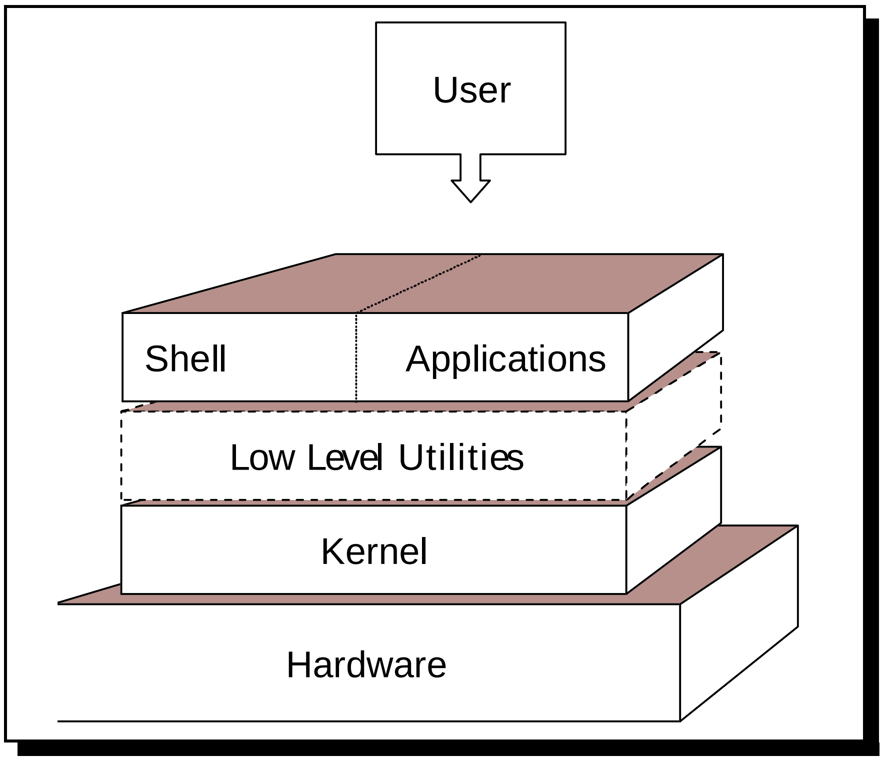


Diagram showing the software layers

The kernel is the heart of the operating system. This is a process that runs continuously managing the computer.
The kernel is a very specific task so to allow programs to communicate with it there are a number of low level utilities that provide an interface between the application and the kernel.
The shell is an application that allows users to communicate with the computer. It is a text based application that allows programs to be started and tasks to be run.

This can also be imagined likened to a nut (the type that does grow on trees). The kernel is the inter most workings, with the shell on the outside protecting the kernel against the user.

In the same way that different variants of UNIX were developed there are also different variants of the shell.

Here’s a list of the most common UNIX shells:

Name of shell	**Command name**	**Description**	**Installed by default on AIX**
Bourne shell	sh	The most basic shell available on all UNIX systems	☑
Korn Shell	ksh	Based on the Bourne shell with enhancements	☑ Default shell
C Shell	csh	Similar to the C programming language in syntax	☑
Bash Shell	bash	Bourne Again Shell combines the advantages of the Korn Shell and the C Shell	☒
tcsh	tcsh	Similar to the C Shell	☒

Table of common UNIX shells

On a standard setup you will normally have the Korn shell by default. You can switch to any other installed shell by entering the command name (e.g. entering csh will give the C Shell). The users default shell for a user is contained within the /etc/passwd or can be changed through SMIT (see section on SMIT).

The shell is more than just a way of typing commands. It can be used to stop, start, suspend programs and by writing script files it becomes a programming language in itself.

More details of the shells are listed below.

Bourne Shell - This is the oldest shell and as such is not as feature rich as many of the other shells. It's feature set is sufficient for most programming needs however it does not have some of the user conveniences that are liked on the command line. There is no option to reedit previous commands or to control background jobs. As the bourne shell is available on all UNIX systems it is often used for programming script files as it offers maximum portability between older UNIX machines.

Korn Shell - This is based on the Bourne shell. It's enhancements are particularly noticeable in it's command-line editing facility. It is possible using either vi or emacs keys to recall and edit previous commands. There are also more powerful programming constructs than the bourne shell, however these are not as portable to older machines. This is generally the default for AIX. Unless specified I will assume the Korn shell is being used for the rest of this book.

C Shell - The c shell takes it's syntax from the C programming language. As such it is a useful tool for anyone familiar with programming C. However some of it's syntax is not compatible with the bourne shell.

Bash Shell - The Bash shell is a combination of features from the Bourne Shell and the C Shell. It has a command-line editor that allows the use of the cursor keys in a more "user friendly" manner than the Korn shell. It also has a useful help facility allowing you to get a list of commands by typing the first few letters followed by the "TAB" key.

tcsh - This is a different shell that emulates the C Shell. It has a number of enhancements and further features even than the bash shell.

The Shell Prompt

When logged into the shell you will normal see one of the following prompts: $, % or #. This is an indication that the shell is waiting for an input from the user. The prompts can be customised but generally the last character should be left as the default prompt character as it helps to distinguish between what shell you are running and whether or not you are logged in as root.

The Bourne, Korn, and Bash shells all accept a similar syntax and unless you are using one of the advanced features you do not necessarily need to know which one of them you are in. If however you are in the C or tcsh shells this uses a completely different syntax and can require commands to be entered differently. To make it a little easier these have two different prompts depending upon the shell.

The default prompts are:

$	- Bourne, Korn and Bash Shells
%	- C Shell

When logged into the computer as root the user should take great care over the commands that are entered (further information about the root user is included in the user section). If you enter something incorrectly you could end up damaging the UNIX installation files or even delete all the data from a disk. For this reason the prompt is different when logged in as a root user as a constant reminder of the risks.

The default prompt for root is the hash sign # this is regardless of the shell being used.

Command Basics

UNIX commands are not necessarily the easiest commands to remember. They are designed to be short commands to reduce the amount of typing:

e.g. ls - List directory contents
cd - Change Directory

cp - Copy
pg - Show output one Page at a time

This can make it a little hard to remember but does save on typing when entering a large number of commands. The commands are also highly customisable by having a large number of options that can be entered on the command line however many other functions are provided through a pipeline. For example most programs will output to the screen without worrying about how many screenfulls they are. However all that is needed so that you can view more than one page is to pipe the output through the pg command. This is explained later.

Format of Commands

Generally most UNIX commands are designed to be run from a command line and accept a number of options or arguments when run. This allows commands to run without interaction from the user (often required on tasks designed to run in the background). These are usually in the format:

```
command option(s) argument(s)
```

an example of this would be the ls command. The ls command will be explained later, however for now it is sufficient to know that the ls command will list the contents of a directory and will accept certain options and arguments. One option is the -l which means provide more details about the files and another is -a which shows all files even hidden ones. The argument provided is a file or directory name.

```
ls -l /home/stewart
```

will show the contents of my home directory. The -l is an option in that it changes to way the program runs and the argument is /home/stewart which tells the program what directory to look in. The ls command doesn't require any options or arguments. For example the following are also perfectly valid commands.

`ls /home/stewart`	shows a brief listing of the directory
`ls -l`	shows a full listing of the current directory
`ls`	shows a brief listing of the current directory

Also if more than one option is required there are two ways of specifying them. Either individually as separate options i.e..

```
ls -l -a /home/stewart
```

or combined i.e.

```
ls -la /home/stewart
```

both the above will run identically.

Getting Help

The universal help tool in UNIX is the Manual Pages. These are accessed by using the “man” command. If you know the command that you want help on then enter man followed by the command will show the manual pages for the command.

For example to get help on the pg command type:

```
    man pg
```

this shows:

```
pg Command

Purpose

Formats files to the display.

Syntax

pg [ -Number ] [ -c ] [ -e ] [ -f ] [ -n ] [ -p String
] [ -s ] [ +LineNumber ] [ +/Pattern/ ] [ File ... ]

Description

The pg command reads a file name from the File parameter and writes
the file to standard output one screen at a time. If you specify a
- (dash) as the File parameter, or run the pg command without options,
the pg command reads standard input. Each screen is followed by a
prompt. If you press the Enter key, another page is displayed. Subcommands
used with the pg command let you review or search in the file.

To determine workstation attributes, the pg command scans the file
for the workstation type specified by the TERM environment variable.
The default type is dumb.
```

The manual pages cover more than just straight forward commands, they also hold subroutines that could have the same name as commands. Also there may be commands with the same name but different actions depending upon the aspect of the system it works on.

During the man pages it may refer to a different part of the man pages. The man pages are actually split into 8 different types. These can be accessed by putting the number before the command.

The format of this is

```
man x command
```

where x is the switch (see list later) and command is what your looking for help on.

Man command switches

1 User commands
2 System calls
3 Library functions
4 Devices and device drivers
5 File formats
6 Games
7 Miscellaneous
8 System and operation commands

Table showing Man sections

The man files are all held within the man directory. This is normally /usr/local/man

Typically each command registered with man will have the following information:

i.	Name	- The title and a one line description
ii.	Synopsis	- The syntax used
iii.	Description	- Information on the function and usage of the command. This normally includes examples.
iv.	Files	- Any associated files
v.	See also	- Any related information (other man pages)
vi.	Bugs	- Known Bugs or odd behaviour experienced

If you don't know what the name of the command is then you can use the -k option to find the commands that perform a certain function. For example if you need to find a directory listing you could try a search on the word list.

```
    man -k list
```

One of the pages shows the following [my highlighting]

```
lindex (n)           - Retrieve an element from a list
linsert (n)          - Insert elements into a list
list (n)             - Create a list
listalias (1)        - list user and system aliases
listbox (n)          - Create and manipulate listbox widgets
listen (2)           - listen for connections on a socket
llength (n)          - Count the number of elements in a list
locate (1)           - list files in databases that match a pattern
lrange (n)           - Return one or more adjacent elements from a list
lreplace (n)         - Replace elements in a list with new elements
ls, dir, vdir (1)    - list contents of directories
lsattr (1)           - list file attributes on a Linux second extended file
em
lsearch (n)          - See if a list contains a particular element
lsmod (1)            - list loaded modules.
```

the ls command is the one that we were looking for.

if this is the first time this has been performed then you may be told that you have to create the whatis database. This is done by issuing catman -w however this may need to be done by root to have the appropriate permissions.

```
$ man -k list

man: 0703-310 file man not found.
  Create the whatis database using the <catman -w> command
```

Another command similar to using man with the -k option is apropos. To search for a command that performs a list function try:

```
      apropos list
```

This needs the whatis database to have been created which can be achieved with the command:

```
      catman
```

Another way of obtaining help for what a command is or what it does is to use the help switch. The most common switch is --help, other switches in common use include -? and -h For example

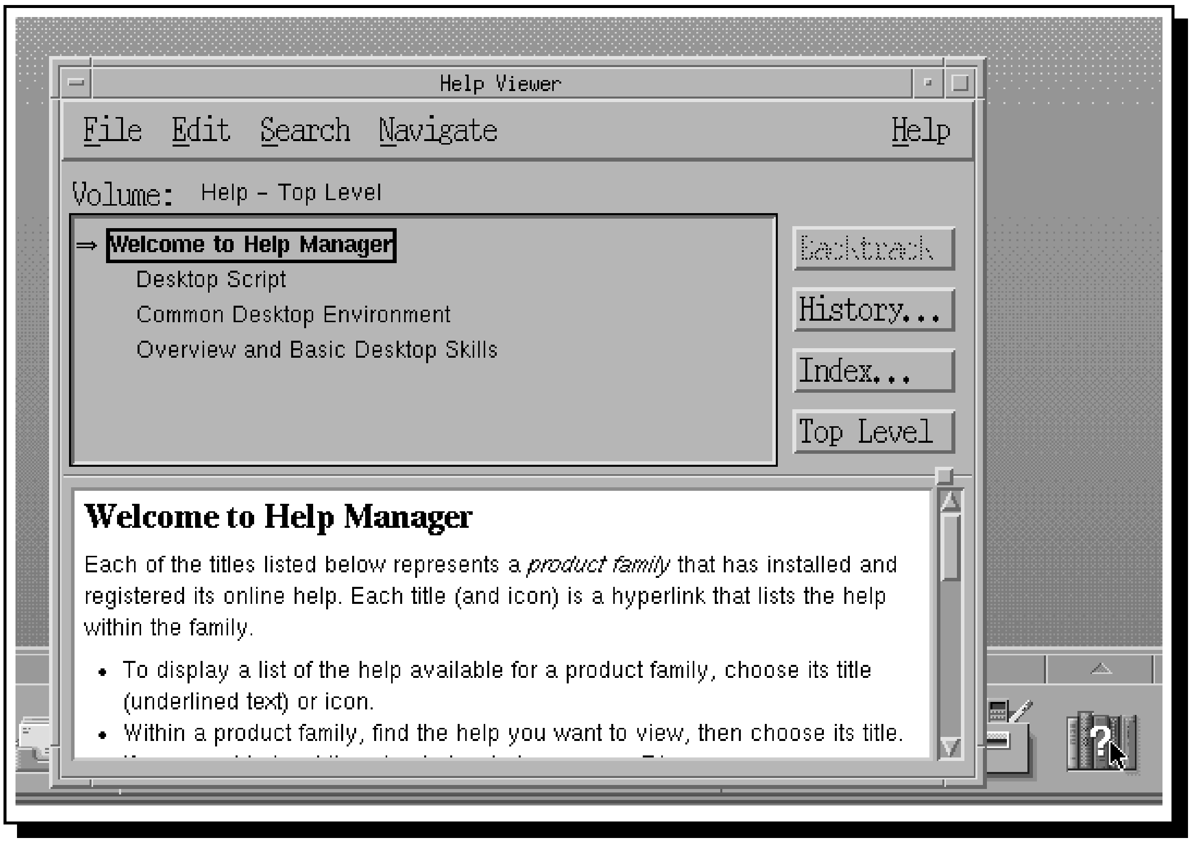
```
      pg --help
```

would show the command options available for the pg command. The --help option is not necessarily as comprehensive as using the man pages. Often the --help switch will give a list of the available options only, compare this to the man display earlier.

```
$ pg --help
Usage: pg [-Number] [-p String] [-cefns] [+LineNumber] [+/Pattern/]
 [File...]
```

The --help option is the older standard that is usually implemented. However some commands support a -h option instead.
Most commands will support both the above however if one doesn't work you could try the other which may work.

If you are running CDE there is also a further source of help in the form of the CDE help viewer (dthelpview). This is launched by clicking on the icon indicating books and a question mark normally in the bottom right hand corner next to the Trash Can.
If you get an error message saying that no applications have registered with the viewer then you may need to install the CDE help files. This can be done using the easy install within smit, using the standard install disks, select Personal Productivity and then CDE Help Files.


Help Viewer
File Edit Search Navigate Help
Volume: Help - Top Level
⇒ Welcome to Help Manager
Desktop Script
Common Desktop Environment
Overview and Basic Desktop Skills
Backtrack
History...
Index...
Top Level

Welcome to Help Manager

Each of the titles listed below represents a *product family* that has installed and registered its online help. Each title (and icon) is a hyperlink that lists the help within the family.

- To display a list of the help available for a product family, choose its title (underlined text) or icon.
- Within a product family, find the help you want to view, then choose its title.

CDE Help Viewer the cursor is over the icon used to launch it

Finding Files and Commands (find)

Sometimes you may have an idea what a command is called or the name of a file, however not whereabouts it is on the disk. The find command will scan the directory structure looking for files that match a certain search requirement.

The command is specified by specifying a directory and then searching all directories underneath that.

The most common way of using the find command is to search for a program or file by it's name. To search the entire computer for a file called filename the following command would be used.

```
find / -name filename
```

Further information can also be searched for such as the creation date of the file or it's file owner.

A few useful commands

Here are a few commands that may come in handy.

Clear

Typing clear from the command line will clear the terminal screen. This is useful if you've completed a command that has left a lot of text on the screen and you'd like it cleared so that you can see what happens with your next command.

Echo

The echo command just echo's whatever you type onto the screen.

```
$ echo This is a message to show on the screen
This is a message to show on the screen
$
```

Whilst this command might not appear to be very useful when combined with other commands via pipes or used in scripts it can indeed be very useful.

Files and Directories

A “classic” file is a collection of data located on a portion of a disk. The reason that I say a classic file is that in UNIX files are often used as a representation of a device or to refer to parts of the operating system. There are 3 types of files

- **Ordinary** - This contains data. One form of file is a text file however this category also includes files created by applications or indeed the applications themselves.
- **Directory** - This is a special kind of file that acts like a table of contents allowing the organisation of files (see later in this section).
- **Special Files** - UNIX operating systems allow you to access logical devices by creating virtual files that refer to them. These are found in the /dev directory. For example /dev/tty1 refers to a tty screen, /dev/cd0 refers to a CD-ROM etc. One particular file is /dev/null which is a pointer to nothing. You may wonder why you would want a file that didn’t point at anything, however this can be useful where you get an output that is not needed. Redirecting the output to /dev/null will cause it to disappear.

The number of files on a directory system can be thousands. To have these in one place would make it very difficult to find what you wanted. Searching a list of thousands of files would be equivalent to searching for a needle in a haystack. The files are therefore collected together by commonality. These are put into groups called directories. To make it even easier these are sorted into a hierarchical tree. Moving down the hierarchical structure allows you to find more specific files.

The files and directories are referred to by filenames. The filenames can be quite long (up to several hundred characters with some implementations). They can contain any letters, digits and punctuation (including spaces). It is however better to avoid any special characters and especially spaces. This is particularly important in a command line environment like UNIX where it is necessary to interpret the differences between a single filename with spaces and two separate parameters. See creating a new file for more rules on filenames.

The directory structure starts at the root directory which is known by a forward slash ‘/’. Each directory below the root has it’s own name and is separated by the forward slash ‘/’. Below the root directory are a number of “Top level Directories” Common top level directories include home; usr; var; tmp and etc

These are used for the following purposes:

home - This directory is used as a starting point for users home directories. These are directories that are given to users for them to store any files they create. The users directory is a subdirectory to home normally the same name as the username. e.g. /home/stewart

usr - This directory is used to store installed user applications.

var - This is used for files of variable length. Files that change a lot such as log files would be kept here.

tmp - If used this directory is used for temporary files that are only needed for a short period of file. Some systems use housekeeping jobs to periodically delete files in this directory so important files should not be stored here.

etc - This directory holds many of the configuration files used by applications.

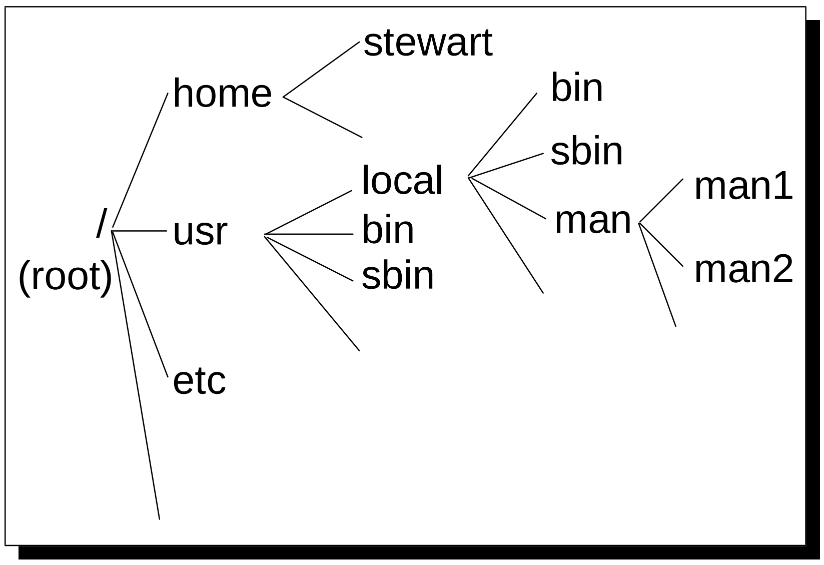
bin or sbin - Whilst you may see bin or sbin directories at the top level these are normally symbolic links to the directories /usr/bin or /usr/sbin respectively.

There are two other top level directories that are used as starting points for other directories.

dev - This is used to hold the devices available on the system. For example the first hard disk (hdisk0) can be referred to as /dev/hdisk0

mnt - This is the mount point which is used as a convenient place to attach other devices such as floppy disk drives and CD-ROM Drives. For example the CDROM drive may be mounted as /mnt/cd0/ although the name does not have to be the same as the device name.

These then subdivide into further directories which can be represented as a tree.



Tree representation of UNIX directory structure

The above tree is far from complete there can actually be over a hundred directories on a UNIX installation.
The full directory path name starts with the root directory (/) and move up the tree with each directory separated by a forward slash. For example the man2 directory would have a full path of
/usr/local/man/man2

Relative Directories

So far we've been looking at the directory name in full this is known as the "absolute directory". As well as being able to specify the directory in full (starting with root) it's also possible to specify a "relative directory". To specify the relative directory start from the current position and specify the rest of the directory name. The difference between an absolute and relative directory is that an absolute directory always have a '/' at the start of the path whereas a relative directory will not. If already positioned at the directory /usr/local the man2 directory can be referenced by man/man2

Special Directories

There are 2 special directory names that apply to all the subdirectories. The current directory is marked by a single dot (.) whereas the further up can be referenced by a double dot (..) .

The man2 directory can be referenced from the /usr/local/bin directory with the relative path ../man/man2

Moving about the directories (cd)

When first logging onto a UNIX system you will normally be in your “home” directory. This is normally /home/*username* (for username stewart that will be /home/stewart).

You can check the current directory at any time by issuing the pwd command (this stands for print working directory).

You can move around the directories by using the cd (change directory command). To change directory use cd followed by either the absolute or relative directory (as detailed earlier).

Here’s a few examples of how to change directory

Command	**What it does**
cd	Moves to your home directory. Normally /home/username
cd /	Moves to the root directory
cd /usr/local/man	Moves to the man directory (absolute path) this will work no matter what directory you are currently in
cd local/man	Moves from the current directory (relatively). If you are currently in the /usr directory this command is identical to the above command.
cd ..	Moves up a level. If you were at /usr/local/man you would now be at /usr/local

Table showing a few examples of moving around the directories

Listing the contents of the directories

So far we have moved around the directories, however we have not looked at the files stored, or indeed seen how to find out what directories exist. This is done using the ls command.

Typing ls on it’s own will list all the files and directories contained within the current directory.

```
$ cd /home/stewart
$ pwd
/home/stewart
$ ls
docs         readme.txt   smit.log     smit.script
```

Here I’ve moved to the usr directory. Displayed the current working directory and then listed the contents of the directory. In my home directory are 3 files and one subdirectory. You cannot tell for certain from the view above (although you could guess from there names) which are the files and which is the subdirectory.
One way of telling is to perform a ls with the -l option which shows a lot more information.

```
$ ls -l
total 6177
drwx------   2 stewart  users        512 Sep 16 17:42 docs
-rw-------   1 stewart  users        124 Sep 16 17:26 readme.txt
-rw-------   1 stewart  users    3156558 Sep 16 17:25 smit.log
-rw-------   1 stewart  users       3505 Sep 16 17:25 smit.script
```

Using this display you can tell that docs is the directory.

As far as UNIX is concerned a directory is just a special type of file so is displayed along with the other files. From the mode display (this is the 10 characters at the left of each file displayed) you can see what type of file it is. The remaining 9 characters show the file permissions however these will be dealt with later.
There are seven different file types for AIX which are listed in the table below.

File Type Letter	Description
-	Regular file
d	Directory
l	symbolic link
b	block special file
c	character special file
p or s	other special files (not covered)

Table of different file types in AIX

A regular file could be an application file or a text file etc.
A directory is a file that holds other files (including other directories)
A symbolic link allows a file to be redirected to another.

A block and a character file will be explained later.

We can also see that docs is the directory by going into it and displaying the contents.

```
$ cd docs
$ pwd
/home/stewart/docs
$ ls
file1  file2
```

Here's an explanation of the Listing format shown earlier with the ls -l option.

```
    (1)     (2)  (3)     (4)        (5)      (6)        (7)
drwx------   2 stewart  users        512 Sep 16 17:42 docs
-rw-------   1 stewart  users        124 Sep 16 17:26 readme.txt
-rw-------   1 stewart  users    3156558 Sep 16 17:25 smit.log
-rw-------   1 stewart  users       3505 Sep 16 17:25 smit.script
```

1. File / Directory and permission bits
2. Link Count
3. Username of person who owns the file
4. Group name for which group privileges apply
5. Character count of the entry
6. Date file was last modified
7. Name of the file/directory

Referring to files within a directory

A file can be referred to by just it's filename, it's absolute directory and filename, or by a relative directory and filename. For example file1 can be referred to in the following ways.

From /home/stewart/docs	file1	(filename)
From anywhere	/home/stewart/docs/file1	(Absolute)
From /home/stewart	docs/file1	(relative)

There is also another way of referring to the current directory using a single dot '.' . This is particularly important when running a program. Some programs that haven't been installed in the standard program directories need to be run by specifying it's directory and filename. The directory can be absolute or relative however when the file is in the current directory the filename alone is not sufficient. To get around this the file is referred to using the dot directory. For example is there is a program called exec in the current directory this would be run by using ./exec

Lost + Found

There is a special directory name called lost+found. Sometimes when the system crashes it may “lose” a file. If this happens when the system runs it’s checks it will store them in the lost+found directory so that any information held in them can be recovered.

Making a new directory (mkdir)

New directories can be created using the mkdir command. Directories can be created either singularly (in which case all directories above it must already exist) or several at a time allowing parent directories to be created at the same time.

Creating a single directory from the current directory is done by mkdir followed by the name of the new directory. For example:

```
mkdir newdir
```

The file directory could also have been created using a full path however all the previous directories must already exist. If the previous directories don’t already exist then using the -p option will also create the parent directory. For example:

```
mkdir -p /home/stewart/newdir1/newdir2
```

Making a new file (touch)

A common way of making a file is to save to a file from an application (see the vi section for details of using a text editor). It is also possible to create a blank file using the touch command. This will create a blank file. The format is to have the filename after the touch command. For example:

```
touch newfile
```

The following rules should be followed when creating a file:

- The name should be descriptive of the comments
- Only use alphanumeric characters i.e. Uppercase, lowercase, numbers, #.@-_
- Although spaces can be included in a filename it is strongly advised against this
- Should not include shell metacharacters (characters used by shell pipes etc)
  *?<>/&;:![]$\’”
- Should not begin with + or -
- Should not be the same as a system command
- Filenames starting with . are hidden (cannot be seen from an ls without the -a option)
- Less than 255 characters

Removing a directory (rmdir / rm)

The safest way to delete a directory is to first remove all files from the directory. Then check the directory is empty by issuing ls -a (the -a will show all files even if they are hidden). Then change to the directory above and type

```
rmdir dirname
```

The rmdir command will only work when the directory is already empty which provides a little protection against accidentally deleting files.

For a more cavalier way of removing a directory or multiple directories the rm command can be used. To do this type rm -r followed by the directory name. For example:

```
rm -r dirname
```

Warning: This is a very dangerous command.
The rm -r command will remove all directories below the one specified.
Be especially careful if logged in as root as this can delete the entire data on the disk.

There is no way of undeleting files that are deleted by mistake.
The only way or restoring the file if deleted accidentally is to copy it from the latest backup disk (if available)!

Removing a file (rm)

Files are removed by using the rm command. To delete the file type rm followed by the file name (can include a path).

There is no way of restoring a deleted file (except from a backup copy) so be careful when deleting files. A slightly safer option is to use the -i option which will prompt before deleting each file. This is particularly useful when using wildcards (see later). It is possible to setup your profile so that you always use the -i option (see alias).

Moving / Renaming a file or directory (mv)

Files or directories can by moved or renamed by using the move command (mv). To move a file enter the mv command followed by the filename and then followed by the new directory. To rename a file or directory enter the mv command followed by the old name and the new name.

You can use the path along with the filename.

Using the -i option will prevent accidentally overwriting an existing file.

Copying a file (cp)

Files can be copied using the cp command. Enter mv followed by the existing file and then the new file (can include paths). Putting the 2nd parameter (new file) as a directory name will copy the file to another with the same name in the new directory.

Using the -i option will prevent accidentally overwriting an existing file if it exists.
Wildcards can also be used to copy multiple files into different directories.

There is also a cpio command that is more versatile and allows the copying of directories and files within them. View the man pages for more information on the cpio command.

Viewing the contents of a text file (cat)

You can view the contents of a text file using the cat command. Whilst cat may seam a strange name for a command the view a file, it’s because the command can be used to combine two files into a single one (the name comes from concatenate).

To view a file using cat

```
cat filename
```

you should not use this on a binary file as the output can do strange things, including sounding the speaker or accidentally re-mapping the keys.

Other commands that can be used including more and pg (see later) however cat is useful for automating commands as it does not need any interaction from a user.

Viewing the beginning / end of a text file (head / tail)

Sometimes you may want to look at the first few or last few lines of a file. For the first few lines you may need to look at the top of a file and take a different action depending upon the type of file, An example of needing to see the last few lines would be a sequential log where you want to view the last few entries rather than having to go through all the entries to reach the bottom.

The commands to do these are the head and tail commands for the top and bottom of the file respectively.. The commands have a number of options allowing you to specify a certain number of lines or start from a particular place in a file however the easiest way to use it is just to enter

```
head filename
```
or
```
tail filename
```

which will display the first / last 10 lines of a file.

Checking the type of file (file)

If you try and view the contents of a none text file you will often get garbage on the screen. Sometimes this will actually prevent you from using that terminal by changing the mapping of the keyboard or the screen. It is therefore a good idea to check what an unknown file is before viewing the contents.

One way of doing this is using the file command. Entering

```
file filename
```

will tell you if the file is a text file, a command file or a directory etc.

Printing a file

To put a file on the print queue use the following command.

```
qprt filename
```

AIX also supports the following commands for compatibility with other versions of UNIX.

AT&T

```
lp filename
```

BSD

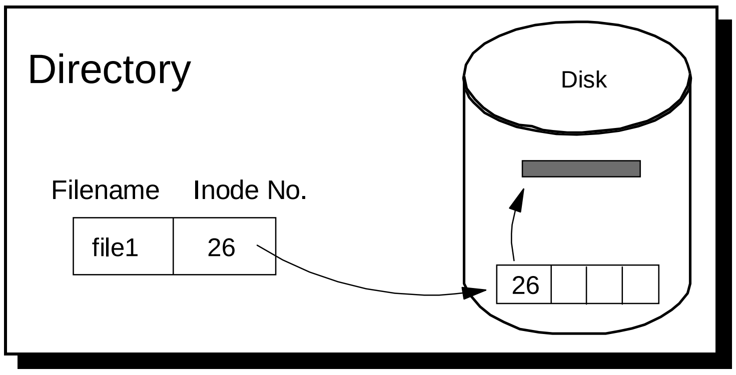
```
lpr filename
```

File Structure (Inodes)

Each file on disk has two portions. The Inode which describes the file and the data blocks that actually hold the data.

The following information is kept within the Inode:
❍ Size of file
❍ Permissions
❍ Date & time of creation
❍ Date & time of last modification
❍ Link count

The directories themselves are made up of links to the Inodes as shown below;



Directory structure

File Permissions

From the ls listing format we saw the file permissions at the left of the output. On a windows PC anyone can access any of the files on the disk. As a UNIX computer is generally used by more than one person it is important to control who can access what files. For example you would not want someone else to read your private files and you certainly wouldn't want anyone to change them, whether by accident or intentional. Also there may be files that you do want some people to read/update but not everyone. For example you may have a file with departmental information in that other people in the same department may need to update, but anyone not in the department should not be able to. There are also files that you don't mind anyone being able to read. For example you may have a list of useful phone numbers that everyone should be able to look at to see who to contact about their particular problem.

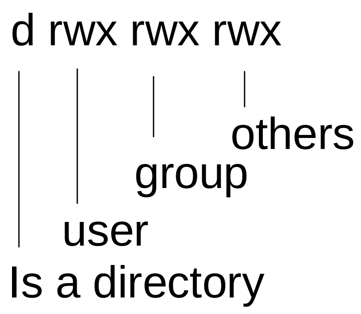
For the reasons above file permissions are split into 3 different categories. These apply to:

- user - the owner of the file
- group - a group of people, e.g. a project team or department
- others - anyone else that has a login to the computer

these are then split into 3 different permissions, that of being able to:

read - Look at the contents of a file / find out what files are in a directory
write - Change or delete the contents of a file / create or remove files in a directory
execute - Can execute (run as a program) a file / can change to the directory or copy from the directory.

These are laid out as follows (note these are the first 10 characters of the ls -l display):



If the entry is filled in then it has affect if it is dashed out '-' then it does not apply.

For more information on users and groups see later. There are also further permissions that can be set, however these are more advanced and are explained later. Also note that root can override most of the permissions.

Changing File Permissions (chmod)

Assuming that you are either the owner of the file or root it is possible for you to change the permissions of a file to either add or remove permissions. This is done using the chmod (**ch**ange **mod**e) command.

The chmod command can be used in one of two ways. The Symbolic Format or the octal format. Symbolic is useful for new users as it is easier to use, however once the octal format is learnt it can be a powerful and quick way of changing file permissions.

The basic format is

```
chmod mode filename
```

It is only the format of the mode parameter that is different.

In Symbolic format permissions are added or deleted using the following symbols

u = owner of the file (user)

g = groups owner (group)
o = anyone else on the system (other)

+ = add permission
- = remove permission

r = read permission
w = write permission
x = execute permission

For example to add write access to the group the following command is used:

```
chmod g+w file1
```

In Octal format the mode is based upon a octal number representing the different mode permissions, where each of the permission groups (user, group, others) has an octal value representing the read, write and execute bits. This requires a little bit of knowledge on binary and octal number bases.

	User	Group	Others
Symbolic	rwx	rw-	r--
Binary	111	110	100
	4+2+1	4+2+0	4+0+0
Octal	7	6	4

File Attributes Symbolic and Octal

The above file would therefore have the octal number 764 and would therefore be changed using the command

```
chmod 764 file1
```

A basic way of working this out is to add the following numbers depending upon the permission required.
Read = 4
Write = 2
Execute = 1

Therefore if you wanted to set read to yes, write to no and execute to yes, this would be 4+1=5

Changing the file owner (chown)

It is possible to change the owner of a file however normally this requires the user to be logged in as root. The owner or group can be changed using the chown (**ch**ange **own**er) command.

The format is as follows:

```
chown user:group filename
```

Basic Mail Handling

Electronic Mail Handling is built into AIX using the “mail” command.
The mail command provides only basic functionality so if you have a better e-mail system you may wish to use that instead. Some examples are:
pine
emacs
dtmail (included in the CDE).
or Netscape Message Centre.

However if you don’t have one of the above installed, indeed on a UNIX computer that’s used purely as a server to handle a specific function this is likely to be the case, then you can use the built in mail handling commands.

To check for new mail enter the command

```
$ mail
Mail [5.2 UCB] [AIX 4.1]  Type ? for help.
"/var/spool/mail/stewart": 2 messages 2 new
>N  1 another           Fri Dec 31 13:06  10/376  "Hello"
 N  2 another           Fri Dec 31 13:06  9/330   "hello2"
?
```

Here the inbox contains 2 messages both from user another one titled Hello and the other titled hello2.
To read the message press t (it will show the current message which is marked with a > sign).

```
? t
Message  1:
From another Fri Dec 31 13:06:44 1999
Date: Fri, 31 Dec 1999 13:06:44 -0600
From: another
To: stewart
Subject: Hello

This is a mail message from another to stewart

?
```

entering d will delete the message r will reply to the message, n will move on to the next message, h will show the subjects of the messages etc. Typing a ? followed by enter will show a list of common commands. Pressing q will quit from mail.
To send a message type mail followed by the name of the user that you wish to send to.

```
$ mail another
Subject: Hello
Thank you for the message you sent. Hope you are well.
Cc:
$
```

After filling in the subject press enter, after filling in the content of the message press CTRL-D.

One thing that you might notice is that after you’ve read a message and then exited from mail it will save message saved. However when going back into mail the message will not show. The reason for this is that once read all messages are moved from the normal mailbox to a personal mailbox. To see messages that have been moved to your personal mailbox run the following command:

```
mail -f
```

Users

Whilst there are no international naming conventions for usernames your company may have it’s own naming convention (examples usernames could consist of: first names; last names; last name + initial; first name + 1st letter of surname; personnel number etc.). The users on the system are known by a userid. This is normally actually a number as far as the system is concerned, whereas it is known by the username as far as most people are concerned. The words username and userid are often used interchangeably.

There are some default userid's on all systems when first installed. Other than the root username however these can normally be disabled from logging in. You should however be more careful about deleting usernames as sometimes these are used by different tasks running on the system.

Root

The root user is the most powerful user on the system. This is considered to be the username of the system administrator who has ultimate control over the computer. As root there is nothing that you are not able to do as far as the local computer is concerned. For this reason it is important that the root password is not compromised. The password should be changed regularly to ensure that it is not compromised.

Great care should be taken whenever logged in as root. If you are not careful it is possible to accidentally do a great deal of damage and often there is no warning to protect vital information.

It is good practice not to actually log onto the computer directly as root and instead to login under a normal username and switch user (su) to root. This is done by entering su then enter and then entering the password for root.

Querying users on the running system

You can check which userid you are logged in as

```
whoami
```

This is a basic command that just shows the username that you are actively using. There is a command that provides more information which is run by typing

```
who am i
```

This will show the username, the terminal and the initial data and time that you were logged in at.

Running the id command will show the userid (uid) and group id (gid) that is being used.

You can also query the terminal that you are currently logged in on by typing tty. If you are connected to a terminal then it will show the terminal that you are logged in on, or if connected over a network (e.g. telnet) then it will show the pseudo-terminal which you are connected in over.

If you were connected to a physical terminal the output will be along the lines of /dev/pts/0. If however you are on a pseudo-terminal it would probably show as /dev/tty1.

The type of terminal you are emulating is shown by the $TERM variable. Even some dumb terminals can have half a dozen or so different terminal types that they can emulate. You can check the terminal type by echoing the value to the screen.

```
$ echo $TERM
  vt100
$
```

In the above example the terminal type is set as vt100 which is a common terminal type. The Windows 9x telnet application can support the vt100 terminal type. However by default the Windows 9x telnet will give ANSI which is not as well supported by AIX. To change to vt100 the following can be entered on the terminal.

```
$ TERM=vt100
$ export $TERM
$
```

The first line sets the terminal variable and then export makes it available for use by the terminal. This can also be included in the .profile to ensure it is run automatically if needed (and you always login from an appropriate terminal).

If you were using the C shell however you should use “set term=vt100” instead. You can check the terminal type by typing “set”.

Looking at other users

Typing users at the command line will give a list of the logged in users.

The “who” command will provide more information by listing the terminals and dates and times similar to issuing a “who am i” command.

The finger command is another useful command for getting information about others users. For security reasons the finger command is sometimes disabled so don’t think that you’re typing the command incorrectly if it doesn’t work. The finger command will display some or all of the following:

- Login Name
- Users Full Name
- Terminal write status (allows direct communication)
- User’s Idle Time

- User's Login Time
- Plan information (in .plan file)
- Project information (.project)
- User's office information
- User's phone number.

This all depends upon what files are configured and what information is held in the password file. To try this out enter finger followed by the username.

```
finger stewart
```

You may also be able to check a user on a different machine, by combining it with the hostname.

```
finger stewart@rs6k.mynet.com
```

The write status is set by using the mesg command. By turning the message status on or off you can choose when to accept messages or not. If messages are turned on you can communicate with them by using "write" or other similar commands.

You can also see some more information about users by looking in the password file. Whilst it may seam a little dangerous the password file is readable by everyone on the system. This does not produce as big an exposure as it first sounds as all passwords are encrypted, even further on most systems the passwords are now normally in a shadow file rather than in the traditional passwords file.

However by looking in the passwords file you can see all the usernames and further details on all the users on the system. To view the passwords file type

```
pg /etc/passwd
```

This will show the following details:

- Login name
- Encrypted password
- Numerical user identification
- Numerical group identification
- User information field
- User's home directory
- Shell program to use at sign-on

Changing the Password

To change your own password you should run the passwd command. This will request the current password followed by the new password which will need to be entered twice.
You may however be made to change your password when logging into the computer. This depends upon the password expiration time set against your userid. Your passwords will not be shown on the screen when typing them in.

If you have sufficient authority it is also possible to change another users password by entering passwd followed by the username of the id you want to change.

Switch User (su)

One of the powerful features of UNIX is the ability to change userid when logged into a system. This command su is sometimes referred to as superuser, however this is not completely correct. In the early days of UNIX it was only possible to change to the root user which made for the superuser command however it is now possible to change to any user using the su command.

The primary use of the su command is to switch to root and this can be achieved by typing su. You will then have to enter the password for root.
Sometimes this is the only way of logging into root over a network. For security reasons it is a good idea to prevent logging as root remotely. Also by using su a log entry is made of who was running as root at a certain time.

It is also possible to change to another user by putting the username after the su command. There are two ways of switching users. By putting a '-' after the command will cause the users profile to be read and variables to be set. Without the '-' the previous users settings will still remain.

To use the new users profile and variables

```
su - username
```

To continue with the current profile and variables

```
su username
```

you can then return to the previous user by entering exit.

Making the most of UNIX commands

Whilst the number of options on each UNIX command may seam overwhelming at first this is part of what makes UNIX so powerful. Another of the features that makes UNIX so powerful is the ability to combine several commands to make them more useful. This can be achieved either by stringing commands together on the command line or by bundling the commands together into a script file which can range from something very trivial to a program in it's own right.

Using command switches

The most basic way of extending the functionality of a command is to try some of the switches available.

These can be found using the man command as mentioned in an earlier section.

The simple example of this is the ls command. Without any switches all that is displayed is a list of all the files in the current directory.

```
$ ls
docs         readme.txt   smit.log     smit.script
```

By adding the '-l' option more information is provided.

```
$ ls -l
total 6177
drwx------   2 stewart  users        512 Sep 16 17:42 docs
-rw-------   1 stewart  users        124 Sep 16 17:26 readme.txt
-rw-------   1 stewart  users    3156558 Sep 16 17:25 smit.log
-rw-------   1 stewart  users      3505 Sep 16 17:25 smit.script
```

The pipe command (|)

There are a number of features missing from some of the commands. For example there is no way of viewing the output a age at a time or of sorting the output into a certain order. Whilst this may sound like a large disadvantage at first it is not really a problem as there are a number of commands that are specially designed to perform just them functions. By restricting these tasks to other commands it makes the commands simpler and the number of options down.

The way this is achieved is by 'piping' the standard output into another command that is specifically designed to perform the certain function. The pipe command is a vertical bar '|'. This is normally found at the bottom left of the keyboard and is typed by using shift and the '\' key.

The first command is entered first followed by the pipe command and then followed by the second command. Any output from the first command is then used as input to the second command.

For example to sort a basic directory listing by name the ls command is piped through the sort command.

```
ls | sort
```

The output can be redirected through any number of pipes each one changing the output in someway. The full command is referred to as a pipeline.

Redirecting stdout, stdin and stderr (> <)

The output from a command goes to the standard output (stdout) which is normally the screen. Whereas the input is normally taken from standard input (stdin) which is normally the keyboard. To automate processes it is sometimes required to change this so that for example the output from a command is routed to a file or the printer. This is done by redirecting the stdout and stdin streams.

The output from the ls command could be redirected to a file in this case called dirlist.txt

```
ls > dirlist.txt
```

If the file dirlist.txt already exists it will be deleted. It is also possible to append the output to the end of an existing file by using >> instead of >.

For example

```
echo “This is the next line of the log” >> log.file
```

Whilst you may see all the output from a command on a single screen this is not all neccessarily coming from stdout. There is also another data stream called standard error which by default is directed to the same screen as stdout. This data stream is used to send messages regarding any error messages. The advantage of having this as a separate stream is that even if you redirect stdout, because you are not interested in the output or just want it for reference you will instantly see any error messages on the screen. However if this is a command running automatically without user interaction then there will not be any one to see messages put on the screen. The standard error data stream can therefore be redirected the same as stdout by prefixing the redirect by the number 2 digit. In fact the stdout data stream should be prefixed by the number 1 digit however this is dropped to save typing. To therefore redirect any error messages to an error.log file and the normal responses to a log file the following would be used.

```
command >log.file 2>error.log
```

The single >'s can be replaced by double >>'s if you would like the output to be appended to the file rather than to overwrite the file.

It is also possible to write both stdout and the standard error stream to the same file. This is not simply a case of using the same file name in the above command as you might expect. The reason for this is that a file can only be opened for writing by 1 process at a time. The two redirects are two different processes and would not allow both streams to write to the same file. This can however be achieved by redirecting the error data stream to the stdout data stream using 2>&1. Which now gives:

```
command >output.file 2>&1
```

In a similar light you cannot use a file used as input to the command to redirect the output to. For example it is not valid to issue the following command

```
sort file1 >file1
```
This is not valid

instead the output would have to be redirected to a temporary file and then renamed to the required name.

```
sort file1 >/temp/tmp$$
mv /tmp/tmp$$ file1
```

The file ending in $$ will actually be created by the system with a unique number. This is useful for temporary files as it prevents you overwriting a temporary file in use by a different process.

These redirects are fine for use in batch programs that are run without anyone monitoring the system. Sometimes it is necessary for someone to monitor the output of the command but also for it to be duplicated into a file for logging purposes or to perform further processing on.

The tee command is used within a pipeline and whatever input it receives it both puts into a file and forwards on the pipeline.

```
command | tee file1
```

The line above will take any output from the command and put a copy in file1 as well as sending it to the screen. This could be combined further by allowing the output to be further processed by another command. The tee command can be used any number of times in a pipeline like this.

```
command1 | tee file1 | command2
```

If you want tee to append to a file rather than overwrite it the -a option is used.

The same basic redirect can also be done in the reverse direction in that an interactive program that requires input from a user can be automated. For example with an interactive program such as ftp (file transfer protocol). The ftp program allows files to be transferred from one computer to another over a network. This however needs a user to type the commands in to transfer the file. Instead the commands should be entered into a text file the same as how they would be entered from the keyboard. The file is then directed into the program in place of the stdin.

```
ftp rs6k.mynet.com <commands.txt
```

Redirecting to a file can have an unfortunate consequence if the file already exists and does not want to be replaced. By using the redirects incorrectly it is possible to accidentally overwrite an important file. In the korn shell there is an option that allows us to prevent overwriting files by mistake. This is the noclobber option and is set by typing

```
set -o noclobber
```

If an attempt is now made to overwrite a file the shell will issue an error message and prevent the file being written to. The no clobber option can be turned off using

```
set + noclobber
```

If required the noclobber option could be put an a users .profile to have this set automatically.

File Descriptor Table

The use of stdin, stdout and stderr is possible using only the single less than / greater than signs because of the way that processes are assigned to a file descriptor table.

The file descriptor table is a list of numbers relating to open files. The first 3 files to be opened are stdin, stdout and stderr, these are numbered 0 for stdin, 1 for stdout and 2 for stderr. Therefore stdin and stdout can be referred to by < and > respectively (no further filename) whereas stderr requires 2> to ensure it is output stream numbered 2 that is to be redirected.

Viewing output one page at a time (more and pg)

As mentioned earlier one of the features of UNIX commands is that they are restricted to a specific task only. One of the things missing from many of the commands is the ability to list the output of a command one screen full at a time. There are instead commands that can be used to view output a single screen at a time as well as the ability to search through the output.

There are two commands that I will mention here one called more and the other called pg.

For a simple example just run the more or pg command followed by the filename, or just pipe the output of a command through more or pg.

```
pg filename
```
or
```
more filename
```

```
ls | pg
```
or
```
ls | more
```

Depending upon the command in use either the space or return key will page through a screen of text.

If using the more command it is possible to move up and down the output or to search within the document.
The following commands allow moving up and down the file.

- **j / k** - up one line at a time / down one line at a time
- **Ctrl+F / Ctrl+B** - move forward a screen at a time / move backwards a screen at a time
- **nG** - go to line n
- **1G / G** - go to first line / go to last line

If these sound hard to remember at first these are the same keys as used by the vi text editor which is covered later, the keys therefore only need to be learned once for both more and vi.

To search through the file for a certain string you can type a forward slash '/' followed by the text to search for. The search can be repeated by a forward slash followed by the enter key.

Extracting relevant lines (grep)

The grep command will take an input (or a file) and only show the lines that contain a certain pattern. To output contents of a file listing only lines containing the word computer enter the following command

```
cat filename | grep computer
```

There are a number of options that can be used such as -i to ignore the case of the search argument. These can be found by using the man command.

You can also search for a string of words containing a space by enclosing the search pattern quotes.

```
cat filename | grep “computer program”
```

Command Substitution (`backquote`)

Sometimes you might like to take the output of one command and use it as a list of files for another.

An example is where you have a list of files that should be deleted. The backquote ` (called the grave accent) can be used to perform this in a single step.

```
rm `cat filelist`
```

Using this any filenames in filelist will be deleted. The command inside the backquotes is run and it’s output given one at a time to the rm command.

Processing a list of files (xargs)

An alternative to using backquotes (see earlier) is to user the xargs command. The xargs command has no limit to the number of files that can be processed. When run the xargs command will run once for each of the filenames given to it, which is different to the backquotes which will try and process all the files in one go.

The above command would be run by using

```
cat filelist | xargs rm
```

Running Commands one after another (;)

Often it is required that a single command is issued that will trigger a series of events. This is normally handled using a script file. Putting all the commands into a script file will cause them all to be executed in order (see separate section on script files).

However sometimes it is not practical to write a script file to perform a number of instructions. Another way is to use a semicolon to separate a single string into multiple commands to be run separately.

For example:

```
cd testdir; ls -l
```

the above command will change to the directory testdir before executing the ls command.

Character Substitutions

Character substitutions will replace certain things typed into a shell and manipulate them before running the command. Certain character substitutions allow a single command to be entered that can process a number of files, without having to list all the files individually. Another way of using character substitutions is to run a command against a filename where you cannot remember the full filename.
A few popular substitutions are listed below.

Wildcards

These are useful if you want to find a file when you can’t remember it’s full name, or if you have some files that all begin with the same letters which you want to process all at once.

The question mark ‘?’ when put in place of a letter will match on a single occurrence of any letter.
e.g.

```
ls file?
```

will match against file1 and file2 but not file10 or filetwo and not on file

The asterix ‘*’ will match againstmultiple characters (or no character in that space).

```
ls file*
```

will match against file1, file2, file10, filetwo, file but not filler.

These wildcards can be used in lots of commands including rm, cp, qprt etc.

Inclusion Lists

Inclusion lists are similar to wildcards however instead of specifying any character allow you to be more specific.

```
ls file[xyz]
```

would match on filex, filey and filez but nothing else.

The ! will ensure that only those not matching the enclosed letters would be processed.

```
ls [!frt]*
```
would list all files except those starting with f r and t.

```
ls file[1-5]
```
would list all files from file1 to file5

Escaping Special Characters (Metacharacters)

Sometimes it may be necessary to use either special characters or spaces.

One example is where a file has been created with a space in it's filename.

Using either single quotes ' or double quotes " you can refer to a filename with spaces.

e.g.
```
touch "temp file"
```

Quotes can also be used to override metacharacters.

```
$ echo '$HOME is your home directory'
$HOME is your home directory
```

Double quotes will interpret $ ` and \ but ignore all other metacharacters

```
$echo "$HOME is your home directory"
/home/stewart is your home directory
```

The backslash will escape the following character (remove it's special meaning)

```
$echo This is an asterisk \*
This is an asterisk *
```

To extend a command on a separate line put a '\' on the end of the first line an '>' prompt will be issued to indicate that you are still typing on the same line.

A few more UNIX commands

I have already covered a lot of UNIX commands however there are a few more that are worth mentioning. These are all command line shells that add to some of the power of UNIX being able to manipulate files or command output to create useful scripts.

Cutting columns from tables or field (cut)

The cut command allows you to specify certain columns to keep and remove the rest. This is useful where a certain command might give a number of columns of information however only a few are needed to be actioned on.

There are two ways of using the cut command one on column number (character position) or on a delimited file.

To use the column position you use the -c option followed by the columns to keep.

e.g. Using the ls command you may just need to know the filenames and sizes. The file size is stored from character position 32 to 41 and the filename from 55 to the end of the line.

```
$ ls -l
total 30
drwxr-xr-x   2 stewart  staff        512 10 Jan 08:09 dir1
drwxr-xr-x   2 stewart  staff        512 10 Jan 08:09 dir2
-rw-r--r--   1 stewart  staff          0 10 Jan 08:08 file1
-rw-r--r--   1 stewart  staff        103 10 Jan 08:08 file2
-r-xr-xr-x   1 stewart  staff      13786 10 Jan 08:09 file3
$ ls -l | cut -c 32-41,55-

      512 dir1
      512 dir2
        0 file1
      103 file2
    13786 file3
$
```

Note: the 55- at the end will provide all input from column 55 onwards.

To show how a delimited file can be used I'll use an example of how to find the username and userid number from the file /etc/passwd.
The default delimiter used by cut is the tab character however in the passwd file it's a colon ':' so we need to change the delimiter using the -d option.

```
$ cut -f1,3 -d: /etc/passwd
root:0
daemon:1
bin:2
sys:3
adm:4
uucp:5
guest:100
nobody:4294967294
lpd:9
imnadm:200
stewart:201
$
```

Sorting files (sort)

The sort command can be used to sort the lines of a file.

Some of the options available are:

-d	Sort in dictionary order (only letters, digits and spaces are used in comparison).
-r	Reverse the order of the sort
-n	Sorts numeric fields in arithmetic value

It’s important to note that unless you specify otherwise the sort will work on your locale cultural convention (this is setup as part of the installation). This defers between the UK and US where UK cultural convention will sort letters A-Z before a-z whereas the US convention sorts Aa Bb Cc etc.

Combining this with the cut command performed earlier we can sort files by filesize.

```
$ ls -l | cut -c 32-41,55- | sort -n

        0 file1
      103 file2
      512 dir1
      512 dir2
    13786 file3
$
```

Handling duplicate lines (uniq)

If you have a text input with duplicate lines you can use the uniq command to remove any duplicate lines. The file takes input from a file, removes any duplicate lines and outputs to another file.

There are also some further options that can be used to change the way uniq deals with these duplicate files.

-c	will precede each output line with a count of the number of times it appears.
-d	displays only the repeated lines
-u	displays only the unique (none repealed lines)

Equating expressions (expr)

The expr command allows you to evaluate arithmetic expressions. This is useful for use in a script where you may want to perform calculations on one or more outputs before deciding what actions to take.

```
expr (expression 1) operator (expression 2)
```

example operators include

\\(\\)	used to fix the evaluation order.
+	addition
-	subtraction
*	multiplication
/	integer division
%	remainder

spaces are required before and after the operator.

These are explained further in the scripting section.

Your Working Environment

Shell Variables

There are a number of shell variables that allow your environment to be adjusted. Probably the best known variables are:

HOME - your home directory path e.g. /home/watkiss
TERM - your terminal type e.g. vt100
PATH - your search path which will be searched whenever a program is asked to be run (except where a full / relative path is provided).

We have already come across one of these which is the TERM variable. When telnetting in from a Windows 9x machine the standard telnet uses a terminal type of ANSI. ANSI is not a type supported by smit along with other applications and therefore it is a good idea to change the TERM variable by entering the following commands:

```
TERM=vt100
export TERM
```

The first line changes the shell variable, the second exports it for use by the shell. The second line is normally only needed when run from inside a script, however it does no harm to run it anyway.

Normally all system defined variables are in uppercase whereas user variables are in lowercase.

You can list all variables by using the set command.

e.g.

```
$ set
_=set
AUTHSTATE=compat
CGI_DIRECTORY=/usr/lpp/internet/server_root/cgi-bin
DEFAULT_BROWSER=netscape
DISPLAY=myrs6k.mynet.com:0.0
DOCUMENT_DIRECTORY=/usr/lpp/internet/server_root/pub
DOCUMENT_SERVER_MACHINE_NAME=myrs6k
DOCUMENT_SERVER_PORT=80
.....
```

The above display shows the first 8 lines of my shell variables.

To assign a new value to a variable enter:

```
myvariable=value
```

to reference a variable put a dollar sign in front of the variable name.

e.g.

```
$ echo $TERM
vt100
```

to delete a variable use the unset command

```
unset myvariable
```

If the variable is being used within a string of text it is necessary to surround the variable name with braces "{ }"

e.g.
```
echo Answer is ${myvariable}metres.
```

The output from a command can also be used to set a variable

e.g.
```
now=`date`
```

Shell Defaults

There are a number of things that are already set when a user first logs on. These are kept in a number of files which can be edited to customise the environment whenever the user logs on.

The first of these applies to all users and is called.

/etc/profile

This specifies variables to be set and commands to be run when a user logs in. Normally only root can update this file.

To environment for an individual user in the file called

$HOME/.profile

from the earlier explanation you can see that this is a file held in the users own directory.
For example on my system with user stewart the file is in

/home/stewart/.profile

You can also see that the file begins with a dot '.' which makes the file hidden. The file will not normally be seen by the user and so prevents a user that does not understand what it does from accidentally deleting it. Hidden files can be viewed with

```
ls -a
```

Normally this can be updated by the user allowing them to customise their environment so that it works better for them. This will override any changes made in /etc/profile.

Most of the commands run in the profiles are obvious to there purpose, have descriptive comments or can be found with the man command. Some settings, like the TERM setting, may have a short script to determine what value is set.
I have however picked out a few of the commands / variables and given a brief explanation.

- LOGNAME - This holds your login name for use by certain commands. This is one variable that cannot be changed.
- MAIL - This holds the name of the file where your mail is sent
- MAILCHECK - How often the shell will check to see if you have new mail
- TERM - The terminal type you are using / emulating
- umask - This determines what permission bits will be set when a new file is created. This is a mask so an inversion will give you the default file settings. Eg. the default of 022 will create a new directory with 755 (user can read, write, execute everyone else can read, execute). When a regular file is created the execute bit is not set so you would get 644. You may want to make this a little more secure by setting to 027 (don't give any permissions to other users) or even 077 (only give owner permissions). See the security section for more details.
- PATH - This is a colon separated list of all the directories that will be searched to find a command typed in by the user. Some people add a :. to this string to say if the command is not found in the rest of the path then try the current directory, however there are security issues associated with this (see security section) and certainly this should not be set for the root user.
- PS1 - This is the system prompt and is set to a $ by default. You may want to change this to include your current directory name using the command:
  ```
  PS1='$PWD $'
  ```
- ENV - Not included by default it can be useful to add this to your .profile. To set the shell environment file to .kshrc (this is the normal name for the Korn Shell) you would include the following line:
  ```
  export ENV=$HOME/.kshrc
  ```

The last file that I have included is one to customise the korn shell. This is not available by default and requires the following line to be included in your .profile

```
export ENV=$HOME/.kshrc
```

$HOME/.kshrc

This file will need to be created and the execute bit set.
Whenever a new Korn shell is created this will be executed. This is different from the .profile which is only run when you login. It is important to understand this distinction particularly when using X.

When using X the .profile is loaded when X-Windows is started. This sets up the system wide variables, however there are commands that can be set that are specific to the shell and not to

the overall session. Normally these will not be passed onto the virtual terminal unless they are put in the .kshrc file.
The name of the file is not important more that it is the same as the ENV variable set in the .profile file.

An example of a command that would work differently if it is in .profile and .kshrc is

```
set -o vi
```

including the above command is the same as starting the shell with

```
ksh -o vi
```

which sets the shell environment into the vi style input.

If this was in the .profile then whenever a login was made into a terminal it would take effect, however when starting a terminal from X it is not a new login and therefore would not be invoked. If however this is included in the .kshrc then whenever the Korn Shell was started it would be invoked and as the Korn Shell is invoked by both a login and a new terminal started in X this would happen all the time.

Some of the commands that could be put into the .kshrc file.

- set -o vi - This command sets the environment into the vi style input. What this does is to provide a level of command recall for the shell. To really understand this you need to understand how to use the vi editor (see the next section). To use the recall keys you first need to press the escape key (this may be mapped differently on your system however is normally the key marked Esc). Then you are in command mode and can recall commands and move around using the keys “hjkl” (‘h’=left, ‘j’=down, ‘k’=up, ‘l’=right). You can then delete a character by using the ‘x’ key. Pressing the ‘i’ key will insert before the cursor and pressing the ‘a’ key will insert after the cursor. Pressing either ‘i’ or ‘a’ will put you into insert mode and to return to command mode you press the escape key again. This may all sound very complex however if you are able to use the vi editor (a useful skill to acquire) then you will find this very familiar.
- alias - The alias command is used to set up alternative names for commands. The format of the command is:
  ```
  alias newcmdname='normalcmdname -args'
  ```
  What will happen then when you enter newcmdname is that it will be replaced with normalcmdname -args. Note: this will happen only on commands entered at the command line and will have no effect on script files. A few examples of where this would be useful is shown below:

```
alias rm='rm -i'
```

This will include the -i option on any delete command. What this does is provide interactive prompting so that you will be asked if you want to delete each file.

```
alias up='cd ..'
```

Whenever you enter the up command it will take you one level up the directory tree. This is useful if you, or a user have trouble remembering a command as you can make easy to remember alias names.

```
alias computer2='telnet computer2.mynet.com'
```

If you often telnet to another computer (computer2) this will save you having to type in the whole command instead of entering 26 characters you just need to enter 9 characters.

History

Another way of recalling commands is to use the history command.

Issuing the history command will show the last few commands entered.

```
$ history
52     clear
53     ls
54     cd test
55     ls
56     pwd
57     cd ..
58     vi temp.txt
59     history
$
```

to recall one of the commands listed enter an r followed by the number

```
r 55
```

by default the history command will give you the last 16 commands entered.

The history is held in a file called .sh_history and can be edited using the fc command (see the man pages for more details).

Using the VI Text Editor

VI is a very powerful text editor. It runs in a standard terminal and uses the standard keyboard (it is not reliant on the arrow keys, or the insert, home, pgup keys). This makes it equally as useful on a graphics workstation as on a minimal text based terminal or indeed via a telnet terminal.

To someone new to UNIX the vi text editor can be very daunting. However learning to use vi can be one of the most useful tools that a UNIX user can know. Whilst anyone with a X-Windows system may prefer the ease of use provided by the X based editors such as dtpad or xedit you cannot always guarantee to have them available. Whereas regardless if you’re working on a graphics or text only terminal almost all UNIX systems will have the vi editor installed. Also once you know the basics for vi you can use these in other circumstances, for example the korn shell can be setup to have the vi keys for command recall and edit (see details on the .kshrc file) or the same keys can be used to scroll through a file using the more command.

Rather than just describe what all the different keys do I’ve worked through a little example of creating and editing a text file, however it is first important to understand about the different modes of the vi editor.

Editor Modes

If you have not used vi before, then having to switch between the different modes can seam a bit alien at first, however is actually quite simple in operation. The most common two are “Insert mode”, and the “Command mode”.

Whenever you start vi it will be in command mode. In this state whenever you type something, instead of being included in the file it will be used as a command for the editor. Example commands include moving around a document, deleting parts of the document, and file operations such as saving the document and exiting.

To add the text that you type in at the keyboard requires you to be in insert mode When in insert mode any text entered at the keyboard is added to the document at the current cursor position.

To switch between the different modes involves pressing the appropriate keys. To move to command mode you would press the ESCAPE key. A useful feature of the ESCAPE key is that it still works when in command mode. Indeed if you forget which mode you are in just press the ESCAPE key and regardless it will put you in the command mode.

There are several different keys that allow you to go from the command mode to the insert mode. By using the appropriate key, the number of keystrokes required can be reduced. Some of the keys are listed below:
i Insert - any text entered will be put immediately before the current letter.
a After - any text entered will be put immediately after the current letter.
A After end of line - any text entered will be put after the end of the current line.

It is also possible to run ed commands by pressing colon ‘:’ when in command mode. The ed commands are based on command line editing used by the “ed” command line editor. The vi editor is based on the ed program that is a basic command line editor.

Worked Example with VI

Following this example will introduce a lot of the functionality of the vi editor.

Starting VI

To start the editor enter vi followed by the filename.

```
vi sjayouth.txt
```

If the file already exists then it will be opened and displayed on the screen. If it is a new file then the initial screen will be blank. The editor is started in command mode. The file opened is shown below. It is an article from a St. John Ambulance Newsletter that I wrote in September 2000:

```
Mention St. John Ambulance and most think of adults
trained in  first aid helping those in need. However
St. John actually has one of the biggest youth
organisations in the country, catering for all
children from the age of 6 upwards.
The youth area is split into two age groups. Badgers
are for children from 6 to 10 and Cadets are for
members aged from 10 to 18. Both groups are open to
boys and girls.

Badgers make friends and have fun. The badgers work
towards 9 different badges in areas from first aid
to hobbies. After gaining all 9 badges he / she can
become a Super Badger.

Cadets play games and learn first aid, like the
badgers. Cadets then get to practice the first aid
on real casualties. There are adult members ready to
give an helping hand whenever it's needed. The
cadets work on an award scheme leading up to the
Grand Prior Award.
```

Inserting a New Line

We will now add a line before the text with the title “Young Members”. The cursor is located in the top left hand corner which is exactly where we want to insert the text. When we started the editor we were in command mode so we now need to change to insert mode by pressing:

```
i
```

We can then type in the title

```
Young Members <Enter> <Enter>
```

Pressing the enter key moved the following text to the next line and then inserted a blank line.

Moving Around the File

Now imagine that I wanted to replace the word “practice” with “use”. To move around the file it may be possible to use the cursor keys. Some terminals do not support the cursor keys and the alternative is to use four of the standard keys. These are
h - left
j - down
k - up
l - right
To use the standard keys as cursors requires that you are in command mode.

To move to the word “practice” first press
ESCAPE
to change to command mode.

Once in command mode then press the ‘j’ key until the cursor is next to the line containing the word. Then subsequent presses of the ‘l’ key will move until the cursor is on they letter ‘p’ of practice.

Deleting Letters and Inserting a New Word

Now press the ‘x’ key, which will delete the letter under the cursor. Press they ‘x’ key 7 more times until the word has been deleted. Then press the ‘i’ key to return to insert mode and enter the word “use”.

Save the Document

We will now save the file in case we accidentally make a mistake in future. To save the current document you need to change to command mode. Then press the colon key (:) to create a command line and enter ‘w’ followed by the enter key.

The file will then be saved.

Replacing a Letter

Prior to the introduction of the children's act the age for cadets was up to 16. Now we shall change the age from 18 to 16. We shall move the cursor and replace the figure 8 with the figure 6.

First press ESCAPE and use the ‘k’ key to move up the document up to the line containing the figure 8. Then move across the line using the ‘h’ and ‘l’ keys until the cursor is over the figure 8.

To replace the character press the ‘r’ key. Then pressing the ‘6’ key will cause the number to be replaced with a 6. You will still be in command mode.

Search and Replace

For this exercise we will now replace the occurrence of 18 back to 16, however this time we will do it using a ed command. We can find the occurrences of a word by using a regular expression. So to search for 18 we would enter /18 followed by enter. This would search from the start of the file until it reached the first instance of 18.

If instead we wanted to replace 18 with 16 directly then the command that would be entered is
:%s/18/16/g
The colon tells the editor that what follows is an ed style command. The %s is to initiate a search and replace. The first string between the ‘/’ characters is the item to search for, the next is the string to replace with. This would work on every line of the file, but only of the first occurrence on each line. The g option makes the command work globally for every possible occurrence.

This may also replace other parts of strings such as 180 would be replaced with 160. So it is important to ensure that the command that is issued is what is you really want it to do.

Save and Exit

Now we will save and exit the document. Press the ‘:’ key followed by ‘w’ (to write - save) followed by ‘q’ to quit the program.

This is then end of this example.

Preferences

Within vi it is possible to change some of the settings. These are done by issuing a ‘:’ followed by the set command with the option as a parameter. For example to display line numbers in the editor you would issue the command:

```
:set number
```

This could then be turned off again by using the word no in front of the options. So to turn the number option back off you would issue:
```
:set nonumber
```

Other options that can be set include:

all	Display all options & current settings
errorbells	Sounds a bell sound whenever an error occurs
ignorecase	Makes search commands case insensitive
showmode	Shows the current mode in the bottom of the screen

These settings only last as long as the current session. To have these commands run whenever vi is run then the required commands should be put in a file called .exrc in your home directory. These should be entered without the colon e.g.

```
set number
set showmode
```

Processes

A process is a task that is running on the computer. Every time you run a program you start one or more processes running.
There are quite a lot of processes running on the computer before you are even able to login to AIX. For example there are processes that control their kernel (the core of the operating system), processes to handle error messages, logging, cron, networking etc.
Also to be able to login the getty process needs to be running (this provides the terminals, either physical or virtual) and if you are logging on using the CDE this also needs to be running.

In UNIX processes are referred to by a unique number known as a Process ID (PID). When one process spawns another process there is a hierarchical link between the two processes. The process that starts the new process is known as the parent and the new process is known as the child. The parent process is referred to by the Parent Process ID (PPID). A parent process can have multiple children however a child can only have one parent.

For example if a user is running under the kshell on PID 18758 and then runs the cat program from within the shell the cat process may have the PID of 14230 (note the PID numbers are arbitrarily set can go up or down depending upon the PID's in use by other processes). The cat process will have a PID of 14230 and a PPID of 18758.

There is one PID that is always the same which is the init process (the first process to start on the system). This is given the value of 1.

You can access the PID of your current shell using the $ variable.

To show the value of the PID of your current shell type:

```
echo $$
```

You can view other processes and their process ID’s by using the ps command. To get a full display of the PID, PPID, etc then use the -f switch.

```
$ ps -f
     UID   PID  PPID   C    STIME    TTY  TIME CMD
 stewart 11552 12110   1 03:54:08  pts/3  0:00 /usr/bin/ksh
 stewart 18664 11552   9 03:57:22  pts/3  0:00 ps -f
 stewart 18772 11552   0 03:56:23  pts/3  0:00 dtterm
 stewart 20182 11552   2 03:57:20  pts/3  0:00 dtpad filename
$
```

From the example above you can see that the user has a session in the korn shell. From that shell a dtterm session was started (CDE terminal) and dtpad (CDE text editor) editing the file called filename. The ps command actually shows itself running.

Using the ps command in this way only shows processes under the current session. To display all running processes including those by other users and the system ones use the -e option. The output will be long (compared tot h3 4 lines issued in the earlier example).

```
ps -ef
```

Daemons

There are processes that are started that are not intended to ever end. These processes work in the background normally to process certain events or to start other processes when certain events occur. These are known as daemons and will normally start when the computer is started and stop when the computer is shut down.
Some common daemons are cron (see earlier), qdaemon (printing), system and error logging daemons, and lots of networking related daemons.
As far as process control is concerned these can be treated the same as processes running in the background.

Return Codes

When commands execute they will send back a return code. This is to indicate whether the program worked successfully or not. The convention is for all programs completing successfully to return 0 and return a none-zero value when an error occurred.

The variable $? contains the return code of the last command.

The return code used can be used to determine whether it is necessary to run a certain command or not. For example you may want to check on the presence or absence of a file. This can be done in the following two ways.

command1 && command2
If command1 completes successfully (returns 0) then run command2

e.g.
```
ls filename && echo filename does exist
```
If the ls command is successful (a file exists called filename) then the echo program will say filename does exist, otherwise it will not echo anything to the screen.

command1 || command2
If command1 does not return 0 then run command2

e.g.
```
ls filename || echo filename does NOT exist
```
If the ls command fails (filename does not exist) then the echo program will report it's none existence, otherwise it will not echo anything to the screen.

Whilst the examples above illustrate the point they are not very useful. They could however be used to run other programs that may create the file that was missing if it was needed.

Controlling Processes

Normally when a command is run from within a shell it will be run in the foreground. What this means is that it will take control of that session whilst it is running only allowing another command to be executed when it is finished. Often it is desirable to set a program running, but instead of having to wait for it to finish then start another program whilst the first is running in the background. This is achieved by putting and & at the end of the command.
Obviously certain programs that require interaction are not suitable for running in the background.

It should also be kept in mind what will happen to any output from the command. You may want the output sent to a file so that you can view / action it later, however you may need to know if there are any errors. Selective use of the re-route commands can help here.

For example you may want the output to be sent to a text file however in the event of any errors you may want to see them on the screen immediately. This is achieved by issuing the following:

command1 >textfile &

Prioritisation

Whilst it may appear that all programs running on a UNIX computer are running at once they are not. Processes can only be running simultaneously if there is more than one CPU within the computer. However there are more processes running on a typical computer than the number of processes. The way the computer makes it appear that all the programs are running at the same time is using a scheduler. The scheduler tells the processes when they can have use of a processor to execute a small amount of their processing. After a certain amount of time the scheduler will tell the process to wait and let another process run. This happens so fast that to the user it appears that the processes are all running simultaneously.

Some programs need to run more frequently or longer, or finish earlier than others and for that reason it is possible to set and change the prioritisation of jobs. The priority of a job is given a value, the higher the value the lower the priority. The priority of a job is related to the nice value which can be changed by the user. The nice value allows a lower number to be given to a process to give a higher priority and vice versa. By default the default nice value is 20 unless the process is running in the background when it is 24. If you want the program to run quicker then give it a lower nice value and if you want it to run slower give it a higher nice value the range for the nice value is from 0 to 39. Only root can make a process run at a higher priority.

You can see the priority and nice values by using the ps command with the -l option.

```
$ find / -name rm* 2>/dev/null >outfile &
[1]     18438
$ ps -l
       F S      UID   PID  PPID   C PRI NI ADDR    SZ    WCHAN    TTY  TIME CMD
  240001 A      201  3424 12634   0  60 20  880   496           pts/2  0:00 ksh
  200001 A      201 18438  3424   3  69 24  4ad   184           pts/2  0:00 find
  200001 A      201 19970  3424   4  62 20  26b   288           pts/2  0:00 ps
$
```

As you can see the find command was run in the background and as a result was given a lower priority than the other processes.

The priority can be reduced further by using the nice command.

```
$ nice -15 find / -name rm* 2>/dev/null >outfile &
[1]     16772
$ ps -l
       F S      UID   PID  PPID   C PRI NI ADDR    SZ    WCHAN    TTY  TIME CMD
  240001 A      201  3424 12634   0  60 20  880   496           pts/2  0:00 ksh
  200001 A      201 16772  3424  20 113 39  a23   184           pts/2  0:00 find
  200001 A      201 19982  3424  10  65 20  34a   284           pts/2  0:00 ps
$
```

As you can see the nice value was increased on the find command by 15.

To change the nice value of a process, that was started using the nice command, the renice command can be used.

Further Job and Process Control

You may need to stop processes or jobs whilst they are running. When a command is running in the foreground it is possible to halt the process using the Interrupt key (CTRL-c). Pressing CTRL-C will cancel the foreground process and return to the shell that started the process.

It is not possible to use CTRL-C on a background process for obvious reasons. Also sometimes the CTRL-C won't work on the process your trying to cancel. Another way therefore of stopping a process is using the kill command. The standard way of using this is to issue:

```
kill PID
```

where the PID can be obtained from a ps command.

To use the kill command you must either have started the command or be the root user.
If the kill command doesn't work, there is an option that will force the process to cancel and that is a -9 option to the kill command.

To force a command to stop:

```
kill -9 PID
```

The kill command works by sending a system signal to the process. There are a number of different system signal that can be sent, either automatically by the operating system or manually using the kill command. The signals available to the kill command are:

```
$ kill -l
 1) HUP        14) ALRM       27) MSG        40) bad trap   53) bad trap
 2) INT        15) TERM       28) WINCH      41) bad trap   54) bad trap
 3) QUIT       16) URG        29) PWR        42) bad trap   55) bad trap
 4) ILL        17) STOP       30) USR1       43) bad trap   56) bad trap
 5) TRAP       18) TSTP       31) USR2       44) bad trap   57) bad trap
 6) ABRT       19) CONT       32) PROF       45) bad trap   58) bad trap
 7) EMT        20) CHLD       33) DANGER     46) bad trap   59) bad trap
 8) FPE        21) TTIN       34) VTALRM     47) bad trap   60) GRANT
 9) KILL       22) TTOU       35) MIGRATE    48) bad trap   61) RETRACT
10) BUS        23) IO         36) PRE        49) bad trap   62) SOUND
11) SEGV       24) XCPU       37) bad trap   50) bad trap   63) SAK
12) SYS        25) XFSZ       38) bad trap   51) bad trap
13) PIPE       26) bad trap   39) bad trap   52) bad trap
$
```

These signals are also sent by the operating system in response to other events.

E.g.
- Hangup (01) - sent to a process if it's parent dies e.g. logging off when a background process is running.
- Interrupt (02) - generated when the Interrupt key is pressed (often CTRL-c)
- Quit (03) - sent when quit key is pressed (often CTRL-\)
- Terminate (15) - issued when the kill command is run with no other options
- Kill (09) - this is a software terminate signal that cannot be ignored by the software.

You might have a situation where you don't want a process to end when you logout. For example you may set a job to run overnight. If you just run the command in the background the Hangup signal will normally cause the program to terminate. This can however be overridden by the nohup command.

```
nohup command &
```

If the output is not redirected as part of the command it will be automatically redirected to a file nohup.out. The nohup command will ignore the signals hangup (01) and quit (03)

You may have a requirement to move a job from the foreground and have it run in the background. For example you may have issued the command expecting it to be quite short and later realise it's taking too long. Or indeed you may just want to suspend a job from running and restart it later. There are a number of commands that allow you to do just these kinds of functions.

<CTRL-Z> will stop a foreground task. Rather than terminate the task as would happen if you used the interrupt key the "job" will be in a suspend state ready to be restarted.

fg will resume a suspended job to run in the foreground

bg will resume a suspended job to run in the background

jobs will list background or suspended jobs.

To run the above commands requires the job number prefixed with a percentage sign '%' (the job number can also be used by the kill command in the same way).
The following example shows how a command can be stopped from running in the foreground and resumed in the background.

```
$ find / -name r* >temp.txt 2>/dev/null              <CTRL-Z>
[1] + Stopped (SIGTSTP)        find / -name r* >temp.txt 2>/dev/null
$ jobs
[1] + Stopped (SIGTSTP)        find / -name r* >temp.txt 2>/dev/null
$ bg %1
[1]     find / -name r* >temp.txt 2>/dev/null&
$
[1] +  Done(1)                 find / -name r* >temp.txt 2>/dev/null
$
```

RS/6000 Architecture

RS/6000 computers are computers based around a RISC (Reduced Instruction Set Cycles) based processor. The RISC processor is a simple processor with only a certain number of possible instructions, this is compared to CISC (Complex Instruction Set Cycles) computers that have a lot of instructions. The advantage that RISC based processors have is that the few instructions it does have it can process very quickly making them very powerful computers. Even PC's running Intel CISC processors (such as the Pentium series) acknowledge the performance gains of RISC technology, so they now have a RISC core with pre-processors to convert the CISC instructions into ones that can be handled faster by the RISC core.

Modern RS/6000's contain the PowerPC processor developed between, IBM, Apple and Motorola. Along with this RS/6000's can handle parallel processing (using pipelining) and have a large address space of up to 4 petabytes (2^{52}). Older RS/6000's were powered by the 32 bit POWER processor.

RS/6000's come in two types. Current RS/6000's have a PCI I/O bus with ISA slots however older RS/6000's had a Microchannel (MCA) bus. As there are a number of differences between these they are referred to as PCI or Classic RS/6000's for the PCI or MCA bus respectively.

The 64bit PowerPC processor replaced the older 32 bit POWER processor. The PowerPC processor is binary compatible with the POWER processor. In that any programs compiled for the POWER processor will run on the PowerPC processor. However for optimal performance programs should be recompiled to allow it to make full use of the 64bit processor.

RS/6000's come in all sizes from workstation PC's to full rack mounted SP computers. You should contact your IBM authorised retailer for an explanation of the range available.

There are a number of different technologies that are in use in RS/6000's. I am not going to go into full details here however I have summarised some of the technologies below:

- MicroChannel Architecture (MCA) - This is a architecture used by IBM, which is the type of slot provided in the Classic RS/6000. MCA adapters are all self-configuring when plugged into the computer will be detected.
- Small Computer System Interface (SCSI) - This is an industry-standard way of connecting devices. It is typically used for storage devices, e.g. hard disks, tapes, CD-ROM's etc. There are a number of different standards for SCSI however the basic principle is that a number of devices can all be daisy changed together.
- PCI - The PCI cards are the same as used in many PC’s. They do not auto-configure as easily as MCA cards. The will normally require configuration using the System Management Services (SMS) program, supplied either as a diskette or as a built-in function of certain models. This is particularly important when more than one type of card is used.
- ISA - ISA is an older form bus used on personal computers prior to the use of PCI. It is provided on all the PCI systems for compatibility with cards that were available at the time. Whenever possible PCI cards should be used in preference to the ISA cards.

Symmetrical Multiprocessing (SMP)

It is possible in many RS/6000’s to have more than one processor. This allows true multiprocessing in that different threads can be running simultaneously on the different processors. This is achieved by adding a second PowerPC chip and Level 2 caches. A software component of AIX allows mirror console activities with the third serial port. This allows one console port connected to a modem and monitor the activity using the local mirrored copy.

Each processor has access to the full memory.

Massively-Parallel Processors (MPP)

MPP computers differs from SMP by instead of having a single operating system shared across all processors each processor (known as a node) runs a separate copy of the operating system. They also have separate disks memory etc. As a result of this there is little traffic across the shared interconnect/switch. This allows true scale ability and provides linear growth as new nodes are added.

SP Computing

SP use standard AIX and RS/6000 components however it has it's own hardware components and software to make management easier.

Installation of AIX

AIX can be installed from any of the following:

- CD-ROM
- Tape (4mm or 8mm)*
- Pre-installed (when purchased with a new system)
- Network Install Manager (NIM) over Token Ring, Ethernet or FDDI

The minimum memory requirement for installation is 32MB (AIX V4.3)

* It is not possible to install from tape on some of the PCI based RS/6000's.

Step 1 - Booting into install mode

The first step is different depending upon whether it is a Classical RS/6000 or a PCI RS/6000, the processes are therefore treat separately.

Classical RISC/6000

1. Turn the key to the service position. This will allow booting from the media, without this the RS/6000 will only boot from the preferred drive (hard disk).
2. Insert the disk / tape in the drive (the drive must have been powered on prior to the starting the system otherwise it will not be detected).
3. Power on all peripheral devices. Any devices not powered on will not have the necessary drivers installed for them.
4. Power on the system or push the reset button twice.
5. The system will now attempt to boot from the first entry in the bootlist, by default this is either the tape or the CD-ROM.

The system will now boot. It will go through the Built-In Self Test (BIST) and the Power-On Self Test (POST) before loading the operating system from the media.

PCI RISC/6000

1. Insert CD into drive
2. Power an all peripheral devices
3. Power on system. Depending upon the system it may be necessary to hit F5 when the Power PC logo is being displayed.

Once the system is powered on it will search through it's bootlist for a bootable image. By default this will be the diskette drive, followed by the CD-ROM and then the hard-drive.

Step 2 - Defining the Console and Language

The following message will then be displayed on all native (graphics) displays and the first built in serial port (S1).

```
****** Please define the System Console. *****

Press the F1 key and press<Enter>
to use this display as the System Console.
```

Define System Console screen

The default Terminal Settings will be used for S1. These are:

```
Terminal Type = dumb
Speed = 9600
Parity = none
Bits per character = 8
Stop bits = 1
Line Control = IPRTS
Operating Mode = echo
Turnaournd character = CR
```

Default Terminal Settings

Whichever display is responded to will be used as the system console for the installation process. The console can then be redefined later if required.

Graphics terminals require F1 then enter to be pressed whereas an ASCII terminal requires the number 2 follwed by enter.

You will then be prompted for the language to use during the install. This will be used for the duration of the install and is not neccessarily the primary language for the environment of the system.

Step 3 - Installation and Maintenance

Up until this point the process is the same for entering Maintenance mode as for installing the operating system.

At this point you can choose between:

1. Installing with default settings
2. Installing with customised settings
3. Starting in maintenance mode to perform system recovery

You should normally choose option 2 to alter any settings before running the install.

From the Installation settings screen you will now be asked where to install the software and what type of install is required. The choice is:

1. New and Complete Overwrite: This should be used for a new install and is the only option if you haven't already installed AIX on the computer.
2. Preservation Install: This is for use when another version is installed and you want to preserve user data. The contents of /usr, / (root), /var and /tmp will be destroyed and the operating system will have to be reconfigured once installed.
3. Migration Install: Upgrades from an earlier version of AIX (V3.2 and later) preserving all file systems except /tmp.

You may choose the disk to install the operating system onto. If the disk is not shown on the list of available disks it can be added by entering the code for the device.

The primary language for the operating system should be chosen at this point. It is far easier to set at this point as it will ensure the correct files are installed from the installation media.

There is not English (UK) Language however there is a UK specific keyboard and cultural convention setting. For example the UK keyboard will support the £ sign and the UK cultural convention will sort letters A-Z before a-z whereas the US convention sorts Aa Bb Cc etc.

The language can be changed later with the chlang command.

The final option is whether to Install Trusted Computing Base (TCB).
This is a feature that must be set at install time and cannot be changed afterwards. It is a security option protecting against spoof programs or trojan horses. You should ensure that you are aware of the full details of the TCB before choosing to install this option. See the security section.

Step 4 Installation process

The Operating system will now be installed. This will take a while so is a good point to leave it to install.

If there is a key set in the service position it should now (at any time during the install process) be set to the normal run mode to allow a reboot into the newly installed operating system.

The install will set enough space for the installation, however will not leave much free space. The amount of free space can be increased after the install.

The steps taken by the install are:

1. Builds the AIX directory structure
2. Restores BOS, locale and filesets from installation media
3. Installs software for the connected and powered on devices

Step 5 Configuration Assistant

The base operating system is now installed however there are still a number of settings that need to be changed. After booting the configuration assistant will be started on the console device allowing the most common things to be configured.

Each of the items should be configured on the menu.

1. Set date and time
2. Set root password
3. Set System Storage and Paging Space
4. Configure Network Communications (TCP/IP)
5. Updated installed software after a migration install
6. Configure Web-based System Manager (if required)
7. Exit the configuration menu

If you select Exit the configuration menu, configuration assistant will not be started on subsequent reboots. To run the configuration assistant enter the following command:

X Windows

```
configassist
```

ASCII Terminal

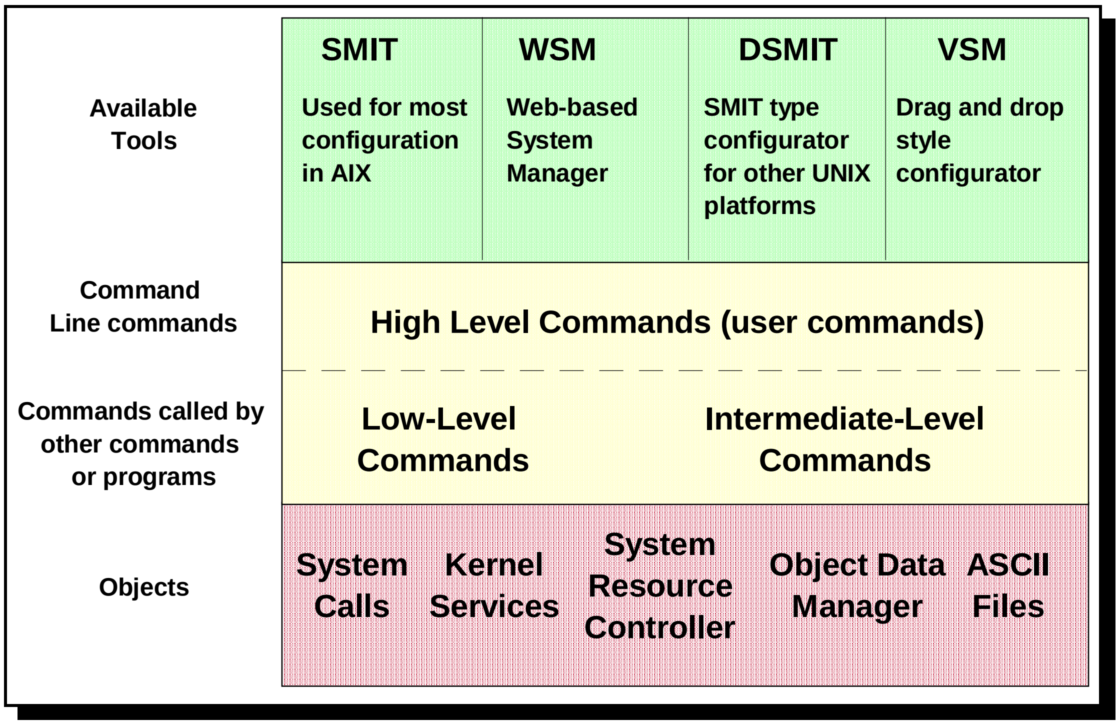
```
install_assist
```

Configuring the AIX System

Traditionally in UNIX almost all the configuration details were kept in flat files. To change any configuration details required either the editing of these files, the entering of commands from the command line or a combination of the two.

IBM has tried to make this into a more consistent method with AIX by firstly moving many of the flat files into a database known as the Object Data Manager (ODM), and by combining the commands needed to change the configuration into a System Management tool called System Management Interface Tool (known as SMIT). The ODM is explained in more detail later.

Through the use of SMIT most of the commands can now be carried out using a single tool.


Available Tools
SMIT
Used for most configuration in AIX
WSM
Web-based System Manager
DSMIT
SMIT type configurator for other UNIX platforms
VSM
Drag and drop style configurator
Command Line commands
High Level Commands (user commands)
Commands called by other commands or programs
Low-Level Commands
Intermediate-Level Commands
Objects
System Calls
Kernel Services
System Resource Controller
Object Data Manager
ASCII Files

Administration Tools Mapping shielding the user from lower level editing

There are for system management tools of which three are designed for AIX. DSMIT is not designed for AIX, it is designed to make the system management of other UNIX operating systems familiar to AIX system administrators. As DSMIT is not used on AIX it will not be considered further.

VSM is a drag and drop style configurator. This is supposed to appeal to users administrators familiar with Windows operating systems. Due to it’s lack of functionality and clumsiness (compared with traditional powerful UNIX constructs) it is better avoided.

WSM is a Web-based System Manager. It runs in a Java window which can be used across other platforms.

The primary System Configuration manager is SMIT. There are two versions of SMIT depending upon whether a text terminal or X Windows is being used.

Using X Windows

Typing

```
smit
```

Will start the X-Windows version of SMIT. This is also known as the Motif smit after the Motif Windows Manager which was a version of X-Windows.

```
smitty
```

Will start the text version of smitty

Using a text terminal

Entering smit or smitty will bring up the text based version.

SMIT does not actually carry out any system management functions directly. Instead SMIT calls other commands to carry out the functions. Sometimes this involves the building of scripts which will carry out a whole sequence of actions. The commands created by SMIT could be run by a user from the command line or by entering them into a script.

The sequence of events to actually carrying out the changes is as follows:

- **High-Level Commands.** These are standard AIX commands that can be entered by a user. They execute multiple low level or intermediate level commands to perform the system management functions.
- **Low-Level Commands**. These are AIX commands that correspond with AIX system calls or kernel services. They will not normally be entered by a user.
- **Intermediate-Level Commands.** These interface with special AIX components such as the System Resource Controller and the Object Data Manager. These commands would not often be entered by a user.

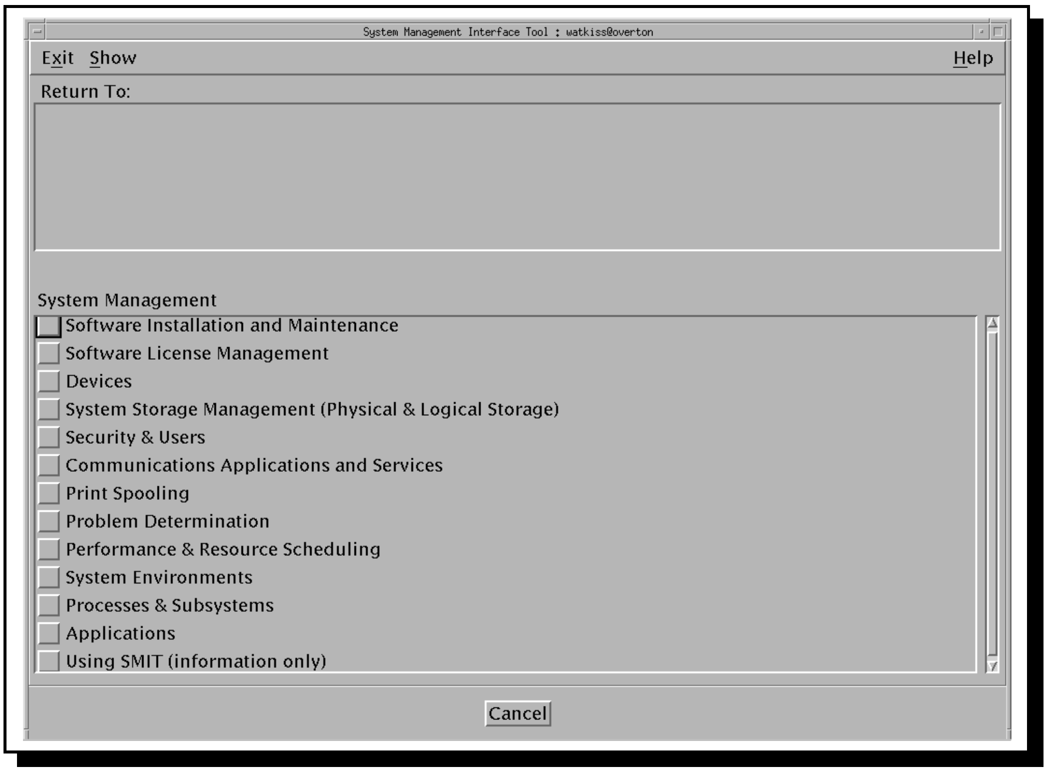
Whilst most functions can be carried out from within SMIT there are inevitably some functions that cannot. It is sometimes necessary to run commands from the command line, or edit ASCII files.

Using SMIT

The following screens show the X-Windows and terminal versions of SMIT on the initial menu.

```
                              System Management

Move cursor to desired item and press Enter.

  Software Installation and Maintenance
  Software License Management
  Devices
  System Storage Management (Physical & Logical Storage)
  Security & Users
  Communications Applications and Services
  Print Spooling
  Problem Determination
  Performance & Resource Scheduling
  System Environments
  Processes & Subsystems
  Applications
  Using SMIT (information only)




F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Text based SMIT


System Management Interface Tool : watkiss@overton
Exit Show
Help
Return To:
System Management
Software Installation and Maintenance
Software License Management
Devices
System Storage Management (Physical & Logical Storage)
Security & Users
Communications Applications and Services
Print Spooling
Problem Determination
Performance & Resource Scheduling
System Environments
Processes & Subsystems
Applications
Using SMIT (information only)
Cancel

Screen Paste of initial screen in X-Windows SMIT

This is the default SMIT menu which can be customised. The menus and functions are the same for both versions however the way that things are displayed differ between the two.

For example in the graphical version when navigating the menus, the previous steps are shown at the top of the screen, when executing a command the command is shown at the top and the output is shown at the bottom. In both these cases the text version doesn't have the screen split up and so the information in the top pane of the graphical version is not displayed.

From now on I will use the text version for any explanations however the graphical version operates in the same way, using the mouse to select the appropriate menu option.

To navigate the menus use the cursor keys. The cursor position is shown by inverted text and pressing the enter key will take you up to the next menu or run the specified command.

A worked Example

The next few steps show how I could change the settings of a userid on the system. I am running as root in this example.

Main Menu

```
                                        System Management

Move cursor to desired item and press Enter.

  Software Installation and Maintenance
  Software License Management
  Devices
  System Storage Management (Physical & Logical Storage)
  Security & Users
  Communications Applications and Services
  Print Spooling
  Problem Determination
  Performance & Resource Scheduling
  System Environments
  Processes & Subsystems
  Applications
  Using SMIT (information only)






F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Security & Users Menu

```
                                         Security & Users

Move cursor to desired item and press Enter.

  Users
  Groups
  Passwords
  Login Controls
  Roles














F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Users Menu

```
                                         Users

Move cursor to desired item and press Enter.

  Add a User
  Change a User's Password
  Change / Show Characteristics of a User
  Lock / Unlock a User's Account
  Reset User's Failed Login Count
  Remove a User
  List All Users










F1=Help              F2=Refresh           F3=Cancel            F8=Image
F9=Shell             F10=Exit             Enter=Do
```

Chuser Menu

```
                    Change / Show Characteristics of a User

Type or select a value for the entry field.
Press Enter AFTER making all desired changes.

                                                        [Entry Fields]
* User NAME                                          []                      +









F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

At this point you are prompted for the username. If you know it then you can just type it in and press enter.
However you can see there is a '+' character at the right hand side of the screen. This indicates that you can get a "List" from the system of possible entries. You can get this by pressing the "List" key, in this case it is F4 (see bottom of screen). This runs a command that lists the users on the system and gives the following screen.

Select User

```
               Change / Show Characteristics of a User

Tylqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqk
Prx                                User NAME                                 x
  x                                                                          x
  x Move cursor to desired item and press Enter.                             x
* x                                                                          x+
  x [TOP]                                                                    x
  x   root                                                                   x
  x   daemon                                                                 x
  x   bin                                                                    x
  x   sys                                                                    x
  x   adm                                                                    x
  x   uucp                                                                   x
  x   guest                                                                  x
  x   nobody                                                                 x
  x   lpd                                                                    x
  x   imnadm                                                                 x
  x [MORE...3]                                                               x
  x                                                                          x
  x F1=Help                 F2=Refresh              F3=Cancel                x
F1x F8=Image                F10=Exit                Enter=Do                 x
F5x /=Find                  n=Find Next                                      x
F9mqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqj
```

From here you can now select the user that you want to change. The cursor keys allow you to scroll down and find the relevant username.

Change Details Panel

```
                    Change / Show Characteristics of a User

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

[TOP]                                                   [Entry Fields]
* User NAME                                           stewart
  User ID                                            [202]                #
  ADMINISTRATIVE USER?                                false              +
  Primary GROUP                                      [staff]             +
  Group SET                                          [staff]             +
  ADMINISTRATIVE GROUPS                              []                  +
  ROLES                                              []                  +
  Another user can SU TO USER?                        true               +
  SU GROUPS                                          [system]            +
  HOME directory                                     [/home/stewart]
  Initial PROGRAM                                    [/usr/bin/ksh]
  User INFORMATION                                   []
  EXPIRATION date (MMDDhhmmyy)                       [0]
[MORE...37]

F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

This is an input screen requiring a number of fields to be filled in. All the above information had already been obtained by smit from querying the system and were filled in automatically. All that is needed is to change any that have the wrong details.
The symbols on the right have certain meanings. We've already mentioned the '+' sign as supporting the list function. Each entry can have none, one or more symbols at the end. Here's a list of some of the uses:

'+' - A list of possible answers can be gained by pressing the "List" key.
'*' - A mandatory field that must be filled in before proceeding.
'#' - A numeric value is needed for this field.
'/' - A pathname is required
'X' - A hexadecimal value is needed
'?' - The value entered will not be displayed

The fields that require a text or number entry (rather than simple choices e.g. true / false) have square brackets "[]", if the value is large and spans more than the available space these change to angled brackets "<>" to indicate that the field can be scrolled.

Once any fields have been changed the enter key will cause the command to be run.

In this case I have filled my full name into the Description Field and pressed enter.

Command Output

```
                                  COMMAND STATUS

Command: OK             stdout: no              stderr: no

Before command completion, additional instructions may appear below.










F1=Help             F2=Refresh          F3=Cancel           F6=Command
F8=Image            F9=Shell            F10=Exit            /=Find
n=Find Next
```

SMIT creates any appropriate scripts and runs them. The output is displayed on this screen. The screen will show status's of OK, RUNNING and FAILED, or will have a icon of a man to indicate the status using the graphics version.

In this example there is no output from the command so the screen is blank. If a command produces more than a single screen of output this will be in a scrollable box within the output screen.

There are also a number of screens that can help understand how SMIT works. Going back to the previous screen I have then displayed the following screens.

Command Screen

```
                    Change / Show Characteristics of a User

Type or select values in entry fields.
Prlqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqk
  x                           SHOW COMMAND STRING                            x
[Tx                                                                          x
  x Press Enter or Cancel to return to the                                   x
  x application.                                                             x #
  x                                                                          x+
  x   x() {                                                                  x+
  x   if [ $# -ge 2 ]                                                        x+
  x   then                                                                   x+
  x           for i in "$@"                                                  x+
  x           do                                                             x+
  x                   spam="$spam \"$i\""                                    x+
  x           done                                                           x
  x           eval chuser $spam                                              x
  x   fi                                                                     x
  x   }                                                                      x
[Mx   x gecos='Stewart Watkiss' stewart                                      x
  x                                                                          x
F1x F1=Help                 F2=Refresh              F3=Cancel                x
F5x F8=Image                F10=Exit                Enter=Do                 x
F9mqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqj
```

Pressing F6 will show details of the command to be executed. This can be useful if you want to know the instruction that you could enter on the command line to perform the same function. This could either show a single command or a script file.

Image Screen

```
                    Change / Show Characteristics of a User

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

[TOP]                                                   [Entry Fields]
* User NAME                                           stewart
  User ID                                            [202]                     #
  ADMINISTRATIVE USER?                                false                   +
  Primary GROUP                                      [staff]                  +
  Group SET                                          [staff]                  +
  ADMINISTRAlqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqk          +
  ROLES     x                     PRINT SCREEN                     x          +
  Another usx                                                      x          +
  SU GROUPS x Press Enter to save the screen image                 x          +
  HOME direcx     in the log file.                                 x
  Initial PRx Press Cancel to return to the application.           x
  User INFORx                                                      xs]
  EXPIRATIONx   Current fast path:                                 x
[MORE...37] x       "chuser"                                       x
            x                                                      x
F1=Help     x F1=Help          F2=Refresh       F3=Cancel          x
F5=Reset    x F8=Image         F10=Exit         Enter=Do           x
F9=Shell    mqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqj
```

Pressing F8 (the image key) will give you a fast path to get to the screen. This can save typing in future. Instead of having to go through the menus you could just start SMIT using

```
smit chuser
```

SMIT Logging

Everything that is done is SMIT is logged. The files used are:

- $HOME/smit.log
  This holds all menu and dialog screens visited, all commands executed and their output. It also records any errors occurred in SMIT.
- $HOME/smit.script
  This is a shell script containing all AIX commands executed by SMIT. This is useful if you want to automate certain commands into scripts.

These files are appended to by every session in SMIT. A system administrator may use SMIT a lot and therefore will have quite large log files, certainly the logs belonging to root will tend to get large overtime. These files should be cleaned up when appropriate.

SMIT options

There are a number of command line options that can be used with SMIT to control what it does. I have already mentioned one of these which is the fastpath command. Typing smit followed by the Fast Path will go straight to that screen.

There are however a number of other command options some of which are listed below:

-C	Starts ASCII smit
-M	Starts Motif smit
-h	Help
-l file	Specify a certain log file
-o dir	Specify an alternate ODM directory. Useful for testing new SMIT Panels before committing them. Default ODM directory is /usr/lib/objrepos
-s file	Specify an alternative for the SMIT script file
-x	Do not run any high-level commands, but still run the discover / list commands. Log the commands for future use. Useful if you want to know the command to run to include in a script file.
-X	Do not run any commands including high-level commands and internal discover / list commands.

Installing Extra Software

There is extra software provided with AIX as well as software that can be obtained from third parties (either at extra cost or available for free). These can optionally be installed after the operating system has been installed.

Normally there will be one or more Bonus Pack CD's included with the AIX CD's. The contents vary over time however an example CD might include:

Adobe Acrobat Reader, Application Development Libraries, DB2 Connect, Encryption for IP security, DSOM, Workgroup Conferencing tools, DCE, Further Networking Support, Mulitimedia tools, Java Libraries, Lotus Domino Go Web Server, IPF, Webserver Search Engine, Netscape and Voicetype Dictation.

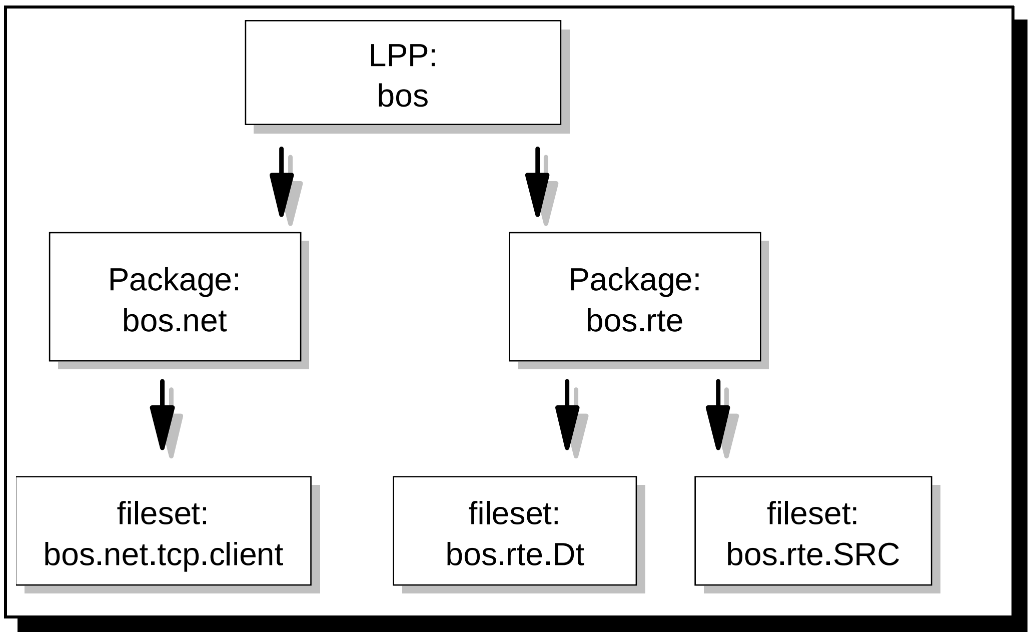
A good source for Free AIX software is the "Bull" Internet site at http://www.bull.de/pub/

Whilst software can be obtained in a number of different forms including source code that needs compiling the most convenient way to get files (which is used by IBM and the downloads available from Bull) is in the package format.

Also see Appendix D for information on the RPM package format.

AIX Packages and Bundles

Files are packaged using a hierarchical naming convention. This makes it easier to locate the particular files associated with a package. This is shown in the diagram below.



Packaging Definitions

This introduces three new definitions.

- Licensed Program Product (LPP) This is a complete software product collection. The Base Operating System (bos) is a LPP which is a collections of packages and filesets.
- Package - This is a group of filesets with a common function. This can also be referred to as an installable image, the bos.net package contains all the networking functions required by the Base Operating System.
- Fileset - This is the smallest individually installable unit. It is a collection of files that provide a specific function. For example the bos.rte.Dt fileset is the dt (CDE) code within the run time environment (rte) package in the Base Operating System.

Fileset naming

Filesets are named as follows:

LPP	**.**	**Package**	**.**	**Fileset**	**.**	**Suffix**
bos	.	terminfo	.	print	.	data

or “bos.terminfo.print.data”

The convention is
LPP.msg[.lang].package.fileset

The names are designed to be self explanatory and in most cases are. The fileset suffixes can sometimes be a little confusing so I have listed them below.

.adt	Application Development Toolkit
.com	Common Code between two like filesets
.compat	Compatibility code that will be removed in a future release
.data	Portion of a fileset to be stored in /usr/share
.dev	Device Support
.diag	Diagnostics
.fnt	Fonts
.info[lang]	InfoExplorer databases (not used much in AIX V4)
.help[lang]	Translated Help files
.loc	Locale specific files
.mp	Multiprocessor specific code
.rte	Run time or minimum set of files to run
.smit	SMIT tools and dialogs
.ucode	Microcode
.up	Uniprocessor specific code

Bundles

Rather than having to search through all the different filesets some are grouped together in a collection known as a “Bundle”. A bundle is designed to install all the filesets required to fulfil a certain requirement e.g. as a client or server. This is also known as “Easy Install”.

The standard bundles available are:

Server
Client
App-Dev
DCE-Client
Pers-Prod
Media-Defined

There can also be customised bundles.

Sharing Software Components

UNIX allows Software components to be shared between different machines. A particular use of this is disk less workstations which need to download the run time code from a server over the network.
For this reason there are 3 different parts to software components.

root '/'
The root part is specific to that particular machine. It cannot be shared across different computers.

usr '/usr'
This is code that can be served to other systems. This can only be shared with other systems also running AIX.

share '/usr/share'
This is code that can be shared with other systems. These files are not specific to AIX and can be shared between different UNIX operating systems.

Software Updates

Software updates can be obtained from:

IBM Support Centre or
FixDist

The FixDist is a free service allowing the downloading of fixes to your own system. Access is via anonymous FTP. There is a FixDist GUI Interface to download the fixes or they can be downloaded using the get command. The FixDist files are accessed via the Internet. Fixes can be applied as Update Bundles, Maintenance Level Bundles and Individual Fixes.

- Update Bundle
  A collection of fixes and enhancements that update software products on the system
- Maintenance Bundle Collection
  A collection of fixes and enhancements that update the operating system

These contain Fixes to known problems, Support for new devices and additional functionality.

Obtaining Fixes (FixDist)

FixDist can be downloaded from the IBM ftp servers. FixDist is not a product for installing fixes it downloads them to your machine allowing you to apply the fixes yourself

The following is a list of anonymous ftp servers can be used:

United Kingdom	ftp.europe.ibm.com	193.129.186.2
United States	service.boulder.ibm.com	198.17.57.66
Canada	rww.aix.can.ibm.com	204.138.188.126
Germany	www.ibm.de	192.109.81.2
Japan	fixdist.yamato.ibm.co.jp	203.141.89.41
Nordic	ftp.nordic.ibm.com	

Table of Anonymous FTP Servers for FixDist

or from the Service Centre at:
http://service.software.ibm.com/aix.us/aixfixes

Connect to any of the ftp servers from the /tmp directory

```
login: anonymous
password: your e-mail address
bin
cd aix/tools/fixdist
```

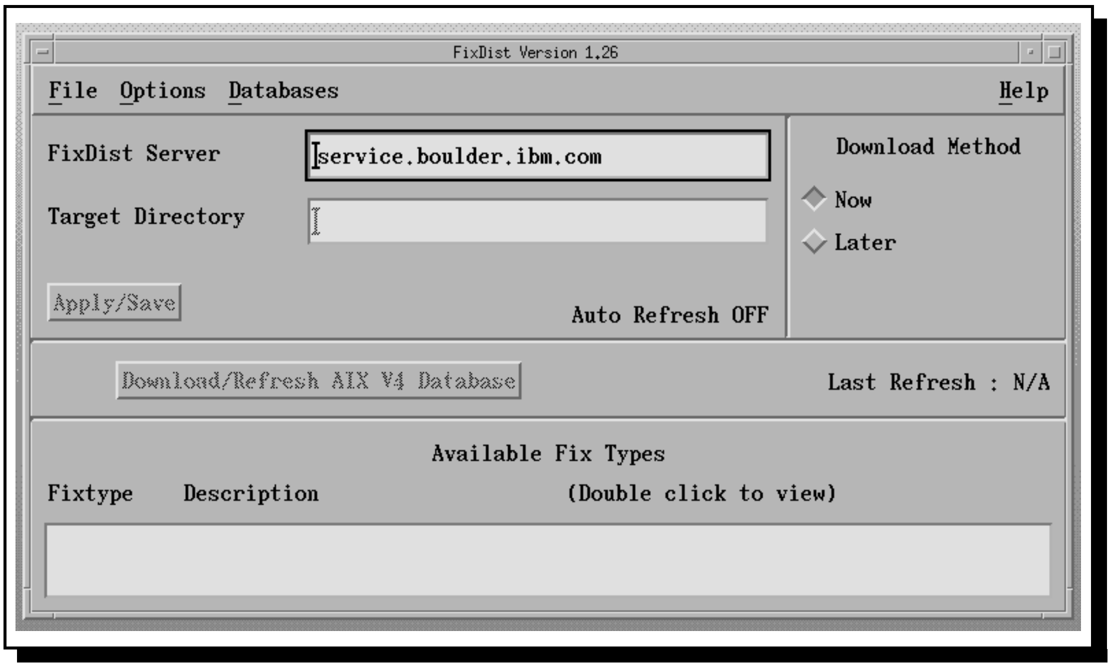
`get fd.tar.Z` (the program code)
also get one of the user guides (available in a variety of formats).
`quit`

Once obtained change to the root directory and

```
zcat /tmp/fd.tar.Z | tar -xpvf -
```

Then to run the program type fixdist. If it can this will run a Motif (X-Windows client) otherwise it will run a text version.

The fixdist program is shown below:



FixDist Version 1.26

File Options Databases Help

FixDist Server: service.boulder.ibm.com

Target Directory:

Download Method: Now / Later

Apply/Save

Auto Refresh OFF

Download/Refresh AIX V4 Database

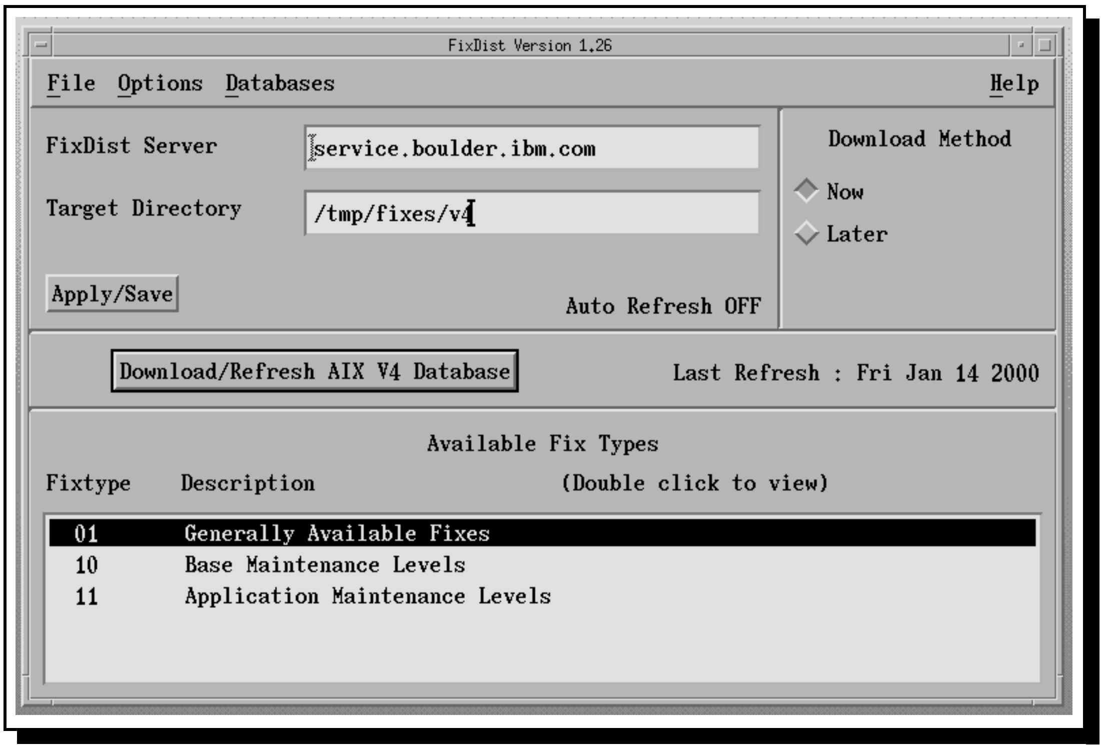
Last Refresh : N/A

Available Fix Types

Fixtype Description (Double click to view)

Fixdist program

You should fill in the target directory as being a local directory on your machine e.g. /tmp/fixes/v4 and then click Download/Refresh Database.



FixDist Version 1.26

File Options Databases Help

FixDist Server: service.boulder.ibm.com

Target Directory: /tmp/fixes/v4

Download Method: Now / Later

Apply/Save

Auto Refresh OFF

Download/Refresh AIX V4 Database

Last Refresh : Fri Jan 14 2000

Available Fix Types

Fixtype	Description (Double click to view)
01	Generally Available Fixes
10	Base Maintenance Levels
11	Application Maintenance Levels

Available Fixes in FixDist

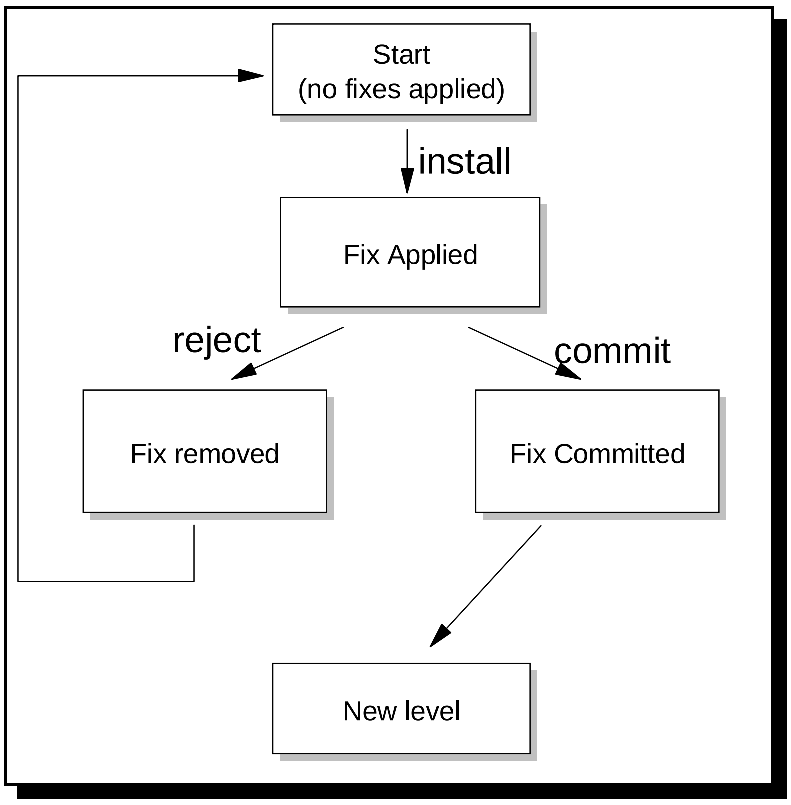
The different fix types can be applied by double clicking on the appropriate choice in the above example, 01 allows you to pick specific fixes, whereas 10 and 11 contain fix bundles for the system and applications respectively.

All details are logged in the target directory in a file called ptfload.log which should be checked if the download fails in any way.

Once you have downloaded the files using FixDist you should install them as specified below.

Applying Fixes and Updates

When fixes are put on a system they are not always immediately committed in place. The reason for this is that there is a slight chance that the fixes could prevent your existing system from working. To allow for this AIX has a number of different states. Depending upon the state of the fix depends upon whether or not it can be removed.



Status diagram for fixes and updates

When the software is first installed or an update is applied it goes into the applied state. Once the reliability of the new software has been determined the software can either be rejected (removed) or committed. If the software is in the applied state it requires more disk space (the

old copy is retained) and it prevents future reinstallation. Upgrades can only be made to software that has already been committed.
The default is normally to commit software when installing, however if you want to be able to remove the update you should change this to no and change “save replaced files to no” (see later for details of installing and upgrading software).

Installing Software, Upgrades and Fixes

Software can be installed and upgraded using SMIT or the WSM however I will show the use of SMIT only.

To get to the Install page go to the option “Software Installation and Maintenance” or use the fastpath install.

```
                     Software Installation and Maintenance

Move cursor to desired item and press Enter.

  Install and Update Software
  List Software and Related Information
  Software Maintenance and Utilities
  System Backup Manager










F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Choosing Install and Update gives the following screen.

```
                          Install and Update Software

Move cursor to desired item and press Enter.

  Install and Update from LATEST Available Software
  Update Installed Software to Latest Level (Update All)
  Install and Update Software by Package Name (includes devices and printers)
  Install Software Bundle (Easy Install)
  Update Software by Fix (APAR)
  Install and Update from ALL Available Software









F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

From this screen there are a number of options explained below:

- **Install & Update from LATEST Available Software** - Install and/or update from latest levels of software available on installation media
- **Updates Installed Software to Latest Level** - Updates all currently installed filesets to the latest level available on installation media. Also used to update currently installed software to a new maintenance level. For example this could be used to upgrade from 4.3.2 to 4.3.3.
- **Install & Update Software by Package Name** - Install & update selected packages. Use to add additional device and printer software.
- **Install Software Bundle** - Install software grouped into a bundle (see earlier explanation).
- **Update software by Fix (APAR)** - If you want to update to fix a specific bug you may have found / been given an APAR number. You should download the files corresponding to this APAR (e.g. FixDist).
- **Install & Update from all available software** - If none of the above fit your circumstances, this option will allow you to do just about anything else.

To update and install software choose the appropriate option. I have chosen Install & Update from LATEST Available Software however with the exception of the Package, or Bundle options the different options give a similar interface.

When prompted for device you should enter the location of the software code e.g. /dev/cd0 for CD-ROM, /tmp/fixes/v4 if that directory was used to download fixes using fixdist, or /usr/sys/inst.images if downloaded images are placed in that directory.

The preview option provides a display of space requirements, and list of updates that will be installed.

As explained earlier you may want to apply updates without committing them. In this case set COMMIT software updates to no and SAVE replaced files to yes.

```
               Install and Update from LATEST Available Software

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                        [Entry Fields]
* INPUT device / directory for software               /dev/cd0
* SOFTWARE to install                                [_all_latest]          +
  PREVIEW only? (install operation will NOT occur)    no                    +
  COMMIT software updates?                            yes                   +
  SAVE replaced files?                                no                    +
  AUTOMATICALLY install requisite software?           yes                   +
  EXTEND file systems if space needed?                yes                   +
  OVERWRITE same or newer versions?                   no                    +
  VERIFY install and check file sizes?                no                    +
  Include corresponding LANGUAGE filesets?            yes                   +
  DETAILED output?                                    no                    +
  Process multiple volumes?                           yes                   +



F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

The instfix command can be used to install a fix using an APAR (Authorised Program Analysis Report) number. The APAR number will normally be given by the support Centre in response to a problem report or by using the RETAIN system (this is a problem reporting tool used by IBM, some customers and business partners can have access and can perform searches on problems and APARs).

The options available are:

-T	Display table of contents
-s	Search for and display table of contents entries containing the string
-k	Install filesets for a keyword or fix
-f	Install filesets for multiple keywords or fixes using an input file using a dash '-' will result in this been taken from standard input. -T will produce the correct format for the file
-i	Show whether a fix is installed or not this will not install any updates
-d	Specify the input device.

To view security related fixes:

```
# instfix -s security -d /tmp/fixes/v4
IX55285 src/security/server/rs/rs.c compile errors.
IX56672 Drop kerberos security exposure fix
IX57331 Apparent memory leaks in security API's to get login context.
IX73099 TFTP following symlinks overrides tftpaccess.ctl security
IX73438 dtappgather security vulnerability: CERT VU#15575
IX73586 security hole in ftp, tftp, utftp
IX74303 Incomplete default X usage message for C2 security
IX74793 security hole in tn3270
IX77163 Can't create a keytab entry for security client's host
IX78349 SECURITY: bad permissions on /etc/security/login.cfg
IX78785 BLDBREAK: sysck: The file /etc/security/acl was not found.
IX79226 directed security initrep added for reliability
IX79227 memleaks in security replication for attribute data
IX81560 chsec not mofidying group and login.cfg in /etc/security
IX81625 Errors applying bos.rte.security in 9832B updates.
IX82452 HOT:reject of bos.rte.security of 9835B failed
```

To check for a specific fix being applied

```
instfix -i -k "IX73586"
```

to apply the fix

```
instfix -k IX73586 -d /tmp/fixes/v4
```

To manage software that is not committed you can use the Software Maintenance & Utilities screen (available from initial software screen in smit).

```
                       Software Maintenance and Utilities

Move cursor to desired item and press Enter.

  Commit Applied Software Updates (Remove Saved Files)
  Reject Applied Software Updates (Use Previous Version)
  Remove Installed Software

  Copy Software to Hard Disk for Future Installation

  Check Software File Sizes After Installation
  Verify Software Installation and Requisites

  Clean Up After Failed or Interrupted Installation







F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Most of the options are pretty self explanatory and straight forward to use. Committing the updates will remove the saved files and allow future updates to be applied.

Viewing installed software

It is possible to view the installed software using smit or by using the lslpp command.

lslpp can be used with the following options

-L	Show name, level, status and description of the fileset
-l	Separate usr, root and share parts of the software
-h	Show installation and update history for the fileset
-p	Show requisite information
-d	Show dependency information
-f	Show names of files added during installation
-w	List the fileset that owns a file

```
# lslpp -L bos.rte.*
  Fileset                      Level  State  Description
  ----------------------------------------------------------------------------
  bos.rte.Dt                 4.3.0.0    C    Desktop Integrator
  bos.rte.ILS                4.3.2.0    C    International Language Support
  bos.rte.SRC                4.3.2.0    C    System Resource Controller
  bos.rte.X11                4.3.0.0    C    AIXwindows Device Support
  bos.rte.aio                4.3.2.0    C    Asynchronous I/O Extension
  bos.rte.archive            4.3.2.0    C    Archive Commands
  bos.rte.bind_cmds          4.3.2.0    C    Binder and Loader Commands
  bos.rte.boot               4.3.2.0    C    Boot Commands
```

Online Documentation

With the web browser taking an increasingly important role in modern computers the AIX documentation is now available in a “Web based format”. This is achieved by a server that is acting as a “documentation server” providing the documentation for any client using a web browser. The server could also be a client targeting itself for the files.

Installing the Online Documentation

To setup the documentation server the following steps need to be carried out

- Configure TCP/IP (see networking section)

- Install Web Server Software
- Configure and start the web server
- Install the web browser
- Install or mount the AIX documentation
- Install and configure the documentation search service

1. TCP/IP is often configured during the initial setup of the operating system.
2. There are a number of different web server software packages that can be used. Three web servers are provided as part of AIX V4.3 which are Netscape Fasttrack server, Lotus Domino Go server and the IBM Internet Connection Server. Other software could be used, e.g. Apache's free web server software, however the software must support CGI (Common Gateway Interface).
3. Configure and Start the web server software. Lotus Domino Go is the default web server software
4. Install Web Browser Software, this is only required if users on that system need to access the documentation. The browser could be installed on a PC or other computer attached to the network. Netscape is provided with AIX V4.3 however any browser supporting HTML V3.2 and Frames can be used.
5. The Base Documentation includes manuals that are used most frequently. This can be installed onto the system or mounted as a CD-ROM file system. The extended documentation contains adapter guides, reference books and technical references. The extended documentation can only be mounted as a CD-ROM file system, it cannot be installed.
6. Install and configure the documentation search service (bos.docsearch) then use smit web_configure to configure the system.

```
                      Internet and Documentation Services

Move cursor to desired item and press Enter.

  Change / Show Default Browser
  Show Documentation and Search Server
  Change Documentation and Search Server
  Web-based System Manager

















F1=Help               F2=Refresh            F3=Cancel             F8=Image
F9=Shell              F10=Exit              Enter=Do
```

From the web configure screen you need to set the browser and server directories etc.

Viewing the Online Documentation

You can view the online documentation on either the web browser on the AIX system or from a web browser on another computer.

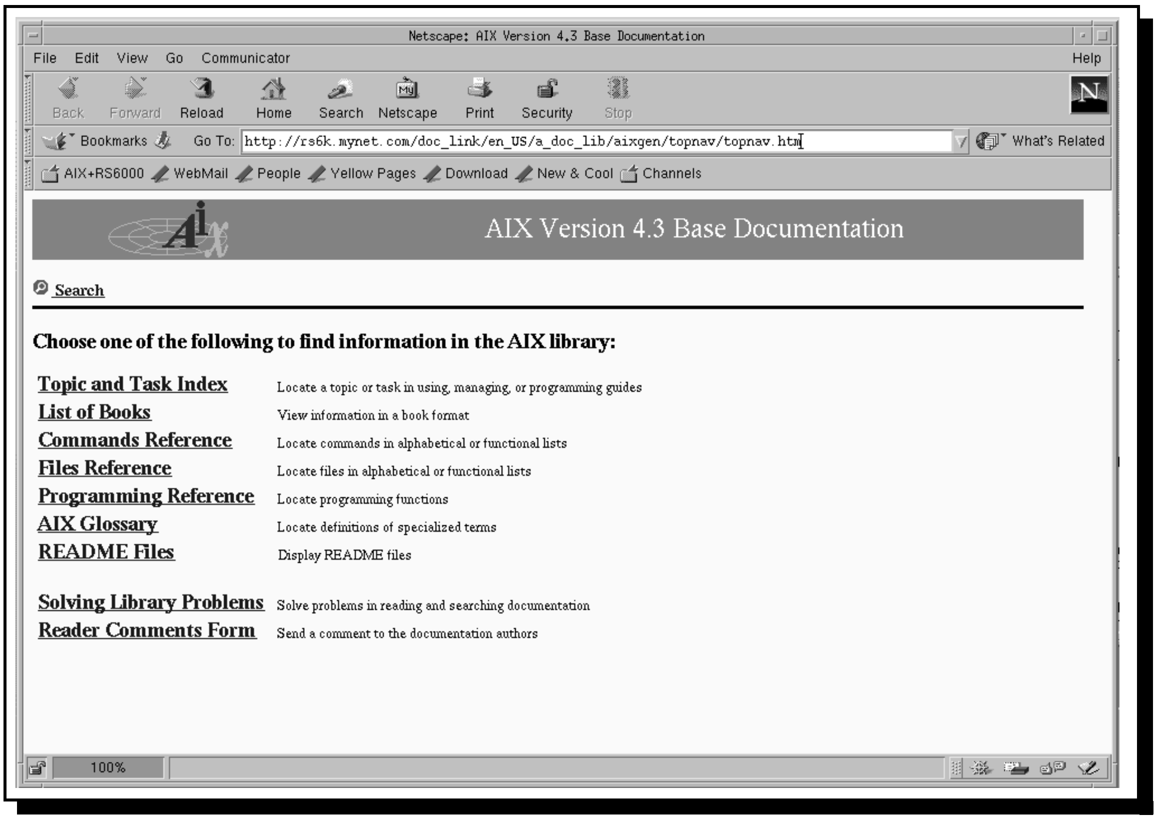
The URL for the page will be:

http://<hostname>/doc_link/en_US/a_doc_lib/aixgen/topnav/topnav.htm

or the extended documentation at:

http://<hostname>/doc_link/en_US/a_doc_lib/aixgen/wxinfnav/topnav.htm

The documentation can also be accessed using the Base Documentation icon using CDE.


Netscape: AIX Version 4.3 Base Documentation
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop
Bookmarks Go To: http://rs6k.mynet.com/doc_link/en_US/a_doc_lib/aixgen/topnav/topnav.htm What's Related
AIX+RS6000 WebMail People Yellow Pages Download New & Cool Channels
100%

AIX Version 4.3 Base Documentation

Search

Choose one of the following to find information in the AIX library:

Topic and Task Index	Locate a topic or task in using, managing, or programming guides
List of Books	View information in a book format
Commands Reference	Locate commands in alphabetical or functional lists
Files Reference	Locate files in alphabetical or functional lists
Programming Reference	Locate programming functions
AIX Glossary	Locate definitions of specialized terms
README Files	Display README files
Solving Library Problems	Solve problems in reading and searching documentation
Reader Comments Form	Send a comment to the documentation authors

AIX documentation Home Page

Printers

There are 5 components in the printer subsystem for AIX. These are:

- Queues - These manage printer use. This is particularly important when there are multiple printers and / or users on a system, where several print requests may be waiting to be printed.
- Spooler - The AIX spooler is not purely for printing jobs. It is a job scheduler, however print jobs can be considered as other jobs waiting to be sent to the printers.
- Backends - These are the programs that manage a job that is queued for printing. There are a number of different backends available (including custom ones) which can handle different types of printing, e.g. remote printers, local printers, network printers etc.
- Virtual Printers - These are a set of attributes defining the data stream that the printers understand. Associated with print queues a different virtual printer can be defined for each data stream a printer understands (e.g. pcl and postscript).
- Real Printers - These are the physical devices attached to the system.

Printing Process

The following process is followed when a job is printed.

1. The user submits the job using, lp, lpr, qprt or enq (these could be called from within an application).
2. The command translates the options into enq's options and calls the enq command.
3. The enq command checks the validity of the queue name specified in the /etc/qconfig file. If it fails at this stage an error is returned to the user and the command fails. The file is put into the qdaemon job directory. This is /var/spool/lpd/qdir
4. The qdaemon is notified of a new job in it's qdir.
5. When the queue is ready to process the job the qdaemon reads the information held in /etc/qconfig describing the queue. The binary command used to perform this is qconfig.bin (also in the /etc directory). The qconfig.bin file is the result of the digest command executed by the qdaemon. This command cannot be run from the command line and should be run by the qdaemon only. The qdaemon updates the stat file for the appropriate queue to indicate that the queue is now working on a new job.
6. The qdaemon starts the backend program passing the file names and options on the command line. Information on the job status is held in /var/spool/lpd/stat for use by the qdaemon and the backend program.
7. The backend program chooses the appropriate virtual printer and processes the file to be printed for that virtual printer.
8. The backend program sends is data stream to the device driver for the printer.
9. The printer then produces the print out.

Configuration of Print Queues

The qconfig file is used by the print spooling subsystem to control configuration of the print queues and the backend program. The qconfig file is a stanza delimited file that describes the printer spool queues and the backend devices they involve. Stanzas in the file represent either a queue or a queue device. Every queue entry requires at least one queue-device defined beneath it. The first entry in the file indicates the default print queue. This can be overridden with the variables LPDEST, or PRINTER (LPDEST takes precedence over PRINTER).

Note: the following displays show a remotely connected printer. The Device name is taken from the IP address of the printer with a class A address of 10

```
lp0:
       discipline = fcfs
       up = TRUE
       device = dlp0

dlp0:
       backend = /usr/lib/lpd/piobe
       file = FALSE
       access = write
       feed = never
       header = never
       trailer = never
```

The lsallq command lists defined queue names.

```
$ lsallq
lppcl
lpps
$
$ lsallq -c
lppcl
lppcl:@10
lpps
lpps:@10
```

The lsquedev command shows the requested extract from the /etc/qconfig

```
$ lsquedev -q lpps -d @10
@10:
        header = never
        trailer = never
        access = both
        backend = /usr/lib/lpd/pio/etc/piorlfb -f !
```

It is possible to have two queue-devices associated with a queue is that it allows a print job sent to the printer to be printed on the first available printer.

In AIX V4 it is possible to define a special batch queue. The batch queue has a backend of /usr/bin/bsh (bourne shell). This would provide a queue that could process shell scripts one at a time. This can be used to run scripts overnight.

```
bsh:
       device = bshdev
       discipline = fcfs
bshdev:
       backend = /usr/bin/bsh
```

The backend program manages the printing of jobs. It is used to create the data stream that communicates with the printer, obviously it therefore needs to be different when different printers are involved (particularly remote / network printers etc.). The default printer backend is piobe (Printer IO BackEnd).

Virtual Printers

As mentioned earlier it is possible to have more than one device for each print queue. It is also possible to have more than one print queue for each device. The first is used to provide shared printing between multiple printers (this provides load balancing between more than one printer or a backup in case of failure of one of the printers) the job will be printed on whichever printer is available at the time. The second is used where a printer is capable of supporting more than one print stream. For example you may want to print a text file to the printer directly using the lp command or you may want to print in postscript using a graphics application

A virtual printer is created using smit mkvirprt, and can be changed using smit chvirprt.
A custom virtual printer file can be used by using the piopredef command.

The configuration is held in the following directories:

Predefined devices
/usr/lib/lpd/pio/predef

ibm*xxxx*.asc	(IBM printer - ASCII mode)
hplj*xxxx*.gl	(HP printer - hpgl mode)
generic.pcl	(Generic printer - pcl mode)
lex*xxxx*.ps	(Lexmark printer - postscript mode)

Custom Devices
/usr/spool/lpd/pio/@local/custom

gl:lp0
lp0:lp0
pcl:lp0
ps:lp0

Digested file for a customised printer
/var/spool/lpd/pio/@local/ddl

xxxx.asc.lp0.def.lp0
xxxx.gl.lp0.gl.lp0
xxxx.pcl.lp0.pcl.lp0
xxxx.ps.lp0.ps.lp0

Header and Trailer Pages
/usr/lib/lpd/pio/burst

Executables
/usr/lib/lpd/pio/etc
piodigest
pioburst

There are also files held in the /usr and /var directories.

/usr/lib/lpd
aixlong, aixshort, bsdlong, bsdshort, attlong, attshort, piobe and rembak.

In the directory /usr/lib/lpd/pio there are a number of subdirectories :
burst, etc, fonts, predef, trans1, trans2

Header and Trailer Files

The header and trailer pages can be customised. They are held in /usr/lib/lpd/pio/burst
The files are:

H.ASCII	T.ASCII	ASCII print queues
H.gl	T.gl	GL print queues
H.ps	T.ps	Postscript queues

To customise the header or trailer files edit the header or trailer files or a copy of them. If you are editing a copy of them then you need to edit the virtual printer to point at the new file. This is done by editing the virtual printer, finding the sh attribute and changing the burst page file name to the new file name.

In the following example I've replaced the computers hostname with a more meaningful "Stewart's RS6K".

```
crlf
crlf
(%t  %T ) wrap_lines
crlf
(%p  %P ) wrap_lines
crlf
(%q  %Q ) wrap_lines
crlf
(%h  Stewart's RS6K ) wrap_lines
crlf
(%s  %S ) wrap_lines
crlf
(%d  =====> %D <===== ) wrap_lines
crlf
```

The entry in the virtual printer is in directory /usr/spool/lpd/pio/@local/custom

```
:321:sh::%Ide/pioburst %F[H] %Idb/H.ps | %Ide/pioformat -@%Idd/%Imm -!%Idf/piofp
t %f[jJ] %IsH -u%IuH -=%I_=
:324:st::%Ide/pioburst %F[H] %Idb/T.ps | %Ide/pioformat -@%Idd/%Imm -!%Idf/piofp
t %f[jJ] %IsT -u%IuT -=%I_=
```

Then run piodigest or chvirprt to digest modification

The following variables can be used in the header and trailer files:

%A	Formatting flag values
%D	The user to deliver the print out to
%H	The hostname of the machine printing the job
%P	Time the print job was printed
%Q	Time the print job was queued
%S	The user who submitted the job
%T	The title of the print job
%%	Print a '%' character

Printing ASCII Files on a Postscript Printer

Sending an ASCII file directly to a postscript printer will not print correctly. However using the qprt command a ASCII file can be printed to a postscript printer.

```
qprt -da -Pps asciifile
```

Configuring a Non-supported Printer

There are 4 different methods of configuring printers where support is not provided by AIX.

1. Choose a supported device that the printer is capable of emulating.
2. Configure the printer as a supported one, then change the virtual printer definitions.
3. Configure the printer as a generic serial/parallel printer and modify it's characteristics.
4. Configure the print subsystem to pass all print requests to the printer in pass-through mode. This allows print requests to be formatted by an application and sent as RAW to the printer. This is useful when using AIX to run a print server with users printing to the AIX server remotely from other computers.

To set the printer in pass through the qconfig file should have the entry with piobe -d p piobe then sends the print request to the printer unmodified. Alternatively for an individual job the qprt command can be used.

```
qprt -d p filename
```

Or change the _d attribute on the virtual printer to p.

Printer Troubleshooting

Here’s a quick checklist to help resolve printer problems:

1. Check the physical connections, cabling etc, and that the printer is switched on and online.
2. Ensure there is sufficient paper in the printer (including alternative trays).
3. Try sending a text file directly to the printer. This can be done using cat (`cat /etc/qconfig > /dev/lp0`). This will bypass the spooling system and check that the printer and cabling are all OK.
4. Check the queues are properly configured and that there is at least one virtual printer per queue. A problem can occur if a device has been removed but the queue is still in place. Errors may be logged in the qdaemon log so check there for errors.
5. Check that the /etc/qconfig file is intact and not corrupted. Make corrections as required which will cause qconfig.bin to be automatically recreated.
6. List running processes and check that the srcmstr and qdaemon are running. There should only be one instance of each. If there are any qdfork processes these should be killed off.
7. Check the status of the print queues. Use the lpstat command and check the status at the top of the queue. If the printer is in DEV_WAIT then it is likely to be a hardware problem. If it is OPR_WAIT then the printer is likely to be in need of more paper or require another operator action.
8. Check the amount of free space in /tmp and /var. If these are full then there may be a message such as “No virtual Printers Defined”. If /var is full then the queues may be brought down. If only root can print than check for write permission to /tmp.
9. Check to see what processes are using the printer. Use the command `fuser /dev/lp0` using -k will automatically kill the processes using the printer.
10. Clear the spooling system to remove any jobs from the queues.
    ```
    stopsrc -s qdaemon
    cd /var/spool/lpd/stat ; rm *
    cd /usr/spool/lpd/pio/@local/custom ; rm *
    cd /var/spool/lpd/pio/@local/ddi ; rm *
    cd /var/spool/lpd/qdir ; rm *
    cd /var/spool/qdaemon ; rm *
    startsrc -s qdaemon
    ```
11. If the printer queue is in OPR_WAIT try sending a write -h n,ok signal (n is job number).
12. Check the system time is correct. The qconfig.bin is updated based on the date of the qconfig file.

Remote Printing

There are several different ways that remote printing can be done. Using:

Remote Queue - The traditional method of remote printing is to have the printer directly attached to a server and clients.
Network Attached - The printer is directly attached to the network and a queue is created on a host server and are directed to the LAN attached printer.
Terminal Server - A separate server attached directly on the network to which terminals, printers and modems are connected. A queue is setup on the host system to send jobs to the terminal server.
Multi-Protocol Network Print Server - A server that handles print requests from hosts and using different network protocols. Printers are directly attached to this.

It is also possible to use SAMBA which is freely available software which allows communication with printers running on Windows Servers.

Configuring a Client for Remote Printing

To configure a client a remote print queue must be created on the local system. The easiest way to do this is to use SMIT.

```
smit spooler
```

```
                                 Print Spooling

Move cursor to desired item and press Enter.

  Start a Print Job
  Manage Print Jobs
  List All Print Queues
  Manage Print Queues
  Add a Print Queue
  Add an Additional Printer to an Existing Print Queue
  Change / Show Print Queue Characteristics
  Change / Show Printer Connection Characteristics
  Remove a Print Queue
  Manage Print Server
  Programming Tools







F1=Help                 F2=Refresh              F3=Cancel               F8=Image
F9=Shell                F10=Exit                Enter=Do
```

Choose “Add a Print Queue”

```
                                 Print Spooling

Move cursor to desired item and press Enter.

  lqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqk
  x                           Add a Print Queue                             x
  x                                                                         x
  x Move cursor to desired item and press Enter. Use arrow keys to scroll.  x
  x                                                                         x
  x   # ATTACHMENT TYPE      DESCRIPTION                                    x
  x     local                Printer Attached to Local Host                 x
  x     remote               Printer Attached to Remote Host                x
  x     xstation             Printer Attached to Xstation                   x
  x     ascii                Printer Attached to ASCII Terminal             x
  x     hpJetDirect          Network Printer (HP JetDirect)                 x
  x     file                 File (in /dev directory)                       x
  x     ibmNetPrinter        IBM Network Printer                            x
  x     ibmNetColor          IBM Network Color Printer                      x
  x     other                User Defined Backend                           x
  x                                                                         x
  x F1=Help                 F2=Refresh              F3=Cancel               x
  x F8=Image                F10=Exit                Enter=Do                x
F1x /=Find                  n=Find Next                                     x
F9mqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqj
```

Which then leaves the following options:

Standard Processing - Traditional remote printing. Print files are sent to the server without modification assuming that any filtering is done by the server.
Standard with NFS access to server print queue attributes - This is similar to the standard processing except that the server’s directory of print attributes is NFS mounted by the client. The server must have exported /var/spool/lpd/pio/@local
Local filtering before sending to print server - Allows the client to filter the print files before sending to the printer

As Standard Processing is the most common this is the one covered here. For more information about the other options see the AIX documentation.

```
                       Add a Standard Remote Print Queue

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                        [Entry Fields]
* Name of QUEUE to add                                [remotelp]
* HOSTNAME of remote server                           [lpserver]
* Name of QUEUE on remote server                      [remqueue]
  Type of print spooler on remote server               AIX Version 3 or 4
  +
  Backend TIME OUT period (minutes)                   []
   #
  Send control file first?                             no
  +
  To turn on debugging, specify output                []
      file pathname
  DESCRIPTION of printer on remote server             []





F1=Help             F2=Refresh          F3=Cancel           F4=List
Esc+5=Reset         F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

The next page will create the information in the /etc/qconfig file. The entry created might be:

```
remotelp
      device = @lpserver
      up = TRUE
      host = lpserver
      s_statfilter = /usr/lib/lpd/aixshort
      l_statfilter = /usr/lib/lpd/aixlong
      rq = remqueue

@lpserver:
      backend = /usr/lib/lpd/rembak
```

The following values have been used

Local Queue Name (used when printing) - remotelp
Remote Server Name (host name) - lpserver
Queue name on remote server - remqueue

The statfilter files may have to be changed depending upon the operating system for the remote system. For example a BSD system has a different file to AIX. It’s possible that the remote server may not be a UNIX machine (e.g. OS/2 or Windows) however will probably emulate the BSD functions for the purposes of remote printing.

Configuring a Server for Remote Printing

Firstly the local queue must be defined. This follows the standard procedure discussed earlier.

For security reasons it is necessary to indicate what hosts will be allowed to print on the server. The file providing a list of the clients able to print is /etc/hosts.lpd . If using NFS or some of the ‘r’ commands then a system may have an entry in /etc/hosts.equiv in which case this would be authorised as well. Unless you do however need the extra features in the hosts.equiv file it is highly recommended that the hosts.lpd file is used instead. Whilst this can be done using smit mkhostslpd it can only be done for a single system at a time. Therefore after the first entry has been added it is often easier to edit the /etc/hosts.lpd file directly.

The daemon that handles remote print requests is called the lpd daemon and this must be started. This can be done using SMIT using:

smit mkitab_lpd

Or just by starting the daemon using startsrc. To have the daemon start at reboot requires an entry in /etc/inittab, if SMIT was used this would have been done anyway (depending upon what option was chosen) . Rather than edit the file directly it is better to the use the mkitab command.

Using the Remote Printer

Once configured the remote printer can be used the same as if it was a local printer. For example using the lp or lpr commands. It is also possible to query the status of a printer and it’s jobs using qchk. Issuing a qchk -A will shows the stat of all the jobs and can be run on both the server and the client.

An example of qchk on the client is shown below:

```
$ qchk -A
Queue   Dev   Status    Job Files              User         PP %   Blks  Cp Rnk
------- ----- --------- --- ------------------ ---------- ---- -- ----- --- ---
remotelp pcl   READY
              QUEUED      1 myfile.txt         stewart             371   1   1
              QUEUED      2 README             root                  2   1   2
```

Jobs can be cancelled either on the client or on the server. To cancel a job on the server the command is:

```
qcan -x jobid
```

where jobid is the number of the job. Or this can be done using smit qcan.

From the client the command is either qcan as shown above or:

```
enq -Premq -x jobid
```

or using smit qcan.

Configuring a Network Attached Printer

Some printers have the ability to run as a network printer with a direct connection to the LAN. IBM produce Network Printers 12, 17, 24 and 32, whereas the Lexmark Optra range have similar functionality. On some printers the network address can be configured through the front panel of the printer although normally they are set to use BOOTP/TFTP by default.

It is necessary to load the software onto the AIX client to support the particular printer. Many of the IBM/Lexmark printers are included in the AIX installation however for others the manufacturer of the printer may have to supply the software.

To configure the printer on the client smit mkpq is used:

If the printer requires the client to act as a BOOTP/TFTP this should be selected and details such as the IP address etc by defined.

Common Unix Printing System (CUPS)

Whilst not included in the standard AIX distribution the Common Unix Printing System allows printing to numerous different printers.

The code will have to be obtained (probably in source code and then compiled for AIX). More details can be found at http://www.cups.org/

Devices

There are many different types of devices that can be attached to an AIX system. These vary from devices built into the computer (e.g. an internal disk), to interfaces (e.g. serial ports) to external devices (e.g. external SCSI drives). To be able to use any of these devices the operating system needs to be told about them, so that it knows how to communicate with them.

The way that AIX communicates to the devices requires the following:

Physical Devices - Hardware that is connected to the system
Ports - Physical connectors and adapters that connect the computer to the physical devices.
Device Drivers - Software in the kernel that controls the activity on a port and the format of data that is sent to the device.
Logical Devices - These are special files in the /dev directory. The are software interfaces allowing users and application programs to communicate with the device drivers.
/dev - directory containing all the logical devices that can be directly accessed by the user or applications.

There are two types of device that can be configured, a block device and a character device. The block device is a structured random access device where buffering is used to provide a block-at-a-time access method. The character device (sometimes referred to as a raw device) has a sequential data stream.

Most block devices will also have a character device allowing raw access to the disk. Normally these will have the same name as the block devices however will be prefixed with an ‘r’. For example a hard disk (hd1) will be a block device however can be accessed directly using it’s raw device driver (rhd1).

A few examples of block devices are:

cd0	CD-ROM
fd0, fdl, fd0h	floppy disk
hd1, lv00	logical volume
hdisk0	physical volume

A few examples of character devices are:

console,lft,tty0	terminal
lp0	printer
rmt0	tape drive
tok0, ent0	network adapter
kmem, mem, null	memory
rfd0, rfd0l, rfd0h	floppy disk
rhd1, rlv00	logical volume
rhdisk0	physical volume

You can view the devices by using the ls command on the /dev directory. These show as files however you will see a ‘b’ or ‘c’ at the beginning of the display for a block and character device respectively.

```
# ls -l /dev
total 24
cr--r----T   1 root     system     8,  0 24 Nov 12:02 audit
crw-------   1 root     system    11,  0 24 Nov 11:31 bus0
crw-------   1 root     system    12,  0 24 Nov 11:31 bus1
br--r--r--   1 root     system    15,  0 24 Nov 11:31 cd0
crw-rw-rw-   1 root     system    16,  0 24 Nov 11:31 clone
crw--w--w-   1 stewart  system     4,  0 24 Nov 11:31 console
crw-rw-rw-   1 root     system    16, 33 24 Nov 12:02 echo
crw--w--w-   1 root     system     6,  0 17 Jan 06:16 error
crw-------   1 root     system     6,  1 24 Nov 11:31 errorctl
brw-rw-rw-   1 root     system    19,  0 24 Nov 11:31 fd0
brw-rw-rw-   2 root     system    19,  1 24 Nov 11:31 fd0.18
brw-rw-rw-   2 root     system    19,  2 24 Nov 11:31 fd0.9
brw-rw-rw-   2 root     system    19,  1 24 Nov 11:31 fd0h
```

Using the -l option with ls the 5 column would normally hold the filesize however when displaying a device it shows the major and minor device numbers (major,minor). This is how the system talks to the devices. Each Device driver has a major node number and when the system sends anything to the driver it will refer to the minor node number to specify which logical device it wishes communicating with. The major node is the same for a device whether referring to the block or character logical device however the minor node number will be unique for each one.

Device Configuration Database

There is a device configuration database managed by the ODM that controls the devices available to the system. This is split into two parts, the predefined and customised databases. The predefined database holds details of all devices supported by AIX whereas the customised database holds devices that have been configured to work with the particular setup of AIX.

There is an option to include non-supported devices and with some ISA devices you may be asked for a disk with the appropriate drivers on.

During startup the Configuration Manager goes through the system looking for devices. It uses information in both the predefined and customised databases, updating the customised database with any new devices it finds.

To view the predefined database use the lsdev command.

```
$ lsdev -P -H
class          type            subclass   description

logical_volume vgtype          vgsubclass Volume group
logical_volume lvtype          lvsubclass Logical volume
lvm            lvdd            lvm        LVM Device Driver
aio            aio             node       Asynchronous I/O
pty            pty             pty        Asynchronous Pseudo-Terminal
sys            rspc            node       System Object
memory         L2cache_rspc    sys        L2 Cache
memory         totmem          sys        Memory
planar         sysplanar_rspc  sys        System Planar
processor      proc_rspc       sys        Processor
adapter        baud4232        isa_sio    Ultimedia Integrated Audio
bus            pci             sys        PCI Bus
tape           1200mb-c        scsi       1.2 GB 1/4-Inch Tape Drive
```

The -P option specifies the predefined database and the -H option shows the Titles (Headings). There are nearly 500 entries on my system so you will probably want to reduce the number listed. You can do this by specifying the class of the device e.g. CDROM, printer etc.

```
$ lsdev -PHc cdrom
class type      subclass description

cdrom cdrom1    scsi     CD-ROM Drive
cdrom enhcdrom  scsi     Multimedia CD-ROM Drive
cdrom enhcdrom3 scsi     Multimedia CD-ROM Drive
cdrom oscd      scsi     Other SCSI CD-ROM Drive
cdrom scsd      scsi     Other SCSI CD-ROM Drive
cdrom enhcdrom2 scsi     Multimedia CD-ROM Drive
cdrom enhcdrom4 scsi     Multimedia CD-ROM Drive
```

The devices are classified by different categories:

Class	- What the device does
Type	- What model it is
Subclass	- How it is attached to the system

The lsattr command with -D will show the attributes of a device in the predefined database.

```
$ lsattr -D  -t cdrom1
reserve_lock  no RESERVE device on open              True
prevent_eject no PREVENT ejection of media when open True
```

This can also be done using “Show Characteristics of a Supported Device” within SMIT. This can be reached from “List All Devices”.

The Customised database can be viewed in a similar way specifying -C for customised database.

```
$ lsdev -C -H
name       status    location     description

sys0       Available 00-00        System Object
sysplanar0 Available 00-00        System Planar
bus0       Available 00-00        PCI Bus
bus1       Available 04-A0        ISA Bus
pmc0       Available 01-A0        Power Management Controller
fda0       Available 01-C0        Standard I/O Diskette Adapter
ide0       Available 01-E0        ATA/IDE Controller Device
ide1       Available 01-F0        ATA/IDE Controller Device
sa0        Available 01-G0        Standard I/O Serial Port 1
sa1        Available 01-H0        Standard I/O Serial Port 2
sioka0     Available 01-I0        Keyboard Adapter
sioma0     Available 01-J0        Mouse Adapter
iga0       Available 04-C0        E15 Graphics Adapter
scsi0      Available 04-B0        Standard SCSI I/O Controller
cd0        Available 04-B0-00-3,0 SCSI Multimedia CD-ROM Drive
hdisk0     Available 04-B0-00-4,0 SCSI Disk Drive
mem0       Available 00-00        Memory
proc0      Available 00-00        Processor
```

The status can have the following values

Available : The device is ready and can be used
Defined : The device is unavailable (may be powered off / may no longer exist)

The location code is used to indicate how it connects to the system (what adapter etc). This is explained later.

The lsattr command can be used to list the attributes using either -l for logical device name or -c for the class of the device. The -E option is used to list the effective attributes

```
$ lsattr -EH -l cd0
attribute     value description                        user_settable

reserve_lock  no    RESERVE device on open             True
prevent_eject yes   PREVENT ejection of media when open True
queue_depth   3     Queue DEPTH                        False
size_in_mb    650   Size in Megabytes                  False
```

Device States

There are 4 different statues that a device can be in. These are listed in the table below:

Operating System	**Device State**	**Comments**

Unknown	Undefined	There is no record of the devices existence. It has not been defined.
Unavailable	Defined	A logical device name and port has been allocated. The device is however unavailable to the system.
Defined and Configured	Available	The device is defined, configured and available for use.
Configured but Unavailable	Stopped	The device is unavailable however is still known by it’s device driver.

Different Device States

Example commands to change between the states

The following shoes adding and removing an external tape drive to the system that was not connected when the system was powered on.

- Define the device
  from Undefined to Defined
  SMIT “Define a Tape Drive”
  ```
  mkdev -d tape -t 8mm -s scsi -p scsi0 -w 40
  ```

- Configure the device
  from Defined to Available
  SMIT “Configure a Defined Tape Drive”
  ```
  mkdev -l rmt0
  ```

- Stop a defined device
  from Available to Stopped
  SMIT “Configure a Tape Drive”
  ```
  mkdev -S rmt0
  ```

- Unconfigure a device
  from Available to Defined
  SMIT “Remove a Tape Drive” - Delete from database = no
  ```
  rmdev -l rmt0
  ```

- Remove a Device Permanently (removes from customised database)
  from Available or Defined to Undefined
  SMIT “Remove a Tape Drive” - Delete from database = yes
  ```
  rmdev -l rmt0 -d
  ```

Device Addressing

We have seen earlier a list of device addresses for devices in the customised database. These addresses are used for device addressing and are assigned for every logical device when it is attached to the system. The location code provides the system with a method of locating the device and establishing relationships between devices and their adapters. If a hardware failure occurs the location code may be displayed or referred to.

The location code is split into 4 fields (although they are not always all used depending upon the device). It provides a path from the adapter in the system through the cables etc. to the device.

The format for the location code is:

AA-BB-CC-DD

These can be found in the following groups

AA-BB	Adapter card locations
AA-BB-CC	Built in devices
AA-BB-CC-DD	Device Ports or connectors, e.g. SCSI, printers, terminals etc.

These codes have slightly different meanings depending upon the type of device.

None-SCSI Devices

AA - Usually 00 for the system unit
Any other value indicates it is attached to an expansion drawer in this case the first digit is for the bus and the second the slot.

BB - The first digit is for the system i/o bus identifier. This is 0 for the standard bus and 1 for the optional bus.
The second digit is the slot number of the adapter or memory card

CC - Connector on an adapter or planer 01 to 04

DD - Asynchronous port number or FRU location on a card or planer

SCSI Devices

SCSI devices have a slightly extended location code in that it includes the SCSI id number and Logical Unit number of the device this takes the format **AA-BB-CC-S,L**

AA -	Usually 00 for system unit
BB -	First digit is for the I/O bus and second for the adapter card slot on the bus
CC -	This is 00 for a card providing a single SCSI bus or a device attached to the internal bus on a dual SCSI bus. This is 01 for a device attached to an external bus on a dual SCSI This is 0S for the external bus connector of an integrated SCSI controller
S -	SCSI address of the device
L -	Logical unit number of the device

PCI Location Codes

The codes are slightly different for PCI based systems

AA -	00	Resources attached to the processor
	01	Resources attached to the ISA bus
	04	Resources attached to the PCI bus
BB -	01-99	Pluggable adapters or cards
	A-Z,0	For integrated adapters
	The order is determined by the order in which they are defined and does not represent any particular slot.	
CC -	The connector ID	
DD -	Port identifier, address, memory modules, device or FRU of the device	

The ISA bus is actually defined as an address on the PCI bus. This can be thought of as a PCI to ISA bridge adapter.
For SCSI devices, the DD can be represented as SL the same as SCSI devices on a Classical RS/6000.

Configuration for PCI / ISA devices

PCI Devices

Many PCI devices are self-configuring. Upon system startup (and on demand) the Configuration Manager (cfgmgr) automatically detects these self-configuring devices. This is done by the cfgmgr querying each slot in turn. It reads the unique identifying code from the ROM chips and compares this against devices in the predefined and customised databases. Any external devices must be powered on for them to be detected by the cfgmgr.

ISA Devices

Some Integrated ISA devices can be self-configuring. This includes keyboard, mouse and audio devices, these can be detected by cfgmgr the same as the PCI devices.

Other devices have to be defined manually. The items that have to be configured include, bus I/O address range, bus memory address ranges, IRQ (interrupt) settings, DMA channels and bus memory DMA address ranges. These are usually configured by switches or jumpers on the cards although some jumper less cards require the System Management Services program to manage the settings.

There are 6 stages to configuring ISA devices

1) Record parameter settings of ISA adapters already configured. This is to prevent values being assigned that are already in use. Use the command `lsdev -Cc adapter -s isa`, then view each individual adapter using the lsaattr command, e.g. `lsattr -l tok0 -E -H`
2) Select parameter values for the new adapters. There will normally be suggested values in the installation guide. The number of values that need defining depends upon the adapter.
3) Install the device driver software, use `smit isa` then “Install ISA Adapter Software”
4) Define the ISA adapter to AIX. From smit isa choose “Add an ISA Adapter”
5) Set the values at the hardware level using the switches or jumpers and after powering off the system install the adapter in a free slot.
6) Use smit isa “Configure a Defined ISA adapter” to then make it available.

Obtaining Hardware Configuration

It is useful to have a copy of the hardware configuration of a system. This can be useful if there are any problems and particularly if AIX needs to be reinstalled.

The following commands can be used to create a list of the hardware details

- `lsdev -CH`
  Provides name, status, location and description of devices
- `lscfg -v`

Lists all configured systems in detail
- `lsattr -E -l sys0`
  Shows detailed information of configured device attributes
- For PCI machines you must manually identify and record the slot and settings of the ISA adapters.

All three commands can be run by creating the following shell script:

```
$ cat devices
for DEV in $(lsdev -CF name)
  do
  echo $(lsdev -Cl $DEV -F "name.location") >> /tmp/devices.log
  lsattr -EHl $DEV >> /tmp/devices.log
  done
lscfg -v >> /tmp/devices.log
```

ISA devices will still need to be documented.

Serial Devices

Whilst the serial ports are configured using the methods earlier there is no way of knowing what serial devices may be attached. Therefore these need to be configured separately. Typically these will be ASCII terminals, Printers and modems.

I am just going to consider terminals for the rest of this section as these are the most common serial devices for UNIX computers. However other devices also need to follow some of the steps in a similar way.

The SMIT fastpath tty allows the management of TTY terminals.

```
                                            TTY

Move cursor to desired item and press Enter.

  List All Defined TTYs
  Add a TTY
  Move a TTY to Another Port
  Change / Show Characteristics of a TTY
  Remove a TTY
  Configure a Defined TTY
  Generate Error Report
  Trace a TTY









F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Adding a Terminal

A terminal is added to the system by defining a TTY logical device. To add a TTY device the mkdev command can be used or the SMIT fastpath mktty.

You must know the following information:

• Port	e.g. s1 / s2
• Adapter	e.g. sa0 / sa1
• Interface type	rs232 / rs432
• Terminal Type (used in TERM variable)	e.g. ibm3151
• Line Characteristics	Speed, parity etc.

When adding a tty device the Enable Login Option will almost certainly need to be changed (except for a dial out line).

The Terminals can be enabled / disabled using the login attribute.

login=disable	(dial out line only)
login=enable	(login prompt on terminal)
login=delay	(user must press key for login prompt)
login=share	(bi-directional)

The following values are needed when configuring a TTY terminal

- Baud Rate (bps) - The speed of the line
- Number of stop bits - Normally 1 however for a poor quality line this can be set to 2
- Parity - What error checking is in place. This is none by default
- Even - Ensures the number of bits transmitted is an even number
  - Odd - Ensures the number of bits transmitted in an odd number
  - Mark - Parity bit is always set to 1
  - Space - Parity bit is always set to 0
  - None - No parity
- Bits per Character - whether 7 or 8 bits are transmitted
- Stop Bits - Number of bits per character to be transmitted to and from the device, between data bytes.
- Operating Mode - How the terminal acts
  - ECHO - The host is responsible for displaying characters on the screen (echoing characters back to the screen)
  - CHAR - The terminal is responsible for displaying characters on the screen when it sends them to the host.
  - BLOCK - Data is sent only to the host in block mode, either through a command initiated at the keyboard or from a command received at the host

The terminal type be default is dumb however you should choose the one that matches your terminal to get maximum functionality. What this does is to setup the terminal to a type specified in the terminfo database. This provides a mapping to the supported functions.

As well as setting the terminal up so that it can communicate with the RS/6000 the TERM variable needs to be set so that full screen applications can be made use of the settings. The terminal variable is one of the standard environment variables. These are held in the directory /usr/share/lib/terminfo/? where ? is replaced with the first letter of the terminal type, e.g. a for AIX, i for IBM, v for VT devices. An example would be for the ibm3151 being file /usr/share/lib/terminfo/i/ibm3151. See later for more details on the terminfo files.

The terminals can often be configured. There are a number of ways of doing this, for example there may be a cartridge that is plugged into the back of the terminal or a certain key combination might be needed such as CTRL-Setup (top right numeric pad key).

The tty settings in AIX can be changed using chdev or through SMIT. However the TTY device must be disabled for it to be changed. This is done using pdisable / penable.

- To enable terminals
  `penable devicename` (one device)
  or
  `penable -a` (all devices)

- To disable terminals
  `pdisable devicename` (one device)

or

```
pdisable -a                    (all devices)
```

(the console cannot be disabled using the pdisable command)

running the penable / pdisable commands without any parameters will display any enabled / disabled devices.

These commands update the /etc/inittab file and then refresh the init process.

If a fault exists whereby the terminal is constantly connecting and disconnecting, the getty program will be killed and restarted a lot of times. It is then possible that a message will appear "tty respawning too rapidly". You should therefore temporarily disable the TTY while carrying out checks and repairing the fault.

Terminal Settings (termcap & terminfo)

For programs that don't run in full screen mode then the type of terminal does not hold a great deal of importance. It is useful to map the appropriate keys to certain actions (e.g. the delete key) however as far as outputting to the screen they just send the lines of text to the screen. The time that the terminal time plays a large amount of signification is when using full screen programs such as vi or SMIT (in character mode also known as smitty). Taking SMIT as an example it wouldn't work if it is unable to interpret the cursor keys correctly, or place the menu items in a consistent position.

This was overcome when vi was written (Bill Joy was the developer responsible). Rather than hardcode all the characteristics of the different terminals in vi he developed a generic terminal handling mechanism. This also had the advantage that if a new terminal came out with new characteristics then the new terminal can be registered with the handling mechanism and programmers would then be able to make use of it. The system that he developed was for Berkeley UNIX and was referred to as termcap (terminal capabilities).

In the true spirit of devolving UNIX [sarcasm intended] a different system was then designed for System V UNIX referred to as terminfo (terminal information). The capabilities of these two methods are similar however terminfo uses a compiled database whereas termcap is a large ASCII database.

The mechanisms work by having a database with the terminal capabilities and a subroutine library that is used to query the capabilities of the terminal type. Both methods use the TERM environment variable.

The terminal type is not held by the shell, instead normally it is worked out at login time using the /etc/profile (or users profile). In a standard profile the following lines hold the terminal settings.

```
TERM_DEFAULT=lft
TERM=`termdef`
TERM=${TERM:-$TERM_DEFAULT}
```

The termdef command looks in the CuAt file in the ODM to see what the terminal is defined as. If the ODM does not have a valid entry (including virtual terminals - e.g. X-Windows and network connections) then it will set the value to dumb. If the terminal is an X-Windows terminal then it will be renegotiated to give a terminal type of aixterm. If it is a network connection then it will be negotiated with the emulator at the client.

The TERM variable can be overwritten in the users .profile (or on the command line) however this should be done with care as it may prevent the terminal from functioning properly. An example of where you may want to do this is when using the telnet client provided in Windows 9x where the terminal can be set to ANSI/VT however will end up with the TERM set to ANSI (which is not supported by SMIT) rather than VT types (which are supported by SMIT).

```
if [ $TERM="ANSI" ]
then TERM=vt100
export TERM
fi
```

As well as terminals having different capabilities they also have different key sent from the keyboard (e.g. delete and page up/down keys) and sent to the screen (e.g. clear the screen etc.).

The definition is stored in the terminfo file for the particular type in the directory /usr/share/lib/terminfo/?. The ? is a directory name based on the first character of the terminal name. For example the IBM terminals (not including the AIX virtual terminals) are in /usr/share/lib/terminfo/i directory.

If you have a terminal that is not directly supported by AIX then one of the sample definition files can be used to create the binary definition file. The samples are in /usr/share/lib/terminfo/*.ti, although sometimes it may be easier to use an emulation feature of the terminal if available. The tic compiler is used to create the binary files from the text files.

The file consists of alias lines so that different names can be used for the same terminal type, capability lines that define what the terminal can do and the codes used to achieve it, finally there are comment lines.

- The alias lines use the vertical bar ‘|’ to separate the multiple entries e.g. ibm3151 | ibm 3151. The lines must end in a comma.
- The capability lines are indented with a tab key. This is not just for tidy formatting as it determines whether the compiler treats the line as a capability line and not an alias line. Multiple items on a line are separated with a tab and the end of a line is signalled by a comma.
- The command lines begin with the hash ‘#’ character.

There are a few different capabilities that are defined.

Boolean Capabilities have a capability name. For example am specifies that the terminal performs automatic right margins. If this is not specified then programs will assume that it is not supported. Some examples are:

- am - The cursor moves to the beginning of the next line when it reaches the right margin
- bel - Produces an audible signal (such as a bell or bleep)
- bw - A backspace from the left edge of the terminal moves the cursor to the last column of the previous row.
- os - When a character is displayed or printed in a position already occupied by another character, the terminal overstrikes the existing character, rather than replacing it with the new character.

Numeric Capabilities have a capability name, a hash sign followed by a number. For example cols#80 says that the terminal has 80 columns. All numeric values are non-negative.

- cols - This specifies the number of columns on each line for the terminal
- lines - The number of lines on a terminal

String Capabilities tell the program how to send a command to the terminal. The capability name is followed by an equals sign and then the command sequence. For example cuul=\EA which specifies Esc-A (move the cursor up a line). For example:

- cub1=\ED - Moves the cursor one space to the left (back-space)
- cud1=\EB - Moves the cursor down a line
- kbs=^H - Back space
- kclr=\EL^M - The clear key
- smul=\E4B - Turns underlining on
- rmul=\E4@ - Turns underlining off
- rmso=\E4@ - Exists standout mode
- sgr0=\E4@\E<@ - Turns off all attributes

The sample below shows some details of the dtterm file.

```
dtterm,
        acsc=``aaffggjjkkllmmnnooppqqrrssttuuvvwwxxyyzz{{||}}~~, am, bel=^G,
        blink=\E[5m, bold=\E[1m,
        box1=\154\161\153\170\152\155\167\165\166\164\156, batt1=f1,
        box2=\154\161\153\170\152\155\167\165\166\164\156, batt2=f1md,
        font0=\E(B, font1=\E(0, civis=\E[?25l, clear=\E[H\E[J, cnorm=\E[?25h,
        colb0=\E[40m, colb1=\E[41m, colb2=\E[42m, colb3=\E[43m, colb4=\E[44m,
        colb5=\E[45m, colb6=\E[46m, colb7=\E[47m, colf0=\E[30m, colf1=\E[31m,
        colf2=\E[32m, colf3=\E[33m, colf4=\E[34m, colf5=\E[35m, colf6=\E[36m,
        colf7=\E[37m, colors#8, cols#80, cr=\r, csr=\E[%i%p1%d;%p2%dr,
        cub=\E[%p1%dD, cub1=\b, cud=\E[%p1%dB, cud1=\n, cuf=\E[%p1%dC,
        cuf1=\E[C, cup=\E[%i%p1%d;%p2%dH, cuu=\E[%p1%dA, cuu1=\E[A,
        dch=\E[%p1%dP, dch1=\E[P, dim=\E[2m, dl=\E[%p1%dM, dl1=\E[M,
        ech=\E[%p1%dX, ed=\E[J, el=\E[K, el1=\E[1K, flash=\E[?5h$<200>\E[?5l,
        home=\E[H, ht=\t, hts=\EH, ich=\E[%p1%d@, il=\E[%p1%dL, il1=\E[L,
        ind=\ED, invis=\E[8m, is2=\E\sF\E>\E[?1l\E[?7h\E[?45l, it#8, kbs=\b,
```

Similar Terminal Types can be defined as being similar to each other. Then the use string can be used to override the settings of the other terminal. Capabilities can be cancelled by placing a commercial-at sign '@' immediately after the capability code name (this is not allowed in a normal terminal definition). This is normally the way that a terminal is defined as different terminals tend to have more similarities than differences.

The terminfo commands can be listed using the infocmp command.

List the details of the lft terminal

```
infocmp -l lft
```

List the common capabilities between ibm3151 and the wyse-60 terminals

```
infocmp -c ibm3151 wyse-60
```

Screen Control (tput)

Using the tput command can control the screen using the characteristics in the terminfo file. The command is an operating system command and is not tied to any particular shell.

Examples

```
tput clear
bold=$(tput smso)
norm=$(tupt rmso)
print “Please ${bold}READ THIS${norm} message”
```

The output would be:

```
Please READ THIS message
```

```
integer num=0
while (( (num=num+1) < = 4)); do print “$(tput cuu1)$num loops”;done
1 loops
```

counter (first prints 1 then over types with 2, 3 etc.

Input / Output Maps

Input / Output maps are normally used to extend the ASCII characters however it is also useful if an NLS cartridge is unavailable.

The Input Maps: Map a keyboard ASCII value to another. This displays a new value on the screen. This could be used to remap the dollar symbol ‘$’ to the pound symbol ‘£’.

The Output maps map application-generated code to another value before terminal display. For example if there is not a pound symbol ‘£’ in the normal display character set then the output map could translate the character to a sequence to enter extended character set, print a character, and then revert to the normal font.

The map files are kept in /usr/lib/nls/termmap

The map files represent ASCII characters by hex values preceded by \x and octal values preceded by \o.

Map files are specified in the terminal attributes using either mkdev or chdev. An input map file must end with the characters .in and output map files with .out.

The setmaps command is an interactive command which will assign a terminal map to the standard input device for the current session. With no flags this displays the names of the current maps.

Setting I/O Options for a Terminal (stty)

The stty command can set or display I/O options for the terminal.

List settings (other than defaults)

```
stty
```

List all settings

```
stty -a
```

Set options for the current session

```
stty options
```

Example options

Intr ^?	Set interrupt key to Ctrl-?
-Ixon	Disables start/sop control
-echo </dev/tty1	Host will not echo any typed characters on /dev/tty1
-a </dev/tty1	View settings for tty1 (not stdout)
ctrl - j stty sname ctrl-j	Reset parameters to reasonable values (use for hung terminal)
stty same </dev/ttyn	Resets parameter on /dev/ttyn

```
$ stty -a
speed 9600 baud; 24 rows; 80 columns;
eucw 1:1:0:0, scrw 1:1:0:0:
intr = ^C; quit = ^\; erase = ^H; kill = ^U; eof = ^D; eol = ^@
eol2 = ^@; start = ^Q; stop = ^S; susp = ^Z; dsusp = ^Y; reprint = ^R
discard = ^O; werase = ^W; lnext = ^V
-parenb -parodd cs8 -cstopb hupcl cread -clocal -parext
-ignbrk brkint ignpar -parmrk -inpck -istrip -inlcr -igncr icrnl -iuclc
-ixon -ixany ixoff imaxbel
isig icanon -xcase echo echoe echok -echonl -noflsh
-tostop echoctl -echoprt echoke -flusho -pending iexten
opost -olcuc onlcr -ocrnl -onocr -onlret -ofill -ofdel
```

TTY Problems

There are a number of things that can be done to check why a terminal is not working as it should.

The first thing to check is if the terminal is in contact with the server. After checking brightness, contrast etc. Running the following command should confirm whether or not they are in contact.

```
echo hello > /dev/ttyn
```

This should output hello onto the screen

If this fails then the power, cabling etc should be checked.

Then check for processes running on the system try:

```
fuser -u /dev/ttyn
```

Try
<CTRL - q>   - Release the screen
<CTRL - c>   - kill current process
Power the terminal off and then on
Check the terminal setup
Is there a getty process running on the device?
Try pdisable then penable
If a key does not work then it may need to be re-mapped. Use the stty command to do this:

```
stty keyname = (press the key)
```

An example keyname is “erase” for the backspace key.
(see earlier for more details on stty)
If full screen programs do not run then you should check the TERM setting is correct.

If you have a large number of terminals connected through concentrators then a plan should be made mapping the terminals to their port numbers.

Keyboard Mapping for Xstation

The keyboard map for an Xstation is controlled by the xmodemap command. This is in /usr/bin/X11 and is used to edit the keyboard modifier map and keymap table used by client applications. It is usually included in the session startup script.

When an Xstation boots, the xmodmap is called from /usr/lpp/x_st_mgr/bin/x_st_mgrd.cf. By default the map file “keyboard” is loaded.

The default maps are stored in /usr/lpp/X11/default/xmodmap/$LANG.

The map can be changed using SMIT or by editing the file /usr/lpp/x_st_mgr/bin/x_st_mgrd.cf which is a script file.

Keyboard Mapping for LFT

The default map for LFT can be changed using the chkbd command. This can be accessed from SMIT using the fastpath smit chkbd or the full command

chkbd /usr/lib/nls/loc/en_GB.ISO8859-1.lftkeymap.

This will not take effect until the next reboot.

Basic Shell Script Programming

In it’s most basic form a script file could be a list of commands to execute (similar to a DOS bat file). However script programming opens up a lot more flexibility which if well applied can reach the functionality of a programming language. The real name for these are shell scripts as they are dependant upon the shell they are running in, however for briefness I will just refer to them as scripts.

Here are some of the basic instructions that can be used to alter how a script might run.

Here’s a first script file.

```
echo "The 10 largest files in directory" `pwd` "are:" >$HOME/largefiles
ls -l | cut -c 32-41,55- | sort -nr | head >> $HOME/largefiles
```

This file will output to the file $HOME/largefiles a title followed by the 10 largest files in the current directory.

The script can be made more useful by allowing it to accept the directory name on the command line. This is done by using the variables $1, $2 etc. to represent argument1, argument2 etc.

```
tempdir=`pwd`
cd $1
echo "the 10 largest files in directory" `pwd` "are:" >$HOME/largefiles
ls -l | cut -c 32-41,55- | sort -nr | head >> $HOME/largefiles
cd $tempdir
```

This script will now.

Store the current directory
Change to the directory given as the first argument (if no directory is given will change to the default home directory)
Put the Title in the file
List the 10 largest files in the current directory

Return to the stored directory

Script Portability

As mentioned earlier there are a number of differences between how the different shells work. This is even more important when we start writing script files. If you will only ever run the scripts yourself in the same shell this is not an issue however if others will run the script, they could run it from a different shell with unpredictable results.

To overcome this the first method is to tell the script to run under a certain shell by including the following command at the start of the file.

```
#!/bin/ksh
```
for korn shell

or

```
#!/bin/sh
```
for bourne shell

You may be wondering why you would want to use the bourne shell when the korn shell provides more functionality. One reason may be for portability to a system that doesn’t have the korn shell. If you script will always be used on a system with the korn shell then this is not an issue however some people prefer to create bourne shell scripts for maximum portability.

There are also a number of different script programming languages that could be used as part of this first line. Once could be expect which can be very powerful for creating scripts.

Shifting Arguments (shift)

The shift command allows you to shift all the arguments one to the left.

The value in $2 is put into $1
The value in $3 is put into $2
etc.

This is particularly useful when programming for the bourne shell as it allows more than 9 arguments to be passed, whereas the korn shell will accept any number of arguments directly.

System Storage

Traditional UNIX disk management was handled by creating a number of partitions for each of the files system. This is still the way that Linux handles storage.

There are a number of disadvantages to this method:

It is a large task to change the size of the partition. To change the size of an existing partitions requires the backing up of data, destroying and recreating any partitions that need changing and then restoring the data.

Another restriction is that the partitions have to be contiguous disk space which limits the partition to existing on a single physical drive and prevents the spanning of multiple physical volumes.

IBM however developed an alternative system based around the “Logical Volume Manager” concept. This is used by AIX to manage disk space between the different filesystems.

The files and directories are all held within a File System which is stored in logical storage, mapped across to physical storage on the disks. The logical volume manager handles all this and provides a solution to the old restrictions. New hard disks can now be added and partitions resized dynamically.

Components of AIX System Storage

Volume Group (VG)

This is the largest form of storage. A volume group consists of one or more physical disks. The volume group could be disconnected from one system and then directly connected to another.

The volume group is managed as a collection of physical partitions (PP). The physical partitions are the same size across all the disks contained within the Volume Group.

There is one volume group that is used by all AIX systems, this is called rootvg and contains the BOS (Base Operating System) and all the system files required by AIX. Typically another volume group may be added for a database storage area to put this on a separate group of disks from the operating system.

Physical Volume (PV)

A Physical Volume is used to distinguish an individual disk in the system. The Physical Volume can be internal or external. Each physical volume must be attached to a volume group before it can be used.

The Physical Volumes have names hdisk*n* and is held in the /dev directory.

Physical Partition (PP)

Physical Partitions are divisions of the Physical Volume.

All physical partitions within a volume group have to be the same size. The default size of the physical partition is 4MB.

Managing Volume Groups

The root volume group rootvg is created by the installation. This is why AIX operating system files are held. Additional disks can be added to rootvg.

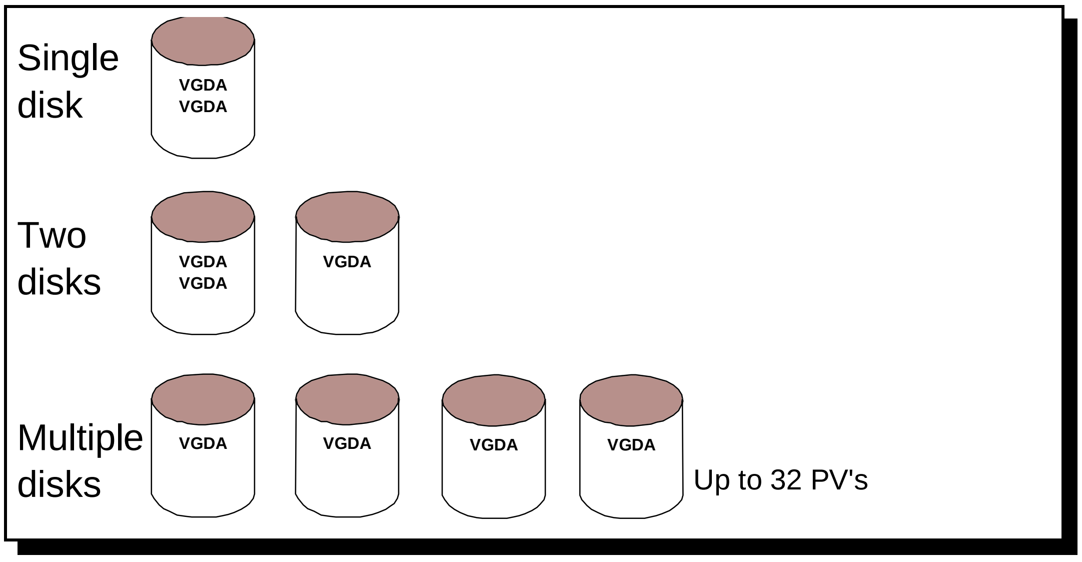
It is sometimes better to create a separate volume group for new disks. It is a good idea for example if you have external disks that they are not part of the rootvg volume group. That way if an external disk is not available it will not prevent the operating system from starting.

It is also a good idea to separate user data from the operating system files. The reduces the risk of user files being damaged during system updates and makes management easier.

Volume Group Descriptor Area (VGDA)

The VGDA is an area of disk containing information for the entire VG. There is at least one VGDA per disk, on systems with one or two PV’s there may be two on a single disk.

The VGDA is needed to ensure data integrity and management data. Before a VG can be activated there must be a quorum of VGDA’s available. This is equal to at least 51% of the total VGDAs for that VG.



VGDA’s on single / multiple disk systems

Logical Volume (LV)

As mentioned earlier the file systems are stored in a Logical Volume. This is managed using the Logical Volume Manager (LVM).

The logical volume was split into logical partitions (LP), each logical partition maps to a physical partition. Each logical volume consists of at least one logical partition and is tied to a specific volume group. The Logical Partitions can be on any disks within the volume group, they can be spread across the different disks and do not need to be contiguous.
Logical Volumes can be increased at any time by assigning available physical partitions to logical partitions within the logical volume. This can be done while the logical volume is in use. Logical Volumes cannot however be decreased without deleting the logical volume and recreating it with less logical partitions, obviously this requires a backup and restore.

There are limits to the number of logical partitions and the number of logical volumes, these are listed in the table below.

Volume Group	Max. 255 per system
Physical Volume	Max. 32 per volume group
Physical Partition	Max. 1016 per physical volume
Size Physical Partition	Max. size 256MB
Logical Volume	Max. 256 per volume group
Logical Partition	Max. 32,512 per logical volume

Table of Maximum Storage Values

The Logical Volume Manager (LVM) makes the allocation of logical volumes invisible to the applications and casual users. It achieves this using a device driver (LVDD) which runs above the traditional UNIX device drivers.

Logical volumes can contain a number of different types of data. The different data types are:

- Journaled File System (i.e. normal files and directories e.g. /dev/hd4)
- Paging space (/dev/hd6 - virtual memory)
- Journal Log (/dev/hd8 - required by Journaled File System)
- Boot Logical Volume (/dev/hd5)
- Raw Data (e.g. some database data areas)

File Systems

A file system is a method of storing data. It allows data to be stored in files and directories. There are 3 files systems supported by AIX

- Journaled File System (jfs)
- CD-ROM File System (cdrfs)
- Network File System (nfs)

The Journaled File system is the normal file system and exists within a logical volume on a disk. CD-ROM File System is how data is stored on CD-ROMs. NFS File System is a way of accessing data over a network.

The NFS file system is explained in more details in the Networking Section.

The use of file systems allows certain data to be positioned differently on the disks for maximum performance. It allows the imposing of disk quotas to limit disk usage by users.

The file system are kept separate so that if one becomes corrupt the others are not affected.

File Systems installed as standard

There are a number of file systems installed as standard as part of the AIX install for a stand alone system. These can be displayed using the df command.

```
$ df
Filesystem    512-blocks      Free %Used    Iused %Iused Mounted on
/dev/hd4          204800    172920   16%     1166     3% /
/dev/hd2         3366912    457096   87%    36277     9% /usr
/dev/hd9var        24576     20928   15%      498    17% /var
/dev/hd3          131072    125040    5%       58     1% /tmp
/dev/hd1          204800      5504   98%     1872     8% /home
```

/dev/hd4 = / (root)
This is at the top of the hierarchical file tree. This holds all the files and directories critical for system operations including the device directory (/dev) the /bin and /sbin directories, and the configuration files (/etc).

/dev/hd2 = /usr
This is the applications file system. It holds operating system commands, libraries and application programs. It can be shared

/dev/hd9var = /var
Variable files. This holds files that change a lot during system usage. This includes log and spool files.

/dev/hd1 = /home
This is where users home directories are. On older versions of AIX this was /u.

/dev/hd3 = /tmp
This is space accessible to all users for temporary files and work space. It should be cleaned out on a fairly regular basis.

The layout characteristics and attributes for the file systems is kept in /etc/filesystems.
The first few lines of the file is shown below.

```
/:
        dev             = /dev/hd4
        vfs             = jfs
        log             = /dev/hd8
        mount           = automatic
        check           = false
        type            = bootfs
        vol             = root
        free            = true

/home:
        dev             = /dev/hd1
        vfs             = jfs
        log             = /dev/hd8
        mount           = true
        check           = true
        vol             = /home
        free            = false
```

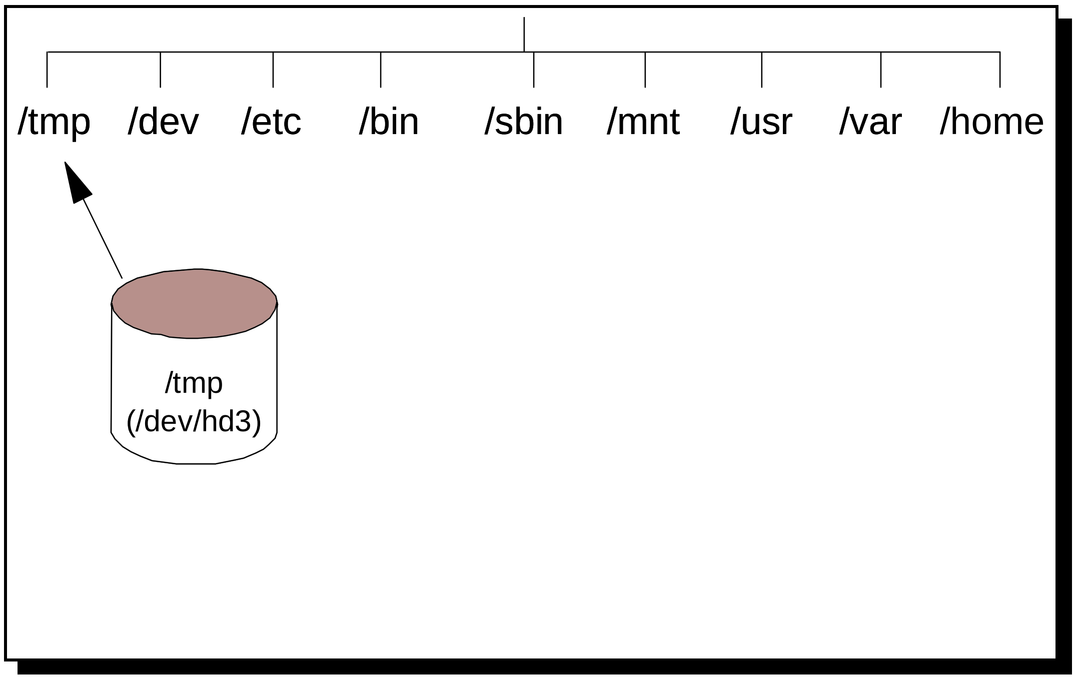
the attributes are:

dev	The device file in the /dev directory
vfs	Type of mount (i.e. type of file system
log	The device where the log file is written (jfs only)
mount	Whether the device should be mounted by default (automatic / true / false)
check	Should the file system be checked. (true / false)
vol	The label of the filesystem

Mounting File Systems

Earlier in the Files and Directories section I explained how the hierarchical directory structure all starts from a single point, however now I have said that by default these are split across 5 different logical volumes. Obviously there needs to be some mechanism for combining the separate Logical Volumes into the standard directory structure. This is done by mounting.

What mounting does is to take a file system and link it into the existing file structure.



File system being mounted

The above diagram shows the main directory tree. The following directories are all mounting points and are empty prior to mounting the other logical volumes:

/tmp /mnt /usr /var /home

When the logical volume is mounted it appears as though it was just another directory in the directory tree. Changing from the root directory '/' to the '/tmp' directory will change the disk that is being accessed, but only if it is already mounted.

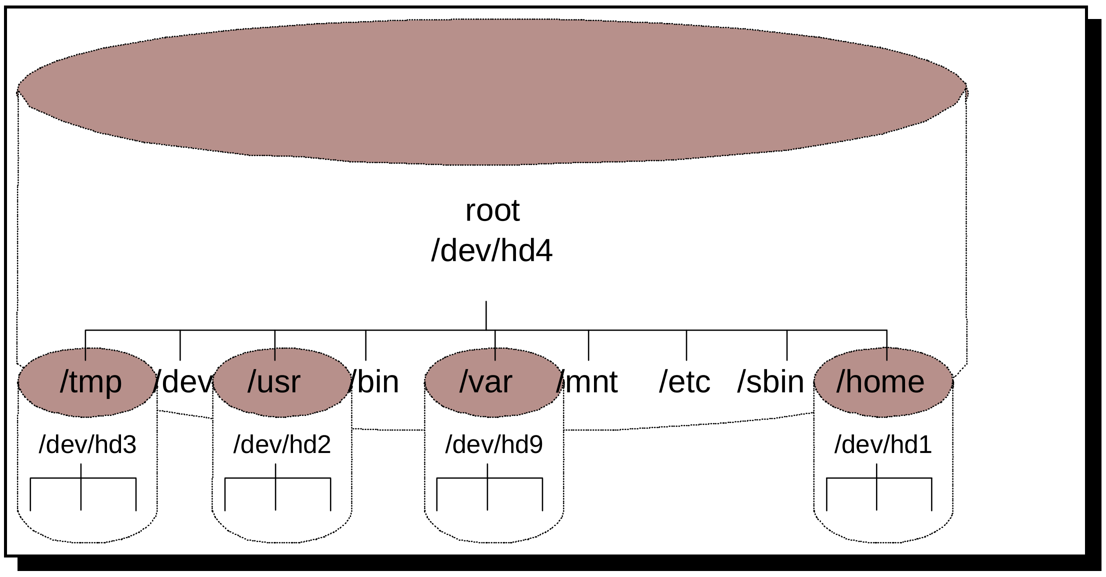


Diagram of mounted file systems

The above diagram shows the Logical Volumes mounted into the directory structure.

The command that is used to mount the file system is called "mount" to unmount a file system the command "unmount" is used. Normally the command is run automatically during system startup however it is sometimes necessary to manually mount a file system, particularly removable file systems such as floppy disks and CD-ROM's.

```
mount  /dev/lv00  /home/stewart
```

The above command would be used if there was a logical volume dedicated for use as my home directory, with a logical volume name of lv00.

The mount directory in the above example is /home/stewart which must exist before the logical volume can be mounted. Anything that exists in the mount directory or below will be hidden when the filesystem is mounted. Nothing is actually deleted through the mount command however files that are hidden would look as though they'd been deleted until the file system was unmounted.

To mount a file system you need sufficient authority. This is dependant upon:

1. Whether the mount is to the default mount point specified in /etc/filesystems
2. Whether or not the user is a part of the system group
3. Whether they have write permission to the mount point.

Obviously root can mount a filesystem regardless of any of the above.

Viewing File Systems

There are three commands that allow you to view the file systems. We have seen one earlier which is the display file system command (df). This is for commonality with other UNIX systems. The AIX command to view file systems is lsfs. The details are in /etc/filesystems however using the lsfs command you can see CD-ROM file systems and nfs file systems.

The lsfs command will normally give a line by line display however it can also create the output in stanza format.

```
$ lsfs
Name            Nodename   Mount Pt               VFS   Size    Options    Auto
Accounting
/dev/hd4        --         /                      jfs   204800  --         yes
no
/dev/hd1        --         /home                  jfs   204800  --         yes
no
/dev/hd2        --         /usr                   jfs   3366912 --         yes
no
/dev/hd9var     --         /var                   jfs   24576   --         yes
no
/dev/hd3        --         /tmp                   jfs   131072  --         yes
no
```

The other command is to view logical volumes by volume group. This is lsvg.

```
$ lsvg
rootvg
$ lsvg rootvg
VOLUME GROUP:   rootvg                   VG IDENTIFIER:   00538690aa0855bf
VG STATE:       active                   PP SIZE:        4 megabyte(s)
VG PERMISSION:  read/write               TOTAL PPs:       515 (2060 megabytes)
MAX LVs:        256                      FREE PPs:        0 (0 megabytes)
LVs:            8                        USED PPs:        515 (2060 megabytes)
OPEN LVs:       7                        QUORUM:         1
TOTAL PVs:      1                        VG DESCRIPTORS: 2
STALE PVs:      0                        STALE PPs:       0
ACTIVE PVs:     1                        AUTO ON:        no
MAX PPs per PV: 1016                     MAX PVs:         32
```

Structure of a Journaled File System

Journaled File systems must exist within a logical volume. All files in the File System are allocated in blocks of 4096 bytes in size (this may be different if fragmentation is implemented or with very large files).

The first addressable logical block is the “superblock”. This contains the information about the file system. It includes information such as the file system name, size, number of inodes etc. There is a backup copy of the superblock in case of corruption. After the superblock are the ‘inodes’ which contain information for the files, such as: file type, size, permissions etc. They also contain pointers to the data block for fragment addresses which hold the data. For larger files there are also ‘indirect blocks’ filled with data block addresses to point at the data block or fragments.

There is an inode for each file. The inode contents are as follows:

- permissions
- number of links
- type of file
- userid
- groupid
- file size
- address of blocks
- time modified
- time accessed
- time changed
- access control information
- reserved other

Some of the inode details can be viewed with ls.

```
$ ls -l
total 8320
-rw-r--r--   1 stewart  staff     425984 02  Feb 09:42 old1.zip
-rw-r--r--   1 stewart  staff     425984 02  Feb 09:42 old2.zip
-rw-r--r--   1 stewart  staff     425984 02  Feb 09:43 old3.zip
-rw-r--r--   1 stewart  staff     425984 02  Feb 09:43 old4444p
-rw-r--r--   1 stewart  staff     425984 02  Feb 09:43 old5.zip
```

Journaled Log

When a write is done to a file it is first written to memory and then stored to disk when the sync command runs every minute. There is therefore a risk that in the event of a system crash the file system integrity could be compromised. To overcome this each journaled file system has a journaled log. This jfslog (/dev/hd8) is circular to allow it to wrap around. The log is the size of one physical partition per volume group. Whenever a write is made to a file metadata is written to the jfslog with details of changes to the structure itself such as inodes and the free list.

Fragmentation

Without fragmentation all files are made up of 4096 byte blocks. This creates wasted space as any space left in the 4k block will be wasted. Fragmentation allows fragments of the 4k logical blocks to be assigned to files and directories. This saves space when there are a lot of small files and directories. Fragment support allows the last direct block of small user files and directories (and long symbolic links) to be a smaller block size. Fragment size has to be specified when a file system is created. The allowable sizes are 512, 1024, 2048 and 4096 bytes. The default is 4096 bytes.

Different file systems can have a different fragmentation values.

Variable Inodes

When a file system is created Inodes are written to disk. Each file and directory uses an inode to describe information about the file. A number of inodes are also reserved by AIX for files and directories in each file system that is created.

Fragment support now means that it is possible to have more files than inodes were available. Therefore JFS allows the number of inodes created to be specified in case more or less than the default are required. The number can be specified at the time of creation this is defined as the Number of Bytes per Inode (NBPI).

Compressed File Systems

AIX supports data compression at the File Systems level. This allows approximately twice as much data to be stored on the disk (the actual value is dependant on the types of data stored). The use of data compression however can put a large demand on the system and have a severe performance impact.

It also increases the rate of fragmentation of the disk’s free space. The defragfs utility can defragment a file system (this can be used against compressed and uncompressed file systems).

Large File Enabled File System

Any file systems over 2GB in size have to be defined as Large File Enabled File Systems. The data stored before the 4MB file offset is in 4096 byte blocks, data beyond the 4MB file offset is allocated in 128K blocks.

The Logical Volume Manager (LVM)

The Logical Volume Manager is used to manage the system’s storage. It is accessed as a panel through SMIT with the fast path lvm

```
                             Logical Volume Manager

Move cursor to desired item and press Enter.

  Volume Groups
  Logical Volumes
  Physical Volumes
  Paging Space













F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

There are four options from the LVM screen in SMIT. These allow the management of Volume Groups, Logical Volumes, Physical Volumes and Paging Space. Whilst many of these can be altered by using other commands in SMIT the LVM allows low level changes to be made to Logical Volumes which is needed if there is not a File System in the Logical Volume.

Many of the commands run from the menus in smit can also be run from the command line.

Volume Groups

```
$ lsvg rootvg
VOLUME GROUP:   rootvg                   VG IDENTIFIER:  00538690aa0855bf
VG STATE:       active                   PP SIZE:        4 megabyte(s)
VG PERMISSION:  read/write               TOTAL PPs:      515 (2060 megabytes)
MAX LVs:        256                      FREE PPs:       0 (0 megabytes)
LVs:            8                        USED PPs:       515 (2060 megabytes)
OPEN LVs:       7                        QUORUM:         1
TOTAL PVs:      1                        VG DESCRIPTORS: 2
STALE PVs:      0                        STALE PPs:      0
ACTIVE PVs:     1                        AUTO ON:        no
MAX PPs per PV: 1016                     MAX PVs:        32
```

To show the Physical Volumes in Volume Group

```
$ lsvg -p rootvg
rootvg:
PV_NAME           PV STATE    TOTAL PPs   FREE PPs    FREE DISTRIBUTION
hdisk0            active      515         0           00..00..00..00..00
```

To show the Logical Volumes in Volume Group

```
$ lsvg -l rootvg
rootvg:
LV NAME             TYPE       LPs    PPs    PVs  LV STATE      MOUNT POINT
hd5                 boot       2      2      1    closed/syncd  N/A
hd6                 paging     32     32     1    open/syncd    N/A
hd8                 jfslog     1      1      1    open/syncd    N/A
hd4                 jfs        25     25     1    open/syncd    /
hd2                 jfs        411    411    1    open/syncd    /usr
hd9var              jfs        3      3      1    open/syncd    /var
hd3                 jfs        16     16     1    open/syncd    /tmp
hd1                 jfs        25     25     1    open/syncd    /home
```

To add or remove a Volume Group

```
To add or remove a Volume Group

                               Add a Volume Group

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                        [Entry Fields]
  VOLUME GROUP name                                  []
  Physical partition SIZE in megabytes                4                        +
* PHYSICAL VOLUME names                              []                        +
  Activate volume group AUTOMATICALLY                 yes                      +
    at system restart?
  Volume Group MAJOR NUMBER                          []                        +#
  Create VG Concurrent Capable?                       no                       +
  Auto-varyon in Concurrent Mode?                     no                       +







F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

The volume group can be managed using smit vgsc

```
                     Set Characteristics of a Volume Group

Move cursor to desired item and press Enter.

  Change a Volume Group
  Add a Physical Volume to a Volume Group
  Remove a Physical Volume from a Volume Group
  Reorganize a Volume Group














F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

The following commands can also be used to manipulate the Volume Groups.

Add new Physical Volume to a Volume Group

```
extendvg -f Volumegroup hdiskn
```

Delete Physical Volume from a Volume Group

```
reducevg -d
```

It is possible to reorganise a Volume Group. This redistributes the physical partitions used by the logical volumes according to their preferred allocation policies. See later for further details.

```
reorgvg rootvg hd2 hd1
```

The logical volumes are given priority based on the order they are provided on the command line. The reorgvg command can take a while to run depending upon the system. Whilst the command is running the Volume Group will be locked and certain commands cannot be run (including most of the LVM commands).

The Volume groups can be activated and deactivated using the following commands, note these should be NOT be used on rootvg.

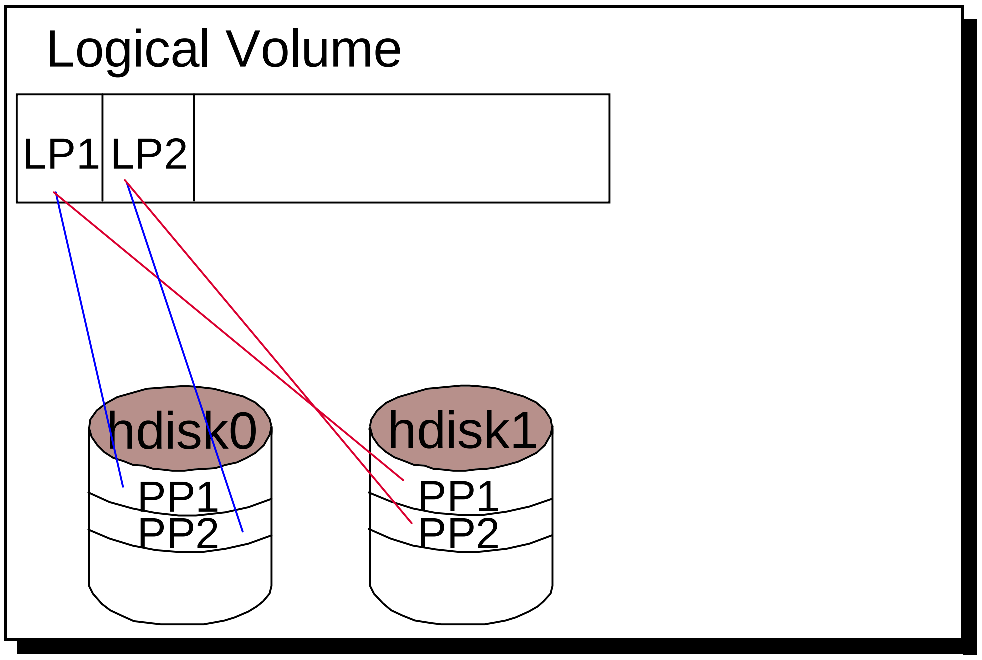
```
varyonvg volumegroup
```

```
varyoffvg volumegroup
```

External Volume groups can be imported and exported so that they can be moved between AIX systems. The commands which can be accessed as SMIT fastpaths are importvg and exportvg

Mirroring

Normally Logical Volumes are setup so that a single Logical Volume is held within a single Physical Partition. However to maintain data integrity in the event of a disk failure Mirroring can be used which creates a “mirror” copy of the data on another disk.



Mirrored Logical Volume across two Physical Volumes

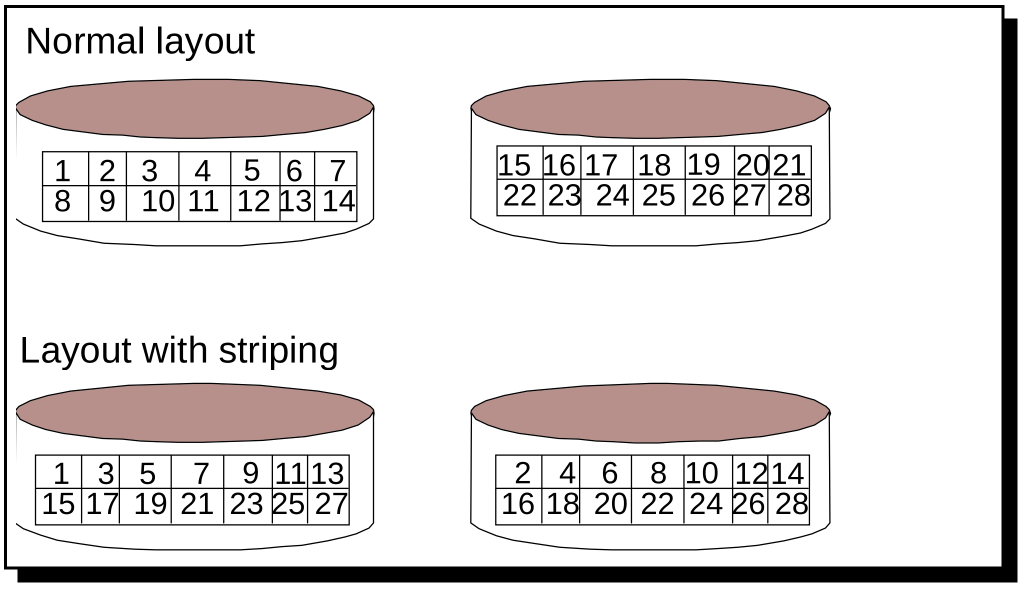
Normally each mirror should be kept on a separate drive, however this does not have to be the case.

The biggest drawback from mirroring is a hit on performance. However you can configure parallel writing to minimise this, or sequential writing for maximum reliability.

Parallel - Here all write requests are sent to both disks simultaneously. The write is complete once both disks have been written to. There is however a risk that data integrity could be lost if there was a disk failure during the write. Write consistency can be turned on to overcome this problem. When turning on mirroring on an existing LV the copies have to be synchronised. This is done by using the -k option in mklvcopy when mirroring is turned on or using syncvg at a later date. Until Synchronisation the new copy is marked stale.
Sequential - Here when data is written it is written to each physical partition in turn. Control is not returned until all logical volumes have been written. This is slower due to the writes being sequential.

Striping

For high performance an option available in AIX is striping. What this does is to spread the data across multiple disks. When a large sequential file is accessed the data is read from the different disks



LV layout with striping

The above diagram shows how striping is implemented. Instead of sequentially writing the LV's onto the disk they are alternated between the available disks. The first chunk of on the first disk, the second on the second disk etc. until it returns to the first disk.

These "chunks" are not directly related to PV's instead they are created using the size of data block specified at creation time. This is specified as a power of 2 in the range 4K to 128K bytes.

A striped logical volume cannot be mirrored. The number of physical partitions allocated to a striped logical volume must be evenly distributed across the disks. The disks (min 2) should be spread across as many adapters as possible to maximise throughput. They should be on a volume group dedicated to striped volumes.

Managing Physical Volumes

A physical volume is the AIX reference for a disk. They are split into Physical Partitions (PP's) which are a fixed size across an entire Volume Group. The Physical Volumes can be manipulated using the fastpath smit pv.

```
                                Physical Volumes

Move cursor to desired item and press Enter.

  List All Physical Volumes in System
  Add a Disk
  Change Characteristics of a Physical Volume
  List Contents of a Physical Volume
  Move Contents of a Physical Volume










F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

The physical volumes can be viewed using lspv

```
$ lspv
hdisk0         005386901d47dbad    rootvg
$
$ lspv hdisk0
PHYSICAL VOLUME:    hdisk0                   VOLUME GROUP:     rootvg
PV IDENTIFIER:      005386901d47dbad         VG IDENTIFIER     00538690aa0855bf
PV STATE:           active
STALE PARTITIONS:   0                        ALLOCATABLE:      yes
PP SIZE:            4 megabyte(s)            LOGICAL VOLUMES:  8
TOTAL PPs:          515 (2060 megabytes)     VG DESCRIPTORS:   2
FREE PPs:           0 (0 megabytes)
USED PPs:           515 (2060 megabytes)
FREE DISTRIBUTION:  00..00..00..00..00
USED DISTRIBUTION:  103..103..103..103..103
```

The contents (logical volumes) of a physical volume can be viewed using lspv with the -l option.

```
$ lspv -l hdisk0
hdisk0:
LV NAME              LPs    PPs    DISTRIBUTION          MOUNT POINT
hd5                  2      2      02..00..00..00..00    N/A
hd2                  411    411    54..71..94..103..89   /usr
hd4                  25     25     23..00..02..00..00    /
hd1                  25     25     24..00..01..00..00    /home
hd6                  32     32     00..32..00..00..00    N/A
hd8                  1      1      00..00..01..00..00    N/A
hd9var               3      3      00..00..01..00..02    /var
hd3                  16     16     00..00..04..00..12    /tmp
```

The Physical Partition map can be viewed using lspv -p

```
$ lspv -p hdisk0
hdisk0:
PP RANGE  STATE   REGION        LV NAME             TYPE       MOUNT POINT
  1-2     used    outer edge    hd5                 boot       N/A
  3-36    used    outer edge    hd2                 jfs        /usr
 37-59    used    outer edge    hd4                 jfs        /
 60-83    used    outer edge    hd1                 jfs        /home
 84-103   used    outer edge    hd2                 jfs        /usr
104-135   used    outer middle  hd6                 paging     N/A
136-206   used    outer middle  hd2                 jfs        /usr
207-207   used    center        hd8                 jfslog     N/A
208-208   used    center        hd4                 jfs        /
209-217   used    center        hd2                 jfs        /usr
218-218   used    center        hd9var              jfs        /var
219-221   used    center        hd3                 jfs        /tmp
222-222   used    center        hd1                 jfs        /home
223-286   used    center        hd2                 jfs        /usr
287-287   used    center        hd3                 jfs        /tmp
288-307   used    center        hd2                 jfs        /usr
308-308   used    center        hd4                 jfs        /
309-309   used    center        hd2                 jfs        /usr
310-412   used    inner middle  hd2                 jfs        /usr
413-501   used    inner edge    hd2                 jfs        /usr
502-513   used    inner edge    hd3                 jfs        /tmp
514-515   used    inner edge    hd9var              jfs        /var
```

Adding Physical Volumes

To add a Physical Volume, it is first connected to the machine. If the system is then restarted (essential for an internal disk) it will be configured by configuration manager. Otherwise the PV can be added using Add a Disk in SMIT.

Once a PV is defined to the system it needs to be added to a Volume Group (existing or new) before it can be used.

Moving the contents from one PV to another

To move the contents of (PP's) from one PV to another the migratepv command can be used. This is useful if the physical volume is to be taken out of service. The command cannot be used against striped logical volumes.

Managing Logical Volumes

Logical Volumes as discussed earlier are a representation of a disk to the operating system. It is used as a method of providing additional features not normally available in the classic UNIX systems. The Logical Volumes part can be accessed by smit lv.

```
                              Logical Volumes

Move cursor to desired item and press Enter.

  List All Logical Volumes by Volume Group
  Add a Logical Volume
  Set Characteristic of a Logical Volume
  Show Characteristics of a Logical Volume
  Remove a Logical Volume
  Copy a Logical Volume













F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

The logical volumes can be viewed using lsvg with the -l option to show the logical volumes in a volume group or using lslv.

```
$ lsvg -l rootvg
rootvg:
LV NAME             TYPE       LPs    PPs    PVs  LV STATE      MOUNT POINT
hd5                 boot       2      2      1    closed/syncd  N/A
hd6                 paging     32     32     1    open/syncd    N/A
hd8                 jfslog     1      1      1    open/syncd    N/A
hd4                 jfs        25     25     1    open/syncd    /
hd2                 jfs        411    411    1    open/syncd    /usr
hd9var              jfs        3      3      1    open/syncd    /var
hd3                 jfs        16     16     1    open/syncd    /tmp
hd1                 jfs        25     25     1    open/syncd    /home
```

```
$ lslv hd1
LOGICAL VOLUME:     hd1                     VOLUME GROUP:   rootvg
LV IDENTIFIER:      00538690aa0855bf.8      PERMISSION:     read/write
VG STATE:           active/complete         LV STATE:       opened/syncd
TYPE:               jfs                     WRITE VERIFY:   off
MAX LPs:            512                     PP SIZE:        4 megabyte(s)
COPIES:             1                       SCHED POLICY:   parallel
LPs:                25                      PPs:            25
STALE PPs:          0                       BB POLICY:      relocatable
INTER-POLICY:       minimum                 RELOCATABLE:    yes
INTRA-POLICY:       center                  UPPER BOUND:    32
MOUNT POINT:        /home                   LABEL:          /home
MIRROR WRITE CONSISTENCY: on
EACH LP COPY ON A SEPARATE PV ?: yes
```

It is also possible to see how the Logical Volume is spread across the physical partitions and how it is positioned on the disk which is useful to determine if the intra-physical allocation policy has been affective (see Policies section).

```
$ lslv -l hd1
hd1:/home
PV                COPIES         IN BAND       DISTRIBUTION
hdisk0            025:000:000    4%            024:000:001:000:000
```

The copies section is in three parts. The first part represents the number of physical partitions used on that disk. The Second and Third parts are 000 unless there is mirroring and the mirrored copies are not all on separate PV’s.

The Distribution field shows how the PP’s are spread across the disk. This is in the format

outer edge **:** outer middle **:** Centre **:** inner middle **:** inner edge

The IN BAND field says how many of these are in the region specified by the intra-allocation policy.

```
$ lslv -m hd1
hd1:/home
LP    PP1  PV1               PP2  PV2               PP3  PV3
0001  0222 hdisk0
0002  0082 hdisk0
0003  0083 hdisk0
0004  0060 hdisk0
0005  0061 hdisk0
0006  0062 hdisk0
0007  0063 hdisk0
0008  0064 hdisk0
0009  0065 hdisk0
0010  0066 hdisk0
0011  0067 hdisk0
0012  0068 hdisk0
0013  0069 hdisk0
0014  0070 hdisk0
0015  0071 hdisk0
0016  0072 hdisk0
0017  0073 hdisk0
0018  0074 hdisk0
0019  0075 hdisk0
```

This shows the mapping between Logical Partition and Physical Partitions for the Physical Volumes.

New Logical Volumes can be created using smit mklv
Deleted using smit rmlv (note that any high-level filesystems such as jfs filesystems and paging space should be deleted separately otherwise the entries would still be left behind, pointing at a Logical Volume that no longer exists).

smit lvsc allows the characteristics of the Logical Volume to be changed:

```
                     Set Characteristic of a Logical Volume

Move cursor to desired item and press Enter.

  Change a Logical Volume
  Rename a Logical Volume
  Increase the Size of a Logical Volume
  Add a Copy to a Logical Volume
  Remove a Copy from a Logical Volume















F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

The following commands could be used instead of SMIT.

chlv - Change that characteristics of the Logical Volume
chlv -n (change the name of the Logical Volume)
extendlv - Increase the site of the Logical Volume

There is no command to reduce the size of a logical volume other than removing the existing one and recreating a new one. This obviously requires a backup and restore.

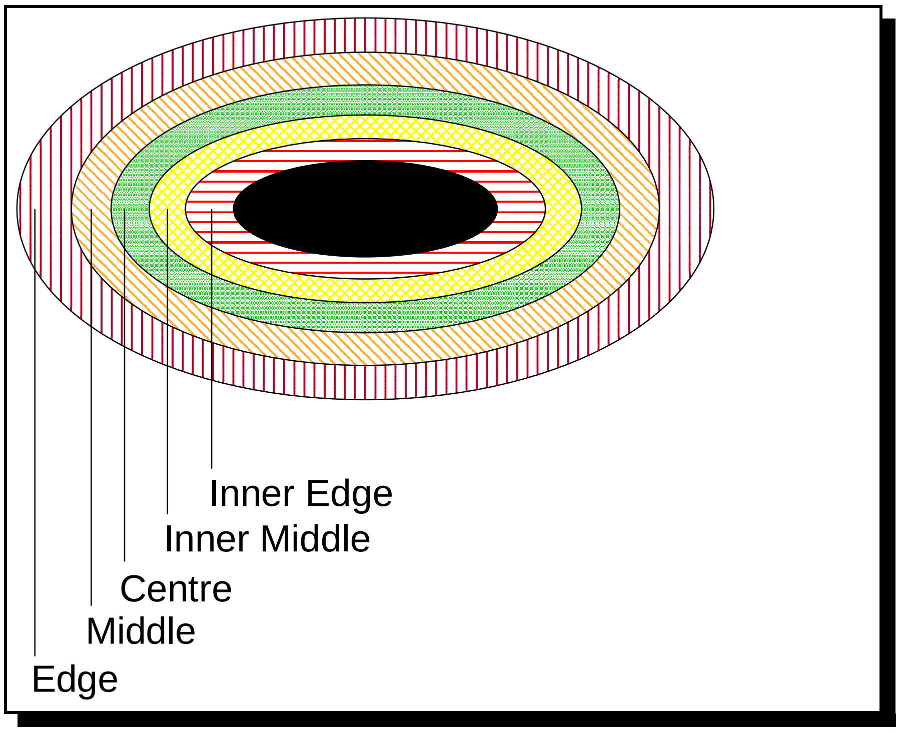
Mirrors can be added or removed using mklvcopy and rmlvcopy respectively. After changing the mirroring the syncvg should be run to synchronise the copies.

If rootvg is mirrored the bosboot command should be run against the Logical Volume to recreate the boot log. The bootlist should have an entry for all disks and for two disks quorum checking should be turned off so that the system can still be started if one disk is damaged.

Policies

Intra-physical Volume Allocation Policy

AIX allows you to specify a preference on the position of Logical Volumes on a Physical Volume. The reason that you might want to do this is for performance. The seek time to get to data held in the centre of the disk is faster than to either the middle or the edge. By putting frequently accessed data in the middle of the disk the overall performance of the system can be improved. This is referred to as Intra-physical volume allocation policy.



Different areas in a disk for Logical Volume Policies

The default value is middle, which has a slow access time. It should be specifically specified if you want faster access.

Inter-physical Volume Allocation Policy

The inter-physical volume allocation policy is another thing that can be set. This determines how the logical volumes are spread across the different physical disks. The maximum number of physical volumes that can be used by the logical volume can be set and the range of volumes to use can be specified. The range of volumes goes from minimum (only use one physical volume) to maximum (allocate across all physical volumes).

Managing File Systems

File Systems can be managed from the File System menu in SMIT. To access this type smit fs.

```
                                  File Systems

Move cursor to desired item and press Enter.

  List All File Systems
  List All Mounted File Systems
  Add / Change / Show / Delete File Systems
  Mount a File System
  Mount a Group of File Systems
  Unmount a File System
  Unmount a Group of File Systems
  Verify a File System
  Backup a File System
  Restore a File System
  List Contents of a Backup









F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

The file systems can be listed using the lsfs command.

```
$ lsfs
Name            Nodename   Mount Pt               VFS   Size    Options    Auto
Acc
/dev/hd4        --         /                      jfs   204800  --         yes
no
/dev/hd1        --         /home                  jfs   204800  --         yes
no
/dev/hd2        --         /usr                   jfs   3366912 --         yes
no
/dev/hd9var     --         /var                   jfs   24576   --         yes
no
/dev/hd3        --         /tmp                   jfs   131072  --         yes
no
```

The mounted file systems can be listed using the mount command. If a file system is not allocated then it cannot be accessed.

```
$ mount
  node       mounted        mounted over    vfs       date        options
-------- --------------- --------------- ------ ------------ ---------------
         /dev/hd4        /               jfs    17 Jan 06:19  rw,log=/dev/hd8
         /dev/hd2        /usr            jfs    17 Jan 06:19  rw,log=/dev/hd8
         /dev/hd9var     /var            jfs    17 Jan 06:19  rw,log=/dev/hd8
         /dev/hd3        /tmp            jfs    17 Jan 06:19  rw,log=/dev/hd8
         /dev/hd1        /home           jfs    17 Jan 06:20  rw,log=/dev/hd8
```

New file systems can be added from within smit. There are two options “Add a Journaled File System” and “Add a Journaled File System on a Previously Defined Logical Volume”. The first option uses default values to create the logical volume to house the file system, whereas the latter option requires the logical volume to already have been created and makes the file system to fit the logical volume.

The command run to create the file system is crfs. This is more functional than mkfs, as (if necessary) it creates the logical volume and updates the odm and /etc/filesystems.

A journaled file system can have it's characteristics changed even if it is in use at the time. The chfs command is used to change a file system. Certain changes may not however take effect until it's mounted again (e.g. changing the mount point). The file system may be increased in size however cannot be decreased again once increased. Increasing the file system extends the logical volume however if you increased the logical volume directly this would not increase the size of the file system.

The rmfs command is used to remove a file system. Before a file system can be removed it must first be unmounted. The file system must also be not in use. When a file system is removed the information in the ODM and /etc/filesystems will be removed and the logical volume will also be removed.

Space Management

AIX provides for dynamic expansion of a file system however does not automatically expand the filesystem. The only way that a file system can be automatically be increased is when software is being installed and is set to automatically increase the file system. The administrator must keep a check on the file system increasing it if it is nearly full. The df command can be run to check the available space.

```
Filesystem    512-blocks      Free %Used    Iused %Iused Mounted on
/dev/hd4          204800    175776   15%     1162     3% /
/dev/hd2         3366912    359432   90%    36459     9% /usr
/dev/hd9var        24576     20840   16%      501    17% /var
/dev/hd3          131072    126344    4%       60     1% /tmp
/dev/hd1          204800      1616  100%     1464     6% /home
/dev/cd0         1197112         0  100%   299278   100%  /.cd_wlvVyc
```

If a file system is continuously increasing or increases more than would be expected then the cause needs to be determined and fixed.

The following files are ones that will continuously grow with the system. These should be checked periodically.

• /var/adm/wtmp	login and logout
• /var/spool/*/*	temporary files used by the printer spooler, and other processes.
• $HOME/smit.log	smit

• $HOME/smit.script	smit
• $HOME/vim.log	vsm
• $HOME/websm.log	wsm
• /etc/security/failedlogin	bad login
• /var/adm/sulog	

The du command can be used to list the disk usage. It lists the number of blocks used by a file or a directory. It is useful to try and get an idea of which files are using the most disk space.

```
# du /home | sort -r -n
195521  /home
195435  /home/stewart
39001   /home/stewart/adobe
39000   /home/stewart/adobe/AIXRS.install
29571   /home/stewart/packages
17569   /home/stewart/.netscape
12325   /home/stewart/.netscape/cache
```

Controlling Disk Usage (quotas)

Left uncontrolled users could use all the available disk space. However the quota system allows limits to be imposed preventing them from using too much space. There are 3 limits that are imposed against a user and / or a group. These are all set against a certain file system.

Soft Limit - defines the amount of space or number of files which the user / group should stay below
Hard Limits - The maximum amount of space or number of files which the user / group can use.
Grace Period - This is a length of time that the user / group is allowed to be between the soft and hard limits. If the user / group does not go below the soft limit during the grace period (normally one week) then the soft limit will effectively be a hard limit preventing any more data being saved until the usage falls below the soft limit.

The disk quota system tracks user and group quotas in the quota.user and quota.group files in the root directories of the file systems enabled with quotas. These are created with quotacheck and edquota commands and can be read using the quota command. Disk quotas are optional and are not setup by default however if there are a lot of users taking up a lot of space it is worth considering turning on quotas.

Before you can use quotas you may need to install the quota management software from bos.sysmgt.quota which is available on the standard install disks. Quotas are turned on by editing the /etc/filesystems file or using the chfs command.

```
/:
        dev             = /dev/hd4
        vfs             = jfs
        log             = /dev/hd8
        mount           = automatic
        check           = false
        type            = bootfs
        vol             = root
        free            = true

/home:
        dev             = /dev/hd1
        vfs             = jfs
        log             = /dev/hd8
        mount           = true
        check           = true
        vol             = /home
        free            = false
        quota           =  userquota,groupquota
```

The file can have userquota, groupquota or both.

To set the individual limits the edquota command is used. A temporary file is created with each user’s or group’s current disk quotas. The EDITOR variable is used to determine the editor to be used.

```
export EDITOR=/usr/bin/vi
```

```
Quotas for user stewart:
/home: blocks in use: 0, limits (soft = 100, hard = 200)
        inodes in use: 0, limits (soft = 0, hard = 0)
~
~
~
~
~
~
~
~
~
~
"/tmp/EdP.a5Hs_ya" 3 lines, 129 characters
```

If you want to setup a standard quota for all users -p option can be used.

The grace period is setup using the edquota with the -t option.

```
Time units may be: days, hours, minutes, or seconds
Grace period before enforcing soft limits for users:
/home: block grace period: 14 days, file grace period: 14 days
~
~
~
~
~
~
~
~
~
~
"/tmp/EdP.aVHs_7a" 3 lines, 166 characters
```

After the quotas have been setup the quotas are turned on using

```
quotaon /home
```

Once on quotaoff is used to turn the quotas back off again.

To check the current limits quotacheck -a is used. The repquota command can be used to read the quota.user and quota.group files.

For regular users the quota command can be used to check on their current disk quota. Included in the users .profile it will give the user a reminder when they login if the limits have been reached.

Defragmenting a File System

Files may get fragmented over time. If the file system is defragmented then the amount of contiguous free space is increased and system performance is improved. The command to run is defragfs

Verify a File System

There is a file system check command to check the integrity of a filesystem. The command is fsck. The following aspects are all checked:

- check the journaled log
- check the blocks to ensure that all files are either allocated to a single file or are free
- check the file sizes
- check directory entries

The -p option (preen) can check a file system and make only minor changes without effecting the user.

Paging Space

There is another use for a logical volume and that is paging space. What paging space does is to allow the system to make use of more memory than is physically available. This is achieved by "paging" chunks of memory to disk when they haven't been used for a while. As disk space is considerably cheaper (per MB) than memory this is a cheap way of increasing the memory available to AIX. However disk access is considerably slower than accessing RAM so there is a price to pay for this.

When applications or data is loaded it is loaded into physical memory and then mapped to the paging space. When the system tries to load something into memory and there is insufficient space, it will take the least-recently-used page of memory and copy it to disk (if it has changed) to free sufficient space. If the page to be removed from physical memory has not changed then it will be stolen as the original copy is already in paging space. If some memory needs to be accessed that has been paged out then it will fetch it back into real memory (paging out further pages if necessary).

This is all managed by the Virtual Memory Manager (VMM).

Paging space is created when AIX is installed. Paging space is set at real memory plus 16MB by default. It is a good idea for up to 256MB of memory to have paging space at twice the amount of physical memory.
The paging space cannot be more than 20% of total disk space.

The amount of paging space is dependant upon the application that is running. A memory intensive application will still need a lot of physical memory it cannot be made up by using virtual memory. What paging memory does is allow data that isn't been accessed to be paged out to disk.

To check on the usage of paging space the lsps command should be run.

```
# lsps -a
Page Space  Physical Volume   Volume Group    Size   %Used  Active  Auto  Type
hd6         hdisk0           rootvg         128MB      41     yes   yes    lv
```

If usage is constantly greater than 70 % then more paging space should be added.

For maximum performance it should be located at the Centre of the disk (see intra-allocation policy). When multiple paging spaces are used, these should be allocated on different physical disks. If multiple disks are available then paging space should be on the one which is fast (if different disk speeds / different SCSI types) and is the less busy.

To query the amount of real memory the lsattr command can be used

```
# lsattr -El sys0 -a realmem
realmem 65536 Amount of usable physical memory in Kbytes False
```

The lsps -s command will give a summary of available paging space.

```
# lsps -s
Total Paging Space   Percent Used
      128MB              42%
```

Details of paging space volumes to activate at startup are held in /etc/swapspaces.

Managing Paging Space

The paging space can be managed from smit using the fastpath smit pgsp.

```
                                 Paging Space

Move cursor to desired item and press Enter.

  List All Paging Spaces
  Add Another Paging Space
  Change / Show Characteristics of a Paging Space
  Remove a Paging Space
  Activate a Paging Space




















F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

An extra paging space volume can be added using the smit mkps fastpath or

`mkps -s 4 -n -a rootvg`
(this would be 4 LV’s in size)

To change paging space use smit chps or the chps command.

The size can only be increased not decreased. The size is specified by giving the number of LV's for the paging space to occupy. Using smit the Logical Volume will be allocated PP's as necessary.
Paging Space can not be deactivated on a running system however can have the autostart option removed so that they are inactive when the system is next booted.

Paging space can be removed using smit rmps or by the rmps command

```
rmps paging00
```

the paging space must first be inactive (see earlier for how to stop paging space from being activated at startup). The system defined paging space (hd6) cannot be removed as this is created dynamically at boot time.

Networking

Most computers nowadays are connected to some kind of network. Even most home computers now connect to the Internet. UNIX computers will usually have at least one network connection. There are some circumstances where a UNIX computer is used through terminals where it doesn't have network connection such as a in a library indexing system. However typically UNIX computers are used for network related purposes such as a network server, or for firewalls and Internet Web Servers.

With the mass take-up of the Internet TCP/IP has become the de-facto standard for networking. Whilst AIX will still support other networking protocols such as SNA (using communications server) it has support for TCP/IP networking built into it's architecture. Indeed when TCP/IP was developed it was incorporated into the UNIX platform.

Basic TCP/IP Networking

TCP/IP is an abbreviation for Transmission Control Protocol / Internet Protocol. It is a set of protocols that define how two or more computers can communicate with each other. This is a set of rules that describe how the data is passed between the computers. The protocol is an open description of how to write the software so that it can be developed for use on any type of computer. Within the TCP/IP networking protocol there are lots more protocols. These provide different functions as part of the networking. These can be integral to the operation of the networking, such as the Domain Name System or could be an application that uses the network such as E-mail (both of these are discussed in further detail later).

TCP/IP is not limited to a certain computer, it is an open protocol that independence from any particular operating system. A heterogeneous network can therefore be created consisting of any combination of UNIX, Windows, Apple or OS/2, OS390 (plus any others) computers.

Whilst discussing I will also be discussing UDP. This often goes alongside TCP. The difference being that TCP is connection based protocol whereas UDP is connectionless. In other words when TCP is being used there is a session setup between the hosts and the transfer is guaranteed. This compares with UDP where the data is sent but there is no checking that it has been received.

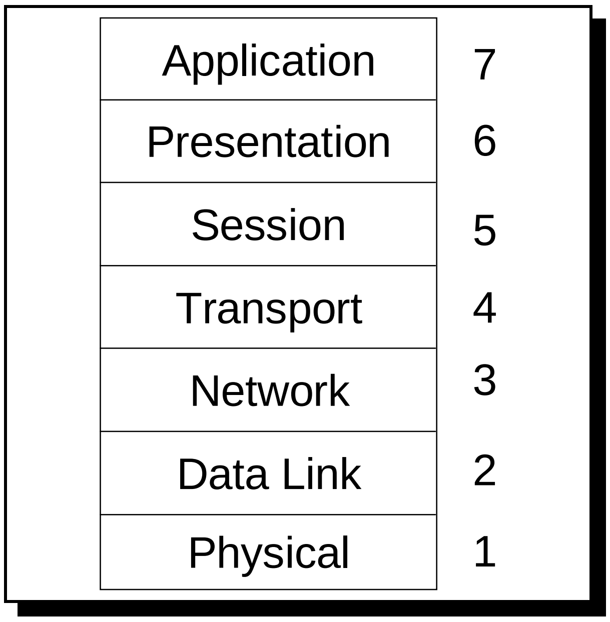
A common way of comparing these is to liken TCP to the telephone system and UDP to the Postal system. With the telephone when you establish a connection with the other person, you know for certain that the user receives the message. If you were disconnected during the telephone conversation then you would know about it and be able to phone the other person again. With the postal system after you post the letter then you do not know for certain whether or not the mail will be received. After you have posted the letter it could be lost or destroyed on it’s way to it’s destination. Or if the person has moved house they may never receive the letter.

At first it may sound that there is no reason to choose UDP over TCP after all if you can have the extra reassurance then why would you care about UDP. The reason for this is that there is a lot of overhead involved in TCP. For each data being sent a confirmation has to be generated and even if there is no data being sent there will normally be some kind of keep alive signal. Whereas for some less important data you may just want to send and forget it with the hope it will reach the other end.

OSI Model

Networking protocols are often described relating to the OSI model. The OSI model splits the different functions of networking into different layers. By describing the networking protocols in layers it allows the layer to be changed without affecting other layers.

The networking models are particularly useful in that it allows the protocol to be implemented on any system. Allowing UNIX computers to talk as a peer with PC’s or even mainframes.

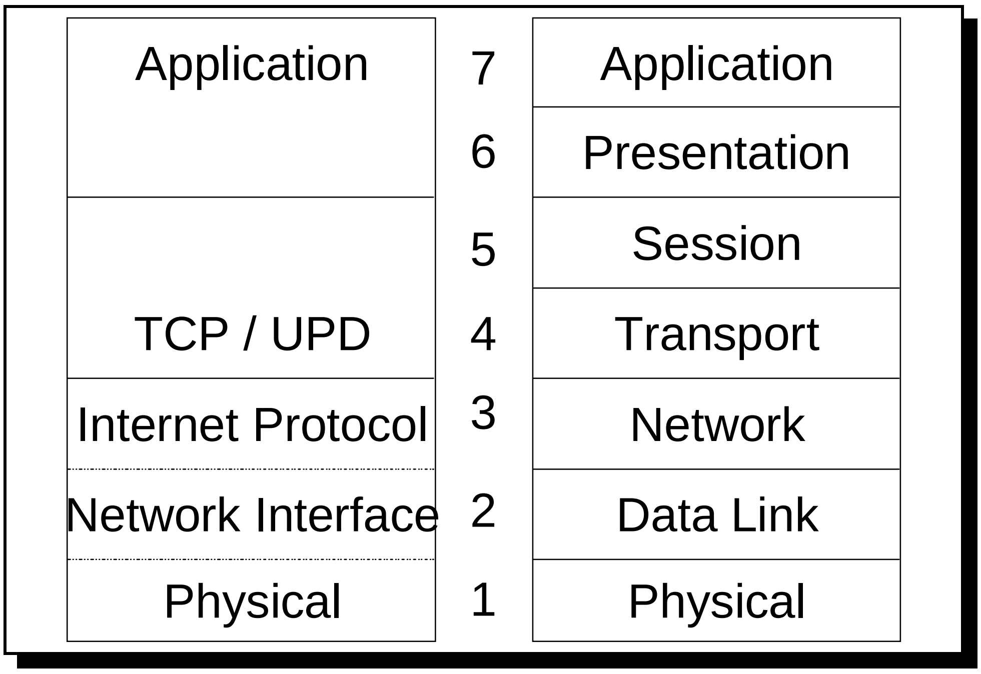


OSI 7-Layer Model

The above diagram shows the 7 layer model. Starting from the bottom the function of the layers is as follows:

Physical Layer - describes the media over which the data travels. For instance this describes the voltage of a 1 or 0 signal across a copper wire.
Data Link Layer - describes the means by which the bits are carried across the physical layer. For example this can describe how the start and end of a data stream is indicated.
Network Layer - this layer handles the routing of data through a network. As an example this describes how routing can happen based upon the address of the computers.
Transport Layer and Session Layer - the transport and session layers provide end-to-end session integrity. This includes keep alives to ensure the connection is maintained.
Presentation Layer and Application Layer. These provide the interface to the application. For example this may include the use of the nslookup command to convert a hostname into an IP address.

Whilst the TCP/IP protocol does not exactly match the OSI 7 layer model it can be approximately mapped across onto it. The following diagram shows the TCP/IP stack compared with the OSI 7 layer model.



TCP/IP Stack Alongside the OSI 7 Layer Model

This model shows how the TCP/IP protocols are mapped onto the 7-layer model. Note that the application and presentation layers have been merged and that the session and transport layers have been merged. The distinction between these layers are not needed in the TCP/IP model. There is however an exception in the NFS application in that it sits on top of the SUN RPC protocol which functions as a presentation layer, however for most purposes they are treat as one layer.

You will also note on the diagram that the borders between layers 1, 2 and 3 are not solid lines. The details are not rigidly defined in the TCP/IP as in the OSI model and the functions are not neccessarily a direct match between the OSI model. I have therefore used the broken line to show that although these layers appear to map directly across between the different models in practice this is not quite the case.

More about TCP/IP

TCP/IP was originally developed for universities and the military to exchange ideas and files. The development of TCP/IP is initiated by the Internet Architecture Board (IAB), and the development of standards is handled by the Internet Engineering Task Force (IETF). The documents produced by the IAB are called Request For Comments (RFC) which describe the protocols and relevant information useful for the implementation. Anyone can submit a document as an RFC which are reviewed before being published as official RFC’s. After an RFC is published and assigned an RFC number is its never revised under the same number.

Instead a new RFC must be created which supersedes the previous version. The RFC's are available from:

ftp://ds.internic.net
http://www.internic.net

other useful sources include
http://www.freenic.net/rfcs/
http://watkiss.members.easyspace.com/computers/network/tcpip/

IP Addressing Scheme

An important part of all networking protocols is the addressing scheme. Without being able to locate the individual machines (or hosts as they are called) then it would not be possible for any communication between the hosts. There will be more than one addressing scheme in use but the most important of these is the Internet Protocol (referred to as IP), this is significant as it provides the addressing for each end of the connection. The other addressing schemes are effectively hidden from the user at layers two or below and are automatically handled by the networking hardware. The current version of IP is called IP version 4 and is the only one that you will normally come across today. The future version of IP lays in version 6 which will most likely replace IP version 4 in the future. When I refer to IP in this book it refers to version 4 unless otherwise specified.

The addresses used in IP consist of four octets and is 32 bits long. The address is stored in a format known as dotted decimal.

ie.
xxx.xxx.xxx.xxx

where xxx is a number between 0 and 255.

Most users however would not actually need to use the IP address. Instead they would refer to the computer using it's host name. The IP address is obtained from the host name using the "Domain Name System" (DNS). There is no actual relationship between the hostname and the IP address instead this uses a lookup table. The Domain Name Service will be discussed later.

The IP addressing scheme provides 2^{32} possible addresses, which could potentially have over 4.2 thousand million individual addresses. The problem with this however is that trying to locate each one of those addresses individual over the Internet would be an enormous task. So instead the address is split into a network and a host portion. The idea being that different organisations can be assigned a network which can have between 256 and 16.7 million addresses available for hosts. The address range now allows up to 3.7 thousand million hosts on 2.1 million network.

To accommodate for different sized organisations which require a different number of host addresses, the addresses are split into different network classes. There are 5 different classes however only 3 are commonly used.

- **Class A** - These are for large organisations. The network portion is 8 bits long and begins with binary 0. There are 126 possible networks each with up to 16.7 million hosts.
- **Class B** - These are for medium sized organisations. The network portion is 16 bits long and starts with binary 10. There are 16 thousand networks each with up to 65 thousand hosts.
- **Class C** - These are for smaller organisations. The network portion is 24 bits long and begins with binary 110. There are 200 thousand possible networks each with up to 254 hosts.
- **Class D** - These are allocated for multicast although are rarely used. The addresses begin with binary 1110.
- **Class E** - These are experimental. The addresses begin with binary 1111.

The table below shows the possible ranges of addresses:

Class A		
0.hhh.hhh.hhh	to	127.hhh.hhh.hhh
Class B		
128.nnn.hhh.hhh	to	191.nnn.hhh.hhh
Class C		
192.nnn.nnn.hhh	to	223.nnn.nnn.hhh
Class D		
224.xxx.xxx.xxx	to	239.xxx.xxx.xxx
Class E		
240.xxx.xxx.xxx	to	255.xxx.xxx.xxx

IP Address Class Ranges

In the above table the nnn's represent the network portion of the address and the hhh's represent the host portion of the address.

The observant, mathematically minded my have noticed that some of the numbers mentioned earlier appear to be incorrect. Some of these are through just rounding down, however the others are due to certain addresses being reserved for other uses.

Reserved Addresses

127.0.0.1	Refers to localhost
All host bits binary 0’s	Refer to the network
All host bits binary 1’s	Broadcast address - send to all addresses

Private Address Ranges (defined in RFC 1918)

Class A		
10.0.0.0	to	10.255.255.255
Class B		
172.16.0.0	to	172.31.255.255
Class C		
192.168.0.0	to	192.168.255.255

Some Reserved IP Addresses

The private address ranges are for use internally within an organisation. They cannot be used on the Internet. To provide Internet access for a host with a private address range the communications have to go through a NAT (Network Address Translation). This is one way that the number of available IP addresses can be preserved.

Apart from the private address ranges all other IP addresses need to be registered with the InterNIC before they can be used.

Subnet Masks

The biggest problem with the IP addressing scheme is that it is rapidly running out of free addresses. The long term solution is to move from IP version 4 to IP version 6 which will provide 2^{128} separate addresses. This should provide for all the Internet will ever need.

One of the problems with the current addressing is that the addresses are given away in large chunks. Subnetting allows these large chunks of addresses to be further split into a further network and host component. This new network component is called the subnet.

The following shows how a class B network address could effectively split into 254 separate virtual class C networks:

nnn.nnn.sss.hhh

nnn = network portion of the address
sss = subnet portion of the address
hhh = host portion of the address

The network portion has been fixed so still stands as the first two octets. The next octet which would normally be part of the host address is then made to signify the subnet and effectively becomes part of the network address. The final octet is left as the host portion of the address.

If we change which part of the address represents the network and host then we need to tell the computer and any routing devices of that. The technique used is known as creating a subnet mask.

The subnet mask for the above example would be 255.255.255.0 as we can see this is in a similar format to the IP address. To explain how this is derived requires a little bit of binary arithmetic. I will attempt to briefly explain how this works, however am unable to devote a large section to it. If you need further explanation then there are a number of different books purely devoted to TCP/IP most of which spend a considerable effort in explaining the concept of subnetting.

Whilst an IP address is generally represented as decimal numbers to make it easier[1] for people to understand, however the computer works on binary numbers which can only represent one or zero. For example the following address shown as dotted decimal and binary.

```
172      .      16       .      3        .      4

10101100        00010000        00000011        00000100
```

As you can see writing this as binary every time would be very tedious and prone to errors.

To then create a subnet mask we need to use a binary one for every bit of the address that represents the network portion and a binary zero for any bit of the address that represents the host portion. This would give us:

```
11111111        11111111        11111111        00000000
```

We convert this to decimal to make it easier to read and it gives us a subnet mask of

```
255      .      255      .      255      .      0
```

Using simple binary arithmetic the computer can use the subnet mask to convert the IP address into it’s network and host portion. It would use a binary AND to get the network portion. To get the host portion the subnet mask is inverted (NOT function) and then AND’d against the IP address.

[1] Some people would argue that hexadecimal would have been a better choice for IP addresses as it is easier to manipulate when calculating subnet masks etc. IP version 6 uses hexadecimal for human readable representation and not the dotted decimal method used in IP.

Just to confuse matters further some machines (e.g. Cisco routers) use a different notation to represent the subnet mask. The would count in the number of ‘1’ bits and give that as the subnet mask number. So in this example the subnet mask would be represented as /24

The example above showed the subnet mask on a octet boundary however it is more common to see a subnet mask within an octet. For example the subnet mask 255.255.255.248 might be used to split a class C network address into 30 subtends each with 6 hosts.

The expanded mask would be:

11111111 11111111 11111111 11111000

Taking only the last eight bits the host portion is

11111	This potentially can have 32 subtends excluding reserved addresses (all ones and all zeros) gives 30 valid addresses.

The network portion is

000	This potentially can have 8 hosts excluding reserved addresses (all ones and all zeros) gives 6 valid addresses.

The subtends are given a number which is when all the host portion are zero. All the rest of the addresses are valid until the part where all the host bits are ones which is the broadcast address for that subnet.

Looking at only the last octet the following table shows how some of the address will be made up.

Subnet Number	First Address	2nd address	...	Last address	Broadcast
8	9	10	...	14	15
16	17	18	...	22	23
24	25	26	...	30	31

To try and understand this better convert the values in binary and then identify the host and network portions of the address.

Whilst I have excluded the 0 address it is sometimes possible to actually use this. For this you may have to ensure that your routers support this and that the feature is turned on. It is however not recommended.

A alternative subnet mask could be 255.255.255.224 which would give 6 valid subs each having a maximum of 30 hosts (this could be useful for splitting up a smaller company which might have 6 different LAN segments with up to 30 machines on each). You may find it a useful exercise to try and calculate these values for yourself.

The opposite of subnetting is called supernetting. instead of dividing network ranges into subtends a number of subtends are joined together to make a supernet. The class A and B network ranges have been all but used up and so instead several class C networks are grouped together for larger organisations and ISP's.

Sockets

Whilst the IP address provides the connection to the correct machine, it cannot distinguish the different service that is required. The port is used to distinguish the application. It is a value from 0 to 65535. The combination of IP address, port and protocol is called a socket, and has to be unique for every service.

The first 1000 ports are reserved for specific applications. These are referred to as well known ports. These are defined in RFC 1340. Some of the most common ports are:

20 & 21	FTP
23	Telnet
25	SMTP (Simple Mail Transfer Protocol)
53	DNS
80	World Wide Web
110	POP3 (Post Office Protocol)
144	News
6000	X-Windows

Whilst 6000 is out of the range of the reserved numbers it is commonly used. Most of the other ports above 1000 can be used for any other purposes.

Other Addressing Protocols

There are other addressing protocols used. These are at lower levels of the protocol stack and differ depending upon the media being used. The most commonly used of these is the MAC (Media Access Control) address. The ARP Protocol (Address Resolution Protocol) is used to allow IP addresses to be translated into MAC addresses. The following diagram is used to show how this works.

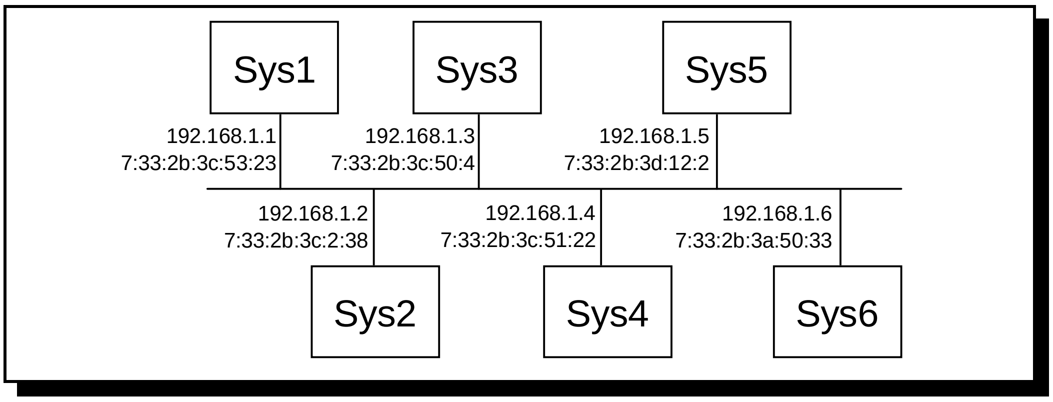


Diagram of Ethernet with ARP addreses

The ethernet does not know anything about IP addressing. The IP addressing occurs at layer 3 which is higher than Layers 1 and 2 that the ethernet works at. Instead they use a MAC address which consists of 6 numbers separated by colons. This allows different networking protocols to be carried over ethernet such as SNA (Used by IBM Mainframes) or IPX (formally the default addressing scheme used by Novel Netware).

The MAC address is hard coded into the ethernet card and are unique across every device made. This is achieved by allocating a block of addresses to each manufacturer of ethernet devices. Normally the user would not know or care about the value of the MAC address as it is transparent to the user.

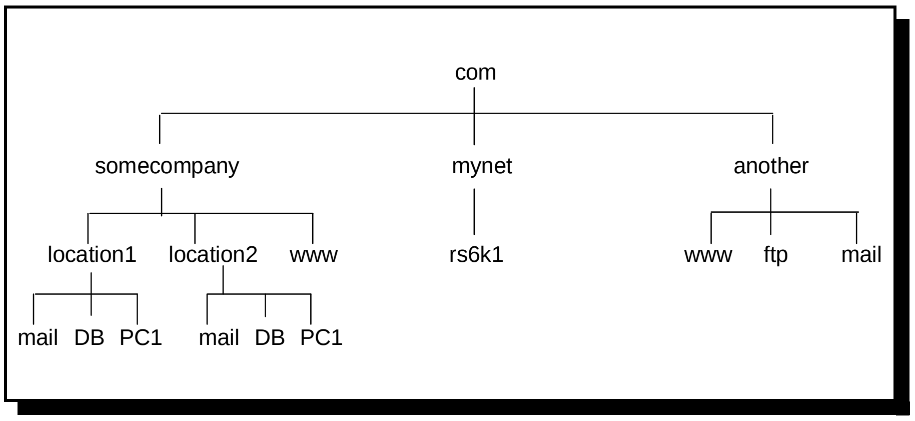
So when a system e.g. Sys1 wants to communicate with another e.g. Sys4 then the user would use it's IP address e.g. 192.168.1.4. Now Sys1 needs to convert this address into the MAC address of Sys4. It therefore issues a MAC broadcast to all machines asking for the machine with IP address 192.168.1.4 to reply. Sys4 will reply with it's MAC address 7:33:2b:3c:51:22. Sys1 then adds the IP address and MAC address of Sys4 to it's ARP table. Sys4 likewise knows the IP address and MAC address of Sys1 (as Sys1 included it's IP address in the original broadcast) so it adds that to it's ARP table. Now in future when ever the systems want to communicate they just lookup the MAC address in the systems ARP table. This process is known as ARP.

If the machine is not located on the same LAN then this requires IP routing which is explained later.

Domain Name System (DNS)

Whilst the IP addressing scheme allows computers to communicate with each other it's not particularly an easy way for people to remember. Which would you find easier to remember www.easytoremember.com or 172.16.35.122 ?

Hostnames have an hierarchical structure. The names read from right to left as though moving down a tree.



Example DNS tree

To take a few of these examples.

1. Starting from the top the first domain below the root is known as the root domain. In this case it is com.
2. The next one is the companies or organisations domain for instance somecompany.
3. In large companies they may then split the domain into further subdomains for example by locations. As you look on the tree however not all the machine names have to be included within a subdomain they can end at this level (or indeed at the level above this if necessary). Also for smaller companies (such as "another") they may not have any need to further divide into subdomains.
4. Finally the hostname is on the last part of the tree. For example the DB machine.

The final name of this machine is db.location1.somecompany.com

The responsibility of dividing up all the names below the company name is owned by the company or organisation. However the organisation domains obviously need to be allocated by a governing body to ensure that two companies don't try and use the same one. This is administered by local organisations dependant upon the top level domain. The top level domain names are allocated by IANA. Currently these are:

arpa	Used for DNS mapping
com	Commercial
edu	Educational
gov	Government
mil	Military
net	Network support groups or ISP's
org	Other organisations (normally charities)

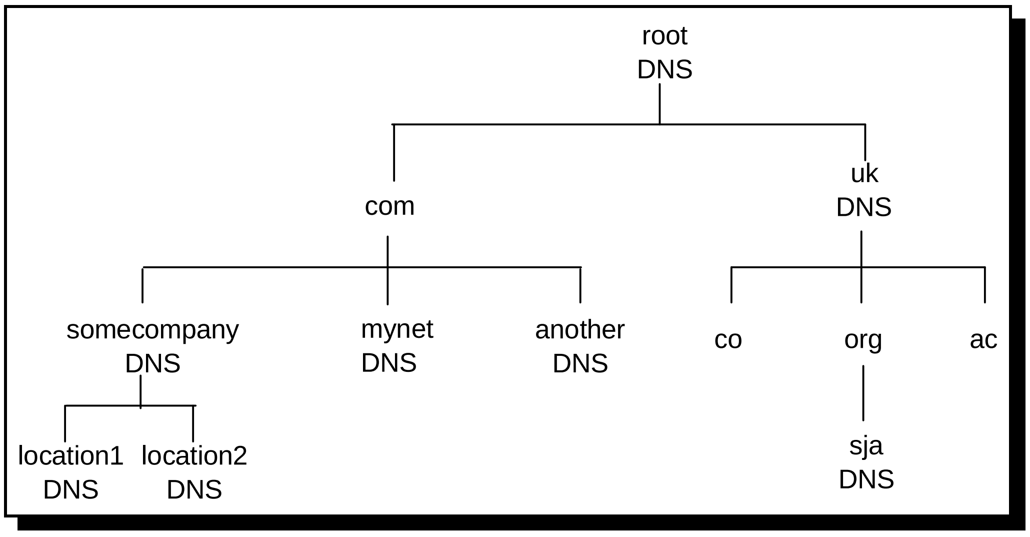
int	International Organisations

These were originally designed for use by US based groups however are also used by anyone wanting an International Domain Name. The other countries are free to allocate domains under their country code. For example the top level domain for the United Kingdom is uk. Some examples are:

ac.uk	Academic Community (Education)
co.uk	Commercial
gov.uk	Government / Councils
ltd.uk	Limited Companies
org.uk	Other organisations (normally charities)

The next question is how these are actually implemented. The most basic way this can be implemented is in the /etc/hosts file. The host file is a list of hostnames and their IP addresses which allows them to be directly mapped. This works fine for a small organisation however if you wanted to access machines across the Internet would require an entry in this file for every computer attached to the Internet. So instead the Domain Name System provides a mechanism to off load this to different organisations.

Each computer has the IP address of it’s local DNS machine. Whenever a program is presented with a host name the machine will make a nslookup to it’s local DNS to get the IP address. The local DNS machine will look in it’s internal tables to see if it has a match and if not it will go to a different DNS machine until either a translation is made or it fails. A DNS machine will have a zone of authority which will be one or more domains owned by that organisation.



DNS Zone Examples (not necessarily representative of the real setup)

The above diagram allows us to explain the use of the DNS hierarchy system using an example.

A computer PC1 at location1 of somecompany wants to access www.sja.org.uk (this is the website of the UK First Aid Charity St. John Ambulance). PC1 performs an nslookup to it's local DNS "location1 DNS". This DNS does not know about the existence of the web site so it asks the next level up which is the overall DNS of the company. This DNS still doesn't know about the machine so it asks the root DNS. This DNS does not know anything about the computer in question however it does now about the DNS that owns the uk domain and then passes onto that DNS machine. In this example the uk DNS still doesn't know about this machine however does now about the sja.org.uk domain so passes it on to that DNS machine. This machine does have an entry for the machine. In this case it is known as an authoritative answer as it is 100% certain that this is the IP address because it owns it. It passes the entry to the uk DNS, which in turn passes it to the root DNS, back to the DNS of somecompany, to location1 and finally back to PC1.

This sounds like a very long process if it has to be carried out for every machine that is to be accessed. To speed up the DNS process a lot of the DNS machines provide a caching feature where they can store the result of some of the lookups they perform. The names cached can either be for specific hosts (although except for popular sites they will be less likely to have a hit on the cache). Alternatively the DNS will cache the address of another DNS server allowing it to bypass some of the process (for example caching the uk DNS at location1 DNS would allow it to skip two different DNS servers). The use of a DNS cache is so significant that there are even caching-only DNS servers that do not act as a zone of authority for any domain.

If a Domain Name Server is unavailable then it would not be possible to access other machines. Therefore a backup server is configured as a fallback these are called secondary name servers. So that each DNS server does not have to be manually updated whenever a new entry is created. Therefore the primary name server will push it's configure to any secondary servers.

The DNS process is discussed in RFC's 1034 and 1035.

Routing

If two machines are connected together as a point-to-point connection over a physical connection then they can communicate between each other directly. However once we start to communicate to computers on other networks, or over the Internet then routing is needed so that the data reaches the correct destination.

The devices that handle the directing of traffic are known as routers.

These routers take an incoming packet and based upon the destination address send them through a different interface to either another router or to the end destination.

For a normal host computer all that is needed to handle the routing of all packets is to define the default gateway. The default gateway is a router directly attached to the same LAN segment as the host that knows how to route the packets on. This is normally set in the "Minimum Configuration" which is described further in the section "Configuring TCP/IP". Then for any address that is not locally held then it forwards the packet to the local router asking it to forward on to it's destination. Alternatively for different networks the system could have multiple routes defined for different networks or hosts, or could participate in a dynamic routing protocol.

The router will then forward the packet on directly to the host or onto another router. Whenever a packet passes through a router this is called a hop.

There are three different types of routes. They could be implicit, static or dynamic. Implicit routes are where the configuration of TCP/IP indicates that the address is local to the machine (i.e. on the same physical LAN segment). Static are individually defined (often this will include a default route) and dynamic is where a networking protocol is used to identify the most appropriate route for different connections.

Static Routing

For static routes each entry in the routing table is added by using the route command (or through SMIT). This is normally used to connect a host to it's networks, but can be used for routers typically in smaller easy to manage networks. If there are two network interfaces on an RS6000 then it will not allow packets to be routed between the interfaces unless routing is enabled by enabling ipforwarding.

```
no -o ipforwarding=1
```

or disabled again using

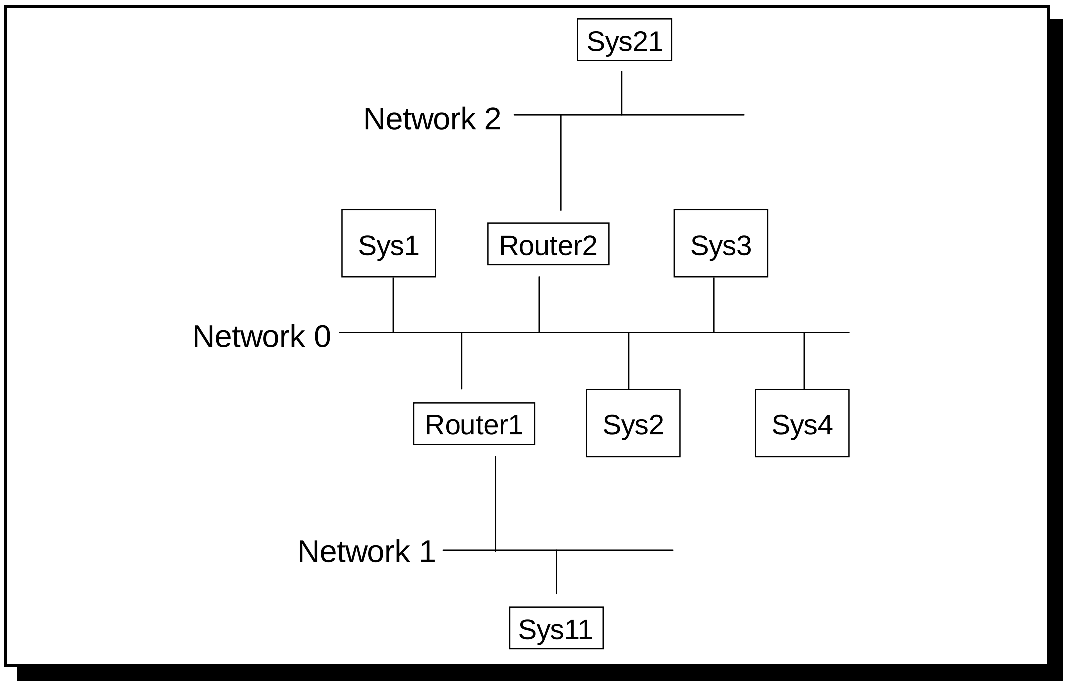
```
no -o ipforwarding=0
```

This needs adding to the /etc/rc.net file to include the change after a reboot.

Static routes are added using the route command (or through SMIT). The route command is explained later under Network Commands.

ICMP Redirects

It is possible that when a packet is sent using static routes that it will not neccessarily go the most direct route. For example if there are two routers on the LAN one of which goes directly to the host but the other would have to pass it to the other. This is illustrated below.



ICMP Redirect Example

Here we have Sys1 which is on network 0. There are two routers on the same LAN segment but Sys1 only has a default route pointing at Router 1. When Sys1 wants to communicate with Sys21 it first sends it's request to Router1. Router1 realises that it has to forward it on to Router2 and that it would have been easier for Sys1 to have sent it directly there. It forwards the packet onto Router2 so that it reaches Sys21, but then also sends an ICMP redirect message to Sys1. Sys1 then adds a route in it's routing table to send any packets for Sys21 to Router2. Then when Sys1 next needs to send a packet to Sys21 it can send it directly to Router2.

AIX can handle these ICMP redirects or it can ignore them. This is set by using the no command to set either icmpsendredirects and icmpignoreredirects, both of which can be enabled or disabled.

Dynamic Routing

There are three dynamic routing protocols in general use all of which are supported by AIX. These are RIP (Routing Information Protocol), RIP 2 and OSPF (Open Shortest Path First). These work by routers constantly communicating to each other describing the network to each other. RIP uses the hop count (i.e. the number of routers the packet would travel through) to determine which route to send the packet through. OSPF is more sophisticated and allows the network administrator to set metrics to indicate a cost in using a certain route. This allows more expensive links (e.g. dialup connections) and for faster links to be preferred (e.g. those with higher bandwidth or shorter delay times).

RIP is enabled in AIX by starting the routed daemon.
RIP 2 is enabled by starting the gated daemon with an empty /etc/gated.conf file.
OSPF is enabled by starting the gated daemon.

These are all interior protocols as they are used within a network. To connect to other networks an exterior protocol is used and this is BGP (Border Gateway Protocol).

Routing Information Protocol (RIP)

RIP is a simple protocol based on distance vectors. It uses a shortest path algorithm to determine the best route to the destination. This is measured in hops which is normally then number of gateways (routers) that are passed through before reaching a destination network. The routing daemon dynamically learns about the network using the RIP protocol and builds it’s own routing tables.

The line speed, reliability or cost are not taken into account when looking at the shortest link. There is a maximum hop count of 15 using RIP. Any destination over 15 hops away is considered to be an infinite number away and cannot be reached. This is a required feature of the RIP protocol as it is possible to get routing loops where the routers through having out of date routes or static routes pass the packet around in a continuous circle.

Whilst suitable for small to medium networks this does not transfer well to a large network, due to it’s inflexibility and it’s low hop count.

The updates between routers are sent using UDP on port 520. When a router joins the network it broadcasts requesting for other routers to send their routing tables. Thereafter the router will advertise it’s tables to it’s neighbours every 30 seconds. Also if there is an update indicating a change in the network a router will send it immediately (almost).

RIP Version 2 (or RIP 2) provides some enhancements to the RIP protocol. This is documented in RFC 1723. The new features include:

- Authentication - only accepts updates when provided with the correct password
- Route Ta - Allows a tag value to be added to indicate that a link is external
- Subnet Mask - Allows RIP to work in variably subnetted networks
- Next Hop - Max RIP more flexible when used in a network with multiple routing protocols (i.e. OSPF and RIP)
- Multicasting - Allow routers to multicast updates which is more efficient that using a broadcast.

The routed daemon is configured by following the following steps (this is used for RIP only):

- Add any known networks to /etc/networks file. This is not often used but makes the loading of routes quicker.
- Add any gateways not directly connected to the network into /etc/gateways. This is optional but speeds upto the updating of the routing table.
- Uncomment the line in the /etc/rc.tcpip that starts routed. The following options can be specified
  - -s active (gateway)
  - -q passive (host)
  - -t activate tracing
  - -d activate debugging
  - -g for gateway
- Start the routed daemon

  This can be done using SMIT or by issuing a startsrc -s routed

If using OSPF or RIP 2 then the gated daemon needs to be started. For RIP 2 then the gated daemon is stated with an empty configuration file.

Open Shortest Path First (OSPF)

OSPF is a Link State protocol, therefore uses a a distributed map concept. The network map is a database help by each node and updates and performed by “flooding”. All map updates must be secured.

In link state protocols each router is responsible for determining the identity of it’s neighbours. The router constructs a link state package (LSP) which lists it’s neighbours and the cost of the link. This is transmitted to all routers which then store the most recent LSP received from each router. The routers then construct a link state packet database from which the routes through the network are calculated.

The routers are normally grouped into areas. The routes in one area will summarise the information to send to the other areas. This limits the size of the link state database and the number of advertisements.

The OSPF protocol provides fast conversion and multiple metrics allowing for throughput, delay, cost and reliability to be taken into consideration. OSPF also allows for multiple paths to a destination providing immediate fallback in the event of a failure. Authorisation is provided for the routers (not available in RIP version 1). There is also no limit on network size with OSPF.

OSPF also allows for load balancing over links although this is only provided by routers it is not supported by AIX.

TCP/IP for AIX

AIX supports a number of different methods of connecting to a network. The following list of interfaces listed some of the most common interfaces:

- Standard Ethernet (also known as DIX ethernet) [en]
- IEEE 802.3 Ethernet [et]
- Token Ring [tr]
- Serial Line Internet Protocol (SLIP) [sl]
- Point-to-point Protocol (PPP) [pp]
- Loopback [lo]
- FDDI [fi]
- ATM [at]
- ISDN [pp]

The loopback interface is used for the host to send messages to itself. The FDDI, ISDN, SLIP and PPP are used for serial connections. The ATM (asynchronous transfer mode) is used for either LAN or WAN networks. These are all layer one devices which talk to the physical medium.

Configuring TCP/IP

Due to the complexity of networking there are a number of different steps that need configuring before it will work correctly. However if the interfaces were connected when the initial installation of AIX is performed then this can be done through the configuration menus.

Networking Software Packages

The appropriate support files must first be installed on the machine. These are all part of the bos.net package. There are different filesets depending upon the protocol and services being used. Some example filesets include:

bos.net.ppp	PPP
bos.net.nfs.client	NFS client
bos.net.nfs.server	NFS server
bos.net.nis.client	NIS client
bos.net.nis.server	NIS server
bos.net.tcp.adt	TCP/IP Application Toolkit
bos.net.tcp.client	TCP/IP Client
bos.net.tcp.server	TCP/IP Server
bos.net.tcp.smit	TCP/IP SMIT menus
bos.net.uucp	uucp (Unix to Unix Copy)

Configuring the Adapters

Installing the adapters will normally cause them to be automatically detected and configured during system startup. If there are any problems or these are not configured correctly then
```
smit commodev
```
or the
```
chdev
```
command can be used to change them.
For example an ethernet card could have two interfaces a BNC connection (10base2) and a Twisted Pair UTP connection (10baseT) and you may have to switch between them.

For each network adapter a network interface is created. The most popular of these are shown in the table below:

Adapter Prefix	Interface Prefix	Description
ent	en	Ethernet (DIX)
	et	Ethernet IEEE 802.3
tok	tr	Token Ring
tty	sl	Serial Line Internet Protcol (SLIP)
tty	pp	Point-to-Point Protocol (PPP)
atm	at	Asynchronous Transfer Mode (ATM)
fddi	fi	FDDI
	lo	Loopback
diva	pp	ISDN

for example if you added two token ring cards they would have an have adapters tok0 and tok1 and would have interfaces defined as tr0 and tr1.

This will normally be added automatically however may need to be added manually if the device was added manually. To add or change network interfaces then you should use goto the “Network Interfaces” page in smit and then onto “Further Configuration”. Or manually configure using the mkdev/chdev commands followed by the ifconfig command.

The above table has two different entries for ethernet. This is because of two different standards used. The most popular is sometimes referred to as DIX Ethernet or more commonly just ethernet. This was developed between DEC, Intel and Xerox and hence was called DIX Ethernet. The other was another standard from the Institute of Electrical and Electronic Engineers.

The actual networking runs in kernel address space although is actually a separate entity. It is called inet0. This can be tuned by using the standard device management commands such as chdev.

Setting up TCP/IP

To actually define the network details use the smit menu “Minimum Configuration and Startup” from the TCP menu.

```
                        Minimum Configuration & Startup

 To Delete existing configuration data, please use Further Configuration
menus

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                        [Entry Fields]
* HOSTNAME                                           [watkiss]
* Internet ADDRESS (dotted decimal)                  [192.168.1.1]
  Network MASK (dotted decimal)                      [255.255.255.0]
* Network INTERFACE                                   en0
  NAMESERVER
           Internet ADDRESS (dotted decimal)         [192.168.32.250]
           DOMAIN Name                               [mydomain.com]
  Default GATEWAY Address                            [192.168.1.10]
  (dotted decimal or symbolic name)
  Your CABLE Type                                     N/A
  +
  START Now                                           no
  +


F1=Help             F2=Refresh          F3=Cancel           F4=List
Esc+5=Reset         F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

We have discussed most of the settings so far. The options will however be slightly different for different interface devices. For example token ring will ask for the ring speed (4/16/auto) whereas ethernet may ask for the different cable types (bnc/utp).

The screen also asks for a default gateway address. For any computers within the same subnet the computer can just write to the appropriate network interface. However beyond the computers own subnet requires a router attached to the local subnet that it can route via. This will be discussed later as part of the route command.

This single smit menu screen provides a way of entering all the details in one go that would normally take several different commands to achieve.

If there is more than one network interface on the machine then the others would be defined using the "Further Configuration Menu".

To display or change the parameters of an interface the ifconfig command can be used. To just display the interface you would use ifconfig followed by the device name (this can be done by any user on the system). To actually reconfigure the interface (root only) you would use the following syntax.

```
ifconfig int inet ipaddress netmask ipnetmask parameters
```

an example might look like:

ifconfig tr0 inet 10.1.3.7 netmask 255.255.255.0 up

Hostname and Address Translation

The hostname of the machine must be stored locally. This is the name of the computer excluding it's domain name. This is displayed using the command hostname (it defaults to localhost). The hostname command can also be used to change the hostname although this will be lost upon restart so instead the chdev command (or smit) should be used.

```
chdev -l inet0 -a hostname=hostname
```

The hostname is also used on the login screen particularly if using the graphical login.

The /etc/hosts file has been mentioned before as part of the explanation of DNS. Basically it provides a mechanism for hostname to be translated into an IP address. At the very least it should have an entry

```
127.0.0.1       loopback localhost
```

This would already have been updated if smit was used for minimum configuration. Other hosts can be added by editing the text file or by using the command

```
smit mkhostent
```

To allow the local computer to access a remote Domain Name Server then the /etc/resolv.conf file is created with the address of the Domain Name Server. The file would look like the following:

```
domain         mydomain.net
nameserver     ipaddrserver1
nameserver     ipaddrserver2
```

/etc/resolv.conf file for a DNS client

Note for a DNS server this file would instead be empty.

The normal order for determining an IP address is:

1. Domain Name Server
2. NIS server
3. /etc/hosts

Networking Commands

Here we will show some common Programs that make use of the network connection. There are a number of commands used to perform problem determination and to monitor the status of the network connection. Some of the common commands are explained in this section. I have however not made an effort to list all possible network commands but more to give an idea of what can be achieved. Many of the commands that I have not included may not be used for security reasons anyway.

netstat Command

The netstat command as you may guess gives you the network status. It can display the active connections and sockets for each protocol as well as the routing information and the statistics of the data transferred over the connection.

It is often useful to run the command on it’s own with no command line options. This displays the status of any connections as well as the status of any sockets.

Here are some more examples of how the netstat command can be used to gain information on the state of the network.

netstat -n Turns off name resolution, so that the command will show only IP addresses and not translate them into host names.
netstat -I Shows the state of the configured interfaces. This includes statistics for errors, collisions and the number of packets transferred.
netstat -v This shows the device drivers including information on collisions etc.
netstat -m Shows memory usage
netstat -u Shows open ports
netstat -r Shows defined routes

```
netstat
Active Internet connections
Proto Recv-Q Send-Q  Local Address          Foreign Address        (state)
tcp4       0     17  myhost1.mynet1..telne remotehost.mynet..1365  ESTABLISHED
tcp        0      0  myhost1.mynet1..ftp-d remotehost.mynet..imsld ESTABLISHED
tcp4       0      0  myhost1.mynet1..ftp   remotehost.mynet..score CLOSE_WAIT
tcp4       0      0  localhost.49213        *.*                    LISTEN

Active UNIX domain sockets
SADR/PCB  Type   Recv-Q Send-Q Inode    Conn     Refs     Nextref  Addr
70043400 stream      0      0 1341c220        0        0        0  /tmp/.X11-unix
/X0
7007f000
7003e000 dgram       0      0 13763ea0        0        0        0 /dev/log
70041e80
7003be00 dgram       0      0 13606260        0        0        0  /dev/.SRC-unix
/SRCmUeQaa
70041e40
7003e200 dgram       0      0 1341c7a0        0        0        0  /dev/SRC
70041ec0
7003b800 dgram       0      0 13049380        0        0        0  /dev/.SRC-unix
/SRCsieQab
70041dc0
7003b600 dgram       0      0 13278d80        0        0        0  /dev/.SRC-unix
```

Netstat Command with Default Display

```
$ netstat -I en0
Name  Mtu   Network     Address            Ipkts Ierrs    Opkts Oerrs  Coll
en0   1500  link#2      8.0.5a.fc.e9.38        90     0      102     0     0
en0   1500  192.168.1   stewart                90     0      102     0     0
```

Netstat Command Showing Ethernet Port

ping Command

The ping command is often used to check that a network host can be reached. Ping stands for (Packet Internet Network Grope). The ping command sends an ICMP ECHO_REQUEST packet. Any computer receiving this should then reply with an ICMP ECHO_REPLY. If the reply is received before the timeout period then this is concerned as a success and the remote computer is reachable. In addition the ping command will record the length of time taken for the reply to get back.

It is worth bearing in mind that this is an ICMP request and so works to prove that the lowest 3 levels of the 7-layer model are working. So it proves that the network interfaces work, up to the IP layer is working OK. It does not check whether or not any applications accept connections. Some firewalls will also block ICMP traffic so it may be possible to connect to the host even though the ping may fail.

The easiest way of running the ping command is to include the remote hosts hostname or IP address after the ping command. e.g.

```
ping 10.1.3.5
```

Using this the ping command will constantly send requests to the remote machine. To stop the command from running you should press CTRL-C .

The alternative is to use the -c option to specify the number of times you would like the command to run. e.g.

```
$ ping -c 5 192.168.1.3
PING 192.168.1.3: (192.168.1.3): 56 data bytes
64 bytes from 192.168.1.3: icmp_seq=0 ttl=255 time=1 ms
64 bytes from 192.168.1.3: icmp_seq=1 ttl=255 time=1 ms
64 bytes from 192.168.1.3: icmp_seq=2 ttl=255 time=1 ms
64 bytes from 192.168.1.3: icmp_seq=3 ttl=255 time=1 ms
64 bytes from 192.168.1.3: icmp_seq=4 ttl=255 time=1 ms

----192.168.1.3 PING Statistics----
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 1/1/1 ms
```

Ping Command

would send 5 pings to the remote machine.

When looking at network problems they sometimes get worse when the packets being sent contain more data. So the -s option can be used to specifying the size of the packet from 1 byte to 8184 bytes.

The -R option can also be set to the ping command to RECORD_ROUTE. This adds details of any hops that the ping took. This is not supported by all networking devices.

```
$ ping -R -c2 test1.mynet.com
PING test1.mynet.com: (10.18.145.6): 56 data bytes
64 bytes from 10.18.145.6: icmp_seq=0 ttl=61 time=20 ms
RR:     10.31.232.6
        10.31.253.163
        router6.mynet.com (10.141.0.6)
        router7.mynet.com (10.18.145.6)
        10.31.253.162
        10.31.232.1
        10.140.160.1
        watkiss.mynet.com (10.18.209.124)
64 bytes from 10.18.145.6: icmp_seq=1 ttl=61 time=37 ms
RR:
----test1.mynet.com PING Statistics----
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 20/28/37 ms
```

host Command

The host command will convert a hostname to it's IP address or vice versa. This uses the contents of the /etc/hosts, the DNS nameserver and any NIS domain servers, just as the computer would use to resolve an hostname given to an application.
e.g.
host myhost2

```
$ host myhost2
myhost2.mynet.com is 192.168.1.3
```

nslookup

The nslookup command converts a hostname to it's IP address. This sounds similar to the host command however nslookup will only query the specified DNS nameserver (or the machines default nameserver). Also this can be used to query specific nameservers.

To run against your default nameserver
nslookup hostname.mynet.com

```
$ nslookup myhost2
Server:  ns.mynet.com
Address:  192.168.1.254

Name:    myhost.mynet.com
Address:  192.168.1.3
```

DNS Lookup

or to query a specific nameserver
`nslookup hostname.mynet.com - dnsserver.mynet.com`

If there is a reverse lookup entry then it’s also possible to get the hostname from the address

```
$ nslookup 192.168.1.3
Server:  ns.mynet.com
Address:  192.168.1.254

Name:    myhost.mynet.com
Address:  192.168.1.3
```

Reverse DNS Lookup

The SOA information can be queried by using the following command

`nslookup -querytype=ANY myzone.mynet.com`

It is also possible to use the nslookup command in interactive mode. The following example shows how you could list all the entries in the domain mynet.com putting the output into a file called mynet.txt .

```
$ nslookup
Default Server:  ns.mynet.com
Address:  192.168.1.254

> ls -d mynet.com > mynet.txt
[ns.mynet.com]
##########
Received 536answers (0 records).
Success
> CTRL-D
```

An exit command or CTRL-D will exit from interactive mode.

The ls -t command can be used in interactive mode to visit the SOA information.

There is also a debugging mode that can be used with nslookup. To change to debug mode whilst in interactive mode then the following commands can be used:

```
set d2
```
or
```
set debug
```

Debug can then be switched off by using nod2 or nodebug. Using d2 will still leave the program in debug mode.

The d2 mode will display queries sent out and received, whereas debug will only show the received queries.

The .nslookuprc file can be used to setup a number of options when nslookup starts.

tracerte Command

The tracerte command is used to identify how a packet travels through the network. It records each hop on the route to the destination. When a host cannot be reached this command is useful in finding how far the packets are getting and hence identify the fault. The command also indicates how long it takes to travel between each hop indicating where potential performance problems could lie. The tracerte command uses techniques that aren't officially built into the IP protocol. It relies instead on setting the Time To Live (TTL) value to one that will expire and hoping that the router will return this. Also each command is based on multiple packets that may take different routes which may give misleading results.

Despite these problems it can still be a useful command.

```
$ traceroute testhost2.mynet.com
trying to get source for testhost2.mynet.com
source should be 10.18.209.124
traceroute to testhost2.mynet.com (10.18.148.136) from 10.18.209.124
 (10.18.209.124), 30 hops max
outgoing MTU = 1492
 1  10.14.160.1 (10.14.160.1)  17 ms  7 ms  12 ms
 2  10.31.232.1 (10.31.232.1)  18 ms  22 ms  28 ms
 3  10.31.253.16 (10.31.253.16)  27 ms *  14 ms
 4  testhost2.mynet.com (10.18.148.136)  26 ms  29 ms  26 ms
```

route Command

The route command can be used to add or change static routes. To display routes the netstat command would be used.

```
$ netstat -r
Routing tables
Destination      Gateway           Flags   Refs     Use  If    PMTU  Exp  Groups

Route Tree for Protocol Family 2 (Internet):
default          myhost1.mynet.co  UGc       0        0  tr0     -   -
10.1.34.3        myhost1.mynet.co  UGHW      4     7211  tr0     -   -
ns.mynet.com     myhost1.mynet.co  UGHW      1   406419  tr0  1492   -
10.7.3/21        myhost.mynet.com  U         3      183  tr0     -   -
127/8            localhost        U         0   397412   lo0     -   -
192.168.1/24     myhost            U         0      156  en0     -   -

Route Tree for Protocol Family 24 (Internet v6):
::1              ::1               UH        0        0  lo0 16896   -
```

Displaying the Routing Table

To add a new route:

```
$ route add -net 172.16.1.0 -netmask 255.255.255.0 192.168.1.1
192.168.1.1 net 172.16.1.0: gateway 192.168.1.1
```

Adding a Route to a Network

```
$ route add -host 172.16.2.7 -netmask 255.255.255.0 10.7.214.210
192.168.1.1 net 172.16.2.7: gateway 10.7.214.210
```

Adding a Route to a Host

```
$ route add default 10.7.214.210
192.168.1.1 net 255.255.255.255: gateway 10.7.214.210
```

Adding a Default Route

The commands that can be used are add, flush (reset all dynamically learned routes), change and monitor

The final entry of a route command is the gateway to send the packet to / out of.

arp

The IP addresses are used at layer-3 but not at layers before then. When the packet is sent as a datagram over the local segment then it needs to be converted into a different address which for ethernet or token ring is known as a MAC (Media Access Control) address. The technique used to translate these addresses is known as the Address Resolution Protocol or arp.

The arp command can then be used to display the contents of the hosts arp table. The entries are typically flushed out after 20 minutes.

Tracing TCP/IP Flows

There are two tools for tracing the TCP/IP connections. One is tcpdump and the other is iptrace.

TCPdump is available for most UNIX operating systems. It provides information about the data passing over a network interface. It is only recommended for those familiar with TCP/IP and the associated protocols. As these monitor network traffic for obvious reasons they can only be run by root.

By default tcpdump will output to the screen however only after it's 4k buffer has been filled up. The -I option is used to output the data as soon as it is captured instead of relying on the buffer, you will probably want to use the -I switch on all your commands. As this is realtime the command cannot be piped through the tee command. However to get a similar effect you could have tcpdump running with it's output going to a file and then issue a tail -f against the file to monitor lines as they are added.

Some example commands are:

```
tcpdump -I -i tr0 ip host sourceaddr
```
Monitor on tr0 but only for pakets to or from the host sourcaddr.

```
tcpdump -I -N -i en0
```
Monitor all traffic on en0 (show short names for hostnames -N)

```
tcpdump -i tr0 ip proto icmp
```
Only capture ICMP traffic.

```
# tcpdump -N -I -i en0
tcpdump: listening on en0
19:13:19.499554858 arp who-has myhost1 tell 192.168.1.3
19:13:19.499631178 arp reply myhost1 is-at 8:0:5a:fc:e9:38
19:13:19.500281280 192.168.1.3.blackjack > myhost1.telnet: S 2112409258:2112409258(0) win
32120 <mss 1460,sackOK,timestamp 169431789 3523215360,nop,wscale 0> (DF)
19:13:19.500660120 myhost1.telnet > 192.168.1.3.blackjack: S 387771002:387771002(0) ack
2112409259 win 16060 <mss 1460>
19:13:19.501632843 192.168.1.3.blackjack > myhost1.telnet: . ack 1 win 32120 (DF)
19:13:19.506623836 192.168.1.3.blackjack > myhost1.telnet: P 1:25(24) ack 1 win 32120 (DF)
19:13:19.704976964 myhost1.telnet > 192.168.1.3.blackjack: . ack 25 win 16036
19:13:19.894932410 myhost1.telnet > 192.168.1.3.blackjack: P 1:7(6) ack 25 win 16036
19:13:19.895772052 192.168.1.3.blackjack > myhost1.telnet: . ack 7 win 32120 (DF)
19:13:19.927455493 myhost1.telnet > 192.168.1.3.blackjack: P 7:37(30) ack 25 win 16060
19:13:19.936210835 192.168.1.3.blackjack > myhost1.telnet: P 25:45(20) ack 37 win 32120 (DF)
19:13:20.035006248 myhost1.telnet > 192.168.1.3.blackjack: P 37:43(6) ack 45 win 16060
```

A sample tcpdump capture

The above sample shows the start of a telnet session establishment.

An AIX specific command is iptrace. There are two separate programs iptrace and ipreport. To run the trace iptrace is used and then ipreport run against the output file to generate a readable file.

The trace is run as:

iptrace -s sourceaddr -b [-d destaddr] filename
kill PID

or

startsrc -s iptrace -a “-s sourceaddr -b [-d destaddr] filename”
stopsrc -s iptrace

Normally the -n and -s options are used to provide packet numbering and to add protocol display respectively.

Then formatting is done using:

ipreport -rns filename > filename.formatted

```
# iptrace -b -i en0 -d 192.168.1.3 -s 192.168.1.1 /tmp/telnet.trace
# ps -ef | grep iptrace
    root 18742     1   0 18:25:06      -  0:00 iptrace -b -i en0 -d 192.168.1.3
-s 192.168.1.1 /tmp/telnet.trace
    root 19612 18118   2 18:25:31  pts/5  0:00 grep iptrace
# kill -9 18742
# ipreport /tmp/telnet.trace >telnet.out
# cat telnet.out


====( 74 bytes received on interface en0 )==== 18:22:11.028314491
ETHERNET packet : [ 00:40:05:5a:00:1f -> 08:00:5a:fc:e9:38 ]  type 800  (IP)
IP header breakdown:
        < SRC =     192.168.1.3 >
        < DST =     192.168.1.1 >  (myhost1)
        ip_v=4, ip_hl=20, ip_tos=0, ip_len=60, ip_id=104, ip_off=0DF
        ip_ttl=64, ip_sum=b6ff, ip_p = 6 (TCP)
TCP header breakdown:
        <source port=1026, destination port=23(telnet) >
        th_seq=9ec34f18, th_ack=0
        th_off=10, flags<SYN>
        th_win=32120, th_sum=7e87, th_urp=0
                mss 1460
                opt-4:          mss 2585 [len 8]
                opt-83:00000000     bd6a0000 00000103 03000000 00000000
|.j..............|
00000010     00000000 00000000 00000000 00000000     |................|
********
000000b0     00000000 00000000 000000                |...........     |

====( 60 bytes transmitted on interface en0 )==== 18:22:11.028350131
ETHERNET packet : [ 08:00:5a:fc:e9:38 -> 00:40:05:5a:00:1f ]  type 800  (IP)
IP header breakdown:
        < SRC =     192.168.1.1 >  (myhost1)
        < DST =     192.168.1.3 >
        ip_v=4, ip_hl=20, ip_tos=0, ip_len=44, ip_id=55308, ip_off=0
        ip_ttl=60, ip_sum=236b, ip_p = 6 (TCP)
TCP header breakdown:
        <source port=23(telnet), destination port=1026 >
        th_seq=371c239, th_ack=9ec34f19
        th_off=6, flags<SYN | ACK>
        th_win=16060, th_sum=1e65, th_urp=0
                mss 1460
```

A Sample iptrace Capture

The above output contains only the first two packets of the communication however gives much more information than tcpdump did.

Networking Programs

There are a number of programs that provide a service over the network connection. The ones that I describe in this section are considered to be the core programs that are used to provide a remote logon and / or transfer files over the network.

telnet

The telnet command allows a remote login to another machine. The remote machine must be running the telnetd daemon and be willing to accept incoming connections. The telnet program provides a text based logon effectively providing terminal access but running over the network instead of a serial cable. The command is run by issuing telnet followed by the IP address or hostname of the remote machine. The default port is 23 but if you want to connect to another port then this would be added as a second parameter.

```
telnet remote.mynet.com
```

To escape from the terminal connection into telnet command mode then the default escape key combination is CTRL-] . If you are already in a telnet connection then the login will show what alternative key combination has been assigned.

AIX also has a tn command that provides the same function, although the default escape key is normally set to CTRL-t .

ftp

The ftp (file transfer program) program is used to transfer files between computers. Normally the ftp program requires the user to enter the login username and password when first connecting to the remote machine. It does however allow auto login using macro definitions. Once connected the get command is used to transfer a file from the remote system to the local system and the put command to transfer a file to the remote system. There are many other commands that can be used a few of which are shown below:

On the remote system:

ascii	Change to ascii / text mode for transferring text files (default value)
binary	Change to binary / image mode (use for transfering binary files)
bye / quit	Close the connection
cd	Change directory
delete	Deletes a file
dir / ls	List directory content
get	Get a file from the remote system
put	Put a file on the remote system
mget	Multiple get
mkdir	Create a directory
mput	Multiple put
rmdir	Remove directory

On the local system

!ls	List directory contents
lcd	Change directory

As well as the bye & quit commands CTRL-d can also be used to exit.

rexec

The rexec command allows a command to be run on a remote system. The command will either use the .netrc file (see later) for login username and password or they will be prompted for. rexec cannot be used for any full screen applications.

rcp / rsh / rlogin

These commands are sometimes referred to as the 'R' commands. The rcp command provides the ability to copy files between systems, the rsh command to run a program on the remote system and the rlogin command to provide a login shell. The commands can use normal login methods or can be authorised using the /etc/hosts.equiv or $HOME/.rhosts file (see later).

finger

The finger command interrogates to gain information about a user on a system. It provides information such as whether the user is logged in, and whether they have read their mail or not. For security reasons however this is normally turned off.

Mail

There are numerous email clients available for AIX. The standard client is called mail and information on it's use is contained in the man pages.

NFS

NFS or Network File System allows a directory to be mounted from a remote system as though it was a local device. This is explained in more detail later.

ssh / scp

Secure Shell and Secure Copy are additional commands that can be used to replace rsh and rcp with a more secure alternative. Whilst not included in AIX this is available free of charge from http://www.openssh.com/
These provide encrypted communications using PKI technology. It is recommended that ssh be used instead of traditional applications including telnet, wherever security is a major concern.

News

There are numerous news clients that allow you to participate in Usernet groups.

WWW Browsers

WWW browsers are available for AIX including Netscape (most popular X-Windows browser) and Lynx (Text based browser).

Networking Services (Server Applications)

There are a lot of network services available for AIX, either as standard or freely available over the Internet. Almost every possible network service is available in some form or another.

Traditionally there would be a daemon running for each service that was offered, however in modern UNIX operating systems, including AIX, a lot of the common services have been incorporated into the inetd daemon. The services not included in the inetd service are normally set to start automatically by including them in the /etc/rc.tcpip file.

Some of the standard services are:

syslogd	Logs Error Messages
portmap	Port Lookup Facility (traditionally for NFS but also used by other services including CDE).
inetd	Inetd Super Daemon
named	Domain Nameserver
lpd	Print Server
sendmail	Mail
timed	Time Daemon
rwhod	Remote Users
snmpd	Simple Network Management Protocol
dhcpcd	DHCP Daemons

Some of these are set to start automatically whereas others will need to be uncommented if they are to be used.

The inetd services are configured in the file /etc/inetd.conf . Some of these are shown below:

```
## service  socket  protocol  wait/  user    server    server program
##  name     type            nowait         program     arguments
##
ftp     stream  tcp6    nowait  root    /usr/sbin/ftpd        ftpd
telnet  stream  tcp6    nowait  root    /usr/sbin/telnetd      telnetd -a
shell   stream  tcp6    nowait  root    /usr/sbin/rshd        rshd
kshell  stream  tcp     nowait  root    /usr/sbin/krshd       krshd
login   stream  tcp6    nowait  root    /usr/sbin/rlogind     rlogind
klogin  stream  tcp     nowait  root    /usr/sbin/krlogind    krlogind
exec    stream  tcp6    nowait  root    /usr/sbin/rexecd      rexecd
#comsat dgram   udp     wait    root    /usr/sbin/comsat      comsat
uucp   stream  tcp     nowait  root    /usr/sbin/uucpd       uucpd
#bootps dgram   udp     wait    root    /usr/sbin/bootpd      bootpd
/etc/bootp
tab
##
## Finger, systat and netstat give out user information which may be
## valuable to potential "system crackers."  Many sites choose to disable
## some or all of these services to improve security.
##
#finger stream  tcp     nowait  nobody  /usr/sbin/fingerd    fingerd
#systat stream  tcp     nowait  nobody  /usr/bin/ps          ps -ef
#netstat stream tcp     nowait  nobody  /usr/bin/netstat     netstat -f
inet
#
```

Allowing Multiple Logins

An important point to note is that where a machine will have multiple simultaneous connections the number of available licenses must allow this. By default it will only allow 2 simultaneous logins which is often far too restrictive.

If you receive the error message “3004-All Available Login Sessions are in use” then this is almost certainly the cause.

The current licenses can be displayed using

```
lslicense
```

To increase the number of licenses enter

```
chlicense -u xx
```

where xx is the number of licenses require.

Anonymous ftp

Anonymous ftp allows users to login using ftp without having to provide a password. To use anonymous ftp the system performs a chroot on the login. What this does is to move the root directory to one further down the directory tree e.g. /home/ftp . As the login requires access to some of the directories that would otherwise be hidden a number of directories need to be setup. These are created by running the script /usr/samples/tcpip/anon.ftp when the service is first configured.

Configuring Domain Name Servers

The configuration of Domain Name Servers and DNS Clients is controlled by the /etc/named and /etc/resolv.conf files.

Setting up a Primary Name Server

Setting up the Primary Name Server is the most involved configuration. The following steps need to be taken to setup the server.

1. Create named control file
2. Create name zone file
3. Create IP zone file(s)
4. Create local IP zone file
5. Create cache file
6. Create /etc/resolv.conf
7. start “named” daemon

The /etc/named.boot file is read by the named daemon when it starts. It specifies the location of the files used to create the initial name server database.

```
directory    /etc
primary      myzone.mynet.com          named.myzone
primary      2.168.192.in-addr.arpa    named.revip2
primary      3.168.192.in-addr.arpa    named.revip3
primary      0.0.127.in-addr.arpa      named.local
cache        .                         named.ca
```

The directory statement indicates the directory where the files are stored.

The first primary entry in this examples shows the domain for which the server is a primary for.

The next two entries are for reverse address lookup. The octets of the ip address are reversed, to show the least significant first (just like a domain name) and is always appended with .in-addr.arpa. There should be a file for each physical network that exists.

The cache entry applies to the . domain meaning any other domains not already covered.

There are some awk scripts provided to start off the construction of the domain files. The scripts are

```
/usr/samples/tcpip/hosts.awk
```

and

```
/usr/samples/tcpip/addrs.awk
```

Then the files should be completed by adding more entries. The commands are run as:

```
/usr/samples/tcpip/hosts.awk /etc/hosts > /etc/named.myzone
/usr/samples/tcpip/addrs.awk /etc/hosts > /etc/named.revip2
/usr/samples/tcpip/addrs.awk /etc/hosts > /etc/named.revip3
```

A name zone file is shown below:

```
; NAME   TTL   CLASS   TYPE      RDATA
;
; setting default domain to “myzone.mynet.com”
;
@             IN    SOA   sys1.myzone.mynet.com. root.sys1.myzone.mynet.com.  (
                                20001130            ; Serial
                                3600         ; Refresh
                                300          ; Retry
                                3600000      ; Expire
                                86400  )     ; Minimum TTL
               IN   NS     sys1
               IN   NS     sys6
sys1           IN   A      192.168.5.1
sys2           IN   A      192.168.5.2
sys3           IN   A      192.168.5.3
localhost      IN   A      127.0.0.1
loopback       IN   CNAME  localhost
```

The following characters have special meanings.

; indicates a comment
. Is used to indicate the current domain for the name field
@ Is used to indicate current origin for the name field
( ) Parentheses allows data to be continued across more than one line

The **name** field can specify a domain, a zone of authority, the name of a host or the alias of a host. It must begin in column1 if left blank then it is set to the previous entry.

The **TTL** field is time to live and specifies the number of seconds before it expires. 9999999 would be used to indicate no timeout. If not specified then it defaults to the SOA entry.

The **class** field is the address class of the record, which can be either IN for Internet or any for all other address classes.

The **type** field is the type of resource.

SOA - Start of Authority
NS - Name Server
A - Address
HINFO - Host Information
CNAME - Canonical
MX - Mail Exchange

The **rdata** field contains specifics for the particular record type.

The following entries are used in the Start of Authority file:
Serial Version number of the data file. Every time there is a change to the file then this should be incremented and will cause the updates to be reflected in any secondary name servers. The number can be given as an integer (e.g. the date in reverse form yyyymmdd) or can include a dot. If a dot is used then the number to the left of the dot is multiplied by 10,000 before being added to the number on the right.
Refresh Time interval that secondary checks for date change in seconds.
Retry Time interval secondary waits after failure to reach primary for a refresh
Expire Upper time limit used by secondary to flush data after continued failure to contact the primary. 3600000 seconds is approximately 42 days.
Minimum is the minimum time to live. This would override any individual entries that were lower.

The IP Zone file looks like the one below:

```
; NAME   TTL   CLASS   TYPE      RDATA
;
@       9999999  IN   SOA   sys1.myzone.mynet.com. root.sys1.myzone.mynet.com.  (
                                 20001130           ; Serial
                                 3600         ; Refresh
                                 300          ; Retry
                                 3600000      ; Expire
                                 86400  )     ; Minimum TTL
         999999  IN    NS    sys1.myzone.mynet.com.
         999999  IN    NS    sys3.myzone.mynet.com.
1        999999  IN    PTR   sys1.myzone.mynet.com.
2        999999  IN    PTR   sys2.myzone.mynet.com.
3        999999  IN    PTR   sys3.myzone.mynet.com.
```

This file is required for reverse address lookups. The first 3 octets of the address are listed in the named.boot file and this file is used to provide a translation of the final octet to it's hostname. Many of the fields have the same meaning as those used in the name zone file.

The valid resource types are
- SOA - Start or Authority
- NS - Name Server
- PTR - Domain name pointer

There should be a reverse host data file per class C network.

The Local IP Zone File is shown below:

```
@              IN   SOA   sys1.myzone.mynet.com. root.sys1.myzone.mynet.com.  (
                                20001130     ; Serial
                                10800        ; Refresh
                                3600         ; Retry
                                604800       ; Expire
                                86400  )     ; Minimum TTL
               IN     NS    sys1.myzone.mynet.com.
1              IN     PTR   localhost.
```

Whilst the Local IP Zone File is needed by the system is doesn't have much use except to provide a localhost entry. The SOA entry is not required in the local IP Zone file although it is in the Zone file and the IP zone file.

The next name server file required by the primary name server is the cache file:

```
.                   9999999  IN     NS    dns1.mynet.com.
dns1.mynet.com.     9999999  IN     A     192.168.2.1
```

If a name cannot be resolved locally then it will contact each name server listed in the file until either the name is resolved or all possibilities have been tried. If the network is connected to the Internet then the name servers will usually be the real Internet root servers. The cache file with the public Internet root servers is available using anonymous ftp from ftp.rs.internic.net

The final file that is needed is the /etc/resolv.conf file. For a name server this needs to be empty and can be created by using the following command.

```
cp /dev/null /etc/resolv.conf
```

Once the files have been edited then the name server can be started.

The hostname should be changed using

```
smit hostname
```

this needs to reflect the fact that the machine is in a domain environment, this will not take effect until after a reboot.

To have the name server start at reboot the named entry should be uncommented from the /etc/rc.tcpip file. Then to start the daemon the command
```
startsrc -s named
```

The /etc/rc.tcpip file and the starting of the daemon can be performed in a single step using
```
smit stnamed
```

The daemons can be stopped and refreshed using stopsrc and refresh respectively.

The active database can be dumped by issuing a kill -2 against the PID. This file can be used to ensure that the zone files are correct. The file is stored as /var/tmp/named_dump.db .

The process to add a host to the domain is as follows:

Update the name zone file
- add host entry A record
- add any optional records (e.g. CNAME for alias entries)
- increase serial value in SOA record

Update IP zone file
- add IP address entry PTR record for each interface
- increase the serial value in SOA record

Refresh named

Setting Up a Secondary Name Server

Setting up a Secondary Name Server involves less stages than the primary name server. The following steps need to be taken to setup the server.

1. Create named control file
2. Create local IP zone file
3. Create cache file
4. Create /etc/resolv.conf
5. start “named” daemon

The /etc/named.boot file is read by the named daemon when it starts. It specifies the location of the files used by the name server.

```
directory    /etc
secondary    myzone.mynet.com          192.168.5.1         named.myzone.bak
secondary    2.168.192.in-addr.arpa    192.168.5.1         named.revip2.bak
secondary    3.168.192.in-addr.arpa    192.168.5.1         named.revip3.bak
primary      0.0.127.in-addr.arpa                          named.local
cache        .                                             named.ca
```

This file is similar to that configured for the primary name server with the following exceptions.

The 3rd field refers to the IP address of the primary name server from which the entries are obtained. The files are appended with a .bak to indicate that these are not the primary files for the domain. The backup files are not actually needed as the entries can be downloaded into memory, however if the primary server is not available after a reboot then it will use the entries from the backup files. The local file is still written as primary as the server still acts as a server for it's local domain.

Whilst the zone and reverse IP files are not required the local IP Zone file should be.

```
@              IN   SOA   sys2.myzone.mynet.com. root.sys2.myzone.mynet.com.  (
                                20001130     ; Serial
                                10800        ; Refresh
                                3600         ; Retry
                                604800       ; Expire
                                86400  )     ; Minimum TTL
               IN     NS    sys2.myzone.mynet.com.
1              IN     PTR   localhost.
```

This is identical to the primary name server except that it relates to itself in the file.

The Cache File is the same as the one used for the primary name server.

```
.                   9999999  IN     NS    dns1.mynet.com.
dns1.mynet.com.     9999999  IN     A     192.168.2.1
```

Then the /etc/resolv.conf file should be created. For a name server this needs to be empty and can be created by using the following command.

```
cp /dev/null /etc/resolv.conf
```

Once the files have been edited then the name server can be started.

The hostname should be changed using

```
smit hostname
```

this needs to reflect the fact that the machine is in a domain environment, this will not take effect until after a reboot.

The named entry should be uncommented from the /etc/rc.tcpip file. Then to start the daemon the command

```
startsrc -s named
```

The /etc/rc.tcpip file and the starting of the daemon can be performed in a single step using `smit stnamed`

Caching-Only Name Server

As a caching-only name server has no authority for any domains. It is setup the same as a secondary name server with the following exceptions.

The /etc/named.boot file does not have any direct entries as a secondary name server:

```
directory    /etc
primary      0.0.127.in-addr.arpa                                 named.local
cache        .                                                    named.ca
```

Forwarder Name Server

A forwarder name server handles off-site DNS queries for other servers in a network. If one of the local DNS servers could not answer the query then it passes it on to the forwarder name server. The forwarder server then queries the root and other authoritative servers passing the reply to the local server.

The forwarder name server is useful for networks that are isolated from the Internet, such as those hidden behind a firewall or those with unregistered IP addresses. The forwarder name server is also useful where the network is connected to the Internet via dial.

The forwarder server is not altered to make it perform forwarding, but rather it is the local name server that is told to contact a forwarding name server.

The following named.boot file shows the changes that are necessary.

```
directory    /etc
forwarders   169.35.21.1 169.35.21.2
options      forward-only
primary      myzone.mynet.com          named.myzone
primary      2.168.192.in-addr.arpa    named.revip2
primary      3.168.192.in-addr.arpa    named.revip3
primary      0.0.127.in-addr.arpa      named.local
cache        .                         named.ca
```

Setting Up a Client

There are only two steps involved in setting up a client to use a domain name server.

1. Change the host name to a fully qualified name
2. Create /etc/resolv.conf (with entries)

The first stage is to change the host name to be fully qualified. This is done using

```
smit hostname
```

For the name servers the resolv.conf file was empty, with the client we use the resolv.conf file to indicate the domain name server that should be used to obtain address resolution. The following shows an example file with two entries:

```
domain       myzone.mynet.com
nameserver   192.168.5.1
nameserver   192.168.5.1
```

This could be created by editing the file using a text editor or by using

```
smit namerslv
```

The file can have between one and three entries which are queried in order until a response is received.
The /etc/hosts file should have an entry for itself (any others are optional).

Delegated Servers

In small to medium size organisations a primary and seconary name server is adequate. However in larger organisations it is often better to split the name servers into smaller easier to manage domains. In our example we have mynet.com which is the top level domain for our fictitious organisation, we have already split off one subdomain called myzone, we might also have another called anotherzone and more zones. Typically these might be different departments or geographical locations (e.g. countries).

The following steps are required to split a domain into subdomains.

Setup the primary and secondary name servers for the subdomain.

Add NS and glue records to point to the new name servers.

The entry to point a subdomain at a different server would look like the following (assuming we are on the server for mynet.com);

```
myzone              IN     NS     dns1.myzone
                    IN     NS     dns2.myzone
dns1.myzone         IN     A      192.168.5.1
dns2.myzone         IN     A      192.168.5.2
```

The following entries would be required to delegate a reverse lookup file.

```
124.12              IN     NS     dns1.myzone.mynet.com
                    IN     NS     dns2.myzone.mynet.com
```

Glue records are not needed here as the dnsservers are not in the domain which is being delegated.

Network File System (NFS)

What is NFS

NFS allows a directory structure stored on a remote machine to be mounted locally as though it was any other file system. The main advantage is that it allows files to be stored and accessed on a remote system, using a technique that is totally transparent to both the user and any applications.

NFS was developed by Sun Micro Systems in 1984 and has been implemented on a whole range of different systems. For MS Windows users NFS cannot be used directly, although is available in the form of a number of third party commercial products.

As mentioned earlier NFS is one of those areas that uses all seven separate layers of the 7-layer model. This is shown for NFS below:

Application	7	NFS
Presentation	6	XDR
Session	5	RPC Library
Transport	4	UDP or TCP
Network	3	IP
Data Link	2	eg. Ethernet
Physical	1	eg. Ethernet

NFS and the 7-layer model

Starting with layer 4 the following explanation explains information on the upper layers.

UDP / TCP NFS traditionally used UDP only. However more recently TCP was added as a alternative. UDP is more suitable for reliable networks as it has less overhead however where the network connection is less reliable then TCP provides better end-to-end connectivity.

RPC (Remote Procedure Call) Remote procedure calls are used to allow processes, execute procedure calls on another system as though it was local. This makes cross platform implementation of NFS easier to implement and provides inter-process communication.

XDR (eXternal Data Representation) This is used to describe protocols in a system independent way. The main use of this is in allowing NFS to be implemented on different systems with very different data storage structures. For example it allows UNIX systems to communicate with mainframe and MS Windows systems with completely different directory structures.

NFS The NFS layer provides the software required to allow the filesystems to be remoted and / or exported.

NFS is a stateless protocol. It does not remember transactions and does not allow for system recovery procedures or notification of a server failure.

To mount a remote directory the mount command is used. This is the same command used to mount any journaled file systems or removable media such as CD’s and floppy disks. The difference is that the node has to be included and that the file system type is NFS. Any mounts can be shown by using the mount command without any options.

NFS Daemons

There are a number of daemons that need to run on both the client and the server.

Client

biod	Improves NFS performance by filling and emptying the buffer cache. By default 6 biod daemons are started however this can be changed to improve performance. These are started in the /etc/rc.nfs file.
rpc.statd	Allows the remote procedure calls
rpc.lockd	Handles File locking

Server

portmap	Provides a standard way of looking up the port for a certain application. The application registers with portmap, then portmap listens on the appropriate port. When a client communicates on the NFS port, portmap replies with the real port number of NFS. Portmap is included in the /etc/rc.tcpip file and must be started before inetd and the RPC servers so that it can accept the registrations.
rpc.mountd	Accepts a mount request from the client and allows the export if authorised.
nfsd	A server daemon that handles the client requests for file system operations. Each daemon accepts one request at a time, however once it has passed the request on to the kernel it is free to accept a new request. Bu default 8 nfsd daemons are started, however if the server is a busy nfs server then a value 50 to 100 might be more appropriate. These are started from /etc/rc.nfs and are under the control of SRC.
rpc.statd	Allows Remote procedure calls
rpc.lockd	Implements file locking

Authorising NFS Access and Protecting Shared Files

There are a number of different ways of authorising NFS connections. However NFS has long been a target for security breaches due to the trust relationships that are used. For this reason some organisations insist that NFS is turned off however if you are willing to accept the security risks, it provides a very convenient way of sharing files over a network. If you consider NFS to be too risky to run then you should see the security section for details of how to disable some of the daemons.

The protecting of shared files is another issue on top of authorisations. It’s possible that a number of different NFS clients have write access to a certain file. If this is not managed correctly then the result is often corrupted files or lost updates. This can be avoided by either mounting the file system as read only to protect the files from update or by using the lock manager (and system calls fcntl() or lockf()). This uses the rpc.lockd and rpc.statd daemons to provide advisory locking (note forced locking is not supported).

To authorise a connection then the UNIX UID's and GID's are used. Note this is based on the numeric number associated with the username and not the username itself. You should therefore ensure that in an NFS environment all the UID's and GID's relate to the same users and groups across all the servers. ACLs are used to as well as the standard UNIX permissions however this must be enabled before it can be used. If root accesses an NFS filesystem then it will be given privileges of the nobody userid. The nobody user does not own any files and so root is only given access as set in the other field.

In large NFS environment NIS can be used to help maintain the same userid numbers across a number of systems. This is described later.

There is also a secure NFS option that ensures that all the UID and GID exchanges are all encrypted.

Configuring the NFS Server

To configure the NFS Server first TCP/IP must be installed and configured.
The NFS Server code also needs to be installed from the AIX disks.

Ensure that portmap is running. This should be configured in /etc/rc.tcpip.
Issue

```
ps -ef | grep portmap
```

To start the NFS daemon (nfsd) choose the option from smit to start nfs.
This is in Communications / NFS
Each fs then has to be explicitly exported using the export a fs option
This will update the /etc/exports file.

The following files are used in the configuration of the server (these are normally updated using SMIT).

/etc/exports
This file has the files systems that are to be exported. The file can be updated using

```
smit mknfsexp
```

The /etc/exports file is also looked at by the /etc/rc.nfs file during system startup. If /etc/exports is detected then the server daemons are started. The file contains a list of all directories to be exported along with options that allow the export to be read-only and to restrict what systems or users are allowed to access it.

```
/usr/local
/usr/man     -ro
/home/test   -access=host1:host2
```

Example of a /etc/exports file

The exportfs command is used to export the directories listed in /etc/exports and copies the entries into /etc/xtab. The /etc/xtab file holds a list of all the currently exported directories. Running the exportfs command on it's own then it will show all entries in the /etc/xtab . The -a option rereads the exports file. If SMIT is used to make the update then this will be run automatically.

/etc/rc.nfs
The /etc/rc.nfs file handles the starting of the daemons. This is automatically updated when NFS is enabled using SMIT. This is automatically run on a reboot or can be started by executing the file. To stop NFS then either of the following commands can be run.

```
/etc/nfs.clean
```
or
```
stopsrc -g nfs
```

The rc.nfs file needs to be defined in the /etc/inittab file to be run automatically on startup. The /etc/inittab file can be updated by running:

```
mkitab "rcnfs:2:wait:/etc/rc.nfs >/dev/console 2>&1"
```

The /etc/rc.nfs entry must be after the /etc/rc.tcpip entry.

Configuring the NFS Client

To configure the NFS Client first TCP/IP must be installed and configured.
The NFS Client code also needs to be installed from the AIX disks.

First the mount point must exist before a filesystem may be mounted. The mkdir command should be used to create the mount point. A mount point is required for every filesystem that will be mounted.

The easiest way to start the NFS client daemons is to use smit mknfs. This allows NFS to start immediately, at system restart or both.
The alternative is to uncomment the daemons in /etc/rc.tcpip and /etc/rc.nfs . The daemons that need to be started are: portmap; biod; rpc.statd and rpc.lockd.

To mount a filesystem then the following command is used:

```
mount rmthost1:/home/exportdir  /home/mntpoint
```

rmthost1 is the remote host
/home/exportdir is the directory exported from the server
/home/mntpoint is the local mount point created earlier with the mkdir command. If the remote file system is not designated as a nfs file system in the /etc/vfs file then the -v option would need to be used.

```
mount -v nfs rmthost1:/home/exportdir  /home/mntpoint
```

However manual mounts are not suitable for file systems that are required by client systems for normal operation.

The alternative is to have the NFS mounts automatically done at system startup. Here the entry is put into the /etc/filesystems file to perform the mount when all the rest of the mounts are performed. However if a remote system is not available the local system will hang for a period of time when the system is started.

Predefined mounts can be added by using smit mknfsmnt or by manually updating the /etc/filesystems file.

```
/home/mntpoint
      dev = “/home/exportdir”
      mount = true
      vfs = nfs
      nodename = rmthost1
      option = bg, hard, intr
      account = false
```

Example entry for /etc/filesystems to add NFS entry

The following attributes are used:
dev = path of the remote exported directory
vfs = specifies that it is an nfs filesystem
nodename = the remote system
mount = whether the file system is mounted automatically. It can be true, false or automatic - if true then the bg option should be set or a failed mount would hang the system startup.
options = various options see the table below
account = whether the file system is processed by the accounting system.

Option	Function	Default
bg	Mount in background if first attempt fails	
fg	All mount attempts in foreground	Yes
soft	Repeated RPC calls eventually timeout	
hard	RPC calls retry indefinitely	Yes
intr	Allows keyboard interupts to halt hard attempts	
retry=#	Number of times to retry the mount	1000
retrans=#	No. of times to repeat RPC request before timeout error on soft mounts	3
timeo	Varies RPC timeout period (tenths of second)	7
ro	Read-only	
rw	Read-write	Yes
ver=<version>	Choose NFS protocol version (2 or 3)	
prot=<pro>	Choose transprt protocol (TCP or UDP)	UDP

NFS mount options

If the filesystems are defined so that they aren't automatically mounted then they can be mounted using the mount command followed by the mount point. Or if the type option is used then a number of file systems can be mounted simultaneously. For example if there are two entries both with type = filesystem1 .

```
mount -t filesystem1
```

The mount all command can also be used to remount the filesystems in the /etc/filesystems file however this will only mount those with mount = true defined.

The file systems can be unmounted again using the umount (or unmount) command. If the filesystem is in use then the fuser command can be used to identify what processes may be using the filesystem. These would have to be stopped (normally or using the kill command) before the umount command can be run.

Managing the NFS Daemons

The NFS Daemons are under the control of SRC. This allows the daemons to be started, stopped and listed using startsrc, stopsrc and lssrc respectively. The following table shows the files and their SRC subsystems.

NFS Daemons		
File Path	**Subsystem Name**	**Group Name**
/usr/sbin/nfsd	nfsd	nfs
/usr/sbin/biod	biod	nfs
/usr/sbin/rpc.lockd	rpc.lockd	nfs
/usr/sbin/rpc.statd	rpc.statd	nfs
/usr/sbin/rpc.mountd	rpc.mountd	nfs
/usr/sbin/portmap	portmap	portmap

NFS Daemons and Subsystems

Some example commands are:

Stopping and starting lockd
```
stopsrc -s lockd
startsrc -s lockd
```

Stopping all NFS Daemons
```
stopsrc -g nfs
```

Listing status of NFS Daemons
```
lssrc -g nfs
```

There are some other commands that can be used to manage NFS.

`chnfs -n` *`nfsdaemons`* `-b` *`biod`*

This allows the number of nfsdaemons and biod daemons that are running.

```
mknfs
```

This sets up nfs by updating the /etc/inittab to start /etc/rc.nfs. It also starts portmap.

```
rmnfs
```

This stops NFS from running and prevents it from starting again at reboot.

This could also be changed using smit. In particular the fastpach smit chnfs .

NFS Commands

There a number of commands that can be used with NFS. For the daemons to be available from a remote system requires them to be uncommented in /etc/inetd.conf and registered with the portmap. And is normally used on both servers and clients. If SMIT is used to start NFS then these will be automatically setup. The table below shows the different daemons associated with the NFS commands.

Command	Description	Daemon
showmount	Displays what clients have mounted	rpc.mountd
rpcinfo	Displays what portmap has listed	portmap
on	Remote command execution	rexd
rup	Displays host uptime information	rstatd
rusers	Shows remote users	rusersd
rwall	Sends message to network users	rwalld
spray	Sends a stream of packets	sprayd
nfsstat	Displays status of NFS and RPC calls	

NFS Command Daemons

Automounter

The automounter provides a way for NFS remote mounts to be transparently mounted and unmounted at will. This is traditionally used for NFS however can be used for different file systems including JFS or CDRFS. It can therefore be used to hide the mount command away from a user, automatically mounting the CD when required.

Whenever a directory is accessed that is monitored by the automounter, then an NFS mount is automatically run. Any directories that are required that don't already exist are created. After a period of inactivity (default is 5 minutes) then it will attempt to unmount the directory.

The automountd daemon provides the automount facility. The automount command can be used to control the automounter.

The automounter can reduce system administration of /etc/filesystems and prevents the client from hanging if a NFS mount can't be performed. Also where a mount is required for read-only access the automounter can load balance across multiple servers.

Indirect and Direct Map Files

Indirect Map Files are used for higher level directories such as /home . The subdirectories may be distributed on several servers. The directory is referenced on the command line when starting automount or in a master map under NIS.

The automount map file must be in the /etc directory and should be called auto . something. Typically /etc/auto.pub is used for public mounts.

```
userdir1      host2:/users1
userdir2      host3:/users2
misc          host5:/other
```

Example /etc/auto.pub File

Direct maps are useful when a directory cannot be used solely for automount, such as /usr . For example if a indirect map was used for /usr/local then it would cover up /usr/bin . Normally the file /etc/auto.direct is used.

```
/usr/local    host3:/usr/export/local
/usr/man      host4:/usr/man
```

Example /etc/auto.direct file

These files cannot be created by SMIT.

automount command

The automount command is used to control the operation of the automounter daemon. Below is an example of a command to specify an indirect map and a direct map.

```
automount -m /mountdir /etc/auto.pub

automount -m /- /etc/auto.direct
```

Only root can use the automount command, not just a member of the system group as in the normal mount commands.
The -m option is an instruction not to use NIS. If it is not specified then the automount will look for a NIS map.

For the direct map we have to tell the automount to use the entries in the direct map. Therefore /- is used instead of the mountpoint.

The directories available for use by the automounter can be shown by running the mount command. When the automounter actually mounts a file system then an additional entry will be shown indicating the active mount.

The automount timeout can be set by using the tl and tw options.

```
automount -m -tl 300 -tw 60 /mountdir /etc/auto.pub
```

The -tl sets the number of seconds until the automounter attempts to unmount an inactive file system. The default is 300 seconds (5 minutes). If the file system timesout only for a request to ask for it to be remounted again then this should be increased.

The -tw switch specifies the number of seconds to wait before retrying the unmount if the first one was unsuccessfully. The default value is 60 seconds.

The automounter can be stopped by running

```
stopsrc -s automountd
```

it is possible to use kill however a kill -9 should be avoided as it could cause a hang.

Network Information Service (NIS)

NIS was developed to simplify the task of administrating a number of machines over a network. In particular was the requirement to maintain copies of common files (e.g. password, group and host) across different systems. When a change is made then this needs to be propagated across all the different systems which creates the risk of editing errors or of changes on one system not being reflected across the others.

NIS addresses this by replacing copies of common configuration files with a single data map for each file which is then located on a central server. This provides a consistent view of configuration files and simplifies the administrative control machines on a network. It is particularly suitable for files that would be the same on all the systems such as /etc/passwd, /etc/group and /etc/hosts . In particular with the password file it is important for authentication for some commands to maintain the same userid. This is greatly simplified if they all use the same configuration file.

NIS Control of /etc/passwd & /etc/group

To allow control of the password file requires there still to be a root userid configured on the system and then a special escape sequence to indicate that the NIS file should be used. A /etc/passwd file showing the root user and escape sequence is shown below:

```
root:!:0:0::/:/bin/sh
+::0:0:::
```

All entries above the escape sequence are considered local and are not included in the NIS password file.

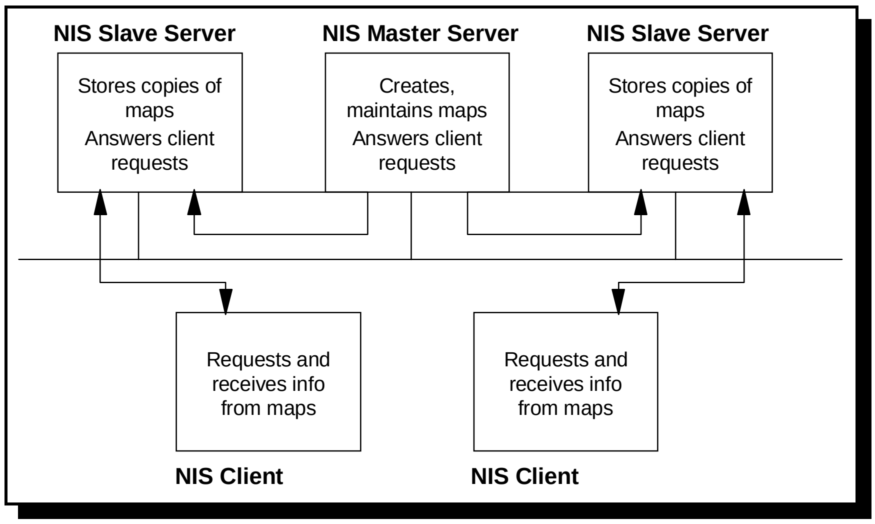
The NIS maps are created in DBM format which is a database system built into BSD systems.

The /etc/group file is similar to the /etc/passwd file except that the escape sequence is different The file is shown below:

```
system:!:0:root,su
+:
```

NIS Systems

The following diagram shows how the client / server relationships are defined.



NIS Client Server Relationship

The NIS Master Server is the true server that maintains and distributes all the maps. The NIS Slave Servers distribute the load for client requests and provides a backup if the NIS Master Server is not available. As with NFS it is possible for a NIS Server to be a NIS Client as well.

A group of hosts sharing the same set of NIS data maps is known as a NIS domain. Note that this is around a set of maps and not around a set of machines. When the maps are created they are stored in a directory called /var/yp/domainname

NIS Daemons

When a NIS client requests information from a NIS data map, it uses it's ypbind daemon. The server will use the ypserv daemon to reply. Once this is established then the information is accessible.

When a NIS user changes a password using the yppassword command then the communication is with the servers yppasswd daemon. The change is made in the NIS master's /etc/passwd file and then is reflected in the passwd data map, this is then transferred to all the NIS slave servers.

Action	**Client**	**Server**
Login	ypbind	ypserv
Change Password	yppasswd	yppasswd

Basic NIS daemons

Once a session is "bound" from a client to server, the client does not need to issue another NIS broadcast. Instead it stores the IP address of the server it is bound to in the /var/yp/binding/*domainname.version* file. If however the server crashes or responds too slowly then the client will break it's binding and request the service from another server.

The ypwhich command can be used to query the server that a client is bound to.

NIS daemons and their Subsystems		
File	**Subsystem Name**	**Group Name**
/usr/lib/netsvc/yp/ypserv	ypserv	yp
/usr/lib/netsvc/yp/ypbind	ypbind	yp
/usr/lib/netsvc/yp/rpc.yppasswd	yppasswd	yp
/usr/lib/netsvc/yp/rpc.ypupdated	ypupdated	yp
/usr/sbin/portmap	portmap	portmap

NIS Daemons under SRC Control

The ypupdated subsystem is only used when running secure NFS. This runs on the master server only. As you will not normally want to use this on the server and as a number of the subsystems are normally not used on the client, then the option startsrc -g yp is rarely used. Instead the subsystems are started individually. The following lists the Default NIS Data Maps that con be configured:

/etc/passwd	Usernames, userids and passwords

/etc/group	User Groups
/etc/hosts	Hostnames and IP addresses
/etc/bootparams	Information about diskless nodes (not used by AIX)
/etc/ethers	Ethernet numbers ie.MAC addresses (not used by AIX)
/etc/aliases	Alias and mailing lists for the mail system
/etc/netgroup	Netgroup definitions (used by NIS)
/etc/netmask	Network Masks
/etc/networks	Network Addresses
/etc/protocols	Network protocol names and numbers
/etc/rpc	Remote procedure call program numbers
/etc/services	Network Port Numbers and service names
/etc/publickey	Keys for secure NFS
/etc/netid	ID info for machine, hosts and groups.

The system administrator can also define their own maps.

Configuring NIS Master Server

After ensuring the server software is installed SMIT handles much of the configuration. The NIS pages are under Communication Applications and Services, NFS and then NIS. To change the domain name the fastpath smit chypdom could also be used.

Next /etc/passwd and /etc/group are edited so that they include all the user accounts and groups that will be used. It is important that there are no duplicate entries in either of these files.

The /etc/hosts file is edited so that it includes entries for all the hosts in the NIS domain, that will be supported by this server.

The machine is then set as a master server using smit mkmaster

The slave server names are included, and the /var/ypdirectory is created along with ypinit -m to initialise the maps. yppasswdd, ypbind and ypupdated can be started this way.

The maps are generated and in the case of hosts and passwords, two different data maps are created one byname and one by address for hosts and one by name and one by UID for passwd.

Files ending with .pag contain the key and value pair in DBM format. If the files are over 1000 entries long then the .pag files are used to provide faster searching. The map names are:

```
/var/yp/domainname/map.key.dir
/var/yp/domainname/map.key.pag
```

The ypcat command can be used to view the contents of the data maps. ypcat passwd displays the passwd.byname map and the command yp -k passwd will display the keys used as well as the contents. There is a nickname translation table that can be viewed by using ypcat -x.

Configuring a NIS Client

The password and group files on the client should be edited and all none-local entries removed. The /etc/hosts file should have all entries removed except for the Loopback entry and the entry for itself.

the NIS domain is set using smit chypdom.

Then the client is setup using smit mkclient. This adds the escape sequences to /etc/passwd and /etc/group and then starts the ypbind daemon.

For the /etc/passwd file the following line is entered

```
+::0:0:::
```

For the /etc/group the following line is entered

```
+:
```

Removing NIS

If you later decide to remove NiS this can be done using

smit rmypserv	For a Server
smit rmypclient	For a Client

Serial Connections (Dial / WAN)

Dial connections and WAN connections normally use a serial connection to communicate with another machine. There are two common protocols that are used SLIP and PPP.

SLIP (Serial Line Internet Protocol)

For a serial connection there are two connections (one on each end) which are in their own network. There is no need for ARP translations as the interface is only capable of communicating with the partner connection.

SLIP is an Internet standard documented in RFC 1055. It is however less used now in favour of PPP (explained later). SLIP is however a lot simpler than PPP.

PPP (Point-to-Point Protocol)

PPP is an alternative method of connecting over a serial connection. It is specified in RFC 1661 (encapsulation method and link control), RFC 1662 (framing method) and RFC 1332 (PPP used with IP). PPP provides additional functionality to SLIP, however is considerably more complex.

PPP Supports multiple protocols on a single link and can dynamically negotiate IP address, authentication (using PAP and CHAP) and compression.

PPP uses the following techniques / protocols
- Link Control Protocol (LCP)
- Network Control Protocol (NCP)
- encapsulation / framing technique

When a connection is established the link must go through a configuration negotiation to agree on the protocols being used. The framing and encapsulation uses between four and eight bytes and the other protocols including IP addresses, authentication and compression.

AIX can be either the calling or the called system. In AIX one system acts as the server and the other as the client. In most cases the client is the calling system and the server the called system, it can however be the other way round. The server provides the IP address that is to be used by the client. The server always knows both addresses, however the client initially doesn’t know either until the connection is established.

Demain PPP connections are also supported allowing a connection to be made whenever an applications attempts to use the connection.

For authentication both PAP (Password Authentication Protocol) and CHAP (Challenge Handshake Authentication Protocol) to allow one or both of the ends to authenticate the other. PAP is the considered greatly inferior to the CHAP protocol.

There are several commands and processes used in PPP:

pppcontrold: This daemon creates the PPP subsystem and manages the PPP configuration. The PPP subsystem must be started before a connection can be established. This must be done on both the client and the server either using SMIT ppp (start ppp) or

```
startsrc -s pppcontrold
```

pppauthd: is used for PAP and CHAP authentication. The daemon is started automatically by pppcontrold.

pppattached: This is used to actually perform the PPP protocol. An instance runs on both systems, connected to the asynchronous link between them. On the calling side the command runs in the background once the connection is made, on the called side, pppattached is normally run as a foreground command, invoked by the .profile of the user ID associated with PPP.

As a client it is started as:

```
pppattached client tty0 connect “pppdial -f chatfile”
```

As a server it is started as:

```
pppattached server
```

pppdial: This is invoked by pppattached on the calling side. The ppdial command is given a “chat script” to indicate how it should dial an log into the remote system. The execution of this is shown above under pppattached.

To install PPP the following steps are followed:

1. Install bos.net.ppp
2. Create a TTY device
3. Create PPP Link Control Configuration
4. Add IP Interfaces for PPP (server only)
5. Define PAP / CHAP authentication (optional)
6. Start PPP subsystem (pppcontrold/pppauthd)

The software must be installed first, using either installp or SMIT.

The TTY device can be created using smity maktty. Most the default values can be used however Enable Login must be set to enable. Since clients will login here, we want a getty to be spawned for the TTY. When setting up the client the Enable Login parameter should be set to disable.

For modem we can use hardware flow control rts / cts. The speed for a modem defines the link between the RS/6000 and the modem so is set high rather than using the actual speed of the modem.

The Link Control Configuration is setup using smit addlcp.
For a server the number of server connections is set to a non-zero value, and normally the maximum number of IP interfaces and maximum number of HDLC attachements is set to the same number. If the server is used as a client as well then the maximum number of IP interfaces and HDLC attachments should include those as well.

For a client or server the following information is included in the addlcp stage (which is stored in /etc/ppp/lcp_config):

PPP Subsystem Name: String to uniquely identify the system within a PPP network (used by PPP only).
max server connections: The maximum number of PPP connections (0 on a client)
max client connections: The maximum number of PPP client connections on this system
max demand connections: Maximum number of demand connections. Normally 0 on a client.
max ip interfaces: Overall maximum number of PPP IP interfaces for this system. The sum of all server and client connections.
max async HDLC attachments: Maximum number of asynchronous PPP connections that can be active concurrently. Set to number of IP interfaces (as this is only layer 3 protocol supported by AIX).
Next for a server only smit addpppserver is run.

This defines the interfaces that are used for PPP. For multiple interfaces the first IP address is used and the number of addresses including that one. These are given the addresses pp0 to pp*n* stored in the file /etc/ppp/if_config. As clients connect, they are assigned one of the interfaces and the address.

Also on the server a copy of pppattachd must be run for each client. This is normally done by having an account for clients to log into e.g. ppp
Start pppattachd in .profile
exec /usr/sbin/pppattachd server 2>/dev/null

The ppp should be started for both the client and the server by using smit startppp. When this is done a pp0 interface will be created:

```
pp0:flags=6000030<POINTOPOINT,NOTRAILERS,GROUPRT,64BIT>
     inet 0.0.0.0 --> 0.0.0.0 netmask 0xff000000
```

Establishing a connection:

Once setup the pppatachd command is used to connect over PPP. The command to connect is:

```
pppattachd client tty0 connect “pppdial -f mychatfile”
```

The pppattachd invokes the pppdial which creates the connection. This will use the mychatfile for the login negotiation. Assuming all goes well the connection will now be setup.

Chat File
The following shows an example chat file (note the line numbering is not part of the file and is used for illustration purposes only).

```
1 ‘’
2 atdt123456
3 CONNECT
4 \d\n
5 ogin:
6 ppp
7 ssword:
8 ppppw
```

Each command takes two lines. The first is what to expect from the chat program and the second is the response sent by the client.

Each line of the file is explained below

1. Expect a null string
2. Send the modem the dial command (123456 is the phone number)
3. Expect the CONNECT string from the modem
4. Delay for 1 second then send a new line
5. Expect the login prompt
6. Send the user id to the remote system (ppp)
7. Expect the password prompt
8. Send the password (ppppw)

To avoid error messages when PPP tries to write it’s PID file users should be in the group uucp. The file created is /etc/ppp .

There is an example of using ppattachd with a chat script in the file /etc/ppp/dial_out.example.

When ppp connects the client sets up a route to the remote system only. However often further route commands are needed. For example if the client was a dialup machine needing to use the server as it’s only network connection it might want the default route to use the PPP server. This must be added to the scripts used to start the connection. As the IP address is unknown before the connection then the ifconfig or netstat -nr commands should be used to get the address of the remote end.

When a client connects to the server a free address is allocated from the pool, this is not based on the client identity or the incoming port.

TCP/IP Startup

When cfgmgr runs on startup it reads entries from the ODM and either /etc/rc.net (AIX default) or rc.bsdnet (BSD-style configuration file). It uses these to initialise the network interfaces and to setup the routing. At this point it would be possible to connect to another host, but it would not be possible for the machine to take an incoming connection.

The next stage is whilst reading the /etc/inittab the system calls rctcpip and starts any daemons included in the rc.tcpip file. Included with this is starting of inetd. The machine is now in a state to accept incoming connections.

The system startup procedure is explained in the next section.

Networking Autorisation Files

There are a number of files that can be used to allow network based logins or to automate them. I have listed some of the files below.

$HOME/.netrc

If there is a $HOME/.netrc file in the local users directory then it can permit automatic logins. It also allows macros to be automatically run when connecting. The file can contain one entry for each different host that you want to allow connections from.
A sample file is shown below:

```
machine host1 login user1 password mypw0rd
macdef download1
  type ascii
  get /data/download/file.txt

macdef files
  type binary
  cd /data/other
  get file1
  put file2

machine host2 login user2 password afkh3kl
macdef download
  get /home/user2/file.txt
```

Sample $HOME/.netrc file

Whenever the ftp command is run to connect to one of these machines then the commands will be automatically run. To run ftp without calling this file then the -n option is used. Caution should be used when using this method of automating login as the password is kept in plain text.

The rexec command can also use the .netrc file to provide the username and password for running a program on the remote system. The rexec command cannot however interpret the macdef instructions and may produce an error message.

/etc/ftpusers

Whilst remote login for telnet is governed by the use of SMIT updates being stored in the ODM ftp has a flat text file. The file is /etc/ftpusers and it lists users that are NOT allowed to login using ftp from a remote machine. This can be updated using a text editor or by using smit ftpusers .

The file below shows some typical entries that you might want to include. These are the root and system usernames.

```
root
nuucp
daemon
bin
sys
adm
uucp
nobody
lpd
```

Sample /etc/ftpusers file

To login with ftp you would still need a user entry in the /etc/passwd and a valid password.

/etc/hosts.equiv

The hosts.equiv file is used to provide a direct login without requesting the password and is used with the 'R' files. It requires that the same usernames are configured on both machines. It would however not allow access to the root username.

The following file explains how this may be implemented:

```
rs6k1
-rs6k2
rs6k3  user3
rs6k4  -user4
```

Sample /etc/hosts.equiv

I have explained each of the 4 different entries below:

1. Allows any user on rs6k1 to login as the same user (excluding root)
2. Does not allow any logins from rs6k2. The rest of the rules would not be checked, including the .rhosts file.

3. Allows user3 on rs6k3 to login as any user other than root - Not normally desirable
4. Allows other users except for user4 to login from rs6k4.

$HOME/.rhosts

The .rhosts file is normally looked at only if the login fails in the hosts.equiv file. The .rhosts file sits in the users directory on the remote machine that the user wants to access. The file contains the system name and username that is allowed to login. If this did not authorise the connection then the only remaining method is a valid password.

IP Version 6 (IPV6)

The new version of the Internet Protocol is version six often referred to as IPV6. It's biggest advantage is in the number of addresses available. It extends the addressing scheme to 128 bits which should provide ample capacity for the future.

IPV6 will also provide further improvements including Quality of Service which will allow real-time applications, such as video conferencing to have priority over batch traffic.

The implementation of IPV6 has been delayed by the use of Network Address Translation (NAT) which allows companies to use private addresses internally requiring less real IP addresses to be used. The shortage of IP addresses may however re-emerge with the popularity of mobile phones, as new technology now means that mobile phones may need a fixed IP address in future.

Other Networking Protocols and Products

Another protocol that may be used from an AIX client is tn3270. This is based upon a telnet data stream carrying 3270 data. 3270 is a mainframe protocol providing text based access to TSO, VM etc. AIX provides a basic tn3270 client called tn3270. This can be invoked by entering tn3270 followed by the hostname. This is particularly limiting software and by default does not support any of the 'F' keys etc. which are essential for use in ISPF and other mainframe applications. A much easier to use alternative is x3270 which as it's name suggests is an X-Client. The x3270 client can be downloaded using anonymous ftp from pdslib4aix.seas.ucla.edu in directory pub/x3270/RISC/4.1/exec

For TN3270 to be used it requires that either you are connecting to a mainframe that supports tn3270. This is standard for any mainframes with a configured TCP/IP stack or a tn3270 gateway. Typically this could be another AIX gateway running SNA Communications Server (this is a LPP that should be purchased separately) to provide a translation from native 3270 to tn3270.

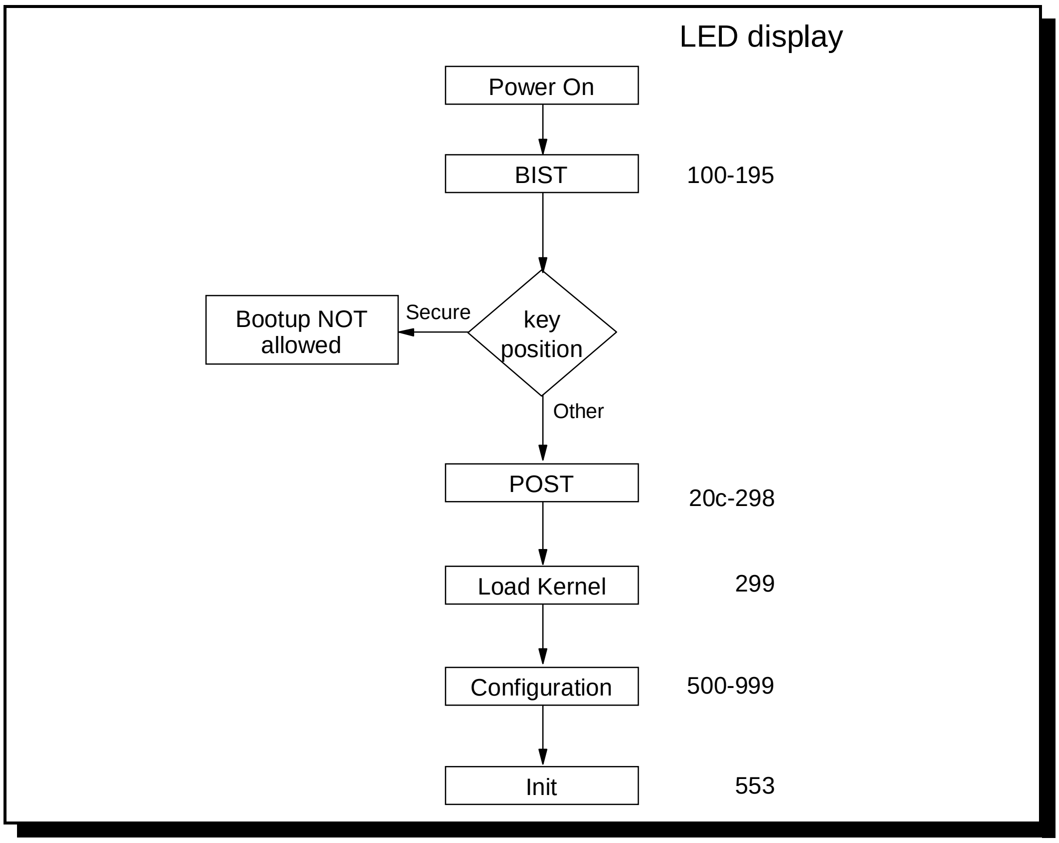
AIX can also support native 3270 with the use of a 3270 client, however this is becoming less popular in favour of some kind of IP based connection (i.e. tn3270).

System Startup

As system startup is different depending upon the hardware. In particular the differences between a Classical RS/6000 and a PCI RS/6000 are great and therefore will be treat differently.

Hardware Initialisation for a Classical RS/6000

The following flow diagram shows the process of system startup.



Flow Diagram for Classical RS/6000 System Startup

During system startup the LED's go through a sequence. In the event of a failure the LED can help determine the cause of the problem.

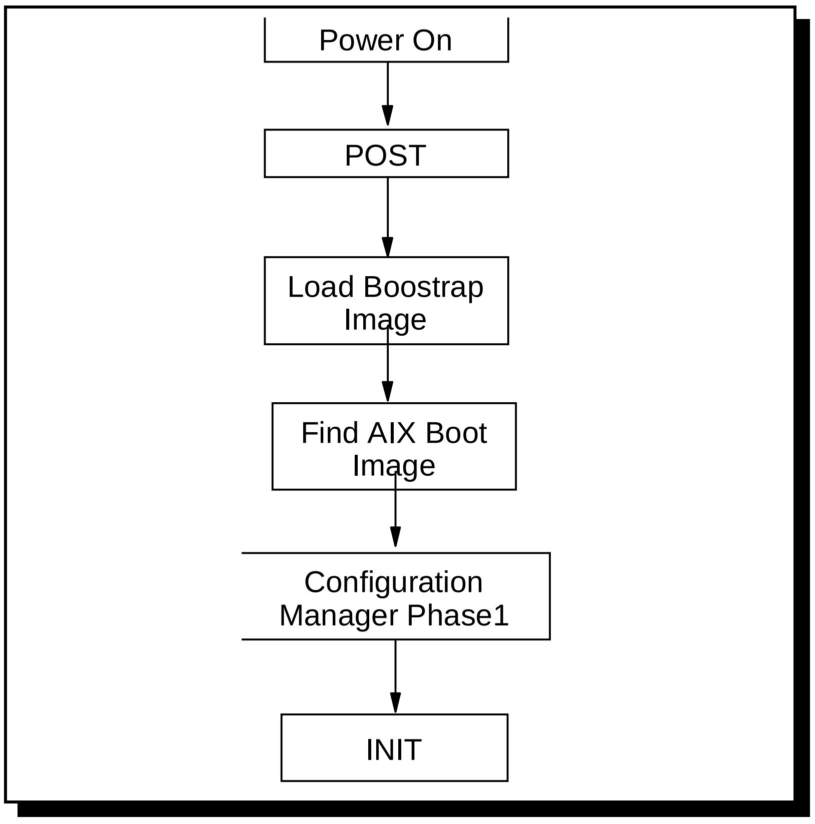
The steps are as follows

BIST - Built-in self test. This initialises the basic components of the system like the processors, memory, etc. The LED displays numbers between 100 and 195.
POST - Power on Self Test. At this point the initial program starts. During this stage the boot device is identified and the boot program is loaded from the boot device into memory. The LED’s display numbers in the range 200-299.
Configuration - During this next step the cfgmgr runs and finds all the devices on the system. The LED displays numbers in the range 500-999.
Init - The final phase is to create the first process (number 1) which is called init. The process will read the /etc/inittab file to determine what other actions need carrying out. The LED now shows 553.

During power up the LED’s are not neccessarily displayed sequentially. The actual meanings of the codes are held within the AIX Messages and Guide Reference.

Hardware Initialisation for a PCI RS/6000

The following flow diagram shows the process of system startup.



Flow Diagram for PCI RS/6000 System Startup

This is similar to the startup process of the classical RS/6000 however there are a number of differences.

A noticeable difference is the lack of LED display, on many PCI models. The LED is used to indicate problems during the bootup sequence on the Classical RS/6000. However on the PCI systems there are a number of other things that can provide the same information. This is carried out as part of the Power-On Self-Test. If there is a LED display then the firmware numbers are normally displayed on it.

First there is an audible bleep. If this is heard then the processor is operations, the memory controller has found the System ROS. The processor was able to load the data and instructions from the System ROS and is OK so far.
The next thing to appear is the PowerPC Logo on the graphics screen. This indicates the system memory has been checked, the system I/O is initialised, that the graphics has been loaded and the device initialised.

Then a device logo is displayed as each of the devices is checked.

Finally an audio bleep is heard to confirm that the audio system is working correctly and that the System ROS has not been corrupted.

The System ROS is an operating system independent piece of firmware. I provides a software equivalent of the function provided in the classical RS/6000.

Another large difference is the lack of a key switch for most RS/6000's. Instead there are two modes, normal mode which is booted by default or Maintenance Mode that can be accessed through System Management Services.

The System Management Services is either provided in the form of a disk or is built in to the system. To access System Management mode either the F1 or F4 keys are pressed during startup. The point at which the key needs to be pressed is during the device checking, after the keyboard but before the last ICON has been displayed.

In maintenance mode, a number of maintenance operations can be carried out. The commands are run before AIX is started providing a means of recovering the system if there is a problem with the hardware or operating system.

There are normally four options within the main System Management Services menu.

• Manage Configuration	- View or change the system setup
• Select Boot Device	- View or change the bootlist order
• Test the computer	- Carry out hardware tests and display the error log

- Utilities - Display the System Management Utilities menu which includes, power-on / supervisory passwords, changing unattended start mode, view error log, update the firmware etc.

To reboot the system from maintenance mode, either <ctrl-alt-delete> or F3 can be pressed, or the operating system started by pressing F9.

Software Initialisation

Now that the hardware has been checked and the BOS located the software is loaded and runs. The Virtual Memory manager (VMM) is initialised. The Virtual Memory Manager is required for the following:

- Virtual address space management for processes.
- Sharing of executables
- Shared memory segments
- Mapped files

Then the kernel storage management is initialised and Interrupt processing is initialised.
Then the swapper, init and wait processes are started and phase 1 of init executes.
Also the wait process is started (this is the process that runs whenever the processor is idle).

There may be a different init run if the system is booted from installation media. In this case it will run a install init known as INST instead.

Before we reach the Init phases a RAMdisk is created to hold a cut down version of the BOS files. This is required as other than the BLV (Boot Logical Volume) the disks have not yet been configured and cannot therefore be accessed.

As the Initialisation stage is very long I have split it into a number of phases.

Initialisation - Phase 1

- **rc.boot 1** The rc.boot shell script is run with parameter 1.
- **chramfs -t** The RAM file system is expanded. This provides the Operating systems in a file system constructed in RAM. This is later mounted over by the root file system.
- **restbase** The ODM is copied from the boot device to the file system on the RAMdisk. This allows the ODM to be reconstructed in the event of any problems.
- **cfgmgr -f** Whilst this is the second run of the cfgmgr it is the first time it is called from software therefore the -f option is used to specify first. This phase is used to identify the boot device so that the root file system can be loaded.
- **IPL Device** The bootinfo -b command is run to determine the boot device. The device is linked to /dev/ipldevice.

Initialisation - Phase 2

- **rc.boot 2** The rc.boot is run a second time (2).
- **ipl_varyon** The rootvg is varied on (this is a special version of the vary command)
- **mount hd4 /mnt** The root file system is now mounted
- **mount /usr** The usr file system is mounted. Around this time logging is also enabled. However the logs cannot yet be written to the alog.
- **mount /var** The var file system is now mounted in case it is needed for a system dump.
- **Handle dump?** If a system dump has occurred the copycore command will create the file /needcopy. If this file exists the dump will be copied to the /var/adm/ras directory. If there is insufficient space for the dump then the copy dump menu is displayed. After this the /var directory is not needed until the end so it is unmounted.
- **turn on paging** Paging is turned on with the swapon command.
- **Copy LVM** Details in the LVM on the RAMdisk is copied to the real disk. (/etc/vg etc.).
- **merge /dev** The /dev directory on the RAMdisk is merged with that on the real disk
- **merge ODM** The ODM on the RAMdisk is merged with the copy on disk
- **destroy RAM-fs** The chroot command is run to change the root directory. The RAMdisk is destroyed and the /usr filesystems is moved to it's proper mount point.
- **mount /var** The /var filesystem can now be mounted
- **Update alog** The /tmp/boot_log file is then copied to the alog.

Initialisation - Phase 3

- **init (process 1)** The /etc/inittab is now read in line by line.
- **rc.boot 3** The third run of rc.boot
- **syncvg rootvg** Synchronise any stale partitions in rootvg
- **mount /tmp** There must be at least 1MB available. If there is insufficient space then files are removed until there is sufficient.
- **cfgmgr -s** cfgmgr is run with the -s option, this reads config_rules and configures the phase=2 devices (normal mode) or phase=3 devices (service mode)
- **cfgcon** Configures the console.
- **rc.dt** Run if setup for a graphic screen
- **savebase** Retrieves customised ODM information and writes to the boot logical volume on the boot disk.
- **start syncd** Used to maintain disk integrity
- **start errdemon** Start the error logging daemon
- **(turn off LED's)** If available
- **rm /etc/nologin** If this is present then no users (other than root) can login. Removing it therefore enables logins.
- **diag mesg** The diagnostics message appears with details of any devices that are now missing.

- **inittab line 3**

The inittab file

The init process runs through the /etc/inittab file starting the necessary processes. It is responsible for initiating terminal, daemons, the console, and the mounting of file systems. The format for the file is :

```
Identifier:Runlevel:Action:Command
```

- Identifier - This is a 14 character field used to identify the object. E.g. tty0 identifies the device /dev/tty0
- Run level - This is a 20 character field which identifies the run level at which the command is processed. Each process is assigned one or more runlevels that it can be running under. These can be 0-9, S, s, M and m. Run level 1 is single user; run level 2 is multi-user, S, s, M and m are used for maintenance. Normally a system will run in 1 (single user mode) or 2 (multi-user mode).
- Action - This 20 character field that informs init what to do with the process. The most common actions are:
  - respawn If the process does not already exist start it, if it stops restart it.
  - wait Start the process and wait for it to stop. Subsequent reads of the inittab file will ignore this.
  - once Start the process, do not restart it.
  - initdefault Process only when the init command is originally invoked
  - sysinit Execute before the init command tries to access the console
  - off Don’t execute
- Command - This is a 1024 character field for the command to run.

```
init:2:initdefault:
brc::sysinit:/sbin/rc.boot 3 >/dev/console 2>&1 # Phase 3 of system boot
powerfail::powerfail:/etc/rc.powerfail 2>&1 | alog -tboot > /dev/console # Power
 Failure Detection
rc:2:wait:/etc/rc 2>&1 | alog -tboot > /dev/console # Multi-User checks
fbcheck:2:wait:/usr/sbin/fbcheck 2>&1 | alog -tboot > /dev/console # run /etc/fir
 stboot
srcmstr:2:respawn:/usr/sbin/srcmstr # System Resource Controller
rctcpip:2:wait:/etc/rc.tcpip > /dev/console 2>&1 # Start TCP/IP daemons
rchttpd:2:wait:/etc/rc.httpd > /dev/console 2>&1 # Start HTTP daemon
rcnfs:2:wait:/etc/rc.nfs > /dev/console 2>&1 # Start NFS Daemons
cron:2:respawn:/usr/sbin/cron
piobe:2:wait:/usr/lib/lpd/pio/etc/pioinit >/dev/null 2>&1  # pb cleanup
qdaemon:2:wait:/usr/bin/startsrc -sqdaemon
writesrv:2:wait:/usr/bin/startsrc -swritesrv
uprintfd:2:respawn:/usr/sbin/uprintfd
logsymp:2:once:/usr/lib/ras/logsymptom # for system dumps
pmd:2:wait:/usr/bin/pmd > /dev/console 2>&1 # Start PM daemon
diagd:2:once:/usr/lpp/diagnostics/bin/diagd >/dev/console 2>&1
dt:2:wait:/etc/rc.dt
cons:0123456789:respawn:/usr/sbin/getty /dev/console
tty0:2:off:/usr/sbin/getty /dev/tty0
```

Rather than update the inittab directly the following commands should be used:
`mkitab` or
`chitab`

If the inittab file is altered it can be reread by using:
`kill -1 1` or
`telinit q`

After the rc.boot 3 has run the /etc/rc script runs. This prepares the system for multiuser mode. This carries out a varyon of the volume groups, activates paging space, and mounts any automount logical volumes (mount=true). This does not include /usr, /var, / and /tmp which have already been mounted by rc.boot (mount=automatic).

At this point the system is ready for use.

The System Resource Controller (srcmstr daemon) controls subsystems including the spooler and networking. It handles subsystem request, passes requests on to a subsystem and handles failure notification. The startsrc command will then start a subsystem or service.

The terminals, lines and ports are controlled by the getty daemon. The terminal state manager (TSM) is invoked. This controls the terminals and generates the herald message, reads the login name and starts the login program.

The inittab can be made to change runmodes by issuing the telinit command followed by the runmode e.g.
```
telinit 3
```

Startup Log (alog)

Whilst the alog program can be used by any program for logging purposes, it’s main use is to provide the logging function for startup, by default it has 3 logs boot, bootinst and nim. What the alog does is to echo stdin to the screen and to write it to a log at the same time. Whilst this could be provided by the “tee” command the added function that alog gives is that the log is kept within a fixed size. This is important for the startup log to be kept to a fixed size. Otherwise every time the system is started the log will get larger and larger filling up all the available disk space. Also as the log contains details of hardware initialisation this is information that is not needed once the system is up and running correctly. The log file is a circular log. Once the file is full new entries are overwritten over the oldest entries.

To view the boot log the following command can be run

```
alog -o -t boot
```

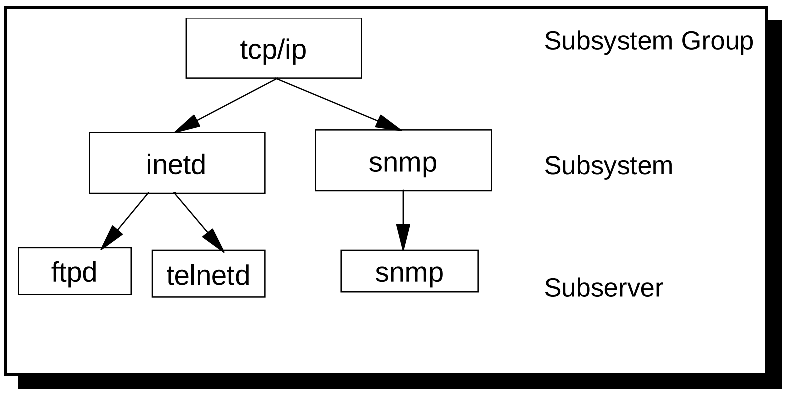
The boot log is kept in /var/adm/ras/bootlog and by default is 4k in size.

System Resource Controller

The System Resource Controller (SRC) provides a consistent user interface to controlling subsystem resources. It provides a means to start, stop and enquire on the status of a subsystem, logging of abnormal termination, tracing facilities, control of a remote system etc.

The SRC srcmstr daemon is started during initialisation by the inittab (/etc/srcmstr).

There is a hierarchical structure for the system resources:



Hierarchy of System Resources (e.g. TCP/IP)

A subsystem group is a number of subsystems that can be controlled together. These are normally grouped around a common function (e.g. the tcpip group covers networking subsystems and the spooler group covers the printing daemons).

A subsystem is a program or set of related programs designed as a unit.

A subserver is a process or daemon that belongs to and controls by a subsystem.

The following commands can be used although not all subsystems support all the options the following all show the inetd path of the subsystem shown in the earlier diagram:

- Start a subsystem
  `startsrc -s inetd`

- Stop a subsystem
  `stopsrc -s inetd`

- Refresh a subsystem
  `refresh -s inetd`

- Listing of a specific Group
  `lssrc -g tcpip`

- Listing of a subsystem
  `lssrc -l -s inetd`

- Listing of a subserver
  `lssrc -t telnet`

Some other options that can be used are:

-a (all defined subsystems)
-f (forced stop)
-c Cancel Stop

-s refers to a subsystem
-g refers to a group
-t refers to a subserver

The SRC commands can also be accessed through SMIT using the fastpath smit src .

Hardware Startup Problems

Built in Self Test Problems (BIST) - LED 100 to 195

During this stage the LED's display the numbers in the range 100 - 195. See appendix *** for a brief description of the messages. The actual meaning of the LED's can be found in Diagnostics Service Guide and the Messages Guide and Reference.

If it fails during this stage then the problem is definitely hardware related.

Things that can be checked is to re-seat the boards and cables (although if the hardware is under warranty this should be checked first). If in any doubt then an IBM engineer should be contacted.

If there is a problem with the fan then the machine would automatically power itself off giving little indication of where the problem lies.

A call should be logged with the AIX Systems Support Centre. Give as much information as possible about the state of the machine, the LED code and any changes which have been made recently.

Power On Self Test Problems (POST) - LED 200 to 299

There are two different problems that can occur during POST. The LED will be in the range 200 to 295. See appendix *** for a brief description of the messages. The actual meaning of the LED's can be found in Diagnostics Service Guide and the Messages Guide and Reference.

If two values are displayed on the LED repeatedly. This indicates that IPL Read Only Storage is in a loop, trying to load the boot program. The device that is attempting to IPL is indicated by the values in the LED display.
Check the mode switch and check the device list.

If a single value is displayed then this is likely to be a device or media not ready or failing.
A check that can be carried out is to turn the mode to secure and wait 5 minutes. If the code changes to 200 then solution is the same as above. If not then record the SRN 101-XXX in the Problem Summary Form for AIX Systems Support.

There are however some circumstances where values in the range 200 to 295 can be related to other problems.

Problem with Boot Logical Volume (BLV) - LED 201

The LED code for this condition is 201, however the LED display should be watched closely to see if it is a problem with the BLV. If the LED passes 299 then this is a problem with the BLV. If the LED does not pass 299 then this is should be treat as a problem during POST.

In this case the BLV will need to be recreated.

Problem Finding the Disk to Boot From - LED 223

The LED code for this condition is 223. This is likely to be a problem with the SCSI disk or associated connectors, adapters etc. The SCSI adapter (including any fuses), connectors and terminator should be checked.

Booting in service mode will allow you to try and query if there are any devices listed after the SCSI adapter.

If the problem is not fixed by checking the cabling then this should be reported to the AIX System Support.

System Crash During Startup - Flashing LED 888

This condition can occur as a software or hardware error. There will be another number following the 888 which will be either 102, 103 or 105.

If the code is 102 then this is a software problem. Following the 102 code will be another number. A 200 would mean a memory bus error whereas a 300 would be a data storage interrupt. The SRN number is 102 followed by the next code.

If the code is 103 or 105 then this is likely to be a hardware problem, however could have been caused by software. Pressing the reset button twice will give the SRN, pressing reset once more will give the FRU and then the following 8 times would give the location code.

PCI Power Up Problems

As the PCI systems don't always have an LED display the above problems cannot neccessarily be identified immediately. However depending upon the phase reached in the boot up process a number of problems can be identified. For example the startup will not fail due to most single device failures as long as it is able to find a boot image. It will instead display an error code for a none critical component failure. The error codes can be found in the System Users guide provided with the system. It is possible to run some device tests by booting into SMS mode.

A failure during the following phases of boot up would indicate a problem in a certain area.

Phase 1 - No audible beep
Exchange the Flash ROM or system board.
Phase 2 - Audible Beep, No Power PC logo
Move the graphics adapter to another PCI slot. Exchange graphics adapter, Flash ROM or System Board
Phase 3 - Power PC logo displayed failure at a device logo
An error code will be generated
Phase 4 - As phase 3
Final Stage - Final Beep heard however still doesn’t startup correctly
Run hardware tests through SMS, or follow steps below.

If the system stops on a white screen then this may be due to it being unable to find the boot image. See below for more details.

Fixing a Corrupted BLV

If the Boot Logical Volume becomes corrupt then it will not be possible to boot in normal mode. Therefore the maintenance or SMS mode should be booted.

First check that the devices that the system is set to boot from are valid. This can be done using bootlist -om command or by a menu option in SMS

The default order is:
Diskette Drive (not supported by AIX V4)
Internal CD-ROM
Internal hard disk
Network Adapter (for installation from NIM master only)

for some systems it is possible to set the system to boot from tape.

Then identify the disk with the BLV

```
lslv -m hd5
```

There must also be sufficient space in the /tmp filesystem and the file /unix must be a symbolic link to the /usr filesystem BLV source file (/usr/lib/boot/unix, /usr/lib/boot/unix_up etc.)

Run the bosboot command to place the boot image on the BLV.

```
bosboot -ad /dev/hdiskn
```

The system should then be restarted in normal mode.

Service Request Numbers (SRN)

The Service Request Number (SRN) is a six digit code representing a specific failure of a specific function.

The source code indicates the program or procedure that produced the SRN. The SRN source codes are:

A	The SRN is from a steady number in the operator panel display
B	The SRN is from a MAP call-out
C	The SRN was due to missing resource at configuration time
D	The SRN is from a diagnostic test after complete isolation testing
E	The SRN is from a POST failure
F	The SRN is from a diagnostic test after partial isolation testing
G	The SRN is from the Error Log Analysis program
H	The SRN is from a diagnostic message after a flashing 888 LED
J	The SRN is from a built-in ROM diagnostics
K	The SRN is from off-line diagnostics.

The SRN is recorded on the Problem Summary Form which should be filled in for a hardware problem. This is included at the front of the Problem Solving Guide.

Software Startup (IPL) Problems

Root File System cannot be mounted - LED 557

If the system stops with the LED 557 then this is likely to be a problem mounting the root file system.

First check the device using the diagnostic tools mentioned earlier.

Correct any problems with the BLV using the procedure described earlier.

If this hasn't fixed the problem then the JFS log may be corrupt. This can be reinitialised using the logform command.

```
logform /dev/hd8
```

Or to create a jfslog use the mklv command with -t jfslog to set it as a jfs log. Then use the logform command to format it.

After running the logform command it is a good idea to run an fschk -y against the file systems in the volume group.

```
fschk -y /dev/hd1
fschk -y /dev/hd2
fschk -y /dev/hd3
fschk -y /dev/hd4
fschk -y /dev/hd9var
```

Corrupt /etc/inittab File - LED 553

If the inittab file is corrupt then the init process is unable to start the required processes. The LED 553 will be displayed. The file should be recreated to allow the system to start.

Phase 3 IPL problems - LED 551, 552, 554, 555 or 556

These failures are likely to be due to one of 3 problems. The BLV is corrupt (see earlier), the JFS log is corrupt (see earlier) or for HACMP/6000 systems both SCSI cards could be set as device number 7 which can be fixed by reconfiguring the SCSI device.

Not an AIX File System

If an error occurs, “Not an AIX filesystem” or “Not a recognised filesystem type” then the File System superblock could be corrupt. To restore this issue the command

```
dd count=1 bs=4k skip=31 seek=1 if=/dev/lvxx of=/dev/lvxx
```

where lvxx is the corrupt filesystems logical volume. It copies the spare copy over the corrupt copy.

Problem reading ODM Files or running cfgmgr - LED 523 to 537

The files will need to be restored from a backup image or a similar machine.

The specific codes are:

523	/etc/objrepos is missing or inaccessible
524	/etc/objrepos/Config_Rules is missing
525	/etc/objrepos/CuDv is missing for inaccessible
526	/etc/objrepos/CuDvDr is missing or inaccessible
528	/etc/objrepos/Config_Rules is corrupt
529	Problem with the file containing the ODM database or the root filesystem is full
531	/etc/objrepos/PdAt is missing or inaccessible
532	Not enough memory for cfgmgr
533	/etc/objrepos/PdDv is corrupt or the program specified is missing
534	cfgmgr is unable to acquire a database lock
536	/etc/objrepos/Config_Rules is corrupt
537	/etc/objrepos/Config_Rules is corrupt

When restoring the ODM file a recent backup should be used. If this is not available then a clone could be taken from another machine but only if an almost identical machine with similar hardware is used. The system should be restarted immediately after restoring a customised object class to have cfgmgr run through the hardware.

There is a cut down copy of the ODM files kept in the BLV. As a final option you could consider copying the ODM files from the BLV. To do this take the following steps:

- Reboot into maintenance mode
  Choose the option “Access the rootvg and start a shell before mounting filesystems”
- `mount /dev/hd4 /mnt`
- `mount /usr`
- Copy the corrupt files from the BLV ODM t the root file system
  `cp /etc/objrepos/Cu* /mnt/etc/objrepos`
- `cd /`
- `umount all`
- `exit`
- `savebase -d /dev/hdisk`*`n`*
  were hdiskn contains the BLV

Remember this should only be used as a last resort if a backup copy is not available.

System Environments

There are a number of System Environment settings that can be changed. These can be accessed through the SMIT fastpath smit system.

```
                             System Environments

Move cursor to desired item and press Enter.

  Stop the System
  Assign the Console
  Change / Show Date and Time
  Manage Language Environment
  Change / Show Characteristics of Operating System
  Change / Show Number of Licensed Users
  Manage AIX Floating User Licenses for this Server
  Broadcast Message to all Users
  Manage System Logs
  Change / Show Characteristics of System Dump
  Internet and Documentation Services
  Change System User Interface
  Manage Remote Reboot Facility





F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

System Shutdown

The system can be shutdown using the option in SMIT or using the shutdown command. Entering just shutdown will alert all users and then after one minutes will kill all processes, sync the disks, unmount all file systems and halt the system.

The system can be set to shutdown a certain length of time in the future. This is done by adding '+' followed by the number of minutes (or hours:minutes would give time of day). For an immediate shutdown the word now should be included on the command.

The following options can be used:

-d	Brings the system down to single user mode
-F	Fast shutdown. No messages are displayed and it is shutdown as fast as possible
-h	Halts the operating system (will shutdown the RS/6000 if power management is supported)
-i	Displays interactive messages during shutdown
-k	Does not shutdown the system displays the messages only
-m	Brings the system down to maintenance mode
-r	Restarts the system after being shutdown
-t	Restarts the system on the time specified (must be supported by the hardware).

If there are any processes that should be cleanly shutdown prior to the system halt they should be put in a file called /etc/rc.shutdown

Date and Time

The date can be changed using the command

```
date [mmddHHMM[.SSyy]]
```

using the command without any options will display the current date and time.

The date is stored internally as CUT time (Co-ordinated Universal Time) however is represented by the current time zone. This can be changed using the chtz command.

To change to CUT / GMT time then use the command

```
chtz CUT0
```

To change to GMT time with daylight saving use

```
chtz GMT0BST
```

(BST = British Summer Time).

The time zone is held in /etc/environment. Using SMIT you can set different dates for daylight saving etc.

Changing the Language Environment

The LANG variable holds details of the default locale. This is in /etc/environment and is chosen by the user at installation time. There are a number of things that are set by the locale including keyboard layout, monetary format, sort order etc. This should be checked particularly with respect to the types of keyboard that is connected.

The chlang command can be used to set the language. e.g.

```
chlang en_GB
```

Backup

For obvious reasons it is important to keep a backup copy of data on a system. Often the data contained on a computer is more valuable than the physical components that make up the system and often you could not put a price on the value of data. Some companies have been so crippled by the loss of data that the companies could no longer continue.

Backup is required in case of hardware failure (disks are notoriously unreliable - especially if you've got valuable data on them), damage whilst upgrading software etc., accidental deletion (accidentally typing rm -r * as root could wipe the entire system deletion of individual files is even easier), malicious attack (see security) or by damage or loss of the system.

Other uses of backup and restore are: transfer of data, archiving of data, defragmentation or the cloning of identical machines.

Backup Strategy

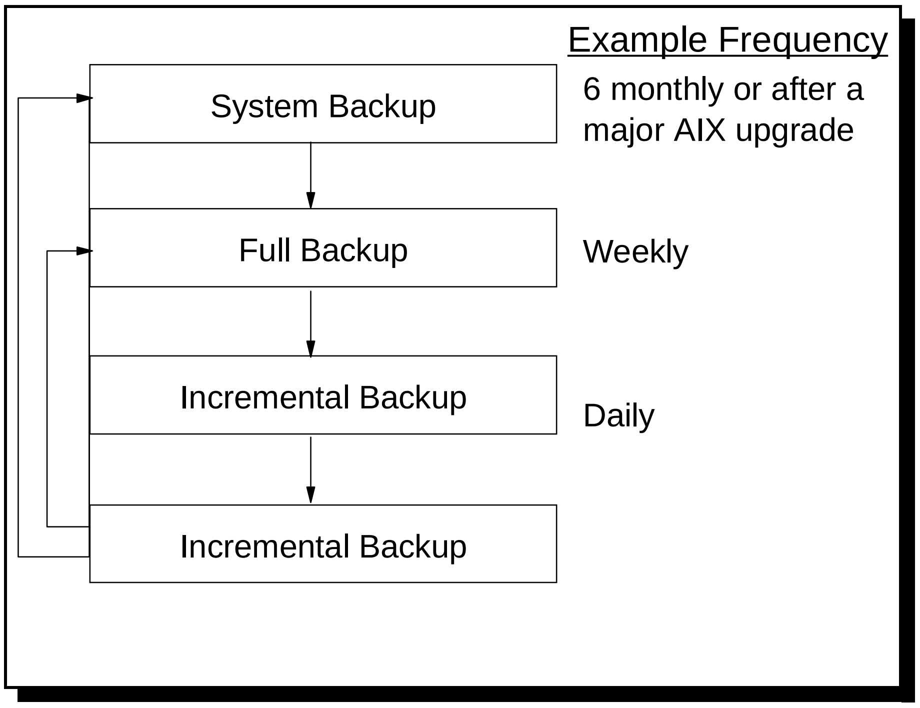
Often the backup strategy is defined as a company wide policy. However different policies can be used depending upon how valuable the data is and how often it changes. There are 3 different types of backup

- System - An image of the operating system is made (mksysb)
- Full - Preserves all user data and configuration files (jfs)
- Incremental - Records changes since last backup, if used carefully can be a fast way to backup recently changed data (jfs changes)

There are 2 different Incremental backup policies:

Where a full incremental backup is made. All changes since the last full backup are stored. In the event of a restore only the last Full Backup and the Last Incremental Backup is needed Where a partial incremental backup is taken. Only data that has been changed since the last Incremental backup is stored. In the event of a restore the Last Full Backup and all subsequent Incremental backups are needed.

Example Backup Strategy



Typical backup sequence

With the example above first a system backup is made. This is a full backup of all the system files. This should be remade periodically however does not have to be particularly frequently unless the system is upgraded or reconfigured a lot. It should certainly be done after a major upgrade.

Then a full backup is made possibly weekly. This is a backup of everything that has changed, user data, system files, application files and any volume groups that are not rootvg.

Then Incremental Backups are made at a short interval possibly daily. This depends upon how frequently user data changes and how important the changing data is.

Backup Devices

Whilst diskettes can be used for backups they are not really practical due to the small storage. Even Incremental Backups might need several diskettes and this is not practical especially for overnight backups which are the most common.

The most common diskettes nowadays are 1.44MB disks which can be accessed as /dev/fd0, which also has the devices /dev/fd0h and /dev/fd0.18

Before using a diskette it must first be formatted. The command to format the disk is:

```
format
```

Disks can be copy using the flcopy command. See later for details of how to read and write DOS formatted disks.

The most common form of media used for backing up of data is a tape drive. The most common supported tape drivers are

¼ inch tape drives
4 mm tape drives (2GB, 4GB and 12GB)
8 mm tape drives (2.3GB, 5GB and 7GB)
½ inch 9-Track drive (1600bpi and 6520bpi)

The device will be given a device name /dev/rmt0 etc. and a number of different devices are created depending upon the required operation.

	Low Capacity	**Retention on Open**	**Rewind on Close**
/dev/rmt0	no	no	yes
/dev/rmt0.1	no	no	no
/dev/rmt0.2	no	yes	yes
/dev/rmt0.3	no	yes	no
/dev/rmt0.4	yes	no	yes
/dev/rmt0.5	yes	no	no
/dev/rmt0.6	yes	yes	yes
/dev/rmt0.7	yes	yes	no

By selecting the appropriate device the correct option can be chosen.

- The correct capacity should be chosen for the type of tape being used.
- The retention option allows the tape to eject afterwards or not.
- The option to rewind on close will cause the tape to be rewound to the start. It is important that if making multiple backups to the same tape that an option that doesn't rewind the tape is used.

The tape is controlled by using the mt and tctl commands and can be copied using the tcopy command.

Backup Command (backup, restore)

There is a backup command within AIX that can be used to backup a system. Most of these can be accessed through the standard SMIT menus. For example under the File System menu you can choose to backup a file system, in the Volume Group menu you can backup a Volume group, under Files and Directories you can backup Files and Directories. There is also a

System Backup Manager menu allowing you to backup the System image. The fastpath to access this is smit backsys.

```
                             System Backup Manager

Move cursor to desired item and press Enter.

  Back Up the System
  List Files in a System Image
  Restore Files in a System Image















F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

However smit is an interactive program and cannot be set to run the jobs overnight as is normally the case. However you can use smit in the “don’t run commands” mode and check the logs for the commands to run.

The backup command can also be run from the command line. The backup command is the preferred way of making backups. The command is as follows:

```
backup -i [-q] [-p] [-v] [-f device] < listfile
```

-q media is ready
-p compress files which are less than 2GB
-v display filenames during backup

Names of files (full or relative) are read from standard input.
The find command can be used to generate the list of files to be backed up.

To backup the entire home directory
```
find $HOME | backup -i -v -f/dev/rmt0
```

To backup the files since the last full backup
```
find / -newer /etc/last_full_backup | backup -i -f /dev/rmt0
```
(here the file last_full_backup should be created using the touch command when a full backup is made to provide a benchmark date).

To restore data that was stored using backup the “restore” command is used. The -T option can be used to list the files in the backup. Before restoring data you should ensure that all the appropriate file systems are mounted.

```
restore -Tvf /dev/rmt0
```

The complete system can be restored using the -r option.

```
restore -rqvf /dev/rmt0
```

Individual files can be restored using the -x option

```
restore -xvf /dev/rmt0 /home/filename
```

When restoring individual files the names and path are all contained within the backup. The restore can be done either by using the full path name or a relative path.

The fastpath smit restfile can also be used for restoring individual files.

Rootvg Backup (mksysb)

The mksysb command backs up the rootvg volume group only. It backs up the definition for the paging space. It creates a bootable tape that can be used for a non interactive install. The inter/intra policy is also saved as part of the backup. Striped logical volume characteristics are retained.
Only mounted file systems are backed up and RAW LV’s such as database systems are not backed up.
Single or multiple files can be restored from a system image.

The mksysb command can be used to create clone machines. However this can only be used when restoring on an almost identical box, same number of processors, same bus architecture, similar devices etc.

Before the command can be used the bos.sysmgt.sysbr must be installed.

The file used for creating the rootvg install is /image.cdata. If the file is edited then mksysb should be run with either the -i or -m options to use the existing image.data file.

The /bosinst.data file specifies the requirements at the target system and the interaction with the install.

To make the install unattended the following procedure should be followed:

1. Edit the bosinst.data
   a. Set CONSOLE=/dev/lft0 or CONSOLE=/dev/tty0 according to the system
   b. Set PROMPT=no
   c. Set EXISTING_SYSTEM_OVERWRITE=yes
   d. Set RUN_STARTUP=no
2. Create the signature file
   ```
   echo “data” > signature
   ```
3. Create the floppy diskette
   ```
   ls ./bosinst.data ./signature | backup -iqv
   ```
4. Run the command
   ```
   mksysb /dev/rmt0.1
   ```

The diskette can then be used along with the tape backup. The diskette is put in the target system prior to starting the installation. The BOS install program will then use the diskette file rather than the default /bosinst.data file.

The purpose of the signature file is to verify that this is a bosinst.data diskette.

The unassisted install can by interrupted by typing `<Enter>` when the start symbols `\|/` are on the display.

For the attended install the following commands are carried out:

If classic RS6000 insert the bootable media and turn the key to the service position. Reboot the system.

If PCI and booting from tape is supported then the tape should be inserted and booted from. To check if the RS6000 is capable of booting from tape then run the command
bootinfo -e
if a 1 is returned it is supported if a 0 is returned it is not supported.
If booting from tape is not supported then the bootable CD-ROM should be put into the disk and the system booted using the CDROM.

If the system is booted in install/maintenance mode then the following options should be followed to restore the rootvg.

- Start Maintenance Mode for System Recovery
- Install from a System Backup
- Select the appropriate tape drive

The data is stored in the backup format so individual files can be restored using the restore command mentioned earlier. However you will first have to forward the tape so that it is after the bootable portion of tape.

Other VG Backups (savevg, restvg)

If there are other volume groups to be backed up / restored then the smit screens savevg and restvg should be used.
This also uses the backup format for storing data.

Tape Archive (tar)

The tar command can be used to create archive files. It is not just used for archives and is a popular file format used to transfer files from one system to another, or for the distribution of program files.

The tar command is used to compile a number of individual files to a single file that can be stored on tape (it does not however have to be stored on tape and the archive file can be treat like any other file on the system). The output by default is not compressed although some versions of tar do have a compress option, this is not the case with AIX. The archive can however be compressed by piping the archive to the compress command.

By default the tar command will not backup empty directories, however using the -d command empty directories and special files can be restored.

If access control lists are in use (ACL) then they will not be saved with the tar archive.

The backup file can be created using tar with the -c option:

```
tar -cvf /dev/rmt0.3 /home
```
(using rmt0.3 will allow multiple backups to be stored on the same tape)

To list the contents of the backup:

```
tar -tvf /dev/rmt0
```

To restore files the -x option:

```
tar -xvf /dev/rmt0 /home/stewart
```

The full backup can be restored, or files can be restored individually.

Copy input to output Command (cpio)

The cpio command allows files to be copied into or out of an archive. It does not have the same flexibility as some of the other methods however can still be used effectively. There can be problems with symbolic links and it has no support for the access control lists (ACL).

To backup files using a pipe

```
find /home | cpio -ov >/dev/rmt0
```

or alternatively it can take the list of files from a text file using the redirect

```
cpio -ov <listfile > /dev/rmt0
```

To list the contents of a backup

```
cpio -itv < /dev/rmt0
```

To restore from the backup

```
cpio -idv files < /dev/rmt0
```

Device To Device (dd)

The dd command is used to convert and copy files. It can make a backup that is an exact image of the system and can be used to perform conversions e.g. ASCII to EBCDIC (useful for moving files to / from mainframe systems)

To make a backup of a file to disk

```
dd if=/filename of=/dev/rfd0
```

To convert from ASCII to EBCDIC

```
dd if=file.ASCII of=file.edbcdic conv=ebcdic
```

or the command can be used as a filter (e.g. to convert from lower to upper case)

```
ls | dd conv=ucase
```

Other commands

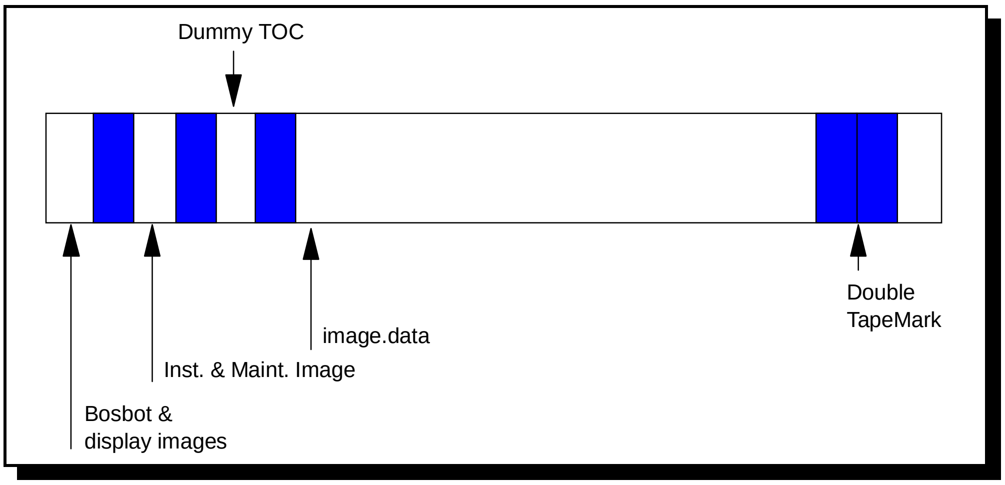
The following commands are useful when making backups:

- tctl Sends commands to a tape device
    - fsf Forwards the tape a number of file marks
    - tctl -f /dev/rmt0.1 fsf 2
    - bsf Rewinds the tape a number of file marks
    - rewind Rewinds the tap
    - tctl -f /dev/rmt0 rewind
    - offline Rewinds and then ejects a tape
- mt Similar to tctl however has different commands
- tcopy Copies a tape
  tcopy /dev/rmt0 /dev/rmt1
- flcopy Copies the diskette to the current directory and then copies it back to a new diskette.

Important Points for Backups

- Backups must be made by root
- Check backups periodically (tapechk)
- Check file systems before backing up (fsck)
- Ensure files are not in use during the backup
  `fuser -u /filename`
  `fuser  -k -u /dev/hd1` (kill any processes using a file system)
- Keep backup media in secure off-site storage
- Label tape with command used & block size
- Test a recovery to ensure the process is working correctly

mksysb Tape Format



Format for a mksysb Tape (standard backup format)

The diagram above shows the layout of a mksysb tape.

To go to the start of the image.data the following command can be run.

```
tctl -f /dev/rmt0.1 fsf3
```

Advanced Disk Management

Disk and storage data is kept in a number of different places. Some of these were already covered in the earlier section on disk management. The following places hold the information.

Volume Group Data Area (VGDA)

The Volume Group Data Area (VGDA) is at the start of each disk which describes the physical volume and logical volumes belonging to the volume group. The information in the VGDA is used to update the ODM when a new volume group is imported to a system. Each VGDA is time stamped with the last time the volume group was updated. Where there are two disks in a volume group the first disk will have two VGDA's however all other disks have only one VGDA. As the VGDA's are essential to the correct operation of the system before the volume group can be varied on (varyonvg) the VGDA's must reach quorum whereby at least 50% of the VGDA's must agree. This can be overridden to force the volume group to varyon however this does not provide for a stable system.

Volume Group Status Area (VGSA)

There is a Volume Group Status Area (VGSA) on all physical volumes in a volume group. It contains information on the state of the physical volumes and physical partitions within the volume group. The important information is whether a physical partition is stale.

Logical Volume Control Block (LVCB)

The Logical Volume Control Block (LVCB) is located at the start of every logical volume. Information is held about the logical volume, including size, type and the last time it was updated.

Object Data Manager (ODM)

The object data manager contains amongst other things information on the volume groups and logical volumes. The database contains a lot of Logical Volume Manager information when the volume group is imported. The ODM has information about the contents of the VGDA, VGSA and LVCB. Logical Devices are always stored as defined and there are no device driver entries in the ODM. This is because logical devices are managed by the logical volume pseudo device driver. There is also no information about logical devices as they don’t have any physical characteristics as such.

Text / Special Files (e.g. /etc/filesystems)

Some of the information is kept in text files such as /etc/filesystems. These are sometimes to keep AIX to be like other UNIX operating systems. There are also a number of special files like those held in the /dev directory. There are special files for the volume groups (like /dev/rootvg) as well as for physical volumes (like /dev/hd1).

The device directories

```
$ ls -l /dev
total 16

crw-rw----   1 root     system    10,  0 24  Nov 11:31 rootvg

brw-------   1 root     system    15,  1 17 Jan 06:13 hdisk0
crw-------   2 root     system    15,  1 24  Nov 11:31 rhdisk0

brw-rw----   1 root     system    10,  8 26  Nov 09:01 hd1
brw-rw----   1 root     system    10,  5 21 Dec 05:13 hd2
brw-rw----   1 root     system    10,  7 14 Jan 05:35  hd3
crw-rw----   1 root     system    10,  8 24  Nov 11:31 rhd1
crw-rw----   1 root     system    10,  5 24  Nov 11:31 rhd2
crw-rw----   1 root     system    10,  7 24  Nov 11:31 rhd3
```

Using the ls command you can see the special device files. The display above shows the volume group, the physical volume and the logical volumes. These files are created automatically by the configuration manager at startup time. In the event of corruption the files can be removed and when the system is next restarted the files will be recreated.

The /etc/vg directory holds pointers to one of the VGDA’s for each volume group. The name of the file reflects the name of the volume group.

```
$ ls -l /etc/vg
total 0
-rw-rw----   1 root     system         0 04  Feb 02:03 vg00538690AA0855BF
```

The files are removed whenever a volume group is varied off, therefore like the /dev files they can be deleted and will be recreated on startup.

Viewing Disk Information

There are a number of commands that can be used to view information on the volume groups, the physical volumes and the logical volumes.

lqueryvg allows you to view the VGDA contents.

```
$ lqueryvg -p hdisk0 -At
Max LVs:        256
PP Size:        22
Free PPs:       0
LV count:       8
PV count:       1
Total VGDAs:    2
Conc Allowed    0
MAX PPs per     1016
MAX PVs:        32
Quorum Setti    0
Auto Varyon     0
Conc Autovar    0
Varied on Co    0
Logical:        00538690aa0855bf.1   hd5 1
                00538690aa0855bf.2   hd6 1
                00538690aa0855bf.3   hd8 1
                00538690aa0855bf.4   hd4 1
                00538690aa0855bf.5   hd2 1
                00538690aa0855bf.6   hd9var 1
                00538690aa0855bf.7   hd3 1
                00538690aa0855bf.8   hd1 1
Physical:       005386901d47dbad 2   0
```

This shows the volume group information as held in memory. If the physical volume name is specified then lqueryvg interrogates the VGDA. The command is undocumented and unsupported the command is used by other high level commands. Details of the command are included for reference in appendix B.

By default the VGDA’s have a maximum number of disks set at 32. However it is very rare that a system would want near this number with disk drives in tens of Gigabytes or even Terabytes this limit is never really reached.

When the volume group rootvg is created it is set with a smaller limit. The command used to create the volume group (mkvg) is called with the -d option to limit the maximum number of drives that can be added. The number of drives that can be subsequently added depends upon the size and number of disks already defined for the volume group at installation. Once the limit is reached the error “Not enough descriptor space” will be seen. At that point it is not possible to add any more disks using the extendvg command.

However generally it is a good idea to limit rootvg to a single disk and add data to different volume groups. This makes recovery easier in the event of problems with a disk.

To remove a physical volume from a volume group the `reducevg` command can be used. If the last physical volume is removed then the volume group will also be removed. The -d option can be used to delete

The LVCB contents can be listed with the command getlvcb

```
getlvcb -AT /dev/hd3
```

This is a low level command used by other commands. It is not intended to be run by users however can give some low level information on the Logical Volume. As this is not documented details are included in appendix B.

The first 512 byes of each logical volume are used for the LVCB. If the LVCB becomes corrupted then it will still be possible to access data on the Logical Volume however LVM commands may stop working.

Moving File Systems

There are a number reasons that you may want to move a File System. There are two different ways these can be needed.

One is to move a Logical Volume from one disk to another within the same Volume Group, this is useful for performance to balance the disk access between multiple drives.

The other is to move a file system from one volume group to another. This might be required if another disk is added to the system and it is put in a new volume group to separate the data from the operating system. The /home directory (holds user data) might need to be moved to the new volume group.

Migrate a LV to a Different Disk in the Same VG

By moving a busy file system from one PV to another the performance of the system can be improved. Another reason may be to move all LV's off a certain disk to free up the disk to be removed from the volume group this may be used if you have got a faster or bigger disk that is to replace an existing disk.

The migratepv command can be used to achieve this.

```
migratepv -l lvname olddisk newdisk
```

or to move all logical volumes

```
migratepv olddisk newdisk
```

Move a File System from One VG to another

The migratepv command shown above cannot be used if the disk to move to is not in the same volume group. The official way of moving file systems is to first backup the file system and then restore it in the other volume group. This can be done by either backing up to tape or to a spare logical volume, however this is slow (particularly using tapes) and / or requires extra space in the current volume group.

Here is an alternative way achieved by mounting the new and filesystems simultaneously. As file systems are to be mounted and unmounted either the computer needs to be rebooted or any users and processes using the file system need to be killed. In this example I am using the /home file system so this must be achieved as root (who's home directory is / and not on the /home file system).

1. Create a new file system on the new volume group however set the mount point to a temporary location (e.g. /mnt/home)
   `smit crfs`
2. Kill any processes that are using the home file system (this should be done now as any changes to the filesystem will not be copied across)
   `fuser -c /home`
   `kill -9` (all PID's from previous command)
3. Copy all files, links etc from old directory to new one
   ```
   cd /home
   cp -hpR * /mnt/home
   ```
4. Check the data has copied correctly
   ```
   $ df
   Filesystem    512-blocks      Free %Used    Iused %Iused Mounted on
   ```

```
/dev/hd4           204800    175720   15%     1174      3% /
/dev/hd2          3366912    329576   91%    36494      9% /usr
/dev/hd9var         24576     20160   18%      503     17% /var
/dev/hd3           131072     65296   51%       70      1% /tmp
/dev/hd1           204800    149464   28%     1230      5% /home
/dev/lv00          139264     80744   43%     1230      7% /mnt/home
```

5. Unmount the /home filesystem
   `umount /home`
6. Change the mount point for the old file system to another temporary mount point in case of problems (e.g. /mnt/oldhome)
   `smit chfs`
7. Change the mount point of the new file system to /home (set to mount on startup)
   `smit chfs`
8. Mount the new file system
   `mount /home`
9. Once everything is OK the old file system can be removed (remember it is now known by it's temporary mount point).
   `smit rmjfs`

Mirroring rootvg

To create a Mirror of rootvg the following steps need to be taken.

1. Add the new disk to the volume group
   `extendvg rootvg hdisk1`
2. Set it so that quorum is not needed to vary the volume group on
   `chvg -ay -Qn rootvg`
3. Make the mirrors of the following LV's on the new disk
   `mklvcopy hd1 2 hdisk1` (/home)
   `mklvcopy hd2 2 hdisk1` (/usr)
   `mklvcopy hd3 2 hdisk1` (/tmp)
   `mklvcopy hd4 2 hdisk1` (/)
   `mklvcopy hd5 2 hdisk1` (BLV)
   `mklvcopy hd6 2 hdisk1` (paging space)
   `mklvcopy hd8 2 hdisk1` (jfslog)
   `mklvcopy hd9var 2 hdisk1` (/var)
   and any other user created file systems.
4. Check that the new BLV mirror is in contiguous PP's
   `lslv -m hd5`
5. Synchronise all new copies
   `syncvg -v rootvg`
6. Update all BLV copies and disk pointers
   `bosboot -a`
7. Make sure NVRAM knows about this new disk to boot from
   `bootlist -m normal hdisk0 hdisk1`

```
   bootlist -m service hdisk0 hdisk1
```

RAID Storage

RAID storage systems are a collection of two or more disks integrated so that the system sees them as being a single disk drive. The acronym RAID originally stood for Redundant Arrays of Inexpensive Disks relating to the fact that lumping a number of cheap disks together can produce benefits. However referring to a device that could cost several thousand pounds (for a very large RAID system) as cheap was not a good name to sell a device. It is now referred to as Redundant Arrays of Independent disks.

RAID does not describe a certain function or use of the devices, instead it refers to a device that contains one or more disk that then presents them to the system as a single disk. There are six different models that can be applied to the RAID device in use depending upon what characteristics are important to the system it is being used on.

The different models are explained below:

RAID-0 Independent Access Array without Parity

This method is for high performance systems. It provides high data transfer rates for large sequential files and a high throughput for random access operations. There is however no mirroring or other methods of preserving data in the event of a disk failure. The total available storage is equal to all the disks in the system.

This method is based on the data striping method striping data across all the disks in the array.

RAID-1 Independent Access Array with Mirroring

This is designed for high reliability providing a full backup of data stored on the disks. This method is based on disk mirroring and two copies of the data are kept on different disks for every write to the disk. The read performance is fast however does not provide an improvement in performance than a single disk (in fact there is a slight degradation in performance however this is not normally noticed). The RAID system only provides half the available disk space for the storage of data.

RAID-2 Parallel Access Array with Parity

This method is a compromise between the reliability of mirroring and keeping the number of disks used down (saving cost). Each chunk of data is written to individual drives however as this is done a parity is written to other disks on the system. This parity is a checksum of the data on the disk. In the event of an error reading a piece of data then it can be reconstructed from the data held on the data disks and the parity disks. This does have a disadvantage in that the disk access for small data files is slower.

This is not often used in favour of the similar RAID-3 method.

RAID-3 Parallel Access Array with Parity

RAID-3 is a slight variation on RAID-2. It operates using the same principle however rather than using multiple parity drives it only has one.

RAID-4 Independent Access Array with Common Parity

RAID-4 is a slight improvement on RAID-3. Larger chunks of data are used (several sectors) and a single drive is used for the parity. Data can be written to different disks at the same time however as the parity is all on a single disk there can be a bottleneck.

RAID-5 Independent Access Array with Distributed Parity

This is a method similar to RAID-4 however instead of having a single drive for parity which can cause a bottleneck, the parity is spread across all the disks in the array. Write performance is often improved by write cache or a write log within the RAID cabinets. There is also load balancing and a high read responsiveness. There can be a performance degradation if a disk fails however in this case the emphasis would be on replacing the drive as quickly as possible and rebuilding the missing disk.

RAID-5 is considered to be the most suitable RAID solution for most commercial applications.

Failures in RAID devices

There are also a number of features included with RAID systems to provide for a failure of a disk.

The following methods are provided and one or more can be implemented in a system:

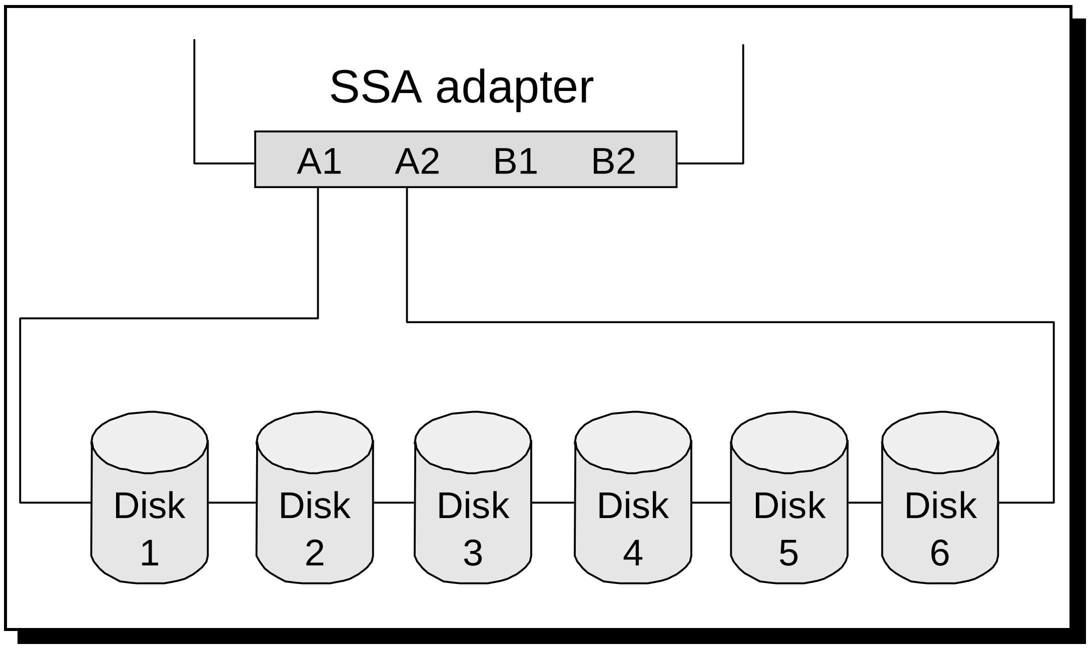
- Hot Pluggable Arrays - Allows a replacement drive to be installed whilst the system is running.
- Hot Spare Disks - Automatically activates a spare disk drive
- Arrays integrated with the CPU - These can ensure that the operator is informed of the failure allowing a replacement to be provided.
- Load levelling Arrays - This allows the data to be rebuilt with minimal impact on the system.

Serial Storage Architecture (SSA) Adapters

The industry standard interface for high performance and fault tolerance of storage devices is to use SSA adapters. The adapter provides for the multiplexing of transmissions to several destinations. Commands are forwarded automatically from device to device along a loop.

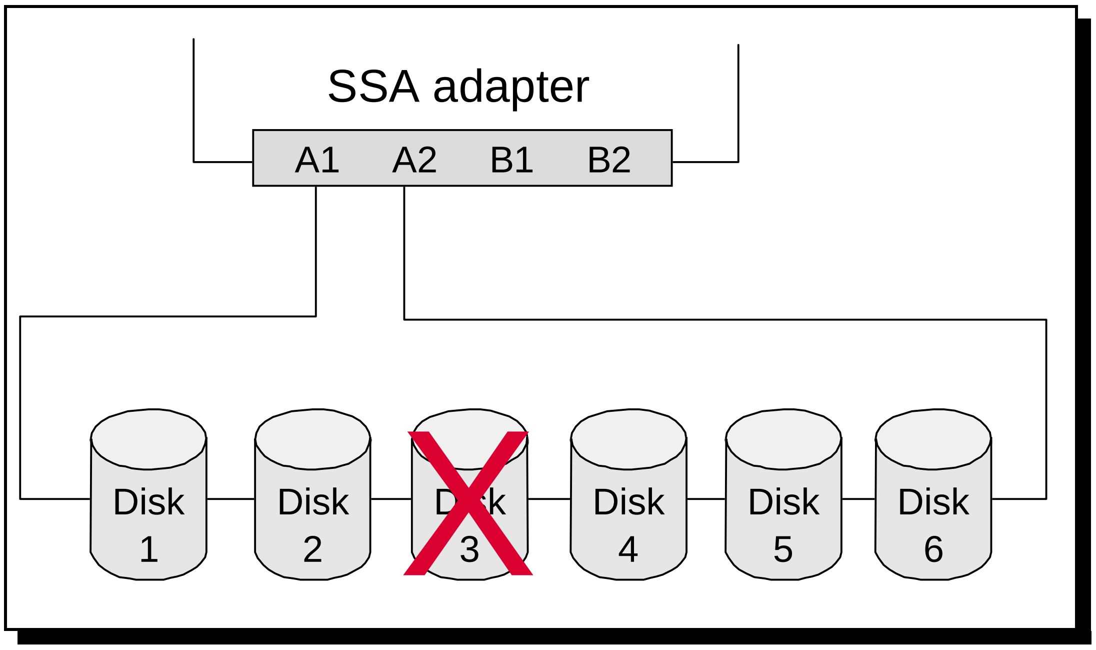
A common adapter provides for two loops each of which can contain a maximum of eight pairs of adapter connectors and a maximum of 48 disk drives.

Using a single loop provides for failure of a single device.



RAID Simple Loop

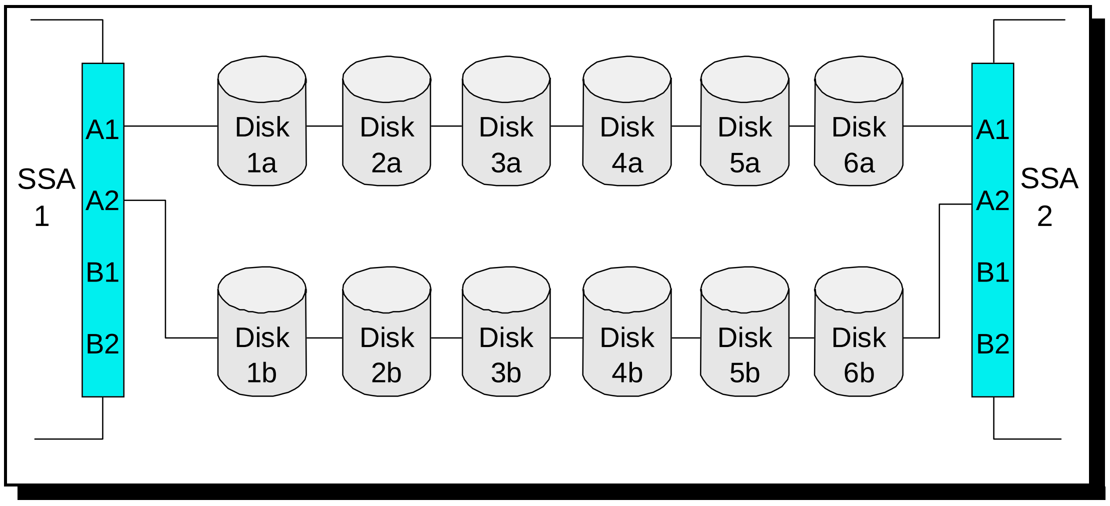
Here data could go through either the A1 or A2 interface and access any of the disks.



RAID Simple Loop - With Failed Disk

In the event of the failure of disk 3 all other disks can still be accessed, 1 and 2 by adapter A1 and disks 4 to 6 by adapter A2. Obviously if more than one disk failed any disks between them would be lost as well.

By adding a second adapter the loop could even handle the failure of an adapter.



High Availability loop.

Using two adapters and both loops up to 96 SSA drives (configured as none-RAID) can be used.

Fixing Problems with Disks

The following commands are low level commands used to fix problems. If you are in a position of needing these commands your system is already in an unpredictable state. They may fix the problems however there are also risks associated with using the commands.

To fix LVCB problems the following steps should be taken:

- Synchronise any mirrors
  `syncvg -v` *`vgname`*
- Find out which LV's have a corrupted LVCB
  `lslv -m` (checks mirroring)
  `lslv` (check information is not corrupt)
- Run synclvodm command
  `synclvodm -v` *`vgname lvname`*
- Check the LVCB
  `getlvcb -AT` *`lvname`*

Alternatively the LVCB can be reconstructed using a copy from another LV.

```
dd if=/dev/goodlv of=/dev/corruptlv bs =512 count=1
```
(copy from a good LV)

`putlvcb -i` (info from lqueryvg -p hdisk0 -At) `-n 1 -t jfslog` *`lvname`*

To remove or activate a PV (done by removing / adding the VGDA).

```
chpv -v {r|a} hdiskx
```

To resync VGDA, LVcontrol blocks and the ODM database

```
synclvodm vgname
```

To remove a physical volume from the VGDA's in a volume group

```
ldeletepv -g vgid -p pvid
```

Some commands will lock a volume group when they are running. If these commands fail they may leave the volume group in a locked state preventing certain commands from being executed. To try and remove the lock the following commands can be tried (in preferred order).

```
chvg -u vgname
```

or

```
putlvodm -K $(getlvodm -v vgname)
```

or

```
odmdelete -o CuAt -q"name=vgname and attribute=lock"
```

If quorum of the VGDA's is not reached then a volume group will not be varied on, or if quorum is lost mid-flight then the VG will be varied off. Obviously in the event of quorum being lost the state of the system is unpredictable however you may want to be able to try and recover certain files prior to restoring the system.

To prevent the VG being varied off if quorum is lost the following command should be run

```
chvg -Qn vgname
```

After running the above command the volume group will only be varied off it loses all VGDA's, rather than if the quorum is 50% or less.

To force a varyon the following command is used

varyonvg -f *vgname*

Note that any disks that have lost their VGDA's will not be available for either of the above methods.

Replacing a failed or failing disk

There are a number of procedures that should be followed depending upon the disk that's having the problems.

Total Disk Failure of a rootvg Disk

1. Replace the failed disk

2. Use the mksysb command is run to reinstall the BOS and rootvg files.
   See the backup section for more details of mksysb.

Total Disk Failure of a non-rootvg Disk

1. Set the failed disk to defined (not available)
   ```
   rmdev -l oldpv
   ```

2. Export the volume group
   ```
   exportvg volumegroup
   ```

3. Check that there is no reference to the file system in /etc/filesystems
   (remove entries from /etc/filesystems if necessary - vi)

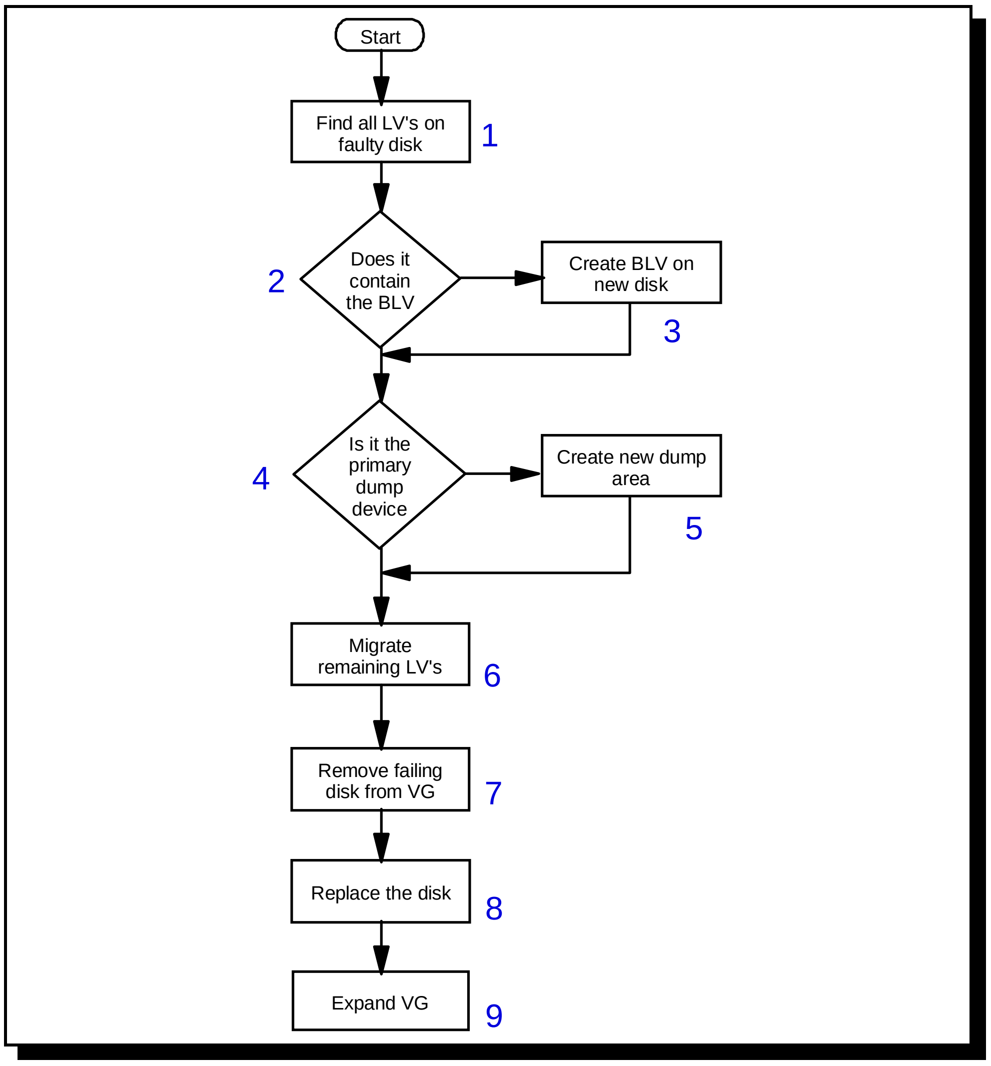
4. Remove failed disk from system and replace with new disk with the SCSI address
   ```
   cfgmgr
   ```
   or
   ```
   mkdev
   ```

5. Create new volume group with the new disk and any other disks that were part of the volume group.
   ```
   mkvg -f -yvgname pvname
   ```

6. Restore the data from backup
   ```
   restore -rvf /dev/rmt0
   ```

Partial Failure of a Disk. - Sufficient space on other disks in the VG to backup the data

This method can be used to minimise the downtime of the computer. This can be used where there are operating system files on the disk. The flow diagram below demonstrates the process to resolve this problem.



Flow Diagram for replacing a failing disk (1)

The following commands relate to the number in the flow diagram.

1.
```
lspv oldpv | grep “USED PPs”
lspv otherdisk | grep “FREE PPs”
```

2. Check if it contains the BLV

3.
```
migratepv -l hd5 olddisk newdisk
```

   Classical RS/6000
```
bootlist -m normal newdisk
bootlist -m service fd0 cd0 rmt0 newdisk
```

   PCI RS/6000

Reboot into System Management Services Menu
Choose “Select Boot Devices”
Choose “Display Current Settings” to see which are configured
Choose “Configure 4th Boot Device” and “select the disk”
press ESCAPE
Reboot

```
mkboot -c -d /dev/oldpv
```

savebase -d /dev/*pvname*

4. Check system dump drive

5. `sysdumpdev -p /dev/sysdumpnull`

6. `migratepv oldpv otherpv`
   `sysdumpdev -p /dev/hd6`

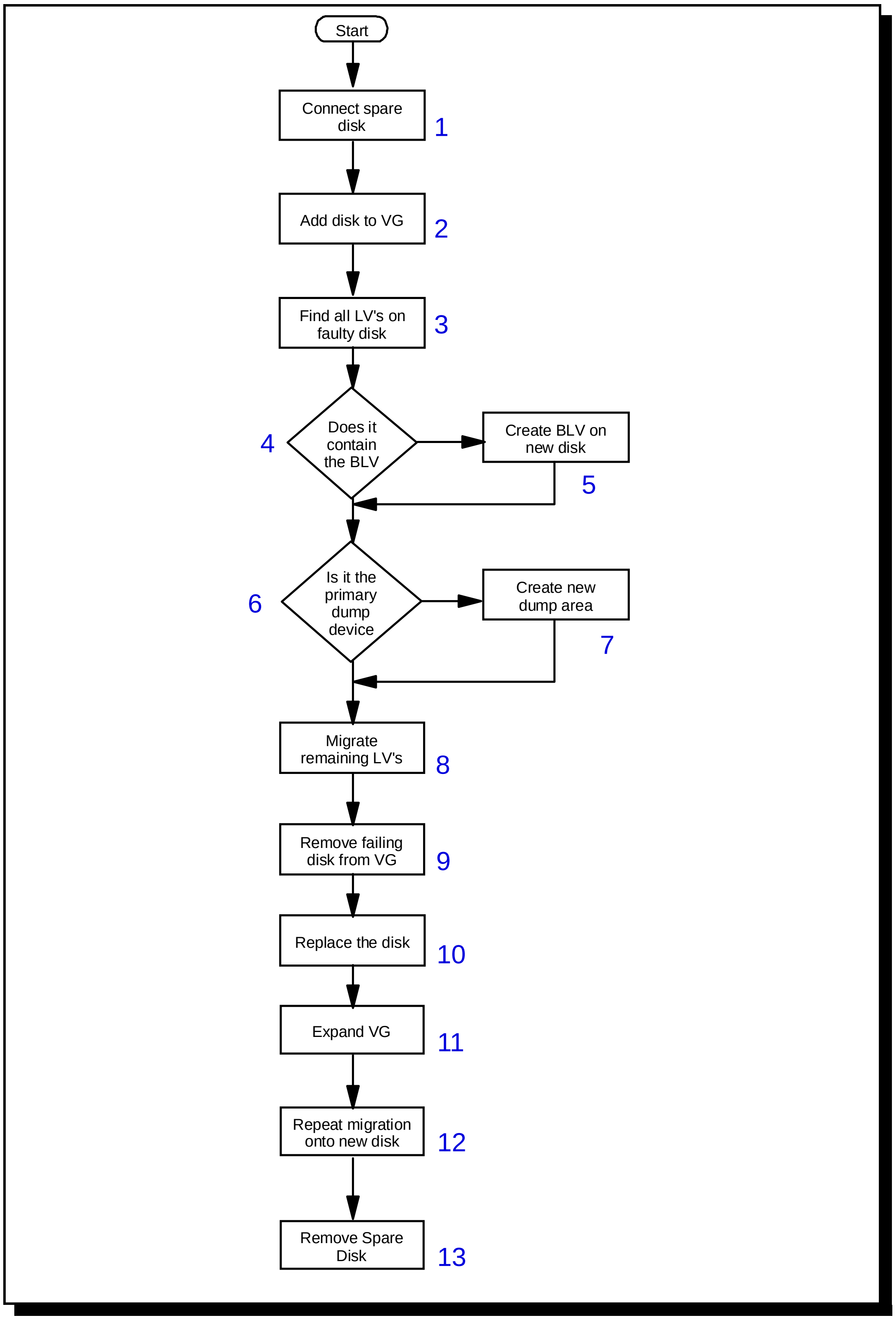
7. `reducevg vgname oldpv`

8. `rmdev -dl oldpv`
   Replace with same disk with the same SCSI ID

9. `extendvg [-f] vgname newdisk`

Partial Failure of a Disk. - Insufficient space on other disks in the VG however a spare disk is available

This method involves adding a spare disk to transfer the data to, then moving the data onto the replacement disk. The following diagram shows the process to follow



Flow Diagram for replacing a failing disk (2)

The following commands relate to the number in the flow diagram.

1. `cfgmgr`
   or
   `mkdev`

2. `extendvg [-f]` *`vgname newdisk`*

3. `lspv` *`oldpv`* `| grep “USED PPs”`
   `lspv` *`otherdisk`* `| grep “FREE PPs”`

4. Check if it contains the BLV

5. `migratepv -l hd5` *`olddisk newdisk`*

   Classical RS/6000
   `bootlist -m normal` *`newdisk`*
   `bootlist -m service fd0 cd0 rmt0` *`newdisk`*

   PCI RS/6000
   Reboot into System Management Services Menu
   Choose “Select Boot Devices”
   Choose “Display Current Settings” to see which are configured
   Choose “Configure 4th Boot Device” and “select the disk”
   press ESCAPE
   Reboot

   `mkboot -c -d /dev/`*`oldpv`*

   savebase -d /dev/*pvname*

6. Check system dump drive

7. `sysdumpdev -p /dev/sysdumpnull`

8. `migratepv` *`oldpv otherpv`*
   `sysdumpdev -p /dev/hd6`

9. `reducevg` *`vgname oldpv`*

10. `rmdev -dl` *`oldpv`*
    Replace with same disk with the same SCSI ID

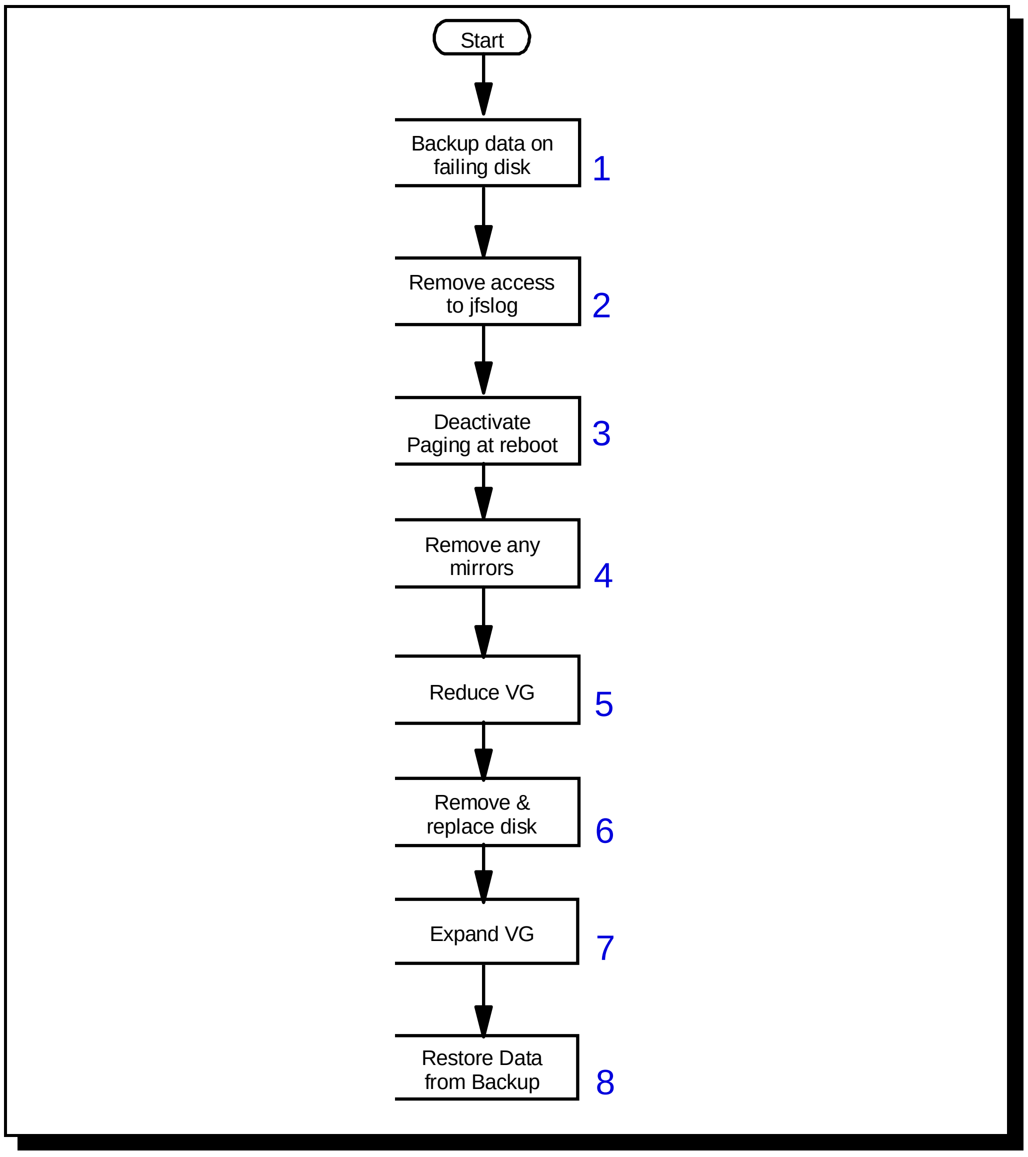
11. `extendvg [-f]` *`vgname newdisk`*

12. See step 8

13. See steps 9 and 10

Partial Failure of a Disk. - Insufficient space on other disks in the VG

This is a slow process to use if previous methods are not suitable.



Flow Diagram for replacing a failing disk (2)

The following commands relate to the number in the flow diagram.

1. `backup -0 -u -f /dev/rmt0` *`/filesystem`*

2. `migratepv -l` *`jfslogname olddisk newdisk`*

3. `chps -a n` *`pagingxx`*
   reboot

4. `reducevg` *`vgname oldpv`*

5. `rmdev -dl` *`oldpv`*

6. Replace disk with same SCSI ID
   `cfgmgr`

7. `extendvg [-f]` *`vgname newdisk`*

8. `restore -rvf /dev/rmt0`

Dealing with corrupted VGDA

The VGDA can be corrupted in a number of different ways.

A common method of a corrupted VGDA is if a disk has been removed however the VGDA's of the existing disks still refer to the removed disk. If a new disk is added, the ODM may allocate it the same name (e.g. hdisk1) however it will have a different PID to the one referred to in the VGDA's of the existing disks.

To get around this the solution is to make the ODM think that the old disk is in the system so that it can be removed from the volume group. Then the new disk can be added to the volume group without any problems.

To first get information on the problem run the lqueryvg and lspv commands.

```
lqueryvg -p hdisk0 -AT
lspv
```
(list the contents of the VGDA)

Comparing the above outputs allows the PVID of the missing disk to be identified. The PVID number are provided with the above command. lqueryvg will show details purely of those in the VGDA, whereas lspv will look at both the VGDA and the ODM possibly showing the disk to be "missing". The missing PVID number is the one that has been removed.

The existing disks should have their PVID's checked to verify the missing PVID.

```
dd if=/dev/hdisk0 count=1 | od -x | grep 000020
```
or
```
lquerypv -h
```

The missing PVID then needs to be added back to the ODM. To do this first we need the correct formats for the ODM entries. This is created using the commands:

```
odmget -q”name=hdisk0” CuDv >> /tmp/hdiskx.add
odmget -q”name=hdisk0” CuAt >> /tmp/hdiskx.add
```

It should be checked that the disk hdisk*x* does not already exist. The files should then be edited with the PVID for the missing disk and added back to the ODM.

```
odmadd /tmp/hdiskx.add
```

You may also need to re-add the PVID attribute to the volume group details.

```
odmget -q”name=vgname and attribute=pv” CuAt >> /tmp/vg
```

Examining the created file there should be an entry “value=PVID” for the disk that is missing. If it is missing you will need to re-add the ODM object with a file containing just the correct details for the missing disk.

Delete the old disk from the system

```
reducevg vgname hdiskx

rmdev -dl hdiskx
```

The new disk can now be added to the volume group

```
extendvg -f vgname hdisky
```

Corruption to the ODM

Another problem with disks is ODM corruption. These problems tend to be relatively straight forward as there are commands that cause the ODM to be updated.

Non-rootvg Volume Group

If the volume group is not rootvg then the ODM can be updated by exporting and then reimporting the volume group.

```
varyoffvg vgname
exportvg vgname
importvg hdiskx
varyonvg vgname
```

rootvg Volume Group

Obviously if there is a problem with rootvg it is not possible to remove and re-add the volume group. There are however a series of commands that can be used to fix the ODM. The commands should be added as a script and then run.

```
PV=/dev/ipldevice
VG=rootvg
      cp /etc/objrepos/CuAt /etc/ovjrepos/CuAt.$$
      cp /etc/objrepos/CuDep /etc/objrepos/CuDep.$$
      cp /etc/objrepos/CuDv /etc/objrepos/CuDv.$$
      cp /etc/objrepos/CuDvDr /etc/objrepos/CuDvDr.$$
      lqueryvg -Lp $PV | awk ‘{print $2}’ | while
read LVname;
      do
            odmdelete -q  “name=$LVname” -o CuAt
            odmdelete -q “name=$LVname” -o CuDv
            odmdelete -q “value3=$LVname” -o CuDvDr
      done
      odmdelete -q “name =$VG” -o CuAt
      odmdelete -q “parent=$VG” -o CuDv
      odmdelete -q “name=$VG” -o CuDv
      odmdelete -q “name=$VG” -o CuDep
      odmdelete -q “dependency=$VG” -o CuDep
      odmdelete -q “value1=10” -o CuDvDr
      odmdelete -q “value3=$VG” -o CuDvDr
      importvg -y $VG $PV             #ignore errors with varyoffvg
      varyonvg $VG
```

Object Data Manager (ODM)

In traditional UNIX operating systems all data and configuration was held in flat ASCII files. IBM developed the ODM to improve on the flat file method. There are still a number of configuration files kept as flat files which are to keep AIX in line with other UNIX systems.

The ODM provides a more robust, secure and shareable resource than the flat file approach. The SMIT interface is one way of interfacing with the ODM which provides a level of error checking. By using SMIT you are prevented from accidentally entering invalid details, compared with flat files where a syntax error could prevent it from working properly.

The ODM is responsible for maintaining the system configuration (device configuration). It provides a reliable, object-orientated database facility for system management. This is done by commands and C-language subroutines to manipulate ODM databases. It is also possible for users to create their own ODM databases.

Components of the ODM

The basic components of the ODM are object classes and objects.

Object Classes (datafiles)
Objects (record within datafile)
Descriptors (field within a record)

Likening the ODM to a database (which it is).

The class is a group of objects with the same definition. This can be thought of as a datafile with field definitions to store data which has something in common.

An object is a member of a defined object class. This can be likened to a record within a datafile.

An object class is made up of one or more descriptors. These can be likened to fields within a record. When an object is added to an object class a value is assigned to the field.

Taking the PdAt object class:

```
PdAt    {
        char uniquetype[48];
        char attribute[16];
        char defit[256];
        char values[256];
        char width[16];
        char type[8];
        char generic[8];
        char rep[8];
        short nis_index;
        };
```

An example of an object in the PdAt:

```
PdAt  {
      uniquetype = “tape/scsi/8mm”
      attribute = “block_size”
      defit = “1024”
      values = “0-245760,1”
      width = “ ”
      type = “R”
      generic = “DU”
      rep = “nr”
      nls_index = 6
```

Accessing the ODM

Whilst the normal method of accessing the ODM is through SMIT however there are a number of direct methods of accessing the ODM. These provide a great deal of flexibility. The commands are:

`odmcreate -p -c -h file`
Creates object classes required for applications that will use the ODM database.

`odmshow object_class_name`
Display an object class definition

`odmadd file`
Adds a new object to an object class. This is acts like an append so if the object already exists it will create another instance.

`odmget -q criteria object_class_name`
Retrieves objects from an object class. These return the output in stanza format.

`odmdelete -o object_class_name -q criteria`
Deletes all objects that meet a specific criteria, from the object class. If no criteria is specified it will delete all objects in that class.

`odmchange -o object_class_name -q criteria file`
Changes all objects that meet a specified criteria.

`odmdrop -o object_class_name`
Changes all objects within an object class that meet a specified criteria.

The ODMDIR variable locates the file. This can point to the system ODM in /etc/objrepos or to a user ODM.

An example of how to change an ODM entry

```
$ odmget -q“uniquetype=tape/scsi/8mm and attribute=block_size” PdAt >file1
$
$ vi file1
PdAt:
      uniquetype = “tape/scsi/8mm”
      attribute = “block_size”
      defit = “1024”
      values = “0-245760,1”
      width = “”
      type = “R”
      generic = “DU”
      rep = “nr”
      nls_index = 6

changing the defit value to 512

$ odmdelete -o PdAt -q“uniquetype=tape/scsi/8mm and attribute=block_size”
$ odmadd file
```

In this case the entry has been deleted and then added, the alternative is that the odmchange command could be used in place of the last two commands.

As part of the criteria the equality can be test i.e. ‘=’ or similarity being ‘like’. The boolean operators that can be used are:

=	equal
!=	not equal
>	greater than
<	less than
>=	greater than or equal to
<=	less than or equal to
like	similar to; finds path names in character string data

The characters ‘*’ and ‘?’ can be used as wildcards. They take their normal meaning.

A number of commands will also automatically query or update the ODM. These are mainly the commands associated with devices and filesystems e.g. lsattr, mklv etc.

The ODM does NOT manage the following which are kept in flat files:

- File system Information (/etc/filesystems)
- Security and user details (/etc and /etc/security)
- Print spooler config (/etc/qconfig and /var/spool)

ODM database files

There are a number of separate database files for the ODM. The following files are:

- Predefined device information - PdDv, PdAt, PdCn
- Customised device information - CuDv, CuAt, CuDep, CuDvDr, Config_Rules, CuVPD
- Software information - history, inventory, lpp, product
- SMIT menus - sm_menu_opt, sm_name_hdr, sm_cmd_hdr, sm_cmd_opt
- Error log, alog and dump information - SWservAt
- Network Install Manager - nim_attr, nim_object, nim_pdattr

Information for devices which can be configured are held in the predefined database. Pd, the class of available devices is PdDv. The attributes are contained within the class PdAt.

Configured devices are held in the Customised database Cu. The class containing the configured devices is CuDv and the attributes in CuAt.

The Config Manager (cfgmgr) is used to change devices from predefined to customised.

Location and contents of ODM repositories

/etc/objrepos
Here the customised devices object classes and the SWVPD (Software Vital Product Database) for the / (root) part of the installable software product. The directory also contains links to the predefined devices object classes and the SMIT menu object classes.
The ODMDIR points here by default

CuDv	CuAt	CuDep	CuDvDr	CuVPD	Config_Rules
history	inventory	lpp	product	nim_attr	nim_object
nim_pdattr	SWservAt				

/usr/lib/objrepos
This contains the predefined devices object classes, the SMIT menu object classes and the four object classes used by the SWVPD for the /usr part of the installable software product. The object classes in this repository can be shared across the network by /usr clients. This can be shared between AIX clients only.

PdDv	PdAt	PdCn	history	inventory	lpp
product	sm_menu_opt	sm_name_hdr	sm_cmd_hdr	sm_cmd_opt	

/usr/share/lib/objrepos

This contains the object classes used by the SWVPD for the /usr/share part of the installable software product. The components that are not AIX dependant.

history	inventory	lpp	product

Error Logging

Error logging is used to provide information on hardware, software and operator errors. There are a number of different logs including security information.

The logging of errors are provided for as part of the base system. There are a number of errlog subroutines including errsave (kernel), the /dev/error (device driver), the errdemon and the errstop command.

Some of the error log commands are delivered in an optional installable package called bos.sysmgt.serv_aid. This includes the commands errpt and errclear.

The process begins when an operating system module detects an error. The error detecting segment of code then sends error information to errsave or errlog. The information is then written to the /dev/error special file. A time stamp is added.

Depending upon the label of the error message the daemon collect additional information from other parts of the system. The error information is collated together into a template from the repository. Most entries in the error log are hardware, or software problems however informational messages can also be stored. The errlogger command allows messages to be written to the error log. One use may be to enter information when a hardware or software change is made.

The errpt -t command can be used to view the Error Record Template Repository.

The errupdate command allows root to update the Error Record Template Repository. It modifies the characteristics of existing entries as well as new one.

The error description includes probable causes recommended actions and detail data. This is kept in the catalog /usr/lib/nls/msg/$LANG/codepoint.cat
The errmsg command can be used to display the message with their identifiers e.g.

```
# errmsg -w ALL -z /usr/lib/nls/msg/en_US/codepoint.cat
SET E
1000 "EQUIPMENT MALFUNCTION"
1001 "CONTROL UNIT MALFUNCTION"
1002 "DEVICE ERROR"
1003 "CPC HARDWARE FAILURE"
1004 "TIME OF DAY CLOCK FAILURE"
1005 "BACK-UP RESOURCE FAILURE"
1006 "OPTICAL SYSTEM BUS FAILURE"
1007 "SERIAL LINK SWITCH ERROR"
1008 "MULTIPLEXER PROBLEM"
```

The errinstall command installs messages in the error logging sets. This is an aid for root or to replace messages. It should not be used by software developers who should instead contact IBM to have a new template added.

Error reports can also be generated, viewed or cleaned using the SMIT fastpath error.

```
                                  Error Log

Move cursor to desired item and press Enter.

  Generate Error Report
  Change / Show Characteristics of the Error Log
  Clean the Error Log















F1=Help             F2=Refresh          F3=Cancel           F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Viewing Error Reports (errpt)

The errpt command can be used to view the error logs.

To view a summary report of all errors in the errlog

```
errpt
```

To view the summary report using a template

```
errpt -t
```

For a detailed report errors

```
errpt -a
```

A number of other options can be used.

-a	Print a detailed listing.
-c	Concurrent mode. Display entries as they arrive
-t	Print error templates instead of error log entries
-d	Select error classes (H - hardware, S - software, O - operator)
-s startdate	All entries after start date
-e enddate	All entries before end date
-j #	Include specific error id's
-k #	Exclude specific error id's
-l seq	Select unique log entry sequence numbers
-m mach	Specific machine id
-n node	Specific node id's
-T	Select specific error types

```
# errpt
IDENTIFIER TIMESTAMP  T C RESOURCE_NAME  DESCRIPTION
2BFA76F6   0204054900 T S SYSPROC        SYSTEM SHUTDOWN BY USER
9DBCFDEE   0204055100 T O errdemon       ERROR LOGGING TURNED ON
192AC071   0204054700 T O errdemon       ERROR LOGGING TURNED OFF
2BFA76F6   0204040000 T S SYSPROC        SYSTEM SHUTDOWN BY USER
9DBCFDEE   0204041300 T O errdemon       ERROR LOGGING TURNED ON
192AC071   0204035800 T O errdemon       ERROR LOGGING TURNED OFF
2BFA76F6   0204020200 T S SYSPROC        SYSTEM SHUTDOWN BY USER
9DBCFDEE   0204020400 T O errdemon       ERROR LOGGING TURNED ON
192AC071   0204020100 T O errdemon       ERROR LOGGING TURNED OFF
C5C09FFA   0202092900 P S SYSVMM         SOFTWARE PROGRAM ABNORMALLY TERMINATED
C5C09FFA   0202092900 P S SYSVMM         SOFTWARE PROGRAM ABNORMALLY TERMINATED
5BF9FD4D   0127150400 T H tok0           PROBLEM RESOLVED
AE57B46D   0127150300 P H tok0           UNABLE TO COMMUNICATE WITH DEVICE
5BF9FD4D   0126165300 T H tok0           PROBLEM RESOLVED
AE57B46D   0126165200 P H tok0           UNABLE TO COMMUNICATE WITH DEVICE
5BF9FD4D   0126155300 T H tok0           PROBLEM RESOLVED
```

```
# errpt -a
---------------------------------------------------------------------------
LABEL:          REBOOT_ID
IDENTIFIER:     2BFA76F6

Date/Time:       Fri  4 Feb 05:49:23
Sequence Number: 68
Machine Id:      005386904C00
Node Id:         overton
Class:           S
Type:            TEMP
Resource Name:   SYSPROC

Description
SYSTEM SHUTDOWN BY USER

Probable Causes
SYSTEM SHUTDOWN

Detail Data
USER ID
           0
0=SOFT IPL 1=HALT 2=TIME REBOOT
           0
TIME TO REBOOT (FOR TIMED REBOOT ONLY)
           0
---------------------------------------------------------------------------
```

Entries in the log can be cleaned up using the errclear command. This should be included in cron so that it is run automatically.
Software / operator errors are purged after 30 days
Hardware errors usually purged after 90 days.

Using the errdemon

The errdemon can be used to show the log attributes.

```
# /usr/lib/errdemon -l
Error Log Attributes
--------------------------------------------
Log File                /var/adm/ras/errlog
Log Size                1048576 bytes
Memory Buffer Size      8192 bytes
```

The log file can be moved using the command

`/usr/lib/errdemon -l` *`/filename`*

To change the log file size

```
/usr/lib/errdemon -s Logsize
```

To change the memory buffer size

```
/usr/lib/errdemon -B buffersize
```

Error Log Analysis

There is a program that will automatically analyse the error log for hardware errors. The diagela program analyses the error log and mails root with details.

The program is invoked / revoked using the command

```
/usr/lpp/diagnostics/bin/diagela ENABLE  / DISABLE
```

There is another error reporting tool called syslogd. This runs via the SRC. It reads datagram sockets and sends messages depending upon a set of rules defined in /etc/syslog.conf

It logs system messages and can send them to users and / or other systems.

Security

Whilst I have put security down towards the bottom of the document this does not indicate that it should be left as an afterthought when planning the implementation of an AIX server. In fact security should be one of the foremost thoughts at all stages of setting up your AIX machine. The reason that it has been relegated so far down this document is that to be able to implement a good security policy on a machine requires a good knowledge of the fundamentals of AIX and UNIX. Therefore this section will lean heavily on details learned earlier in the document.

Security of AIX computers could fill an entire book in itself so obviously I couldn’t put everything in here. This does however give a basic introduction to security and how the techniques, and tools can be used to provide additional security on an AIX computer. Hopefully this will provide sufficient information to be able to investigate other sources of information.

Why Do We Need Security ?

One of the most important steps in any task is to identify why you are doing it. Rather than just saying we need to make a system secure you need to consider what is meant by secure, what risks there are associated with any data that's available, what impact your security measures will have on your users. Without first considering any of these factors how else will you know if you've met your goal of making a system secure.

Security Requirements

After establishing why security is to be implemented you should consider the aspects of security that are required. The main security requirements are:

Authorisation - Only allow those that need access to the data
Authenticity - Verifying they are who they say they are
Privacy / Confidentiality - Ensure personal information is not being compromised
Integrity - Ensuring that the data has not been tampered with
Non-repudiation - Confirmation that data is received. The ability to prove it in court.
Availability - Ensure that the system can perform it's required function

Imposed Requirements

Some security requirements are not ones that are directly under your control but are instead imposed upon you. These may be legal requirements (e.g. Data Protection Act 1998), compliance with standards (e.g. ISO 7984-2 International Standards Organisation Security Standard), or corporate policy.

Some of these standards are very vague (e.g. the Data Protection Act just specifies that appropriate security should be in place) whereas some may be more specific (e.g. a corporate policy may insist on a minimum length of passwords etc.).

Knowing the Enemy

Before being able to effectively protect a computer system you need to know who it is that is trying to attack your systems and what they are trying to do. I have shown some examples by answering a few questions about those who could potentially attack a computer system.

1. Who wants to?
2. Why are they doing this?
3. What do they try and achieve?
4. How do they do it?

Hackers, Crackers and Phreakers

These words are commonly used when referring to security attacks, however the meanings are often misinterpreted or understood. I have taken these in order of simplicity to understand:

Phreakers - Also known as Phone Phreakers, this term originates from what could be considered to be the earliest form of attacks against electronic systems. It's earliest for was to bypass the systems used in telephone systems allowing free or reduced price international phone calls. One of the earliest forms of this was when the American pay phone system used a certain frequency signal to indicate that a coin had been placed in the phone. It was discovered that the frequency of the signal was 2600 Hz, which was also the same frequency emitted from a toy whistle distributed with a popular make of cereals. By blowing the whistle into the phone when a request was made for payment the Phreaker could fool the operating into thinking that money had been deposited in the pay phone. Whilst this has been secured for the US phone networks there are reportedly still phones in developing countries where similar techniques can be used.

Crackers - These are people that gain unauthorised access to a computer. When people refer to hackers in the new meaning of the word then they are really referring to crackers.

Hackers - Using the traditional meaning of the word Hacker is not meant to imply any kind of illegal or immoral activities. The true meaning is of a computer enthusiast that understands the inner workings of a system and uses that knowledge to "hack" together programs etc to perform a function. This was different to the traditional techniques or programming that are designed to follow a set structure and procedure to produce a finished piece of software. Due to incorrect use by the press the word hacker has now come to take on two meanings. One is it's original meaning and the other is that of anyone who tries to penetrate a computer (crackers) or those who cause intentional disruption or damage (none-physical) to computer systems.

Throughout this book I will normally refer to the perpetrator as an attacker, regardless of which of these categories she comes under, however where I do refer to a hacker I will normally mean the newer of these meanings.

Whilst some may object to my use of the word hacker, my justification is to turn to the definition held in the Oxford English dictionary which describes the popular use of the language and is considered a definitive guide to the English language:

"Hacker - computer enthusiast, esp. one gaining unauthorised access to files"
The Oxford Popular Dictionary, Parragon, 1995

The Stereotypes - Why be a Hacker?

By understanding the reasons for the attacks gives a basis for what protection can be used to protect the data. I have therefore taken a few examples of stereotypical and maybe not quite so expected attackers. This is by no means complete however highlights that there are different reasons that someone would want to attack your system.

Just for Fun - This “School Boy Hacker” is perhaps the first thing that people think about when talking about “Hackers”. The school boy attacker (can be female) is typically someone in further or higher education that uses the college or universities computer facilities to attack another computer over the Internet. Whilst there are indeed a number of attackers that match this description it is important to recognise that these are not the only type of hackers. The “school boy” attacker typically has limited resources and normally does it, just for fun; or to prove their intelligence etc. However they may be part of a larger group united using the Internet. Whilst many do not intend to commit malicious damage they may discredit your company name, they may cause accidental damage, and may open the door for others.

Commercial Espionage / Sabotage - Whilst Espionage normally congers up the image of James Bond fighting a host of bad guys before running off with Pussy Galore the reality is much less dramatic. There is potentially a risk from competitors wanting to gain a competitive edge. For example if you are bidding for a contract and your competitor is able to find out details of your bid, they could easily undercut you and win the contract. Alternatively by putting your web page out of action, customers could be encouraged to try the competition. This kind of attacker normally has a lot of resources, both financial and in man power, at it’s disposal and has very specific targets. If your organisation is involved in military contracts there may be a real Ernst Stavro Blofeld trying to steal the technology to take over the world.

Fighting a Cause - Other groups that may wish to attack your company are those that are fighting for a cause or defending a belief. Whilst there are a number of obvious extremist groups such as the extremist animal rights groups this could equally apply to less controversial areas. The Internet was has origins that did not tolerate commercial activities. Whilst now it is normal to see the Internet being used as a commercial shopping centre this does infuriate a minority of people. These groups tend to go for the larger high profile companies however could attack any company that they see to be an easy target.

Disgruntled Employees - So far I have mentioned attackers external to the organisation, however it is generally believed that the greater risk lies from employees within the organisation. These could already have authorised access to a computer, and already be inside the firewall. They could then use that access against the organisation and exploit other holes in the system. Whilst these people can have different motives one of the most obvious is for someone that has been fired, disciplined or who is not satisfied with their current standing in the organisation. Defending against the internal employee can be more challenging as methods need to be found to limit access without preventing others for performing their job. To tighten up security to the point where employees cannot do their job properly is an indirect Denial of Service attack.

Unintentional User Error - Whilst normal users may not be trying to cause any damage to the system it’s possible that they could cause some accidental damage to data. By limiting a users access user errors can be contained to a reasonable extent. This could be in the form of a programming error as well as incorrectly typing instructions into a program.

Types of Attacks Against Systems

There are a number of different types of attacks that take place. These may be different depending upon the services you offer or the type of attacker that is targeting you. These are areas that are looked at later to determine methods of protection. This list is not intended to be a complete list however it does give an idea of what areas to focus your attention on. New methods are still being developed and a security administrator has to ensure that they don’t get left behind.

Reading Data - Typically associated with espionage or theft computer systems often contain information that needs to be kept confidential or secure. This could vary from emails discussing the price of a bid for a project to personal information or bank details. The disclosure of this information could severely damage the company or even have legal implications. In the UK the storing of personal data is covered by the Data Protection Act (1988). The principle of the act states that personal data shall “Be surrounded by proper security.” See http://www.dataprotection.gov.uk/ for more details.

Changing Data - Potentially more serious is that an attack could gain sufficient access to be able to update data. This could be for sabotage, as a means of discrediting the organisation or something as simple as leaving a calling card. One of the biggest risks is that data could be modified and not noticed. The cases that tend to get a high profile in this area are where attackers replace web pages with their own modified versions. A notable example of this is that the Labour Party’s web page was replaced with a altered one in November 1997, followed by the Conservative Party’s web page in April 1997.

Denial of Service - Denial of Service (DoS) attacks are where the attacker disables, or makes unusable the services provided by the system. An earlier DoS attack was the “Ping of Death”. By creating a ICMP echo command that was larger than the maximum allowable size a computer could be made to fail. Many of the DoS attacks in the past have been removed by the fixing of bugs there is now a new threat known as Distributed Denial of Service. The best known of these was the attacks against Altavista and Yahoo in early 2000. The distributed Denial of Service attack works by the attacker, or more likely attackers, planting trojan horses on lots of different machines. When these trojan horses are triggered simultaneously they mount an attack directly against a single system. The combined effect of thousands of simultaneous attacks prevents the system from operating. This form of attack is getting more and more sophisticated and security administrators are devoting more resources to tackling this kind of problem.

Access to Computer - Whilst for some systems you may allow other users onto your system sometimes these user accounts could come under attack. The computer may not contain any confidential material and the user may not be able to write to any data however they could still use your system to cause damage. If someone manages to attack a computer that borders between a secure and insecure network then they could use your machine as a method of traversing between the two networks. Another technique to use your computer to attack another is in the distributed denial of service. The attacker could plant a Trojan horse on your computer so that when triggered it attacks another computer. This could be potentially embarrassing if someone found that systems belonging to your organisation were used to commit one of these crimes. Indeed it could even look as though it was someone from inside your organisation that perpetrated the crime.

Methods of Attacking a System

In the past the hackers were all intelligent people with a great deal of understanding of how computers work. They could identify bugs in systems and then use their knowledge of computers to exploit the bug. Whilst there are still a lot of hackers that can do this there is also another type of attacker that waits until someone else has found a way into a computer and then uses the same technique. They can just take programs and scripts written by hackers and run them against systems hoping to find a way in. You may notice that I have not referred to them as hackers as they do not have the knowledge of computers to back up their curiosity, they are sometimes more appropriately referred to as “Script Bunnies”. Here is a list of some of the techniques used to gain access. This purely gives an idea of some of the methods used and does not list all the available methods.

Password Guessing - Some systems or services may have default passwords when they are installed. Some attackers will just try some standard userid’s and passwords in the hope they will be lucky and find a easily guessed password. Some people may set the password to be the same as the userid which is one of the standard things the person will try. This method relies on users or administrators not using secure passwords.

Social Engineering - This technique works by working on the failings of people rather than the insecurity of computers. One technique is to phone a help desk pretending to be an employee and trying to get them to give you the password over the phone, or indeed the other way around pretending to be a system administrator and asking the user for their username and password. Another technique, known as shoulder surfing, is where the attacker would stand behind someone whilst they typed their password into the keyboard, watching what keys are pressed. These are techniques that depend upon the security training (or rather lack of it) that the employees have.

Trojan Horses - Trojan Horses are programs planted in a computer which appear to be harmless. These could be left by a legitimate user or placed on a previously hacked site that is used to distribute software, they could also be sent as E-mail with some kind of useful tool.

When a trigger is activated then the hacker can gain access to computer or get the program to run a certain command.

Virus - A virus is a programs designed to self replicate itself. These are not normally considered to be a threat to UNIX computers due to the way that the UNIX security model works, however it is possible to write a Virus for UNIX they are not generally found in the wild. However if running a UNIX computer as a mail server or a file server you may want to consider running virus checkers against executable code.

Software Bugs - If software has not been written correctly there are sometimes bugs that can lead to a security exposure. In particular a program designed to handle a certain amount of data can be broken by bombarding it with too much data. This is sometimes done by overrunning the buffers with data causing data to be stored in memory not allocated for that purpose. This could result in the system crashing (e.g. the Ping of Death) or in a trusted program providing access to the system.

Address Spoofing - In trusted environments it is sometimes configured that other computers known to be safe are allowed access to that computer without any further authentication. Whilst this makes administration easier it does have potential problems in that another computer could masquerade as one of these trusted computers. By configuring a computer with the same IP address as a trusted one, which is down or has been forced down, the attacker would have access to other systems the same as if they had been given official access on the trusted server. In a lot of environments it is therefore considered to be bad practice to enable services that rely on these trusted computers

This is by no means a comprehensive list of methods however it does give an idea of areas that are vulnerable.

Security Risks

The first thing to consider with security is why are you implementing security and what does your organisation stand to lose if it is not implemented properly. By assessing the risks in this way a picture can be generated as to what level of security is needed.

This needs to be considered across all the different environments and systems covered by the organisation. This may even mean expecting certain standards from other companies who may host part of your computing services or who provide a service on which you are reliant.

Part of the security policy should also identify any systems that are more vulnerable and therefore need a higher level of security than others. Whilst in the ideal world every system would be completely secure and safe from attack this is not the case in reality. The resources required and the effort involved in trying to secure all systems to the same level may leave exposures or cause such an impact as to prevent the system operating correctly.

Categorising Systems Based on Function

An example of how different systems can be categorised is shown below. In this scenario each system is categorised under three headings. These are red for high risk, yellow for medium risk and green for minimum risk. These may be very different depending upon the requirements of your organisation.

Red - High Risk	**Yellow - Medium Risk**	**Green - Low Risk**
Internet Web Server	Intranet Web Servers	Users Personal Computers
Firewalls	Internal Database Servers	Print Servers
Secure Servers		None Essential Services (e.g. messaging systems)
Payroll / Financial Systems		

Categorising Systems Into Different Risk Categories

The red high risk systems are firstly those that customers will see such as the Web Server; those that protect the internal networks from the external networks such as firewalls (see later for details); those carrying sensitive information such as the Secure Servers and any systems that hold information that could damage the company if they were released such as the payroll and financial systems.

The yellow medium risk systems are those on which the business relies however are not accessible to the general public. These are usually internal servers that are protected from outside by firewalls. The systems under this category should still be considered a high risk if a business function is completely reliant upon them. For example a call receipt desk that handles customer orders would be unable to operate if it could not access the order system.

The green low risk systems are those where an attack would only have limited effect. For example a PC on a single persons desk or a print server; or a service that the organisation does not depend upon to continue to function, e.g.. a peer to peer messaging system that in the event of failure could be bypassed by using the telephone. When systems are put into the low risk category then the overall risks must be considered. For example do users PC’s contain confidential material or if someone was able to break into a print server or personal computer could they then use this to inflict more serious damage.

These categories must also be used in conjunction with the organisations security policy (see later). As an example you may decide that because of confidential information stored on a users laptop computer you really want to upgrade this to a medium risk, however the security policy states that all medium risk systems must be stored in a secure room with no user access except when the system needs repair. By putting the laptop in the medium risk category you have just removed it's usefulness as a portable system, indeed the user may not even have access to it. Therefore a degree of flexibility needs to be included when the risk analysis is performed.

Identifying the Different Risks to a System

After identifying which systems are at risk, it is also important to consider what the risks to that system are. This should be achieved by considering the impact of the different types of attacks. I have used an example below of some of the high risk systems to determine the impact of different types of attacks. These are shown in the table below:

	Web Server	**Firewall**	**Secure Servers**	**Payroll / Financial**
Reading Data	Low	Medium	High	High
Changing Data	High	High	High	High
Denial of Service	High	High	High	Medium
Normal User login	Low	High	High	High

Impact of Different Types of Security Breaches

As you can see from the table certain types of breaches can have a more damaging effect. Looking at the Web server someone being able to read data is pretty low as the data stored on the web server is generally available to anyone, however if someone was able to change the data or prevent the service from working (Denial of Service) then the impact, and reputation would be severely hit. Whereas on the firewall every type of breach can be serious as it could then be used to target a less secure system inside of the internal network. Obviously though this is also dependant upon the data for which they can read. It is obviously much worse if the user is able to read the information that is categorised as confidential than say a document giving directions to the location of the next social event.

By identifying the impact a better strategy can be developed on where to invest the available resources.

Assessing the Data

As well as identifying which systems hold any sensitive data the actual data itself should be categorised depending upon it's sensitivity. This is particularly useful in deciding which users on a system should be able to access what files. If you limit the access of individual users then this limits the damage that the user can do and if the userid becomes compromised limits the damage that the attacker can do.

More useful than categorising by sensitivity it is useful to group the data into similar access requirements. This is likely to be grouped by departments that handle a certain function.

Then against each of the categories it should be identified who should have what access. Once method of doing this is known as CRUD analysis. This is simply a case of

Who can **c**reate the data?
Who can **r**ead the data?
Who can **u**pdate the data?
Who can **d**elete the data?

You may however set this up differently depending upon what policies are available with your system. For example you may be running DB2 or Lotus Notes which can have different access methods than the standard UNIX authorisation of read, write, execute.

This analysis can then be used to create groups with the appropriate access. This also has the advantage of making access easier to manage by associating users with groups rather than having to setup authority on an individual basis.

Security Policy

Having analysed the security requirements in the previous section you can now set about devising your own security policy. It may be just a case that you decide to adopt the corporate security policy, although even then you may think of certain things that you can add to make this even more secure.

If however you are creating your own security policy there are a number of factors you need to consider.

You should ensure that you have covered any of the principles Authorisation, Authenticity, Privacy / Confidentiality, Integrity, Non-repudiation, Availability as they apply to your system. However also consider how this is going to be implemented by the users and system administrators. If a security process is hard to implement or restricts someone from doing their job then you may find that the process gets abused.

When setting up a security policy you should also consider how this can be enforced and audited.

Closing the Holes

There are a number of possible security exposures on any computer. Whilst it is practically impossible to fully secure a computer whereby the computer can still fulfil a purpose there are a number of steps that can be carried out to minimise the exposures. These are often referred to as security holes or back doors which need to be closed.

Also there are a number of steps that can be taken to try and identify if a machine is under attack or indeed if it has already been penetrated. By regular monitoring of suspicious activities then steps can be taken to limit any damage and to secure against further attack.

Physical Security

Most physical security principles are fairly obvious. All production servers should be kept in a secure machine room with restricted access. No matter what other procedures are in place if you gain physical access to most RS/6000's then the system can be accessed by inserting a AIX CD-ROM and rebooting in maintenance mode.

There is one area of physical security that is less obvious than others and that is access to the physical LAN. Many internal networks implement DHCP for the allocation of IP addresses, which makes accessing a network as simple as connecting a portable computer to a network point. This also makes internal network access to a hacker as simple as connecting their portable computer to a convenient network point.

Also using telnet and FTP passwords are sent unencrypted across the LAN so anyone able to put a sniffer or LAN trace tool on the LAN is able to see these unencrypted passwords.

User Authorisations

The normal user authentication is based upon the user being able to provide the correct username and password. The username is not something that has to be kept secret as it is readable by anyone on the system however the password is encrypted and should only be known by the user. The algorithm used to encrypt the password is a one way password which cannot be reversed. Instead when a user enters their password it is encrypted using the same algorithm and compared against the original.

Traditionally in UNIX the password was kept in the /etc/passwd file which was readable by all users on the system. AIX along with other UNIX systems uses a shadow password file.

The following extract shows how this is implemented.

The /etc/password file still contains username details however has an asterix (*) where the password should normally be:

```
root:!:0:0::/:/bin/ksh
daemon:!:1:1::/etc:
bin:!:2:2::/bin:
sys:!:3:3::/usr/sys:
adm:!:4:4::/var/adm:
uucp:!:5:5::/usr/lib/uucp:
guest:!:100:100::/home/guest:
nobody:!:4294967294:4294967294::/:
lpd:!:9:4294967294::/:
imnadm:*:200:200::/home/imnadm:/usr/bin/ksh
stewart:!:201:1:Stewart Watkiss:/home/stewart:/usr/bin/ksh
```

Then the password is stored encrypted in the /etc/security/passwd file that can only be accessed by root or members of the security group.

```
stewart:
        password = rY9dYuHPm5IeE
        lastupdate = 953139523
        flags = ADMCHG
```

The password shown above is actually the same as the userid stewart. I have shown this as an example of how a userid is encrypted so that it cannot be easily understood. This is a very insecure password, the password should NOT be the same as the userid. If you try giving another username the same password as I've used it will almost certainly have a different encrypted password. The reason for this is that the lastupdate entry is also used in the encryption algorithm to produce unique passwords.

It is important that the password file is kept secure, because although the passwords are encrypted it is possible to perform a dictionary attack against the encrypted passwords. These work by taking words from a dictionary and encrypting them as passwords and then comparing them against the password file. If there are any matches then the password is provided. These can be more sophisticated by replacing letters with numbers e.g. 1 instead of I etc. and by using different dictionaries the chance of getting the password is better. These programs are freely available two examples being “Crack” or “John the Ripper”, it is a good idea for system administrators to run these periodically against their systems to ensure that people are using secure passwords. Alternatively using the system can prevent dictionary words being used by updating the users profile.

The following SMIT screen accessed by smit chuser shows some of the security restrictions that can be put against a user.

```
                    Change / Show Characteristics of a User

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

[MORE...21]                                          [Entry Fields]
  Days to WARN USER before password expires         [0]                     #
  Password CHECK METHODS                            []
  Password DICTIONARY FILES                         []
  NUMBER OF PASSWORDS before reuse                  [4]                     #
  WEEKS before password reuse                       [0]                     #
  Weeks between password EXPIRATION and LOCKOUT     [-1]
  Password MAX. AGE                                 [4]                     #
  Password MIN. AGE                                 [0]                     #
  Password MIN. LENGTH                              [6]                     #
  Password MIN. ALPHA characters                    [1]                     #
  Password MIN. OTHER characters                    [1]                     #
  Password MAX. REPEATED characters                 [2]                     #
  Password MIN. DIFFERENT characters                [1]                     #
[MORE...16]

F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Days to warn user before password expires - By implementing this a user will have a warning upon login that their password is about to expire. This gives them time to think of a good password rather than being forced to think of one on the spot.
Password check methods - This allows administrator supplied code to check the password in addition to the other methods.
Password dictionary files - Putting dictionary files in this field will check that the password does not match the entry before changing the password.
Number of passwords before reuse - This remembers some earlier passwords to ensure that they are not being reused.
Weeks before password reuse - Prevents passwords being reused until a number of weeks has passed. This prevents uses from changing the password a number of times so that they can reuse and older password.
Weeks before password expiration and lockout - This can lock a user from logging in if they have not changed the password after it expired. This can be used to revoke userid's that have not been used for a long time.
Password max. age - This forces the user to change their password after a set time
Password min. age - This prevents a user from changing their password until a time has expired. It is not recommended as it will prevent the user from changing the password if they know it has been compromised.
Password min. length, alpha characters, other characters - This enforces the user to choose a password with a minimum length a minimum number of letters and a minimum number of numbers or other characters. This prevents passwords consisting of just letters or just numbers.

Password max. repeated characters - This is the number of identical characters that can exist within the password. By setting this to a low figure it prevents passwords like AAAAAA.
Password max. different characters - This ensures a minimum number of characters different to those in the previous password. This stops a password from being the same as the previous password but reordered.

There are also other authentication methods that can be used instead of usernames and passwords, or that can be used in addition of the standard authentication. For example you may want to implement a smartcard scheme or perform retina or fingerprint scans before allowing access. These are defined in the /etc/security/user file and have entries auth1 and auth2. If a new entry is added the details must also be included in /etc/security/login.cfg.

The auth1 check is forced upon the user. Unless the program completes correctly the login will fail. There can be more than one entry, or it could be set to NONE for no authentication. The auth2 check does not have to return any particular values for the user to continue to use the system. This is sometimes used to prompt the user to supply details of the reason for them accessing the machine which is added to a log.

The ROOT User

Every UNIX system must have a ROOT user defined. This user has ultimate authority over the system and has authority to do anything. The user can read, write and execute any command (although he may have to explicitly change the permissions). Because of this the root user must be very securely protected. It is not normal to login directly as root as accidents by root can cause a lot of damage (running rm -r * from the root directory will literally delete every file on the system).

The normal method of using the root userid is to switch user (su) to root from a normal userid. This can be forced by setting the root userid to be unable to login and unable to login remotely (use smit chuser). When su’ing to root you should minimise the time spent as root and where root authority is not needed you should work under your normal userid. Also the root password should be held by only those that really need to have full access (preferably only one person).

Where other users need to perform administrator functions this can sometimes be achieved by adding the user to an appropriate group. For example adding a user to the security group would enable them to be able to reset users passwords without ever knowing the root password. This is preferably to having multiple people with root access.

Network Security

Although bugs are fixed in software fairly quickly after they are identified there are sometimes software running with bugs that can be exploited. Also there are protocols that are designed to be open and “friendly” which by their very nature can be exploited (NFS is a good example of this which is explained more in the Networking Section). The best approach to this is to ensure that the software is not running (or the port not open) to present that opportunity. Therefore a decision needs to be made as to what services are needed and what services can be disabled or removed. Any services that are not needed as part of the machines function can be turned off.

Telnet - This is a primary way of logging onto and maintaining a UNIX machine. It is fairly secure in that there is no way of bypassing the userid / password login set up on the computer. The biggest risk is if the userid and password is compromised. One way of reducing the risk is to use wrappers. The telnetd is started as part of /etc/inetd.conf

FTP - File Transfer Protocol is a popular way of transferring files onto / from a computer. FTP uses telnet as it’s primary login protocol and therefore has similar risks as telnet. There are a couple of particular risks associated with ftp, one is that although you can prevent remote logging in of certain userid’s for telnet this is not always the case for FTP. Another is that FTP will not increment the failed login count if the login fails. This means that using FTP the password can be retried indefinitely without the userid becoming revoked.

NFS and the ‘R’ commands - There are a series of commands associated with NFS and the SUN RPC protocol. The commands include, rlogin, rshell, rwho, rusers, rstatd, rexec, pnfs etc. The way that these work is by allowing users on “trusted hosts” to login directly from another machine without providing a userid or password on the remote system. There are potentially serious security risks associated with these and therefore if the machine needs to be secure (e.g. a firewall) then these should certainly be turned off. The best way of achieving this is to remove the client code and then disable any services that remain in inetd.conf.

The client code can be removed by:

from smit going to “Software Installation & Maintenance”
then “Software Maintenance and Utilities”
then “Remove Installed Software”

or using the fastpath “smit remove”

```
                         Remove Installed Software

Type or select values in entry fields.
Press Enter AFTER making all desired changes.


                                                        [Entry Fields]
* SOFTWARE name                                      []                      +
  PREVIEW only? (remove operation will NOT occur)     yes                    +
  REMOVE dependent software?                          no                     +
  EXTEND file systems if space needed?                no                     +
  DETAILED output?                                    no                     +










F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Press F4 against software name to list all the installed software. Press the forward slash key ('/') and enter nfs to search for nfs. Press F7 against any entries referring to nfs, e.g. bos.net.nfs.client. Press enter.

Back at the remove screen change preview to no and press enter. The nfs client software will then be removed.

Kshell and Klogin - These are the same as the rshell and rlogin commands however they use kerberos authentication. These are therefore more secure than the traditional rshell and rlogin commands. These can be disabled by commenting out the lines in /etc/inetd.conf.

Talk - This is the talk daemon used for interactive text messaging between users. If this is not needed then it can be disabled by commenting out the line in /etc/inetd.conf.

Echo - The echo daemon simply returns whatever is sent to it. This however can be used for a denial of service attack. It can be disabled by commenting out the lines in /etc/inetd.conf.

Discard - This is used for debugging and performance measurement. Any data received on the port is discarded. The port can be targeted for a denial of service attack. It can be disabled by commenting out the lines in /etc/inetd.conf.

Chargen - The Character Generator. This sends data back to the requester. It can be targeted for a denial of service attack. The service can be disabled by commenting out the lines in /etc/inetd.conf.

Daytime - The daytime service returns a date string. This service is relatively harmless however if it is not needed then it may as well be turned off by commenting out the lines in /etc/inetd.conf

Calendar - This is the cmsd line daemon. It is part of the SUN RPC commands and should therefore be disabled unless used. It can be disabled by commenting out cmsd from /etc/inetd.conf.

dtspc - The Desktop Subprocess Control Daemon is used by the Distributed File System. The DFS works in a similar way to NFS in that the normal authentication can be bypassed for trusted hosts. It is therefore recommended that this is disabled by commenting it out in /etc/inetd.conf

ttdbserver - This is part of the SUN RPC commands and should be disabled if these are not used. To disable comment out the line in /etc/inetd.conf.

Sendmail - Also known as SMTP. If the machine does not need to be able to handle mail then the SendMail daemon should be removed. The daemon can be disabled by commenting out the qpi and start sendmail lines from /etc/rc.tcpip.

Portmap - The Portmapper daemon is used by a number of services. In particular it is used by NFS and CDE. This can be removed from /etc/rc.tcpip however be aware that if it is removed you will no longer be able to use CDE directly on the machine. It is still possible to login using the Failsafe Login option or by using the text based login. It is also still possible to export X-Clients to another machine running an X-server. If you need to disable the portmapper but still use a graphics screen you may want to consider using an alternative window manager such as Motif Window Manager (MWM) or After Step.

SNMP - The Simple Network Management Protocol is often used in large networks to monitor the status of hosts and routers across the network. There is a security exposure in that SNMP will broadcast a lot of information about the machine that if intercepted can be used to identify possible exposures and back doors. If SNMP is not used to monitor the computer then this should be turned of by commenting out the entry in /etc/rc.tcpip. If SNMP is used then it is recommended that the community name is not one that can be easily guessed. At the very least it should be changed from it’s default of public. The community name is defined in /etc/snmpd.conf.

XDMCP - XDMCP allows logins via the CDE screen. This provides another way of login into a computer and should therefore be removed if it is not needed.

To remove this risk the /usr/dt/config/Xaccess file should be edited replacing the line

```
*                       # grant service to all remote displays
```
with
```
localhost               # grant service to no remote displays
```

The file is read-only so should be changed to read-write for the owner, edited by root and then changed back to read-only.

Limiting the Access of Services (wrappers) - Wrappers can be used to limit who can access a certain port. Normally they restrict based upon the IP address of the computer trying to connect. The ones normally used are known as tcpwrappers and are freely available. These can either be used to allow only certain addresses. A typical use would be to restrict telnet to the IP addresses of the computers used by the system administrators. If that was not practical then it could instead be set to deny certain address ranges, such as those in none trusted networks.

Applying Software Fixes (PTF's)

When bugs and vulnerabilities are identified in software then updates are created and alerts recommending that they are installed are issued. These PTF's (as IBM call them) should be downloaded and installed onto all vulnerable machines as soon as possible (depending upon the severity of the risk).

The PTF's can be downloaded using either FixDist or by anonymous FTP to one of the IBM anonymous FTP servers (e.g. ftp.software.ibm.com - directory aix/fixes/v4). Once downloaded they should be installed using SMIT.

Using smit go to "Software Installation and Maintenance" then "Install and Update Software" then "Install and Update from LATEST Available Software".

Alternatively enter `smit update_all`

Using the updated all option should prevent any filesets being installed unless they already exist. However after applying PTF's it is sometimes the case that ports that were previously disabled have been re-enabled. After applying any PTF's all the other security checks should be performed to ensure no new exposures have been created.

Using Alerts and Tripwires

If someone is able to penetrate your system the final line of defence is being able to detect their presence allowing you to either tighten up the security or identify what data may have been compromised. Whilst there are a number of tools to aid detection the biggest problem is to ensure that the logs are checked and to then separate the signs of an intruder from the routine messages and false alarms. The filtering of messages really needs to be tailored to the needs of your system. For example it is possible to monitor all executable code to see if it changes. This may be fine on most web servers and the like however if you have a system that is used for application development you may get a lot of false alarms as a result of the normal use of the system.

Trusted Computing Base - The trusted computing base (TCB) is a method of checking programs and files to ensure they are not changed. The only way to install the TCB is to select it when first installing the operating system. The reason for this is that the only time you can guarantee that all the programs are intact and clean is when the system is installed. The TCB also checks file permissions to ensure that no files have gained the SUID bit allowing it to run as root. The program tcbck is used to check that no changes have been made and a Secure Attention Key (SAK) can be pressed to connect to the trusted shell. The SAK key is normally defined as ctrl-x ctrl-r. More information on TCB can be found as part of the AIX documentation in "System Management Guide: Operating System and Devices".

AIX Audit - AIX provides a tool to monitor for possible security breaches. This is called audit and is run b the auditbin daemon. Audit is capable of detecting hundreds of different conditions that it interprets as alerts, however it is down to the system administrator to select which one of these are suitable for a particular system. Of course the auditing is of no use unless it is frequently checked to look for relevant alerts. This could be handled automatically by having a program search the audit log for multiple signs of attack and then alerting root.

Checking the User Environment - There are a number of commands that can check the user and password files to check that there are no syntax errors etc. The commands are userck, grpck and pwdck. For more details you should see the man pages, however here's a quick overview of some of their users.

- `userck -t All` - This checks the userid definitions. Typically it may find an expired expiration date and prompt to have the account disabled. Problems with ROOT should NOT be changed with this command.
- `grpck -t ALL` - This checks that users listed as members of the groups are valid users, that the GID is unique and that the group name is valid. Typically you may be prompted to remove entries for users that don't exist or where there is conflicting data.
- `pwdck -t ALL` - This checks the authentication stanzas in /etc/passwd and /etc/security/password. Typically you may be prompted where a username does not have an asterix in the password field. This does not check that the password rules, such as min. length.

Other Tools - There are other tools that can be obtained to provide further checking. MD5 or Tripwire are examples of fingerprinting tools that can be used to check when code is changed.

Auditing the Systems

To provide good security is not simply a case of setting it up, going away and hoping there are no more problems. Instead the system should be periodically checked to ensure that no holes have been inadvertently reopened, no PTF's have been missed and that there are no signs of break in. There are a number of tools that can be used to simplify this.

Some of these provide scanning of network ports such as NSA or SATAN or others can check the user policies such as AIX Cops. You could also check that users are using secure passwords by using a password cracking program such as "Crack" or "John the Ripper".

You should always be vigilant to security. Along with most of the technological world it's something that doesn't stand still. The sophistication being used by hackers is increasing at an alarming rate and just keeping up is a continual battle.

Remember being auditable is NOT the same as being secure !

More Security Information

There are numerous Web Sites with information on security. Web Sites are by their nature volatile and likely to change. If you find any of the following are out of date then try using a search engine to find what your looking for. Here's a few starting points.

Official Sites

http://www.cert.org	Computer Emergency Response Team
http://www.ibm.com/security/	IBM's Security Pages
http://lists.gnac.net/firewalls/	Firewalls Mailing List
http://www.tis.com/	Trusted Information Systems
http://www.w3.org/Security/Faq/www-security-faq.html	WWW Security FAQ
http://www.cs.purdue.edu/ar95/AR95Book-127.html	Tripwire and Auditing
http://www.fbi.gov/nipc/trinoo.htm	Distributed DoS Information
http://www.socks.nec.com/	Socks Servers Information

Many of the tools can be downloaded from:

ftp://ftp.cert.org/pub/tools

Newsgroups

comp.security.firewalls	Firewalls
comp.security.unix	UNIX specific
comp.security.announce	CERT announcements

comp.security.misc	Other
ibm.ibmunix.security	AIX Security
alt.security	Unofficial
alt.computer.security	Unofficial

Hacker Sites

When someone manages to penetrate a system rather than just keep quite about it then often decide to tell everyone about it. By looking at the hacker sites you can keep yourself up to date with new techniques being used. These are often much more recent than the official bulletins distributed by the official organisations.

http://www.2600.com	2600 Group
http://www.astalavista.com	Underground Search Engine

Newsgroups
alt.2600
alt.hacker

Diagnostics

There are a number of different diagnostics programs as part of AIX.

On classical RS/6000's run in the following modes:

- Maintenance or Single User Mode
- Concurrent Mode
- Stand-alone Mode

There are also programs available on a separate diagnostic CD-ROM which must be ordered separately.

The diag program is shipped as part of the AIX operating system.

The diag program can be used in different modes however has a different level of access in the different modes.

In Maintenance or Single User mode normal system activity is stopped. The SCSI adapters and the disk drive used for paging cannot be checked. The other devices can however be analysed. To run in single user mode:

1. Stop all programs except the operating system

2. Log on as root
3. shutdown -m
4. Wait for message indicating that the system is in maintenance mode
5. diag
6. Diagnostic Operating Instructions will be displayed. Follow the instructions.
7. Function Selection menu - select diagnostic routines
   a. Select System verification to analyse the error log and should be used to verify the machine is functioning after a repair or upgrade
   b. Select Problem determination to test the system and analyse the log if available. This option should be used when a problem is suspected on the machine.
8. When testing is complete press F3 twice to return to the prompt
9. Press <Ctrl-D> to logoff.

In concurrent mode the normal operating system runs while diagnostics are running. However there is a limited testing of components. Including the SCSI adapters, the disk used for paging, display adapters, input devices and network adapters (if active).

Maintenance mode diagnostics perform better as it allows the devices to be tested in isolation.

To run diagnostics in concurrent mode:

1. Log on as root
2. enter diag command

Stand-alone mode can be used where the system can be rebooted. The system is booted from CD or Tape allowing the adapters and SCSI desks to be tested.
To run diagnostics in Stand-alone mode:

Stop all programs including the operating system
Turn off the power
Set the key to service position and load the media
Turn on the power
Follow the instructions on the console display.

Stand-alone mode diagnostics can also be run for the hard disk after booting in service mode.

The following items cannot be tested, the device used to boot from, the adapter that supports the device booted from.

For PCI Systems diag command cannot be used on all the different types. Instead the machine needs to be rebooted into SMS and then choosing the “Test the computer” option.

Each device can then be selected for testing.

System Dumps and Crashes

The system dump facility copies the kernel data structure to a dump device. This can be used by AIX support for analysis of persistent problems.

Taking a dump

A dump can be invoked in a number of ways:

1. Using the keyboard (on a classical RS/6000 with key in service mode)
2. Using the reset button (on a classical RS/6000 with key in service mode)
3. A dump will be taken automatically if the kernel panics.
4. By using the dump command from the command line or using SMIT

Usually the raw dump data is placed on a tape and sent to AIX support for analysis. Or the data can be formatted into a readable format using the crash command.

To force a dump the following methods can be used.

sysdumpstart -p

or

If a lft keyboard is attached then with the key in service mode <CTRL><ALT><NUMPAD1> or <CTRL><ALT><NUMPAD2>

The default set-up can be altered with the sysdumpdev command.

```
sysdumpdev device
```

The following options can also be used

-l	list
-p	primary
-s	secondary
-P	make change permanent
-d directory	Directory to copy the dump to at boot time. If it fails the forced copy flag is set at false
-D directory	Directory to copy the dump to at boot time. If it fails the forced copy flag is set to TRUE, allowing the user to copy the dump

-K	reset button will force a dump with the key in the normal position, or on a machine without a key switch.
-z	writes to standard output the string containing the size of the dump.
-e	Estimate of the space required if a dump was taken at the current time

```
# sysdumpdev -l
primary              /dev/hd6
secondary            /dev/sysdumpnull
copy directory       /var/adm/ras
forced copy flag     TRUE
always allow dump    FALSE
```

When a dump is taken the following information is saved:

- System vars and stats - These are the kernel parameters set by the user or hard coded into the kernel
- process table - List of all currently running processes
- User areas - Information on the current running processes on the system. Includes the file descriptor table.
- VFS information - Information about mounted filesystems, the inode table and the open file table
- Kernel stack - Kernels record of the processes.
- TTY information - About configured tty's their characteristics and current state
- System buffers - Area where incoming data is stored while waiting memory allocation
- Memory buffers - Memory buffers for data which has been sent / received across a network
- Timers - These handle processes such as sleep
- Sockets - A logical device for network traffic

Typically when a dump is taken the amount of space required is approximately ¼ the amount of physical memory. This can be queried using:

```
sysdumpdev -e
```

For Classical RS/6000's there are a number of LED codes associated with Dumps. If a dump is taken as a result of a system panic, the LED will show 0c9 then when the dump is completed will then flash 888. By pressing the reset button the complete code will be shown (0c0 for a successful dump) and then pressing the reset button will cycle through more codes relating to the reason for the dump.

Other codes are:

0c0	Dump Completed successfully
0c2	Dump Started by user

0c3	The dump is inhibited
0c4	Not enough space on dump device. Only partial dump stored
0c5	Dump failed to start. Error occurred trying to write to dump device (tape may not have been loaded)
0c6	Secondary dump started by user
0c7	Network dump in progress. The LED should alternate between 0c7 and 0c2 or 0c9. If it stops then an error may have occurred
0c8	No dump device specified, therefore disabled
0c9	System-initiated panic dump started
c20	Kernel panic with debug program on console

When the system is rebooted after a dump the dump will be copied from paging space to the set directory (normally /var/adm/ras). If there is insufficient space then depending upon when forced copy is turned on the user will be prompted to copy the dump. This is called by rc.boot and is a script called /lib/boot/srvboot. The copydumpmenu command is called by srvboot and is the prompt that appears on the screen. The menu allows the dump to be copied directly onto tape.

Providing Information for AIX Systems Support Centre

If the dump is happening repeatedly or the dump has been taken as part of problem determination for a system problem the dump should be sent to the AIX Systems Support Centre. There is a command called “snap” that can be used to collect information to send to AIX support. The command should be run as follows

```
/usr/sbin/snap -fgkD -o /dev/rmtx
```

or as specified by the support Centre.

This should be stored on a tape labelled with the Problem Management Record (PMR) number, and details of how the tape was created and it’s block size.

The options shown above are:

-f	Include File System Information
-g	Include General Information
-k	Include Kernel Information
-D	Gather dump and /unix

Dump Analysis

Normally dump analysis is carried out by the AIX Systems Support Centre, however the tools used are also available to the system administrator.

The tool used for dump analysis is called crash and details can be found using the man pages. Shown below are a few screen captures demonstrating the use of crash.

```
# crash
WARNING: Using crash on a live system can potentially
         cause a system crash and/or data corruption.
> stat
        sysname: AIX
        nodename: overton
        release: 3
        version: 4
        machine: 005386904C00
        time of crash: Fri 11 Feb 07:51:47 2000
        age of system: 6 day, 20 hr., 1 min.
        xmalloc debug: disabled
> status
CPU     TID  TSLOT     PID  PSLOT  PROC_NAME
  0    b577    181    43d0     67  crash
```

```
> p -r
SLT ST    PID   PPID   PGRP   UID  EUID  TCNT  NAME
  2 a     204      0      0     0     0     1  wait
        FLAGS: swapped_in no_swap fixed_pri kproc
 67 a    43d0   46dc   43d0     0     0     1  crash
        FLAGS: swapped_in execed
 71 a    47bc   3d42   3d42   201   201     1  dtscreen
        FLAGS: swapped_in orphanpgrp execed
>
> proc -71
SLT ST    PID   PPID   PGRP   UID  EUID  TCNT  NAME
  0 a       0      0      0     0     0     1  swapper
        FLAGS: swapped_in no_swap fixed_pri kproc

Links:  *child:0xe3002c90  *siblings:0x00000000  *uidl:0xe3000170
    *ganchor:0x00000000  *pgrpl:0x00000000  *ttyl:0x00000000
Dispatch Fields:  pevent:0x00000000  *synch:0xffffffff
    lock:0x00000000  lock_d:0x00000000
Thread Fields:  *threadlist:0xe6000000  threadcount:1
    active:1  suspended:0  local:0   terminating:0
Scheduler Fields:   fixed pri: 16  repage:0x00000000  scount:0  sched_pri:0
    *sched_next:0x00000000  *sched_back:0x00000000 cpticks:19431
    msgcnt:0    majfltsec:0
Misc:  adspace:0x00000f0f  kstackseg:0x00000000  xstat:0x0000
    *p_ipc:0x00000000  *p_dblist:0x00000000  *p_dbnext:0x00000000
Signal Information:
```

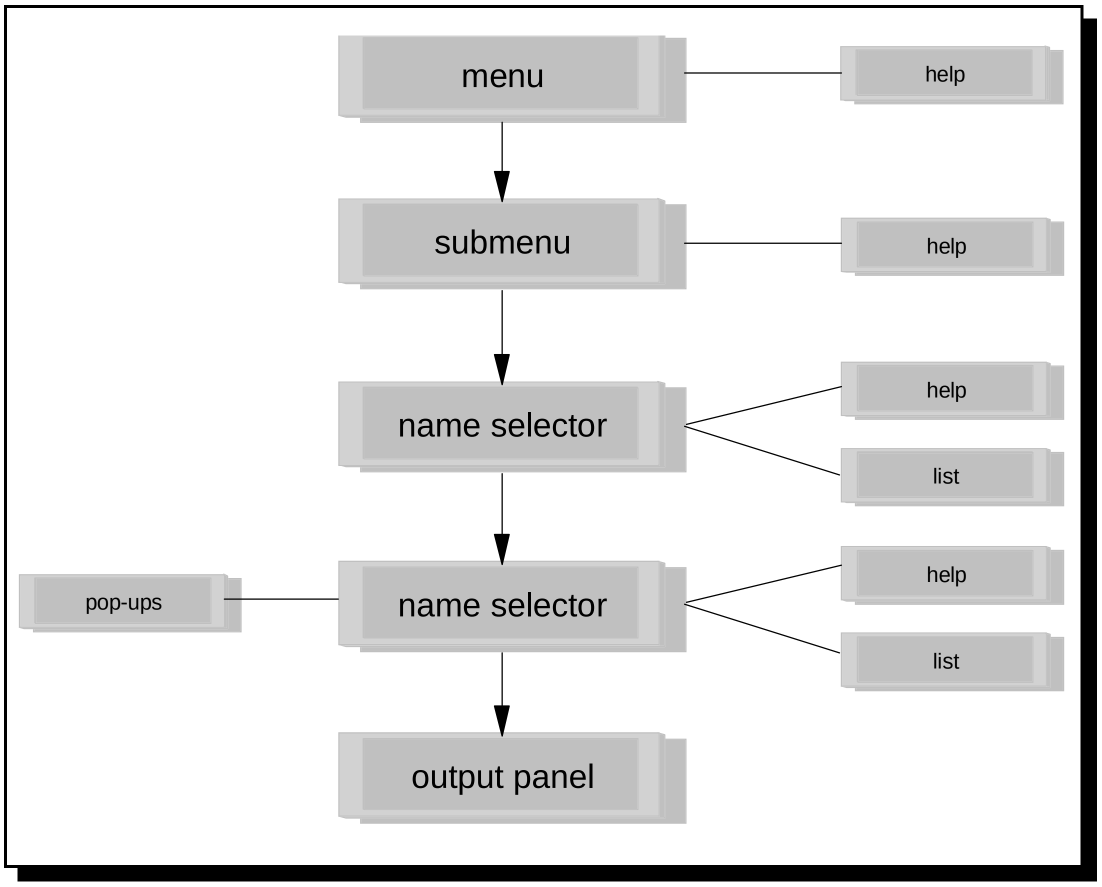
```
> trace 26
STACK TRACE:
        .e_block_thread ()      f0299978
        .[nfs.ext:svc_getreq] ()        f02999d8
        .threadentry () f0299f78
        .thread_terminate ()    f0299fb8
```

Advanced Customisations

SMIT menus

The menu that SMIT uses are all contained within the ODM (Object Data Manager). A search through the ODM is performed every time that a new menu page is viewed or a command run. As the menus are obtained from the ODM it is possible for them to be different on different AIX systems depending upon the installed components and on any customisations carried out.

The ODM is by default stored in /etc/objrepos



User Interface Components of SMIT

The above diagram shows how the menu systems of SMIT link together. The components can be described as:

- **Menus** - There are a hierarchical structure of menus which puts the commands into related areas. Some of the menus may be repeated in different sections where the use spans more than one function.
- **Selector / Dialog Screens** - This allows the user to select which object the action is to be performed on. For example you may select which disk drive to carry out a certain action on.
- **Pop-up Lists** - Where there are a number of possible values for a parameter. These can either select a single object or allow multiple selections.
- **Output Panels -** SMIT runs either standard commands or scripts. The standard output and standard error streams from the commands are captured and displayed in a special SMIT output screen which allows it to be reviewed by the user.
- **Contextual Help** - There is on-line help available which provide help for each submenu, dialog and screen.

Performance Tuning

To really understand performance tuning requires a knowledge of how the RS6000 works under the cover. I am not able to go into such depth here, so instead I will just cover some of the tools that are available and some of the basic principles in performance tuning.

There are three key ways to improve performance of a system. One is to through money at it (faster CPU, more CPU's, more memory, better disk systems etc.). The second is to offload some of the tasks (i.e. move an application to another machine to share the load). Unfortunately both these so far include additional cost, so the third way is to perform tuning on the system to provide the best service with what's available.

When tuning a system there is normally a trade off between throughput and response time. Throughput is sometimes referred to as batch and applies to processes that are not neccessarily time critical and will run in the background. Response Time is related to the real time expectations of a user and often applies to interactive applications. Often when someone says a system is performing badly what they mean is that the interactive applications are taking a long time, reallocating resources from the batch applications this can sometimes be improved. Hopefully the degraded service given to the batch applications is not noticed.

There are a number of tools available for analysing / controlling system performance. Some of these could be used for all three of the different ways of improving performance by
1/ Highlighting Bottlenecks that should be upgraded
2/ Identifying the resource hungry processes
or
3/ Managing the tuning or processes and the timing of applications to reduce conflicts.

Some of these are:

•	nice	- sets priority value when a command is run
•	renice	- changes the priority of a command started with nice
•	ps	- Reports real time activity on processes
•	sar	- Reports on overall system activity
•	vmstat	- Reports Virtual Memory Statistics
•	iostat	- Reports I/O statistics
•	tprof	- Process CPU utilisation information and application profiling*
•	svmon	- Shows how memory is used by processes*
•	filemon	- I/O details in terms of logical volumes and files*
•	PDT	- Reports on the general health of the system*
•	perfpmr	- Collects information for use by the Support Centre*
•	WLM	- Work Load Manager provides a means of allocating resources*

* AIX specific commands. Some of which maybe in the Performance Toolkit which is a separate licensed product.

Setting Goals

The first stage of any performance tuning is to set the goals and ensure it is possible to verify any improvements.

The goals could be contractual (e.g. Service Level Agreements) or just to make the best use of all available resources. However someone needs to decide the priorites of the different tasks and what is an acceptable level of performance.

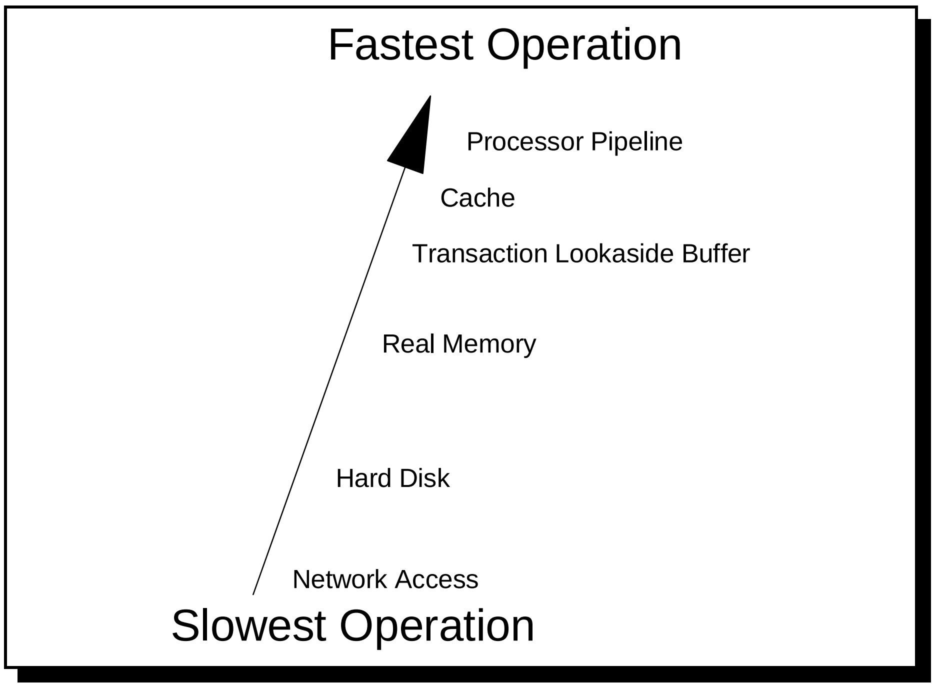
Then the current state of the system should be benchmarked. When tuning a system it is often a case of try something and see the effect, therefore benchmarks are needed to ensure any tuning improves the required performance rather than degrades it.

There is not standard approach or “cookbook” to performance tuning, however there are a number of rules of thumb that can help in tuning the system. We are often up against unanticipated usage in many environments. Imagine that you have a website selling a certain product, the product is given a really good review by a number of magazines and suddenly your website is flooded by people wanting to get more information. It is not always possible to predict these events so some leeway should be introduced into these systems. However on a payroll system, where there is a constant number of employees the load on the system is known and unlikely to change rapidly. On these systems then you do not need as many reserve resources.

The extent at which you can influence the performance is also something that varies. Typically I am looking at this from a System Administrator point of view, so some techniques such as rewriting of the code cannot always be applied unless you actually own that piece of code.

Basic System Theory

The following diagram shows some of the different hardware components in relation to how significant their impact on performance.



The Effect of Hardware on System Performance

As an application runs it has to work it’s way up the layers. The exception being the network layer that may or may not be utilised by an application. Working up the layers the hardware is more scarce and more expensive however is faster. Looking at each of the components

Network Access If data has to be accessed over a network then this will certainly have the greatest impact on performance. Although this will be different depending upon the media used (e.g. dialup modem at 56kbps compared with ethernet at 100Mbps) it will still be considerably slower than accessing the data directly off the hard drive. By controlling what information is stored over the network and what is held locally then performance can sometimes be improved.

Hard Disk By far the slowest access within the system is accessing data from a Hard Disk or CD-ROM drive. This is where applications are stored before they are loaded into memory.

Real Memory When a program is set to run it is loaded into memory. Once this process is done then the program is able to run without constantly accessing the disk. However when data is requested that is not in memory it still needs to be loaded from disk. Applications that have been loaded into memory are waiting threads or interrupt handlers.

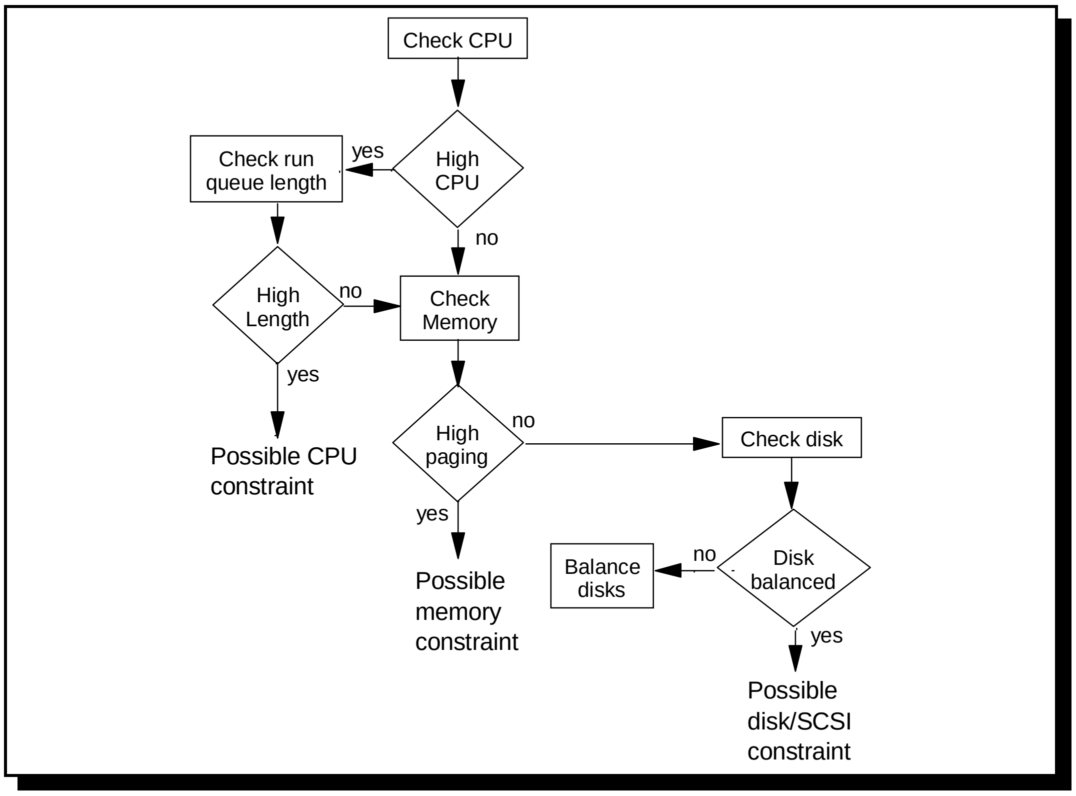
Transaction Lookaside Buffer (TLB) The TLB keeps the addresses of recently accessed pages of memory to minimise address translation. When the address is stored in the buffer then the application is a dispatchable thread.

Cache There can be more than one cache on a system often referred to as level 1 and 2 where 1 is on the die of the CPU and 2 is stored on the motherboard. The Current dispatched thread would be located in cache.

Processor Pipeline This is the current instruction running inside the CPU.

Performance Analysis

The following flow chart indicates a few basic checks to identify the area where a performance problem may lie.



Flow Chart for Performance Analysis

Each of the different end results could have multiple solutions.

CPU constraint - Rather than looking at adding another processor or upgrading the processor, it is useful to see what processes are hogging the CPU. This could be a process that is stuck in a loop or something that might be better running on a different system.

Paging - This may be a runaway program that is using too much memory, otherwise installing more memory should help.

Disk constraint - If there are more than 3 or 4 disks on a single SCSI adapter then you may want to add another adapter to share the load. Otherwise more disks allowing the data to be spread out between them.

Dynamic Kernel

One of the fundamental things about AIX is that it has a dynamic kernel that can be tuned whilst the system is running. Any changes will take effect immediately without needing a restart.

The lsattr command can be used to list the kernel attributes.

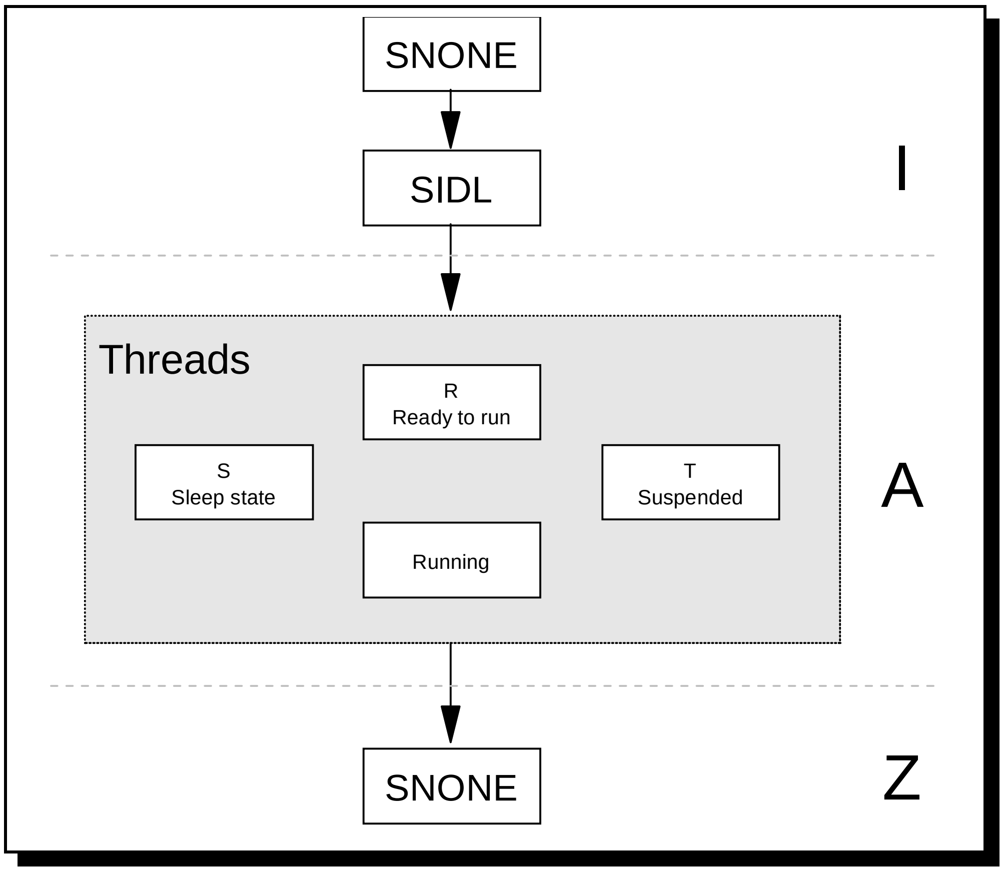
```
$ lsattr -E -l sys0
keylock     normal                 State of system keylock at boot time
maxbuf      20                     Maximum number of pages in block I/O BUFFER C
maxmbuf     0                      Maximum Kbytes of real memory allowed for  MBU
maxuproc    500                    Maximum number of PROCESSES allowed per user
autorestart false                  Automatically REBOOT system after a crash
iostat      true                   Continuously maintain DISK I/O history
realmem     196608                 Amount of usable physical memory in Kbytes
conslogin   enable                 System Console Login
fwversion   0.1                    Firmware version and revision levels
maxpout     0                      HIGH water mark for pending write  I/Os per fi
minpout     0                      LOW water mark for pending write  I/Os per fil
fullcore    false                  Enable full CORE dump
pre430core  false                  IBM PowerPC  CHRP Computer
modelname   IBM PPS Model 7248 (E) Machine name
systemid    IBM72485538690         Hardware system identifier
```

The chdev command is used to change the values.

`chdev -l sys0 -a` *`attrib=value`*

Processes and Threads

A process goes through a number of states during it’s lifetime. This is shown below:



Process Flow Diagram

Initialisation Period (I)

This is the period where the process is setup and starts to run. This is normally a very brief period.

SNONE State - Before a process is created it requests a slot in the process table.

SIDL State - After being allocated a position in the process table, it then needs to wait for resources to be allocated to it before it can run.

Running State (A)

Once a process has been initialised the threads of the process move between 4 different states.

R - Ready to Run State. Here a thread is waiting to be given a time slot for processing by the CPU.

S - Sleep state. The thread is waiting for an event from I/O. One the I/O is completed the thread will return to the Ready to Run State.

T - Suspended state. This state is reached if a user (or process) sends a SIGSTOP signal to the thread. This suspends the running until a SIGCONT signal is sent.

Zombie State (Z)

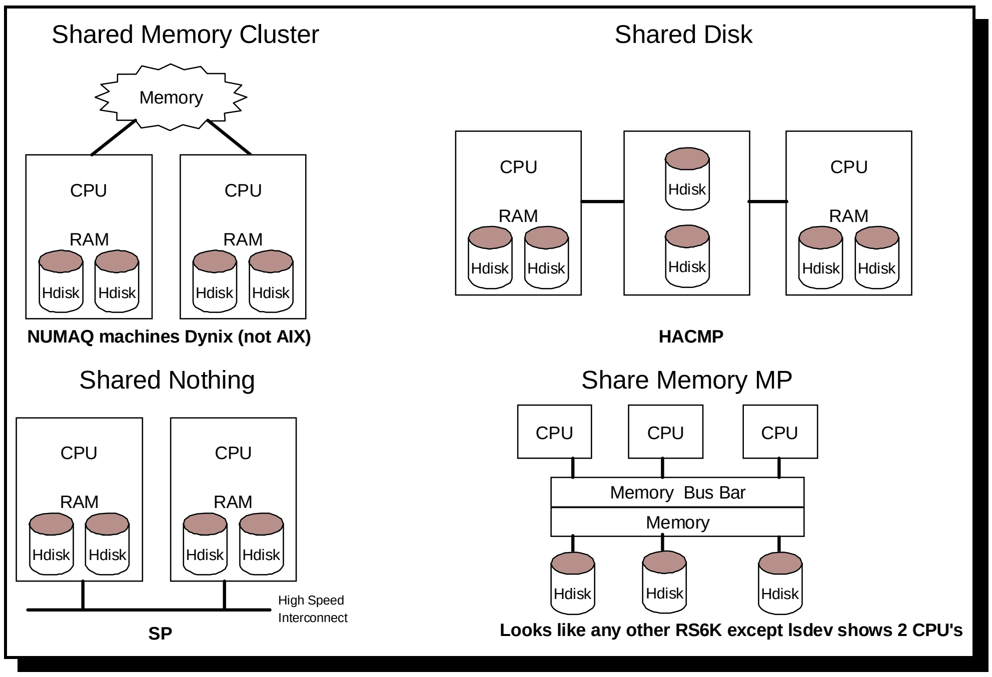
When a process exits it becomes a zombie. This occupies a slot in the process table, however all other resources are released. The Parent process should then after a short while be terminated by it’s parent process. If this does not happen (a problem with the parent process or badly coded parent process) then it is possible that the process will always remain in a zombie state. This is not a performance issue on it’s own unless a lot of zombie processes fill up the process table.

The only way to remove a process that is in zombie state is to reboot.

Multithreading and Multiprocessors

Prior to AIX V4 one process represented one thread. However AIX V4 allows for multiple processers to be used and therefore processes can be split into separate threads to run simultaneously on different processors. How well a process can be split into threads depends upon how it has been written. There are certain steps that can be taken when writing a process that can reap benefits when run on a machine with multiple processors. An unoptomised program could even take longer (due to the overhead of splitting the process into threads) although this is generally not the case.

There are different ways that multiple processors can be built into a machine depending upon what components are duplicated and what are shared.



Different Multiprocessor Architectures

In future all RS/6000’s will fit into one of these architectures. Most uniprocessor machines will be replaced with a Shared Memory MP, with one CPU by default and the option to add at least one more.

The multiprocessing can be either asymmetric or symmetric. Under asymmetric there is a master slave relationship with one processor acting as a master which is the only one able to access I/O. This is source of a potential bottleneck. With symmetric all processors are functionally equivalent. Conflicts to access storage are dealt with by the hardware and conflicts to system-wide tables are resolved by software.

Even in symmetric systems a processor still needs to be assigned as a master. The only effect of this is that only that processor can run device drivers designed for a uniprocessor machine. This hides the other processors from the device driver software.

Viewing Processes (ps)

The command to view processes is the ps command. There are a lot of flags that can be used however here are a few common displays.

Show Processes Owned by User (this session only)

```
$ ps
   PID    TTY  TIME CMD
 17052  pts/3  0:00 /usr/bin/ksh
 17388  pts/3  0:00 dtpad
 19654  pts/3  0:00 ps
```

Show all user processes (-e) and give the priority information (-l).

```
$ ps -el
       F S      UID   PID  PPID   C PRI NI ADDR    SZ    WCHAN    TTY  TIME CMD
  200003 A        0     1     0   0  60 20  a0a   628              -  0:14 init
  240001 A      201  2156 13276   0  60 20 14f2   756          pts/1  0:00 ksh
   40001 A        0  2352     1   0  60 20  465   656              -  0:00
dtlogin
  240001 A        0  2646  4648   0  60 20 1311   928              -  0:03
portmap
  240001 A        0  2894  2352   3  61 20 1071  8032 50001094      - 155:39 X
  240001 A      201  3292 12504   0  60 20  1aff   796              -  0:00 bsh
  240001 A      201  3500 14904   1  68 24  1ebb 23440              - 24:29
netscape_aix4
  240401 A      201  3952 13954   0  60 20  1b7e  2188              -  0:26 dtwm
  240001 A      0  4154     1   0  60 20  ccd   348 5055b698      -  1:37  syncd
   40401 A      0  4390     1   0  60 20 18d9   708     b288c      -  0:00
errdemon
  240001 A        0  4648     1   0  60 20 16d7   816              -  0:00
srcmstr
  240001 A        0  4934     1   0  60 20  604   508  50665c2c      -  0:05 cron
  240001 A        0  5190  4648   0  60 20  1e1c   528              -  0:00  inetd
  240001 A        0  5420  4648   0  60 20  7e6   988              -  0:00
sendmail
  240001 A        0  5700  4648   0  60 20 1012   480              -  0:01
syslogd
  240401 A        0  6022     1   0  60 20  6a4   296    1be080      -  0:00
uprintfd
   40401 A      201  6396 12580   0  60 20  422  1776  5074c82c      -  0:00
dtfile
```

Show all processes, including kernel processes.

```
ps aux
USER       PID %CPU %MEM   SZ  RSS    TTY STAT    STIME  TIME COMMAND
root       516 91.3  0.0  264    8      - A      04  Feb 9448:20 kproc
root      2894  1.5  5.0 8032 2936      - A      04  Feb 155:41 /usr/lpp/X11/bin/
watkiss   3500  0.6  9.0 23440 5568      - A      08  Feb 24:30 /usr/netscape/com
watkiss  16014  0.5  1.0  908  752      - A    09:51:24  0:07  /usr/dt/bin/dtter
watkiss  13276  0.2  1.0 1212  524      - A    07:19:34  0:22  /usr/dt/bin/dtter
root      1032  0.2  0.0  320   56      - A      04  Feb 18:55 kproc
nobody    7488  0.1  1.0 3376  408      - A      04  Feb  5:16 /usr/sbin/httpd
watkiss  16740  0.0  2.0 2748 1108      - A    07:21:05  0:04  dtwm
root         0  0.0  0.0   12   12      - A      04  Feb  3:18 swapper
watkiss  14752  0.0  0.0  960  292      - A    09:19:41  0:01  /usr/dt/bin/dtter
root      4154  0.0  0.0  348   36      - A      04  Feb  1:37 /usr/sbin/syncd 6
watkiss  13954  0.0  2.0 1712 1360      - A      07  Feb  0:52 /usr/dt/bin/dtses
watkiss  12504  0.0  1.0 1176  484      - A      07  Feb  0:42 /usr/dt/bin/ttses
watkiss   3952  0.0  2.0 2188 1048      - A      07  Feb  0:26 dtwm
root      6710  0.0  0.0 1268  204      - A      04  Feb  0:17 /usr/sbin/snmpd
root         1  0.0  0.0  628  164      - A      04  Feb  0:14 /etc/init
root       774  0.0  0.0  272   16      - A      04  Feb  0:11 kproc
```

Show the BND column (-o THREAD) showing individual thread details (-m)

```
$ ps -mo THREAD
    USER   PID  PPID    TID ST  CP PRI SC    WCHAN        F     TT BND COMMAND
 watkiss 17052 16014      - A    0  60  1        -   240001  pts/3   0
/usr/bin/ksh
       -     -     -  42717 S    0  60  1        -      400      -   0 -
 watkiss 17388 17052      - A    0  80  1        -   200001  pts/3   0 dtpad
       -     -     -  48021 S    0  80  1        -   408410      -   0 -
 watkiss 19162 17052      - A   10  65  1        -   200001  pts/3   0 ps  -mo
THREAD
       -     -     -  49581 R   10  65  1        -        0      -   0 -
```

Process Priority

A process’s priority value determines it’s run-queue slot. Those with a higher priority (lower value) run more and those with the lowest priority (highest value) get less access to the CPU.

The priority has a value between 0 and 127.

There are two categories of priorities, fixed and non-fixed.

The fixed priority processes, include some kernel processes, real time application processes (using setpri) and processes started without a nice value.

Variable priority processes are started at an initial priority level that can subsequently change. I/O intensive processes are favoured over CPU-intensive processes as they go into a sleep state more frequently freeing the CPU to handle other processes.

Setting a Process Priority (nice)

A process can be started with a certain priority using the nice command. The nice command allows an increment from 1 to 19 to be applied to a process. The higher the value the lower the priority.

By default a foreground process is assigned a priority of 20 whereas for a background process it is assigned a process of 24.

To decrease the priority of a job the command is used

```
nice -10 command
```

To increase the priority of a job (can only be done by root) the command is:

```
nice --10 command
```

The nice value can be viewed using ps -l command see later for an example.

Changing the nice Value (renice)

If a process was started with the nice command then it’s priority can be altered using a renice command. The maximum nice value that can be set is 40.

```
renice -n 10 pid
```

The above command increases the nice value for a process.
Only root can decrease a nice value even if it has already been increased by a user.

An example is shown below.

```
$ nice dtpad &
[1]     17388
$
$ ps -l
       F S      UID   PID  PPID   C PRI NI ADDR    SZ    WCHAN    TTY  TIME CMD
  240001 A      201 17052 16014   0  60 20 1277   488           pts/3  0:00 ksh
  200001 A      201 17388 17052   0  88 34 1255   680           pts/3  0:00 dtpad
  200001 A      201 19634 17052   9  64 20   87   288           pts/3  0:00 ps
$
$ renice -n 10 17388
$
$ ps -l
       F S      UID   PID  PPID   C PRI NI ADDR    SZ    WCHAN    TTY  TIME CMD
  240001 A      201 17052 16014   1  60 20 1277   488           pts/3  0:00 ksh
  200001 A      201 17388 17052   0 100 40 1255   680           pts/3  0:00 dtpad
  200001 A      201 19640 17052  10  65 20 1077   288           pts/3  0:00 ps
```

You can see the nice value in the header NI. The program is a background process so has a default priority of 24. Using nice by default will add 10 to this making 34.

When the renice command is run 10 is added giving 44, however the maximum nice value is 40 which is what the program is set to.

In the following screenshot root decreases the nice value by 10 taking it back to 30.

```
# ps -l
       F S      UID   PID  PPID   C PRI NI ADDR    SZ    WCHAN    TTY  TIME CMD
  240001 A      201 17052 16014   0  60 20 1277   488           pts/3  0:00 ksh
  200001 A      201 17388 17052   0 100 40 1255   680           pts/3  0:00 dtpad
  200001 A        0 18896 19650   4  62 20 1394   288           pts/3  0:00 ps
  200001 A        0 19650 17052   1  60 20 1077   488           pts/3  0:00 ksh
#
# renice -n -10 17388
#
# ps -l
       F S      UID   PID  PPID   C PRI NI ADDR    SZ    WCHAN    TTY  TIME CMD
  240001 A      201 17052 16014   0  60 20 1277   488           pts/3  0:00 ksh
  200001 A      201 17388 17052   0  80 30 1255   680           pts/3  0:00 dtpad
  200001 A        0 18900 19650  10  65 20 1394   288           pts/3  0:00 ps
  200001 A        0 19650 17052   1  60 20 1077   488           pts/3  0:00 ksh
```

CPU Penalties

If a processor hungry process is given a high priority value then it is possible that it could hog the CPU and prevent any other processes from running. So to overcome this there is a further value called the CPU penalty. The CPU penalty is increased for every time a process is in the CPU when a timer interrupt occurs (10 ms). This is then used to lower the priority of the CPU intensive application using the formula:

```
priority value = base priority + nice value + CPU penalty
```

Base priority = 40
nice value defaults to 20

```
CPU Penalty = CPU usage x R
```

(R = 0.5 by default).

Then once every second the CPU usage is decreased

CPU usage = CPU usage x D

(D = 0.5 by default)

By manipulating these two numbers (R and D) the effect on CPU intensive processes can be altered. To alter these the schedtune command is used. However instead of using the numbers directly these are expressed as r and d where:

R = r / 32

D = d / 32

So for the default values of 0.5 the values of r and d are 16.

```
schedtune -r 16 -d 16
```

The value is lost at reboot so needs to be included in /etc/inittab for a permanent change.

Managing Jobs

There are a number of commands that can be used to control the running of commands.

Only two of the commands listed below can actually be used to control jobs that have already been started. They are the renice and kill commands.

- nice / renice - Allows the priority of a job to be changed
- bsh queue - Allows shell scripts to be queued for batch processing
- batch, at, crontab - Automates the running of commands or sets commands to run later
- ulimit, /etc/security/limits - The limits file can control processes or the users processes. The ulimit command can be used to query them.
- kill - terminates a process.

The /etc/security/limits file can restrict the processes using the following limits:

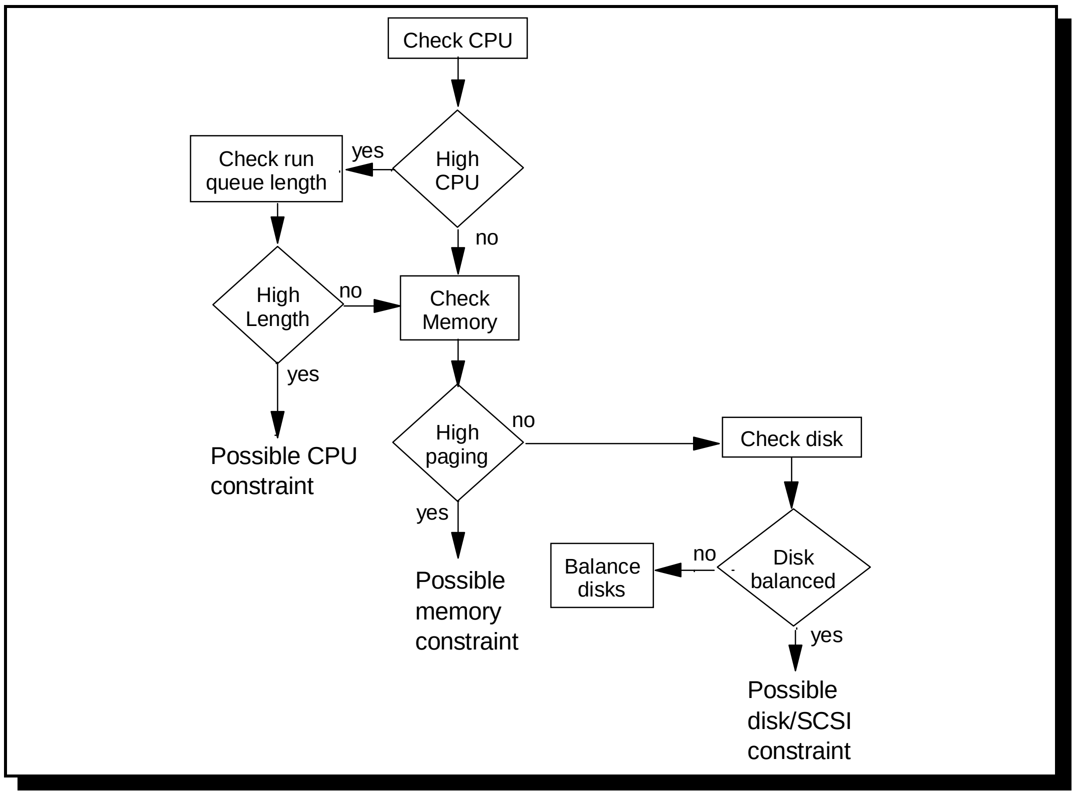
fsize	Largest file a user can create. The default is 2,097,151 blocks. The smallest possible value is 8192.
core	Largest core file allowed in 512 bytes
cpu	Maximum number of CPU seconds a process is allowed before being killed. Normally this is disabled by setting to -1.
data	The largest data segment allowed in units of 512 bytes
stack	Maximum stack size a process is allowed in 512 bytes
rss	Maximum real memory a process can acquire in 512 bytes

The ulimit reports user process resource limits based on the values in /etc/security/limits. The following flags can be used

-a	List all current resource limits
-c	List / specifies the size of core dumps
-d	List / specifies the size of the data area
-f	Lists / sets the file size limit in blocks
-H	Specifies the hard limit
-m	Lists / specifies the size of physical memory in K bytes
-s	Lists / specifies the stack size in K bytes
-S	Specifies the soft limit
-t	Lists / specifies the number of CPU seconds to be used by each process

If neither H or S are specified then the change will be made to both.

CPU Utilisation



CPU Utilisation (sar)

The CPU utilisation can be displayed using the sar command.

```
# sar -u 60 3

AIX myrs6k 3 4 005386904C00    02/11/00

11:52:28    %usr    %sys    %wio   %idle
11:53:28       3       3       3      90
11:54:28       4       3       0      93
11:55:28       4       4       1      92

Average        3       3       1      92
```

The -u options specifies display utilisation. The first value is how long an interval to monitor for and then the 2nd number is for the number of samples to take. The following information is given:

%usr = percentage of CPU time devoted to user processes
%sys = percentage of CPU time devoted to kernel processes
%wio = percentage of CPU time waiting for disk I/O to complete

%idle = percentage of CPU time idle

There are a series of system activity counters that record various activities and provide the data that sar reports. They are run automatically regardless of whether sar is run or not.

sar can be run by members of the system group.

If the idle time is high then it is unlikely to be a problem with the processor speed.

The sar command can also be used to check the run queue for the system. This is done using the -q flag.

```
# sar -q 60 3

AIX tsthost1 3 4 005386904C00    02/11/00

12:20:44 runq-sz %runocc swpq-sz %swpocc
12:21:44     1.2      15
12:22:44     1.4      15     1.0       2
12:23:44     1.3      12

Average      1.3      14     1.0       1
```

The details are:

runq-sz	Average length of the run queue.
%runocc	Percentage of time run queue occupied
swpq-sz	Average number of kernel threads waiting to be paged in. This is not just a measure of paging space activity as it could be held within a file system
%swpocc	Percentage of time the swap queue is occupied.

The data is more useful if collected over an extended length of time. Also this is dependant upon the type of jobs that are waiting to be run. For example if there are a lot of short processes that run briefly before finishing then this is not as much a concern if they are large processes that will run for hours at a time.

Memory Information (vmstat and svmon)

To view the paging information then run the vmstat command.

```
# vmstat 2 10
kthr     memory             page              faults        cpu
----- ----------- ------------------------ ------------ -----------
 r  b   avm   fre  re  pi  po  fr   sr  cy  in   sy  cs us sy id wa
 0  0 29420  3419   0   0   0   0    1   0 207 1134 282  3  2 94  1
 0  0 29420  3416   0   0   0   0    0   0 237  850 256  1  4 73 22
 0  0 29420  3416   0   0   0   0    0   0 235  866 245  4  3 88  6
 0  0 29420  3416   0   0   0   0    0   0 218  877 241  2  5 93  0
 0  0 29420  3416   0   0   0   0    0   0 221  869 243  2  1 97  0
 0  0 29420  3416   0   0   0   0    0   0 217  821 239  2  1 97  0
 0  0 29420  3416   0   0   0   0    0   0 221  784 238  0  3 97  0
 0  0 29420  3416   0   0   0   0    0   0 222  833 240  4  0 95  0
 0  0 29420  3416   0   0   0   0    0   0 219  836 243  2  0 97  0
 0  0 29420  3416   0   0   0   0    0   0 235  815 243  2  4 94  0
```

The time intervals have the same meaning as before, however as the paging space access changes frequently it is a good idea to make the intervals shorter.

If there is insufficient RAM you will see a lot of page-stealing and paging space activity.

The columns are for

kthr (like sar -q)

(r)	Number of the kernel threads placed on the run queue for CPU attention
(b)	Number of kernel threads placed on the wait queue (waiting for I/O)

Memory (totals at instant)

avm	Number of active virtual 4k pages
fre	Size of the free list - the number of 4k frames of data that are free (this may be small as a lot of memory is used for file system cache)

Page (per second)

re	Page reclaims (always 0 for V4)
pi / po	Page ins / Page outs.
fr / sr	Pages freed / scanned by the page stealer
cy	Cock cycles used by page replacement algorithm (normally 0)

faults (per second)

in	Device interrupts
sy	Number of system calls

kernel thread context switches

cpu (like sar -u)

us	User
sy	System
id	Idle time
wa	Waiting for I/O

Any user may run vmstat. If the cy values are large then this means that the page stealer is not freeing enough memory and hence the amount of memory is over committed.

If pi and po are high then this is demand paging. The pages are having to be reread from disk and the wait on I/O will increase. This also indicates that there is insufficient free memory.

To see a summary of memory usage since system startup then the -s option can be used.

```
$ vmstat -s
    2809542 total address trans. faults
     127866 page ins
     312532 page outs
        178 paging space page ins
        695 paging space page outs
          0 total reclaims
    1335153 zero filled pages faults
      11977 executable filled pages faults
     344517 pages examined by clock
          7 revolutions of the clock hand
      83982 pages freed by the clock
      20307 backtracks
          0 lock misses
          0 free frame waits
          0 extend XPT waits
      47172 pending I/O waits
     313333 start I/Os
     313333 iodones
  200027156 cpu context switches
  349229902 device interrupts
          0 software interrupts
          0 traps
  343494647 syscalls
```

The ps command can also be used to show memory usage and to indicate which processors are using the most memory. The aux switch is used to who this:

```
$ ps aux
USER       PID %CPU %MEM   SZ  RSS    TTY STAT    STIME  TIME COMMAND
root       516 97.5  7.0    8 13980      - A      03  Nov 38133:38 kproc
watkiss   4070  0.9 11.0 21200 21540      - A    10:38:40  3:33  /usr/local/netsc
a
watkiss  14840  0.4  1.0 2108 2420      - A      27  Nov 19:45 /usr/local/bin/xf
root      1032  0.3  7.0   64 14036      - A      03  Nov 122:06 kproc
root      5676  0.2  0.0  624  520      - A      03  Nov 82:11 /usr/sbin/dpid2
root     17306  0.1  3.0 5924 5112      - A      21  Nov 16:49 /usr/lpp/X11/bin/
watkiss  17596  0.1  1.0 1456 1532      - A      27  Nov  4:25 xfwm
root         0  0.0  7.0   12 13984      - A      03  Nov  8:01 swapper
root     12014  0.0  0.0 1892  388      - A      14  Nov  2:24 /usr/local/bin/ht
root      8776  0.0  7.0   16 13988      - A      03  Nov  3:11 kproc
root         1  0.0  0.0  668  704      - A      03  Nov  2:28 /etc/init
root       774  0.0  7.0   16 13988      - A      03  Nov  0:47 kproc
watkiss   3808  0.0  1.0  896 1064      - A      27  Nov  0:05 xterm
root      6718  0.0  0.0  472  484      - A      03  Nov  0:32 /usr/bin/AIXPower
watkiss  15686  0.0  1.0  892 1064      - A      28  Nov  0:02 xterm
root      5182  0.0  0.0  316  368      - A      03  Nov  0:23 /usr/sbin/cron
watkiss  12502  0.0  1.0  896 1072      - A      27  Nov  0:02 xterm
root      3162  0.0  0.0  320  324      - A      03  Nov  0:05 /usr/sbin/syslogd
root      4392  0.0  0.0  480  488      - A      03  Nov  0:04 /usr/lib/errdemon
root      3390  0.0  0.0  704  544      - A      03  Nov  0:03 /usr/sbin/sshd -f
root      5942  0.0  0.0  232  288      - A      03  Nov  0:02 qdaemon
```

The %MEM column does however tend to exaggerate the amount of memory being used where it is shared with other processes.

The svmon command is used to view a snapshot of the current state of memory.

I/O Information

I/O information is measured using the iostat command. This works differently to the other commands in that the first interval is taken from the last system reboot, all subsequent intervals use the current time.

```
# iostat 60 2

tty:      tin         tout   avg-cpu:  % user    % sys     % idle    % iowait
          0.0          0.9              3.1      2.4       93.6       0.8

Disks:        % tm_act     Kbps      tps    Kb_read   Kb_wrtn
hdisk0           0.8      15.0       0.4    4035475   5193483
hdisk1           0.1       0.3       0.0      58583    101268
cd0              0.0       0.0       0.0       6730         0

tty:      tin         tout   avg-cpu:  % user    % sys     % idle    % iowait
          0.0         51.7              4.8     15.5       46.6      33.1

Disks:        % tm_act     Kbps      tps    Kb_read   Kb_wrtn
hdisk0          34.3     196.2      49.2       9631      2140
hdisk1           4.4       8.3       1.7        340       156
cd0              0.0       0.0       0.0          0         0
```

The following outputs are given

tty Characters read from (tin) and sent out (tout) to terminals.

cpu Gives the same as sar -u

Disk This gives the I/O statistics for each disk and CD-ROM on the system.
%tm_act - percentage of time the device was active over the period
kbps - the number of k bytes per second transferred
tps - the number of transfers per second
kb_read and kb_write are the amount of data read and written during the interval. This shows the load balancing of the disks.

Anyone can run the iostat command. If the amount of time the processor is busy is greater than 80% (i.e. user + sys) then the system is CPU bound.

AIX Performance Toolbox /6000

The following commands are all part of the performance toolbox that is a separate licensable product with some versions of AIX.

Identifying CPU Intensive Processes (tprof)

The tprof command monitors the cpu usage of the different processes. It will highlight those that use the most cpu time.

Reporting Memory Used (svmon)

The svmon provides information on the memory being used.
Running the command with the -P option will show the processes using the most memory.

Locating I/O Hot Spots (filemon)

The filemon command identifies the most active logical volumes on the system. If these are on highly utilised disks they can be moved onto more suitable disks or positioned in the centre of a disk (see inter-physical / intra-physical allocation policy).

Performance Diagnostic Tool (PDT)

The PDT is a tool that collects information on the system and provides a reporting facility to identify the bottlenecks.

The command to run is

```
/usr/sbin/perf/diag_tool/pdt_config
```

this can only be run by root.

```
________________PDT customization menu__________________

1) show current  PDT report recipient and severity level
2) modify/enable PDT reporting
3) disable       PDT reporting
4) modify/enable PDT collection
5) disable       PDT collection
6) de-install    PDT
7) exit pdt_config
Please enter a number:
```

Enabling collection will setup a number of cron jobs to run and collect the information.

This uses a collection of programs in /usr/sbin/perf/diag_tool that collect and record the data.

The time for the data to be retained is held in /var/perf/cfg/diag_tool/.retention.lsit

The reporting component periodically produces a diagnostics report from the current historical data. This is mailed to adm and written to /var/tmp/PDT_REPORT. The previous days report is held in /var/tmp/PDT_REPORT.last

Errors are written to /var/perf/tmp/.stderr

The thresholds are held in /var/perf/cfg/diag_tool/.thresholds

```
DISK_STORAGE_BALANCE 800
PAGING_SPACE_BALANCE 4
NUMBER_OF_BALANCE 1
MIN_UTIL 3
FS_UTIL_LIMIT 90
MEMORY_FACTOR .9
TREND_THRESHOLD .01
EVENT_HORIZON 30
```

Specific monitors can be configured in /var/perf/cfg/diag_tool/.files
or for network performance in /var/perf/cfg/diag_tool/.nodes

Performance Problem Management (perfpmr)

The perfpmr script (/usr/sbin/perf/pmr/perfpmr) can be used where a performance problem is believed to be attributable to a software problem in the operating system. This should be run if requested by the AIX System Support Centre.

Once it has been collected it can be put on tape using either of the following processes.

Tar Method:

```
tar -cvf /tmp/tarbin /var/perf/tmp
compress /tmp/tarbin
tar -cvf /dev/rmt0 /tmp/tarbin.Z
```

Backup Method:

```
find /var/perf/tmp -print | backup -ipqvf /dev/rmt0
```

Appendix A - Dealing with files from other operating systems

Situations may arise where you want to transfer data between computers running different operating systems.

Transferring files to / from DOS disks

AIX has the built-in ability to read and write to DOS disks.

The built in commands are:

```
dosdir -l
```
List the contents of a DOS disk

```
dosread file1.doc file1
```
Copy file1.doc from a DOS disk to file2 in the current UNIX directory

If the file is a text file then the DOS and UNIX formats differ slightly. The DOS format has CRLF (Carriage Return - Line Feed) where UNIX uses a NL (newline) to denote the end of a line. Also DOS uses (CTRL-Z) whereas UNIX uses EOF(end-of-file) to mark the end of a file. To have the file converted between the formats whilst copying use the following command.

```
dosread -a file1.doc file1
```

```
doswrite file1 file1.doc
```
Copy file1 from the current UNIX directory to a DOS disk.

```
dosformat
```
Format a disk in DOS format

all the commands assume /dev/fd0 for the DOS disk, which can be changed.

Another way of accessing DOS format disks is to use the Mtools commands. These are not provided as standard with AIX however can be freely obtained. These have some of the normal DOS commands prefixed with an ‘m’. For example

```
mdir a:
```
will list the contents of the device /dev/fd0

```
mcopy a:file1.doc file1
```

will copy the file file1.doc on /dev/fd0 to file1 in the UNIX current directory. The -t option will perform the same text conversions as the -a option on dosread.

Appendix B - Undocumented Commands

```
lqueryvg [i-g VGid] -p PVname [i-NsFncDaLPAvt]
```

The PP size represents 2 to the power. For the display above this is 2^2=4MB

The following options can be used

-p	Show VG data (must be used with other options)
-A	Gives output for all the fields
-t	Displays the title for each field
-N	Shows the Max LV's
-s	Shows the PP size
-F	Shows the free PP's
-n	Shows the LV count
-c	Shows the PV count
-D	Shows the total VGDA's
-L	Shows the LV data
-P	Shows the PV data
-v	Shows the VGID
-a	No output

```
getlvcb [-AT] -[aceilLmnrsptufxy] lvname
```

The following options can be used:

-a	Shows the intra-policy (e, ei, c, mi, m - for edge ... middle)
-c	The number of copies
-e	The inter-policy (m - minimum, x - maximum)
-i	The lvid
-l	The logical volume name
-L	The label of the logical volume name
-m	The machine id (CPU)
-n	Number of logical partitions
-r	If the relocate able field is set to yes
-s	Stripe size (0 - no striping, 12 - 4k, 14 - 6k, 15 - 32k, 16 - 64k, 17 - 128k)
-p	Stripe width (0 - no striping otherwise the number of PV's)
-t	Type of the logical volume
-u	Upper bound
-f	Lines extracted from /etc/filesystems
-x	The creation time
-y	The last modified time
-A	Gives output for all the fields
-T	Displays the title for each of the fields.

Appendix C RS/6000 LED codes

BIST LED codes

100	BIST completed successfully
101	BIST is running
102	Starting BIST after power on reset
103	Model number not determined
104	Could not find common on-chip processor bus address
105	Not able to read the OCS EPROM
106	Module Failure
111	OCS Stopped; module error
112	A checkstop occurred bu the logout cannot start
113	Checkstop count is equal 3
120 - 127	CRC checks on EPROM and NVRAM
130	Start of presence test
140,142,144	Procedure error, BIST unsuccessful
151 - 154	AIPGM, DCLST, ACLST, AST tests
160	Missing Early Power-Off Warning (EPOW) connector
161	The Bump quick I/O tests failed
162	The JTAG tests failed
164	Error while reading low NVRAM
165	Error while writing low NVRAM
166	Error while reading high NVRAM
167	Error while writing high NVRAM.
168	Error while reading the serial input / output (SIO) register
169	Error while writing the serial input / output (SIO) register
180	Progress indicator - logout in progress
182	COP bus is not responding
185	A checkstop condition occurred
186	System logic-generated checkstop (250 models only)
187	Unable to identify chip release level
195	Logout completed

POST LED Codes

200	Keylock in the Secure position
201	Checkstop occurred (fatal)
202 - 210	Unexpected interrupt
20C	Error detected in L2 cache
211	IPLROM CRC miscompare
212	RAM POST found processor bad
213	RAM POST failure, memory can not be configured

214	I/O planar failure
215	A low voltage condition present (fatal)
216	IPL code being uncompressed
217	End of boot device list reached
218	RAM POST is looking for 1M good memory
219	RAM POST bit map generation
21C	L2 cache not detected as part of systems configuration
220	IPL control block initialisation
221	NVRAM CRC miscompare
222 - 238	IPL from devices specified in NVRAM or ROM (normal mode)
22C - 23C	Attempting a normal-mode IPL from FDDI
239	System failed to IPL
240 - 258	IPL from devices specified in NVRAM or ROM (service mode)
24C, 25C	Attempting a service-mode IPL from FDDI
260	Menus are being displayed on the local display or terminal
261	No support for display adapter found
262	No keyboard found
263	Normal mode system restart from device specified in NVRAM
269	Stalled state. Cannot boot system
270	Ethernet/FDX 10 mbps MC adapter test is running
271 - 287	Mouse, Keyboard, ports and adapters POST
288	Adapter card slots queried
289	Gt0 POWER graphics adapter POST is running
290	I/O planar test started
291 - 293	Std. I/O, SCSI, DBA disk POSTs
294	TCW SIMM in slot J is bad
295	Color Graphics Display test is running
296	Family 2 Feature ROM test is running
297	Model number could not be determined
298	Attempting a warm system start
299	IPL ROM passed control to the program code.
2E6	A PCI Ultra/Wide differential SCSI adapter is being configured
2E7	An Undetermined PCI SCSI adapter is being configured

Appendix D AIX Toolbox

IBM has indicated that the future direction of AIX will involve closer ties with the Linux Operating System. The big changes will be seen in version 5 and above where the aims are to introduce linux compatibility into the operating system. Initially this will be at a compiler level (if code is written for Linux it should compile under AIX without any changes), however may include a binary level in future (Applications compiled for Linux on a Power PC will run under AIX on a Power PC).

An early stage in the process is the creation of an AIX Toolbox for Linux. This provides an RPM installer, a number of libraries and some opensource applications.

I would expect to see this mature as AIX becomes more compatible with Linux. Maybe even to the point that installp is replaced by the RPM installer, however this is only speculation at this stage.

Care should be taken when using both SMIT installable images and RPM files side by side in that dependancies required for one may already have been installed using the other method. For new systems it may be prudent to follow the RPM route for all opensource software if possible.

The programs and information can be found at:
http://www-1.ibm.com/servers/aix/products/aixos/linux/download.html

Appendix E vi Keys

Navigating

h	cursor left
j	cursor down
k	cursor up
l	cursor right
/*pattern*	search forwards
?*pattern*	search backwards
0	start of line
$	end of line

Inserting Text

i	insert before cursor
I	insert at start of line
a	add after cursor
A	add at end of line
o	open line below
O	open line above
R	start replacing text

Deleting Text

x	delete current character
X	delete previous character
dw	delete word
D	delete to end of line
dd	delete one line
5dd	delete 5 lines

Changing Text

r	replace one character
s	substitute for one character
cw	change word
C	change to end of line
cc	change one line
5cc	change 5 lines

Corrections

u	undo last command
U	restore whole line
ESC	Escape to command mode

Exit from vi

:wq	end edit - save file
:q!	end edit - discard changes

Glossary

|. See pipe
&. See background process
>. See redirection
<. See redirection

A

access permission. The access allowed to a particular file. This can be based on the different levels of access (read, write, execute) at different classes of users (owner, group, all others).

ACL. Access Control List. Used to control access to a file (see access permission).

ACLST. Alternating Current Logic Self-Test

address Space. The range of addresses availaable for a processes code and data.

AIPGM. Array Initialisation Program

AIX. Advanced Interactive Executive. IBM's implementation of the UNIX operating system.

alias. The process of assigning a new name to a command

ANSI. American National Standards Institute

APAR. Authorised Program Analysis Report. This is a fix provided by the support Centre to eliminate a certain bug or problem.

application. Program used to perform a certain task.

ASCII. American Standard Code for Information Interchange. Collection of character sets used throughout the computer industry.

AST. Array Self-Test

awk. Interpreted programming language. Useful for it's pattern matching abilities and is often used in combinations with shell scripts.

B

background process. A process running independently of the initiating terminal. Started by ending the command with an &. The starting process is then no longer waiting for the "death of the child process".

backup. Having a copy of data or code used incase the original information is destroyed.

BOS - Base Operating System

BSD. Berkely Software Distribution. A UNIX thread with differences from the base UNIX platform, although now mostly combined with other platforms.

block device. A device that transfers data in fixed block sizes. Normally 512 or 1024 bytes.

booting. Starting the computer from a power off or a system reset.

byte. Storage used to represent a character. Equal to 8 bits of data.

C

C. Programming language in which the UNIX operating system and most applications are coded in.

CDE. Common Desktop Environment. X Window Manager developed jointly between Sun, IBM and HP to provide a common feel to UNIX running on different machines.

CGI. Common Gateway Interface. A method of providing client server interactivity between a web browser and a server. CGI is implemented by scripts on the server and requires nothing special on the client. This is known as “server side scripting”

character I/O. The transfer of data byte by byte as used by slower devices, such as terminals and printers.

child. A process emerging from a fork.

CISC. Complex Instruction Set Cycles. Processors with a large number of instructions. These tend to be slower due to the number of instructions that the processor has to be able to handle

client. User of a network service.

command. A request to perform an operation or run a program. When arguments are added to the command the entire string is considered to be the command.

console. Terminal used by the kernel to send messages to.

CRC. Cyclic Redundancy Check

current directory. The currently active directory. If no directory specified then the directory that a command will act upon.

CUT. Co-ordinated Universal Time (the same as GMT)

D

DASD. Direct Access Storage Device. IBM’s name for hard disk storage.

DCLST. Direct Current Logic Self-Test

device driver. Program that controls a hardware device, such as printer, disk or screen.

directory. Special file containing the names and controlling information for files and directories.

disk / diskette. Storage medium used for transferring data between machines.

DoS. Denial of Service. Where a computer is attacked so that it can no longer perform it’s role.

E

EBCDIC. Alternative character set to ASCII. Used on IBM mainframes.

editor. Program used to enter and modify text or other files.

environment. Collection of variables passed to a program or shell script by the invoking process.

EPOW. Early Power-Off Warning

escape. The \ character used to indicate that the next character in a command is normal text with no special meaning.

Ethernet. Networking protocol used to connect a LAN.

execute permission. Permission to run a program or list the contents of a directory.

F

field seperator. Character used to separate one field from the next. Normally space or tab.

FIFO. First In, First Out. Queuing process where first into a queue is the first out.

file. Collection of related data stored and received by it’s given name.

file system. Collection of files and structures on a physical or logical media.

flag. Option sent to a program.

FRU. Field Replaceable Unit. Term used by IBM for a part that can be fitted onto an IBM computer. This could be for an upgrade or replacement.

full path name. Directory name given starting from the root (/) directory.

G

gateway. Device acting as a connector between two separate networks.

GMT. Greenwich Mean Time (the same as CUT). The time at Greenwich England, but not taking into account any changes due to British Summer Time.

group. Collection of AIX users show share a set of files.

H

hardware. The physical equipment.

heterogeneous. Applies to networks consisting of products from multiple vendors.

homogeneous. Applies to networks consisting of products from a single vendor.

I

interpreter. Program which “interprets” program statements directly from a text file.

IP. Internet Protocol

IPL. Initial Program Load - the booting sequence of a computer.

K

kernel. The core of the operating system that provides the fundamental running of the system.

kill. To prematurely terminate a process.

L

LAN. Local Area Network. Network normally within a single location.

line editor. Editor that processes one line at a time.

link. Alias one directory or file to another.

Linux. Open Source UNIX like operating system.

login. To provide your identity to the system to gain access.

logout. To inform the system that you no longer need access for this session.

M

make. Programming tool that helps make a program from the source code.

memory. Storage medium, normally refers to volatile memory held in the system.

metacharacters. Characters or combinations of characters to represent different characters or sequence of characters.

Motif. Graphical user interface for X Windows.

N

NFS. Network File System

null device. A logical device used to obtain empty files or dispose of unwanted data.

O

OCS. On-Card Sequencer

ODM. Object Data Manager. Stores some of the core configuration details for AIX

OEM. Original Equipment Manufacturer.

operating system. Software interfacing between the applications and the hardware. It controls how applications can run on the system

owner. The person who created a file, or to whom ownership has been transfered to.

P

parallel processing. Having more than one processor in the same system.

password. Secret character string used to verify user identification.

PATH. Variable which specifies where to search for program files.

path name. Filename specifying directories leading to that file.

permission. Authority given for a set file.

pipes. System routines that can take the output from one process and feed it in as the input to another. The bar '|' character is used to indicate to the system that you wish to use a pipe.

POSIX. Portable Operating System for Computer Environments. Set of open standards for an operating system.

process. Unit of activity known to the system (usually a program).

profile. File in the users home directory executed at login to customise the environment. The actual file is called .profile

R

read permission. Allows reading of a file

redirection. The use of a different device or file instead of STDIN and STDOUT which use the symbols < and > respectively.

regular expression. An expression that specifies a set of character strings using metacharacters.

relative path name. The name of a directory or file using the directories from the current directory.

RETAIN. Problem reporting tool used by IBM, some customers and business partners can have access and can perform searches on problems and APARs

RISC. Reduced Instruction Set Computer. Processer with a small number of individual instructions. By only being able to have a small number of instructions the processer can handle them quickly and generally faster.

root directory. The topmost directory that contains all other directories in the file system.

ROS. Read Only Storage. Firmware included in PCI RS/6000's.

RPM. Redhat Package Manager. Used to bundle applications so that they can be easily installed under Linux. Also available for AIX.

S

scalability. The ability for a computer to accomodate growth with the minimal of effort.

SCCS. Source Code Control System

server. A provider of service in a computer network.

setuid. A permission that allows a program to run as though started by a different user.

shell. User Interface of a UNIX operating system.

shell program / shell script. Program consisting of shell commands in a text file.

signal. Software generated interrupt to another process. As used by the kill command.

SIO. Serial I/O Register

SMIT. System Management Interface Tool. Provides a consistent way of configuring all manner of settings in AIX.

sockets. When a network session is connected a socket is used to represent the address and ports of the systems.

software. The programs run on a system (the part of a system that is not physical).

STDERR. Standard Error. The data stream where errors are normally sent. This is normally the console although it can be redirected.

STDIN. Standard Input. The data stream where input comes from. Normally this is the keyboard although it can be redirected.

STDOUT. Standard Output. The data stream standard messages are outputted to. This is normally the terminal although it can be redirected.

subdirectory. A directory that is subordinate to another directory.

superuser. The system administrator with priviliages allowing them to access every file in the system. This is normally the root user.

swap space. A space on disk where memory can be swapped into to make space for other programs.

system. The computer and it's associated devices and programs.

System V. The thread of UNIX that retained the AT&T style rather than the BSD style.

T

TCB. Trusted Computing Base

TCP. Transmission Control Protocol

TCP/IP. Transmission Control Protocol over Internet Protocol.

termcap. File containing the capabilities and functions of a terminal.

U

UNIX. Multi-user Multitasking operating system. Originally developed at Bell Laboratories in the early 1970's.

V

vi. Visual Editor. A text editor used within a text terminal. Very powerful, but can be difficult to learn initially. Available on just about every UNIX like operating system.

W

wild card. Metacharacter used to specify one or more replacement characters. e.g. * allows any number of charcters to match.

window. Are of the screen in which the running program is displayed.

working directory. Directory in which the current program is running and upon where any actions (not specifiying a directory) will be taken.

write permission. Permission to change the contents of a file or directory.

X

X-Windows. Interface to the system that provides windows. Also useful in distributing applications as the application can run on a different machine to the one where the screen and keyboard are being used.

Command Summary

The following are a list of useful commands.

at *time job*
Runs a command at a specific time.

backup *filesystem*
Backup the filesytem

backup -i
Backup by filename

cat *file1*
Display the contents of the text file “file1”.

cat *file1 file2* > *file3*
Combine the files file1 and file2 and output into file3

cd
Change to the users home directory

cd ..
Move up a directory

cd *directory*
Change to the specified directory

chgrp *group file*
Change group ownership for a file.

chmod [ugo][+/-][rwx] *file*
Change permissions of a file using symbolic form

chmod *XXX file*
Change permissions of a file using numeric form

chown *owner:group file*
Change the owner / group of a file or directory

chown -R *owner:group directory*
Change the ownership of subdirectories and files.

compress ***filename***

Compresses a file so that it takes up less space. Useful prior to transferring the file to another system using a network or tape. Files compressed with the compress program are suffixed with .Z

cp ***file1 file2***

Copy file1 to file2

cp -R ***dir1 dir2***

Copy a directory and subdirectories from dir1 to dir2

date

Shows and sets the system date and time.

del ***file1***

Deletes a file after asking for confirmation. Ignores file protection allowing the owner to delete a file it owns.

df

Displays available space on all file systems

diff ***file1 file2***

Compares two different text files and indicates the differences

du

Shows a summary of filesystem usage

e ***file***

Edits a file using the INed editor

ed ***file***

Uses the ed editor to edit the file.

env

Display environment variables

find ***path*** **-name** ***filename***

Finds files named filename starting from directory path.

ftp ***hostname***

Interactive file transfer program for transferring files over the network.

grep ***pattern file***

Searches a file for the pattern

gzip ***file***

Compress a file using the gzip program. This is not included with AIX as standard but is often installed by system administrators. Files that have been compressed as suffixed with a .gz

gunzip *file*
Uncompress a file that has been compressed with the gzip program. This is not included with AIX as standard but is often installed by system administrators. Files to be uncompressed will normally end with .gz

head *-count file*
Display count number of lines from a file

help
One page display of help for new users

iostat *interval count*
Shows CPU and io usage. Displays information obtained during interval and repeats this count times.

kill pid
Terminates a process

ln *file1 file2*
Links file1 to file2

ln -s *file1 file2*
Creates a softlink instead of a hard link

ls *file*
Lists a file. If the file is a directory lists files in the directory.

mail
Read and send mail

man *command*
View manual pages for a command.

mkdir *directory*
Creates a new directory.

mount -t *type* /dev/*device* /mnt/*mountpoint*
Mounts the filesystem of type from /dev/device to /mnt/mounpoint

mount -t cdrfs /dev/cd0 /mnt/cdrom
Mount the CD-ROM drive so that the files can be read as part of the normal file structure.

mv *file1 file2*

Moves or renames a file or directory.

passwd
Changes the password

pg *file1*
View a text file one page at a time

ps -ef
Show all processes

pwd
Shows the current working directory.

qcan -x *jobnumber*
Cancels a print job.

qchk
Checks the status of a print queue

qprt *file*
Print a file

restore
Restores files / filesystem from a backup made with the backup command.

rm *file1*
Deletes (unlinks) a file

rm -r *file1*
Removes a directory (including all files and subdirectories)

rmdir *directory*
Removes a directory and it's contents

sar -u
Display system utilisation

sed *file*
Edits a file using the stream editor.

shutdown *time*
Shutdown the computer at the specified time interval

smit
System Management Interface Tool

stty

Sets terminal settings

stty sane
Resets terminal to default settings

tail -n *count file*
Shows count number of lines from the bottom of file
tail -f *file*
Shows bottom of file, showing new lines as they are added.

tar -c *file1 file2*
Archives file1 an file2 to the default backup device (normally a tape drive)

tar -cvf *filename.tar file1 file2*
Archive file1 and file2 into a file called filename.tar. The -v option will list all files archived

tar -x
Extract the files from the default backup device.

tar -xvf *filename.tar*
Extract all files from an archive. The -v option shows all the files as they are extracted.

telnet *hostname*
Logs into a remote system

tn *hostname*
Lots into a remote system

touch *file*
Updates the access time for a file. If the file does not exist then it creates an empty file.

umount *directory*
Unmounts the directory

umount -f *device*
Unmounts using the device name. The -f option forces the umount if the filesystem is in use.

uname
Shows the name and version of the operating system.

ucompress *filename*
Uncompresses a file compressed using the compress file. Files will normally be suffixed with .Z prior to being uncompressed.

vi *file*
Edits a file using the vi editor

vmstat *interval count*

Show memory usage statistics, measured over interval repeated count times.

who

Displays users on a system

who am i

Displays your username

Index