UNIX Administration Course

Version 1.0

mapleson@gamers.org
Tel: (+44) (0)1772 893297
Fax: (+44) (0)1772 892913
WWW: http://www.futuretech.vuurwerk.nl/

Detailed Notes for Day 3 (Part 3)

UNIX Fundamentals: Typical system administration tasks.

Even though the core features of a UNIX OS are handled automatically, there are still some jobs for an admin to do. Some examples are given here, but not all will be relevant for a particular network or system configuration.

Data Backup.

A fundamental aspect of managing any computer system, UNIX or otherwise, is the backup of user and system data for possible retrieval purposes in the case of system failure, data corruption, etc. Users depend the admin to recover files that have been accidentally erased, or lost due to hardware problems.

Backup Media.

Backup devices may be locally connected to a system, or remotely accessible across a network. Typical backup media types include:

- 1/4" cartridge tape, 8mm cartridge tape (used infrequently today)
- DAT (very common)
- DLT (where lots of data must be archived)
- Floptical, ZIP, JAZ, SyQuest (common for user-level backups)

Backup tapes, disks and other media should be well looked after in a secure location [3].

Backup Tools.

Software tools for archiving data include low-level format-independent tools such as dd, file and directory oriented tools such as tar and cpio, filesystem-oriented tools such as bru, standard UNIX utilities such as dump and restore (cannot be used with XFS filesystems - use xfsdump and xfsrestore instead), etc., and high-level tools (normally commercial packages) such as IRIS NetWorker. Some tools include a GUI frontend interface.

The most commonly used program is tar, which is also widely used for the distribution of shareware and freeware software. Tar allows one to gather together a number of files and directories into a single 'tar archive' file which by convention should always have a '.tar' suffix. By specifying a device such as a DAT instead of an archive file, tar can thus be used to archive data directly to a backup medium.

Tar files can also be compressed, usually with the .gz format (gzip and gunzip) though there are other compression utilities (compress, pack, etc.) Backup and restoration speed can be improved by compressing files before any archiving process commences. Some backup devices have built-in hardware compression abilities. Note that files such as MPEG movies and JPEG images are already in a compressed format, so compressing these prior to backup is pointless.

Straightforward networks and systems will almost always use a DAT drive as the backup device and tar as the software tool. Typically, the 'cron' job scheduling system is used to execute a backup at regular intervals, usually overnight. Cron is discussed in more detail below.

Backup Strategy.

Every UNIX guide will recommend the adoption of a 'backup strategy', ie. a combination of hardware and software related management methods determined to be the most suitable for the site in question.

A backup strategy should be rigidly adhered to once in place. Strict adherence allows an admin to reliably assess whether lost or damaged data is recoverable when a problem arises.

Exactly how an admin performs backups depends upon the specifics of the site in question. Regardless of the chosen strategy, at least two full sets of reasonably current backups should always be maintained. Users should also be encouraged to make their own backups, especially with respect to files which are changed and updated often.

What/When to Backup.

How often a backup is made depends on the system's frequency of use. For a system like the Ve24 SGI network, a complete backup of user data every night, plus a backup of the server's system disk once a week, is fairly typical. However, if a staff member decided to begin important research with commercial implications on the system, I might decide that an additional backup at noon each day should also be performed, or even hourly backups of just that person's account.

Usually, a backup archives all user or system data, but this may not be appropriate for some sites. For example, an artist or animator may only care about their actual project files in their ~/Maya project directory (Maya is a professional Animation/Rendering package) rather than the files which define their user environment, etc. Thus, an admin might decide to only backup every users' Maya projects directory. This would, for example, have the useful side effect of excluding data such as the many files present in a user's .netscape/cache directory. In general though, all of a user's account is archived.

If a change is to be made to a system, especially a server change, then separate backups should be performed before and after the change, just in case anything goes wrong.

Since root file systems do not change very much, they can be backed up less frequently, eg. once per week. An exception might be if the admin wishes to keep a reliable record of system access logs which are part of the root file system, eg. those located in the files (for example):

```
/var/adm/SYSLOG
/var/netscape/suitespot/proxy-sysname-proxy/logs
```

The latter of the two would be relevant if a system had a Proxy server installed, ie. 'sysname' would be the host name of the system. Backing up /usr and /var instead of the entire / root directory is another option - the contents of /usr and /var change more often than many other areas of the overall file system, eg. users' mail is stored in /var/mail and most executable programs are under /usr.

In some cases, it isn't necessary to backup an entire root filesystem anyway. For example, the Indys in Ve24 all have more or less identical installations: all Indys with a 549MB disk have the same disk contents as each other, likewise for those with 2GB disks. The only exception is Wolfen which uses IRIX 6.5 in order to provide proper support for an attached ZIP drive. Thus, a backup of one of the client Indys need only concern specific key files such as /etc/hosts, /etc/sys_id, /var/flexlm/license.dat, etc. However, this policy may not work too well for servers (or even clients) because:

- an apparently small change, eg. adding a new user, installing a software patch, can affect many files,
- the use of GUI-based backup tools does not aid an admin in remembering which files have been archived.

For this reason, most admins will use tar, or a higher-level tool like xfsdump.

Note that because restoring data from a DAT device is slower than copying data directly from disk to disk (especially modern UltraSCSI disks), an easier way to restore a client's system disk - where all clients have identical disk contents - is to clone the disk from another client and then alter the relevant files; this is what I do if a problem occurs.

Other backup devices can be much faster though [1], eg. DLT9000 tape streamer, or military/industrial grade devices such as the DCRsi 240 Digital Cartridge Recording System (30MB/sec) as was used to backup data during the development of the 777 aircraft, or the Ampex DIS 820i Automated Cartridge Library (scalable from 25GB to 6.4TB max capacity, 80MB/sec sustained record rate, 800MB/sec search/read rate, 30 seconds maximum search time for any file), or just a simple RAID backup which some sites may choose to use.

It's unusual to use another disk as a backup medium, but not unheard of. Theoretically, it's the fastest possible backup medium, so if there's a spare disk available, why not? Some sites may even have a 'mirror' system whereby a backup server B copies exactly the changes made to an identical file system on the main server A; in the event of serious failure, server B can take over immediately. SGI's commercial product for this is called IRIS FailSafe, with a switchover time between A and B of less than a millisecond. Fail-safe server configurations like this are the ultimate form of backup, ie. all files are being backed up in real-time, and the support hardware has a backup too. Any safety-critical installation will probably use such methods.

Special power supplies might be important too, eg. a UPS (Uninterruptable Power Supply) which gives some additional power for a few minutes to an hour or more after a power failure and notifies

the system to facilitate a safe shutdown, or a dedicated backup power generator could be used, eg. hospitals, police/fire/ambulance, airtraffic control, etc.

Note: systems managed by more than one admin should be backed up more often; admin policies should be consistent.

Incremental Backup.

This method involves only backing up files which have changed since the previous backup, based on a particular schedule. An incremental schema offers the same degree of 'protection' as an entire system backup and is faster since fewer files are archived each time, which means faster restoration time too (fewer files to search through on a tape).

An example schedule is given in the online book, "IRIX Admin: Backup, Security, and Accounting':

```
"An incremental scheme for a particular filesystem
 looks something like this:

   1. On the first day, back up the entire filesystem.
      This is a monthly backup.

   2. On the second through seventh days, back up only
      the files that changed from the previous day.
      These are daily backups.

   3. On the eighth day, back up all the files that
      changed the previous week. This is a weekly backup.

   4. Repeat steps 2 and 3 for four weeks (about one month).

   5. After four weeks (about a month), start over,
      repeating steps 1 through 4.

You can recycle daily tapes every month, or whenever you feel safe
about doing so. You can keep the weekly tapes for a few months.
You should keep the monthly tapes for about one year before
recycling them."
```

Backup Using a Network Device.

It is possible to archive data to a remote backup medium by specifying the remote host name along with the device name. For example, an ordinary backup to a locally attached DAT might look like this:

```
tar cvf /dev/tape /home/pub
```

Or if no other relevant device was present:

```
tar cv /home/pub
```

For a remote device, simply add the remote host name before the file/directory path:

```
tar cvf yoda:/dev/tape /home/pub
```

Note that if the tar command is trying to access a backup device which is not made by the source vendor, then '/dev/tape' may not work. In such cases, an admin would have to use a suitable lower-level device file, ie. one of the files in /dev/rmt - exactly which one can be determined by deciding on the required functionality of the device, as explained in the relevant device manual, along with the SCSI controller ID and SCSI device ID.

Sometimes a particular user account name may have to be supplied when accessing a remote device, eg.:

```
tar guest@yoda:/dev/tape /home/pub
```

This example wouldn't actually work on the Ve24 network since all guest accounts are locked out for security reasons, except on Wolfen. However, an equivalent use of the above syntax can be demonstrated using Wolfen's ZIP drive and the rcp (remote copy) command:

```
rcp -r /home/pub guest.guest1@wolfen:/zip
```

Though note that the above use of rcp would not retain file time/date creation/modification information when copying the files to the ZIP disk (tar retains all information).

Automatic Backup With Cron.

The job scheduling system called cron can be used to automatically perform backups, eg. overnight. However, such a method should not be relied upon - nothing is better than someone manually executing/observing a backup, ensuring that the procedure worked properly, and correctly labelling the tape afterwards.

If cron is used, a typical entry in the root cron jobs schedule file (/var/spool/cron/crontabs/root) might look like this:

```
0 3 * * * /sbin/tar cf /dev/tape /home
```

This would execute a backup to a locally attached backup device at 3am every morning. Of course, the admin would have to ensure a suitable media was loaded before leaving at the end of each day.

This is a case where the '&&' operator can be useful: in order to ensure no subsequent operation could alter the backed-up data, the 'eject' command could be employed thus:

```
0 3 * * * /sbin/tar cf /dev/tape /home && eject /dev/tape
```

Only after the tar command has finished will the backup media be ejected. Notice there is no 'v' option in these tar commands (verbose mode). Why bother? Nobody will be around to see the output. However, an admin could modify the command to record the output for later reading:

```
0 3 * * * /sbin/tar cvf /dev/tape /home > /var/tmp/tarlog && eject /dev/tape
```

Caring for Backup Media.

This is important, especially when an admin is responsible for backing up commercially valuable, sensitive or confidential data.

Any admin will be familiar with the usual common-sense aspects of caring for any storage medium,

eg. keeping media away from strong magnetic fields, extremes of temperature and humidity, etc., but there are many other factors too. The "IRIX Admin: Backup, Security, and Accounting' guide contains a good summary of all relevant issues:

"**Storage of Backups**

Store your backup tapes carefully. Even if you create backups on more durable media, such as optical disks, take care not to abuse them. Set the write protect switch on tapes you plan to store as soon as a tape is written, but remember to unset it when you are ready to overwrite a previously-used tape.

Do not subject backups to extremes of temperature and humidity, and keep tapes away from strong electromagnetic fields. If there are a large number of workstations at your site, you may wish to devote a special room to storing backups.

Store magnetic tapes, including 1/4 in. and 8 mm cartridges, upright. Do not store tapes on their sides, as this can deform the tape material and cause the tapes to read incorrectly.

Make sure the media is clearly labeled and, if applicable, write-protected. Choose a label-color scheme to identify such aspects of the backup as what system it is from, what level of backup (complete versus partial), what filesystem, and so forth.

To minimize the impact of a disaster at your site, such as a fire, you may want to store main copies of backups in a different building from the actual workstations. You have to balance this practice, though, with the need to have backups handy for recovering files.

If backups contain sensitive data, take the appropriate security precautions, such as placing them in a locked, secure room. Anyone can read a backup tape on a system that has the appropriate utilities.

How Long to Keep Backups

You can keep backups as long as you think you need to. In practice, few sites keep system backup tapes longer than about a year before recycling the tape for new backups. Usually, data for specific purposes and projects is backed up at specific project milestones (for example, when a project is started or finished).

As site administrator, you should consult with your users to determine how long to keep filesystem backups.

With magnetic tapes, however, there are certain physical limitations. Tape gradually loses its flux (magnetism) over time. After about two years, tape can start to lose data.

For long-term storage, re-copy magnetic tapes every year to year-and-a-half to prevent data loss through deterioration. When possible, use checksum programs, such as the sum(1) utility, to make sure data hasn't deteriorated or altered in the copying process. If you want to reliably store data for several years, consider using optical disk.

Guidelines for Tape Reuse

You can reuse tapes, but with wear, the quality of a tape degrades. The more important the data, the more precautions you should take, including using new tapes.

If a tape goes bad, mark it as "bad" and discard it. Write "bad" on the tape case before you throw it out so that someone doesn't accidentally try to use it. Never try to reuse an obviously bad tape. The cost of a new tape is minimal compared to the value of the data you are storing on it."

Backup Performance.

Sometimes data archive/extraction speed may be important, eg. a system critical to a commercial operation fails and needs restoring, or a backup/archive must be made before a deadline.

In these situations, it is highly advisable to use a fast backup medium, eg. DDS3 DAT instead of DDS1 DAT.

For example, an earlier lecture described a situation where a fault in the Ve24 hub caused unnecessary fault-hunting. As part of that process, I restored the server's system disk from a backup tape. At the time, the backup device was a DDS1 DAT. Thus, to restore some 1.6GB of data from a standard 2GB capacity DAT tape, I had to wait approximately *six hours* for the restoration to complete (since the system was needed the next morning, I stayed behind well into the night to complete the operation).

The next day, it was clear that using a DDS1 was highly inefficient and time-wasting, so a DDS3 DAT was purchased immediately. Thus, if the server ever has to be restored from DAT again, and despite the fact it now has a larger disk (4GB with 2.5GB of data typically present), even a full restoration would only take three hours instead of six (with 2.5GB used, the restoration would finish in less than two hours). Tip: as explained in the lecture on hardware modifications and installations, consider swapping a faster CPU into a system in order to speedup a backup or restoration operation - it can make a significant difference [2].

Hints and Tips.

- Keep tape drives clean. Newer tapes deposit more dirt than old ones.
- Use du and df to check that a media will have enough space to store the data. Consider using data compression options if space on the media is at a premium (some devices may have extra device files which include a 'c' in the device name to indicate it supports hardware compression/decompression, eg. a DLT drive whose raw device file is /dev/rmt/tps0d5vc). There is no point using compression options if the data being archived is already compressed with pack, compress, gzip, etc. or is naturally compressed anyway, eg. an MPEG movie, JPEG image, etc.
- Use good quality media. Do not use ordinary audio DAT tapes with DAT drives for computer data backup; audio DAT tapes are of a lower quality than DAT tapes intended for computer data storage.
- Consider using any available commands to check beforehand that a file system to be backed up is not damaged or corrupted (eg. fsck). This will be more relevant to older file system

types and UNIX versions, eg. fsck is not relevant to XFS filesystems (IRIX 6.x and later), but may be used with EFS file systems (IRIX 5.3 and earlier). Less important when dealing with a small number of items.

- Label all backups, giving full details, eg. date, time, host name, backup command used (so you or another admin will know how to extract the files later), general contents description, and your name if the site has more than one admin with responsibility for backup procedures.

- Verify a backup after it is made; some commands require specific options, while others provide a means of listing the contents of a media, eg. the -t option used with tar.

- Write-protect a media after a backup has finished.

- Keep a tally on the media of how many times it has been used.

- Consider including an index file at the very start of the backup on the media, eg.:

```
ls -AlhFR /home > /home/0000index && tar cv /home
```

  Note: such index files can be large.

- Exploit colour code schemes to denote special attributes, eg. daily vs. weekly vs. monthly tapes.

- Be aware of any special issues which may be relevant to the type of data being backed up. For example, movie files can be very large; on SGIs, tar requires the K option in order to archive files larger than 2GB. Use of this option may mean the archived media is not compatible with another vendor's version of tar.

- Consult the online guides. Such guides often have a great deal of advice, examples, etc.

tar is a powerful command with a wide range of available options and is used on UNIX systems worldwide. It is typical of the kind of UNIX command for which an admin is well advised to read through the entire man page. Other commands in this category include find, rm, etc.

Note: if compatibility between different versions of UNIX is an issue, one can use the lower-level dd command which allows one to specify more details about how the data is to be dealt with as it is sent to or received from a backup device, eg. changing the block size of the data. A related command is 'mt' which can be used to issue specific commands to a magnetic tape device, eg. print device details and default block size.

If problems occur during backup/restore operations, remember to check /var/adm/SYSLOG for any relevant error messages (useful if one cannot be present to monitor the operation in person).

Restoring Data From Backup Media.

Restoring non-root-filesystem data is trivial: just use the relevant extraction tool, eg.:

```
tar xv /dev/tape
```

However, restoring the root '/' partition usually requires access to an appropriate set of OS CD(s)

and a full system backup tape of the / partition. Further, many OSs may insist that backup and restore operations at the system level must be performed with a particular tool, eg. Backup and Restore. If particular tools were required but not used to create the backup, or if the system cannot boot to a state where normal extraction tools can be used (eg. damage to the /usr section of the filesystem) then a complete reinstallation of the OS must be done, followed by the extraction of the backup media ontop of the newly created filesystem using the original tool.

Alternatively, a fresh OS install can be done, then a second empty disk inserted on SCSI ID 2, setup to be a root disk, the backup media extracted onto the second disk, then the volume header copied over using dhvtool or other command relevant to the OS being used (this procedure is similar to disk cloning). Finally, a quick swap of the disks so that the second disk is on SCSI ID 1 and the system is back to normal. I personally prefer this method since it's "cleaner", ie. one can never be sure that extracting files ontop of an existing file system will result in a final filesystem that is genuinely identical to the original. By using a second disk in this way, the psychological uncertainty is removed.

Just like backing up data to a remote device, data can be restored from a remote device as well. An OS 'system recovery' menu will normally include an option to select such a restoration method - a full host:/path specification is required.

Note that if a filesystem was archived with a leading / symbol, eg.:

```
tar cvf /dev/tape /home/pub/movies/misc
```

then an extraction may fail if an attempt is made to extract the files without changing the equivalent extraction path, eg. if a student called cmpdw entered the following command with such a tape while in their home directory:

```
tar xvf /dev/tape
```

then the command would fail since students cannot write to the top level of the /home directory.

Thus, the R option can be used (or equivalent option for other commands) to remove leading / symbols so that files are extracted into the current directory, ie. if cmpdw entered:

```
tar xvfR /dev/tape
```

then tar would place the /home data from the tape into the cmpdw's home directory, ie. cmpdw would see a new directory with the name:

```
/home/students/cmpdw/home
```

Other Typical Daily Tasks.

From my own experience, these are the types of task which most admins will likely carry out every day:

- Check disk usage across the system.
- Check system logs for important messages, eg. system errors and warnings, possible suspected access attempts from remote systems (hackers), suspicious user activity, etc. This applies to web server logs too (use script processing to ease analysis).

- Check root's email for relevant messages (eg. printers often send error messages to root in the form of an email).

- Monitor system status, eg. all systems active and accessible (ping).

- Monitor system performance, eg. server load, CPU-hogging processes running in background that have been left behind by a careless user, packet collision checks, network bandwidth checks, etc.

- Ensure all necessary system services are operating correctly.

- Tour the facilities for general reasons, eg. food consumed in rooms where such activity is prohibited, users who have left themselves logged in by mistake, a printer with a paper jam that nobody bothered to report, etc. Users are notoriously bad at reporting physical hardware problems - the usual response to a problem is to find an alternative system/device and let someone else deal with it.

- Dealing with user problems, eg. "Somebody's changed my password!" (ie. the user has forgotten their password). Admins should be accessible by users, eg. a public email address, web feedback form, post box by the office, etc. Of course, a user can always send an email to the root account, or to the admin's personal account, or simply visit the admin in person. Some systems, like Indy, may have additional abilities, eg. video conferencing: a user can use the InPerson software to request a live video/audio link to the admin's system, allowing 2-way communication (see the inperson man page). Other facilities such as the talk command can also be employed to contact the admin, eg. at a remote site. It's up to the admin to decide how accessible she/he should be - discourage trivial interruptions.

- Work on improving any relevant aspect of system, eg. security, services available to users (software, hardware), system performance tuning, etc.

- Cleaning systems if they're dirty; a user will complain about a dirty monitor screen or sticking mouse behaviour, but they'll never clean them for you. Best to prevent complaints via regular maintenance. Consider other problem areas that may be hidden, eg. blowing loose toner out of a printer with an air duster can.

- Learning more about UNIX in general.

- Taking necessary breaks! A tired admin will make mistakes.

This isn't a complete list, and some admins will doubtless have additional responsibilities, but the above describes the usual daily events which define the way I manage the Ve24 network.

Useful file: /etc/motd

The contents of this file will be echoed to stdout whenever a user activates a login shell. Thus, the message will be shown when:

- a user first logs in (contents in all visible shell windows),

- a user accesses another system using commands such as rlogin and telnet,

- a user creates a new console shell window; from the man page for console, "The console provides the operator interface to the system. The operating system and system utility programs display error messages on the system console."

The contents of /etc/motd are not displayed when the user creates a new shell using 'xterm', but is displayed when winterm is used. The means by which xterm/winterm are executed are irrelevant (icon, command, Toolchest, etc.)

The motd file can be used as a simple way to notify users of any developments. Be careful of allowing its contents to become out of date though. Also note that the file is local to each system, so maintaining a consistent motd between systems might be necessary, eg. a script to copy the server's motd to all clients.

Other possible ways to inform users of worthy news is the xconfirm command, which could be included within startup scripts, user setup files, etc. From the xconfirm man page:

> "xconfirm displays a line of text for each -t argument specified (or a file when the -file argument is used), and a button for each -b argument specified. When one of the buttons is pressed, the label of that button is written to xconfirm's standard output. The enter key activates the specified default button. This provides a means of communication/feedback from within shell scripts and a means to display useful information to a user from an application. Command line options are available to specify geometry, font style, frame style, modality and one of five different icons to be presented for tailored visual feedback to the user."

For example, xconfirm could be used to interactively warn the user if their disk quota has been exceeded.

UNIX Fundamentals: System bootup and shutdown, events, daemons.

SGI's IRIX is based on System V with BSD enhancements. As such, the way an IRIX system boots up is typical of many UNIX systems. Some interesting features of UNIX can be discovered by investigating how the system starts up and shuts down.

After power on and initial hardware-level checks, the first major process to execute is the UNIX kernel file /unix, though this doesn't show up in any process list as displayed by commands such as ps.

The kernel then starts the init program to begin the bootup sequence, ie. init is the first visible process to run on any UNIX system. One will always observe init with a process ID of 1:

```
% ps -ef | grep init | grep -v grep
   root          1          0  0 21:01:57 ?       0:00 /etc/init
```

init is used to activate, or 'spawn', other processes. The /etc/inittab file is used to determine what processes to spawn.

The lecture on shell scripts introduced the init command, in a situation where a system was made to reboot using:

```
init 6
```

The number is called a 'run level', ie. a software configuration of the system under which only a selected group of processes exist. Which processes correspond to which run level is defined in the /etc/inittab file.

A system can be in any one of eight possible run levels: 0 to 6, s and S (the latter two are identical). The states which most admins will be familiar with are 0 (total shutdown and power off), 1 (enter system administration mode), 6 (reboot to default state) and S (or s) for 'single-user' mode, a state commonly used for system administration. The /etc/inittab file contains an 'initdefault' state, ie. the run level to enter by default, which is normally 2, 3 or 4. 2 is the most common, ie. the full multi-user state with all processes, daemons and services activated.

The /etc/inittab file is constructed so that any special initialisation operations, such as mounting filesystems, are executed before users are allowed to access the system.

The init man page has a very detailed description of these first few steps of system bootup. Here is a summary:

An initial console shell is created with which to begin spawning processes. The fact that a shell is used this early in the boot cycle is a good indication of how closely related shells are to UNIX in general.

The scripts which init uses to manage processes are stored in the /etc/init.d directory. During bootup, the files in /etc/rc2.d are used to bring up system processes in the correct order (the /etc/rc0.d directory is used for shutdown - more on that later). These files are actually links to the equivalent script files in /etc/init.d.

Each file in /etc/rc2.d (the 2 presumably corresponding to run level 2 by way of a naming convention) all begin with S followed by two digits (S for 'Spawn' perhaps), causing them to be executed in a specific order as determined by the first 3 characters of each file (alphanumeric). Thus, the first file run in the console shell is /etc/rc2.d/S00anounce (a link to /etc/init.d/announce - use 'more' or load this file into an editor to see what it does). init will run the script with appropriate arguments depending on whether the procedure being followed is a startup or shutdown, eg. 'start', 'stop', etc.

The /etc/config directory is used by each script in /etc/init.d to decide what it should do. /etc/config contains files which correspond to files found in /etc/rc2.d with the same name. These /etc/config files contain simply 'on' or 'off'. The chkconfig command is used to test the appropriate file by each script, returning true or false depending on its contents and thus determining whether the script does anything. An admin uses chkconfig to set the various files' contents to on or off as desired, eg. to switch a system into stand-alone mode, turn off all network-related services on the next reboot:

```
chkconfig network off
chkconfig nfs off
ckkconfig yp off
chkconfig named off
init 6
```

Enter chkconfig on its own to see the current configuration states.

Lower-level functions are performed first, beginning with a SCSI driver check to ensure that the system disk is going to be accessed correctly. Next, key file systems are mounted. Then the

following steps occur, IF the relevant /etc/config file contains 'on' for any step which depends on that fact:

- A check to see if any system crash files are present (core dumps) and if so to send a message to stdout.
- Display company trademark information if present; set the system name.
- Begin system activity reporting daemons.
- Create a new OS kernel if any system changes have been made which require it (this is done by testing whether or not any of the files in /var/sysgen are newer than the /unix kernel file).
- Configure and activate network ports.
- etc.

Further services/systems/tasks to be activated if need be include ip-aliasing, system auditing, web servers, license server daemons, core dump manager, swap file configuration, mail daemon, removal of /tmp files, printer daemon, higher-level web servers such as Netscape Administration Server, cron, PPP, device file checks, and various end-user and application daemons such as the midi sound daemon which controls midi library access requests.

This isn't a complete list, and servers will likely have more items to deal with than clients, eg. starting up DNS, NIS, security & auditing daemons, quotas, internet routing daemons, and more than likely a time daemon to serve as a common source of current time for all clients.

It should be clear that the least important services are executed last - these usually concern user-related or application-related daemons, eg. AppleTalk, Performance Co-Pilot, X Windows Display Manager, NetWare, etc.

Even though a server or client may initiate many background daemon processes on bootup, during normal system operation almost all of them are doing nothing at all. A process which isn't doing anything is said to be 'idle'. Enter:

```
ps -ef
```

The 'C' column shows the activity level of each process. No matter when one checks, almost all the C entries will be zero. UNIX background daemons only use CPU time when they have to, ie. they remain idle until called for. This allows a process which truly needs CPU cycles to make maximum use of available CPU time.

The scripts in /etc/init.d may startup other services if necessary as well. Extra configuration/script files are often found in /etc/config in the form of a file called servicename.options, where 'servicename' is the name of the normal script run by init.

Note: the 'verbose' file in /etc/config is used by scripts to dynamically redefine whether the echo command is used to output progress messages. Each script checks whether verbose mode is on using the chkconfig command; if on, then a variable called $ECHO is set to 'echo'; if off, $ECHO is set to something which is interpreted by a shell to mean "ignore everything that follows this symbol", so setting verbose mode to off means every echo command in every script (which uses the $ECHO test and set procedure) will produce no output at all - a simple, elegant and clean way of

controlling system behaviour.

When shutting a system down, the behaviour described above is basically just reversed. Scripts contained in the /etc/rc0.d directory perform the necessary actions, with the name prefixes determining execution order. Once again, the first three characters of each file name decide the alphanumeric order in which to execute the scripts; 'K' probably stands for 'Kill'. The files in /etc/rc0.d shutdown user/application-related daemons first, eg. the MIDI daemon. Comparing the contents of /etc/rc2.d and /etc/rc0.d, it can be seen that their contents are mirror images of each other.

The alphanumeric prefixes used for the /etc/rc*.d directories are defined in such a way as to allow extra scripts to be included in those directories, or rather links to relevant scripts in /etc/init.d. Thus, a custom 'static route' (to force a client to always route externally via a fixed route) can be defined by creating new links from /etc/rc2.d/S31network and /etc/rc0.2/K39network, to a custom file called network.local in /etc/init.d.

There are many numerical gaps amongst the files, allowing for great expansion in the number of scripts which can be added in the future.

References:

1. Extreme Technologies:

   ```
   http://www.futuretech.vuurwerk.nl/extreme.html
   ```

2. DDS1 vs. DDS3 DAT Performance Tests:

   ```
   http://www.futuretech.vuurwerk.nl/perfcomp.html#DAT1
   http://www.futuretech.vuurwerk.nl/perfcomp.html#DAT2
   http://www.futuretech.vuurwerk.nl/perfcomp.html#DAT3
   http://www.futuretech.vuurwerk.nl/perfcomp.html#DAT4
   ```

3. "Success With DDS Media", Hewlett Packard, Edition 1, February 1991.