1

The Beginning

> *The beginning of knowledge is the discovery of something we do not understand.*
> Frank Herbert

1.1 Outline

This book explores the distribution of fundamental network services in the UNIX world based on a client-server model. Historically the *Network Information System*, NIS, together with the *Network File System*, NFS, both developed by Sun Microsystems, have been employed frequently for this purpose. Here we will present a different approach, mainly characterized by the application of OpenAFS, which to some degree resembles the former Distributed Computing Environment, DCE, a software system developed in the 1990s by a consortium of software and hardware companies, including Apollo Computer (later part of the Hewlett-Packard Company), Digital Equipment Corporation (bought by Compaq which subsequently merged with Hewlett-Packard, too), and the International Business Machines Corporation.

This book is divided into three parts, providing a live description of services running on UNIX servers. The first part of the book describes the fundamental architecture of our software environment, from the basic services such as DNS and NTP, to the core consisting of Kerberos V, OpenLDAP, OpenAFS, and Samba as a gateway to the Windows world.

The second part includes additional services such as DHCP, TFTP, a Certificate Authority, and an emergency operating system which clients could boot directly from the network in case of system failures. On the top of the backbone services, the book will provide an overview of web and message services such as email, news, and mailing lists.

The third part is dedicated to the description of various application scenarios, including a basic cluster setup, a laboratory installation and additional collaborative services such as an instant messaging system and source version control services.

One comment should be added here about the realization of this book. All the services have been actually implemented on real machines connected to a network, and the output has been recorded live through the `script` UNIX

command, which creates a complete typescript of a terminal session. Occasionally the other command `screendump` got used to record the screenshot of a given terminal.

Conventions

This book requires a basic knowledge of the functioning of a UNIX system and an essential networking background. Our exemplary domain will be named `example.edu`, referring to our hypothetical institution named "*Example Organization*".

In the following occur many screenshots where the UNIX shell prompt will appear: the convention used in this book is to indicate with a dollar sign "`$`" a user prompt, and a `root` shell prompt with the pound "`#`". This convention is often adopted by the shell commands themselves. Command outputs may exceed the limits imposed by typographic margins, so to indicate that a particular line continues on the following, we will use the backslash "`\`" character. Not all the output will be reproduced in cases where considered not necessary.

Each chapter ends with a practical part suggesting exercises. These are just hints to reflect further the material presented, with the last one generally significantly more difficult.

1.2 Preparation

This section shows a really minimal Debian GNU/Linux installation, a rather server oriented distribution, started in 1993 by Ian Murdock, at the time a student at the Purdue University[1]. Its primary objective is to provide a free and open source operating system based on the UNIX tools released by the GNU Project, started by Richard Stallman in 1983.

Linux itself is essentially a kernel, and a complete UNIX-like operating system is obtainable in the form of *distributions* provided by independent vendors, either commercial as RedHat and Novell (former SuSE Linux), or by organizations developing the distribution free of charge such as the Debian GNU/Linux distribution. A notable non-commercial distribution is Slackware, started by Patrick Volkerding, which was one of the first distribution and today the oldest. We have been choosing the Debian GNU/Linux distribution because of its known stability and long-term support of packages, making it a suitable option for servers; moreover, it provides many integrated packages with an advanced management system that installs all dependent packages when needed. Several distributions have been choosing this software maintaining system, notably Ubuntu, an offspring of Debian and sponsored by

---

[1] The name "Debian" comes from the initials of Ian, and his girlfriend, and now wife, Debra.

Canonical Ltd. founded by Mark Shuttleworth with the objective of promoting the free software.

A minimal CD-ROM ISO image for a Debian system requires about 200 MB of free disk space, and provides a bootable CD-ROM image. The Debian installer guides the user through the setup process with a command-line based wizard, allowing an easy partitioning of hard drives with the choice of several file system types. A working network is vital during the installation of a Debian system, since all the needed packages will be downloaded from remote repositories: any ISP provides top-level network access to DNS services, and there are DNS services free of charge, too.

The installation process creates besides the administrative `root` user another one which we call `admin`. It is sufficient to configure the mail transport agent `exim` for local delivery only, redirecting mail for `root` to this second user `admin`. Anyway, the `exim` settings are not critical since they are going to be disabled in the following.

Debian Basics

We want to secure the freshly installed system as much as possible from the very beginning. Since by default Debian activates some services and opens some ports, we will manually correct this before we go on. To perform the following operation we need to gain `root` access. In general it is not desirable to run unnecessary processes on a server: they clearly consume resources, but more critically, open ports on a networked machine which might might provide an entry point of a possible break in, posing a security threat.

Before starting to close inessential services on our new host, we perform a basic update of the Debian system. The main tool used to handle package installation and removal is `apt-get`. The program provides an easy to use interface to manage the package repository, the database of all known software to the Debian system. It is common practice to synchronize the package list with the remote software sources, done via the `update` subcommand:

```
# apt-get update
```

Once the local package list is in sync with the remote repositories, with the `upgrade` subcommand it is possible to update all outdated packages to their new version:

```
# apt-get upgrade
```

The upgrade process should perform flawlessly, resolving all conflicts with different package versions and dependencies. The `dist-upgrade` subcommand is a shortcut to perform an upgrade, and handle package dependency conflicts automatically, giving higher priority to the most critical software if needed:

```
# apt-get dist-upgrade
```

As a detailed example for the installation of a package, let us install a useful command called `less`. The `less` command is a screen pager program for files, with displaying and searching capability. First we search the name of the package with the help of the `apt-cache` tool, retrieving all the packages with a "less" string in its name or description, using the `search` subcommand:

```
# apt-cache search less
3ddesktop - "Three-dimensional" desktop switcher
aircrack - wireless WEP cracker
 ...
smstools - SMS Server Tools for GSM modems
util-vserver - tools for Virtual private servers and context switching
```

The tool is shipped with the homonymous package, which can be inspected by the same tool with the `show` subcommand followed by the package name:

```
# apt-cache show less
Package: less
Priority: standard
Section: text
Installed-Size: 256
Maintainer: Thomas Schoepf <schoepf@debian.org>
Architecture: i386
Version: 382-1
Depends: libc6 (>= 2.3.2.ds1-4), libncurses5 (>= 5.4-1), debianutils (>= 1.8)
Filename: pool/main/l/less/less_382-1_i386.deb
Size: 101816
MD5sum: 49c50edc45a6ba8faf231873fbfef6e0
Description: Pager program similar to more
 Less is a program similar to more(1), but which allows backward
 movement in the file as well as forward movement.  Also, less does not
 have to read the entire input file before starting, so with large input
 files it starts up faster than text editors like vi(1).  Less uses
 termcap (or terminfo on some systems), so it can run on a variety of
 terminals.  There is even limited support for hardcopy terminals.
 .
 Homepage: http://www.greenwoodsoftware.com/less/
```

A package information includes the list of all prerequisite packages, the current available version, and a brief description. Installing the `less` package can be done via the `apt-get` tool with the `install` subcommand followed by the package name:

```
# apt-get install less
Reading Package Lists...
Building Dependency Tree...
The following NEW packages will be installed:
  less
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 102kB of archives.
After unpacking 262kB of additional disk space will be used.
Get:1 http://mirror.switch.ch stable/main less 382-1 [102kB]
Fetched 102kB in 0s (204kB/s)
Selecting previously deselected package less.
(Reading database ... 13098 files and directories currently installed.)
```

```
Unpacking less (from .../archives/less_382-1_i386.deb) ...
Setting up less (382-1) ...
```

In case the `apt-get` command also installs any required package, eventually prompting for the user approval.

The default way of enabling and disabling services at boot time is the Debian tool `update-rc.d`, which handles the startup script links in the `rc` directories on a per run-level basis. Apart from the bare bones command line tool, we find a graphical utility more practical. For this we install the `rcconf` tool with the standard `apt-get` tool, feeding it with the `install` subcommand followed by the package name:

```
# apt-get install rcconf
```

Using this text-based graphical interface, we can start removing all the unnecessary services such as `exim4`, `inetd`, and `ppp` - a mail daemon, a super-server[2], and a point-to-point dial-up service, respectively. For the moment just `atd`, `cron`, `klogd`, `makedev`, and `sysklogd` are needed: a user-available job scheduler, a system-level command scheduler, the Linux kernel log handler, the device creating tool, and the system events logger. Any running services are not stopped by the `rcconf` interface, and need to be stopped manually, for instance the `exim4` and `inetd` server:

```
# /etc/init.d/exim4 stop
Stopping MTA: exim4.

# /etc/init.d/inetd stop
Stopping internet superserver: inetd.
```

Afterwards the `rcconf` tool shows all the boot-time services, similar to the following output:

```
---------]] rcconf - Debian Runlevel Configuration tool [[-----------
|                                                                   |
|     [ ] anacron                                  ^                |
|     [*] atd                                      #                |
|     [*] cron                                     |                |
|     [ ] exim4                                    |                |
|     [ ] gpm                                      |                |
|     [*] klogd                                    |                |
|     [*] makedev                                  |                |
|     [ ] inetd                                    |                |
|     [ ] ppp                                      |                |
|     [ ] ssh                                      |                |
|     [*] sysklogd                                 v                |
|                                                                   |
|                                                                   |
|                                                                   |
|                                                                   |
|                                                                   |
|           <Ok>                        <Cancel>                    |
```

---

[2] We will introduce and explain such a service in the Kerberos chapter.

```
    |                                                                    |
     --------------------------------------------------------------------
```

Stopping services results in a decrease in the running process list, viewable with the standard UNIX command `ps`:

```
# ps auxg
USER       PID %CPU %MEM   VSZ  RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0  1496  512 ?        S    10:51   0:00 init [2]
root         2  0.0  0.0     0    0 ?        S    10:51   0:00 [keventd]
root         3  0.0  0.0     0    0 ?        SN   10:51   0:00 [ksoftirqd_CPU0]
root         4  0.0  0.0     0    0 ?        S    10:51   0:00 [kswapd]
root         5  0.0  0.0     0    0 ?        S    10:51   0:00 [bdflush]
root         6  0.0  0.0     0    0 ?        S    10:51   0:00 [kupdated]
root        99  0.0  0.0     0    0 ?        S    10:51   0:00 [kjournald]
root       457  0.0  0.0     0    0 ?        S    10:51   0:00 [khubd]
root      1083  0.0  0.1  1544  616 ?        Ss   10:52   0:00 /sbin/syslogd
root      1086  0.0  0.2  2208 1380 ?        Ss   10:52   0:00 /sbin/klogd
daemon    1128  0.0  0.1  1672  636 ?        Ss   10:52   0:00 /usr/sbin/atd
root      1131  0.0  0.1  1748  724 ?        Ss   10:52   0:00 /usr/sbin/cron
root      1138  0.0  0.0  1484  476 tty2     Ss+  10:52   0:00 /sbin/getty 38400 tty2
root      1139  0.0  0.0  1484  476 tty3     Ss+  10:52   0:00 /sbin/getty 38400 tty3
root      2293  0.0  0.0  1484  476 tty4     Ss+  10:58   0:00 /sbin/getty 38400 tty4
root      2294  0.0  0.0  1484  476 tty5     Ss+  10:58   0:00 /sbin/getty 38400 tty5
root      2295  0.0  0.0  1484  476 tty6     Ss+  10:58   0:00 /sbin/getty 38400 tty6
root      2301  0.0  0.3  3000 1684 tty1     Ss   10:58   0:00 -bash
root      2468  0.0  0.1  2480  864 tty1     R+   12:26   0:00 ps auxg
```

The `ps` tool shows the process list on our system, while the `netstat` command prints on the console all the network connections, statistics, and routing information, and with the `-a` option it displays both listening and non-listening sockets:

```
# netstat -a
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address          Foreign Address        State
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State        I-Node Path
unix  3      [ ]         DGRAM                   939      /dev/log
unix  2      [ ]         DGRAM                   970
```

For a more extended check of connections, we want to use the `lsof` and `nmap` security tools, installable with the usual `apt-get` command:

```
# apt-get install lsof nmap
```

The `nmap` program is a network security scanner and exploration tool, allowing many options for displaying various information about open ports and their status, running services and operating system version. For instance, we can use `nmap` to check all the open ports on the local machine both TCP and UDP with the `-sT` and `-sU` switches, respectively:

```
# nmap -sT -sU localhost
```

On UNIX systems all network connections are usually handled by files (e.g. sockets or pipes), and the `lsof` command is a practical tool to inspect all the open files in a system. It can be fed with the `-i` switch to show all the Internet connections, followed by the IP version number, i.e. `4` for IPv4 and `6` for the new IPv6 protocol:

```
# lsof -i4
```

Nothing should be shown open and that is the clean state we want to start out with.

Practice

Exercise 1. Test different file systems of your choice like `ext2`, `ext3`, and `xfs`. Use varying disk sizes too, for operations as creating the file system, checking it for inconsistencies, or producing and deleting big files. You should experience significant differences in speed.

Exercise 2. Reflect the choice of the Debian distribution in your case. Could the Ubuntu Server LTS (Long Term Support) version be an alternative? Damn Small Linux, KNOPPIX, or some Ubuntu LiveCD can give you a first impression of a Debian based distribution.

Exercise 3. To prepare for the next steps, review some available technical overview of AFS, DCE/DFS, and Microsoft's DFS. What do they have in common, and where do they differ?

Exercise 4. Examine whether you require further server hardening. Possible options may include a firewall for better service protection, SELinux for a stronger privilege separation between different services, the OpenBSD operating system as a safer choice for critical core services. All of these require significant technical skill.

Springer