

LinuxDays Luxembourg Tutorial Script

Linux Server

The Linuxdays Team

Luxembourg 25 and 26 January 2005

Version 1.1

This tutorial explains in step-by-step instructions the installation and configuration of a complete IT infrastructure for a small enterprise. It includes setting up basic network services like LDAP for user authentication, DNS for Internet name resolution and DHCP for the network configuration of client computers. Furthermore it explains how to set up a web server including the Plone content management to facilitate the writing of content for this server.

The next chapters are focusing on the configuration of a mail server, including virus and SPAM protection. The tutorial ends with the installation of a file server using SAMBA and print services using CUPS.

Contents

| | | |
|----------|---|-----------|
| 1 | Introduction | 1 |
| 1.1 | The Debian Project | 1 |
| 1.2 | Debian Package Management | 2 |
| 1.2.1 | Tools | 2 |
| 1.2.2 | Packages | 3 |
| 1.2.3 | Debian packages in practice | 3 |
| 1.3 | The Linux Kernel | 4 |
| 1.3.1 | Compiling the kernel | 5 |
| 1.4 | Booting the system | 6 |
| 1.4.1 | The Boot loader | 6 |
| 1.4.2 | The <code>init</code> process | 8 |
| 1.5 | Basic Administration | 8 |
| 1.5.1 | Processes | 8 |
| 1.5.2 | Troubleshooting your system with the logfiles | 12 |
| 2 | LDAP | 16 |
| 2.1 | What is LDAP? | 16 |
| 2.2 | OpenLDAP | 17 |
| 2.2.1 | Concepts and Architecture | 17 |
| 2.2.2 | Directory Access | 18 |
| 2.3 | Installation & Configuration | 19 |
| 2.3.1 | Creating the database | 20 |
| 2.4 | LDAP authentication | 21 |
| 2.5 | Additional features | 23 |
| 2.5.1 | System tuning | 23 |
| 2.5.2 | Replication | 23 |
| 2.5.3 | Graphical Tools | 23 |
| 2.6 | Useful Links | 24 |
| 3 | DNS (Bind 9) and DHCP | 25 |
| 3.1 | Introduction | 25 |
| 3.2 | DNS | 26 |
| 3.2.1 | Dissecting a host or domain name | 26 |
| 3.2.2 | Workings of a DNS query | 26 |
| 3.2.3 | When is DNS used anyway? | 27 |
| 3.2.4 | TLD - Top Level Domain | 28 |
| 3.2.5 | WHOIS | 28 |
| 3.2.6 | .lu domain registration | 29 |

| | | |
|----------|--|-----------|
| 3.2.7 | DNS record types | 30 |
| 3.2.8 | Debian & DNS | 30 |
| 3.2.9 | Client tools | 31 |
| 3.2.10 | bind9 configuration | 33 |
| 3.2.11 | running bind9 | 37 |
| 3.2.12 | slave nameserver | 37 |
| 3.2.13 | Security | 37 |
| 3.2.14 | Delegation & Parenting | 39 |
| 3.3 | DHCP | 39 |
| 3.3.1 | Debian & DHCP | 40 |
| 3.4 | Further Reading | 43 |
| 4 | Setting up a web server using the Plone CMS | 44 |
| 4.1 | Introduction | 44 |
| 4.1.1 | What is a Content Management System | 44 |
| 4.1.2 | Do you need a Content Management System? | 45 |
| 4.2 | Plone | 46 |
| 4.2.1 | A quick tour of Plone services | 46 |
| 4.2.2 | Zope | 47 |
| 4.3 | Installation | 48 |
| 4.3.1 | Pre-installed packages | 48 |
| 4.3.2 | Additional Packages | 48 |
| 4.3.3 | Configuring Zope | 49 |
| 4.3.4 | Installing Plone | 51 |
| 4.3.5 | Creating a Plone site | 52 |
| 4.3.6 | Adding and Editing content | 54 |
| 4.3.7 | Administering Plone Sites | 60 |
| 4.3.8 | Extending Plone | 65 |
| 4.4 | Customising the Look and Feel Plone | 67 |
| 4.5 | Proxying Plone | 72 |
| 5 | Mail and Virus checking | 75 |
| 5.1 | About this chapter | 75 |
| 5.2 | Introduction to Postfix | 75 |
| 5.2.1 | What is Postfix? | 75 |
| 5.2.2 | Where to get Postfix? | 76 |
| 5.2.3 | Why Postfix | 76 |
| 5.3 | Architecture of the Postfix system | 76 |
| 5.4 | Installing Postfix | 78 |
| 5.4.1 | Installation | 78 |
| 5.4.2 | Where are the files? | 79 |
| 5.5 | Basic configuration | 80 |
| 5.5.1 | Postfix configuration files | 81 |
| 5.5.2 | My own hostname | 82 |
| 5.5.3 | My own domain name | 82 |
| 5.5.4 | My own network addresses | 82 |
| 5.5.5 | What domain name to use in outbound mail | 83 |
| 5.5.6 | What domains to receive mail for | 83 |

| | | |
|----------|--|------------|
| 5.5.7 | What clients to relay mail from | 84 |
| 5.5.8 | What destinations to relay mail to | 84 |
| 5.5.9 | What delivery method: direct or indirect | 85 |
| 5.5.10 | What trouble to report to the postmaster | 86 |
| 5.5.11 | What you need to know about Postfix logging | 86 |
| 5.5.12 | Running Postfix daemon processes chrooted | 86 |
| 5.6 | The basic configuration files for our example | 87 |
| 5.7 | Additional configuration | 87 |
| 5.7.1 | The <code>virtual</code> table | 87 |
| 5.7.2 | The <code>aliases</code> table | 88 |
| 5.7.3 | Integrating Postfix with OpenLDAP | 89 |
| 5.8 | Virus scanner | 90 |
| 5.8.1 | AMaViS - A Mail Virus Scanner | 90 |
| 5.8.2 | Configuring Postfix to use Amavis-new | 91 |
| 5.8.3 | Configuring amavisd-new | 91 |
| 5.8.4 | Which Virus-scanner to use? | 92 |
| 5.8.5 | ClamAV | 92 |
| 5.8.6 | Installing and configuring ClamAV | 93 |
| 5.8.7 | Integrate ClamAV with AMaViS | 94 |
| 6 | Anti-Spam configuration with Amavis, SpamAssassin and Postfix | 95 |
| 6.1 | Short intro on spam | 95 |
| 6.1.1 | Risk assessment | 95 |
| 6.1.2 | Where does spam come from? | 96 |
| 6.1.3 | What characteristics can we use to detect spam? | 96 |
| 6.1.4 | Blacklists/RBLs/DNSBLs | 96 |
| 6.1.5 | Using DNSBLs in Postfix | 97 |
| 6.2 | SpamAssassin | 100 |
| 6.2.1 | Installation | 100 |
| 6.2.2 | Configuration | 101 |
| 6.2.3 | Feeding the Bayes DB | 102 |
| 6.2.4 | Testing SpamAssassin | 103 |
| 6.3 | Amavis | 104 |
| 6.3.1 | Testing | 105 |
| 6.4 | Advanced issues | 106 |
| 6.4.1 | Whitelisting/Blacklisting in SpamAssassin | 106 |
| 6.4.2 | Performance Issues | 106 |
| 6.4.3 | Getting Feedback to SpamAssassin | 107 |
| 7 | File services | 108 |
| 7.1 | What is Samba | 108 |
| 7.2 | Installation & Configuration | 108 |
| 7.2.1 | Structure of the configuration file | 109 |
| 7.3 | Samba as a simple file server | 109 |
| 7.3.1 | Global parameters | 109 |
| 7.3.2 | Per share parameters (all shares) | 110 |
| 7.4 | Parameters for file shares | 110 |
| 7.4.1 | Parameters for printer shares | 110 |

Contents

| | | |
|----------|--|------------|
| 7.4.2 | Parameters for the global printers share | 110 |
| 7.4.3 | User management | 111 |
| 7.4.4 | Testing | 111 |
| 7.4.5 | Example | 112 |
| 7.5 | Primary domain controller | 112 |
| 7.5.1 | Global settings | 113 |
| 7.5.2 | Homes share | 113 |
| 7.5.3 | Roaming Profiles | 114 |
| 7.5.4 | Netlogon share | 114 |
| 7.5.5 | Adding a workstation to the domain | 114 |
| 7.5.6 | Setting up a “Domain administrator” | 115 |
| 7.5.7 | Example | 115 |
| 7.6 | Password synchronization | 116 |
| 7.6.1 | Unix password follows Samba | 116 |
| 7.6.2 | Samba password follows Unix | 117 |
| 7.7 | Access control | 117 |
| 7.7.1 | Access control by user | 117 |
| 7.7.2 | Access control by IP | 118 |
| 7.7.3 | Unix rights granted to share users | 118 |
| 7.8 | Samba variables | 120 |
| 7.9 | Using samba with an LDAP backend | 120 |
| 7.9.1 | Motivation | 120 |
| 7.9.2 | Setting up opendap server for samba | 121 |
| 7.9.3 | Samba configuration | 121 |
| 7.9.4 | List of LDAP attributes relevant to Samba | 123 |
| 7.9.5 | Samba groupmap object | 124 |
| 7.10 | Miscellaneous Gimmicks | 124 |
| 7.10.1 | User monitoring | 124 |
| 7.10.2 | Time synchronization | 124 |
| 7.10.3 | Hiding files | 125 |
| 7.10.4 | Included configuration files | 125 |
| 8 | Print services | 126 |
| 8.1 | Setting up a Print Queue with CUPS | 126 |
| 8.1.1 | Connecting your printer | 126 |
| 8.2 | Your distribution’s printer setup tool | 127 |
| 8.3 | General printer setup with CUPS | 127 |
| 8.4 | PPDs for PostScript printers | 129 |
| 8.5 | Non-PostScript printers, GhostScript, and Foomatic | 129 |
| 8.6 | Multi-function devices from HP and Epson | 132 |
| 8.7 | Ethernet-Connected Printers | 132 |
| 8.7.1 | General | 132 |
| 8.7.2 | Assigning an IP address to a printer | 133 |
| 8.7.3 | Discovering the printer’s network protocols | 133 |
| 8.7.4 | Setting up a print queue | 134 |
| 8.8 | Networked Printing with CUPS | 136 |
| 8.8.1 | Basic CUPS configuration for remote printer auto-discovery | 136 |

| | | |
|--------|---|-----|
| 8.8.2 | High availability | 138 |
| 8.9 | Print quotas and accounting | 139 |
| 8.9.1 | LPD clients | 140 |
| 8.10 | Accessing Printers from Windows Clients via Samba | 141 |
| 8.10.1 | Sharing the CUPS queues to Windows clients | 141 |
| 8.10.2 | Emulate PostScript printers | 143 |
| 8.10.3 | More possibilities | 144 |
| 8.11 | CUPS Troubleshooting | 144 |
| 8.11.1 | RTFM - Read The F... Manual! | 144 |
| 8.11.2 | Supply all important information | 144 |
| 8.11.3 | Check device file permissions | 145 |
| 8.11.4 | Send print jobs with CUPS in debug mode | 145 |
| 8.11.5 | Where to ask | 145 |
| 8.11.6 | More about troubleshooting ... | 146 |

List of Tables

| | | |
|-----|--|----|
| 2.1 | Questions and answers for slapd during debconf | 19 |
| 2.2 | Current OpenLDAP debug level | 20 |
| 2.3 | Q&A during installation of libnss-ldap | 21 |
| 2.4 | libpam-ldap questions and answers | 22 |

List of Figures

| | | |
|-----|---|----|
| 2.1 | Simple Directory Information Tree (DIT) | 17 |
| 2.2 | gq | 24 |
| 5.1 | Postfix Architecture | 77 |
| 5.2 | Postfix in combination with AMaViS | 90 |

1 Introduction

by Eric Dondelinger

For the Server Tutorial of LinuxDays 2005, the Debian Sarge distribution was chosen. Why, should become obvious by the description of Debian.

1.1 The Debian Project

The Debian Project has its home at <http://www.debian.org/>. Debian is an operating system based on the Linux kernel (although there also is a version of Debian using Hurd), with most of the basic OS tools coming from the GNU Project - hence the name GNU/Linux.

The Debian distribution comes in a few flavours.

The *stable* distribution is the official Debian release. It is available for a number of platforms, with identical version numbers of the different packages, contrary to other distributions which favor the x86 architecture. The packages are grouped into three categories: *main*, *non-free* and *contrib*. The *main* packages are all free, i.e. complying with the Debian Free Software Guidelines (DFSG). The *non-free* packages fail to comply with the DFSG, for instance by prohibiting commercial redistribution, or by being redistributable but shareware. The *contrib* packages finally themselves comply with the DFSG, but depend on a non-free package.

The *testing* distribution is where packages end up after some measure of testing in unstable. They must be in sync with all architectures they have been built on and must be installable. They also must have fewer release-critical bugs than the version in unstable. Thereby, these packages should be close to a releaseable state. The main, non-free and contrib subdivision is identical to the stable release.

The *unstable* distribution is a snapshot of the most recent development system. It allows for testing of very recent packages, but there are no guarantees that things will actually work. Chances are they will break (though no more than on a number of other supposedly stable distributions, some would say), so you should not entrust your most important data to such a system. The main, non-free and contrib subdivision is identical to the stable release.

When the *testing* distribution is mature enough, it becomes *frozen*, meaning that only bugfixes will be accepted, but no more new code. Once the bug count is low enough, *frozen* becomes the new *stable* release, and the old *stable* one becomes obsolete. Once testing becomes frozen, a new testing distribution is created.

Each Debian distribution has its codename, taken from characters from the movie *Toy Story*. Currently, *woody* is the *stable* distribution, *sarge* is *testing*, *sid* is (permanently) *unstable*.

woody has reached a certain age, and *sarge* is poised to replace it as the *stable* distribution soon. The name of the next testing distribution will be *etch* (as is already reflected in some Debian documentation).

1 Introduction

The DFSG are a number of conditions to be fulfilled by the software. The software must be freely distributable. It must contain the source code. Derived works must be allowed. If the integrity of the author's source code must be guaranteed, the distribution of patches must still be allowed. No discrimination of persons or groups is allowed. Also, no restriction of its use for certain endeavours is allowed. The people the software is distributed to must not be subjected to an additional license. The license must not be attached specifically to the Debian project. The license must not place restrictions on the other software that is distributed along.

Examples of licenses complying with the Debian Free Software Guidelines are the *GPL* (GNU Public License), *BSD* (Berkeley Software Distribution) and *Artistic* licenses.

The ideal represented by the FSF (Free Software Foundation) resp. GNU (GNU's not UNIX) is one of collaborative development of software for the benefit of all. It is exemplified by four freedoms:

- ▶ The freedom to run the program, for any purpose (freedom 0)
- ▶ The freedom to study how the program works, and adapt it to your needs (freedom 1). Access to the source code is a precondition for this.
- ▶ The freedom to redistribute copies so you can help your neighbor (freedom 2).
- ▶ The freedom to improve the program, and release your improvement to the public, so that the whole community benefits (freedom 3). Access to the source code is a precondition for this.

A program is free software if users have all of these freedoms. Thus, you should be free to redistribute copies, either with or without modifications, either gratis or charging a fee for distribution, to anyone anywhere. This also means that you do not have to ask or pay for permission.

Debian being very dedicated to the Free Software ideals, it has found a large number of followers and benefits of a lot of support from the free software community. It is thus very easy to find documentation, questions and answers etc. about Debian GNU/Linux online. If you encounter problems, you will need some kind of help - which you should easily find on your own online, without having to pay for expensive support. Still, also commercial support can be found for Debian GNU/Linux. Updates can very easily be downloaded and installed without much fear of breakage from the Internet. Since Debian is very dedicated also to provide a stable platform, this makes it very interesting for use on servers.

Many other distributions are based on Debian, such as Knoppix, Ubuntu, etc.

1.2 Debian Package Management

1.2.1 Tools

When getting to know Debian, you will encounter a number of different tools for package management. The first one, during installation, is *tasksel*. Then, you will likely encounter the older *dselect* or the newer *aptitude*, which are "text-GUIs". At the lower level, you will see *dpkg*, which

does more or less the same as *rpm* on other distributions (RedHat, Fedora, SuSE, Mandrake, ...). *apt* then is a manager that automatically resolves dependencies, handles fetching of necessary packages etc.

1.2.2 Packages

Debian packages come as binary (.deb) or source (.dsc), which are manipulated respectively with *dpkg* and *dpkg-source*. Each package defines its dependencies, as declared by the package maintainer. Thus, if package B depends on package A, an attempt to install B without A already being on the system will result in an error message. Debian packages also have priorities. *Required* packages are essential for the system to function. *Important* packages should be found on any Unix-like system. *Standard* packages are usual on Linux character-mode systems. *Optional* packages include stuff that's reasonable to want, even if it's not necessarily all used - this includes X11. *Extra* packages either conflict with others of higher priority, or are very specialized stuff, or have requirements making them unsuitable for *optional*.

1.2.3 Debian packages in practice

When a Debian system is installed, a source for the packages is defined. This can be a number of CD-ROMs, HTTP, FTP servers etc. These sources are noted down in `/etc/apt/sources.list`. These sources define which Debian distribution will be used, and whether only *main* packages will be considered, or *non-free* and *contrib* too. Normally, a standard Debian mirror such as ftp.belnet.be will be noted here:

```
deb ftp://ftp.belnet.be/debian/ testing main
deb-src ftp://ftp.belnet.be/debian/ testing main
deb http://security.debian.org/ stable/updates main
```

Additional sources may provide packages not normally found in a Debian distribution. This may include multimedia packages such as *mplayer* etc.

It is also possible to mix packages from different Debian distributions. The file `/etc/apt/preferences` can define which distribution should be given priority. Individual packages can also be specified to be installed from a certain distribution (apt-pinning). Especially in a server environment, this mixing is not recommended, but can be necessary (example: webserver needing latest version of PHP, unavailable in stable, but in testing or unstable). This mixing is made heavy use of in Knoppix, which may cause problems later when installed to hard disk and debianized.

If all is configured well, an upgrade of the system is as easy as:

```
apt-get update
apt-get upgrade
```

These commands will respectively update the package list (these are downloaded from the sources) and upgrade any packages that have higher version numbers than the ones installed.

To select which packages should actually be installed, there are several possibilities. A very rough one is *tasksel*, which is normally run during the installation. It allows for a very rough

1 Introduction

selection of which sort of tasks should be fulfilled by the system (desktop system, mailserver, webserver, ...). The most direct way is by directly calling *apt-get* to install specific packages along with their dependencies:

```
apt-get install <package name>
```

The trouble there of course is that you first need to know the package name. So the more user-friendly text-oriented tools for the task of updating/selecting/installing/removing/... packages are *dselect* and *aptitude*. They allow searching for packages, selecting them, they automatically show you which packages will be installed along, which are recommended or suggested by the current selection etc. Once everything is selected, you may then fetch and install these packages. The packages are intermittently stored under `/var/cache/apt/archives`, but are normally deleted after a successful installation.

When individual packages are installed, certain scripts can be run - pre- and post-install. These may configure the package, possibly asking for input from the system administrator. If the first configuration goes wrong, this can be re-done by issuing the command `dpkg-reconfigure <package name>`. There are two typical examples: reconfiguring X11 (the graphical system) or the console keyboard layout. The corresponding package names are *xserver-xfree86* and *console-data*.

1.3 The Linux Kernel

by Patrick Harpes

The heart of the Linux system is its kernel.

The Linux kernel includes true multitasking, virtual memory, shared libraries, demand loading, shared copy-on-write executables, proper memory management, and TCP/IP networking. Today Linux is a module-loading monolithic kernel. Device drivers and kernel extensions typically run in ring 0, with full access to the hardware, although some run in user space. Unlike standard monolithic kernels, device drivers are easily configured as modules, and loaded or unloaded while running the system. Also unlike standard monolithic kernels, device drivers can be pre-empted under certain conditions. This latter feature was added to handle hardware interrupts correctly, and to improve support for symmetric multiprocessing.

Because the kernel is the most important part of the operating system, it's necessary to adapt and to optimise this piece of software to your needs. All Linux distribution comes with a precompiled kernel, including many features and many device drivers, to be sure to run on nearly all hardware. Once the system is installed it is recommended to recompile the kernel and adapt it to your specific needs. Leave out all features, all drivers that you don't need. The second reason why to recompile the kernel is due to the fact that the kernel included into your distribution has normally be compiled towards the i386 instruction set. If you have newer processors, it may be better to recompile the kernel to take benefit of new processors features and performance advantages.

1.3.1 Compiling the kernel

Compiling a new kernel is not difficult. Here are the main steps to download, configure, compile and install it.

The source code for the Linux kernel can be downloaded from kernel.org (<http://www.kernel.org/>).

1.3.1.1 Kernel Compilation: Requirements

- ▶ Kernel source package can be downloaded from www.kernel.org

Odd kernel versions are development kernels

2.1, 2.3, 2.5, 2.7

Even kernel versions are stable versions

2.0, 2.2, 2.4, 2.6

Minor version numbers don't specify the stability of the kernel

2.4.23 -> stable version

2.5.4 -> unstable version

Latest stable kernel is 2.6.10

- ▶ Recent C-Compiler gcc
- ▶ Recent GNU binutils
- ▶ Recent Ncurses dev library
- ▶ Module-init tools

1.3.1.2 Kernel Compilation- Steps

Untar the kernel source into `/usr/src`

Create a link to the kernel source:

```
# ln -s /usr/src/linux-2.6.10 /usr/src/linux
```

Configure the kernel

```
# make menuconfig
```

Make the kernel

For kernel < 2.6

1 Introduction

```
# make dep
# make bzImage
# make modules
# make modules_install
```

For kernel ≥ 2.6

```
# make bzImage
# make modules_install
```

Install the kernel image

The compiled image is located in:

```
/usr/src/linux/arch/i386/boot/bzImage
```

Copy this image into /boot

```
# cp /usr/src/linux/arch/i386/boot/bzImage /boot/vmlinuz-2.6.10
```

Copy the System.map file into /boot

```
# cp /usr/src/linux/System.map /boot/System.map-2.6.10
```

To test the new kernel it is highly recommended to add the new kernel image to your boot loader. Do not remove the old kernel image before you tested the new one.

Reboot the system

1.4 Booting the system

The boot process can be divided into 2 steps: The first step of the boot process is to locate the kernel, load it into memory and run it. This is normally done using a boot loader . Once the kernel runs and has initialised the hardware, it runs the first process called init process. This process is responsible to start the different services of the system.

1.4.1 The Boot loader

A boot loader is a program that resides in the starting sectors of a disk, e.g., the MBR (Master Boot Record) of the hard disk. After testing the system during boot up, the BIOS (Basic Input/Output System) transfers control to the MBR if the system is set to be booted from there. Then the program residing in MBR gets executed. This program is called the boot loader. It's duty is to transfer control to the operating system, which will then proceed with the boot process. For "normal" PC's there are several boot loaders available:

- ▶ GNU GRUB (Grand Unified Boot Loader)
- ▶ LILO (Linux LOader)
- ▶ NTLDR (boot loader for Windows NT systems)
- ▶ OS/2 Boot manager

In this tutorial we only want to treat the GNU GRUB boot loader. As every boot loader, GRUB has to be configured. On a Debian system this configuration is stored into `/boot/grub/menu.lst`.

Sample menu.lst file:

```
default=0
timeout=10
splashimage=(hd0,0)/grub/splash.xpm.gz
title Debian Linux
    root (hd0,0) kernel /vmlinuz-2.4.7-10 ro
    root=/dev/hda6
    initrd /initrd-2.4.7-10.img
title Windows2000
    map (hd0,0) (hd0,2)
    map (hd0,2) (hd0,0)
    rootnoverify (hd0,2)
    chainloader +1
```

Let's go through and see what this means.

"default=0" and "timeout=10" will cause Linux to boot in 10 seconds if you don't touch anything.

"splashimage" parameter can be used to pop-up a picture during boot

"title" denotes each boot setting and the text that follows is what will appear in your menu at startup.

"root" specifies which partition contains your Linux kernel image (this may or may not be your actual root (/) partition). So, "root (hd0,0)" tells GRUB that the kernel is on the first partition of /dev/hda. You can see that GRUB has a funny numbering system, 0-3 for primary partitions and 4+ for logical partitions. The next line tells GRUB just where to find your kernel and where your actual root partition with your Linux system is.

"initrd" tells where your init ramdisk image is located.

The "map" lines under the Windows 98 section are essential for getting your installation to work. These are the magical lines that trick Windows into believing that it's installed on the first partition of the first disk. If you don't map the Windows partition to (hd0,0), Windows will destroy your partition table and you won't be able to boot anything.

"rootnoverify" tells GRUB to boot from the Windows partition, but not to attempt to mount it, and "chainloader +1" tells GRUB to chain to Windows' bootloader which will start Windows.

Now save grub.conf and exit your text editor. Unlike LILO, GRUB does not require you to run any executable after you've modified the boot configuration.

1 Introduction

1.4.2 The init process

`init` is a general purpose program that spawns new processes and restart certain programs when they exit. `init` is also responsible for running a number of programs and scripts when the system boots. Everything `init` does is controlled by the file `/etc/inittab`. The start-up process is subdivided into different run-levels. In each run-level different services can be started or stopped. The following run-levels have been defined:

- ▶ 0 is halt
- ▶ 1 is single-user
- ▶ 2 - 5 are multi-user
- ▶ 6 is reboot

For each run-level a directory with links to different start/stop scripts exists. Contents of `/etc/rcX.d` are links to `/etc/init.d/*`, depending on name `Snn*` or `Knn*` they start or stop services.

1.5 Basic Administration

by Nicolas Hoffmann

1.5.1 Processes

The basic unit of execution in UNIX (and many other operating systems) is called a process. Your login shell and all the commands you've used in this tutorial run as single processes. In some cases, one program actually consists of multiple processes communicating with each other. We will see how to list the processes currently running and how to control them.

1.5.1.1 Viewing processes

Use the `ps` command to see the processes associated with the current shell.

```
user@myserver:~$ ps
PID TTY TIME CMD
10776 pts/3 00:00:00 bash
10779 pts/3 00:00:00 ps
```

Use `ps aux` to get a full listing of all processes on the system. Since there are generally many processes to be listed, you'll want to pipe the output into a pager, such as `less`:

```

user@myserver:~$ ps aux | less
USER PID %CPU %MEM  VSZ  RSS TTY  STAT  START  TIME  COMMAND
root   1  0.0  0.0  1216  424  ?    S     2004  0:04  init [2]
root   2  0.0  0.0    0    0  ?    SW    2004  0:02  [keventd]
root   3  0.0  0.0    0    0  ?    SWN   2004  0:00  [ksoftirqd_CPU0]
root   4  0.0  0.0    0    0  ?    SW    2004  1:27  [kswapd]
root   5  0.0  0.0    0    0  ?    SW    2004  0:00  [bdflood]
root   6  0.0  0.0    0    0  ?    SW    2004  0:00  [kupdated]
root   9  0.0  0.0    0    0  ?    SW    2004  0:00  [khubd]
root  10  0.0  0.0    0    0  ?    SW    2004  0:03  [kjournald]
root  75  0.0  0.0    0    0  ?    SW    2004  0:01  [kjournald]
root  76  0.0  0.0    0    0  ?    SW    2004  14:09  [kjournald]
root  77  0.0  0.0    0    0  ?    SW    2004  0:01  [kjournald]
...

```

This table shows the meaning of each field displayed in the full listing for `ps` in the last exercise.

| | |
|----------------|--|
| field | description |
| USER | the user who owns the process |
| PID | the process id, a unique identifier assigned to each process |
| %CPU | the percentage of CPU used by each process |
| %MEM | the percentage of memory used by each process |
| TTY | the controlling terminal for the current process |
| TIME | the amount of CPU time accumulated by the current process |
| COMMAND | the command used to invoke the process |

In some cases, you may want to restrict the listing to a few processes perhaps all processes belonging to a given user, or all occurrences of a certain program. You can use the `grep` command to do the filtering. It prints lines matching a specified pattern in its input while discarding lines that don't match.

Create a pipeline using `ps aux` to get a full listing of all processes and `grep` look for all occurrences of the apache web server:

```

root@myserver:~# ps aux | grep apache
root      419  0.0  0.7  5048  1832  ?    S     2004  0:01  /usr/local/apache/bin/httpd
nobody  12713  0.0  1.9  7616  5096  ?    S     06:25  0:00  /usr/local/apache/bin/httpd
nobody  12714  0.0  1.9  7616  5096  ?    S     06:25  0:00  /usr/local/apache/bin/httpd
nobody  12715  0.0  1.9  7616  5096  ?    S     06:25  0:00  /usr/local/apache/bin/httpd
nobody  12716  0.0  1.9  7616  5096  ?    S     06:25  0:00  /usr/local/apache/bin/httpd
nobody  12717  0.0  1.9  7616  5096  ?    S     06:25  0:00  /usr/local/apache/bin/httpd
nobody  13323  0.0  1.9  7616  5096  ?    S     06:29  0:00  /usr/local/apache/bin/httpd
root      4705  0.0  0.1  1336  468 pts/0  S     09:20  0:00  grep apache

```

1 Introduction

Use top to display the processes using the most CPU time. To exit top when you're done, press 'q'.

```
root@myserver:~# top

09:23:41 up 161 days, 22:28, 1 user, load average: 0.18, 0.33, 0.35
68 processes: 66 sleeping, 2 running, 0 zombie, 0 stopped
CPU states: 18.0% user, 3.5% system, 0.0% nice, 78.4% idle
Mem: 257212K total, 238968K used, 18244K free, 66312K buffers
Swap: 506512K total, 43468K used, 463044K free, 22756K cached
PID  USER  PRI  NI  SIZE  RSS  SHARE  STAT  %CPU  %MEM  TIME  COMMAND
20594 snort  17   0 46408 26M  2012    R   11.0 10.6 615:54 snort
5221  root   18   0   956  956   748    R    5.5  0.3   0:00 top
1     root   8    0   472  432   412    S    0.0  0.1   2:03 init
2     root   9    0    0    0     0     SW   0.0  0.0   0:00 keventd
3     root  19  19    0    0     0     SWN  0.0  0.0   0:03 ksoftirqd_CPU0
4     root   9    0    0    0     0     SW   0.0  0.0   0:48 kswapd
5     root   9    0    0    0     0     SW   0.0  0.0   0:00 bdflush
6     root   9    0    0    0     0     SW   0.0  0.0   0:00 kupdated
7     root   9    0    0    0     0     SW   0.0  0.0   0:00 khubd
8     root   9    0    0    0     0     SW   0.0  0.0   6:24 kjournald
100  daemon 9    0   372  300   300    S    0.0  0.1   0:00 portmap
286  root   9    0   780  768   692    S    0.0  0.2   0:34 syslogd
289  root   9    0  1136  392   392    S    0.0  0.1   0:00 klogd
305  root   9    0   412  364   364    S    0.0  0.1   0:00 inetd
319  root   9    0  1080  896   896    S    0.0  0.3   0:00 safe_mysqld
354  mysql  9    0 21416 7812  5244    S    0.0  3.0   0:00 mysqld
356  mysql  9    0 21416 7812  5244    S    0.0  3.0   0:00 mysqld
```

The displayed data are : Uptime, user count, load average, process state counters CPU State(user%, system%, nice%, idle%) Memory Usage(av, used, free, shared, buff) Swap(av, used, free, cached).

1.5.1.2 Controlling processes

Up to this point in the tutorial, you've been running just one process at a time. However, since UNIX is a multi-tasking operating system, you are not limited to a single process – you can run an arbitrary number of processes simultaneously.

It is possible to run a process in the background. In this case, the program is launched and control is returned to the shell immediately, allowing you to run other commands. In the meantime, the initial process continues to run in the background. To place a process in the background when you launch it, add the ampersand character ('&') to the end of the command line (e.g., "myprog &").

The kill command can send a signal to a running process. Syntax : kill -signal-number PID . To kill a process. i.e. force it to end, we use the signal number 9 (SIGKILL) : this kills the process whatever his state, even when crashed or unstable.

Run sleep in the background. Then use ps and grep to find its process id and use kill to terminate it.

```

user1@myserver:~$ sleep 600 &
[1] 10897
user1@myserver:~$ jobs
[1]+  Running sleep 600 &

user1@myserver:~$ ps aux | grep sleep
user1 10897 0.0 0.0 1708 468 pts/6 S 09:35 0:00 sleep 600
user1 10899 0.0 0.0 1276 416 pts/6 S 09:35 0:00 grep sleep
user1@myserver:~$ kill -9 10897
[1]+  Killed sleep 600

```

To confirm the process has been terminated, you can repeat the jobs, or ps aux | grep sleep command above:

```

user1@myserver:~$ jobs
user1@myserver:~$ ps aux | grep sleep
user1 10901 0.0 0.0 1276 416 pts/6 S 09:37 0:00 grep sleep

```

If your server is suddenly slow or if some processes don't respond, it's a good idea to use top to verify the load on the system and the CPU user of your processes. You might discover a crashed process using all the resources of the server.

```

root@myserver:~# top
09:23:41 up 161 days, 22:28, 1 user, load average: 3.18, 0.33, 0.35
68 processes: 66 sleeping, 2 running, 0 zombie, 0 stopped
CPU states: 48.0% user, 33.5% system, 0.0% nice, 18.4% idle
Mem: 257212K total, 238968K used, 18244K free, 66312K buffers
Swap: 506512K total, 43468K used, 463044K free, 22756K cached
PID   USER  PRI  NI  SIZE  RSS  SHARE  STAT %CPU %MEM TIME  COMMAND
20594 snort  17   0 46408 26M  2012    R  97.0 10.6 815:54 sendmail
5221  root   18   0  956  956   748    R   1.5  0.3  0:00 top
1     root   8    0  472  432   412    S   0.0  0.1  2:03 init
2     root   9    0    0    0     0     SW   0.0  0.0  0:00 keventd
3     root  19  19    0    0     0     SWN  0.0  0.0  0:03 ksoftirqd_CPUO
4     root   9    0    0    0     0     SW   0.0  0.0  0:48 kswapd
5     root   9    0    0    0     0     SW   0.0  0.0  0:00 bdflush
6     root   9    0    0    0     0     SW   0.0  0.0  0:00 kupdated
7     root   9    0    0    0     0     SW   0.0  0.0  0:00 khubd
8     root   9    0    0    0     0     SW   0.0  0.0  6:24 kjournald
100  daemon 9    0  372  300   300    S   0.0  0.1  0:00 portmap
286  root   9    0  780  768   692    S   0.0  0.2  0:34 syslogd

```

In that case, try to shut it down nicely if possible. (Most services can be managed with a control script, located in the /etc/init.d/ directory).

For example the sendmail daemon :

1 Introduction

```
user1@myserver:~$ /etc/init.d/sendmail restart
Stopping sendmail daemon: sendmail.
Starting sendmail daemon: sendmail.
```

If that doesn't work, use `kill` to stop the process, just like we have done for `sleep` before, and restart it afterwards. Check again with `top` to see if that solved the problem.

1.5.1.3 Further Reading:

<http://www.sims.berkeley.edu/~kevin/unix-tutorial/> Kevin HeardB.

1.5.2 Troubleshooting your system with the logfiles

There is a wide variety of Linux applications available, each with specific configuration files and help pages. Fortunately, most Linux applications use the `syslog` utility to export all their errors and status messages to files located in the `/var/log` directory. Checking your logfiles is often a very valuable step when you need to troubleshoot your system. Knowing the exact message that accompanies an error can be vital in researching problems in product documentation, and Web searches.

1.5.2.1 Syslog

`syslog` is the system logger : a utility for tracking and logging all system messages from the server. Each system message sent to the `syslog` has two descriptive labels associated with it that makes the message easier to handle.

The first describes the function (facility) of the application that generated it. For example, applications such as `mail` and `cron` generate messages with easily identifiable facilities named `mail` and `cron`.

The second describes the degree of severity of the message. There are 8 severity levels

| Syslog Facilities Table | Level | Keyword | Description |
|-------------------------|---------------|---------|------------------------------------|
| 0 | emergencies | | System unusable |
| 1 | alerts | | Immediate action required |
| 2 | critical | | Critical condition |
| 3 | errors | | Error conditions |
| 4 | warnings | | Warning condition |
| 5 | notifications | | Normal, but significant, condition |
| 6 | informational | | Informational messages |
| 7 | debugging | | Debug-level messages |

You can configure `syslog`'s `/etc/syslog.conf` configuration file to place messages of differing severities and facilities in different files. This procedure will be covered next.

1.5.2.2 The /etc/syslog.conf File

The files to which syslog writes each type of message received is set in the `/etc/syslog.conf` configuration file. This file consists of two columns. The first lists the facilities and severities of messages to expect and the second lists the files to which they should be logged. By default, on Debian systems, `/etc/syslog.conf` file is configured to put most of the messages in the file `/var/log/messages`. Here is a sample:

```

*.=info;*.=notice;*.=warn;\
auth,authpriv.none;\
cron,daemon.none;\
mail,news.none -/var/log/messages

```

In this case, all messages of severity "info", "notice" and "warn" are logged, but none from the mail, news, cron, daemon or authentication facilities/subsystems. You can make this logging even more sensitive by replacing the line above with one that captures all messages from debug severity and above in the `/var/log/messages` file. This may be more suitable for troubleshooting.

```

*.debug /var/log/messages

```

Some applications will additionally log to their own specific log files and directories independent of the `syslog.conf` file. Here are some common examples:

Files:

```

/var/log/maillog : Mail

```

Directories:

```

/var/log
/var/log/samba : Samba messages
/var/log/mrtg : MRTG messages

```

1.5.2.3 Activating Changes to the syslog Configuration File

Changes to `/etc/syslog.conf` will not take effect until you restart syslog. Issue this command to do so:

```

[root@myserver]#/etc/init.d/sysklogd reload

```

The simple way to check your syslog is with the `less` command :

```

[root@myserver]# less /var/log/messages

```

If you want to get new log entries to scroll on the screen as they occur, then you can use this command:

1 Introduction

```
[root@myserver]# tail -f /var/log/messages
```

Similar commands can be applied to all log files. This is probably one of the best troubleshooting tools available in Linux. Another useful command to use is `grep`. `grep` will help you search for all occurrences of a string in a log file; you can pipe it through the `less` command so that you can scroll into the results. Here is an example:

```
[root@myserver]# grep ldap /var/log/syslog | less
Jan 24 08:22:16 myserver postfix/cleanup[20123]: warning: dict_ldap_connect: Unable to bind
to server localhost as :
81 (Can't contact LDAP server)
```

1.5.2.4 Logrotate

The Linux `logrotate` utility renames and reuses system error log files on a periodic basis so that they don't use excessive disk space. The `/etc/logrotate.conf` file is the general configuration file in which you can specify the frequency with which the files are reused by `logrotate`.

Default file from a Debian system :

```
-----
# see "man logrotate" for details
# rotate log files weekly
weekly

# keep 4 weeks worth of backlogs
rotate 4

# create new (empty) log files after rotating old ones
create

# uncomment this if you want your log files compressed
#compress

# packages drop log rotation information into this directory
include /etc/logrotate.d

# no packages own wtmp, or btmp -- we'll rotate them here
/var/log/wtmp { monthly create 0664 root utmp rotate 1}
-----
```

- ▶ You can select either a weekly or daily rotation parameter. On this system, the weekly option is currently selected.
- ▶ The `rotate` parameter specifies the number of copies of log files `logrotate` will maintain. In our case, it's the 4 copy option.

- The `create` parameter creates a new log file after each rotation. The `compress` parameter performs gzip compression of the older rotated files. (depending on your applications, the size of your log files can become quite large.)

Therefore, our sample configuration file will create daily archives of all the logfiles and store them for seven days. The files will have the following names :

```
[root@myserver]# ls -l /var/log/messages*
-rw-r----- 1 root adm 2700 Jan 24 07:50 messages
-rw-r----- 1 root adm 66897 Jan 23 06:45 messages.0
-rw-r----- 1 root adm 2344 Jan 16 06:46 messages.1
-rw-r----- 1 root adm 2558 Jan 9 06:46 messages.2
-rw-r----- 1 root adm 1254 Jan 2 06:46 messages.3
```

`messages` is the current active version.

In the case of compressed files 9

```
[root@myserver]# ls -l /var/log/syslog*
-rw-r----- 1 root adm 14815767 Jan 24 06:25 syslog.0
-rw-r----- 1 root adm 456169 Jan 23 06:25 syslog.1.gz
-rw-r----- 1 root adm 459225 Jan 22 06:25 syslog.2.gz
-rw-r----- 1 root adm 467865 Jan 21 06:25 syslog.3.gz
-rw-r----- 1 root adm 471720 Jan 20 06:25 syslog.4.gz
-rw-r----- 1 root adm 472249 Jan 19 06:25 syslog.5.gz
-rw-r----- 1 root adm 471101 Jan 18 06:25 syslog.6.gz
```

Viewing the contents of the gzipped files still remains easy because the `zcat/zless/zmore/zgrep` commands can quickly output their contents to the screen.

```
[root@myserver]# zless /var/log/mail.log.2.gz
Jan 9 06:29:45 myserver postfix/smtpd[31080]: 8E50C3E811:
client=server111.crpht.lu
```

```
[111.11.1.1]
Jan 9 06:29:45 myserver postfix/cleanup[31167]: 8E50C3E811:
message-id=<BELBAWGGMQBNZEWGZKNJS@yahoo.com>
Jan 9 06:29:45 myserver postfix/smtpd[31080]: disconnect from server
111.crpht.lu[111.11.1.1]
```

```
[root@myserver]# zgrep ipop3d mail.log.3.gz
Jan 3 05:14:34 myserver ipop3d[19409]:
Login user=user1 host=host1.pt.lu [22.22.22.22] nmsgs=0/0
Jan 3 05:14:34 myserver ipop3d[19408]: Logout user=user2 host=host1.pt.lu [22.22.22.22]
nmsgs=0 ndele=0
```

1.5.2.5 Further Reading:

<http://www.linuxhomenetworking.com/> Peter Harrison

2 LDAP

by Thorsten Ries

The Lightweight Directory Access Protocol (LDAP) is a relatively simple client-server protocol for accessing directory services and is widely used for authentication and the organization of e-mail address-books. In the GNU/Linux area, OpenLDAP is the de facto standard for directory services, so this course give a first overview how to install OpenLDAP and how to use it on a GNU/Debian server to simplify daily administration with a centralized directory service.

Starting with the reasons why to use it, the course will convey the necessary knowledge to install OpenLDAP, configure and use it in an appropriate way. This will include information about schemes, classes and attributes as well as graphical tools for administration.

The tutorial will be completed with further possibilities of using OpenLDAP in a small or mid-size company.

2.1 What is LDAP?

In 1997, a RFC¹ came out that describes the Lightweight Directory Access Protocol (v3) (LDAP). This RFC2251 describes a protocol for accessing directory services, especially x.5xx-based directory services. In principle, a directory service is nothing else than a phone book. You often read in it, but the entries are written very rarely. Exactly this is the application for a directory service.

OpenLDAP.org has a very good description of a directory [?]: "A directory is like a phone book, and is not like a directory (folder) on your computer. Like a phone book, the directory holds information about a thing, like a doctor: First, you find the phone book, then you find "Doctors," then you look for the type of doctor, then you decide which doctor you want to see."

The reason for such a protocol together with the directory service itself -especially in comparison with a common database like mySQL or PostgreSQL- are manifold:

- ▶ LDAP allows the centralized storage of information about users, passwords, etc in a single place.
- ▶ Optimized for reading not for writing
- ▶ It's easy... sometimes ;-)
- ▶ LDAP is an open standard (developed from the Internet Engineering Task Force (IETF))

¹Request For Comments (<http://www.ietf.org>)

Coming from the DAP (Directory Access Protocol), LDAP runs over any connection-oriented transfer protocol (mainly TCP/IP). Thus, the usage of LDAP is independently of the Operating System (OS) and available for all common OSs.

To permit communication between different directories, the most important attribute-types are standardized and listed in the RFCs.

2.2 OpenLDAP

Well established in the world of directory services is OpenLDAP[?]. This open source implementation of the protocol is widely used in the GNU/Linux environment and in its current version 2.2.20 already stable.

Started in 1998, the suite now includes:

- ▶ slapd - a stand-alone LDAP daemon (server)
- ▶ slurpd - a stand-alone LDAP update replication daemon
- ▶ libraries implementing the LDAP protocol, and
- ▶ utilities, tools, and sample clients.
- ▶ some Java related libraries and tools

However, the development of OpenLDAP is still active and an alpha version of the next version (2.3) is already available. As one of the main topics, transactions will be included.

2.2.1 Concepts and Architecture

On the first view, OpenLDAP is nothing else than a hierarchical tree with the data in nodes and leafs [see Fig. 2.1].

Figure 2.1: Simple Directory Information Tree (DIT)

One of the major advantages of OpenLDAP (and other directory services too) is the possibility of storing several kinds of information. It is possible to store values in ASCII and also in binary format and so to store beside 'common information' as name or address also images or certificates.

A typical user entry in the directory could be similar to the following entry:

```
dn: uid=ries,ou=People,dc=germany,dc=tux-industries.com
objectclass: account
objectclass: posixAccount
objectclass: top
objectclass: shadowAccount
```

2 LDAP

```
objectclass: inetLocalMailRecipient
uid: ries
uidNumber: 1000
gidNumber: 1000
homeDirectory: /home/ries
userPassword: {crypt}32bf6gwefo7q3
mailLocalDelivery: thorsten.ries
mailRoutingAddress: ries
...
```

This example shows clearly the structure of an OpenLDAP entry. The first line (dn) is unique for the whole directory and is called 'Distinguished Name'.

The different object types are defined with the attribute 'objectclass' and a new object (dn: ...) has to be inherited from at least one other object.

Example:

Amongst other things, the object dn: uid=ries,ou=People,dc=tux-industries,dc=com is inherited from the class shadowAccount. That means, all attributes (e.g. userPassword) in this class are inherited!

Similar to other database systems, OpenLDAP is simple client-server oriented with a daemon ([slapd](#)) running on port 389 and some command-line tools to query the directory. But graphical tools exist also.

2.2.2 Directory Access

Not everyone is allowed to write into the directory. That's for sure! Thus, like other directories and databases, OpenLDAP has a relatively easy to configure access mechanism and the config entries are more or less self explaining:

```
# The userPassword by default can be changed
# by the entry owning it if they are authenticated.
# Others should not be able to see it, except the
# admin entry below
# These access lines apply to database #1 only
access to attribute=userPassword
by dn="cn=admin,dc=germany,dc=tux-industries,dc=com" write
by anonymous auth
by self write
by * none

# The admin dn has full write access, everyone else
# can read everything.
access to *
by dn="cn=admin,dc=germany,dc=tux-industries,dc=com" write
by * read
```

In this example, the admin and the user are allowed to change the password of the user (attribute `userPassword`), authentication against the password is allowed and that's it. Nobody else will see the encrypted password.

The second paragraph defines then the read access for the rest of the attributes for everyone and permits the admin to write.

2.3 Installation & Configuration

The installation of OpenLDAP on Debian GNU/Linux is normally quite easy! The packages are prebuilt and you just have to run `apt-get` for the installation of the needed packages. Thus, the first step is:

```
apt-get install slapd ldap-utils
```

This has to be done as root! During the installation, a few questions will be asked to build an initial database.

| Question | Answer |
|-----------------------|------------------------------|
| Domain name | [country].tux-industries.com |
| Organization name | Tux-Industries |
| Admin password | [password] |
| Verify password | [password] |
| Allow LDAPv2 protocol | yes (needed for postfix!) |

Table 2.1: Questions and answers for slapd during debconf

After the installation you should be able to do a first test (slapd is automatically started from the system):

```
ldapsearch -x -b "dc=[country],dc=tux-industries,dc=com" cn=admin
```

The normal start/stop-script is located in `/etc/init.d`. For debugging purposes, slapd can be started manually:

```
/usr/sbin/slapd -d <level>
```

Current debug levels are:

For the Postfix-connectivity you need to make an modification in `/etc/slapd.conf` (the main OpenLDAP configuration file): an additional schema (`misc.schema`) has to be added to allow local mail delivery.

| Level | Description |
|-------|--|
| -1 | enable all debugging |
| 0 | no debugging |
| 1 | trace function calls |
| 2 | debug packet handling |
| 4 | heavy trace debugging |
| 8 | connection management |
| 16 | print out packets sent and received |
| 32 | search filter processing |
| 64 | configuration file processing |
| 128 | access control list processing |
| 256 | stats log connections/operations/results |
| 512 | stats log entries sent |
| 1024 | print communication with shell backends |

Table 2.2: Current OpenLDAP debug level

```
# Schema and objectClass definitions
include /etc/ldap/schema/core.schema
include /etc/ldap/schema/cosine.schema
include /etc/ldap/schema/nis.schema
include /etc/ldap/schema/misc.schema
include /etc/ldap/schema/inetorgperson.schema
```

Additionally, recheck for the Postfix connectivity that the following entry in `slapd.conf` exist:

```
allow bind_v2
```

2.3.1 Creating the database

When starting with a plain installation of Debian, there should exist at least one user-account. To migrate at least this account to ldap, a special tool is available, but has to be installed:

```
apt-get install migrationtools
```

With the integration of one account, you have an initial object to work with, when adding additional users.

After this installation, you should check `/etc/migrationstools/migrate-common.ph` and modify `$DEFAULT_MAIL_DOMAIN` and `$DEFAULT_BASE`. There, some adaptations could be done, e.g. the UserIDs and GroupIDs to migrate can be bordered (it has to be ensured that the root account is still in the `/etc/passwd` and `/etc/shadow` files for the case of an OpenLDAP problem).

After that, the migration script needs to be executed:

```
cd /usr/share/migrationtools
ETC_SERVICES=/dev/null ETC_ALIASES=/dev/null ./migrate_all_online.sh
```

This command will exclude the migration of `/etc/services` and `/etc/aliases`, because these files could lead to problems and it is a good question, if it makes sense, to include `/etc/services` into the LDAP directory.

For bigger databases, the offline-migration is recommended. Therefore, the OpenLDAP has to be offline. The script will then create a LDIF file that can be imported, when OpenLDAP is started again.

2.4 LDAP authentication

After the creation of the database, you probably want to use the stored information for several purposes. It's possible to connect Postfix, samba or also Plone to a LDAP directory. However, the most used 'application' with LDAP is the user authentication and can be used just for this server or for the whole company. That means, every user has just one password for all the and computers.

To set-up this centralize user management, it is necessary, that pam (Pluggable Authentication Modules) for GNU/Linux is installed (this should normally be the case).

Additionally, some extensions for the connectivity have to be installed too:

```
apt-get install libnss-ldap
```

With the following questions and answers:

| Question | Answer |
|---------------------------------------|---------------------------------------|
| LDAP Server host address | 127.0.0.1 |
| distinguished name of the search base | dc=[country],dc=tux-industries,dc=com |
| LDAP version to use | 3 |
| database requires login | no |
| make configuration r/w by owner only | no |

Table 2.3: Q&A during installation of libnss-ldap

Before the `/etc/nsswitch.conf` has to be modified, the file rights needs to be changed to 644 (otherwise, the authentication will not work!):

```
chmod 644 /etc/nsswitch.conf
```

And add `ldap` as primary connectivity:

```
passwd: ldap compat
group: ldap compat
shadow: ldap compat
```

Next step is the installation of the pam-ldap libraries:

2 LDAP

| Question | Answer |
|---------------------------------------|--|
| Make local root Database admin | yes |
| Database requires login | no |
| Root login account | cn=admin,dc=germany,dc=tux-industries,dc=com |
| Root login password | [password] |
| Local crypt to use when changing pwds | crypt |

Table 2.4: libpam-ldap questions and answers

```
apt-get install libpam-ldap
```

Again, some questions have to be answered:

After the installation, some modifications have to be done too here. In `/etc/pam.d`, the files `common-account`, `common-auth` and `common-password` need to be changed.

The configuration on the test-system looks like that:

```
common-account:
account sufficient pam_ldap.so
account required pam_unix.so
common-auth:
auth sufficient pam_ldap.so
auth required pam_unix.so nullok_secure try_first_pass
common-password:
password sufficient pam_ldap.so
password required pam_unix.so nullok obscure min=4 max=8 md5
```

Important here is the parameter `try_first_pass` in the file `common-auth`. As pam evaluates the files sequentially, the system will normally ask for the password twice without it.

With this configuration, it should be possible to authenticate to the server, so you now can disable the existing user account(s) in `/etc/passwd` (just uncomment the user with `#`).

2.5 Additional features

2.5.1 System tuning

First thing after the installation of OpenLDAP could be the installation of the Name Server Cache Daemon. With `nscd`, the name service switch has not to query the LDAP server every time, cause the most recent and most requested directory entries are cached.

```
apt-get install nscd
```

2.5.2 Replication

If you need a more reliable directory server, a replication of the entries in the LDAP server would be an advance. Therefore, OpenLDAP provides also a tool: `slurpd`. It is directly installed with `slapd`, you just have to add a few lines to the `/etc/ldap/slapd.conf` (IMPORTANT: the `slapd` process has to be stopped before!)

```
repllogfile /var/lib/slapd/slapd.repllog
replica     host=ldapslave.germany.tux-industries.com
binddn='cn=replica,dc=germany,dc=tux-industries,dc=com'
bindmethod=simple
credentials=[password]
```

After changing the file, the complete database has to be copied to the slave server, because OpenLDAP just replicates the changes in the master db.

A copy of the directory is best done, when it is down. With

```
slapcat > copy.ldif
```

a ldif-file of the complete database is written. Add the entries on the secondary installation with

```
slapadd -l copy.ldif
```

After that, you can restart `slapd` on both machines. Before starting `slurpd` (`/etc/init.d/slurpd start`) it could make sense to change an entry in the master-db, so that a first `repllog`-file is created.

2.5.3 Graphical Tools

Well, sometimes OpenLDAP is not easy to configure or to work with on command-line. So, it is not astonishing that many graphical tools still exist to simplify daily routine work.

An overview of these tools can be found on: <http://www.openldap.org/faq/data/cache/271.html>. I will not discuss the tools here, but we mostly use `gq`. This has two reasons:

1. It is easy to install (`apt-get install gq`). There are no requirements to other installed software like `apache` or `python`...
2. The usage is quite easy

Figure 2.2: `gq`

2.6 Useful Links

- ▶ **OpenLDAP**: <http://www.openldap.org>
- ▶ **gq**: <http://biot.com/gq/>
- ▶ **Using LDAP for name resolution** (parts of this script based on this doc, thanks for that)(Torsten Landschoff)
<http://people.debian.org/~torsten/ldapnss.html>
- ▶ **LDAP on Woody** (Markus Amersdorfer): <http://aqua.subnet.at/~max/ldap>
- ▶ **Linux Authentication Using OpenLDAP** (David “Del” Elson)
<http://www.securityfocus.com/infocus/1427> + 1428

3 DNS (Bind 9) and DHCP

by Eric Dondelinger

This part of the server tutorial will concentrate on DNS and DHCP services. After a look into administrative necessities such as registering a domain, the course goes into how to set up a primary DNS server with a domain, including forward and reverse lookups, MX settings etc., a secondary server, and a DHCP server to automate the distribution of addresses. As time permits, slightly more advanced subjects such as subdomain delegation will be included.

This tutorial part will contain the following parts:

- ▶ TLDs, Registrars, RIPE, ARIN - WHOIS etc.
- ▶ registering & activating a .lu domain with DNS.lu
- ▶ basic setup of a bind9 DNS server
- ▶ record types (A, CNAME, PTR, MX, NS, SOA)
- ▶ basic tools like nslookup, dig
- ▶ configuring forward lookups for our domain (A, CNAME)
- ▶ configuring reverse lookups for our domain (PTR)
- ▶ configuring MX settings
- ▶ configuring a second bind9 server to be secondary (backup) server for our domain
- ▶ configuring dhcpd3 to distribute IP addresses, gateway and DNS server settings

As time permits:

- ▶ defining delegations for subdomains to other DNS servers (NS)
- ▶ using dhcpd3 for network boot configurations

3.1 Introduction

DNS stands for Domain Name System. It was originally created in 1983 because the machine way of addressing each other - by IP addresses - is not very friendly to human users, which prefer easy-to-remember names. Thus, DNS offers a conversion mechanism between IP addresses and names, based on a distributed database. It is described mainly by the RFCs (Requests for Comments) 1034 and 1035.

3 DNS (Bind 9) and DHCP

DHCP stands for Dynamic Host Configuration Protocol. It allows for providing a basic network configuration to machines on a network, such as its own IP address, the addresses of the DNS servers, gateway address and some more information. It can also be used to provide the complete system running on diskless clients.

3.2 DNS

3.2.1 Dissecting a host or domain name

A hostname might look something like `machine.some.domain.tld.`. That last dot after *tld* is the so-called *root domain*, *tld* stands for Top-Level Domain. The `machine.some.domain` part can be more or less freely chosen. The host name may be up to 127 levels deep, the total host name may not exceed 254 characters, with a single part of the host name (label) being limited to 63 characters. There are certain rules to what a host name may look like, it must consist of alphanumeric characters (i.e. letters and numbers), and may contain a hyphen “-” somewhere in the middle of a label - underscores “_” are explicitly not allowed. These rules are defined in RFC952. The name is a leaf of a tree structure, with the root of the tree being at its right end. The TLD is the top-level domain, *domain* in our example would be second level, *some* would be third level etc., i.e. this reads from right to left. Often, the leftmost part of the name indicates a specific service, such as `www` or `ftp`, offered by the specified host. The different levels are usually handled by different parts of the distributed DNS database system. There are 13 DNS root servers, which point to the DNS servers handling the TLDs. The DNS servers handling certain TLDs then point to the DNS servers handling second level domains under their TLD, and so on, until you get to the leaves, i.e. the wanted IP address.

3.2.2 Workings of a DNS query

When a remote machine is addressed by its host name, the client needs to get this translated to an IP address. Thus it sends a query to the DNS server which it is configured to ask. Within a company, this is often a local DNS server, for private persons, it will usually be their Internet Service Provider’s DNS server. There are now three cases:

- ▶ the DNS server queried itself handles the addressing of that particular host. In this case, it will authoritatively reply with the IP address of the requested host.
- ▶ the DNS server does not itself handle the addressing of that particular host, but already resolved its IP address, still has it in its cache, and the entry has not expired. In this case, it will again reply with the IP address of the requested host, but non-authoritatively.
- ▶ the DNS server does not itself handle the addressing of that particular host, but does not have the IP address in the cache, or the cache entry has expired. In this case, some more is happening. The DNS server will now send a query to a root server for the requested TLD. It will receive the address of an authoritative DNS server for the TLD. The DNS server will now query that server for the second level, and so on, until it has reached an authoritative DNS server for the complete hostname - at this point, it will get the IP address of the hostname, and can provide it to the client.

Example:

We request the IP address of `www.linux.lu`.

The address is not handled by the local DNS server, thus it queries a root server for who is responsible for the `.lu` TLD - which are servers of the Fondation Restena (a list of 6 machines actually, two of which belong to Restena). These servers now know who is handling DNS for `linux.lu` (two machines, one of which at P&T Luxembourg) - and these are now authoritative for `www.linux.lu`. Thus, the local DNS server has sent 3 queries to get the actual IP address of `www.linux.lu`, before being able to answer the client non-authoritatively: `213.166.63.242`

3.2.3 When is DNS used anyway?

DNS is usually not the first way of resolving a host name or IP address, it depends on the settings found in `/etc/nsswitch` - the sequence indicated there is normally “files dns”, meaning that the system will first look into the `/etc/hosts` file, and issue a DNS query only if no corresponding entry is found in that hosts file.

This mechanism can be used for a poor man’s ad-filter - insert a number of entries in the `/etc/hosts` file for known advertizement servers (`ads.doubleclick.net` etc.) and have them point at `localhost` - you’ll see a lot less uninteresting stuff. This actually even works on MS Windows systems, the file being located in `$WINDIR\system32\drivers\etc\hosts`. Normally, you will find information about the local host in there - normal `localhost` entries, but also the actual hostname, fqdn and IP address of the host [if it doesn’t use DHCP].

3.2.4 TLD - Top Level Domain

There are different types of TLDs:

- ▶ ccTLDs: Country Code TLDs, such as `.lu`, `.fr`, `.de`, ... These codes correspond to the ISO 3166 2-letter (A2) country codes.
- ▶ gTLDs: generic TLDs, such as `.com`, `.org`, `.net`, ...
- ▶ infrastructure TLD: `.arpa`
- ▶ reserved TLDs: `.example`, `.invalid`, `.localhost`, `.test`

The organisations handling these TLDs are the registries - one registry for each TLD. The actual registrations of second-level domains are handled by one or more registrars (which may be the same as the registry). The registration is then the process of a customer getting his own subdomain, telling the registrar which name servers will handle the subdomain, which are the contact and billing information etc. The registrar will add this information to its WHOIS database.

The actual definition of new TLDs is since 1998 done by the ICANN - Internet Corporation for Assigned Names and Numbers, while the distribution of IP addresses is handled by IANA - Internet Assigned Numbers Authority.

3.2.5 WHOIS

WHOIS is a protocol for querying a registry's database to determine the owner of a domain.

An example:

```
domainname:  linux.lu
domaintype:  ACTIVE
nserver:     sendar.prophecy.lu
nserver:     ns1.pt.lu
ownertype:   PRIVATE
registered:  01/09/97
source:      LUDOMAINS
adm-name:    Coutelier Thierry
adm-address: 7, rue Jacques Sturm
adm-zipcode: 2556
adm-city:    Luxembourg
adm-country: L
adm-email:   thierry.coutelier@prophecy.lu
bil-name:    Coutelier Thierry
bil-address: 7, rue Jacques Sturm
bil-zipcode: 2556
bil-city:    Luxembourg
bil-country: L
bil-email:   thierry.coutelier@prophecy.lu
tec-name:    Coutelier Thierry
tec-address: 7, rue Jacques Sturm
tec-zipcode: 2556
tec-city:    Luxembourg
tec-country: L
tec-email:   thierry.coutelier@prophecy.lu
```

On Unix systems, the whois command is often available.

```
$ whois -a linux.lu
domainname:  linux.lu
domaintype:  ACTIVE
...
```

Most registrars also provide a web interface. An example would be the one of the former monopoly, Network Solutions:

```
http://www.networksolutions.com/en\_US/whois/index.jhtml
```

Some web-based tools allow querying any WHOIS database.

```
http://www.iks-jena.de/cgi-bin/whois
```

A problem often arising with WHOIS is that you are obliged to provide real email addresses here, which then get spammed. Therefore, usually the registrars put up legal notices forbidding such use of their database, or put in place limits to querying the database. Such limits would be “no more than n queries per time unit from a single IP address” or for web interfaces, a code is displayed as a hard-to-read-by-OCR graphic which must be entered before getting results of the query.

3.2.6 .lu domain registration

How to go about a .lu domain registration? First, you need to verify that the domain you want is actually available. Availability does not mean that there’s simply no website at `www.domain.lu`. You need to check the WHOIS database. Probably the simplest way to do this is to surf to `http://www.dns.lu/` and enter the domain there. The reply will show whether the domain is already registered or still available. If available, you can now download a PDF document at `http://www.dns.lu/domain-registration/application-form.pdf`, fill in the required information and fax it back to dns.lu. If you can immediately provide the IP addresses and host names of the authoritative DNS servers, the domain can immediately be activated (within a day), otherwise it will only be reserved and will need to be activated later.

Since the DNS service must be reliable for many services to work properly, most, if not all, registrars require at least two separate DNS servers to be configured as authoritative for a domain. From experience, if possible, they should also be located on different network segments, just in case, or at least provide some other means of redundancy. For such setups, there is one “main” DNS server, the master. The other (or others) is (are) then secondary (or slave) servers, which simply mirror the data of the master. If the master becomes unavailable, any secondary can take its role.

3.2.7 DNS record types

So what kind of information is stored in DNS anyway? First, a single DNS server may be a repository for information on several domains or zones. There are several record types within these databases: A, CNAME, HINFO, PTR, MX, NS, SOA, TXT, LOC, and some more. A full list of DNS record types can be found at: `http://www.iana.org/assignments/dns-parameters`

- ▶ The A record is the anchor. The contents is a host address. It describes the IP address that a given DNS node has.
- ▶ The MX record is the mail exchanger. It describes which machines to contact with which priority to deliver email for the domain.
- ▶ The NS record is for the name servers. These are the servers to contact for DNS information within this domain.
- ▶ The SOA is the start of authority. It defines zone name, email contact, various time and refresh values (including how soon data should expire from caches).
- ▶ The PTR record is the reverse of an A record, pointing from the IP address to the host name.

3 DNS (Bind 9) and DHCP

- ▶ The TXT record contains text information associated with the name. It finds special use in SPF (Sender Policy Framework, used as antispam measure).
- ▶ The HINFO record contains host information (OS, HW, ...).
- ▶ The LOC record stores GPS data (experimental).

3.2.8 Debian & DNS

Debian offers a number of client tools for querying DNS. Among these are *host*, *nslookup*, *dig*. Such tools can be found in the package `dnsutils`.

On the server side, the main package is `bind9`, which depends on the additional packages `libisc0` and `libiscfg0`. If you want statistics about the usage of your `bind9` server, use `bindgraph`, which generates data for `rrdtool` (which is also used by MRTG and other statistics software).

For debugging resp. simply getting informed about DNS configurations, some more tools may be interesting, such as `dnstop`, `dnstracer`, `dnswalk`.

3.2.9 Client tools

Before starting on the configuration of the server, it will be useful to know how to use the typical client tools.

The most well-known one (as it's quite old and in use on other operating systems too) is `nslookup` - apparently considered deprecated though on Debian GNU/Linux. Using `nslookup`, you may simply look up the IP address of a host, or in reverse get the hostname corresponding to an IP address. Unfortunately, reverse lookups are often forgotten in the real world, causing diverse problems. So, an example usage is:

```
$ nslookup www.restena.lu
Server:          10.1.0.6
Address:         10.1.0.6#53
Non-authoritative answer:
Name:   www.restena.lu
Address: 158.64.1.39
```

So we first get to see which DNS server is answering the query. This DNS server answering is normally the one configured as default on the local system - i.e. as defined in the `/etc/resolv.conf` file in our case. The "non-authoritative" means that the queried nameserver is not an authoritative server for that host / domain. The IP address then for this particular host is 158.64.1.39.

In reverse:

```
$ nslookup 158.64.1.39
Server:          10.1.0.6
Address:         10.1.0.6#53
39.1.64.158.in-addr.arpa      name = faramir.restena.lu.
```

This time, nslookup shows us the hostname corresponding to this IP address, it would seem the original (canonical) name of Restena's webserver is actually "faramir", instead of defining *www* as a CNAME, they created a second A record.

It is possible to use nslookup directly from the command line, but it can also be used as a command interpreter, which you will get into by calling nslookup without any options. This command interpreter allows setting the DNS server to be queried, which record type should be returned (default is A) etc. An example session:

```
$ nslookup
> server localhost
Default server: localhost
Address: 127.0.0.1#53
> ss20.tux-industries.com.
Server:          localhost
Address:         127.0.0.1#53
Name:   ss20.tux-industries.com
Address: 10.35.1.1
> set type=mx
> tux-industries.com.
Server:          localhost
Address:         127.0.0.1#53
tux-industries.com      mail exchanger = 10 mailserver.tux-industries.com.
> set type=ns
> tux-industries.com.
Server:          localhost
Address:         127.0.0.1#53
tux-industries.com      nameserver = mailserver.tux-industries.com. tux-industries.com
> exit
```

The `host` command allows for essentially the same queries, but has different syntax. It is relatively simple in its output. Contrary to nslookup, it does not have an interactive mode. Example:

```
$ host www.tux-industries.com.
www.tux-industries.com is an alias for web.tux-industries.com. web.tux-industries.com ha
```

The `dig` command also allows for querying DNS servers, and is very flexible and clear in its output. It is therefore often used to troubleshoot DNS problems. It also features a batch mode, allowing to read queries from a file. Example:

```
$ dig @localhost -t mx tux-industries.com.
; <<>> DiG 9.2.4 <<>> @localhost -t mx tux-industries.com.
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 60521
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 1
```

3 DNS (Bind 9) and DHCP

```
;; QUESTION SECTION: ;tux-industries.com.          IN      MX
;; ANSWER SECTION:
tux-industries.com.  172800 IN      MX      10 mailserver.tux-industries.com.
;; AUTHORITY SECTION:
tux-industries.com.  172800 IN      NS      lxddns2005.tux-industries.com. tux-indus
;; ADDITIONAL SECTION:
mailserver.tux-industries.com. 172800 IN A      10.35.1.5
;; Query time: 1 msec ;; SERVER: 127.0.0.1#53(localhost) ;; WHEN: Mon Jan 10 11:21:35 20
```

Please consult the manpages to the client tools for more details.

3.2.10 bind9 configuration

On Debian GNU/Linux, the configuration files of bind9 reside in `/etc/bind`. The more interesting files there are `named.conf`, `named.conf.local`, `named.conf.options`. These contain the main configuration on how bind is supposed to operate.

The default contents of these files are as follows:

`named.conf`:

```
// This is the primary configuration file for the BIND DNS server named.
//
// Please read /usr/share/doc/bind9/README.Debian.gz for information on the
// structure of BIND configuration files in Debian, *BEFORE* you customize
// this configuration file.
//
// If you are just adding zones, please do that in /etc/bind/named.conf.local
include "/etc/bind/named.conf.options";
// prime the server with knowledge of the root servers zone "." {
    type hint;
    file "/etc/bind/db.root";
};
// be authoritative for the localhost forward and reverse zones, and for
// broadcast zones as per RFC 1912
zone "localhost" {
    type master;
    file "/etc/bind/db.local";
};
zone "127.in-addr.arpa" {
    type master;
    file "/etc/bind/db.127";
};
zone "0.in-addr.arpa" {
    type master;
    file "/etc/bind/db.0";
};
```

```

zone "255.in-addr.arpa" {
    type master;
    file "/etc/bind/db.255";
};
// zone "com" { type delegation-only; };
// zone "net" { type delegation-only; };
// From the release notes:
// Because many of our users are uncomfortable receiving undelegated answers
// from root or top level domains, other than a few for whom that behaviour
// has been trusted and expected for quite some length of time, we have now
// introduced the "root-delegations-only" feature which applies delegation-only
// logic to all top level domains, and to the root domain. An exception list
// should be specified, including "MUSEUM" and "DE", and any other top level
// domains from whom undelegated responses are expected and trusted.
// root-delegation-only exclude { "DE"; "MUSEUM"; };
include "/etc/bind/named.conf.local";

```

named.conf.local:

```

//
// Do any local configuration here
//
// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";

```

named.conf.options:

```

options {
    directory "/var/cache/bind";
    // If there is a firewall between you and nameservers you want
    // to talk to, you might need to uncomment the query-source
    // directive below. Previous versions of BIND always asked
    // questions using port 53, but BIND 8.1 and later use an unprivileged
    // port by default.
    // query-source address * port 53;
    // If your ISP provided one or more IP addresses for stable
    // nameservers, you probably want to use them as forwarders.
    // Uncomment the following block, and insert the addresses replacing
    // the all-0's placeholder.
    // forwarders {
    //     0.0.0.0;
    // };
    auth-nxdomain no;    # conform to RFC1035
};

```

3 DNS (Bind 9) and DHCP

By default, the Debian configuration for bind puts zone files into `/var/cache/bind`. This location can be modified via the `named.conf.options` file, using the `directory` option.

To add your own zone files, indicate these within `named.conf.local`, with a notation like the following:

```
zone "tux-industries.com" {
    type master;
    file "master.tuxindustries";
};
```

A DNS server may be primary for many different zones, and may be secondary for other zones at the same time.

3.2.10.1 forward lookups

The file `master.tuxindustries` (which may have any name) must reside in the directory indicated in the corresponding option. This file will look as follows:

```
$TTL 172800
@      IN      SOA      ss20.tux-industries.com. root.tux-industries.com. (
                                1          ; Serial
                                3600       ; Refresh
                                1200      ; Retry every 1/2 hour
                                604800    ; Expire
                                86400 ) ; Minimum ttl of 1 day
                IN      NS       ss20.tux-industries.com.
                IN      NS       lxddns2005.tux-industries.com.
tux-industries.com.      IN      MX 10  mailserver.tux-industries.com.
tux-industries.com.      IN      MX 20  lxddns2005.tux-industries.com.
localhost                IN      A      127.0.0.1
lxxdns2005                IN      A      10.35.1.1
web                       IN      A      10.35.1.6
www                       IN      CNAME  web.tux-industries.com.
```

This merits some more detail.

The first value, `$TTL`, defines the Time To Live of the record, or the time how long a non-authoritative server should cache entries resolved via this authoritative server. The number indicated here is defined in seconds, but this may be modified using a notation with `h` (hours), `d` (days), `w` (weeks).

The next entry defines the domain for which we define the entries. This may be explicitly written out here (in the example, that would be `tux-industries.com.`) - be careful not to forget the trailing dot -, or using the placeholder `@`, which refers to the corresponding zone directive in the `named.conf(.local)` file. This is the origin of the zone data, i.e. this gets appended to all entries not terminated by a trailing dot.

The IN then defines the class, where IN stands for Internet. There are other classes, but none except IN are in widespread use. Then we specify the SOA, or Start Of Authority.

The first entry right after SOA is the primary DNS server of the current domain. The next one is the email address of the administrator, the “@” being transformed into a dot. This is not intended for machine use, but for human eyes.

Now follow parenthesis with 5 values:

1. serial number (must be increased after all updates to the zone file)
2. refresh time (how often should a slave DNS server check for updates of the zone data)
3. retry delay (how soon should a slave retry if the check fails)
4. expiry time (when are zone data to be considered invalid on a slave, lacking updates)
5. negative caching TTL (how long should failed lookups be cached)

The next entries then are the NS records (which servers are in charge of the domain), the mailservers (MX), and finally hostnames, both A records and CNAMEs (aliases). You will notice that hostnames needn't be the FQDN (fully qualified domain name) of the host, as the domain part (origin) will be appended.

The MX records not only define which servers handle email for the current domain, but also the priority of these servers. Lower values here mean higher priority. In the example above, *mailserver* has a priority value of 10, which makes it into the primary mailservers for emails destined to *tux-industries.com*, while *lxddns2005* is only secondary with a value of 20. If another record were added with a wildcard for the subdomain, then that MX record would hold true for all subdomains, e.g.:

```
*.tux-industries.com. IN MX 10 mailserver.tux-industries.com.
```

Note also that CNAME records may point to hostnames from other domains. For instance, *www.linux.lu* is an alias of *sendar.prophecy.lu*, i.e. from an entirely different domain.

3.2.10.2 reverse lookups

To define the reverse lookups, we need to create a file that is in essence similar to the one containing the forward lookups. The difference is that the zone is defined as a reversed network address prepended to the *in-addr.arpa.* domain., and that it contains SOA, NS and PTR records.

So, for the 10.35.1.0/24 network, a typical file might be *master.1.35.10* - the unused trailing 0 is dropped, the sequence reversed, and some prefix is prepended. The zone name is then *1.35.10.in-addr.arpa.* (which gives us the right network when read from right to left). The PTR records then look as follows:

```
1          IN PTR  lxddns2005.tux-industries.com.
```

This record then specifies the hostname (*lxddns2005.tux-industries.com.*) corresponding to the IP address *1.1.35.10.in-addr.arpa.* - or *10.35.1.1* in “usual” notation.

Of course, the zone entry needs to be made in the */etc/bind/named.conf.local* file.

3 DNS (Bind 9) and DHCP

3.2.10.3 localhost and root hints

There are two more interesting things in the basic bind9 setup.

One of them is the lookup of the localhost network, i.e. 127.0.0.0/8, and more specifically, the 127.0.0.1 localhost entry. Since noone is really responsible for these addresses, it makes sense to add this to every local DNS server. Debian GNU/Linux has this pre-configured, in the `/etc/bind/db.local` and `/etc/bind/db.127` files.

The other one is the root hints information. When unknown addresses are queried, the DNS server needs to search for them, starting at the DNS root servers. These must therefore be known to the DNS server, the corresponding addresses are located in the `/etc/bind/db.root` file. The reference version of this zone file can be downloaded via FTP:

```
wget ftp://198.41.0.6/domain/named.root
```

3.2.11 running bind9

Once the server is configured to handle its zones, it may be started using the corresponding start script, here on Debian GNU/Linux it is:

```
/etc/init.d/bind9 start
```

The stopping is analogous. The logs of the bind server, again for Debian GNU/Linux, will be located within the `/var/log/syslog` log file.

To check whether the bind server is running, check whether processes called *named* are running, or whether port 53 is open (both tcp and udp). There are a number of possibilities for this check:

```
$ ps -aef | grep named
$ nmap -p 53 localhost
$ nmap -sU -p 53 localhost
$ lsof | grep LISTEN | grep bind
```

3.2.12 slave nameserver

As seen above, it is specified for each zone file whether a server is primary or secondary. So, to make a bind server a secondary server for a certain zone, it will suffice to specify this in the zone directive along with the IP address of the primary server. Example:

```
zone "tux-industries.com"
{
    type slave;
    file "sec.tuxindustries";
    masters {
        10.35.0.1;
    };
};
```

As can be seen, several masters (up to 10) could be defined, but that is comparatively tedious to keep synchronized - it is simpler to have a single master, with one or more slaves.

If no file is specified, the secondary server will not keep a backup file for this zone. It is highly recommended though to keep such a backup file. If the primary is down when the secondary starts, the secondary can thus read the zone data from that backup file and synchronize later, otherwise it would be unable to fulfill its task at all.

3.2.13 Security

So far all that has been seen is a simple setup to get things working. bind has a history of vulnerabilities, thus thoughts on security are in order. Why is security with DNS even an issue? If an attacker manages to redirect certain sites (think financial transactions) from the legitimate server to one of his own machines, (s)he may get to pocket someone else's money - in particular, yours, and in the process, get you into a lot of trouble.

A lot has been done to get security into bind, this tutorial will not go into all of them but some simpler things.

3.2.13.1 Determining the version of bind running

A simple query to the DNS server may reveal it's version, allowing an attacker to select his attack specifically to the version of bind you're running. Specifically, look up TXT records in the CHAOSNET class attached to the domain name version.bind:

```
dig @nameserver txt chaos version.bind.
```

This will, even on a modern version of bind (in our case, bind 9.2.4), return the correct version number. Thus, withholding this information from potential attackers may already divert a lot of script kiddies. This can be achieved through another option (add this in `/etc/bind/named.conf.options`):

```
version "none of your business";
```

3.2.13.2 Restricting queries, transfers

You may not want everyone to have access to certain zone data. Specifically, zone transfers can be a simple way for an attacker to get a nice overview of your network. Thus, such zone transfers should be restricted to your own slave nameservers. You can test whether a transfer is currently allowed by issuing the following command:

```
dig @nameserver -t axfr domain.name
```

If you get a complete listing of that domain - well, if you didn't issue the query from a slave nameserver, you may have a problem.

You may restrict the zone transfers using the directive `allow-transfer { host; network; };` where *host* is the IP address of a host that's allowed to get this information, or *network* is

3 DNS (Bind 9) and DHCP

a network address (ex. 10.35/16 for LinuxDays 2005). This directive can be added to the global configuration among the options in `/etc/bind/named.conf.options`, or within the zone definition:

```
zone "tux-industries.com" {
    type master;
    file "master.tuxindustries";
    allow-transfer { 10.35/16; };
}
```

Similarly, you can restrict simple queries to certain hosts or networks, using the `allow-query` directive.

3.2.13.3 Other security mechanisms

There are many other ways to improve security, such as using TSIG (transaction signatures), running bind in a chroot configuration (an attacker won't get to see all the rest of the server if (s)he manages to break into bind), etc. etc. This tutorial will not go into these.

3.2.14 Delegation & Parenting

If your own domain is getting crowded, or is regionally distributed, you may want to create subdomains and put the corresponding machines into their own zones. You may even want to set up separate DNS servers to handle these. In the example of tux-industries.com, we have both variants of such a setup - a central DNS server handling some subdomains itself, but delegating some to other DNS servers (yours).

The first thing to do is to create the zone files of the subdomains. This is exactly the same as previously seen. Once this is in place, the parent server will have to be configured for these. If it handles the subdomain itself, things are exceedingly simple - just add them via the normal mechanism of adding a zone definition in `/etc/bind/named.conf.local`. It won't hurt though if you add an entry for this subdomain in the parent domain's zone file:

```
example 86400 IN NS ss20.tux-industries.com.
```

Here `example(.tux-industries.com.)` is the subdomain handled by the same DNS server as `tux-industries.com`. The 86400 is the TTL, IN is the internet class, NS is the name server record, and `ss20.tux-industries.com.` is the canonical name of that name server (i.e. what's in the PTR record for it's IP address).

To delegate a subdomain to a different server, we have to refer to that machine. The entry is basically the same as previously, i.e.:

```
test 86400 IN NS dns.test.tux-industries.com.
```

Do you see the problem? We are referring to a machine that's itself within the domain we are referring to. So we have to put the IP address here, too:

```
dns.test.tux-industries.com. 86400 IN A 10.35.1.209
```

Now, we have all that's needed - `dns.test.tux-industries.com` will handle the `test.tux-industries.com` subdomain, but the parent server (here, `ss20.tux-industries.com`) knows which IP address it has to refer to.

3.3 DHCP

DHCP is the Dynamic Host Configuration Protocol. It is normally used to provide a client with an IP address, its default gateway, and the IP addresses of the nameservers. Additionally, it can be used to provide for a complete network boot using `about`, `bootp` or `PXE`, which is useful for diskless workstations or completely network-based system installations. It is relatively obvious that multiple DHCP servers active on the same subnet could easily tread one upon another, thus, unless there are very good reasons to operate otherwise, only one should be put in place.

3.3.1 Debian & DHCP

3.3.1.1 Client tools

The client tools are `dhcp-client` (installed by default), `dhcpcdump` (allows visualizing `tcpdump`-caught DHCP packets, depends on `tcpdump`), `dhcpping` (allows checking whether a DHCP server is running), `dhcpcd` (DHCP client, allows running a script when config changes), and `pump` (DHCP client from RedHat).

3.3.1.2 Server

The *dhcp3* packages (`dhcp3-server`, depends on `dhcp3-common`) provide the DHCP server shown in this tutorial. The configuration file of `dhcp3-server` resides in `/etc/dhcp3` and is named `dhcpd.conf`. The server is run via the start/stop script `/etc/init.d/dhcp3-server`. By (Debian) default the `dhcpd.conf` contains a lot of commentary and examples on how to configure the server. This includes offering a range of IP addresses for use by the DHCP clients, offering specific IPs to hosts with a specific MAC address - the MAC address being the (supposedly) unique address of the ethernet network interface.

The original Debian `dhcp.conf` looks as follows:

```
#
# Sample configuration file for ISC dhcpd for Debian
#
# $Id: dhcpd.conf,v 1.1.1.1 2002/05/21 00:07:44 peloy Exp $
#
# The ddns-updates-style parameter controls whether or not the server will
# attempt to do a DNS update when a lease is confirmed. We default to the
# behavior of the version 2 packages ('none', since DHCP v2 didn't
# have support for DDNS.)
```

3 DNS (Bind 9) and DHCP

```
ddns-update-style none;
# option definitions common to all supported networks...
option domain-name "example.org";
option domain-name-servers ns1.example.org, ns2.example.org;
default-lease-time 600;
max-lease-time 7200;
# If this DHCP server is the official DHCP server for the local
# network, the authoritative directive should be uncommented.
#authoritative;
# Use this to send dhcp log messages to a different log file (you also
# have to hack syslog.conf to complete the redirection).
log-facility local7;
# No service will be given on this subnet, but declaring it helps the
# DHCP server to understand the network topology.
#subnet 10.152.187.0 netmask 255.255.255.0 {
#}
# This is a very basic subnet declaration.
#subnet 10.254.239.0 netmask 255.255.255.224 {
# range 10.254.239.10 10.254.239.20;
# option routers rtr-239-0-1.example.org, rtr-239-0-2.example.org;
#}
# This declaration allows BOOTP clients to get dynamic addresses,
# which we don't really recommend.
#subnet 10.254.239.32 netmask 255.255.255.224 {
# range dynamic-bootp 10.254.239.40 10.254.239.60;
# option broadcast-address 10.254.239.31;
# option routers rtr-239-32-1.example.org;
#}
# A slightly different configuration for an internal subnet.
#subnet 10.5.5.0 netmask 255.255.255.224 {
# range 10.5.5.26 10.5.5.30;
# option domain-name-servers ns1.internal.example.org;
# option domain-name "internal.example.org";
# option routers 10.5.5.1;
# option broadcast-address 10.5.5.31;
# default-lease-time 600;
# max-lease-time 7200;
#}
# Hosts which require special configuration options can be listed in
# host statements. If no address is specified, the address will be
# allocated dynamically (if possible), but the host-specific information
# will still come from the host declaration.
#host passacaglia {
# hardware ethernet 0:0:c0:5d:bd:95;
# filename "vmunix.passacaglia";
# server-name "toccata.fugue.com";
#}
```

```

# Fixed IP addresses can also be specified for hosts. These addresses
# should not also be listed as being available for dynamic assignment.
# Hosts for which fixed IP addresses have been specified can boot using
# BOOTP or DHCP. Hosts for which no fixed address is specified can only
# be booted with DHCP, unless there is an address range on the subnet
# to which a BOOTP client is connected which has the dynamic-bootp flag
# set.
#host fantasia {
# hardware ethernet 08:00:07:26:c0:a5;
# fixed-address fantasia.fugue.com;
#}

# You can declare a class of clients and then do address allocation
# based on that. The example below shows a case where all clients
# in a certain class get addresses on the 10.17.224/24 subnet, and all
# other clients get addresses on the 10.0.29/24 subnet.
#class "foo" {
# match if substring (option vendor-class-identifier, 0, 4) = "SUNW";
#}

#shared-network 224-29 {
# subnet 10.17.224.0 netmask 255.255.255.0 {
# option routers rtr-224.example.org;
# }
# subnet 10.0.29.0 netmask 255.255.255.0 {
# option routers rtr-29.example.org;
# }
# pool {
# allow members of "foo";
# range 10.17.224.10 10.17.224.250;
# }
# pool {
# deny members of "foo";
# range 10.0.29.10 10.0.29.230;
# }
#}

```

As can easily be seen, it is possible to define the default domain name, name servers, how long the address received should remain valid before it needs to be renewed. There are then examples of how to assign IP addresses from a pool, or specific IPs to specific MAC addresses.

An example of a configuration, offering a configuration for the LinuxDays 2005:

```

subnet 10.35.0.0 netmask 255.255.0.0 {
    range 10.35.128.1 10.35.128.254;
    option domain-name-servers ss20.tux-industries.com;
    option domain-name "tux-industries.com";
    option routers 10.35.0.254;
    option broadcast-address 10.35.255.255;
    default-lease-time 600;
}

```

3 DNS (*Bind 9*) and DHCP

```
    max-lease-time 7200;  
}
```

For LinuxDays 2005, we use the 10.35.0.0/16 network. Within this network, the pool of IP addresses between 10.35.128.1 and 10.35.128.254 is used for DHCP. The official DNS server is ss20.tux-industries.com (10.35.0.1), the default domain name is tux-industries.com, the default gateway is 10.35.0.254, and finally the broadcast address is 10.35.255.255.

3.3.1.3 A word on network booting

The problem with network booting is that it really depends on the hardware you are using. For instance, for booting an Alpha machine from the network (say, to install Debian GNU/Linux on it), you will have to install a TFTP server (Trivial File Transfer Protocol), find out the MAC address of that Alpha box, find a boot image that will work with that machine [or even create it], possibly you will have to rename it to reflect the MAC address of the target machine, and finally point your DHCP server to that file, for that MAC address. This particular procedure is described in detail in the install instructions for Debian on Alpha. Similar contorsions are necessary for Sun (Ultra)SPARC machines or for the PXE boot method (Intel Pre-Execution Environment).

3.4 Further Reading

Paul Albitz & Cricket Liu, DNS and BIND, O'Reilly. 4th edition, April 2001, ISBN 0-596-00158-4

Wikipedia: <http://www.wikipedia.org/>

The Linux Documentation Project: <http://www.tldp.org/>

Debian documentation: `/usr/share/doc` and <http://www.debian.org/>

4 Setting up a web server using the Plone CMS

by Nico Mack

A company without a website is unthinkable - and most companies and organisations have more than one site. Whether it's an external site for communicating with customers, an intranet for employees to use, or a site for direct customer communication and feedback, all Web sites have a common problem - how to manage the content on them. This is a challenge that can often cost organisations large amounts of time and effort. That's where Content Management Systems come into play. This section of the Server Tutorial intends to give you a brief introduction of what a content management is and how it works. There won't be a rundown of all the competing open-source content management framework. We, that is the Linuxdays team, have picked Plone as our Content Management System. This tutorial will show you how to get and install Plone, where to get Extensions for Plone and last but not least, how to use it.

Parts of this tutorial are based on Andy McKays "Definitive Guide to Plone" [?] and Kamon Ayeva, Olivier Deckmyn, Pierre Julien Grizel and Maik Röder's "les Cahiers du Programmeur Zope/Plone" [?].

4.1 Introduction

A company without a website is unthinkable - and most companies and organisations have more than one site. Whether it's an external site for communicating with customers, an intranet for employees to use, or a site for direct customer communication and feedback, all Web sites have a common problem - how to manage the content on them. This is a challenge that can often cost organisations large amounts of time and effort. That's where Content Management Systems come into play. This section of the Server Tutorial intends to give you a brief introduction of what a content management is and how it works. There won't be a rundown of all the competing open-source content management framework. We, that is the Linuxdays team, have picked Plone as our Content Management System. This tutorial will show you how to get and install Plone, where to get Extensions for Plone and last but not least, how to use it.

4.1.1 What is a Content Management System

I'll start with a broad definition of *content*: Content is a unit of data with some extra information attached to it. That piece of data could be a web page, information about an upcoming event, a document, a picture, a movie clip or any piece of data that has meaning to the organisation deploying the system. All these items share similar attributes, such as the need to be added or edited by certain users and be published in various ways. A system called *workflow* controls

4 Setting up a web server using the Plone CMS

these attributes. A workflow is logic defined by the organisation's business rules, and it describes a system for managing the content. Basically it defines the different states content can be in, how it moves from one state to the other and who is allowed to change states.

The following is one definition of a CMS (<http://www.contentmanager.eu.com/history.htm> `http://www.contentmanager.eu.com/history.htm`):

A CMS is a tool that enables a variety of (centralized) technical and (decentralized) nontechnical staff to create, edit, manage and finally publish a variety of content (such as text, graphics, video and so on) whilst being constrained by a centralized set of rules, process, and workflows that ensure a coherent, validated Web site appearance.

4.1.2 Do you need a Content Management System?

To better understand what the advantages of a CMS are, I'd like to quickly take you to a tour of how the web evolved and how web sites used to be created.

In the early days, web sites were created by scientific teams driven by the sole desire to put information online. Early sites had only a couple of pages and look and style was not an issue. Later, more and more people discovered the web as a publishing media, wishing to add their personal touch to their content. They rapidly required more visual tools, thus the emergence of visual HTML editors. Still a bit later, companies and organisations discovered the potential the web had to offer to their business. They hired specialists to create, maintain and update their web sites, The professional webmaster was born. With the explosion of the web and the professionalisation of the web creation process, webmasters became a victim of their own success. Maintaining an ever growing number of static web pages also entailed an ever growing number of technical problems. Furthermore, the webmaster had keep up with ever faster evolving technologies, with a growing number of nontechnical users who do not understand the technical limits the web imposes on them. The webmaster quickly became a bottleneck and a liability in the web publishing process. A more rational approach to manage content was needed. It became more and more important to organise the management of the web site. Be it the graphical charter, the creation, the maintenance, the update, the evolution of the corporate web site: everything must be managed, decided and implemented at different levels and by different persons. What's needed, is a system that does the following:

- ▶ **Separate the page content from the presentation:** If the actual content is separated from the presentation method, then the content author doesn't need to know any HTML or how the pages are delivered. In fact, one piece of content could have many different *templates* applied to it. When you want to change to look and feel of the site, you have to change only that one template rather than all the content
- ▶ **Allows certain users to add and edit content:** If specified users can add and edit content easily, then there's no need to send everything to the Webmaster or web team. Instead, the user who wants to create a page can do so and edit it as much as necessary.
- ▶ **Applies rules to whom can publish what and when:** your business rules might not want just anybody publishing content on your Web site; for instance, people in marketing would be able to publish the press release part and not the engineering section.

- ▶ **Can apply business rules to content:** If a person from marketing creates a press release, somebody in legal might need to review that document. In this case, the document will be passed through a review process that ensures it won't go live until these reviews are done.
- ▶ **Can search and index information intelligently:** Since the CMS can keep track of structured information about the content (such as author's name, publication date, modification dates, categories and so on), it can produce a listing of content by author, recent content, and so on. It can also provide searching capabilities that are much smarter and more useful than just a simple textual search.

If the thought “*yes, that's what I need*” crossed your mind at least once while going through the list above, then the answer to the title question is: Yes, you need a CMS!

4.2 Plone

As already mentioned in the introduction, there's a plethora of open-source content management systems, Plone being one among many. Plone is the CMS of choice of the Linuxdays team, that's why we're going to use it for our tutorial.

Plone is a content management framework that works hand-in-hand and sits on top of Zope, a widely-used Open Source web application server and development system. We'll have a closer look on Zope a little bit later on.

4.2.1 A quick tour of Plone services

4.2.1.1 Member Management

A Plone website distinguishes anonymous users and members. Anonymous users have very limited permissions, in general they are only allowed to view already published content. Members on the other hand, are allowed to add and edit content. Plone offers different models of site membership. Depending on the configuration, users are centrally created by the site administrators or anonymous users may join the site by registering themselves. Members own a private area where they can manage and edit the content they are contributing to the web site. At this point, I'd like to point out that member authentication can be delegated to an external **LDAP** server.

4.2.1.2 Group Management

Plone allows to manage groups of members. Every group owns its own *workspace*. Group members have sufficient privileges to create and manage common content in their group's workspace. For instance, you could create a Marketing group and add all members of the marketing department to that group. The Marketing group's workspace would then become the place to publish press releases.

4 Setting up a web server using the Plone CMS

4.2.1.3 User Interface Management

Plone uses a *skin* based user interface model. Skins are implemented via Cascading Stylesheets (CCS2). Doing so allows to offer various presentations for one site. You may completely customise the user interface or only parts of it without having to touch the provided HTML templates. Most of the customisation is done via stylesheets.

4.2.1.4 Management of the Content Creation and Editing Process

A *content type* is a definition allowing to manage content. Content types are objects that may be customized. Default content types like News, Document, Links, etc, can be extended or modified to become new content types like a job offer, a product description, etc. The creator of the content type defines the type's behaviour and makes it available to the site members so that they can use it.

4.2.1.5 Workflow Management

Plone comes with a default publishing workflow that will be good enough for most web sites. Like most things in Plone, you can customize the workflow so that it will suite your needs. You can define for every content type, different *states*, *transitions* and *actors*.

4.2.1.6 Indexing and Search Engine

Every single content item created in Plone will automatically be indexed. This allows to quickly find them via the build in search engine.

4.2.1.7 Metadata

4.2.1.8 Version Management

4.2.1.9 Syndication

4.2.2 Zope

Zope is an open source web application server primarily written in the Python programming language. It features a transactional object database which can store not only content and custom data, but also dynamic HTML templates, scripts, a search engine, and relational database (RDBMS) connections and code. It features a strong through-the-web development model, allowing you to update your web site from anywhere in the world. To allow for this, Zope also features a tightly integrated security model. There are numerous products (plug-in Zope components) available for download to extend the basic set of site building tools. These products include new content objects; relational database and other external data source connectors; advanced content management tools; and full applications for e-commerce, content and document management, or bug and issue tracking. Zope includes its own HTTP, FTP, WebDAV, and XML-RPC serving capabilities, but can also be used with Apache or other web servers. Zope is available at <http://www.zope.org> <http://www.zope.org>

4.3 Installation

Plone and the underlying Zope framework exists for a wide variety of platforms (Linux, MacOSX, Windows...). In the scope of this tutorial we're going to limit ourselves to an installation on a Linux system, on a Debian distribution to be more specific. As with most open-source products, there are different ways to do so. You can install every single package (and all those they depend from) from source or via CVS and compile them for your machine. This approach gives you obviously the most freedom to get things installed the way you want, but will probably cause quiet a headache if you're not absolutely surefooted on command lines. That's why we're going to use Debian's package manager to install the required packages.

4.3.1 Pre-installed packages

Debian Sarge already includes a certain number of packages required for our Zope/Plone installation. These packages are:

- ▶ **Python** (v2.3.4): Python is a powerful object-oriented, open-source programming language comparable to Perl or Tcl. Most of Zope and Plone's code base is written in Python. Knowledge of Python isn't required to use Plone or even to do some basic administration; however, customizing products and scripting Plone does require some Python.
- ▶ **libjpeg62** (v6b-9): A library for handling the JPEG (JFIF) image format. Not directly required to run Zope/Plone, but optional image gallery products will need this library.
- ▶ **zlib1g** (v1.1.2): is designed to be a free, general-purpose, legally unencumbered, that is, not covered by any patents, lossless data-compression library for use on virtually any computer hardware and operating system. Again, not an absolute must to run Zope/Plone, but optional image gallery products will need this library.
- ▶ **libldap** (2.1.30): Run-time libraries for the OpenLDAP (Lightweight Directory Access Protocol) servers and clients. Will be required for LDAP user authentication.

4.3.2 Additional Packages

The following packages don't come with a standard installation, that is, we have to install them manually. Depending on whether we have a graphical environment or not, we're going to use Debian's **Synaptic Package Manager**, respectively **dselect** on the command line.

The packages to install by order of appearance are:

- ▶ **python-imaging** (v1.1.4) : The Python Imaging Library (PIL) adds an image object to Python.
- ▶ **python2.3-ldap** (v2.0.4) : Python interface to the OpenLDAP client library.
- ▶ **zope2.7** (v2.7.3) : Open-source application server serving as foundation for Plone.
- ▶ **apache** (v1.3.33): High performance HTTP server.

There's a bit of configuration awaiting us once the above mentioned packages are done installing.

4.3.3 Configuring Zope

Before digging any deeper into Zope, we have to change a few things at system level. First of all, it is not recommended to run Zope as user root. Its good practice to run and configure Zope as a non-root user. Luckily, the package manager already created a user zope, but unfortunately without a shell. I personally prefer to have a real user zope to do all things Zope'ish. First of all, we have to create a home directory for the user zope. Assuming that you're logged in as root, create a `zope` directory in the `/home` directory:

```
mkdir /home/zope
```

Next we have to make user zope the owner of the newly created directory:

```
chown zope:zope /home/zope
```

Next step consists in assigning this directory to user zope and giving him permission to login. To do so, we have to edit the `passwd` file in the `/etc` directory.

***A Word of Warning:** Risking to sound patronising, I like to remind you that editing the `passwd` file is a delicate thing. Be careful not to erase or break it. I personally know somebody who accidendally encrypted his `passwd` file by exiting `vi` with an upper-case `X` instead of a lower-case `x`! Its generally a good idea to make a copy of the file before editing it.*

This being said, change the line somewhere at the end of the `passwd` file:

```
zope:x:###:###:/:var/lib/zope2.7/var:/bin/false
```

`###` standing for the user ID, respectively the group ID, into:

```
zope:x:###:###:Zope:/home/zope:/bin/bash
```

Safe the `passwd` file. Next we have to set a password for user zope.

```
passwd zope
```

You will be prompted to provide a password. In the context of this tutorial we're going to use `1xd2005`. As of now, user zope is able to login. Test it by changing to user zope:

```
su - zope
```

If your prompt changes to `zope@...` then its working correctly. Exit user zope by typing:

```
exit
```

Lets get back Zope. Debian's Package Manager uses the following paths to install Zope:

- ▶ `/usr/lib/zope2.7` : Main Zope package, including binaries, libraries and documentation.
- ▶ `/var/lib/zope2.7/instance` : Directory where Zope will store its instances.

4.3.3.1 Zope Instances

A Zope instance is a stand-alone zope application, with its own binaries, configuration and Products. This allows for different applications to exist in parallel on one machine. Every single instance can be stoped and started individually.

Before creating our Tux Industries instance, we have to make sure that we're allowed to do so as user `zope`. The instance directory that the package manager created is owned by user `root`. Assuming that you're still logged in as user `root`, change the directory's ownership to user `zope`:

```
chmod -R zope:zope /var/lib/zope2.7
```

Next step consists in creating a Tux Industries `zope` instance. To do so, please make sure you're logged as user `zope`:

```
su - zope
```

Zope provides a python script to create instances. As user `zope`, just run:

```
/usr/lib/zope2.7/bin/mkzopeinstance.py
```

You will be prompted to provide a directory. Use: `/var/lib/zope2.7/instance/tuxindustries`

Next you will be prompted for a user name. Use: `ploneadmin`

The last piece of information the script requires is a password for the new user. In the context of this tutorial we're going to use: `1xd2005`. Needless to say that you'll need a tougher password in a real production environment.

4.3.3.2 Instance configuration

Zope creates a configuration file inside each instance installed. All the configuration for that instance is located in that file. The configuration file of the instance is located in the `etc` directory. Edit the `zope.conf` configuration file.

```
vi /var/lib/zope2.7/instance/tuxindustries/etc/zope.conf
```

Changing ports Zope includes its own HTTP, FTP, WebDAV server. The ports these servers are listening to can be configured in the `zope.conf` file. Locate the following directives to discover which ports are used:

1. `<http-server>`
2. `<ftp-server>`

You'll see, that the `http` and `ftp` server listen to port **9673**, respectively **8021**. Port 9673 is a particularity of the Debian installation. Zope's `http` server usually listens on port 8080, but to avoid conflicts with proxy ports, the *Debian-ised* distribution listens on port 9673. In a real production environment you'd have to make sure that these ports are blocked at firewall level. We're not going to change anything here.

4.3.3.3 Debug Mode

By default in Zope debug mode is enabled. Note that Zope runs significantly slower in debug mode. To turn this off, located the `debug-mode` directive in the configuration file and replace it with:

```
debug-mode off
```

Make sure to un-comment the directive by removing the leading `#` (hash) sign.

Save the config file.

4 Setting up a web server using the Plone CMS

4.3.3.4 Starting an Instance

Starting and stopping an instance is done using the instance's `zopectl` script located in the instance's `bin` directory. To start the Tux Industries instance execute:

```
/var/lib/zope2.7/instance/tuxindustries/bin/zopectl start
```

The `zopectl` script launches the instances as a background task. If you're experiencing problems starting an instance and you require a higher degree of verbosity, you can start the instance using the `runzope` script. This blocking script dumps logging message to the terminal allowing you to see what's going wrong at which point. To check whether your instance is running, type

```
ps -ef|grep python
```

This command should at least return a list with two processes running as user `zope`. If this is the case, you're instance is running. In case you're working in a graphical environment, open a web browser and point it to: `localhost:9673`. The *Zope Quick Start* page should show up after a while.

4.3.4 Installing Plone

Now that we have a running Zope framework, we can proceed with installing Plone. As already mentioned earlier, Plone sits on top of Zope, in fact, Plone is nothing more than a aggregate of Zope products and will be installed as such.

As of now I would recommend using the graphical environment. Remember to login as user `zope`. Open a web browser and go to www.plone.org, the development and community site of Plone. Click on the *downloads* tab in the top bar. This brings you to the page with available downloads of Plone, from the nicely packaged installer downto source tarballs. Scroll down to the *Plone Core* section. The current version of Plone is 2.0.5. Download the Plone 2.0.5 tarball. Once the download is completed, unpack the downloaded file:

```
tar -zxvf Plone-2.0.5.tar.gz
```

You'll end up with a `Plone-2.0.5` directory. This directory contains the Zope products that Plone is made of. Installing Plone boils down to installing these products. Installing Zope products is relatively straight forward. Copy the (unpacked) product into the `Products` directory of the Zope instance. In our case, the command will be

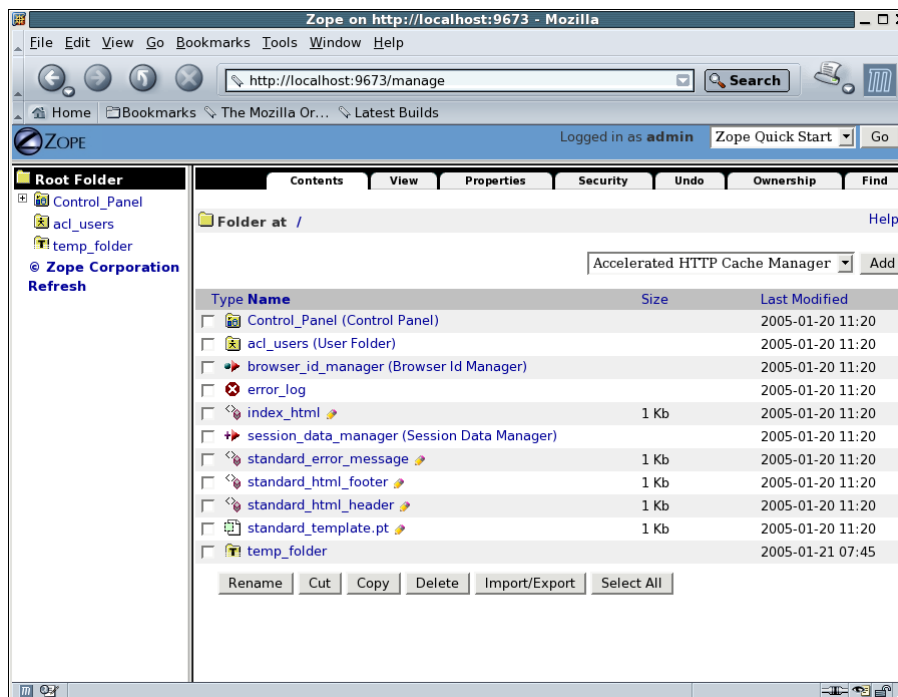
```
cp -r /home/zope/Plone-2.0.5/* /var/lib/zope2.7/instance/tuxindustries/Products/
```

Restart the Tux Industries Zope instance by executing

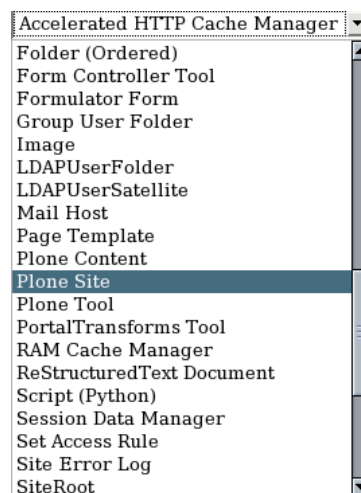
```
/var/lib/zope2.7/instance/tuxindustries/bin/zopectl restart
```

4.3.5 Creating a Plone site

Even though Plone is installed as of this point, we won't see a web site yet. We have to create a *Plone Site* object first. To do so, open the web browser and go to `localhost:9673`. The *Zope Quick Start* page shows again. Go down to the seventh bullet on this page and click on the *Zope Management Interface* link. You will be prompted to provide a username and a password. Use the username and password you've specified when creating the Zope instance, in our case `ploneadmin` and `1xd2005`.

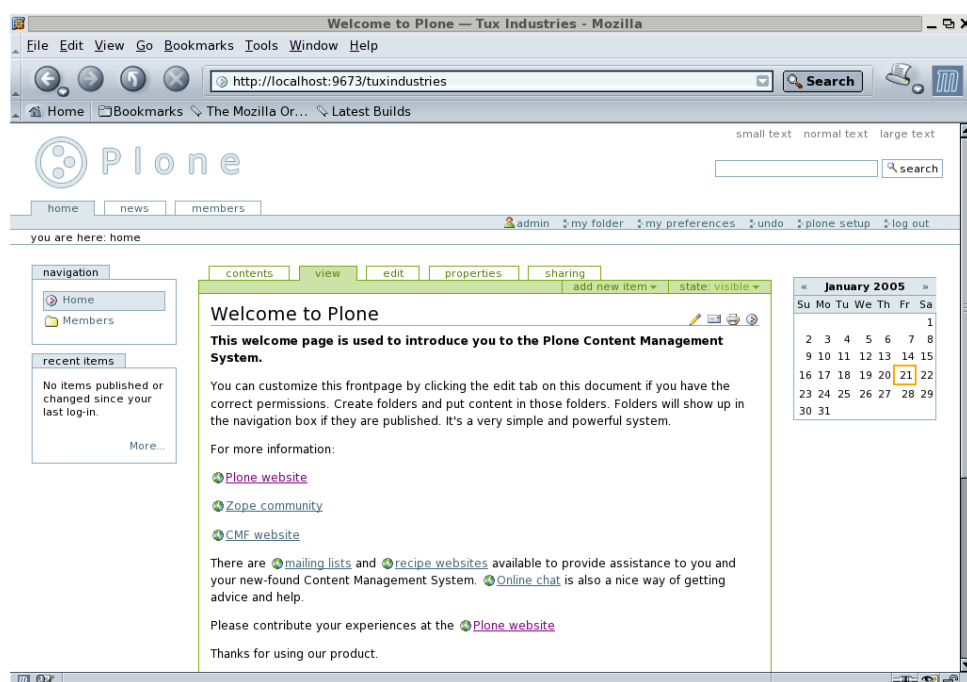


The Zope Management Interface is the core of our Zope Instance. Most low level configuration tasks will be performed via the Zope Management Interface, hereafter called the ZMI. To create a *Plone Site* object, click on the select box displaying *Accelerated HTTP Cache Manager*. Scroll through the popup list and select *Plone Site*.



Click on the Add button on the right. The *Add Plone Site* form will show. We have to provide an object *Id* and a *title*. The object Id will be used by Zope to uniquely identify the object whereas the title attribute will be the title of our web site. The object Id will very often be used to build the URL (Uniform Resource Locator) to that object. Specify **tuxindustries** for the Id and **Tux Industries** for the title. Leave the *Membership source* select box unchanged. Click on the *Add Plone Site* button to create the Plone Site object. That's it! You've just create your first Plone site. To have a look at it, point your web browser to **localhost:9673/tuxindustries**.

4 Setting up a web server using the Plone CMS



That's what an out of the box Plone site looks like. We're going to see a little bit later how to customise the look of our web site.

4.3.6 Adding and Editing content

In the context of this tutorial, we're going to limit ourselves to a very basic introduction of Plone's functions. I reckon that adding and editing content is a good candidate to illustrate how to work with Plone.

As a site member, you automatically have your own private folder where you can store content. Of course, you can add content to any folder that the site administrator, namely you, has given you the right to do so.

Each type of content you can add is distinct, and you can edit and view it in different ways. For this reason, Plone references each type of content differently; for example, you can add images, links, documents and so on. Out of the box, Plone provides the following *content types*:

- ▶ **Document:** This is an item that presents some static information to the user. This is the most common type of content added and most closely represents a typical Web page.
- ▶ **News Item:** This is a document that's to be shown under the news tab (for example, a press release)
- ▶ **Link:** This is a link to another item, which may be internal or external to another Web site.
- ▶ **Image:** This is an image, such as a .gif or a .jpeg file.
- ▶ **Event:** This is an upcoming meeting, conference, or other event.

- ▶ **Folder:** This is like a directory on a hard drive; its organising content in a hierarchic manner.
- ▶ **Topic:** This a grouping of other content. This is essentially a saved search criteria that you can reuse later. Only privileged site users can add topics.
- ▶ **File:** This is another piece of content such as a movie, sound clip, text file, spreadsheet, archive, or anything you'd like to upload.

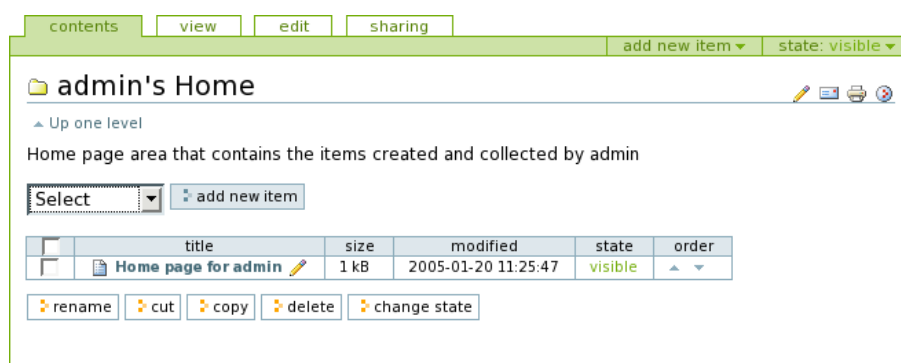
Using these content types, you can build a dynamic site through the browser, without doing any programming.

Actually, you have many ways of adding and editing content in a Plone site than just through a Web browser. Access via FTP, via WebDAV or via scripts is possible.

Rather than detailing how to add and edit all the different types of content available, we'll cover adding one type of content, a document, in detail. After adding a few of these documents you'll be familiar enough with the approach of adding and editing content.

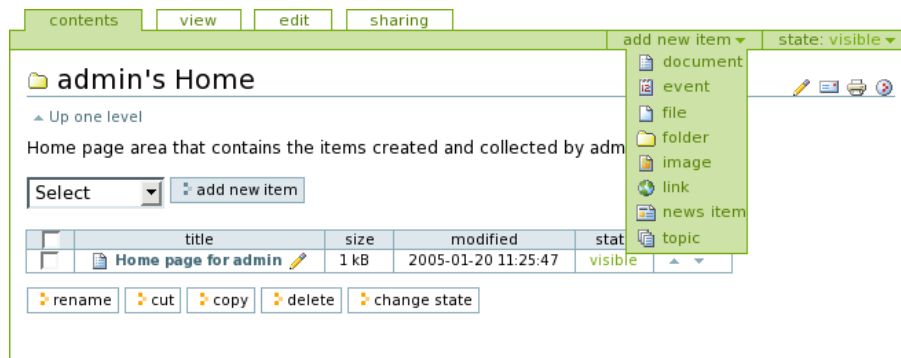
4.3.6.1 Adding a Document

You have two ways to add any piece of content using a Web browser. First, ensure you're logged in, because only logged-in users can add content. Second, select the *my folder* link from the top-right navigation bar. This will take you to your home folder. If you're allowed to add content to a folder, then the folder will show up with a green border around the top.

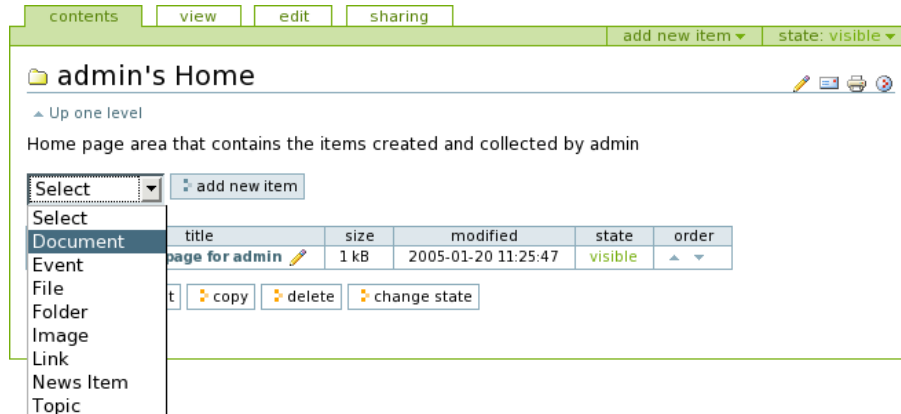


The green border contains the actions you can perform in the current location. In the previous figure you can see that the page shows the content of the folder, because that's the highlighted tab. Other tabs appear such as *view*, *edit* and *sharing* for more advanced options. In the top-right corner of the green border, you'll see an *add new item* drop-down menu. Click the down arrow of the menu to open a drop-down list of items to add:

4 Setting up a web server using the Plone CMS



To add a new document, select *document*. Alternatively, if you look in the body of the page, you can see another *add new item* drop-down box. Again, click the *add new item* button to open a list of items that can be added and then select the item you'd like to add:



4.3.6.2 Editing a Document

Once you've clicked to add a document, you'll be taken immediately to the edit page with a message telling you that the document has been created.

Document has been created.

Edit Document

Fill in the details of this document.

Document Details

Short Name
Short Name is part of the item's web address. For accessibility reasons, do not use spaces, upper case, underscores, or special characters.

Title ■

Description
A brief description of the item.

Body text

Now you can edit the document in your Web browser, using the form provided. If you look at the URL in your browser's address bar, you'll note a temporary short name for the object has been created for you, something like `Document.2005-01-21.14536`. The following is a list of the fields and their meaning:

- ▶ **Short Name:** The short name will become part of the document's URL, so keep the name short and descriptive, preferably without spaces and special characters. Use something like `ti_prod_report_2005`. If you don't provide a short name for yourself, Plone will use the temporary name it has created for you.
- ▶ **Title:** This is the title of the item, and it will be shown throughout the site, i.e. on top of the page, in the search interface, in the title of the browser, and so on). This field is a required field. Required fields are identified by a tiny red square after the field's label.
- ▶ **Description:** This is a short lead-in to the document, usually about 20 words to introduce the document and provide a teaser for the remainder of the document. This is useful for pages that show summaries of documents, such as search results and folder contents.
- ▶ **Body text:** This contains the body of the document. The format of the content is set using the *Format* field (described next).
- ▶ **Format:** You have three choices for the format of the body content: *Structured Text*, *HTML* and *Plain Text*.
- ▶ **Upload document:** If you do have a file on your computer, you can upload it instead of typing the content into the *Body Text* field. Use the Upload button at the bottom of the page to select a file. The content of an uploaded file will always replace previous content of the *Body Text* field.

4 Setting up a web server using the Plone CMS

Once you've finished editing your document, click the *Save* button to commit your changes.

4.3.6.3 Publishing a Document

When a document is created, it's given an initial *state*, called *visible*. By default, content isn't automatically published and available to the world; instead, others can view your content, but it doesn't show up in searches or the navigation tree. At any point in time, each item of content in your Plone site is in a particular state. By having items in different states, it's possible to apply different privileges to each state. These different states are part of the publishing *workflow*.

Workflow States The default workflow states of an out of the box Plone installation are:

- ▶ **Visible:** Content is created in the visible state. All site members can find visible content through the search function and can access it directly by visiting the object URL. Visible content doesn't show up in the navigation tree. Visible content is editable by their owners and site managers.
- ▶ **Pending:** Pending content includes items that have been submitted for publishing by site members. From a user point of view, pending content behaves like content in the visible state. The difference between the two states is that pending items are flagged for review; site reviewers are prompted to publish or reject pending items. Pending items are editable only by managers and reviewers.
- ▶ **Published:** Published items are visible to all site visitors. They appear in search results and the navigation tree. They may also appear in other areas specific to that type (news items, for example, also appear when you click the news tab). Published items are editable only by managers, but owners can retract them for editing (retracting reverts an item to the public draft state).
- ▶ **Private:** Items in the private state are visible and editable only by their owners and others with manager access to the folder in which they exist. They won't appear in search results or on the navigation tree for other users. Private items are editable only by their owner and managers.

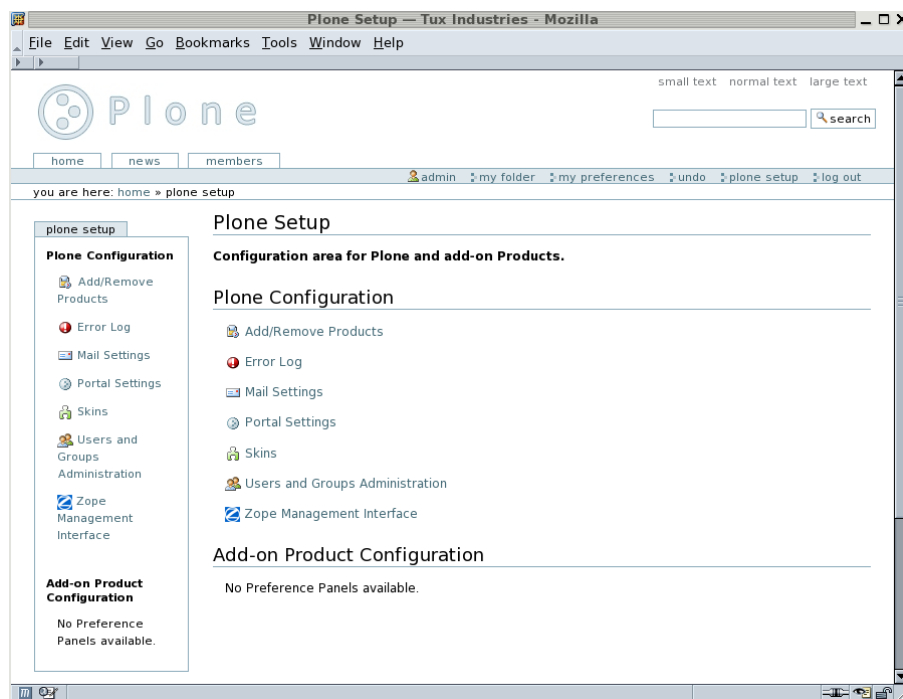
4.3.7 Administering Plone Sites

The first place site administrators should visit is the Plone control panel. This is the way to access some of the administration functions of a site, including name and description of your Plone site, user and group administration, and any error that occur withing your site.

The term control panel is common, so don't confuse it with the Zope Management Interface (ZMI) control panel that shows the low level ZMI options. The Plone control panel is an ongoing attempt to provide a more user friendly interface to the functions provided in the ZMI. To access the control panel, log into Plone. Click on the *plone setup* tool on the personalbar located at the top right of your browser window.



The Plone configuration area will show up.



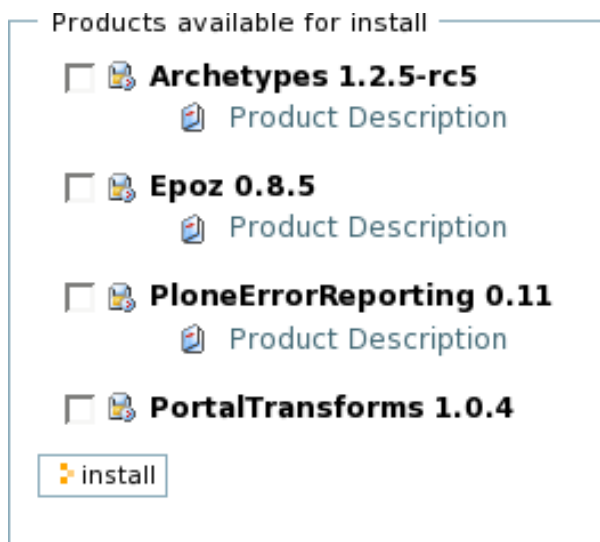
The following functions are available in the control panel:

- ▶ **Add/Remove Products:** Clicking this link allows you to automate the installation products.
- ▶ **Error Log:** Clicking this link accesses the log of errors that have occurred in your Plone site.
- ▶ **Mail Settings:** Allows you to alter the SMTP server Plone uses to send e-mail.
- ▶ **Portal Settings:** Allows you to alter portal settings, such as your site's name, e-mail addresses used etc.
- ▶ **Skins:** Allows you to set the current skin
- ▶ **Users and Groups Administration:** Allows to create and alter users and groups
- ▶ **Zope Management Interface:** Takes you to the ZMI for low level tasks.

In the context of this tutorial we're going to limit ourself to a minimum of administration.

4.3.7.1 Adding the WYSIWYG editor

Every Plone 2 installation ships with a what-you-see-is-what-you-get editor for editing document bodies. The Editor is called Epoz, and to install it, proceed as follows: In the Plone control panel, click on Add/Remove Products:



Tick the Epoz 0.8.5 checkbox and click the install button.



That's it! The WYSIWYG editor is installed and may be used.

Note: Adding and removing Plone products is actually a two step process. In the case of the Epoz editor, we only had to perform the last step. We're going to see a little bit later how products are installed from scratch.

4.3.7.2 Setting up a Mail Server

Plone will send e-mail using the *MailHost* object, which provides an interface to an SMTP server and allows the developer to write forms and tools that send e-mails. The *send-to-a-friend* function and mailing of a forgotten password use the settings configured here.

The default configuration is for a mail server on the localhost port 25. If the SMTP server is located elsewhere in the network, then you can change the mail server as well as the port it is listening to. The default settings are okay for our Tux Industries network.

4.3.7.3 Changing E-Mail Addresses

The title, description and e-mail addresses are stored in a Plone site as properties on an object inside Plone. You can access these fields by clicking the *Portal Settings* in the Plone control panel. Change the *portal from address* to `postmaster@tux-industries.com`

4.3.8 Extending Plone

There exists a wide variety of plug-in components for Plone. The Plone community itself provides an ever growing number of extensions, distributed via the *Collective* project on *sourceforge.net*. Installing Plone extensions is done in exactly the same way than installing Plone, download the extension, uncompress it and copy the resulting directory into your site's Zope instance **Product** directory. Don't forget to restart your site's Zope instance afterwards.

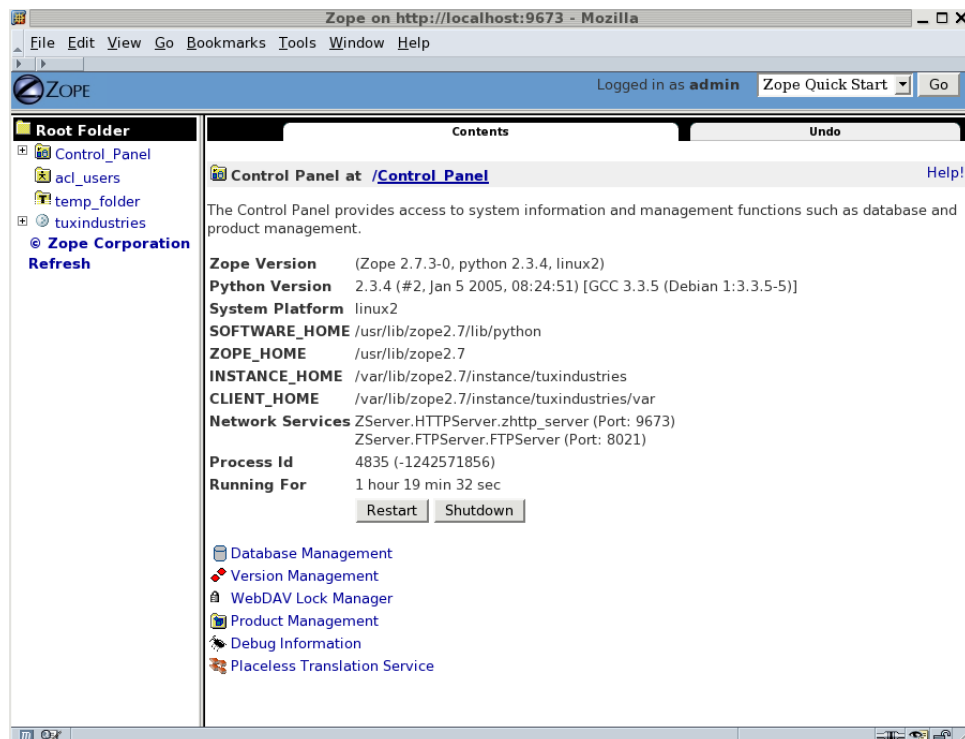
In the scope of this tutorial we're going to install two extensions. Lets start with the first one.

4.3.8.1 LDAPUserFolder

LDAPUserFolder is actually a low level Zope extension, delegating user authentication to a central LDAP server. Installing LDAPUserFolder requires the Python interface to the OpenLDAP client library `python2.3-ldap` (v2.0.4) to be installed.

Using your web browser, go and get the LDAPUserFolder extension from <http://www.dataflake.org/software> the file, untar it, and move it into the **Products** directory of your Zope instance. Restart your Zope instance. To verify whether installation was successful, open your Zope instance's Management Interface in your web browser by going to `localhost:9673/manage`.

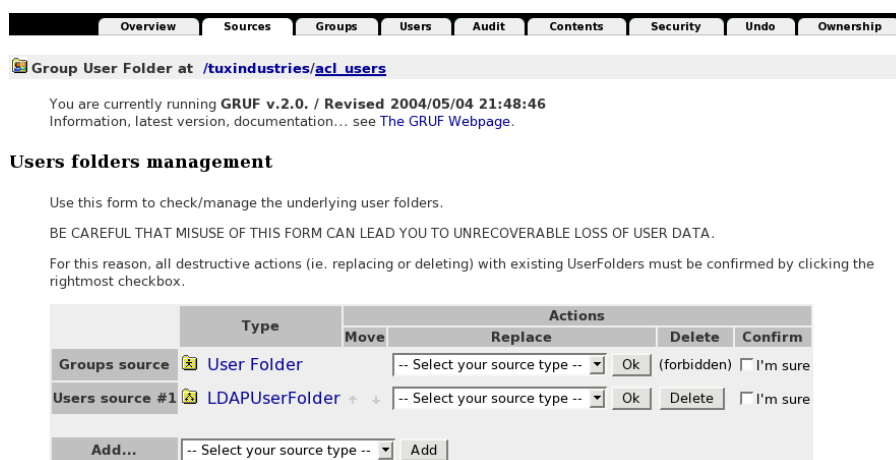
Click on the *Control Panel* item in the list.



4 Setting up a web server using the Plone CMS

The Control Panel provides access to system information and management functions such as database and product management. To see which Products are installed, click on the *Product Management* link at the bottom of the Control Panel. A list of all installed Products shows up. Go through the list and try to locate the LDAPUserFolder product. If you see it in the list, then the LDAPUserFolder extension has been successfully installed. If it shows with a “broken” icon, then the product was attempted to be registered in Zope, but an error occurred. Click the broken icon to get a traceback which will tell you the error and should give you a chance to fix it. Finally, if you’ve been unable to access the management interface after restarting, it could be that you have a more serious problem. In that case, you better try starting your Zope instance using `runzope` instead of `zopectl`.

Assuming installation was successful, we’re going to proceed to configure Plone so that it delegates user authentication to an LDAP server. To do so, go to the Plone *control panel* and click on the *Zope Management Interface* link. Locate the `acl_users` item and open it. An overview of users, groups and roles will show up. Click on the *Sources* tab on top of the page.



Overview Sources **Groups** Users Audit Contents Security Undo Ownership

Group User Folder at [/tuxindustries/acl_users](#)

You are currently running GRUF v.2.0. / Revised 2004/05/04 21:48:46
Information, latest version, documentation... see [The GRUF Webpage](#).

Users folders management

Use this form to check/manage the underlying user folders.

BE CAREFUL THAT MISUSE OF THIS FORM CAN LEAD YOU TO UNRECOVERABLE LOSS OF USER DATA.

For this reason, all destructive actions (ie. replacing or deleting) with existing UserFolders must be confirmed by clicking the rightmost checkbox.

| | Type | Actions | | | |
|-----------------|-------------------------------|---------|-------------------------------|--------|---|
| | | Move | Replace | Delete | Confirm |
| Groups source | User Folder | | -- Select your source type -- | Ok | (forbidden) <input type="checkbox"/> I'm sure |
| Users source #1 | LDAPUserFolder + | | -- Select your source type -- | Ok | Delete <input type="checkbox"/> I'm sure |
| Add... | -- Select your source type -- | | Add | | |

Scroll down to the *Users source #1* option. Then select LDAPUserFolder and check the *I'm sure* checkbox. This will create a new user folder replacing the existing one. Click on the *Ok* button. Next we have to add the settings that match the configuration of our existing LDAP server. Click on the *LDAPUserFolder* link and set the settings as follows:

LDAPUserFolder at [/tuxindustries/acl_users/Users/acl_users](#) [Help!](#)

Change the basic properties of your LDAPUserFolder on this form.

Title

Login Name Attribute

User ID Attribute

RDN Attribute

Users Base DN **Scope**

Group storage

Groups Base DN **Scope**

Manager DN **Password**

Manager DN Usage **Read-only**

User object classes

User password encryption

Default User Roles

4.4 Customising the Look and Feel Plone

Customising the look and feel of Plone is very waste field that would definitely blow up the frame of this tutorial. We're going to limit ourselves to a few basic customisations though.

When a document displays in Plone, the content of that document displays in the now-familiar Plone green and blue interface. A *skin* determines exactly how the document displays to the user, including the images and styles surrounding the content. A skin groups elements, wrapping that piece of content, and present them in a certain manner.

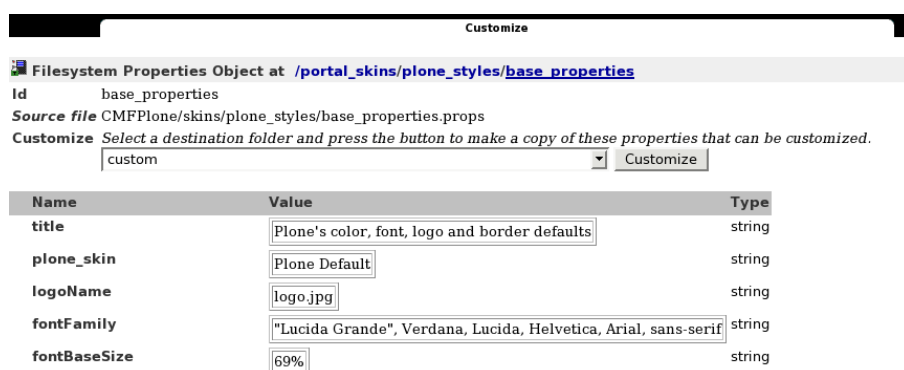
You use to *portal_skins* tool in Plone to define the skin behavior. To access the *portal_skins* tool, go to the ZMI and click *portal_skins*.

Plone Skins Tool at [/portal_skins](#) [Help!](#)

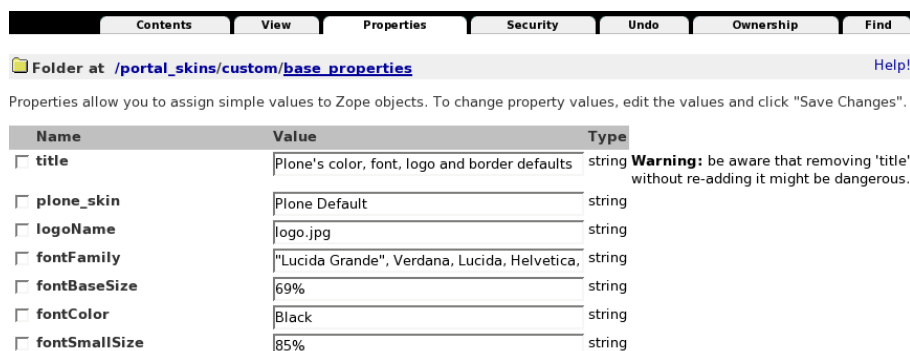
| Type | Name | Size | Last Modified |
|--------------------------|--------------------|------|------------------|
| <input type="checkbox"/> | cmf_legacy | | 2005-01-24 11:18 |
| <input type="checkbox"/> | custom | | 2005-01-24 10:46 |
| <input type="checkbox"/> | epoz | | 2005-01-24 11:18 |
| <input type="checkbox"/> | gruf | | 2005-01-24 11:18 |
| <input type="checkbox"/> | plone_3rdParty | | 2005-01-24 11:18 |
| <input type="checkbox"/> | plone_content | | 2005-01-24 11:18 |
| <input type="checkbox"/> | plone_ecmascript | | 2005-01-24 11:18 |
| <input type="checkbox"/> | plone_form_scripts | | 2005-01-24 11:18 |
| <input type="checkbox"/> | plone_forms | | 2005-01-24 11:18 |
| <input type="checkbox"/> | plone_images | | 2005-01-24 11:18 |
| <input type="checkbox"/> | plone_portlets | | 2005-01-24 11:18 |
| <input type="checkbox"/> | plone_prefs | | 2005-01-24 11:18 |
| <input type="checkbox"/> | plone_scripts | | 2005-01-24 11:18 |
| <input type="checkbox"/> | plone_styles | | 2005-01-24 11:18 |
| <input type="checkbox"/> | plone_tableless | | 2005-01-24 11:18 |
| <input type="checkbox"/> | plone_templates | | 2005-01-24 11:18 |
| <input type="checkbox"/> | plone_wysiwyg | | 2005-01-24 11:18 |

Locate the *plone_styles* directory and click on it. This directory contains all the properties and CSS stylesheets defining portal skins. Click on the *base_properties* item.

4 Setting up a web server using the Plone CMS



The *base_properties* property object defines Plone's color, font and border defaults. Customising skin related objects, i.e property objects, CSS stylesheets, templates, and so on, is done by opening the object and clicking the *customize* button. By default, Plone offers to store the customised copy of the file in the *portal_skins/custom* folder. Okay, click the customize button.



If you're looking at the path displayed on top following *Filesystem Properties Object at*, you'll notice that the customised version of the object effectively located at */portal_skins/custom*.

First of all we're going to customise a few colors. Locate the following properties, and change the values to the following:

| | |
|----------------------------|---------|
| globalBorderColor | #ba7dbb |
| globalBackgroundColor | #ba7dbb |
| globalFontColor | #ffffff |
| contentViewBorderColor | #7b0d81 |
| contentViewBackgroundColor | #f2d0f2 |
| contentViewFontColor | #7b0d81 |

Click on the *Save Changes* button to make the changes effective. If you open a second tab in your web browser and you have a look at `localhost:9673/tuxindustries` you'll notice that certain background and border colors have changed.

Next we're going to customise our web site logo. To do so, go to */portal_skins/plone_images* and locate the *logo.jpg* image. Open the image and click on the *customize* button. Next, go to */portal_skins/custom* and open the *logo.jpg* image again

Using the *Browse* button of the *File Data* field, go to `/home/guest` and select the `tuxindustries.gif` image. Click on the *Upload* button to upload the image to the server. Next, click on *Save Changes* to make your changes effective.

Last but not least, we're going to change the web site header so that the background matches the background color of our logo. Furthermore, we'd like to remove the margin around the logo so that our header looks slimer. To do so, we have to undertake a few changes in Plone's stylesheets.

Go to `/portal_skins/plone_styles` and open Plone's main stylesheet `plone.css`. Locate the style information for `#portal-logo` and `#portal-top`. Even though we could customise the `plone.css` stylesheet as such, it is recommended to customise the `ploneCustom.css` stylesheet. Any style information found in that file will automatically overwrite style information coming from `plone.css`. Go ahead and customise the `ploneCustom.css` stylesheet. Next, go to `/portal_skins/custom` folder and open customised copy of `ploneCustom.css`. In that file, locate the line saying:

```
/* DELETE THIS LINE AND PUT YOUR CUSTOM STUFF HERE */
```

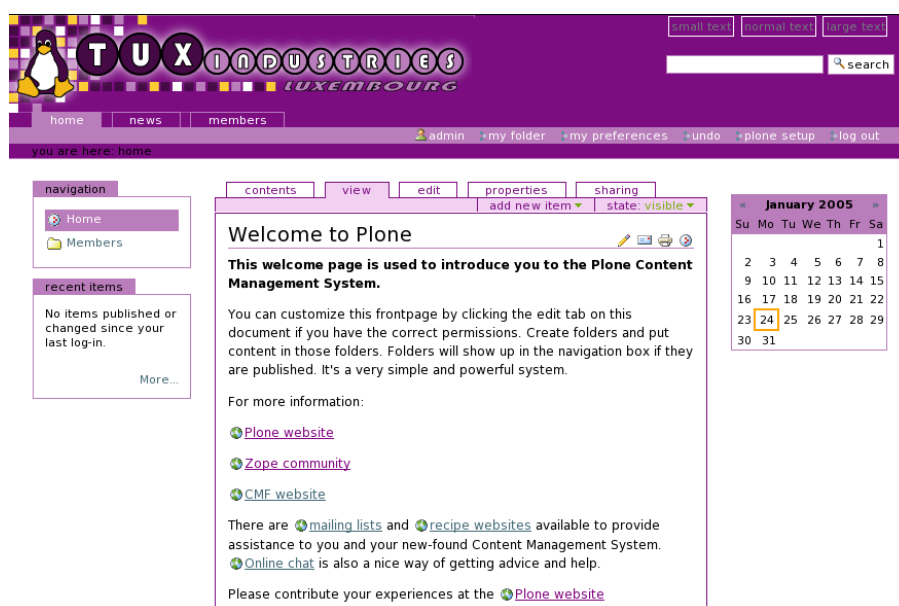
delete it and put the following as a replacement:

```
#portal-logo
{
    background: url(&dtml-portal_url;/&dtml-logoName;)    no-repeat;    border: 0;
    margin: 0;
    padding: 0;
}

#portal-top
{ /* Top section */
    margin: 0;
    padding: 0;
    background: #7b0d81;
}
```

Click on the *Save Changes* button to make your changes effective.

4 Setting up a web server using the Plone CMS



4.5 Proxying Plone

Although Plone uses Zope's underlying web server, ZServer works just fine - it's not a complete industry-strength web server that should be exposed to the world. Using a proxy server is a desirable approach with Plone. A proxy server sits between a client and a server and forwards requests from the client to the server. A proxy server should be transparent to the user.

Putting a web server, such as Apache, in front of Plone provides a whole host of useful services that ZServer does not have. For example, Apache can provide URL rewriting, SSL support, caching, content deflation, virtual hosting, request sanitation and so on. A favorite method for proxying to Plone is to use an HTTP proxy. Apache achieves this using the *mod_proxy* module.

4.5.0.2 Configuring Plone

Before configuring our Proxy Web server, we need to configure Plone. Since only one server can connect to a port at one, make sur that multiple Zope instances are running on distinct ports.

Another thing we want to do is is URL rewriting to change the URL of your site. Because the Plone Site object lives inside the Zope Object Database (ZODB) and has an id, it's accessed by putting that id in the URL, such as `http://localhost:9673/linuxdays`. In our case we would like this to be `http://www.tux-industries.com`. We can achieve this with the help of a *Virtual Host Monster*. Despite its name, the VHM object is a friendly and powerful object that will make our life so much easier. You only need one VHM object in the root of a Zope instance. The VHM object sits at the root of a Zope site and intercepts all incoming requests; it then alters the request so that the request goes to the part of Zope that you want. To create a VHM, in the ZMI, go to the root of your Zope instance and select *Virtual Host Monster* from the drop-down box. In the form that opens, enter an ID. Enter `vhm_tuxindustries` or anything you want, the actual ID doesn't matter.

4.5.0.3 Configuring Apache

Configuring apache is done as user root. Open a terminal and log in as user root.

The easiest way to rewrite a URL in Apache is to use the built-in rewrite and proxy modules. This means enabling Apache's `mod_rewrite` and `mod_proxy` modules.

The `mod_rewrite` and `mod_proxy` modules needs to be enabled first. You need to log in as root to perform the following task. Enabling apache module is done by executing:

```
/usr/sbin/apache-modconf apache enable mod_rewrite
```

```
/usr/sbin/apache-modconf apache enable libproxy
```

To configure Apache, we need to access Apache's `httpd.conf` configuration files. In the case of our Debian Sarge distribution, this file is located in `/etc/apache/`

In Apache, each site is usually contained within a virtual host directory. Locate the section in Apache's `httpd.conf` file where virtual hosts are defined by searching for `VirtualHost` directives. Add the following, making sure to replace `###` with values corresponding your machine's configuration:

```
<VirtualHost 10.35.1.###>
    ServerName dns.###.tux-industries.com
    ServerAdmin web@tux-industries.com
    DocumentRoot /var/www/tuxindustries/html
    RewriteEngine on
    RewriteCond %{HTTP:Authorization} ^(.*)
    RewriteRule ^/(.*)
    http://10.35.1.###:9673/VirtualHostBase/
    http/{%{HTTP_HOST}}:80/tuxindustries/VirtualHostRoot/$1 [P]
    ErrorLog /var/www/tuxindustries/logs/error.log
    CustomLog /var/www/tuxindustries/logs/access.log common
</VirtualHost>
```

Next we have to make sure that the directories we specified for `DocumentRoot`, `ErrorLog` and `CustomLog` do exists. When installing the Debian Apache package, the installer created a `/var/www` directory owned by `www-data` user. We're going to use that directory to store our static html files and log files. In general its good practice to separate logs and data off different web servers. In the scope of this tutorial we're going to have only one web server, but in a real production environment, your apache web server may well serve a multitude of them. So, first of all, we're going to create a directory for our website.

```
mkdir /var/www/tuxindustries
```

Next we're going to create the html and log directory inside our web server's directory:

```
mkdir /var/www/tuxindustries/html
```

```
mkdir /var/www/tuxindustries/logs
```

One last thing we have to do is to make user `www-data` the owner of the newly created subtree. To do so, execute:

4 *Setting up a web server using the Plone CMS*

```
chown -R www-data:www-data /var/www/tuxindustries
```

That's it! All that's left to do is to restart Apache: Do so, by executing the following:

```
/etc/init.d/apache stop
```

```
/etc/init.d/apache start
```

To check whether proxying is working correctly, open your web browser and go to `dns.###.tux-industries.com`. If your Plone website appears, then your apache proxy is up and running correctly.

5 Mail and Virus checking

by Patrick Harpes

This chapter will show how to setup a Postfix based mail server to use in a small enterprise network. To clarify the system, we will exemplify the mail server in a so called virtual enterprise “Tux-Industries”.

Tux-Industries is a company with different offices located in different countries. The top-level domain name is `tux-industries.com`, and each country has a sub-domain called `<country>.tux-industries.com`. The setup of the different domains and related MX-records are shown in chapter 3 - DNS and BIND and are not treated here.

Imaging now that you are responsible of the Tux-Industries office located in Luxembourg. So you have setup your DNS for the sub-domain `luxembourg.tux-industries.lu`. The next step will be to install and setup the mail server for your office. Additionally you want to handle 2 other sub-domains: `sales.luxembourg.tux-industries.lu` and `support.luxembourg.tux-industries.lu`. The employees mail-addresses have the following format: `firstname.name@luxembourg.tux-industries.lu`.

To achieve a higher security level, your incoming mails should be scanned by a virus scanner. Furthermore Spam mail should be blocked. This part will be handled in the next chapter.

5.1 About this chapter

This Postfix introduction is mainly a collection of existing documents. There are many parts copied from the official Postfix documentation. The original documentation can be found at www.postfix.org/documentation.html. I tried to make a selection of interesting parts out of these very complete documentation, so that to setup a basic Mail Transport Agent (MTA) you can start reading this synthesis.

5.2 Introduction to Postfix

5.2.1 What is Postfix?

The goal of the Postfix project is to implement a viable alternative to the UNIX Sendmail program. Specific goals, and the ways that Postfix attempts to achieve them are:

- ▶ Wide dissemination. Postfix must be adopted by lots of people in order to make a significant impact on Internet mail performance and security. Therefore the software is given away for free, with no strings attached to it.

5 Mail and Virus checking

- ▶ **Performance.** Postfix is up to three times as fast as its nearest competitor. A desktop PC running Postfix can receive and deliver a million different messages per day. Postfix uses web server tricks to reduce process creation overhead and uses other tricks to reduce file system overhead, without compromising reliability.
- ▶ **Compatibility.** Postfix is designed to be sendmail-compatible to make migration easy. Postfix supports `/var[/spool]/mail`, `/etc/aliases`, NIS, and `~/forward` files. However, Postfix also attempts to be easy to administer, and therefore it does not use `sendmail.cf`.
- ▶ **Safety and robustness.** Postfix is designed to behave rationally under stress. When the local system runs out of disk space or memory, the Postfix software backs off, instead of making the problem worse. By design, no Postfix program keeps growing as the number of messages etc. increases. Postfix is designed to stay in control.
- ▶ **Flexibility.** Postfix is built from over a dozen little programs that each perform only one specific task: receive a message via SMTP, deliver a message via SMTP, deliver a message locally, rewrite an address, and so on. Sites with specific requirements can replace one or more little programs by alternative versions. And it is easy to disable functionality, too: firewalls and client workstations don't need local delivery at all.
- ▶ **Security.** Postfix uses multiple layers of defence to protect the local system against intruders. Almost every Postfix daemon can run in a chroot jail with fixed low privileges. There is no direct path from the network to the security-sensitive local delivery programs - an intruder has to break through several other programs first. Postfix does not even trust the contents of its own queue files, or the contents of its own IPC messages. Postfix filters sender-provided information before exporting it via environment variables. Last but not least, no Postfix program is set-uid.

5.2.2 Where to get Postfix?

Postfix can be downloaded from www.postfix.org. If you are using a Linux distribution precompiled Postfix packages can preferably be installed. Postfix packages exist for nearly all Linux distributions.

5.2.3 Why Postfix

The Postfix software package provides higher performance, better security, and simpler configuration than other standard compliant Mail Transport Agents (MTA). Postfix uses a modular design rather than the monolithic sendmail design. Due to this modularity, Postfix is a very robust and efficient MTA. Furthermore the configuration of Postfix is much easier than a sendmail configuration.

The combination and integration of Postfix, virus scanner and SPAM blockers is simple.

5.3 Architecture of the Postfix system

To handle the complexity of the MTA, Postfix has been divided into different modules. Each module is responsible for a specific task. Once this task has been done, the module gives the mail to another module. The figure shows also the configuration files for the different modules.

Figure 5.1: Postfix Architecture

When a message enters the Postfix mail system, the first stop on the inside is the incoming queue.

Network mail enters Postfix via the `smtpd` server. This server removes the SMTP protocol encapsulation, enforces some sanity checks to protect Postfix, and gives the sender, recipients and message content to the cleanup module. The `smtpd` module can be configured to block unwanted mail. This can be done using the `access` table.

Local submissions are received with the Postfix `sendmail` compatibility command, and are queued in the maildrop queue by the privileged `postdrop` command. This arrangement even works while the Postfix mail system is not running. The local pickup module picks up local submissions, enforces some sanity checks to protect Postfix, and gives the sender, recipients and message content to the `cleanup` module.

Mail from internal sources is given directly to the cleanup module.

The `cleanup` module implements the final processing stage before mail is queued. It adds missing `From:` and other message headers to the message. Furthermore, it gives the mail to the `rewrite` module, which rewrites addresses to the standard "user@fully.qualified.domain". Postfix currently does not implement a rewriting language, but a lot can be done via table lookups and, if need be, regular expressions. Optionally, the `cleanup` module can be configured to do light-weight content inspection with regular expressions. The `cleanup` module places the result as a single file into the incoming queue, and notifies the queue manager of the arrival of new mail.

The queue manager (the `qmgr` server process in the figure) is the heart of Postfix mail delivery. It contacts the `smtp`, `local`, `virtual`, `pipe`, `discard` or `error` delivery agents, and sends a delivery request for one or more recipient addresses. The `discard` and `error` delivery agents are special: they discard or bounce all mail, they are not shown in the figure above.

The queue manager maintains a small active queue with the messages that it has opened for delivery. The active queue acts as a limited window on potentially large incoming or deferred queues. The limited active queue prevents the queue manager from running out of memory under heavy load.

The queue manager maintains a separate deferred queue for mail that cannot be delivered, so that a large mail backlog will not slow down normal queue accesses.

The `trivial-rewrite` server resolves each recipient address according to its local and remote address class. Additional routing information can be specified with the optional transport table. The `trivial-rewrite` server optionally queries the relocated table for recipients whose address has changed; mail for such recipients is returned to the sender with an explanation.

The `smtp` client looks up a list of mail exchangers for the destination host, sorts the list by preference, and tries each server in turn until it finds a server that responds. It then encapsulates the sender, recipient and message content as required by the SMTP protocol; this includes conversion of 8-bit MIME to 7-bit encoding.

The local delivery agent understands UNIX-style mailboxes, `qmail`-compatible `maildir` files, Sendmail-style system-wide aliases databases, and Sendmail-style per-user `.forward` files. Multi-

5 Mail and Virus checking

ple local delivery agents can be run in parallel, but parallel delivery to the same user is usually limited.

The local delivery agent has hooks for alternative forms of local delivery: you can configure it to deliver to mailbox files in user home directories, you can configure it to delegate mailbox delivery to an external command such as procmail, or you can delegate delivery to a different Postfix delivery agent.

The virtual delivery agent is a bare-bones delivery agent that delivers to UNIX-style mailbox or gmail-style maildir files only. This delivery agent can deliver mail for multiple domains, which makes it especially suitable for hosting lots of small domains on a single machine.

The pipe mailer is the outbound interface to other mail processing systems (the Postfix sendmail command being the inbound interface). The interface is UNIX compatible: it provides information on the command line and on the standard input stream.

5.4 Installing Postfix

5.4.1 Installation

During the Linuxdays Server tutorial, Postfix will be installed on a Debian Sarge system. The quickest way to install it is to use the Debian package management system. As root enter the following command to auto install postfix:

```
# apt-get install postfix
```

You need to have a network connection or Debian CD's to use the prior command. The package management of Debian automatically checks for dependencies and installs the missing packages on which Postfix relies. exim is installed as default MTA on a Debian system. The package management disables this MTA and configures Postfix as the new default MTA.

Using the Debian package management is easy, but has some disadvantages. You cannot specify all the options and features that Postfix offers. You have to live with the Postfix that Debian people compiled for you. Another problem is the processor compatibility. Debian packages for the Intel platform are compiled towards the i386 instruction set. If you have now newer processors, it may be better to recompile Postfix for your processor to take benefit of new processor features and performance advantages.

For a small MTA, the package is sufficient, you don't need to recompile it. This tutorial we will use the precompiled Debian package.

5.4.2 Where are the files?

After installing the package, you want to know where Debian stores the different Postfix files. There are 2 groups of files : binaries and configuration files.

The postfix binaries:

| Path | Tool | Function |
|-----------|------------|--|
| /usr/sbin | postfix | The postfix command controls the operation of the mail system. It is the interface for starting, stopping, and restarting the mail system, as well as for some other administrative operations. This command is reserved to the super-user. |
| /usr/sbin | postalias | The postalias command maintains Postfix aliases type databases. This is the program that does the work for the newaliases command. |
| /usr/sbin | postcat | The postcat command displays the contents of Postfix queue files. |
| /usr/sbin | postconf | The postconf command displays or updates Postfix main.cf parameters and displays system dependent information about the supported file locking methods, and the supported types of lookup tables. |
| /usr/sbin | postdrop | The postdrop command is can be used to take mails from command line and put them into the queues |
| /usr/sbin | postkick | With the postkick command you can send commands to the postfix modules |
| /usr/sbin | postlock | The postlock command provides Postfix-compatible mailbox locking for use in, for example, shell scripts. |
| /usr/sbin | postlog | The postlog command provides Postfix-compatible logging for shell scripts. |
| /usr/sbin | postmap | The postmap command maintains Postfix lookup tables such as canonical, virtual and others. It is a cousin of the UNIX makemap command. |
| /usr/sbin | postqueue | The postqueue command is the privileged command that is run by Postfix sendmail and mailq in order to flush or list the mail queue. |
| /usr/sbin | postsuper | The postsuper command maintains the Postfix queue. It removes old temporary files, and moves queue files into the right directory after a change in the hashing depth of queue directories. This command is run at mail system startup time and when Postfix is restarted. |
| /usr/bin | mailq | The Postfix mailq command lists the content of the mail queues. It is a replacement of sendmail's mailq |
| /usr/bin | newaliases | The Postfix newaliases command is the same as postalias and replace sendmail's newaliases. |
| /usr/sbin | sendmail | The Postfix sendmail command is a sendmail compatible application and can be used for local applications. |

The postfix configuration files:

| Path | Filename | Function |
|--------------|-----------|--|
| /etc/postfix | main.cf | Contains operational parameters used by the Postfix modules when processing messages |
| /etc/postfix | master.cf | Contains parameters used by the Postfix master program when running core programs |
| /etc/postfix | access | Maps remote SMTP hosts to an accept/deny table for security |
| /etc | aliases | Maps alternative recipients to local mailboxes |
| /etc/postfix | canonical | Maps alternative mailbox names to real mailboxes for message headers |
| /etc/postfix | relocated | Maps an old user mailbox name to a new mailbox name |
| /etc/postfix | transport | Maps domain names to delivery methods for remote host connectivity and delivery |
| /etc/postfix | virtual | Maps recipients and domains to local mailboxes for delivery |

5.5 Basic configuration

The first step after the installation is to setup a basic configuration to receive and to send mails for the Tux-Industries domain. Remember that we want to setup a mail server for the luxembourg.tux-industries.com domain. We assume that DNS has the corresponding MX records. Our mail-server will be a member of our private network 192.168.1.0/24, and has the name mailserver.luxembourg.tux-industries.com with the corresponding IP-Address 192.168.1.5.

Postfix has several hundred configuration parameters that are controlled via the main.cf file. Fortunately, all parameters have sensible default values. In many cases, you need to configure only two or three parameters before you can start to play with the mail system.

The first parameters of interest specify the machine's identity and role in the network.

- ▶ What domain name to use in outbound mail
- ▶ What domains to receive mail for
- ▶ What clients to relay mail from
- ▶ What destinations to relay mail to
- ▶ What delivery method: direct or indirect

The default values for many other configuration parameters are derived from just these.

The next parameter of interest controls the amount of mail sent to the local postmaster:

- ▶ What trouble to report to the postmaster
- ▶ What you need to know about Postfix logging

If your machine has unusual security requirements you may want to run Postfix daemon processes inside a chroot environment.

- ▶ Running Postfix daemon processes chrooted

If you run Postfix on a virtual network interface, or if your machine runs other mailers on virtual interfaces, you'll have to look at the other parameters listed here as well:

- ▶ My own hostname
- ▶ My own domain name
- ▶ My own network addresses

5.5.1 Postfix configuration files

By default, Postfix configuration files are in `/etc/postfix`. The two most important files are `main.cf` and `master.cf`; these files must be owned by root. Giving someone else write permission to `main.cf` or `master.cf` (or to their parent directories) means giving root privileges to that person.

In `/etc/postfix/main.cf` you will have to set up a minimal number of configuration parameters. Postfix configuration parameters resemble shell variables, with two important differences: the first one is that Postfix does not know about quotes like the UNIX shell does.

You specify a configuration parameter as:

```
/etc/postfix/main.cf: parameter = value
```

and you use it by putting a "\$" character in front of its name:

```
/etc/postfix/main.cf: other_parameter = $parameter
```

You can use `$parameter` before it is given a value (that is the second main difference with UNIX shell variables). The Postfix configuration language uses lazy evaluation, and does not look at a parameter value until it is needed at runtime.

Postfix uses database files for access control, address rewriting and other purposes. Here is a common example of how Postfix invokes a database:

```
/etc/postfix/main.cf: virtual_alias_maps = hash:/etc/postfix/virtual
```

Whenever you make a change to the `main.cf` or `master.cf` file, execute the following command as root in order to refresh a running mail system:

```
# postfix reload
```

5.5.2 My own hostname

The `myhostname` parameter specifies the fully-qualified domain name of the machine running the Postfix system. `$myhostname` appears as the default value in many other Postfix configuration parameters.

By default, `myhostname` is set to the local machine name. If your local machine name is not in fully-qualified domain name form, or if you run Postfix on a virtual interface, you will have to specify the fully-qualified domain name that the mail system should use.

Alternatively, if you specify `mydomain` in `main.cf`, then Postfix will use its value to generate a fully-qualified default value for the `myhostname` parameter.

Example

```
/etc/postfix/main.cf:  
myhostname = mailserver.luxembourg.tux-industries.com
```

5.5.3 My own domain name

The `mydomain` parameter specifies the parent domain of `$myhostname`. By default, it is derived from `$myhostname` by stripping off the first part (unless the result would be a top-level domain).

Conversely, if you specify `mydomain` in `main.cf`, then Postfix will use its value to generate a fully-qualified default value for the `myhostname` parameter.

Example:

```
/etc/postfix/main.cf:  
mydomain = luxemburg.tux-industries.com
```

5.5.4 My own network addresses

The `inet_interfaces` parameter specifies all network interface addresses that the Postfix system should listen on; mail addressed to "user@[network address]" will be delivered locally, as if it is addressed to a domain listed in `$mydestination`.

You can override the `inet_interfaces` setting in the Postfix `master.cf` file by prepending an IP address to a server name.

The default is to listen on all active interfaces. If you run mailers on virtual interfaces, you will have to specify what interfaces to listen on.

IMPORTANT: If you run MTA's on virtual interfaces you must specify explicit `inet_interfaces` values for the MTA that receives mail for the machine itself: this MTA should never listen on the virtual interfaces or you would have a mailer loop when a virtual MTA is down.

Example:

```
/etc/postfix/main.cf:  
inet_interfaces = $myhostname localhost
```

Note: you need to stop and start Postfix after changing this parameter.

5.5.5 What domain name to use in outbound mail

The `myorigin` parameter specifies the domain that appears in mail that is posted on this machine. The default is to use the local machine name, `$myhostname`, which defaults to the name of the machine. Unless you are running a really small site, you probably want to change that into `$mydomain`, which defaults to the parent domain of the machine name.

For the sake of consistency between sender and recipient addresses, `myorigin` also specifies the domain name that is appended to an unqualified recipient address.

Examples :

```
/etc/postfix/main.cf:
myorigin = $myhostname (default: send mail as "user@$myhostname")
myorigin = $mydomain (probably desirable: "user@$mydomain")
```

For our example we prefer `user@luxembourg.tux-industries.com`:

```
myorigin = $mydomain
```

5.5.6 What domains to receive mail for

The `mydestination` parameter specifies what domains this machine will deliver locally, instead of forwarding to another machine. The default is to receive mail for the machine itself. To handle additional domains or sub-domains, you have to specify this using the virtual table. This will be explained later in this document.

IMPORTANT: If your machine is a mail server for its entire domain, you must list `$mydomain` as well.

Example 1: default setting.

```
/etc/postfix/main.cf:
mydestination = $myhostname localhost.$mydomain localhost $mydomain
```

Example 2: `luxembourg.tux-industries.com` example

```
/etc/postfix/main.cf:
myhostname=mailserver.luxembourg.tux-industries.com
mydomain=luxembourg.tux-industries.com
mydestination = $myhostname localhost.$mydomain localhost $mydomain
```

Caution: in order to avoid mail delivery loops, you must list all hostnames of the machine, including `$myhostname`, and `localhost.$mydomain`.

5.5.7 What clients to relay mail from

By default, Postfix will forward mail from clients in authorised network blocks to any destination. Authorised networks are defined with the `mynetworks` configuration parameter. The default is to authorise all clients in the IP subnetworks that the local machine is attached to.

IMPORTANT: If your machine is connected to a wide area network then your default `mynetworks` setting may be too friendly.

The `mynetworks_style` parameter can take one of the three parameters: `host`, `subnet` (default), `class`.

Specify `"mynetworks_style = host"` when Postfix should forward mail from only the local machine.

Specify `"mynetworks_style = subnet"` (the default) when Postfix should forward mail from SMTP clients in the same IP subnetworks as the local machine. On Linux, this works correctly only with interfaces specified with the `ifconfig` command.

Specify `"mynetworks_style = class"` when Postfix should forward mail from SMTP clients in the same IP class A/B/C networks as the local machine. Don't do this with a dial-up site - it would cause Postfix to "trust" your entire provider's network. Instead, specify an explicit `mynetworks` list by hand, as described below.

Alternatively, you can specify the `mynetworks` list by hand, in which case Postfix ignores the `mynetworks_style` setting. To specify the list of trusted networks by hand, specify network blocks in CIDR (network/mask) notation, for example:

```
/etc/postfix/main.cf:  
mynetworks = 168.100.189.0/28, 127.0.0.0/8
```

For our example we only allow to relay mail from our subnet:

You have 2 possibilities to specify this :

1. We specify to use `subnet` in `mynetwork_style`:

```
/etc/postfix/main.cf:  
mynetwork_style = subnet
```

2. We ignore `mynetwork_style` and we specify directly our subnet:

```
/etc/postfix/main.cf:  
mynetworks = 127.0.0.0/8 192.168.1.0/24
```

5.5.8 What destinations to relay mail to

By default, Postfix will forward mail from strangers (clients outside authorised networks) to authorised remote destinations only. Authorised remote destinations are defined with the `relay_domains` configuration parameter. The default is to authorise all domains (and subdomains) of the domains listed with the `mydestination` parameter.

The `relay_domains` parameter restricts what destinations this system will relay mail to. By default, Postfix relays mail

- ▶ from "trusted" clients (IP address matches `$mynetworks`) to any destination
- ▶ from "untrusted" clients to destinations that match `$relay_domains` subdomains thereof, except addresses with sender-specified routing.

The default `relay_domains` value is `$mydestination`.

In addition to the above, the Postfix SMTP server by default accepts mail that Postfix is final destination for:

- ▶ destinations that match `$inet_interfaces` or `$proxy_interfaces`,
- ▶ destinations that match `$mydestination`
- ▶ destinations that match `$virtual_alias_domains`,
- ▶ destinations that match `$virtual_mailbox_domains`.

These destinations do not need to be listed in `$relay_domains`.

For the Tux-Industries example we should disable this parameter as we only want to accept mails for our domain, resp. sub-domain.

5.5.9 What delivery method: direct or indirect

By default, Postfix tries to deliver mail directly to the Internet. Depending on your local conditions this may not be possible or desirable. For example, your system may be turned off outside office hours, it may be behind a firewall, or it may be connected via a provider who does not allow direct mail to the Internet. In those cases you need to configure Postfix to deliver mail indirectly via a relay host.

Example:

In our example we want to deliver the mails directly to the Internet, so we specify the following:

```
relayhost =
```

To make things more complicate, we could imagine to relay our mails first to our top level mail-server (`mailserver.tux-industries.lu`) which sends the mails to the Internet. In this case we have to specify the parameter:

```
relayhost = [mailserver.tux-industries.lu]
```

5.5.10 What trouble to report to the postmaster

You should set up a postmaster alias in the aliases table that directs mail to a human person. The postmaster address is required to exist, so that people can report mail delivery problems. While you're updating the aliases table, be sure to direct mail for the super-user to a human person too.

```
/etc/aliases:  
postmaster: root, harpes
```

Execute the command "newaliases" after changing the aliases file.

The Postfix system reports problems to the postmaster alias. You may not be interested in all types of trouble reports, so this reporting mechanism is configurable. The default is to report only serious problems (resource, software) to postmaster:

Default setting:

```
/etc/postfix/main.cf:  
notify_classes = resource, software
```

5.5.11 What you need to know about Postfix logging

Postfix daemon processes run in the background, and log problems and normal activity to the syslog daemon. The syslogd process sorts events by class and severity, and appends them to log-files. The logging classes, levels and log-file names are usually specified in `/etc/syslog.conf`. At the very least you need something like:

```
/etc/syslog.conf:  
mail.err /dev/console  
mail.debug /var/log/maillog
```

After changing the syslog.conf file, send a "HUP" signal to the syslogd process.

5.5.12 Running Postfix daemon processes chrooted

Postfix daemon processes can be configured (via the master.cf file) to run in a chroot jail. The processes run at a fixed low privilege and with file system access limited to the Postfix queue directories (`/var/spool/postfix`). This provides a significant barrier against intrusion. The barrier is not impenetrable (chroot limits file system access only), but every little bit helps.

With the exception of Postfix daemons that deliver mail locally and/or that execute non-Postfix commands, every Postfix daemon can run chrooted.

Sites with high security requirements should consider to chroot all daemons that talk to the network: the smtp and smtpd processes, and perhaps also the lmtp client.

The default `/etc/postfix/master.cf` file specifies that no Postfix daemon runs chrooted. In order to enable chroot operation, edit the file `/etc/postfix/master.cf`, and follow instructions in the file. When you're finished, execute "postfix reload" to make the change effective.

During this Tux-Industries tutorial, we will not handle the chroot jail, but for mail-servers that resides directly on the Internet it is highly recommended to run postfix in a chroot environment.

5.6 The basic configuration files for our example

/etc/postfix/main.cf

```
myhostname = mailserver.luxembourg.tux-industries.lu
mydomain = luxembourg.tux-industries.lu
inet_interfaces = $myhostname localhost
myorigin = $mydomain
mydestination = $myhostname localhost.$mydomain localhost $mydomain
mynetwork_style = subnet
# relay_domain =
relayhost =
notify_classes = resource, software
# Some installation specific parameters
command_directory = /usr/sbin
daemon_directory = /usr/lib/postfix
mail_owner = postfix
alias_maps = hash:/etc/aliases
smtp_banner = $myhostname ESMTP $mail_name
```

/etc/aliases

```
postmaster: root, harpes
patrick.harpes: harpes
test.testperson: test
...
```

Using the above configuration files, a Postfix system will be able to handle mails from and to luxembourg.tux-industries.com domain.

5.7 Additional configuration

5.7.1 The virtual table

For our domain (luxembourg.tux-industries.com) we want to create two new sub domains: sales.luxembourg.tux-industries.com and support.luxembourg.tux-industries.com. After configuring the name-servers to handle these new domains, Postfix has to accept mails for this domains. Postfix can handle additional domains by using the virtual table. The virtual table can be seen as an extension to the old sendmail alias table. The difference of these two table is that the alias table can only rewrite local mail addresses. The virtual table at the other side can be used to rewrite whole domains.

There are four different record types in the virtual table. Each one defines a different type of a virtual domain:

| | |
|--|--|
| <code>domain.name anything</code> | Allows Postfix to receive messages for <code>domain.name</code> and place them in the message queue. |
| <code>user@domain address1, address2, ...</code> | Receives messages for <code>user@domain</code> and forwards them to addresses listed. This redirection has the highest priority. |
| <code>user address1, address2, ...</code> | Receives messages for <code>user</code> on the local host and forwards them to the addresses listed. |
| <code>@domain address1, address2, ...</code> | Receives messages for all users in <code>domain</code> and forwards them to the addresses listed. |

Example

```
test.lu          anything
info@linuxdays.lu  patrick.harpes@tudor.lu, marc.seil@tudor.lu
@support.linux.lu  eric.dondelinger@tudor.lu
```

This configuration defines the following:

All mails from `test.lu` should be accepted. Incoming mails for `info@linuxdays.lu` should be directed to `patrick.harpes@tudor.lu` and `marc.seil@tudor.lu`. Finally every message for the `support.linux.lu` domain should be forwarded to `eric.dondelinger@tudor.lu`.

After editing the virtual table, you have to tell Postfix to use this table. In the `main.cf` configuration file you have to specify the following line:

```
virtual_maps = hash:/etc/postfix/virtual
```

This line says that postfix should look in `/etc/postfix` to find the virtual table. The `hash` keyword tells Postfix that this table uses the DBM-hash format. Because the file we created is in normal ASCII format, we have to convert the file into the DBM-hash format. This can be done with the `postmap` command:

```
# cd /etc/postfix
# postmap virtual
```

5.7.2 The aliases table

You can use the aliases table to map “fake”, mail addresses to actual system users. This is directly compatible with the sendmail program’s system of aliases table.

The aliases table differs from most of Postfix’s other lookup tables in that it has a slightly different form:

```
pattern: result
```

The colon is added to that pattern to make the Postfix alias file compatible with sendmail alias file. This simplifies converting sendmail mail servers to Postfix.

Example

```
postmaster: root
patrick.harpes: pharpes
```

This configuration defines the following:

Mails to `postmaster@localdomain.com` are forwarded to the `root` account on this local machine. Mails for `patrick.harpes@localdomain` are forwarded to the `pharpes` user account on this local machine.

After editing the aliases table, you have to tell Postfix to use this table. In the `main.cf` configuration file you have to specify the following line:

```
alias_maps = hash:/etc/aliases
```

This line says that postfix should look in `/etc` to find the aliases table. The `hash` keyword tells Postfix that this table uses the DBM-hash format. As the file we created is in normal ASCII format, it has to be converted into the DBM-hash format. This can be done with the `postalias` command:

```
# cd /etc/
# postalias aliases
```

5.7.3 Integrating Postfix with OpenLDAP

To show how to integrate Postfix with OpenLDAP, we want to replace the `/etc/aliases` file by a LDAP query. Because in this tutorial we have a single sign on system implemented using a LDAP system, it's also a good idea to take the "alias" information out of this system. The Postfix configuration for LDAP is simple. The first step to do is to install the Postfix-LDAP extension:

```
# apt-get install postfix-ldap
```

The second step consists in telling Postfix to use the LDAP database for the alias file instead of the hash function. In `main.cf` you should add the following lines:

```
alias_maps = ldap:/etc/postfix/aliases.ldap
```

This line points to the LDAP configuration file for the aliases: `/etc/postfix/aliases.ldap`:

```
server_host = localhost
server_port = 389
search_base = dc=luxembourg,dc=tux-industries,dc=com
ldap_scope = sub
query_filter = (mailLocalAddress=%s)
result_attribute = mailRoutingAddress
```

The LDAP configuration files defines which LDAP server to query on which port. Furthermore the LDAP search base as well the attributes are defined here.

5.8 Virus scanner

To achieve a more secure mail server, we want to implement a virus scanner to scan for viruses on incoming mails.

5.8.1 AMaViS - A Mail Virus Scanner

AMaViS is a script that interfaces a mail transport agent (MTA) with virus scanners. It is a high-performance interface between mailer (MTA) and content checkers: virus scanners, and/or SpamAssassin. It communicates to MTA via (E)SMTP or LMTP, or by using wrapper software. There are different versions of AMaViS out: amavis, amavis-perl, amavisd, amavis-ng and amavis-new. This tutorial will only handle the amavis-new version.

To filter the incoming mails, Postfix must send the message to AMaViS. AMaViS then unpacks the message and related attachments and forwards the contents to one or more virus scanners. Additionally the message can also be forwarded to SpamAssassin.

To do this work, there are different steps to do:

- ▶ amavisd-new opens on localhost:10024 a port with a SMTP server
- ▶ Postfix relays the message to localhost:10024. That means that the message leaves the Postfix system
- ▶ amavisd-new filters the incoming message on its localhost:10024 port. If the message contains no virus, amavisd-new gives the message back to Postfix using the localhost:10025 port. To avoid mail loops, the messages that enters on localhost:10025 port will not be forwarded to amavisd-new
- ▶ If a mail includes a virus, the message will not send back to Postfix, but amavisd-new will store it in a special directory.

Figure 5.2: Postfix in combination with AMaViS

5.8.2 Configuring Postfix to use Amavis-new

The first step is to install amavisd-new. As we are using Debian, the package management can install it:

```
# apt-get install amavisd-new
```

After installation, we have to tell Postfix to send the mails out to amavisd-new. We have to define a new transport method by adding the following into the `/etc/postfix/master.cf` file:

```
smtp-amavis unix - - n - 2 smtp -o smtp_data_done_timeout=1800  
-o disable_dns_lookups=yes
```

Furthermore we have to define the smtp server without `content_filter` running on the `localhost:10025` port. We have to add this line to `/etc/postfix/master.cf`

```
localhost:10025 inet n - n - - smtpd -o content_filter=
```

After restarting Postfix, the new smtp server should answer on port 10025. We can test it using the following command:

```
# telnet localhost 10025
```

Now we have to tell Postfix to relay all incoming mails to `amavisd-new` which listens on port `localhost:10024`. This parameter is specified in the `/etc/postfix/main.cf` file:

```
content_filter = smtp-amavis:[127.0.0.1]:10024
```

5.8.3 Configuring amavisd-new

The `/etc/amavis/amavis.conf` file is a large Perl script to define the parameters. For the moment we only define a few essential parameters, to tell `amavisd-new` to establish a smtp server on port 10024:

```
$mydomain = 'tux-industries.lu';
$myhostname = 'mailserver.tux-industries.lu';
$forward_method = 'smtp:127.0.0.1:10025'; # where to forward checked mail
$inet_socket_port = 10024;
```

On a Debian configuration, you should now be able to start `amavisd-new` and test it using the following command:

```
# telnet localhost 10024
```

`amavisd-new` should now answer on port 10024.

5.8.4 Which Virus-scanner to use?

AMaViS itself doesn't contain a virus-scanner. So we need a virus-scanner to scan and to find viruses in our messages. There are many products on the market, but for this tutorial we want to use a free virus scanner, so the decision to use ClamAV has been taken. ClamAV is maybe not the best anti-virus, but it shows how to integrate a virus checker it might be sufficient. AMaViS also allows to use more than one anti virus product. That means that you can install more than one virus-scanner, and configure AMaViS to use all scanner on your system. This is even recommended for mission-critical systems.

Important for your choice of a virus-scanner is that the virus-definition files are up to date, and that these files can be installed easily using a script, so that the system administrator has not to pay attention to these updates.

5.8.5 ClamAV

Clam AntiVirus is a GPL anti-virus toolkit for UNIX. The main purpose of this software is the anti-virus integration in mail servers (attachment scanning). The package provides a flexible and scalable multi-threaded daemon, a command line scanner, and a tool for automatic updating via Internet. The programs are based on a shared library, distributed with the Clam AntiVirus package, which you can use with your own software. Most importantly, the virus database is kept up to date .

Main features are:

- ▶ command-line scanner
- ▶ fast, multi-threaded daemon
- ▶ milter interface for sendmail
- ▶ database updater with support for digital signatures
- ▶ virus scanner C library
- ▶ on-access scanning (Linux and FreeBSD)
- ▶ detection of over 29000 viruses, worms and trojans
- ▶ built-in support for RAR (2.0), Zip, Gzip, Bzip2, Tar, MS OLE2, MS Cabinet files, MS CHM (Compressed HTML), MS SZDD
- ▶ built-in support for mbox, Maildir and raw mail files
- ▶ built-in support for Portable Executable files compressed with UPX, FSG, and Petite

5.8.6 Installing and configuring ClamAV

Before installing ClamAV you should verify if the user and group clamav exists. If not they can be created with `useradd` and `groupadd`.

ClamAV can be installed as a Debian package. The following command will install the software:

```
# apt-get install clamav
# apt-get install clamav-daemon
```

Because a virus-scanner needs up to date virus definition files, the first step after installation is to download the latest virus definition file. This can be done using `freshclam`. `freshclam` updates the virus database. It's a part of the Clam AntiVirus package. It requires an Internet connection. `freshclam` can be run in background to check regularly for new virus definition files. For the first time, we will start it in foreground:

```
# freshclam
ClamAV update process started at Sat Jan 15 09:10:35 2005
Downloading main.cvd [*] main.cvd updated (version: 29, sigs: 29086, f-level: 3, builder: tomek)
Downloading daily.cvd [*] daily.cvd updated (version: 665, sigs: 344, f-level: 3, builder: tomek)
Database updated (29430 signatures) from db.lu.clamav.net (195.70.36.141).
```

ClamAV comes with two binaries for scanning files:

- ▶ `clamscan` is a stand-alone binary to scan a file, a directory
- ▶ `clamdscan` works together with the `clamd` daemon. With this version you can have a gain of performance when scanning a huge amount of files.

To test a virus-scanner we need of course a virus to check if the virus-scanner filters it. Using “real” viruses is dangerous. There are test viruses out, and these viruses can be used without compunction. They can be downloaded at www.eicar.org.

- ▶ Download the virus sample files from www.eicar.org.
- ▶ Copy them into a directory `/home/harpes/virus-test`
- ▶ Copy other files to this directory
- ▶ `$ cd /home/harpes/virus-test`
- ▶ `$ clamavscan`
- ▶ To test `clamavd` run `clamavscand`

5.8.7 Integrate ClamAV with AMaViS

Once the virus-scanner and AMaViS run, we have to configure AMaViS to use ClamAV for checking viruses. This could be done in the `/etc/amavis/amavis.conf` file. The configuration file is a Perl script divided into different section. The virus-scanner settings are located in section VII. In this section the Perl list `@av_scanner` is defined. This list contains many virus-scanner. To configure AMaViS to use ClamAV please locate first this list, and then look if there is an entry for ClamAV. This entry should look like this:

```
### http://www.clamav.net/
['Clam Antivirus-clamd',
 \&ask_daemon, ["CONTSCAN {}\n", "/var/run/clamav/clamd.ctl"],
 qr/\bOK$/, qr/\bFOUND$/,
 qr/^.*?: (?!Infected Archive)(.*) FOUND$/ ],
# NOTE: run clamd under the same user as amavisd; match the socket
# name (LocalSocket) in clamav.conf to the socket name in this entry
# When running chrooted one may prefer: ["CONTSCAN {}\n", "$MYHOME/clamd"],
```

The expression `/var/run/clamav/clamd.ctl` is the path to the localsocket to connect to `clamavd`. To be sure that AMaViS and ClamAV use the same socket, you have to check in the ClamAV configuration file the socket settings. This configuration file can be found in `/etc/clamav/clamad.conf`.

```
LocalSocket /var/run/clamav/clamd.ctl
```

5 Mail and Virus checking

The last thing to do to let AMaViS work with ClamAV is to change the owner of `/var/run/amavis`:

```
# chmod -R amavis:amavis /var/run/amavis
```

Per default, AMaViS doesn't notify the mail recipient that the mail has been infected by a virus. To notify the mail recipient that he received a infected mail we should uncomment the `$warnvirusrecip = 1;` parameter in `amavis.conf`.

After restarting Postfix and AMaViS your system should now be able to filter viruses. You can test it by sending the sample virus `eicar`.

6 Anti-Spam configuration with Amavis, SpamAssassin and Postfix

by Christian Mock, linuxwochen.at@quintessenz.org <cm@quintessenz.org>

6.1 Short intro on spam

6.1.1 Risk assessment

Automatically classifying messages as either spam or ham (non-spam) has the risk of misclassification. A piece of spam classified as ham is a False Negative, a piece of ham classified as spam is a False Positive. In an ideal world, you would be able to reduce both counts to zero.

Since we don't live in an ideal world, you need to think about the issue: what level of FP and FN can you and your organization accept? Would you rather have your users see no spam at all and risk losing legitimate mail, or can they live with a few spams per day and person and virtually never lose legitimate mail?

The next question, then, is what to do with the spam.

- ▶ **The choices are Rejecting during the SMTP session**

This means the sender will get a bounce message in the case of an FP, but the faked sender of a spam will not get one (because spam-sending SMTP engines don't generate bounces).

- ▶ **Bouncing**

This means the sender of an FP will get a bounce, but also that the faked sender of a spam will be spammed by you with a bounce for a message he didn't send.

- ▶ **Dropping**

This means you leave the faked spam sender alone, but the sender of an FP gets no notice of his mail disappearing, either.

- ▶ **Tagging**

You pass along the message to the recipient, but tag it in the subject and/or header. The recipient can automatically filter it to his "junk" folder, which also means that it most probably has the same effect as dropping, except you (as the mail admin) aren't the one dropping the important message from the important customer (which will inevitably come up in the discussion), but the end-user will.

6.1.2 Where does spam come from?

Spam is either sent from the spammer's machine (hosted in a datacenter of an anti-social ISP somewhere), or from a machine the spammer abuses for his purposes. This can be an open SMTP relay, an open proxy, or a "zombie" machine infected with malware that is used to spam and DDoS.

6.1.3 What characteristics can we use to detect spam?

- ▶ Where does the message come from? (IP address, hostname, ISP, country)
- ▶ How is it sent? (Characteristics of the SMTP session)
- ▶ Technical features (Headers, MIME structure, Text and HTML parts)
- ▶ Content (Easy for humans, hard for computers)

More material is in my talk held at the last linuxdays.lu:

<http://www.tahina.priv.at/~cm/talks/spamblocking.pdf>.

6.1.4 Blacklists/RBLs/DNSBLs

"Realtime Blackhole Lists"/"DNS based Blacklists" DNSBLs are lists of IP addresses or host/domain names that match certain criteria, like "open relay", "open proxy", "has sent mail to a spamtrap address recently", "assigned to a well known spammer", "located in country".

DNSBLs are queried via DNS; for IP address lists, the address is reversed like for a TR query; if we were to look for 209.88.103.4:

```
$ host -a 4.103.88.209.proxies.relays.monkeys.com
4.103.88.209.proxies.relays.monkeys.com IN A 127.0.0.2
4.103.88.209.proxies.relays.monkeys.com IN TXT "BLOCKED: See
http://www.monkeys.com/upl/listed-ip-0.cgi?ip=209.88.103.4"
```

For lists of host/domain names ("RHSBLs" for "Right Hand Side"), the hostname is prepended to the list's domain:

```
# host -t txt dvd-porno-shop.com.multi.surbl.org
dvd-porno-shop.com.multi.surbl.org text "Blocked,
dvd-porno-shop.com on lists [ws][ob][ab][jp],
See: http://www.surbl.org/lists.html"
```

IP DNSBLs can be used to refuse mail from the listed machines; RHSBLs can be used to check URLs in the messages against.

You rely on an external entity, so choose a trustworthy source; DNSBL operators have listed ISPs they didn't like in the past, and one has closed down his list by listing ALL IP addresses.

My recommendations:

- ▶ cbl.abuseat.org - based on spamtraps, lists proxies/zombies, no FP seen; <http://cbl.abuseat.org/>
- ▶ list.dsbl.org - open relays/proxies/etc; no FP seen; <http://dsbl.org/>
- ▶ sbl.spamhaus.org - lists well-known spammer's hosting IP addresses; virtual no chance of FP, operators highly regarded in the community; <http://www.spamhaus.org/>

More:

- ▶ <http://www.declude.com/Articles.asp?ID=97>
- ▶ <http://www.moensted.dk/spam/>
- ▶ <http://openrbl.org/>
- ▶ <http://www.sconsult.com/bill/dnsblhelp.html>

6.1.5 Using DNSBLs in Postfix

Using a DNSBL in Postfix means mail from IP addresses on that DNSBL will be REJECTED!
Does your policy allow that?

Using IP-based DNSBLs in Postfix is very simple, you just add the list to `smtpd_recipient_restrictions` in `/etc/postfix/main.cf`.

Postfix also has "`smtpd_client_restrictions`" (for the connecting host), "`smtpd_helo_restrictions`" (to check the SMTP HELO/ESMTP EHLO parameter given by the client), "`smtpd_sender_restrictions`" (for the envelope sender address); these react directly to the respective SMTP command by giving an error code if the result is "REJECT", but some SMTP implementations ignore that, compatibility is biggest if the error is returned only when the client issues the RCPT TO command, even if it is the result of an earlier command. Plus we get a single configuration value to implement our SMTP-level policy...

"`smtpd_sender_restrictions`" contains a list of tests which can return "PERMIT", "REJECT" or nothing; the first match is used, so the general layout should be:

```
smtpd_recipient_restrictions =  
<permit LAN users to send mail>,  
<deny mail to non-local destinations>,  
<whitelist for "spam lovers">,  
<whitelist for spam FP>,  
<block spam>,  
permit
```

DNSBLs can be queried via "`reject_rbl_client dnsbl.domain.name`".

Example:

6 Anti-Spam configuration with Amavis, SpamAssassin and Postfix

```
smtpd_recipient_restrictions =
  permit_mynetworks,
  reject_unauth_destination,
  check_recipient_access hash:/etc/postfix/spamlovers,
  check_sender_access hash:/etc/postfix/sender_whitelist,
  check_client_access hash:/etc/postfix/client_whitelist,
  check_helo_access hash:/etc/postfix/helo_restrictions,
  reject_non_fqdn_sender,
  reject_unknown_sender_domain,
  reject_rbl_client sbl.spamhaus.org,
  reject_rbl_client cbl.abuseat.org,
  permit
```

In Detail:

permit_mynetworks Permit LAN users to send mail (LAN addresses defined in mynetworks in main.cf)

reject_unauth_destination Deny mail for non-local destinations (i.e. not mydestination, virtual_*_domains, relay_domains, ...); prevent unauthorized relaying. Everything which passes this line is therefore for a local recipient.

check_recipient_access hash:/etc/postfix/spamlovers, Allow certain addresses to receive unfiltered mail, e.g. <postmaster@luxembourg.tux-industries.com>

check_sender_access hash:/etc/postfix/sender_whitelist,
check_client_access hash:/etc/postfix/client_whitelist,
Whitelists for sender addresses (SMTP envelope!) and client IP addresses/hostnames, so you can whitelist people whose servers/ISPs happen to be on DNSBLs.

check_helo_access hash:/etc/postfix/helo_restrictions, Reject HELO/EHLO with my hostname, domain name or IP address

reject_non_fqdn_sender Reject sender addresses without fully qualified domain part

reject_unknown_sender_domain Reject sender addresses with non-existing domain (returns temporary 4xx error code, can take 5 days for mail to bounce if sender has DNS problems)

reject_rbl_client sbl.spamhaus.org

reject_rbl_client cbl.abuseat.org Reject IP addresses on one of these lists

permit Anything else will be accepted

Example map files:

```
spamlovers:
postmaster@luxembourg.tux-industries.com PERMIT
abuse@luxembourg.tux-industries.com PERMIT

sender_whitelist:
user@example.com PERMIT
some-domain.com PERMIT

client_whitelist: 10.2.3.4 PERMIT
mail.example.com PERMIT

helo_restrictions: mailserver.luxembourg.tux-industries.com 554 That's me, liar!
tux-industries.com 554 That's me, liar!
10.4.5.6 554 That's my address, spammer!
my-other-domain.lu 554 That's also me, liar!
```

Don't forget "postmap filename" after changing a map file.

6.1.5.1 Testing the setup

If you know what you want, to avoid configuration errors (typos, ...):

```
soft_bounce = yes
```

Turns all 5xx SMTP errors (hard errors) into 4xx (soft errors), which causes the sender to retry, so mail won't get lost.

If you want to know what a certain configuration would achieve without interfering with mail delivery, use `warn_if_reject`, e.g.

```
smtpd_recipient_restrictions = [...]
warn_if_reject reject_rbl_client new-dnsbl.org, [...]
```

"`reject_rbl_client new-dnsbl.org`" is evaluated, but the results are only logged ("`reject_warning`"), not used to block. Analyze the log, and if you're happy, remove the "`warn_if_reject`" from the line.

After a while, the log file `/var/log/mail.log` should contain lines like these:

```
Jan 23 06:21:21 trumm postfix/smtpd[25929]: NOQUEUE: reject: RCPT
from
DSL01.212.114.234.165.NEFkom.net [212.114.234.165]: 571 Service unavailable;
Client host [212.114.234.165] blocked using list.dsbl.org;
http://dsbl.org/listing?212.114.234.165;
from=<savspxseakkoqf@dcemail.com> to=<user@tahina.priv.at>
proto=SMTP helo=<DSL01.212.114.234.165.NEFkom.net> Jan 23 06:22:09 trumm
postfix/smtpd[25929]: NOQUEUE: reject: RCPT from unknown[211.187.64.233]:
```

6 Anti-Spam configuration with Amavis, SpamAssassin and Postfix

```
554 <194.152.163.253>: Helo command rejected: Don't lie to me.;  
from=<savspkxseakkoqf@dcemail.com> to=<user@tahina.priv.at>  
proto=SMTP helo=<194.152.163.253>
```

The first was rejected because the sending machine's IP address is listed in the dsbl.org DNSBL, the second because the HELO command contained my IP address.

6.2 SpamAssassin

SpamAssassin (<http://www.spamassassin.org/>) is the Open Source anti-spam filter, continually improved and adapted to spammer's new techniques. How does it work? SpamAssassin analyzes structure and content of a message with about 670 tests. Each test has a score associated with it; SA sums up the scores of the tests that matched, and if the sum is bigger than the (configurable) limit then the message is considered spam and can have "*****SPAM*****" added to the subject, X-Spam-Status headers, etc. The default limit is 5.0 points.

Tests include: MIME structure, keywords and key-phrases in headers and body, header analysis, DNSBL and RHSBL lookups, and Bayesian filtering.

Bayesian filtering is a technique where a database is trained on existing data sets; it works on single words (or "tokens"), where the training phase supplies the probability that a token occurs in spam and ham; then, you can look up the tokens from a new message in the DB and calculate the probability that it is spam or ham.

There's an "auto-whitelisting" feature which adapts the score of the current message based on the past scores messages from that sender have got; so if somebody who sent you low-scoring mail in the past and sends a message which triggers a few tests in SA now, the score will be adjusted downwards. The AWL database works on a combination of sender address and sending IP address, so a spam faking your friend's email address will not have its score adjusted since it will not come from your friend's mail server.

6.2.1 Installation

Use a current version of SpamAssassin! If you're using Debian Woody (3.0), put

```
deb http://www.backports.org/debian woody spamassassin
```

in your `/etc/apt/sources.list`, or get the source from <http://www.spamassassin.org/> and install it yourself. Sarge, the Debian release we're using, has an up-to-date version, but this will be frozen once Sarge becomes stable...

Take care to install all the recommended modules, too - especially `libnet-dns-perl` and `libmail-spf-query-perl` on Debian, or you'll lose all DNS and SPF-related functionality.

Do not start `spamd` - we'll integrate SA into Amavis in a few moments.

Debian's SpamAssassin installs into

```
/etc/spamassassin - Site-Wide configuration
```

```
/usr/share/spamassassin - Rules
```

6.2.2 Configuration

The system-wide SpamAssassin config file is in `/etc/spamassassin/local.cf`, per-user config is in `~/.spamassassin/user_prefs`. For reasons of simplicity, we'll only use site-wide config here. We're using SA 3.0.2 in the examples; make sure to read the `Mail::SpamAssassin::Conf` documentation either via "man" or "perldoc" to see which options other versions support.

What do we want to achieve?

- ▶ Messages classified as spam are marked with "*****SPAM*****" in the subject
- ▶ A system-wide bayesian database with a reasonable size
- ▶ Use network tests (DNSBLs etc)
- ▶ Use the auto-whitelisting feature

Debian's `local.cf` is empty. We add the following statements:

```

auto_whitelist_path /var/lib/amavis/.spamassassin/whitelist
bayes_path /var/lib/amavis/.spamassassin/bayes
bayes_expiry_max_db_size 1500000
trusted_networks 127/8 192.168.1/24
skip_rbl_checks 0

ok_languages de en it
ok_locales en

rewrite_header subject *****SPAM*****

report_safe 0
report_contact postmaster@tux-industries.com

lock_method flock

```

Explanation:

```

auto_whitelist_path /var/lib/amavis/.spamassassin/whitelist
bayes_path /var/lib/amavis/.spamassassin/bayes

```

These are in `~/.spamassassin/` (per-user) normally, we put them in a central location to make them system-wide.

```

bayes_expiry_max_db_size 1500000

```

The default is much too low; 1 Mio uses about 32MB of disk space, so see how much "cached" RAM there is in the "free" output and size accordingly if you run a busy installation - you want the bayes DB to be cached all the time, and you want a large DB so the spams with lists of random words in them don't force the interesting spam keywords out of the DB.

```

trusted_networks 127/8 192.168.1/24

```

This should be automatically detected, but the algorithm isn't very reliable; give all IP addresses/networks of machines you trust, i.e. your LAN, your secondary MX etc.

The difference (302 vs 297) is because SpamAssassin doesn't learn from all messages; there may be duplicates (by Message-ID), or messages may be too old or too small.

In the end, it should look like this:

```
# ls -l /var/lib/amavis/.spamassassin/
total 1116 -rw----- 1 amavis amavis 1520 Jan 23 03:15 bayes.mutex
-rw----- 1 amavis amavis 86016 Jan 23 03:15 bayes_seen
-rw----- 1 amavis amavis 1314816 Jan 23 03:15 bayes_toks
# sa-learn --dump magic 0.000 0 3 0 non-token data: bayes db version
0.000 0 299 0 non-token data: nspam
0.000 0 297 0 non-token data: nham
0.000 0 47549 0 non-token data: ntokens
0.000 0 1094658931 0 non-token data: oldest atime
0.000 0 1106314819 0 non-token data: newest atime
0.000 0 0 0 non-token data: last journal sync atime
0.000 0 0 0 non-token data: last expiry atime
0.000 0 0 0 non-token data: last expire atime delta
0.000 0 0 0 non-token data: last expire reduction
count
```

"bayes.mutex" is a lock file; "bayes_seen" contains the Message-IDs of the messages trained (and auto-learned), and bayes_toks has all the token data.

"nspam" and "nham" are the number of learned spams and hams, respectively; "ntokens" is the number of tokens (words) in the DB, and the "atime" values are related to the expiry of the DB, which is the process to keep the size below the limit (bayes_expiry_max_db_size).

6.2.4 Testing SpamAssassin

Save a single piece of spam to a file, in mbox format (e.g. "save" from mutt); run "spamassassin -t < file | less" and look at the output. Try this with a few messages and make sure there's RCVD_IN_DNSBL and BAYES_nn in at least some of the X-Spam-Status headers, i.e. that DNS lookups and the bayes DB work.

At the end of the output, it shows a summary:

```
Content analysis details: (3.4 points, 5.0 required)
pts rule name description
-----
0.7 DATE_IN_PAST_12_24 Date: is 12 to 24 hours before Received: date
0.7 SUBJ_ALL_CAPS Subject is all capitals
1.6 PORN_URL_MISC URI: URL uses words/phrases which indicate porn (misc)
0.4 BAYES_60 BODY: Bayesian spam probability is 60 to 80%
[score: 0.7436]
```

There's something wrong here - no DNS tests seem to have happened. Let's look at SA's debug output (the "-D" flag):

6 Anti-Spam configuration with Amavis, SpamAssassin and Postfix

```
# spamassassin -D -t < file >/dev/null [...]
debug: failed to load Net::DNS::Resolver: Can't locate Net/DNS.pm in @INC
(@INC contains: /usr/share/perl5 /etc/perl /usr/local/lib/perl/5.8.4
/usr/local/share/perl/5.8.4 /usr/lib/perl5 /usr/lib/perl/5.8
/usr/share/perl/5.8 /usr/local/lib/site_perl) at
/usr/share/perl5/Mail/SpamAssassin/Plugin/URIDNSBL.pm line 113.
[...]
```

Seems I didn't follow my own instructions and forgot the Net::DNS perl module; so "apt-get install libnet-dns-perl libmail-spf-query-perl" and the result looks much better:

```
Content analysis details: (18.0 points, 5.0 required)
pts rule name description
-----
0.7 DATE_IN_PAST_12_24 Date: is 12 to 24 hours before Received: date
0.7 SUBJ_ALL_CAPS Subject is all capitals
1.6 PORN_URL_MISC URI: URL uses words/phrases which indicate porn (misc)
0.4 BAYES_60 BODY: Bayesian spam probability is 60 to 80% [score: 0.7436]
3.8 RCVD_IN_DSBL RBL: Received via a relay in list.dsbl.org [<http://dsbl....
...org/listing?61.143.199.38>]
1.2 RCVD_IN_BL_SPAMCOP_NET RBL: Received via a relay in bl.spamcop.net ...
...[Blocked - see <http://www.spamcop.net/bl.shtml?61.143.199.38>]
0.4 RCVD_IN_NJABL_PROXY RBL: NJABL: sender is an open proxy ...
...[61.143.199.38 listed in combined.njabl.org]
3.1 RCVD_IN_XBL RBL: Received via a relay in Spamhaus XBL ...
...[61.143.199.38 listed in sbl-xbl.spamhaus.org]
1.0 URIBL_SBL Contains an URL listed in the SBL blocklist ...
...[URIs: dvd-porno-shop.com]
0.4 URIBL_AB_SURBL Contains an URL listed in the AB SURBL blocklist ...
...[URIs: dvd-porno-shop.com]
1.5 URIBL_WS_SURBL Contains an URL listed in the WS SURBL blocklist ...
...[URIs: dvd-porno-shop.com]
3.2 URIBL_OB_SURBL Contains an URL listed in the OB SURBL blocklist ...
...[URIs: dvd-porno-shop.com]
```

We went from 3.4 points to 18.0, i.e. from "not detected" to "very high probability of spam".

6.3 Amavis

Config file is /etc/amavis/amavisd.conf; the following variables are of interest:

```
#@bypass_spam_checks_acl = qw( . );
$sa_local_tests_only = 0;
$sa_auto_whitelist = 1;
$sa_mail_body_size_limit = 200*1024;
```

```

$sa_tag_level_deflt = -1000;
$sa_tag2_level_deflt = 5.0;
$final_spam_destiny = D_PASS;
$warnspamsender = undef;

```

Explanation:

```
#@bypass_spam_checks_acl = qw( . );
```

By default, no spam checks are run so Amavis needn't depend on SpamAssassin; we have to comment out this line to enable SA.

```
$sa_local_tests_only = 0;
```

Enable the DNS-based tests.

```
$sa_auto_whitelist = 1;
```

Use the auto-whitelist feature.

```
$sa_mail_body_size_limit = 200*1024;
```

Spam is usually below 200kB, and SA can take up a lot of RAM for big mails.

```
$sa_tag_level_deflt = -1000; $sa_tag2_level_deflt = 5.0;
```

We want X-Spam-headers in all messages (first line), "***SPAM***" in the subject from 5 points on.

```
$final_spam_destiny = D_PASS;
```

D_PASS means tagged spam goes to the recipient, who can filter it into a folder. D_DISCARD drops the message (meaning nobody will detect false positives!), and D_BOUNCE would harass the innocent victims whose mail addresses were stolen for this spam run.

```
$warnspamsender = undef;
```

DO NOT CHANGE THIS! Spam is sent with faked sender addresses. The only situation where setting this variable could make sense is when your mailserver is outgoing only and you want to alert your users that their mail is detected as spam...

There's many more Amavis features, but these are the basics to get going, and the Amavis config file is 90% comments explaining what all the options mean.

Run `/etc/init.d/amavis reload` to apply the new settings.

6.3.1 Testing

Since all mail through our server will be scanned, we just need to send ourselves a piece of mail; start with a simple test to see if there's any signs of SpamAssassin in the headers:

```
date | mail -s test user
```

and look at it with "mutt" - the "h" keys shows all headers and should reveal something like this:

```

X-Virus-Scanned: by amavisd-new-20030616-p10 (Debian) at tux-industries.com
X-Spam-Status: No, hits=-4.3 tagged_above=-1000.0 required=5.0
tests=ALL_TRUSTED, BAYES_00, NO_DNS_FOR_FROM
X-Spam-Level:

```

6 Anti-Spam configuration with Amavis, SpamAssassin and Postfix

This means SpamAssassin is being used by Amavis. So let's try with our previously used sample:

```
/usr/sbin/sendmail user < file
```

and look at the result in mutt, again:

```
X-Virus-Scanned: by amavisd-new-20030616-p10 (Debian) at tux-industries.com
X-Spam-Status: Yes, hits=21.1 tagged_above=-1000.0 required=5.0
tests=BAYES_99, DATE_IN_PAST_12_24, NO_DNS_FOR_FROM, PORN_URL_MISC,
RCVD_IN_BL_SPAMCOP_NET, RCVD_IN_DSBL, RCVD_IN_NJABL_PROXY, RCVD_IN_XBL,
SUBJ_ALL_CAPS, URIBL_AB_SURBL, URIBL_OB_SURBL, URIBL_SBL, URIBL_WS_SURBL
X-Spam-Level: ***** X-Spam-Flag: YES
```

6.4 Advanced issues

Or: "Issues I will only mention in passing because time will run out well before the basics are done"

6.4.1 Whitelisting/Blacklisting in SpamAssassin

You may be getting newsletters which you actually subscribed to tagged as spam; or you may get spammed from a constant From: address and SA doesn't detect it. The solution is "whitelist_from add@ress" or "blacklist_from add@ress" in /etc/spamassassin/local.cf; don't forget "/etc/init.d/amavis reload" afterwards.

If you are subscribed to a mailinglist which gets messages tagged as spam (e.g. a ML for discussing spam issues), use Amavis' "whitelist_sender"; there's already a long list of addresses in the default amavisd.conf (search for "nobody@cert.org"), make sure you use the envelope sender address.

6.4.2 Performance Issues

The maximum amount of messages/time SpamAssassin can handle is limited by the scan time per message; this is usually around 5-10 seconds for SpamAssassin, plus the virus scanning overhead. Our settings limit amavis to 2 concurrent connections, both in

```
/etc/postfix/master.cf smtp-amavis unix - - - - 2 smtp
```

and in /etc/amavis/amavisd.conf

```
$max_servers = 2;
```

If that is too low (i.e. your mail queue gets long during high activity periods), you need more concurrent connections. For SA processing the limiting factor is RAM (assuming you're running on a CPU dating from this millenium), roughly 20-30MB per concurrent process, and the on-disk size of the databases, so those can stay cached in RAM ("du -sh /var/lib/amavis/.spamassassin" gives an estimate for this value).

Increase both of these values depending on the amount of RAM in your machine; I run 5 concurrent processes on 256 MB mail gateways for a customer which do nothing but spam checking.

6.4.3 Getting Feedback to SpamAssassin

The efficiency of the Bayesian Filter depends heavily on the feedback you give - the auto-learning feature is nice, but can only detect extreme cases, so the fine-tuning is left to the user.

Basically, what you need to do is to feed back at false positives and false negatives using "sa-learn -ham" or "sa-learn -spam". Doing this the manual way can get boring quite fast, so set up a pair of aliases in /etc/aliases:

```
secret+spam: amavis+spam
secret+ham: amavis+ham
```

(The "secret" part is so that spammers can't easily guess the address and send their spam to the ham training address).

Create .forward files in /var/lib/spamassassin

```
# ls -la [...]
-rw-rw-r-- 1 amavis amavis 40 Mar 2 2004 .forward+ham
-rw-rw-r-- 1 amavis amavis 41 Mar 2 2004 .forward+spam
[...]
# cat .forward+ham "|/usr/bin/sa-learn --ham -p /dev/null"
# cat .forward+spam "|/usr/bin/sa-learn --spam -p /dev/null"
```

What this does is to use Postfix' "recipient_delimiter" feature to create two "sub-aliases" for the amavis account, which then get handled by Postfix according to the ".forward+extension" files; those feed the messages to sa-learn with the correct parameters.

You then can use the "bounce" functionality (and only that!) of your MUA to send a 1:1 copy of the interesting messages to these accounts; a pair of keybindings (I use "H" for "learn as ham" and "S" for "learn as spam") makes this very easy to use.

If you happen to use a MUA which doesn't allow bouncing, you can forward the messages as MIME attachments and need to write a little script which extracts the attachment and feeds that to sa-learn... Or a script which extracts multiple of those attachments, sanitizes them, and feeds them to sa-learn, so Outlook users can drag-and-drop their false positives/false negatives into a new message (as attachments) and send this to the learning aliases; you get the idea.

7 File services

by Alain Knaff

Samba (named after Microsoft's **S**erver **M**essage **B**lock protocol) is an Open Source/Free Software suite that provides seamless file and print services to Windows clients. It can act as a primary domain controller (authentication server) to all major variants of Windows.

The course will show how to:

- ▶ install the necessary software
- ▶ configure samba for some basic file service tasks
- ▶ configure samba for to operate as a PDC
- ▶ use an LDAP server to manage its user data base

7.1 What is Samba

As the front page at samba.org says, "Samba is an Open Source/Free Software suite that provides seamless file and print services to SMB/CIFS clients." Samba is freely available, unlike other SMB/CIFS implementations, and allows for interoperability between Linux/Unix servers and Windows-based clients.

Samba is a software package that gives network administrators flexibility and freedom in terms of setup, configuration, and choice of systems and equipment. Because of all that it offers, Samba has grown in popularity, and continues to do so, every year since its release in 1992.

7.2 Installation & Configuration

The installation of Samba on Debian GNU/Linux is normally quite easy! The packages are pre-built, and you just have to run `apt-get` for the installation of the needed packages. Thus, the first step is:

```
apt-get install samba smbclient
```

This has to be done as root! During the installation, a few questions will be asked to build an initial database.

| Question | Answer |
|--|-----------|
| Workgroup/Domain Name? | [country] |
| Use password encryptions? | Yes |
| Modify <code>smb.conf</code> to use WINS settings from DHCP? | No |
| How do you want to run Samba? | daemons |
| Create samba password database? | No |

After the installation, you should be able to do a first test (samba is automatically started from the system). This example assumes that your machine is called `bruxelles`:

```
smbclient -L bruxelles
```

This will ask for a password, simply type return, and you will see a list of all shares that are defined, as well as some other servers on which exist on your LAN ¹.

7.2.1 Structure of the configuration file

The configuration file is made up of various *sections*. These are named `[sectionname]`. Most sections represent file or print shares. Parameters which apply to samba as a whole are set in a `global` section.

There are a number of reserved shares or sections, such as for example

- ▶ `printers` defines parameters for printers which are not explicitly listed
- ▶ `netlogon` is used for holding the Windows startup scripts when operating as a PDC
- ▶ ...

7.3 Samba as a simple file server

7.3.1 Global parameters

The following global parameters are relevant to simple file server operation:

| Name | Description |
|---------------------------|---|
| <code>workgroup</code> | Windows workgroup or Domain |
| <code>netbios name</code> | NetBIOS name by which a Samba server is known. By default it is the same as the first part of the Unix host name. |
| <code>printing</code> | Identifies the printing system. One of <code>plp</code> , <code>lprng</code> or <code>cups...</code> |
| <code>wins support</code> | <code>yes</code> if this server should be a wins server, <code>no</code> otherwise. |
| <code>wins server</code> | the IP address of the Wins server (only set this if <code>wins support</code> is <code>no</code>). |

Set `wins support` to `no` if you intend to put your server into an existing Windows network, which already has a Wins server.

¹the listing will only be shown on a samba server configured as a wins server, and will in most cases only contain those machines that use your server as a Wins server. Use `smbclient -L yourwinsserver` to explicitly query your wins server, rather than the local machine.

7 File services

7.3.2 Per share parameters (all shares)

The following parameters are useful on all shares:

| Name | Description |
|-------------------------|---|
| <code>comment</code> | Informative text to be displayed near share in Windows Browser |
| <code>browseable</code> | If set to yes, share shows up in network neighborhood, else is hidden |
| <code>public</code> | All users may access this share |
| <code>read only</code> | Users may only read from share |
| <code>available</code> | If set to no, share is switched off |

7.4 Parameters for file shares

The following parameters are useful on file shares:

| Name | Description |
|-------------------|--|
| <code>path</code> | Path of Unix directory which is exported in this share |

7.4.1 Parameters for printer shares

The following parameters are useful on printer shares:

| Name | Description |
|---------------------------|--|
| <code>printable</code> | must be set to yes |
| <code>printer</code> | Unix (Cups, ...) printer corresponding to this share |
| <code>path</code> | Path of temporary directory where print jobs are to be spooled to (by default /tmp) |
| <code>cups options</code> | If your printing system is cups, this specifies the options passed on to cups. Usually, "raw,media=a4". These mean that "raw" mode should be used (no postscript processing, because in the Windows world, the "printer driver" lives on the client), and that the paper size is A4. |

7.4.2 Parameters for the global printers share

If you don't want to define each printer individually, you can set up a global `printers` share which exports all printers known locally to Windows.

For this, set `load printers = yes` in the global section, and define a `printers` section.

7.4.3 User management

So far, we have not yet defined any users in Samba. The file server is already usable, but only for anonymous access (guest user).

If you want to set up named access, the following parameters need to be defined:

| Name | Description |
|-------------------------------|--|
| <code>encrypt password</code> | <code>yes</code> |
| <code>guest user</code> | Name of Unix user who will serve anonymous (guest) requests (usually <code>nobody</code>) |
| <code>username map</code> | (Optional) Name of a file which maps long Windows user names to short Unix login names |

The `username map` has the following format (left is the Unix login, right the long Windows name):

```
root = admin administrator
tridge = "Andrew Tridgell"
```

Once these changes are done, you add samba users using the following command:

```
smbpasswd -a user
```

These users have to be existing Unix users; the `smbpasswd` command only enables them for samba.

7.4.4 Testing

The following tools are available for testing:

- ▶ `testparm`: this parses the `/etc/samba/smb.conf`, mentions any errors that it finds, and waits for a keystroke. After the keystroke, it prints out the whole configuration, as understood by samba
- ▶ `smbclient`: `smbclient` is a samba client that allows you to access your file server, just as a Windows workstation would. Of course, it can also be used to access a real Windows server.

```
smbclient -L server -U user
```

```
smbclient //server/share -U user
```

The first command logs in as *user* and lists all shares on *server*.

The second command logs in as *user* connects to *share* on *server*. Once connected, the you get an ftp-like command line interface to get and put files on the server.

- ▶ log files are put into `/var/log/samba/machine.log`, where *machine* is the netbios name of the client having connected. Set `log level` to at least 3 in the global section of `smb.conf` to see a log of all files that are opened.

7.4.5 Example

This example shows the configuration file of a simple file server:

```
[global]
    workgroup = samba
    printing = cups
    cups options = "raw,media=a4"
    load printers = yes
    encrypt passwords = yes
    log level = 3
[public]
    comment = A Test Share
    browseable = yes
    public = yes
    read only = yes
    path = /samba/public
[authenticated]
    comment = An authenticated share
    browseable = yes
    read only = no
    path = /samba/auth

[printers]
    comment = Printers share
    printable = yes
```

Exercises:

- ▶ Create the share directories on Unix (`/samba/auth`, `/samba/public`).
- ▶ Log in using `smbclient`, using various users, and put files into the shares, where possible

7.5 Primary domain controller

A primary domain controller acts as an authentication server for all windows workstations in its domain. Users authenticate to the PDC when they log in to their workstation. Once authenticated, they have access to all resources in the domain, be it on their local workstation, on the PDC, or on other windows servers participating in the domain.

To set up a primary domain controller, you need to

- ▶ add some attributes to the global sections
- ▶ define a `netlogon` share
- ▶ define a `homes` share, which will contain Windows' users home directory
- ▶ set up a place where roaming profiles are stored

7.5.1 Global settings

| Name | Description |
|---|--|
| <code>workgroup</code> | name of the domain |
| <code>domain logons</code> | <code>yes</code> |
| <code>wins support</code> | <code>yes</code> in most cases, unless you've defined several domains which share a same LAN |
| <code>add machine script</code> | Script to handle the joining of workstations to the domain: <pre>\hbox{\small\texttt{add machine script=/usr/sbin/useradd -d / -G " -g 100 -s logon drive logon home logon path profile location for Windows NT/2000/XP³</pre> |
| Drive letter for home directory (example: H:) | <code>logon home</code> |
| profile location for Windows 95/98 ² | <code>logon path</code> profile location for Windows |

Notes:

- ▶ Unlike with earlier Samba versions, the drive letter (`logon drive`), profile directory (`logon path`), and others should **not** be enclosed in quotes
- ▶ The purpose of the `add machine script` is to create the Unix accounts that back the machine accounts which are created when a workstation joins the domain. Care must be taken that those cannot be abused for interactive logins; that's why we set the login share to `/bin/false`.
- ▶ `logon home` and `logon path` are interpreted by the Windows workstation (after substitution of samba variables), and should refer to an existing share. Example: `\\%L%\%U\profile`. After substitutions of samba variables by the server, this will be `\\server\user\profile`, which is then interpreted by the workstation to mean the `profile` directory in the user's home share.

7.5.2 Homes share

The homes share represents the user's home directories. Each user will "see" a share named after himself, containing his own home directory.

```
[homes]
```

```
comment = Home Directories
browseable = no
```

```
# By default, the home directories are exported read-only. Change next
# parameter to 'yes' if you want to be able to write to them.
writable = yes
```

7 File services

```
# File creation mask is set to 0700 for security reasons. If you want to
# create files with group=rw permissions, set next parameter to 0775.
    create mask = 0700

# Directory creation mask is set to 0700 for security reasons. If you
# want to create dirs. with group=rw permissions, set next parameter
# to 0775.
    directory mask = 0700
```

7.5.3 Roaming Profiles

On login the user's is copied from the location identified by `logon path` to the local workstation.

On logout, it is copied back to the server.

On first login, when the user does not yet have a profile on the server, his profile gets initialized from the "Default User"'s profile.

7.5.4 Netlogon share

The main purpose of the netlogon share is as a location for the startup script (identified by the `logon script` parameter), which is executed on the client workstation on login

7.5.5 Adding a workstation to the domain

When joining a workstation to the domain, you need to supply a user name and a password on the server, who has the appropriate privileges.

One such user is `root`; however in Debian, `root` is marked as `invalid user` in `smb.conf`. In order to enable him, you need to comment out the following line, if present:

```
invalid users = root
```

After having made sure that the above line is gone, `root` can now add new workstations to the domain.

In many circumstances, however, it may not be desirable to hand out the root password of the server to the people doing the maintenance on the clients. In such cases, you may set up another user in such a way that he is entitled to add machines to the domain.

After having created this user using `useradd` and then `smbpasswd -a`), you declare him to be `admin user` on the reserved `IPC$` share.

```
[IPC$]
    admin users = winjoin lenin bigmouse
    path = /ipc
```

The IPC share is a reserved file share which is used for administrative communications among windows machines. Windows workstations log in to this share for numerous tasks, including joining the domain.

The `admin users` parameter defines a space-separated list of users who will enjoy root privileges when connecting to this share.

Caution: In addition to its special meaning, the `IPC$` share may contain files just like any other shares, and the admin users have full privileges on those files. Therefore it is important to point it to a directory which contains nothing of value (create an empty directory `/ipc` just for this purpose).

After having set up the `IPC$` share in this way, the user named `winjoin`, `lenin` and `bigmouse` are now entitled to add machines to the domain.

7.5.6 Setting up a “Domain administrator”

“Domain administrators” are users, defined on the PDC, which have administrative privileges on the workstations. They do not, however, have any particular privileges on the server itself.

In order to define domain administrators, you need to:

- ▶ Create or chose a Unix Group which will hold the domain administrators:

```
groupadd domadm
```

- ▶ Define this group as domain administrators:

```
net groupmap modify ntgroup="Domain Admins" unixgroup=domadm
```

- ▶ add users to this group (by editing `/etc/group`. These users now enjoy administrative privileges on the workstations.

Notes:

- ▶ The `net groupmap` database may get corrupted, especially when samba’s SID changes due to re-installation. In such case execute the following command: `net groupmap cleanup`, and try again.
- ▶ Use `net groupmap add ...` if the Windows group does not yet exist, and `net groupmap modify ...` if it does exist, or else the command will happily create two groups with different SIDs!

7.5.7 Example

```
[global]
## Browsing/Identification ###

# Change this to the workgroup/NT-domain name your Samba server will part of
```

7 File services

```
workgroup = test47
domain logons = yes
security = user
encrypt passwords = yes
add machine script = /usr/sbin/useradd -d / -G '' -g 100 -s /bin/false %u

printing = cups
cups options = "raw,media=a4"
load printers = yes
username map = /etc/samba/user.map
...
# Windows Internet Name Serving Support Section:
# WINS Support - Tells the NMBD component of Samba to enable its WINS Server
wins support = yes
...
[homes]
comment = Home Directories
browseable = no
# By default, the home directories are exported read-only. Change next
# parameter to 'yes' if you want to be able to write to them.
writable = yes
```

7.6 Password synchronization

Due to the peculiar way how Windows workstations authenticate to servers and PDCs, the Windows password record format (as stored in `/etc/samba/smbpasswd` is fundamentally incompatible with Unix' password records (as stored in `/etc/shadow`).

By default, these passwords are independent of each other: if the end user changes his Unix password, his Windows password is unaffected, and vice-versa. However, this is rather confusing to the end users, and this is where password synchronization steps in.

With password synchronization, the `smbpasswd` utility also changes the Unix password, and vice versa.

7.6.1 Unix password follows Samba

In order to make the Unix password follow the samba password, two steps are needed:

configure samba : Add (or uncomment) the following lines to `/etc/samba/smb.conf`:

```
unix password change = yes
pam password change = yes
```

configure pam : Add the following line to `/etc/pam.d/samba`:

```
@include common-password
```

Note: from yesterday's LDAP presentation, Unix authentication is still configured to use LDAP. Disable this temporarily by commenting out the `password sufficient pam_ldap.so` line in `/etc/pam.d/common-password`, or by only using users that do not exist in LDAP for our samba tests.

You may test password change:

```
smbpasswd -r server -U user
```

This command performs the samba password change the same way as a windows workstation would. Change the samba password of user, and then check (by logging in via `ssh`, for instance), that the Unix password has been changed as well.

7.6.2 Samba password follows Unix

In principle, a similar method should allow to do the synchronization in the other direction, by appending the following line to `/etc/pam.d/common-password`:

```
password sufficient pam_smbpass.so nullok try_first_pass use_authok
```

However, unlike other distributions, such as SuSE, Debian does not ship the `pam_smbpass` module with its samba.

To solve this, you may either:

- ▶ compile samba from source, and enable the feature:

```
./configure --with-pam --with-pam_smbpass
```

- ▶ simply symlink `/usr/bin/passwd` to `/usr/bin/smbpasswd`

7.7 Access control

In addition to the normal Unix file permissions, samba allows very fine-grained access control to shares.

This section describes how to control access by user, by workstation IP, and how to control the mapping of samba users to Unix users.

7.7.1 Access control by user

The following settings define which users may access a share:

| Name | Description |
|----------------------------|---|
| <code>valid users</code> | Users who may connect to this share |
| <code>invalid users</code> | Users who may not connect to this share |

7 File services

Both lists may contain individual users or groups. If a user ends up in both lists at once, **invalid users** takes precedence.

The following settings define which users may access a share:

| Name | Description |
|-------------------------|---------------------------------------|
| <code>write list</code> | Users who may write to this share |
| <code>read list</code> | Users who may not write to this share |

Both lists may contain individual users or groups. If a user ends up in both lists at once, **write list** takes precedence, i.e. users that are in both lists at once may write.

The **admin users** setting defines a list of users who are granted administrator access to the share (i.e. they perform all operations on the share as root).

7.7.2 Access control by IP

The following settings define which IP addresses may access a share:

| Name | Description |
|--------------------------|----------------------------------|
| <code>hosts deny</code> | IP addresses who may not connect |
| <code>hosts allow</code> | IP addresses who may connect. |

Both lists may contain individual machines or subnets (IP/netmask). If a machine happens to be in both lists at once, **allow** takes precedence.

7.7.3 Unix rights granted to share users

The following settings define which Unix rights the Samba users get:

| Name | Description |
|---------------------|--|
| force user | If this is set, all valid users connecting to the share act as this Unix user |
| force group | If this is set, all valid users connecting to the share act as belonging to this Unix group |
| create mask | ‘maximal’ set of permissions set on newly created files. If client who creates a file asks to grant more permissions than specified in mask, the additional permission bits are silently ignored. For instance, if the mask does not include the <i>world writable</i> bit, samba will not create any world writable files, even if client asks it to. |
| directory mask | same create mask, but for new directories, rather than files |
| force create mode | ‘minimal’ set of permissions set on newly created files. If client who creates a file asks to grant less permissions than specified in mask, the missing permission bits are set anyways. For instance, if the mode includes the <i>group readable</i> bit, samba will make files group readable, even if client didn’t ask for group readable files. |
| directory mask | same as force create mode, but for new directories, rather than files |
| force security mode | same as mode, but applies to permission bit changes (chmod), rather than new object creation. There is a force security mode, a security mask, a force directory security mode and a directory security mask (minimal/maximal bit masks, applicable to directories or plain files) |
| dos filemode | The default behavior in Samba is to provide UNIX-like behavior where only the owner of a file/directory is able to change the permissions on it. However, this behavior is often confusing to DOS/Windows users. Enabling this parameter allows a user who has write access to the file (by whatever means) to modify the per- missions on it. |

7.8 Samba variables

Often it is interesting to make values of samba configuration parameters dependant on the environment, such as properties of the client or user connecting to the service. This can be done using *samba variables*. Samba variables start with a percent sign (%) followed by a letter.

| Variable | Description |
|----------|---|
| %U | User name who connected to the share. This is the user name “requested”, by the client, i.e. before it is changed by username map, force user or admin users. |
| %u | User name assigned by samba (after taking into account any remapping performed by username map, force user and admin user) |
| %G | Primary Unix group of %U |
| %g | Forced group (if set), or primary Unix group of %u if there is no forced group not |
| %H | Unix home directory of %u |
| %m | Net BIOS name of client workstation |
| %I | IP address of client workstation |
| %a | Windows variant of client (one of WfWg, Win95, WinNT, Win2k, WinXP. This variable is particularly useful to use different profile directories for different windows versions (there may be some issues when WinNT and Win2k use the same profile, so we better keep them separate). |
| %L | Net BIOS name of server |

7.9 Using samba with an LDAP backend

7.9.1 Motivation

LDAP is useful if the user database

- ▶ is huge
- ▶ changes frequently
- ▶ needs to be shared among many hosts (NIS clients, other Samba servers, ...)

LDAP also allows to specify some settings per user, which would otherwise be global:

- ▶ profile path
- ▶ startup script

7.9.2 Setting up slapd server for samba

The following steps need to be performed on the `slapd` configuration to support samba

Schema Get the `samba.schema` from the following location, and put it into `/etc/ldap/schema/samba.schema`.

```
http://samba.org/~jerry/patches/post-3.0.6/samba.schema
```

Then declare it in `slapd.conf` by adding the following line:

```
include      /etc/ldap/schema/samba.schema
```

Object tree : Make sure you set up an object tree such as the following, to support samba:

```
com---tux-industries---belgium----People
    |
    |--Group
    |
    |--Samba---Idmap
    |
    |--Samba---Machine
```

This can be done using `ldapadd` or using a graphical tool such as `gq`

Security : As Windows' password encryption scheme is not secure, special care must be taken to protect the password attributes used by samba:

```
access to attribute=sambaLMPassword
    by dn="cn=admin,dc=belgium,dc=tux-industries,dc=com" write
    by self write
    by * none
```

```
access to attribute=sambaNTPassword
    by dn="cn=admin,dc=belgium,dc=tux-industries,dc=com" write
    by self write
    by * none
```

7.9.3 Samba configuration

The following changes needed to be performed in `smb.conf` to instruct samba to use LDAP:

7 File services

| Name | Description |
|---------------------|---|
| passdb backend | Set this to the URL of your Samba server: ldapsam:ldap://localhost. N.B. You should remove any previously existing passdb backend lines if there are any. |
| idmap backend | URL of samba server used for Idmapping. This is mostly needed if you import accounts from other domains. Example: ldap:ldap://localhost |
| ldap admin dn | DN (Ldap user) used for accessing the repository: cn=admin,dc=belgium,dc=tux-industries,dc=com |
| ldap ssl | Use SSL to connect to LDAP: on |
| ldap suffix | Base path under which all Samba-relevant objects are stored. This is an absolute path. Example: dc=belgium,dc=tux-industries,dc=com |
| ldap user suffix | Path where user account objects are stored. This is interpreted relatively to ldap suffix. Example: ou=People |
| ldap group suffix | Path where group and groupmap objects are stored (such as domadm). This is interpreted relatively to ldap suffix. Example: ou=Group |
| ldap idmap suffix | Path where idmap objects are stored. This is interpreted relatively to ldap suffix. Example: ou=Idmap,ou=Samba |
| ldap machine suffix | Path where machine account objects (workstations that are member of the domain) are stored. This is interpreted relatively to ldap suffix. Example: ou=Machine,ou=Samba |
| ldap filter | Filter used to locate entries by user name. Example: (cn=%u) |
| ldap passwd sync | Set this to yes for synchronizing LDAP password with windows passwords (the LDAP password is used for Unix authentication with LDAP) |
| unix passwd sync | In order for ldap password synchronization to work correctly, unix passwd sync, which is used for a non-LDAP authentication DB, needs to be switched off. Remove any line that sets it to yes |

Example:

```
passdb backend = ldapsam:ldap://localhost

idmap backend = ldap:ldap://localhost
ldap admin dn = cn=admin,dc=belgium,dc=tux-industries,dc=com
ldap ssl = on

ldap suffix = dc=belgium,dc=tux-industries,dc=com
ldap user suffix = ou=People
```

```
ldap group suffix = ou=Group
ldap idmap suffix = ou=Idmap,ou=Samba
ldap machine suffix = ou=Machine,ou=Samba
```

```
ldap filter = (cn=%u)
```

```
ldap passwd sync = yes
unix passwd sync = no
```

After having performed these changes in `smb.conf`, you still need to tell samba the password for the `ldap admin dn`. This can be done using the following command:

```
smbpasswd -w lxd2005
```

Yes, the password shows up on the command line, and no, this is not very secure! Let's hope that the samba team will address this issue in one of the next versions...

7.9.4 List of LDAP attributes relevant to Samba

| LDAP attribute | description |
|-----------------------------------|---|
| <code>objectClass</code> | Samba user object should have classes <code>sambaSamAccount</code> and <code>account</code> |
| <code>uid</code> | user name, equivalent to <code>cn</code> |
| <code>sambaSID</code> | SID of samba user (usually constructed by concatenating the SID of the server as obtained with <code>net getlocalsid</code> with the user's Unix id) |
| <code>sambaLMPassword</code> | LanManager password hash |
| <code>sambaNTPassword</code> | NT password hash |
| <code>sambaLogonScript</code> | script to be invoked at logon time (equivalent to <code>logon script</code> parameter of <code>smb.conf</code>). If empty, default from <code>smb.conf</code> is used. |
| <code>sambaHomeDrive</code> | drive letter of home share (equivalent to <code>logon drive</code> parameter of <code>smb.conf</code>). If empty, default from <code>smb.conf</code> is used. Note: this string must contain the colon following the drive letter! |
| <code>sambaProfilePath</code> | Windows Path where 95/98 profile is stored (logon home) |
| <code>sambaHomePath</code> | Windows Path where NT/2000/XP profile is stored (logon path) |
| <code>sambaUserWorkstation</code> | comma-separated list of workstations from where user may log on |

The easiest way to add a samba user is to use `smbpasswd -a`. This will convert the existing LDAP object for the Unix user into an object that is also suitable for samba, by adding the

7 File services

needed object class and attributes. The optional attributes (`sambaLogonScript`, etc.) may then be added by `ldapmodify` or `gq`.

7.9.5 Samba groupmap object

Groupmap objects (used for setting up domain administrators) are also stored in LDAP:

| LDAP attribute | description |
|-----------------------------|---|
| <code>objectClass</code> | <code>top, posixGroup, sambaGroupMapping</code> |
| <code>cn</code> | group's Windows name |
| <code>sambaSID</code> | group's SID |
| <code>gidNumber</code> | group's Unix group id. |
| <code>sambaGroupType</code> | must be 2 |

The easiest way to create these entries is to use `net groupmap modify` or `net groupmap add`. With LDAP, the groupmap does not automatically contain the default users (`Domain Admins`, `Domain Guests` etc.), so you need to create these explicitly (use `add`, rather than `modify`), and you need to supply the SID (machine sid followed by `-512`).

```
bruxelles:~# net getlocalsid
SID for domain BRUXELLES is: S-1-5-21-1180513021-3148254490-3206183951
bruxelles:~# net groupmap add ntgroup="Domain Admins" unixgroup=domadm \
  sid=S-1-5-21-1180513021-3148254490-3206183951-512
Successfully added group Domain Admins to the mapping db
bruxelles:~#
```

7.10 Miscellaneous Gimmicks

7.10.1 User monitoring

`smbstatus` : The `smbstatus` command displays the currently logged in users, as well as the shares, locks and files that they have currently open.

`root preexec` : The `root preexec` and `root postexec` share parameters specify a program that is executed whenever the share is mapped and/or unmapped. It can be used to propagate samba login/logout activity to Unix's last facility:

```
root preexec = /usr/X11R6/bin/sessreg -l %m -h %M -a %u
root postexec = /usr/X11R6/bin/sessreg -l %m -h %M -d %u
```

7.10.2 Time synchronization

The following line, in the global section, enables the samba server to act as a timeserver for its workstations:

```
time server = yes
```

To make use of this feature, the client workstation needs to execute the following command (for instance, from its startup script):

```
net time \\bruxelles /set
```

7.10.3 Hiding files

hide The following hides the files with the named extensions (slash-separated list). They can still be accessed by their name, but will not show up in directories (their Dos H bit is set)

```
hide files = *.exe/*.scr
```

veto The following makes the files with the named extensions (slash-separated list) completely inaccessible to samba. These files cannot be accessed, even if the user knows their name

```
veto files = *.exe/*.scr
```

7.10.4 Included configuration files

It is possible for `smb.conf` to refer to other configuration files. This may be useful for better organizing the samba configuration, and also for making some configuration aspects dependant on samba variables.

complete override If the named file exists, the current configuration file is overridden (i.e. all settings read from the current file, except location of new file, are forgotten), and new file is read instead. If the named file does not exist, the settings from the current file are retained

```
configuration file = /etc/samba/lib/smb.conf.%m
```

include / merge

The new file is read, and its setting merged with those read from the current file:

```
include = /etc/samba/lib/smb.conf.%m
```

8 Print services

by Till Kamppeter, linuxprinting.org, Mandrakesoft till.kamppeter at gmx.net

8.1 Setting up a Print Queue with CUPS

8.1.1 Connecting your printer

Connect your printer to the parallel port or USB. then restart the CUPS daemon with:

```
killall -HUP cupsd
```

and check with

```
lpinfo -v
```

whether CUPS has recognized your printer. You should get something like

```
... direct parallel:/dev/lp0 ... direct usb://hp/dj450?serial=SG31K210C23S ...
```

If this is not the case check whether the printer is correctly recognized. On USB use the "lsusb" command:

```
[root@majax root]# lsusb
Bus 001 Device 016: ID 0924:3d53 Xerox Bus
001 Device 014: ID 03f0:0512 Hewlett-Packard
Bus 001 Device 007: ID 03f0:1e11 Hewlett-Packard PSC-950
Bus 001 Device 005: ID 04b8:0112 Seiko Epson Corp. Perfection 2450
Bus 001 Device 003: ID 0451:2077 Texas Instruments, Inc. TUSB2077 Hub
Bus 001 Device 001: ID 0000:0000
[root@majax root]#
```

Your printer should lead to an entry here, also in the `/proc/bus/usb/devices` file it should appear. Check with "lsmod" whether the correct kernel modules (kernel 2.4.x: "printer", kernel 2.6.x: "usbulp") are loaded and load them with "modprobe <module>" if needed.

For parallel printers check with: `/proc/sys/dev/parport/parport?/autoprobe*`

```
[till@huxley till]$ cat /proc/sys/dev/parport/parport*/autoprobe*  
CLASS:PRINTER;  
MODEL:dj450;  
MANUFACTURER:hp;  
DESCRIPTION:hp dj450;  
COMMAND SET:MLC,PCL,PML,BIDI-ECP,ECP18,DW-PCL,DYN,DESKJET;  
[till@huxley till]$
```

If nothing is listed, reload the relevant kernel modules

```
modprobe -r lp parport_pc parport  
modprobe lp
```

For USB printers the `/dev/` file system should contain

```
crw-rw---- 1 lp sys 180, 0 Jan 4 19:34 /dev/usb/lp0  
crw-rw 1 lp sys 180, 1 Jan 4 19:34 /dev/usb/lp1 ...
```

for parallel

```
crw-rw---- 1 lp sys 6, 0 Jan 13 16:34 /dev/lp0 ...
```

8.2 Your distribution's printer setup tool

First, try to go the way your distribution does. Most printers can simply be set up by the tools coming with the distributions: "printerdrake" (or "mcc") on Mandrakelinux, "yast2" on SuSE, or "printconf-gui" on Red Hat or Fedora. They lead you step-by-step to the configuration of your printers, they auto-detect new printers, suggest the best driver, and they are even sometimes able to fully automatically set up all locally printers without any user interaction.

8.3 General printer setup with CUPS

CUPS itself has two possibilities to add new print queues. The easier way is the built-in web interface of the CUPS system. You reach it with any web browser accessing the address:

```
http://localhost:631/
```

Choose "Do Administration Tasks", enter "root" and the root password in the login/password window, and then choose "Add Printer". Then you will be led step-by-step through the set up of the printer.

The name of the print queue should not contain other characters than letters, numbers, or underscores and not more than 12 characters. This assures compatibility with non-CUPS clients, especially Windows, and makes it easier to access the printer via the command line. "Location"

and "Description" need not to be filled in, they are only comments, but filling them in makes it easier for users to find the most suitable printer in a network.

After clicking "Continue" you are asked for the "Device" for your queue. Choose the appropriate parallel or USB port. The entry should contain the model name of your printer.

On the next two pages choose manufacturer and model. If your model is not listed, follow the steps in the next sections.

If you come to the step "Printer ... has been added successfully" your queue is set up, click on the printer's name to get to the printer's administration page. Now click on "Configure Printer" and choose the default settings for it. Make sure that the default paper size is A4 when you are outside the US or Canada. Click on "Print Test Page" to check whether the printer really works.

Alternatively, you can set up a printer by the command line with the "lpadmin" command. See "man lpadmin" for more info.

NOTE:

- ▶ On SuSE systems another authentication method is used ("Digest"). To use the web interface, you must give a digest password to root at first, using the command "lppasswd":

```
lppasswd -a -g sys root
```

and enter a password (not necessarily the normal root password. This password has to be used for the web interface (or any other CUPS frontend) then.

- ▶ In certain error cases a print queue can get disabled and jobs stay waiting in the queue. To enable it, you need to run the command "enable" of CUPS:

```
/usr/bin/enable <queue name>
```

to re-enable the printer or click the green "Start Printer" button on the printer's page in the web interface.

- ▶ A very good graphical CUPS administration interface is the KDE Printing Manager. On every machine with installed KDE you find it in the KDE Control Center under "Peripherals" -> "Printers". Take care that at the bottom of the Window CUPS is chosen as your printing system. The "Administrator Mode" gives you the possibility to do several administration tasks without entering the root login and password again and again. Operation should be intuitive, right-click on the printer queue entries and on the free area around them to get menus with all functions. On the web you will find information on

```
http://printing.kde.org
```

Further info you get in the CUPS documentation on

```
http://localhost:631/documentation.html
```

especially:

```
http://localhost:631/sam.html
```

8.4 PPDs for PostScript printers

PostScript is a platform-independent page description language which is used by many mid-range and high-end laser printers and even by some high-end inkjets. Adobe has not only created PostScript, but also a way how to describe the properties and special commands of the different PostScript printers: The PPD (Postscript Printer Description) files, which also CUPS uses to describe printers. All this is well documented by Adobe:

<http://partners.adobe.com/asn/tech/ps/specifications.jsp>

PostScript printers are always shipped with appropriate PPD files. And these files describe their properties exactly. The files are also used by the drivers for Windows and Mac OS. Using these files with CUPS makes the full functionality of the printer (trays, resolutions, staplers, ...) available and therefore this is highly recommended.

To use a PPD simply copy it to `/usr/share/cups/model`, make it readable for everyone, let CUPS register doing `"killall -HUP cupsd"`, and it will appear as a printer model in the web interface of CUPS. Set up your print queue as shown above then.

You find the PPDs either on the driver CDs which come with printer (but unfortunately often compressed in a way that one cannot uncompress them with free software), on installed Windows boxes, on the manufacturer's web site, at Adobe (mainly older models):

<http://www.adobe.com/products/printerdrivers/winppd.html>

or on linuxprinting.org (these files are released as free software by the manufacturers):

<http://www.linuxprinting.org/download/PPD/>

If there are PPDs for different Windows and Mac OS systems, the NT4 or Mac OS X versions should work best. More info about PPDs you can find on:

<http://www.linuxprinting.org/ppd-doc.html>

Check the integrity of the PPD files with the `"cupstestppd"` utility. Often the output helps you fixing small flaws by editing the PPD file. CUPS only accept PPD files which pass the test.

8.5 Non-PostScript printers, GhostScript, and Foomatic

Under Unix and Linux applications usually produce PostScript when one sends a document to a printer and all printers are treated as PostScript printers. If a printer does not understand PostScript by itself the software PostScript interpreter GhostScript is used to translate the incoming PostScript into the printer's own language (PCL, ESC/P 2, ... or something proprietary).

To code which generates the printer's native language is the so-called printer driver. There are more or less sophisticated drivers for many printers and languages.

There are different types of drivers:

► **GhostScript built-in**

This driver concept is the oldest one. Drivers are pieces of code compiled into the GhostScript executable. One has to rebuild GhostScript to add a driver and so adding drivers for new printers is not trivial for most users. It is not recommended to create new drivers of this type. To determine which drivers are available, do "gs -h" and see the "Devices" section of the output. The current CVS of ESP GhostScript 7.07 contains all known drivers of this type.

► **Filter**

Filters are separate executables which convert a printer-independent raster image format produced by GhostScript (PNM, PPM, ...) into the printer's native language. They appeared because some people wanted to create a printer driver quickly without needing to study the internals of GhostScript. The advantage is that you do not need to patch and compile GhostScript to install them. So they are very easy to install. To find out whether the driver you need is already installed, check whether the driver's executable is in the execution path (Use "which <driver name>").

► **CUPS Raster**

CUPS raster drivers are a special form filter type drivers. The concept is developed as a part of CUPS to make driver installation easy. CUPS calls GhostScript to translate the PostScript input into the CUPS raster format and an additional CUPS filter to translate the CUPS raster format into the printer's language. These drivers come always with PPDs which do not only contain options and printer properties but also instruct CUPS to call the correct CUPS filter. Which of these drivers are installed you see by the PPD files in /usr/share/cups/model (and its subdirectories) and by the printer models showing up in the lists when setting up a print queue with the web interface of CUPS (entries contain "CUPS").

► **IJS plug-in**

Like filters IJS plug-ins are also separate executables and so they also do not require GhostScript to be patched and recompiled to add support for a new printer. But in contrary to filters IJS plug-ins communicate bi-directionally with GhostScript and so GhostScript can ask them for certain printer properties and adapt its rendering appropriately. The executables have to be in the execution path, like filters. They usually contain "ijs" somewhere in their names and some of them can be called with the "-h" command line option to get version information.

► **Uniprint**

This method is not common any more. GhostScript has a built-in driver named "uniprint" which is a universal raster driver. One can supply all commands which have to be sent to a printer to do raster printing on the GhostScript command line or by a file. For some printers there are several such files with the extension ".upp" in the /usr/share/ghostscript/<version>/lib/ directory.

As the drivers come from many different sources (often students who simply needed their printers for their work) they are of so many different types and are also invoked by very different and cumbersome command lines. There are also usually no PPD files supplied with them to easily use them with CUPS.

Foomatic is a layer between the GhostScript/driver combo and the printing system. It provides a PPD file for every valid printer/driver pair and a universal filter, which can be easily integrated into the well-known printing systems (CUPS, LPRng, LPD, PPR, PDQ) or be used stand-alone for direct, spooler-less printing.

The filter, foomatic-rip, is then called by the printing system to convert the job data from PostScript to the printer's language. It reads in the PPD file which contains all information about the driver and how it is used, the command line prototype, all user-settable options and what has to be inserted into the command line or into the job data to execute the users's choices. With this information and the user-supplied settings foomatic-rip builds the correct GhostScript command line and inserts PostScript or PJI commands into the job data stream. It finally executes GhostScript using the command line and pipes the PostScript data through it.

Using Foomatic with CUPS is easy. You look up your printer in the database on linuxprinting.org, choose a driver and then download the appropriate PPD file. Put the PPD file into `/usr/share/cups/model` and make it readable for everyone.

If your printer is not listed on linuxprinting.org, you could either try a driver for one of its predecessor models or you can check in the manual whether it uses a standard language (PostScript, PCL,ESC/P, ...) and use one of the "Generic" printer entries in the database.

Verify that the appropriate driver is installed. See the driver type descriptions above to learn how to determine whether a driver is installed. Install the driver if needed.

Download also foomatic-rip and foomatic-gswrapper, put them into `/usr/bin` and make them readable and executable for everyone. Create a link to make foomatic-rip visible for CUPS:

```
ln -s /usr/bin/foomatic-rip /usr/lib/cups/filter/
```

Now set up a print queue with CUPS as described above. In the web interface choose an entry which contains your printer's model name and "Foomatic".

NOTE:

- ▶ Do not use Foomatic PPDs for PostScript when you have a manufacturer-supplied PPD. The Foomatic PPDs for PostScript printers are generic and support only a few options, the manufacturer-supplied PPDs give access to the full functionality of your printer.

Some links:

linuxprinting.org printer database:

```
http://www.linuxprinting.org/printer\_list.cgi
```

Instructions for setting up printers with CUPS and handling PPDs:

```
http://www.linuxprinting.org/cups-doc.html  
http://www.linuxprinting.org/ppd-doc.html
```

8.6 Multi-function devices from HP and Epson

On multi-function devices you probably want use not only the printer but also the scanner and memory card reader. Using fax facilities is not supported in any device with free software, unfortunately.

On Epson devices printer, scanner, and card reader behave as independent USB devices. The card reader appears as a USB mass-storage device. The scanner is usually automatically discovered by SANE, via the "epson" backend. See

<http://www.sane-project.org/>

for whether and how the scanner in your device is supported.

On HP it is not so easy, but they are very well supported, too. Some distros set them up with HPOJ for you. You can also scan with SANE then. The card readers of some newer models are USB mass-storage, on the other models they work with HPOJ, but they are so slow that buying a USB-2.0 card reader for around 10 EUR is recommended.

If your distro does not set up HPOJ for you, or if you want to have the newest software, get HPLIP from

<http://hpinkjet.sourceforge.net/>

to be able to scan with SANE. Card reader support is slow here, too. But the newest devices are supported and you get printer maintenance, as nozzle cleaning or head alignment. Note that HPOJ is not maintained any more. So switch to HPLIP in case of problems.

Note that we are in a server tutorial: SANE does excellent network scanning via saned on the server and the "net" backend on the clients. And SANE-TWAIN on the Windows clients.

With non-HP/Epson multi-function devices you can be lucky to get printing to work.

8.7 Ethernet-Connected Printers

8.7.1 General

Ethernet-connected printers (or conventional printers connected to print boxes like HP JetDirect or so) are very convenient in networks, as they do not rely on a single server. In small, especially home networks every PC has its own queue to the printer and so no PC needs to keep running day and night to have the printer accessible. In big networks several redundant servers have queues to the printer. If all are running they form a cluster and share the workload, if one fails, the jobs are simply done by the others.

8.7.2 Assigning an IP address to a printer

On some more sophisticated devices you can enter the IP via the front panel, on others it has to be done by software running on a computer and accessing the printer or printbox through the network.

If you look into the manual of your printer or ethernet printing adapter, in many cases they simply tell you to install a Windows program from the supplied CDs to configure the ethernet interface. Do not think you have bought a ver expensive paperweight then, you can usually configure the IP of such a device with the "arp" (Address Resolution Protocol) utility, giving the command

```
arp -s <ip-address> <ethernet-address>
```

for example

```
arp -s 192.0.2.2 08:00:69:00:12:34
```

The ethernet address you find on the sticker on the device or by printing a self-test/configuration page with the device. Alternatively, you can install and run "rarpd" on your machine and enter all IP assignments in the `/etc/ethers` file like:

```
08:00:69:00:17:e3 myprinter.mydomain.com 08:00:69:00:12:34 192.0.2.2
```

Then the IP will be assigned whenever any device listed in the file is turned on. This way one has a central management for the IPs, there are no problems with devices not remembering their IPs when power-cycling, and if you have temporarily used a device at another place, it gets its IP back automatically.

Also bootp can be used to assign IP addresses to printers.

Check whether the printer accepted the IP address with

```
ping <ip-address>
```

8.7.3 Discovering the printer's network protocols

To find out which protocols a printer supports, simply look which ports are open on it. This is done with the "nmap" command. Do

```
nmap <ip-address>
```

for example

8 Print services

```
[root@majax root]# nmap 192.168.100.20
Starting nmap 3.78 ( http://www.insecure.org/nmap/ ) at 2005-01-13 20:56 EST
Interesting ports on printer.mandrakesoft.com (192.168.100.20):
(The 1657 ports scanned but not shown below are in state: closed)
PORT STATE SERVICE
21/tcp open ftp
23/tcp open telnet
80/tcp open http
280/tcp open http-mgmt
515/tcp open printer
631/tcp open ipp
9100/tcp open jetdirect MAC Address: 00:10:83:BB:B1:29 (Hewlett-packard Company)

Nmap run completed -- 1 IP address (1 host up) scanned in 12.671 seconds
[root@majax root]#
```

Now you can see by the ports which protocols are supported:

```
port protocol
-----
23 printer admin via telnet
80 printer admin via web
139 printing via SMB 515 printing via LPD 631 printing via IPP 9100 printing via TCP/Soc
```

If your printer has an admin interface via telnet (do "telnet <printer IP>") or web (use URL "http://<printer IP>/" in your browser) you can probably activate or deactivate the possible communication protocols of your printer. So open the desired protocol and close the others for example to avoid that a print quota system on your server will be surrounded.

See

```
http://localhost:631/sam.html#COMMON_NETWORK
```

for more information about setting up network printing devices.

8.7.4 Setting up a print queue

In principle, the print queues are set up the same way as for local printers, via the printer setup tool of your distribution, the CUPS web interface or the "lpadmin" command. The only difference is what has to be entered for the device to which the jobs should go.

CUPS does not auto-detect the presence of network-connected printers. You have to know its IP and its protocol, in some cases even more information. In the web interface you choose on the "Device" page which protocol should be used to access your printer, on the next page you have to enter the so-called device URI (Unified Resource Identifier) then. If you use "lpadmin" you have to supply the device URI on the command line ("-v" option). The choice of printer model and driver is exactly the same as for a USB or parallel printer.

Easiest to set up is the access through the TCP/Socket protocol (also known as AppSocket or JetDirect). One only needs the IP of the printer and the port number. The IP you have assigned to your device and the port number you get from the "nmap" output. Most common is 9100, for multi-port print boxes 9101, 9102, and 9103, but other port numbers greater than 1023 are possible. So use TCP/Socket whenever it is possible.

The device URI is

```
socket://<printer IP>:<port number>
```

Example for a printer with IP 192.158.100.20 and port 9100:

```
socket://192.158.100.20:9100
```

If your device does not provide access via TCP/Socket, the second choice is to try SMB (Windows NT protocol). This will work if the printer's port 139 is listed as open by "nmap". SMB is not supported by the CUPS software itself, but by Samba. It is enough to install the Samba client software coming with your distribution. You will not need running Samba daemons. Restart the CUPS daemon with

```
killall -HUP cupsd
```

after installing Samba. "lpinfo -v" should list "network smb" then.

The information which you need to supply is the printer's Ip and the printer share name. The share name varies between the different printer models and can be determined with the "smbclient" command:

```
smbclient -N -L <printer IP>
```

You should have at least one share of the type "Printer". The device URI looks as follows then:

```
smb://<printer-ip>/<share>
```

The third and less recommended way to print on a network printer is the LPD protocol. Its problem is that besides the IP you have assigned you need the name of the LPD queue on the network device and this you cannot discover by software running on your computer. You can either find the queue name in the manual (if you are lucky), guess common names as "LPD", "LPR", "ps", "PORT", "LPT", ... or these name with an added number, or find it on

```
http://localhost:631/sam.html#COMMON_NETWORK
```

The device URI is then

```
lpd://<printer-ip>/<queue-name>
```

where "queue-name" is the name of the LPD queue on your printer or print box.

More information you can find on

```
http://localhost:631/sam.html#PRINTING_OTHER
http://localhost:631/sam.html#COMMON_NETWORK
```

8.8 Networked Printing with CUPS

8.8.1 Basic CUPS configuration for remote printer auto-discovery

Sharing printers between computers running CUPS (Linux, Unix, Mac OS X) is very easy, when the correct CUPS configuration is used. Once having the correct configuration, you set up a print queue on the machine where the printer is attached to and the other machines will have the printer available automatically within the next 30 seconds. And if you move the printer to another box, no change has to be done on the other machines, they will discover the move by themselves.

For the automatic discovery of remote printers every machine with locally defined print queues (CUPS server) broadcasts its host name and the names of the queues into the network every 30 seconds. Every machine running a CUPS daemon listens to such broadcasts, collects this information, and treats these queues like its own local queues (CUPS client). So a CUPS client lists all remote CUPS queues it gets knowledge of with "lpstat -p" command. Jobs sent to these remote queues the CUPS client passes automatically to the CUPS server so that the server can print them. Also the user-settable options of remote CUPS queues can be accessed on the CUPS client. "lpoptions" and the graphical printing frontends like "kprinter" or "xpp" show all the options as defined in the PPD file on the CUPS server and the user can set them. The processing of the jobs is done always on the server, so neither drivers nor PPD files need to be installed on the CUPS clients. And nothing needs to be removed from the clients when a printer is removed from a server. A machine with a running CUPS daemon can be both a CUPS server and a CUPS client.

Do not think the broadcasts apply a high load to your networks. Only a few bytes are broadcasted every 30 seconds, not more than the host name and the queue names. All other info which you see about remote printers on a CUPS client is polled from the server on demand, only when an application requests it.

To get this working as easy as possible and with a basic security for small office or home networks edit the `/etc/cups/cupsd.conf` file as follows:

```
LogLevel info
TempDir /var/spool/cups/tmp
Port 631

BrowseAddress @LOCAL
BrowseDeny All
BrowseAllow 127.0.0.1
BrowseAllow @LOCAL BrowseOrder deny,allow

<Location />
Order Deny,Allow
Deny From All
Allow From 127.0.0.1
Allow From @LOCAL
</Location>
```

```

<Location /admin>
AuthType Basic
AuthClass System
Order Deny,Allow
Deny From All
Allow From 127.0.0.1
</Location>

```

Comment lines and most blank lines are not shown here. All other lines not shown here can be (or stay) commented out ("`#`" added at the beginning of the line).

NOTE:

- ▶ Do not forget to restart the CUPS daemon with "killall -HUP cupsd" whenever you edit anything in `/etc/cups/cupsd.conf` or in any file in `/etc/cups` or its subdirectories.

The "LogLevel" setting tells how much should be logged in `/var/log/cups/error_log` by default, leave it on "info" as long as you do not try to troubleshoot some problem. The "TempDir" can be different on your system. Do not modify or delete this line. 631 is the default port of CUPS (and IPP, the Internet Printing Protocol, the communication protocol CUPS is using), so do not change it.

The "Browse..." settings control the broadcasting of printer information between CUPS daemons.

"BrowseAdress" tells to which CUPS clients information about the queues on your machine is broadcasted. "@LOCAL" means all local networks, but not PPP, or dialed connections, so you printers will not get broadcasted into the internet and no costly dial-on-demand connections will be triggered. Yo can also specify an address range ("192.168.100.*") or several "BrowseAddress" lines with address ranges or even the IP addresses of single machines.

"BrowseDeny", "BrowseAllow", and "BrowseOrder" define from which CUPS servers broadcasted printers will be accepted and from where not. In our case we have a "BrowseOrder deny,allow" so the "BrowseDeny" are treated first and then the "BrowseAllow". Without any such line all broadcasted printers are accepted, the "BrowseDeny all" leads to no printer being accepted, but after that line the "BrowseAllow" lines are treated so the broadcasts from the IP 127.0.0.1 (the local machine) and from the local networks ("@LOCAL") are accepted again. The configuration of the example prevents queues appearing from faulty sites which broadcast into the internet. Using a more restrictive configuration you can get rid of long printer lists in your printing frontends so that you only see the printers you really need.

The "<Location ...> ... </Location>" blocks control the access to your machine by CUPS clients. Possible locations are:

```

Location Purpose
-----
/ The overall CUPS server
/admin Adding/modifying/removing print queues
/printers Printing on all queues

```

8 Print services

```
/printers/<queue name> Printing on a particular queue
/jobs Modifying
/removing jobs
```

The locations behave like a directory hierarchy, if a sublocation does not exist, the access rules of the parent location apply. For example if there are the queues "bwlaser", "colorlaser", and "inkjet" and the locations "/printers" and "/printers/inkjet", the rules of "/printers" are valid only for "bwlaser" and "colorlaser", for "inkjet" the rules of "/printers/inkjet" apply.

The "Deny From", "Allow From", and "Order" rules work the same way for client access permission as "BrowseDeny", "BrowseAllow", and "BrowseOrder" for accepting broadcasted printers. So in the example general access (Location "/") is allowed for the local machine and all clients on the local networks. This applies especially for printing on all print queues as there is no "/printers" location. Print queue manipulation is even only allowed from the local machine.

Access cannot only be restricted per-network or per-machine but also per-user. For user-wise access restrictions the "AuthType" and "AuthClass" rules are used.

"AuthType" can be "None" (no per-user access restriction, no login/password required), "Basic" (permitted users can access with their usual Unix login/password on the server), or "Digest" (CUPS has its own user accounts defined with the "lppasswd" command).

"AuthClass" can be "System", then only users of the system group (usually "sys") can access. It can also be "Basic", to give all Unix or Digest users on the server access or "Group" to give access to the users in the group specified by the "AuthGroupName" rule.

In our example everyone can print, as there are no per-user restrictions in the "/" location. But queue manipulation can only be done by the members of the Unix System group, usually this is only "root". Therefore you need to log in as "root" in the web interface to set up queues.

8.8.2 High availability

In CUPS it is also easy to set up high availability by having redundant printers or servers.

For example you can connect three (preferably equal) printers via USB to one CUPS server. You define a queue with the same configuration to each of these printers, for example with the names "laser1", "laser2", and "laser3". Now you set up a so-called printer class. This is a queue which does not point to a printer, but to a set of other queues. In this case you let it point to our three queues "laser1", "laser2", and "laser3". Let us call the class "laser". To define the class one uses again the web interface of CUPS, simply clicking the "Add Class" button on the Administration page. "lpadmin" also allows to define classes.

If a user would send a job to "laser2" the job has to wait until all other jobs on "laser2" are ready and "laser2" must be ready to print (turned on, paper and toner loaded). Otherwise the user's job waits in the queue, even if "laser1" and "laser3" are doing nothing.

If the user sends the job to the class "laser". The first printer which gets free will take the job. So if "laser1" is printing a 100-page job and "laser2" is out of paper, the user's one-page letter goes to "laser3" and so it comes out quickly.

The problem would be if the server fails. Then none of the printers is available and the users cannot print. So we connect each printer to another computer, which means that if one of the

computers fails, the other two printers are still available. But there is one problem: On which server do we define the class "laser"? If this server fails, the printers are all not available again.

The solution is a so called "implicit class", a class automatically defined by CUPS. Therefore we do not call the three queues on the three servers "laser1", "laser2", and "laser3". It is much simpler, we call them all "laser" and let them get broadcasted into our network. Now the clients receive broadcasts of three equally-named queues. For the users of the clients only one queue named "laser" appears. And this queue behaves as a class: the first of the three printers which gets free takes the job. So independent which of the three server/printer combos are working, if at least one is working the jobs sent to "laser" get printed. Also there is no single point of failure which defines the class "laser". "laser" is defined on every client and only if at least one of the three servers is working.

If you have an ethernet-connected printer you could define a queue for it on every client, so the clients are their own CUPS servers and do not depend on a single server. But imagine the number of print queues you see in graphical printing frontends when every queue has another name, and that in a network with only one printer. And imagine also the administration nightmare when the IP of the printer has to be changed.

So here you better choose three machines and define equally-named queues pointing to this one printer. You get an implicit class which will be the only one visible queue on the CUPS clients. And you do not need to rely on one server, you can print if at least one of the three is running.

Also combinations of this are possible: For example two servers and three ethernet printers, where on each of the two servers is exactly the same CUPS configuration: Three queues pointing to the three printers and a class putting together the three queues. The two classes on the two servers form an implicit class and so printing into this implicit class works with any arbitrary pair of server and printer.

8.9 Print quotas and accounting

CUPS has also basic page accounting and quota capabilities.

Every printed page is logged in the file `/var/log/cups/page_log`. So one can everytime read out this file and determine who printed how many pages.

In addition, one can restrict the amount of pages (or kBytes) which a user is allowed to print in a certain time frame. Such restrictions can be applied to the print queues with the "lpadmin" command.

```
lpadmin -p printer1 -o job-quota-period=604800 -o job-k-limit=1024
lpadmin -p printer2 -o job-quota-period=604800 -o job-page-limit=100
```

The first command means that within the "job-quota-period" (time always given in seconds, in this example we have one week) users can only print a maximum of 1024 kBytes (= 1 MByte) of data on the printer "printer1". The second command restricts printing on "printer2" to 100 pages per week. One can also give both "job-k-limit" and "job-page-limit" to one queue. Then both limits apply so the printer rejects jobs when the user already reaches one of the limits, either the 1 MByte or the 100 pages.

8 *Print services*

This is a very simple quota system: Quotas cannot be given per-user, so a certain user's quota cannot be raised independent of the other users, for example if the user pays his pages or gets a more printing-intensive job. Also counting of the pages is not very sophisticated. The pages are counted by the CUPS filters, simply by analyzing the PostScript data stream. This does not work with arbitrary PostScript data or with jobs already arriving in the printer's native language (for example from a Windows client). Another problems are paper jams or similar hardware problems of the printer. The pages do not get printed but were already counted by the filters.

So for more sophisticated accounting it is recommended to use add-on software which is specialized for this job. This software can limit printing per-user, can create bills for the users, use hardware page counting methods of laser printers, and even estimate the actual amount of toner or ink needed for a page sent to the printer by counting the pixels.

The most well-known free software packages are

PyKota (recommended):

```
http://www.librelogiciel.com/software/PyKota/
```

PrintBill:

```
http://ieee.uow.edu.au/~daniel/software/printbill/
```

8.9.1 LPD clients

If you have some older boxes, especially with commercial Unix systems, you have probably LPD on them and no CUPS. You do not need to install LPD or LPRng on your print servers to get print jobs from these boxes printed. CUPS has an interface to LPD clients, the so-called CUPS-LPD mini-daemon.

The CUPS-LPD mini-daemon maps the CUPS queues known to your server's CUPS daemon to the LPD protocol. The queues will have the same name for LPD as they have for CUPS.

To activate the CUPS-LPD mini-daemon, you have to get it invoked on demand by inetd or xinetd. For inetd add the line

```
printer stream tcp nowait lp /usr/lib/cups/daemon/cups-lpd cups-lpd
```

to `/etc/inetd.conf` and give a kick to the inetd with "killall -HUP inetd". For xinetd create a file named `/etc/xinetd.d/cups-lpd` containing

```
service printer
{
  socket_type = stream
  protocol = tcp
  wait = no
  user = lp
```

```

group = sys
server = /usr/lib/cups/daemon/cups-lpd
server_args = -o
document-format=application/octet-stream
passenv =
env = TMPDIR=/var/spool/cups/tmp
disable = no
}

```

and kick the xinetd daemon with "killall -HUP xinetd". From now on LPD clients can print on your queues the classical LPD way. Note that any access restrictions in `/etc/cups/cupsd.conf`, in `/etc/hosts.allow` and in `/etc/hosts.deny` are ignored. So every user and every machine can print vial LPD on your server.

8.10 Accessing Printers from Windows Clients via Samba

8.10.1 Sharing the CUPS queues to Windows clients

At first check whether the needed Samba server packages are installed and that your Samba version is linked against the CUPS library (should be the case in most modern Linux distributions). To check whether Samba is linked against the CUPS library, use the "ldd" command:

```

root# ldd `which smbd`
libssl.so.0.9.6 => /usr/lib/libssl.so.0.9.6 (0x4002d000)
libcrypto.so.0.9.6 => /usr/lib/libcrypto.so.0.9.6 (0x4005a000)
libcups.so.2 => /usr/lib/libcups.so.2 (0x40123000) [...]

```

If "libcups" is listed as shown in this example, your Samba is OK. If not, try to get CUPS-enabled packages of Samba for your distro or compile Samba from source. If you compile Samba from source, you need to have the header/development package of the CUPS library ("libcups-devel", "cupsys-dev", or similar) installed.

You also need to take care that the names of all your printers are not longer than 12 characters. If a queue name is longer, edit `/etc/cups/printers.conf` to modify the queue name and then activate the change with the usual "killall -HUP cupsd". CUPS has no on-board tool to rename print queues, but some other printer setup tools, like "printerdrake" in Mandrakelinux, have.

Now do the following modifications on `/etc/samba/smb.conf`:

```

[global]
load printers = yes
printing = cups
printcap name = cups

printer admin = @ntadmins

```

8 Print services

```
[printers]
comment = All Printers
path = /var/spool/samba
browseable = no
public = yes
guest ok = yes
writable = no
printable = yes
printer admin = root, @ntadmins

[print$]
comment = Printer Driver Download Area
path = /etc/samba/drivers
browseable = yes guest ok = yes
read only = yes
write list = @ntadmins, root
```

All lines not supposed to be modified are not shown. Do not touch that lines, otherwise you could loose other Samba services, like file sharing or so.

Restart Samba (usually `"/etc/init.d/smb restart"`) to get your changes active.

Create the "ntadmins" group in `/etc/group` and add users which should be allowed to to Samba printer administration. "root" is always allowed to do so and does not need to be added to this group.

If some of the directories referred to in the lines shown above does not exist, create it. Let the ownerships of these directories be root and the ntadmins group. Set permissions for everyone to read and execute and for user and group to write into the directories.

Create Samba accounts for "root" and every user in the "ntadmins" group with "smbpasswd":

```
root# smbpasswd -a root
New SMB password: secret
Retype new SMB password: secret
```

Samba has no access to the Unix passwords in `/etc/shadow`. Therefore Samba needs its own passwords. Use always the Samba passwords for authentication against Samba.

Most important in the `smb.conf` is `"printing = cups"`. This makes Samba use the functions of the CUPS library to access printers, instead of shell command lines defined in `/etc/samba/smb.conf`. So if your `smb.conf` contains lines like `"print command"`, `"lpq command"`, ... in the `"[global]"` or `"[printers]"` sections, they get ignored.

The `"[print$]"` section defines a file share where Windows printer drivers can be supplied so that Windows clients can download them automatically ("Point'n'Print").

The shown configuration makes all printers known to the local CUPS daemon being shared via Samba. This means they will all be detected in the network Neighborhood or in the Add Printer Wizard under Windows.

To really print with these printers under Windows one can use the Windows drivers which came with the printers and set up queues with them on each client. Or one uploads the drivers into the `print$` share on the Samba server (can be done with the Add Printer Wizard under Windows). This requires also uncommenting the "octet-stream" lines in the end of the `/etc/cups/mime.types` and `/etc/cups/mime.convs` files and then restarting CUPS, so that CUPS accepts already completely processed print data on regular print queues and does not try to filter it. For more about this, see the printing sections of the Samba HOWTO collection:

```
http://de.samba.org/samba/docs/man/Samba-HOWTO-Collection/printing.html
http://de.samba.org/samba/docs/man/Samba-HOWTO-Collection/CUPS-printing.html
```

This approach of printing has the disadvantage that once many different printer drivers are on each Windows client and this can compromise the stability of Windows. Another problem is that CUPS does not understand the job data which passes through it and so it cannot count the printed pages. This prevents print accounting from working correctly.

The better way is to let our server emulate PostScript printers, as it does for Unix applications. For that a PostScript driver for Windows and the PPD file of the CUPS queue is put up in the `print$` share and so Windows will use the PostScript driver for all printers and send PostScript, as a Unix client. This way every Windows client has only one printer driver installed and CUPS gets data on which it can do accounting and other types of processing.

8.10.2 Emulate PostScript printers

At first, you need to get a PostScript driver for Windows, from CUPS (recommended) for Windows NT/2000/XP or newer, from Microsoft for Windows NT/2000/XP or newer or from Adobe for Windows 95/98/Me. You can also use both the CUPS or Microsoft and the Adobe driver to support all Windows versions on your clients.

The CUPS driver you get in the download section of

```
http://www.cups.org/
```

the Adobe driver on

```
http://www.adobe.com/
```

and the Microsoft driver on a machine with Windows installed (NT/2000/XP or newer), in the `%WINDOWS%\SYSTEM32\SPOOL\DRIVERS\W32X86\3` directory.

Create a directory named `drivers/` in `/usr/share/cups` and put the following files of the PostScript drivers for Windows there:

```
Driver Files
CUPS: cupsui6.dll cupsdrv6.dll ps5ui.dll pscript.hlp pscript.ntf pscript5.dll
Microsoft: ps5ui.dll pscript.hlp pscript.ntf pscript5.dll
Adobe: ADFONTS.MFM ADOBEPS4.DRV ADOBEPS4.HLP ICONLIB.DLL PSMON.DLL
```

8 Print services

The CUPS driver is the recommended driver as it supports several CUPS-specific printing options.

Use "unzip" or "cabextract" to uncompress the Adobe driver package. If your distribution does not include cabextract, get it from here:

```
http://www.kyz.uklinux.net/cabextract.php3
```

Now run either

```
cupsaddsmb -U root -v -a
```

to set up the PostScript driver in print\$ for all printers known by the local CUPS daemon, or

```
cupsaddsmb -U root -v laser inkjet
```

to only set up the driver for the queues "laser" and "inkjet". If you get asked for a password, enter the Samba password for "root". Now you can set up print queues on your Windows clients without taking care of drivers. The Add Printer Wizard will download the PostScript driver automatically from your server. See also the man page of "cupsaddsmb".

8.10.3 More possibilities

There is excellent documentation about printing via Samba written by Kurt Pfeifle. It is part of the Samba HOWTO collection:

```
http://de.samba.org/samba/docs/man/Samba-HOWTO-Collection/printing.html  
http://de.samba.org/samba/docs/man/Samba-HOWTO-Collection/CUPS-printing.html
```

8.11 CUPS Troubleshooting

8.11.1 RTFM - Read The F... Manual!

Before trying to get support on mailing lists, newsgroups, or web forums, read the documentation of the software you are using. There is a lot of documentation for CUPS, Foomatic, and Samba, simply follow the links in this document. Most command line tools have also man pages.

8.11.2 Supply all important information

- ▶ Which distribution are you using? Mandrakelinux 10.1, Fedora Core 2, but NOT Linux 7.2.
- ▶ Which kernel? Do "uname -a" and you get the version number, for example 2.6.9.

- ▶ Which version of CUPS, Gimp-Print, HPIJS, Samba, ...
- ▶ Which exact command lines did you use? What was the output? Cut and paste from your terminal Window.
- ▶ What did you fill into the fields in the web interface of CUPS?

Supply also config files like `/etc/cups/printers.conf`, `/etc/cups/cupsd.conf`, `/etc/cups/ppd/*.ppd`, `/etc/samba/smb.conf`, ...

Strip away the comments and blank lines in the config files when posting them, for example with a command like this:

```
cat /etc/cups/cupsd.conf|grep -v ^#|grep [[:alnum:]] > cupsd.conf-stripped
```

Then it is much easier to find the important things. Run diagnostic commands like `"lsmod"`, `"lsusb -vvv"`, ...

Check the integrity of manufacturer-supplied PostScript PPD files with the `"cupstestppd"` utility. Often the output helps you fixing small flaws by editing the PPD file. CUPS only accept PPD files which pass the test. Unfortunately often printer manufacturers do not comply well with Adobe's standards.

8.11.3 Check device file permissions

Device files for printers (like `/dev/lp0`, `/dev/usb/lp0`, ...) must be readable and writable for the user `"root"` and the group `"sys"`.

8.11.4 Send print jobs with CUPS in debug mode

Edit `/etc/cups/cupsd.conf` switching the `"LogLevel"` to `"debug"`. Kick the CUPS daemon with `"killall -HUP cupsd"` and send a print job. Then have a look into `/var/log/cups/error_log` and look for error messages, GhostScript command lines, option settings. Post relevant parts of the file if you didn't find a solution for your problem.

8.11.5 Where to ask

And finally, after reading what is important when asking for help, the places where you can ask:

<http://www.cups.org/newsgroups.php>

The home of the CUPS-centered newsgroups and mailing lists.

<http://www.linuxprinting.org/forums.cgi>

8 *Print services*

The resource to look for answers if you have vendor- or model specific questions. There is a separate forum for all major manufacturers. here you can also post your questions concerning Foomatic.

`http://gimp-print.sourceforge.net/p_Mailing_Lists.php3`

This is the place of choice if you are suffering from difficulties with using Gimp-Print drivers.

NOTE:

- ▶ **DO NOT** send questions to the private mail addresses of people who answer questions on the lists regularly as long as they do not ask you explicitly to do so. Post always on the lists and always use "Reply to all" or "Reply to list" in your e-mail when answering. From every problem discussed and solved on the list everyone can learn. And everyone can read the discussion and help, not only a single person to whom you mail privately. And the more people read your questions, the quicker someone helps.

8.11.6 More about troubleshooting ...

... you can read in Kurt Pfeifle's HOWTO:

`http://www.cups.org/cups-help.htm`

Glossary

CMS Content Management System

FTP File Transfer Protocol

SMTP Simple Mail Transfer Protocol

URL Abbreviation of Uniform Resource Locator, the global address of documents and other resources on the World Wide Web.

WebDAV Web-based Distributed Authoring and Versioning

ZMI Zope Management Interface

8 *Print services*