Guide for Linux Internet Server Professionals

Networking/Security/Administation

Collective Work

Guide for Linux Internet Server Professionals : Networking/Security/Administation
by Collective Work

$Id: book.sgml,v 1.49 1999/08/10 09:17:41 om Exp $

Table of Contents

List of Examples

TOC

Preface

Pre-Test edition - 1999 - Work in progress :)

CVS + SGML DocBook are great tools !

TOC

I. Project Information

That is the Project-Howto, which is currently being written. If you find mistakes, errors or other flukes, please report them by sending a mail to one of the maintainers.

Chapter 1. Common issues

Contributed by Olivier, 25 January 1999.

Two Chapters in this book part :

- In the first one, you'll learn how to work with the SGML structure of our project: creating new parts/chapters, moving them, importing text, etc.
- Then, the second chapter can be used as a kind of SGML playground, with some easy to understand, commented, code examples.

Chapter 2. Basic Project Info

2.1. General Idea

This document should be an help to everybody who use or want to use linux as operating system for an internet server. It assumes you have a basic knowledge of linux/Unix and internet technology. Before you read further, you should first read your Linux Distribution Handbook, and the Network Administrators Guide written by Olaf Kirch.

A real-life example : ...

Some more ideas (taken from one of my ML posts, will be worked out later) : I'd like to have things in the book :

- some high quality info and links about all the server stuff
- the description of a Real Life Example (tm), from the first chapter to the last one., with so many details as possible. Let me try to explain what I mean exactely. In the first chapter, we set up some goals : for example, installing a perfect web hosting server with let's say 3 domains for web, mail, ftp, dns, etc. And then, in each chapter, we explain and show the steps how to install the the software on this virtual server : show some real qmail configuration/compilation screenshots/configfiles in the qmail chapter, the same for sendmail, apache, named, the usual beginners problems with cgi, etc. Then it could nearly be read like a roman :)
- For this Real Life example, there will be a kind of "test-server" available, where you will be able to install and test your programs without having to use your own production server. This server should be ready in the middle of september.

2.2. Book Structure

Let's assume we are in the `/book` directory of the CVS tree. Here are the important files to look at or to modify if you want to add a part or chapter.

2.2.1. Important files in `/book` directory

In `/book`, only the book parts and some other entities (urls, authors) are defined. All chapter information resides in the `/book/part-partname` directories.

- `book.sgml` : The most important file of all, which calls all the others : `book.ent`, `parts.ent`, and all the book parts.
- `book.ent` : currently, just contains <!ENTITY lisa-url SYSTEM "http://lisa.8304.ch/">
- `parts.ent` : Contains pointers to link the part names to their directories.

  As a convention, entity names begin with a Capital, and all paths (directory names) are in lower case.

- `lisa.dsl` : a style-sheet to produce nicer html output
- `author.ent` : List of all authors (with initials, email)
- `url.ent` : List of all used urls

2.2.2. Which file should I edit to add a book part ?

Add an entry to `parts.ent` and `book.sgml`, create a `part-` directory containing at least the following two files : `index.sgml` and `part.ent`.

2.2.3. Which file should I edit to add a chapter to a part ?

Almost the same procedure. Add an entry to `part.ent` and `index.sgml`, create a `chap-` directory containing at least an `index.sgml` file.

 This is a note. Cute, isn't it ? :-)

2.2.4. Which file should I edit to add a section to a chapter ?

Add entry to . . . , ... [to be completed]

2.2.5. What must a project sgml file look like ?

Every file _must_ contain these 3 lines at their beginning:

Example 2-1. File Header

```
<!- The Linux Server Guide Project - http://lisa.8304.ch -
Maintainer : om ->
<!- $Id: bookstruct.sgml,v 1.12 1999/02/07 14:45:40 om Exp $  ->
<!- $Source: /home/lisa-cvs/cvsroot/book/part-demo/chap-
chap1/bookstruct.sgml,v $  ->
```

When you create a file, the following is enough, then the CVS will add the other information automagically (path, date...). Make sure to add your initials on the first line (Maintainer : __), and that your initials can be found in the `/book/author.ent` index.

Example 2-2. File Header for a new file

```
<!- The Linux Server Guide Project - http://lisa.8304.ch -
Maintainer : __ ->
<!- $Id_$  ->
<!- $Source_$  ->
```


```
(Important note : remove the "_" from $Id_$ and $Source_$ other-
wise it won't
work. Also replace the "__" after "Main-
tainer" by your initials.)
```

Nothing special for the end of the file. Just be sure to close the tags that you opened (sect1, sect2, chap...)

2.3. How to submit texts/corrections

Here is some information how non-maintainers can submit texts and corrections to the project. [to be completed...]

2.3.1. New texts

- Convert the text to sgml, or at least raw text.
- Look up the maintainer of the book-part for which your text is most likely suited.
- Send it via email to the maintainer.

2.3.2. Corrections

- Get the .sgml file where you found the error using the WebCVS or by downloading the cvs .tgz or .zip snapshot. Both are available on the Project Homepage.
- Make your corrections
- Send the file or (preferably) a diff to its official maintainer(s). You can find the email adress(es) by looking at the initials of the maintainer(s) and look(ing) them up in the

`author.ent` file, located in the `/book` directory.

2.4. Project Maintainers

Here is a list of project mainainers (the *staff*). They all have read/write access to the CVS tree.

`Olivier Mueller`

om, *om@8304.ch*, Goodies book part

`Mathieu Arnold`

mat, *arn@multimania.com*, Webserver or Nameserver

`Jan-Philip Velders`

jpv, *jpv@jpv.ddns.org*, Anything :)

`Mira Warren`

wowie, *wowx@usa.net*, IP-stuff

`Eng Siang Tan`

es, *es@oliva.modlang.denison.edu*, Security stuff

`Jochen Knuth`

jok, *jok@stotze.stuttgart.netsurf.de*, Email, Ftp, ...

`John Lombardo`

jl, *john@deltanet.com* To be defined...

2.5. How to play with the CVS

Some basic info about how to setup and handle cvs commands.

2.5.1. How to login

- Setup the CVSROOT environment variable : **export CVSROOT=":pserver:user1@orion.8304.ch:/home/lisa-cvs/cvsroot"**. Replace user1 by your login if you are part of the project staff.
- Login : **cvs login**. The password for the anonymous "user1" is also "user1" (read only access).

2.5.2. How to get a copy of the current tree

- After you are logged in, run this command : **cvs checkout book**. The files should then be transfered, it will create a `book` directory and fill it with the `/book` CVS-tree.
- If you just want to update you local tree (some time after a checkout), do a **cvs update**.

2.5.3. How to upload a new file/directory in the cvs

- Create the file/directory in your local tree.
- Run this command for every file : **cvs add FILENAME**

2.5.4. How to permanently remove a file

- Go into your local tree.
- Run this command for every file : **cvs remove -f FILENAME**. Note: you can't remove empty directories. Don't care about that.

2.5.5. How to submit your changes to the project CVS

- Go to the root of your local CVS tree (in for example `~/lisabook`).
- Run: **cvs commit** and give some information about what you did. If there is a conflict between versions, you may have to run a **cvs update** before you are able to commit.

2.5.6. Could you give me some links where I can find more information about CVS ?

- the man page of the **cvs** command
- CVS Version Control for Web Site Projects : very nice document.
- Cyclic Books/Documentation Page : lots of pointers to CVS documentation
- CVS Documentation-FAQ : CVS Faq, tutorial, slides...

2.6. Sgml examples and some essential tags

This is a list of some really useful DocBook-SGML Tags. Feel free to add yours to the list!

`<para> ... </para>`

Like <p> in HTML.

`<ulink url="http://..."> ... </ulink>`

To add an URL to the text. It will show up as a real link in the HTML output. Example : Nessus Homepage.

`<note> ... </note>`

To add a note

`<listitem> ... </listitem>`

List item

`<varlistentry> ... </varlistentry>`

Var list (like this list of usefull SGML tags...)

`<> ... </>`

...

Chapter 3. SGML Playground

3.1. Introduction

text

3.2. Installation

The SGML-Tools is essentialy a very sofisticated set of wrapper utilities. The main program `sgmltools` itself calls a wide variety of software to generate output. The SGML-Tools package holds for example “JadeTeX”, “Jade”, “Hyperref” etc.

To make the most out of the SGML-Tools, you need “TeX”. Under Linux you will most likely use “teTeX”. Make sure that you have a fairly recent BETA version of 0.9 (whilst writing this, teTeX-0.9-19990126 has been just released, teTeX-0.9-19990123 is the version this maintainer is now using). You can find all sorts of TeX implementations from the TeX Used Group at http://www.tug.org. You need at least teTeX-0.9 to create PostScript or DVI output, you also have to edit your global `texmf.cnf`, otherwise the SGML-Tools’ JadeTeX will only be able to process the first two pages of this SGML file:

- hash_extra = 60000
- hash_extra.jadetex = 15000
- pool_size.jadetex = 200000
- pool_size = 1000000
- max_strings.jadetex = 50000
- max_strings = 70000
- save_size.jadetex = 15000

As for the SGML-Tools, most recent version is version 2.0.2 (as of the 26th of January 1999). You can pick it up from:

- http://www.sgmltools.org (Located in The Netherlands)
- http://www.us.sgmltools.org (Located in the USA)

When you download the .tar.bz2 version, just extract it (**tar xvyfp sgmltools-2.0.2.tar.bz2**) in for example `/usr/local/src`. Then enter the `sgmltools-2.0.2` directory, read the `README` and `INSTALL`. Follow their instructions if you want to do something non-default. If you just want to "do it", issue a **./configure**, **make** and finally a **make install**.

Now you should be all set to use the SGML-Tools !!!

3.3. Configuration

text

3.4. References, links

- For information about TeX, and getting it, look at the pages of the TeX User Group: http://www.tug.org
- For information about the SGML-Tools, look at the main website, or the mailinglistarchive: http://www.sgmltools.org (Main site, The Netherlands) or http://www.us.sgmltools.org (Mirror site, USA)

3.5. Thanks

text

II. Basic Server Installation

In this part we will try to help you decide what kind of system you want or need, so that it can serve your needs without problems. Also will we explore several aspects of installing Linux on a server and deploying that server. Wherever possible we will include examples of real-life situations, so that you can choose more easy (we hope...).

Chapter 4. Common issues when setting up a server

When setting up a server, you need to have an idea about what you want that server to do. Based upon those tasks you start forming an idea about the actual hardware that will comprise the server.

You also have to look ahead. If you want to serve 200 people now, but you expect that within a year that might be 2000, it's best to base the server on those 2000 people. That way, you won't have to start putting in extra diskspace, memory, cpu power and other stuff in little chunks just to keep up with demand. Sizing your server should also be made as easy as possible. If you were to provide webspace, and you want to give your customers twice the space they have now, it's easier to just replace that 4GB with an 8GB disk that *only* contains `/home`, than to get yourself a new 12GB disk and start repartitioning and transferring every partition you have (*which would include directories like* `/`,`/var`, `/usr`, `/home`, `/home/ftp`, `/home/http` *etc.*).

You have to keep that in mind with nearly all aspects of a server which are covered in this document. For example, when also using a server as a primary or secondary DNS server, one usually uses "ns.example.com" as a machine name. DNS server-names must be A-records, they can't just be a pointer (called a "CNAME") to, say, "tiny.example.com". So when operating a machine that has several tasks besides DNS, it would be better to choose the machine name "tiny.example.com" and have your DNS "SOA"-record point to "tiny.example.com" instead of "ns.example.com". That way, if you start spreading services or tasks over other machines, you won't be faced with a lot of hassle just to get the DNS stuff right.

So, before just plunging head-on in, take a moment and write down what you want and perhaps how you want it. It will help you make decisions much easier.

Chapter 5. Hardware considerations

When setting up a server, you need to be sure that all the used hardware will indeed do what you want it to do. A spontaneous reboot because the scsi-channel is being reset one time too many isn't something you'd want.

So, you want a server. Now, you have to think about *what* you want it to serve. You really need to have a good idea of *what* will keep your server busy whilst serving. Do you want it to do very intensive I/O (Input/Output) stuff, like being a mailserver, thus spooling mails to users' mailspools, delivering them to other servers, holding mail for bSMTP (Batched SMTP) users etc. Or do you want it to be your primary DNS server, that all of your users use, thus requiring a lot of memory for BIND to store all the cached queries.

But remember, we can't give you a cookbook that will have your server operate at maximum performance. We can only give you guidelines about certain issues, and what area of operation they involve. Add some salt or pepper to taste.

- *SCSI versus IDE*

  SCSI and IDE are two very different I/O subsystems. SCSI offers higher sustained throughput with very low CPU usage and independant diskoperations (SCSI disk 1 is reading a file, whilst SCSI disk 2 is writing something) with independant datarates. IDE is cheaper, and offers some good quality disks which can rival SCSI performance, but it all boils down to the combination of motherboard, IDE chipset and Harddrive.

  When doing very I/O intensive work, SCSI is usually the better choice. Especially if you would like to add extra disks, without worrying that you can only have 2 per controller (and thus IRQ !), as is the case with IDE. SCSI supports a maximum of 7 devices per controller (and thus IRQ !), the more modern Ultra and UltraWide controllers even support 14 devices. IDE can keep up with most SCSI setups, but you'll need a good IDE chipset (one which does UDMA or fast DMA access, like the very common Intel Triton chipset) and drives that support it. Also note that on an IDE bus, the slowest device sets the maximum datarate on the bus. With SCSI that

isn't a problem, every device will work at it's own pace (although more older devices don't handle that gracefully, aswell some bare-bones device-drivers in the Linux Kernel; look at stuff like "disconnect" or "reconnect").

When thinking about RAID, a setup in which you combine several disks in to one logical disk, or mirror one disk to another, or even have a set of disks of which a given number may fail without interrupting the reliable operation of the array, SCSI is usually the way to go. Although you could use RAID-0 (striping disks/partitions into one logical one) or RAID-1 (disk-to-disk mirroring) with IDE (place the disks on a seperate controller each to improve parallel performance with RAID1 or RAID5!).

So for normal servers, which for example don't act as a busy SQL-Server or have to accept several thousands of emails a minute, IDE usually suffices. A combination is also possible, storing an SQL database on the SCSI datadisk, whilst the main server system runs of an IDE drive.

- *Memory & swap*

  When operating a server, remember that everything uses memory. A system will usually benefit more from additional memory than from additional CPU power (unless you are doing stuff that needs to calculate at Cray performance). When operating with little memory, swapping can cause serious performance degradation. For example, a simple bash shell can consume 1 Megabyte, and a single Apache instance can consume 2 MegaByte, Bind 8 needs almost 2 MegaByte for just it's binary. So planning ahead with memory is a must. A minimum memory footprint would be 32 or 64 Megabytes, that way a server could have several memory intensive programs, and have some breathing space. For swapspace, the rule of thumb is twice your memorysize. There is some debate whether swap space should exceed 256 Megabytes, because it would be terribly inefficient to swap that much memory out.

- *ISA, VESA, MCA vs PCI*

  When adding add-on cards to PC's, you are limited to the kind of "slots" your motherboard provides. There are several kinds of slots. It all started with ISA, which still is available and in wide use, it provides 8 or 16 bit slots with an 8 MHz bus. Then IBM devised MCA (Micro Channel Architecture) for its PS/2-series, it is better

than ISA and has some features which are very handy, like automagic card configuration, on-board on-boot configuration menus etc. . VESA and PCI were rivals once. VESA lost, it was an extension slot placed in the length of the 16-bit ISA slots, cards were very long. PCI won, and provides 32 or 64 bits with an 33 MHz bus or even faster with special chipsets and cards, IRQ and IO-adresses are easily allocated.

Nowadays, motherboards come with several ISA and PCI slots, and perhaps a special PCI slot for graphics called AGP. For 100Mb Ethernet-cards and Ultra-/UltraWide-SCSI Controllers and things alike, you'd be pretty well of with PCI version. For more moderate cards, like a simple VGA graphics cards or ISDN cards or Multi-Serial-Ports cards, ISA is more suited.

- *10 vs 100 Megabit Ethernet*

  Nowadays you usually connect your server through an Ethernet networkcard. Ethernet has several physical forms and speeds. The most used speed is 10 Megabit/s, which can be implemented using Coax (also called Thin Ethernet), Thick Ethernet coupled with Transceivers and UTP/STP/TP-cables with HUBs. The new and faster 100 Megabit/s can only be achieved with UTP/STP/TP-cables and special 100 Megabit/s hubs/switches/routers/etc. For a server that is going to transmit or receive a lot of traffic now, and which will increase steadily in the future, it's best to buy a 100 Megabit networkcard, and operate it in 10 Megabit/s mode now. That way, you can simply 'upgrade' to 100 Megabit by installing the newer (and costlier) routers/switches/hubs/etc. later and rebooting the server or reconfiguring the network card.

- *Cheap vs Expensive*

  When buying components for your server, you have a wide price-range to choose from. You can go with that very cheap NE2000 networkcard, or that very expensive 3COM super-duper card. The same applies to case, motherboard, CPU, memory, disks etc.

  You have to remember that not all the good and reliable stuff is expensive, but also that the cheap stuff isn't all that good and reliable. Spend some extra money on a good case with power-supply, a reliable motherboard, some good RAM, and a CPU.

Intel isn't necessarily the choice for a CPU. Whilst writing this, the new AMD K6-III is said to cream the Pentium-III. But you could go even further, why not get an Alpha ? Just be sure to buy something that will suit *your needs*, not because it was cheap or the new hot-thing!

III. Web Servers

Chapter 6. Common issues for Web Servers

text


TOC

Chapter 7. Apache

7.1. Introduction

text

7.2. Installation

text

7.3. Configuration

text

7.4. References, links

text

7.5. Thanks

text

TOC

Chapter 8. Roxen

8.1. Introduction

text

8.2. Installation

text

8.3. Configuration

text

8.4. References, links

text

8.5. Thanks

text

TOC

IV. Mail Servers

Chapter 9. Common MTA issues

To mail or not to mail

When installing a mailserver, it's vital that you plan ahead. Also, you must have a good idea of what mail-related things will be running. A mailinglistserver perhaps ? With all the planning, you can select *The right tool for the right task*.

What type of mailserver will you be running ? Will it be the primary MX for several domains ? Will it also be running a POP3/IMAP server ? Will it have to do complex forwarding ? All things one has to take into account. These issues nowadays don't fixate the choice of mailserver software, but some packages have a more flexable setup then other, although this varies amongst administrators.

SPAM. One of the most irritating aspects of e-mail, is *Unsollicited Commercial E-mail* a.k.a. SPAM. Most mailsoftware has anti-spam provisions, some have them activated by default, some not. Some provide special hooks to make your own list of known spammers, spam-domains, spam-IP's etc. Others even go as far as providing support for Realtime Blackhole Lists (a sort of 'global warning' system, administrators all around the world feed this database with sites they can't trust, like open-relay systems).

When operating a mailserver, always keep in mind that you might have to migrate your setup to a different machine. You never know how a certain piece of software will keep up with the demand put on it. Or how the DNS changes will work out. So, let your mailserver operate under it's own name, for example *mail.example.com*, if that machine also has other names, use CNAME-records for them.

Chapter 10. Exim

10.1. Introduction

Exim is a Mail Transport Agent, developed by Philipp Hazel.

10.2. Installation

text

10.3. Configuration

text

10.4. References, links

text

10.5. Thanks

text

Chapter 11. Qmail

11.1. Introduction

text

11.2. Installation

text

11.3. Configuration

text

11.4. References, links

text

11.5. Thanks

text


TOC

Chapter 12. Sendmail

12.1. Introduction

text

12.2. Installation

text

12.3. Configuration

text

12.4. References, links

text

12.5. Thanks

text

TOC

Chapter 13. Postfix

13.1. Introduction

text

13.2. Installation

text

13.3. Configuration

text

13.4. References, links

text

13.5. Thanks

text

V. FTP Server

This part is for the FTP stuff

Chapter 14. Text

Text


TOC

VI. Nameserver (DNS)

This part is for the DNS stuff

Chapter 15. Text

Text

VII. Server Administration

This part is for the Admin stuff

Chapter 16. Text

Text

VIII. Security

This chapter intends to aid the Linux system administrator with security issues that revolves around maintaining a server. Internal and external securities are covered. Methods of security include configuring and fine-tuning system files as well as externally available programs and scripts.

Chapter 17. Title

Text

Chapter 18. Title

Text

Chapter 19. Title

Text

Chapter 20. Title

Text

Chapter 21. Title

Text

IX. Monitoring

This part is for the Monitoring stuff

Chapter 22. Text

Text

X. Firewall and Proxy

This part is for the Firewall & proxy stuff

Chapter 23. Text

Text


TOC

XI. Goodies

This part is for the Goodies (oh goody !!!)

Chapter 24. Text

Text

XII. F.A.Q.

This part is for the FAQ

Chapter 25. Text

Text

TOC