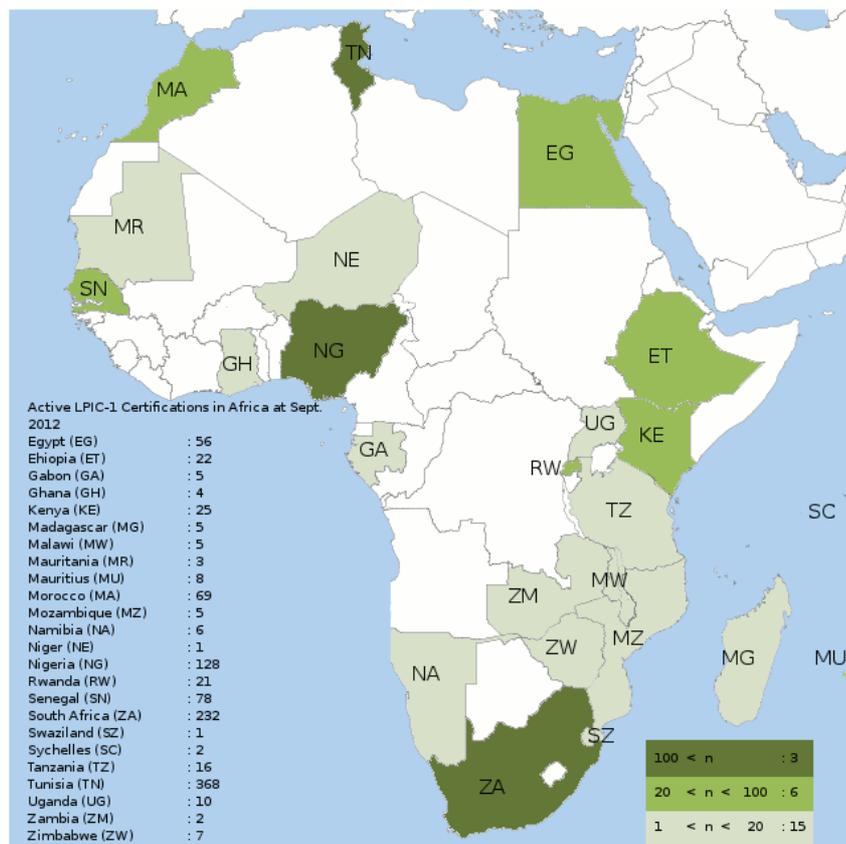


ict@innovation

Creating Business and Learning Opportunities
with Free and Open Source Software in Africa

ict@innovation: Training Guide on Linux System Administration LPI Certification Level 1

Supporting African IT-Enterprises to get Open Source Skills by Getting
Certified on Level 1 of the Linux Professional Institute (LPI) Certification



The map shows the number of active LPIC-1 Certifications in Africa as at September 2012

www.ict-innovation.fossfa.net

Published by



Version 1.1, November 2012

giz Deutsche Gesellschaft
für Internationale
Zusammenarbeit (GIZ) GmbH

On behalf of
BMZ Federal Ministry
for Economic Cooperation
and Development

[ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification] Created during the initiative "ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software in Africa", a programme of FOSSFA and GIZ. For more information see www.ict-innovation.fossfa.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.



This page intentionally left BLANK

Imprint

Published by

GIZ – Deutsche Gesellschaft für Internationale Zusammenarbeit GmbH

Competence Center Human Capacity Development (HCD) Africa
Friedrich-Ebert-Allee 40
53113 Bonn
Germany
Phone +49 (228) 4460-0
www.giz.de

FOSSFA – Free Software and Open Source Foundation for Africa

Secretariat hosted at
Advanced Information Technology Institute (AITI) of the
The Ghana-India Kofi Annan Centre of Excellence in ICT
PMB, State House, Accra
Ghana
Phone +233 (244) 954 413
www.fossfa.net

For more information, please contact:

secretariat@fossfa.net
cem@fossfa.net
thorsten.scherf@giz.de

FOSSFA Secretariat
FOSSFA Community Empowerment Manager (CEM)
GIZ Division Economic Development & Employment,
ICT Advisor, Sector Project ICT4D

Funding

This Training Guide was produced with the financial assistance of the German Federal Ministry for Economic Cooperation and Development (BMZ). The content of this document are the sole responsibility of the authors and can under no circumstances be regarded as reflecting the position of the BMZ, GIZ, or FOSSFA.

License

This Training Guide is provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.

Introduction by FOSSFA and GIZ

"How do I know that this IT company from Kampala will be able to maintain my IT server infrastructure?", asks a contract-giving government agency in Uganda. The answer lies in a trust-building certification – a crucial ingredient of any economic development agenda. Therefore, the Free Software and Open Source Foundation for Africa (FOSSFA) and the Deutsche Gesellschaft für Internationale Zusammenarbeit GmbH (GIZ) are proud to present "**ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification.**"

This Training Guide is part of the programme "**ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software (FOSS) in Africa**". ict@innovation is a capacity building programme of FOSSFA and GIZ which supports small and medium enterprises in the field of Information and Communication Technologies (ICT), aiming to encourage the growth of African ICT industries. ict@innovation promotes

- Free and Open Source business models: [Free your IT Business in Africa!](#)
- [Linux Administration Certification](#)
- [Coding FOSS in Africa](#)

This Training Guide is part of the pillar "Linux Administration Certification". In the format of a Train-the-Trainer programme, this pillar builds capacities of African SMEs in offering high quality FOSS services to improve the level of trust of customers through certification of their FOSS skills.

We hope that the "ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1", together with the associated Train-the-Trainer scheme, will contribute to removing a major barrier against the adoption and deployment of FOSS in Sub-Saharan Africa: the lack of human resources with FOSS skills demonstrated by recognized certificates.

In order to address a wide range of capacity needs and training environments in Africa, the Training Guide and the Train-the-Trainer programme builds on the Linux Professional Institute (LPI) Certification as a world-wide recognized distribution- and vendor-neutral standard for evaluating the competency of Linux professionals with the possibility to hold low-cost paper-based examinations.

The "ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1" is released under an open license (Creative Commons Attribution-Share Alike 3.0 Germany License) which allows free distribution, remixing and updating of the material. Our goal is thereby to empower local African training institutions to offer low-cost trainings. And we are looking forward to seeing further development and updating of this Training Guide in the spirit of sharing and mutual capacity building.

We would like to thank and attribute the authors of the commons resources, which form the basis of this manual (for full acknowledgement, please see section "Authors, Attribution and Licensing").

We would also like to thank all those who played a major role in the production of this Training Guide: First and foremost, our thanks go to the lead editor **Evans Ikua**, the ict@innovation FOSS Certification Manager (Kenya), who oversaw the entire production process. We thank the content editors, **Mark Clarke** of Jumping Bean (South Africa), **Brian Sennoga** of the University of Health Sciences (Uganda), and **Dr Chris Brown** of Stay Awake Training Ltd (UK). We also thank the co-editors **John Matogo** (Kenya), **Ken Mutua** (Kenya), **Bernard Adongo** (Kenya), and **Trust Zifa** (Zimbabwe). Many thanks also go to **George Nyambuya**, the ict@innovation Africa Coordinator (South Africa), **Petra Hagemann**, GIZ Project Manager and **David Paulus**, editor for GIZ (Germany). Finally, we thank all those who participated in the first regional LPI training of trainers held in Nairobi for their initial review, especially **Sisay Adugna** (Ethiopia), and all the people who contributed to making this manual a reality.

ict@innovation hopes that this Training Guide will indeed support the acquisition of adequate FOSS skills by a large population of Linux System Administrators in Sub-Saharan Africa. We believe that this will help achieve the goal of promoting the socio-economic development of Africa by supporting the growth of the ICT industries in the SME sector through the implementation of quality FOSS services through certification.

May a thousand more local innovative IT solutions and services blossom on the continent!

Nnenna Nwakanma, FOSSFA Council Chair and CEO, NNENNA.ORG
Balthas Seibold, **Petra Hagemann**, **Steffi Meyer**, GIZ Project Management Team

[ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification] Created during the initiative "ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software in Africa", a programme of FOSSFA and GIZ. For more information see www.ict-innovation.foSSFA.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.

Who this Training Guide is for

This Training Guide was mainly written for African experts and institutions wanting to become qualified trainers on the subject of Linux System Administration. It is geared towards trainers who want to incorporate FOSS certification training based on the Linux Professionals Institute (LPI) community certification programme into their institution's curriculum and/or developing FOSS certification training as a new revenue stream.

This open Training Guide provides a set of learning modules with learning objectives, key knowledge areas, introductions and concrete learning steps and handouts as well as a module on how to be a FOSS trainer. The guide is therefore particularly suitable for use in Training-of-Trainers settings and the development of advanced courses within ICT-associations, their member organisations, ICT-training institutions and universities. It can be used in tutored learning environments (e.g. 2-week courses preparing for the certification exams) or settings such as peer-to-peer learning, self-study, blended learning and e-learning.

In addition, any learner anywhere in the world will find this guide useful for study and preparation of the LPI level 1 certification. S/he will learn how to fulfil the first essential steps in becoming a Linux System Administrator charged with installing, supporting, and maintaining Linux-based computer systems.

The Training Guide builds on LPI community certification as a world-wide recognized distribution-neutral and vendor-neutral standard for evaluating the competency of Linux professionals with the possibility to hold low-cost paper-based examinations. It has three levels. The LPIC-1 (exams 101 and 102) level taught in this training guide covers the fundamental Linux system administration skills. Like the LPI curriculum itself, the course is "distribution neutral", that is, it does not favour a specific Linux distribution. Learners can therefore use this manual not only to pass the LPIC-1 exams but also be in a position to operate a range of different Linux distributions. The detailed objectives of LPI EXAM 101 AND 102 are online at: http://www.lpi.org/eng/certification/the_lplic_program/lpic_1

Learners are expected to have

- Extensive experience (several years) in using computers, including a strong knowledge of hardware components and their interaction with basic Operating System (OS) components.
- A general knowledge of computing and networking basics such as binary and hexadecimal maths, file-system structures, Ethernet and Internet networking operations and hardware, etc.
- More than three cumulative months of practical experience using a GNU/Linux, BSD or Unix OS, working at the command-line (in a text terminal or console) either locally or remotely.

Those with less experience, however, should not be discouraged from using this guide, if (and only if) they are willing to spend extra time catching up on the prerequisite background skills and knowledge; a challenging task, but not an impossible one.

What you can learn through this Training Guide

I) In the **LPI 101 Module** you will learn how to:

- Install Linux, making appropriate choices for disk partitioning
- Boot the system, change run levels, shut-down and reboot
- Work effectively at the shell command prompt
- Install and manage packages using both RedHat and Debian tools
- Manage, find, copy, delete, archive and compress files and directories
- Process text streams using pipes, filters and re-direction
- Manage processes and modify process execution priorities
- Search text files with regular expressions and edit files with vi
- Create partitions and filesystems, and maintain their integrity
- Control file access permissions

II) In the **LPI 102 Module** you will learn how to:

- Customise the shell and write simple shell scripts
- Query databases and manipulate data using SQL
- Install and configure the X server and set up a display manager
- Manage user accounts and groups
- Schedule jobs at regular intervals using cron
- Localise the system for a language other than English
- Keep your system clock correct
- Manage printers and printing
- Understand IP networking and set up a basic network configuration
- Maintain host security and enable secure login with ssh

III) In the **FOSS Trainer Module** you will learn how to become a trainer in Free and Open Source Software (FOSS):

- Understand some of the requirements for becoming a FOSS trainer
- Be able to identify and seize the opportunities that exist for FOSS training as a business
- Gain the knowledge and skills required to organise and provide FOSS training
- Appreciate the benefits of peer production of Open Educational Resources and Open Content

Outline and Timeline

The modules of the first two chapters are structured closely around the topics in the LPI curriculum. Topics 101, 102, 103 and 104 relate to the LPIC 101 exam. Topics 105 through 110 relate to the LPIC 102 exams. This guide covers all 10 topics. Each sub-topic is assigned a weight. For each exam the total weight is 60. There are 60 questions on each exam, so the weight gives an indication of how many questions you can expect to receive on that topic.

This manual covers the new LPI syllabus (as of 2010/2011). It is thus up to date, which also means that it covers the most current issues pertaining to the Linux Operating System. The writers assumed that the learner would be using the Centos 5.0 Operating System. This is because this is a more stable Linux distro that does not change much over the years being a server based distro, as opposed to other distros whose release cycle is faster.

The tables below lists the topics and gives their weighting in the LPI certification. The classroom times given for each topic are approximate and are based on a total duration of 40 hours for each course. For page numbers of the topics, please see the full table of content.

LPIC 101

	Topic	Weight	Time
101	System Architecture		
101.1	Determine and configure hardware settings	2	1h 20m
101.2	Boot the system	3	2h 00m
101.3	Change run levels and shutdown or reboot the system	3	2h 00m
102	Linux installation and package management		
102.1	Design hard disk layout	2	1h 20m
102.2	Install a boot manager	2	1h 20m
102.3	Manage shared libraries	1	0h 40m
102.4	Use Debian package management	3	2h 00m
102.5	Use RPM and YUM package management	3	2h 00m
103	GNU and Unix Commands		
103.1	Work on the command line	4	2h 40m
103.2	Process text streams using filters	3	2h 00m
103.3	Perform basic file management	4	2h 40m
103.4	Use streams, pipes and redirects	4	2h 40m
103.5	Create, monitor and kill processes	4	2h 40m
103.6	Modify process execution priorities	2	1h 20m
103.7	Search text files using regular expressions	2	1h 20m
103.8	Perform basic file editing operations using vi	3	2h 00m
104	Devices, Linux Filesystems, Filesystem Hierarchy Standard		
104.1	Create partitions and filesystems	2	1h 20m

104.2	Maintain the integrity of filesystems	2	1h 20m
104.3	Control mounting and unmounting of filesystems	3	2h 00m
104.4	Manage disk quotas	1	0h 40m
104.5	Manage file permissions and ownership	3	2h 00m
104.6	Create and change hard and symbolic links	2	1h 20m
104.7	Find system files and place files in the right location	2	1h 20m

LPIC 102

	Topic	Weight	Time
105	Shells, scripting and data management		
105.1	Customise and use the shell environment	4	2h 20m
105.2	Customise or write simple scripts	4	2h 40m
105.3	SQL data management	2	1h 20m
106	User Interfaces and Desktops		
106.1	Install and configure X11	2	1h 20m
106.2	Set up a display manager	2	1h 20m
106.3	Accessibility	1	0h 40m
107	Administrative Tasks		
107.1	Manage user and group accounts and related system files	5	3h 20m
107.2	Automate system administration tasks by scheduling jobs	4	2h 40m
107.3	Localisation and internationalisation	3	2h 00m
108	Essential System Services		
108.1	Maintain system time	3	2h 00m
108.3	Mail Transfer Agent (MTA) basics	3	2h 00m
108.4	Manage printers and printing	2	1h 20m
109	Networking Fundamentals		
109.1	Fundamentals of internet protocols	4	2h 40m
109.2	Basic network configuration	4	2h 40m
109.3	Basic network troubleshooting	4	2h 40m
109.4	Configure client side DNS	2	1h 20m
110	Security		
110.1	Perform security administration tasks	3	2h 00m
110.2	Set up host security	3	2h 00m
110.3	Securing data with encryption	3	2h 00m

As this manual is not only designed for Linux system administrators, who want to prepare themselves for the LPI-Certification or want to upgrade their Linux skills, but mainly to encourage trainers to offer their own FOSS trainings, the last chapter deals with how to improve trainer skills and how to offer FOSS trainings. Of course chapter 3 “FOSS TRAINING” is NOT part of the LPI Certification, but offers support for offering and implementing successful FOSS trainings. Please see a suggested timeline for this chapter below.

FOSS TRAINING

	Topic	Time
III.1	HOW TO BE A FOSS TRAINER	1 h 30m
III.2	FOSS TRAININGS AS A BUSINESS	1h 30m
	Invited talks: Discussion of FOSS training Experience, FOSS Business in Africa, FOSS in Government, FOSS in Education	1h 45m
III.3	ORGANISING TRAININGS	1h 30m
III.3.1-3	Training Material Development	1h 30m
III.4	OPEN EDUCATIONAL RESOURCES AND OPEN CONTENT	1h 30m
III.5	TRAINING COMMUNICATION SKILLS	1h 45m

Classroom Setup Guide

The LPI Certifications are practical, hands-on courses. This guide has been written with a bias that whoever uses it will have a computer onto which to install a Linux distribution, to be able to perform the commands and routines.

Note for Instructors: *Each student needs a PC with a minimum of 512 Mbytes of memory (1 Gbyte is better) and 20 Gbytes of disk space. Machines should be networked, and an external internet connection is required for some exercises. Machines require a DVD drive and must be able to boot from the drive. Although the course is largely distribution-neutral, the authors of this guide strongly recommend using RedHat Enterprise Linux 5 (or equivalently CentOS 5) because this matches most closely the expectations of the LPI level 1 curriculum. Each student requires a copy of the installation media (CDs or DVD) before hand. CentOS ISO images are available from: <http://mirror.centos.org/centos/5/isos/>*

Because the students will install Linux directly onto the PC, do not use machines that have some other installation, such as Windows, that needs to be preserved.

*The classroom should have a data projector for demonstrations and for presenting slides. Be sure to check that the laptop you are planning to use to present slides and demonstrations will drive the projector at the correct resolutions **before** the start of class.*

About ict@innovation - Creating Business and Learning Opportunities with Free and Open Source Software in Africa

The programme ict@innovation focuses on Free and Open Source Software (FOSS) as a key technology to drive innovation, add local value and create sustainable and affordable ICT-solutions. ict@innovation is a partnership of FOSSFA (Free Software and Open Source Foundation for Africa) and GIZ (Deutsche Gesellschaft für Internationale Zusammenarbeit GmbH) and aims to enhance regional networking and strengthen consulting capacities of regional and national ICT associations and training institutions as well as of other change agents. Funding partners are the German Federal Ministry for Economic Cooperation and Development (BMZ) and the Open Society Initiative for Southern Africa (OSISA).

Linux Certification in Africa – The ict@innovation Training of Trainers Programme

One pillar of the ict@innovation programme aims to help African SMEs involved in Open source Solutions development, implementation and training to improve the quality of their solutions offering through certification of their FOSS skills. The project partners have appreciated the importance and urgency of increasing the number of Linux systems administrators on the African continent so as to have a bigger impact on the uptake of FOSS on the continent (see map on front page).

To achieve this, the “ict@innovation Linux Certification in Africa” programme is implementing the following capacity building actions:

- Support the development of open content Linux Professional Institute Certification (LPI) training guide;
- Offer Training of Trainers to individuals in the participating countries, leading to the LPI exams for certification;
- Support the trainers in delivering the LPI training and certification in their countries and institutions as a business model.

Implementing Partners



Funding / Strategic Partners



Join ict@innovation: Meet the Linux Admin and Certification Trainers community! Find a local training offer!

- Do you want to network with African and international Linux System Administrators?
- Do you want to know who can provide Linux System Administrator in your country in Africa?
- Do you want to expand or kickstart the business of providing Linux Administration training?
- Do you have experience in training others on Linux and/or are you part of a training institution?
- Do you want to network with other Linux trainers in your region?

Then feel free to join the ict@innovation Certification community by registering on the ict@innovation website and creating a profile, indicating your interests and skills in FOSS and Linux Administration. This will enable you to contribute to the respective website fora and wikis, to join in a knowledge exchange with other persons interested in FOSS Certification and/or involved in the ict@innovation programme.

Please register here: <http://www.ict-innovation.fossfa.net/user/register>

At the portal, you can also access the pool of **African ict@innovation Linux Certification trainers and their training offerings**. These trainers from 13 countries have been qualified through ict@innovation's Training of Trainer initiative in facilitating the course "ict@innovation: Linux System Administration, LPI Level 1 - Supporting African IT-enterprises to get Open Source skills and certification on level 1 of the Linux Professional Institute (LPI) Certification":

You will find the growing pool of trainers and their offerings online at: <http://www.ict-innovation.fossfa.net/fct>

You are also welcome to join the certification group through its mailing list by going to this link: <http://mail.fossfa.net/cgi-bin/mailman/listinfo/certification>

In addition to the ict@innovation community on "Linux Certification in Africa", you can meet communities around the following themes on the ict@innovation portal:

- Free your IT-Business in Africa! - Advanced Training on African FOSS Business Models for IT-SMEs
- Coding FOSS in Africa - the mentored internship programme (MIP) of AVOIR & ict@innovation
- International and regional exchange

Find more information on Linux Certification in Africa

The following overview provides you with further information on Linux Certification such as open learning material, presentations and evaluation forms. If you require additional information, including on services such as making use of the Pool of African ict@innovation Experts & Trainers on Linux System Administration, please contact the FOSSFA Community Empowerment Manager (CEM) by email cem@fossfa.net.

Full set of training material of ict@innovation: Training Guide on Linux System Administration, LPI Level 1 – Supporting African IT-enterprises to get Open Source skills and certification on level 1 of the Linux Professional Institute (LPI) Certification]	
<p>Training Guide & Trainers Manual: Introduction, Overall Learning Objectives, Outline and Timetable, Background, Information on main content, exercises, lead questions, samples, Module on how to be a trainer etc. The training guide is available in two forms as an open educational resource:</p> <p>a) Download as a printable book b) Online Wiki for updates, corrections etc.</p>	<p>a) <u>Download as a printable book:</u> http://www.ict-innovation.fossfa.net/blp b) Online Wiki for updates, corrections etc. http://www.ict-innovation.fossfa.net/blp</p>
<p>Model Presentations for Trainers: Presentation slide decks in OpenOffice format</p>	<p>http://www.ict-innovation.fossfa.net/blp</p>
<p>Evaluation Forms: Model online evaluation forms to assess training</p>	<p>http://www.ict-innovation.fossfa.net/blp</p>
<p>Pool of African ict@innovation Experts & Trainers on Linux System Administration: Contact to Africa-based experts and trainers who have experience with the course content.</p>	<p>http://www.ict-innovation.fossfa.net/fct</p>

For more information, see <http://www.ict-innovation.fossfa.net>

AUTHORS, ATTRIBUTIONS AND LICENSING

How you can reuse this guide – License and invitation to share

In the spirit of sharing and mutual capacity-building, this Training Guide is released under an open licence: Creative Commons Attribution-Share Alike 3.0 Germany. We are therefore looking forward to seeing further distribution, remixing and updating of the courses. Please see previous section for download links and wiki for sharing of updates. Please note: **With this license, you are free to copy, distribute, transmit and adapt the work under the following conditions:**

- **Attribution**

You must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work) – see next chapter for full attribution.

- **Share Alike**

If you alter, transform, or build upon this work, you may distribute the resulting work only under the same, similar or a compatible license.

To view a copy of this license, visit <http://creativecommons.org/licenses/by-sa/3.0/de/deed.en> or send a letter to Creative Commons, 171 Second Street, Suite 300, San Francisco, California, 94105, USA.

Copyright for this version: FOSSFA & GIZ

We are eager to learn about your experiences and stories around the material and wish you a good and efficient learning experience. Therefore, FOSSFA & GIZ would kindly request a brief notice in case of use of this Training Guide indicating context of use/ modification and number of people reached- in addition to the attribution under the license. Please give us a feedback at secretariat@fossfa.net.

Full attribution – Who built this guide and its source material

Please note that the license of this material requires attribution of all licensors and authors in future versions. It should therefore provide all information of the following two section **Licensors and authors of this version**

- **Licensors and authors of this version**

This version of the training guide is licenced under a Creative Commons Attribution-Share Alike 3.0 Germany License and should include the following attribution:

"Created during the initiative "ict@innovation - Creating Business and Learning Opportunities with Free and Open Source Software in Africa", a programme of FOSSFA and GIZ. For more information see www.ict-innovation.fossfa.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ. Under the license, the copyright holder (FOSSFA & GIZ) do not endorse any previous or future versions of the material or the use of the work."

It should also attribute all editors and authors as named below:

	Name	Role	Country	contact
1	Evans Ikua	Lead Editor	Kenya	ikua@fossfa.net
2	Chris Brown	Content Author	UK	cb11840@gmail.com

3	Mark Clarke	Content Author	RSA	mark@jumpingbean.co.za
4	Brian Sennoga	Content Author	Uganda	bssennoga@ihsu.ac.ug
5	Trust Zifa	Material co-editor	Zimbabwe	trust@jumpingbean.co.za
6	John Matogo	Material co-editor	Kenya	jmatogo@strathmore.edu john.matogo@gmail.com
7	Ken Mutua	Material co-editor	Kenya	kenmutua@gmail.com
8	Bernard Owuor	Material co-editor	Kenya	b_owuor@yahoo.com
9	Sisay Adugna	Material co-editor	Ethiopia	sisayie@gmail.com
10	Balthas Seibold	GIZ Senior Project Manager	Germany	Balthas.seibold@giz.de
11	Petra Hagemann	GIZ Project Manager	Germany	Petra.hagemann@giz.de
12	George Nyambuya	Africa Coordinator - ict@innovation	RSA	George.nyambuya@giz.de
13	David Paulus	GIZ Intern	Germany	david.paulus@giz.de

A large part of this Training Guide is the original work of these authors. However, the guide also built on prominent, high quality training material under open licenses. Thus we made use of LinuxIT's 2005 Study Guides For Linux System Administration I and II. An estimated 30 per cent of our material is based on LinuxIT's material (LinuxIT is the trading name of LinuxIT Europe Ltd. www.linuxit.com). At this point we want to thank and honor LinuxIT' and all the authors of the sources we provided in section b) below and which should also be attributed in all future versions of the material:

- **Licensors and authors of source material with licenses and links**

Study Guide for Linux System Administration I - Lab work for LPI 101 - version 0.2

Released under the GFDL by LinuxIT (LinuxIT is the trading name of LinuxIT Europe Ltd. www.linuxit.com) with funding by UNDP-APDIP International Open Source Network (IOSN) and the International Development Research Centre Canada.- Copyright (c) 2005 LinuxIT.
<http://savannah.nongnu.org/projects/lpi-manuals/>
<http://www.scribd.com/doc/2056167/LPI-101Guide>

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with the Invariant Sections being History, Acknowledgements, with the FrontCover Texts being "released under the GFDL by LinuxIT". - permission given by LinuxIT to relicense as Creative Commons Attribution-Share Alike 3.0 Germany License

Editors / Authors: Adrian Thomasset <adriant@linuxit.com> <http://www.linuxit.com/> / Andrew Meredith <andrew@anvil.org> <http://www.anvil.org/>
 Andrew D Marshall <admarshall@gmail.com> <http://h0lug.sourceforge.net/> / Duncan Thomson <thom-ci0@paisley.ac.uk> <http://www.paisley.ac.uk/>

Study Guide for Linux System Administration II - Lab work for LPI 102

Released under the GFDL by LinuxIT (LinuxIT is the trading name of LinuxIT Europe Ltd. www.linuxit.com) with funding by UNDP-APDIP International Open Source Network (IOSN) and the International Development Research Centre Canada.- Copyright (c) 2005 LinuxIT.
<http://savannah.nongnu.org/projects/lpi-manuals/>

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with the Invariant Sections being History, Acknowledgements, with the FrontCover Texts

[ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification] Created during the initiative "ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software in Africa", a programme of FOSSFA and GIZ. For more information see www.ict-innovation.fossf.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.

being "released under the GFDL by LinuxIT" - permission given by LinuxIT to relicense as Creative Commons Attribution-Share Alike 3.0 Germany License

Editors / Authors: Adrian Thomasset <adriant@linuxit.com> <http://www.linuxit.com/> / Andrew Meredith <andrew@anvil.org> <http://www.anvil.org/>
Andrew D Marshall <admarshall@gmail.com> <http://h0lug.sourceforge.net/> / Duncan Thomson <thom-ci0@paisley.ac.uk>
<http://www.paisley.ac.uk/>

http://en.wikibooks.org/wiki/LPI_Linux_Certification/Configure_Fundamental_BIOS_Settings

Text is available under the Creative Commons Attribution-ShareAlike License; additional terms may apply. 19/01/2011

http://en.wikibooks.org/wiki/LPI_Linux_Certification/LPIC1_Exam_101/Linux_Installation_&_Package_Management

Text is available under the Creative Commons Attribution-ShareAlike License; additional terms may apply. 19/01/2011

http://en.wikibooks.org/wiki/LPI_Linux_Certification/LPIC1_Exam_101/Linux_Installation_%26_Package_Management

Text is available under the Creative Commons Attribution-ShareAlike License; additional terms may apply. 19/01/2011

http://en.wikibooks.org/wiki/LPI_Linux_Certification/Using_Red_Hat_Package_Management

Text is available under the Creative Commons Attribution-ShareAlike License; additional terms may apply. 19/01/2011

http://docs.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/6/html/Storage_Administration_Guide/xfsothers.html

Text is available under the Creative Commons Attribution-ShareAlike 3.0 Unported license ("CC-BY-SA") 10/02/2011

<http://en.wikipedia.org/wiki/Fstab>

Text is available under the Creative Commons Attribution-ShareAlike License; additional terms may apply. 19/01/2011

http://en.wikibooks.org/wiki/Linux_Guide/Using_the_shell

Text is available under the Creative Commons Attribution-ShareAlike License; additional terms may apply. 19/01/2011

LPIC 102 Preparation Guide ver. 3 (Alan McKinnon, Michel Bisson)

www.kroon.co.za/lpi/lpi102prepv3.pdf

This is a free document. You may distribute, modify, or improve it for personal or commercial use as you wish. I take no responsibility of any kind for the accuracy of the information in this document, nor for the success or failure of any participants in passing the exam. I would appreciate it that if you make modifications to this document, you send me a copy of the new version.

<http://www.ict-innovation.fossfa.net/wiki/public-wiki/ictinnovation-training-materials>

ict@innovation: Free your IT-Business in Africa - Created during the initiative "ict@innovation - Creating Business and Learning Opportunities with Free and Open Source Software in Africa", a programme of FOSSFA and GIZ. For more information see www.ict-innovation.fossfa.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ. Under the license, the copyright holder (FOSSFA & GIZ) do not endorse any previous or future versions of the material or the use of the work.

Authors and editors: [Overall editors, curriculum developers, public relations: Sulayman K. Sowe, Karsten Gerloff (The Netherlands), Petra

[ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification] Created during the initiative "ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software in Africa", a programme of FOSSFA and GIZ. For more information see www.ict-innovation.fossfa.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.

Hagemann, Jin Soo Kim, Balthas Seibold (Germany), Geraldine De Bastion, Andrea Götzke (Germany), George Nyambuya (South Africa)] [content contributors, authors and co-editors : Yese Bwalya (Zambia), Clara Alice Chirwa (Malawi), Alex Gakuru (Kenya), Derek Lakudzala (Malawi), James Wire Lunghabo (Uganda), Nhlanhla Mabaso, Arnold Pietersen (South Africa), Joseph Sevilla (Kenya), Celso Timana (Mozambique)] [content contributor and co-editors (volunteers): Samer Azmy (Egypt), Nico Elema (South Africa), Thomas Jonas, Foibe Kalipi (Namibia), Timothy Kasolo (Zambia), Faye Macheke (South Africa), Glenn McKnight (Canada), Irene Fernandez Monsalve (Spain), Paschalia Ndungwa Ouma (Uganda), Frank Tilugulilwa (Tanzania)] [lead training/ facilitation team for Training-of-Trainers on the ict@innovation FOSS Business Models Training: Shirley Akasreku, Kofi Kwarko, Frederick Yeboah (Ghana)] [Resource persons, Advice: John 'Maddog' Hall (USA), Arjan de Jager, Rishab A. Ghosh (The Netherlands), Francois Letellier (France), Nnenna Nwakanma (Cote d'Ivoire), Kim Tucker (Switzerland), Victor van Reijswoud (Papua New Guinea), Philipp Schmidt, Thilo Thormeyer (South Africa)] [Managers of business models component of ict@innovation, training partners and training material facilitators: Karsten Gerloff (Germany), Andreas Meiszner (Portugal), Sulayman K Sowe (The Netherlands), Gregor Bierhals (The Netherlands)]

Please also note the following indications and disclaimers

All trademarks mentioned in this document and potentially protected by third parties shall be subject to the unqualified provisions of the pertinent trademark law and property rights of the registered owner.

Under the license, FOSSFA/GIZ do not endorse any previous or future versions of the material or the use of the work.

This Training Guide has been produced with the financial assistance of the German Federal Ministry for Economic Cooperation and Development (BMZ). The contents of this document are the sole responsibility of the authors and can under no circumstances be regarded as reflecting the position of the BMZ, GIZ, or FOSSFA.

This set of material has been compiled with great care. Nevertheless, it cannot be guaranteed that the information shown is up-to-date, complete and correct. Consequently no liability can be accepted for any damages arising directly or indirectly as a result of the use of this set of material, unless such damages are the result of intent or of gross negligence.

FOSSFA/GIZ are not responsible for the contents of websites to which hyperlinks exist within this set of material, unless FOSSFA/GIZ has full knowledge of illegal contents and it is possible to prevent visitors of this site from viewing those pages. Neither can FOSSFA/GIZ accept any liability or give any guarantees for external links. If you are of the opinion that certain external websites to which links exist within this material contravene existing legislation or contain otherwise inappropriate materials we would ask you to inform us.

Table of Contents

Who this Training Guide is for.....	5
What you can learn through this Training Guide.....	6
Outline and Timeline.....	7
Classroom Setup Guide.....	9
About ict@innovation - Creating Business and Learning Opportunities with Free and Open Source Software in Africa.....	10
Join ict@innovation: Meet the Linux Admin and Certification Trainers community! Find a local training offer!.....	11
Find more information on Linux Certification in Africa.....	12
AUTHORS, ATTRIBUTIONS AND LICENSING.....	13
How you can reuse this guide – License and invitation to share.....	13
Full attribution – Who built this guide and its source material.....	13
Chapter I: LPIC 101.....	25
Topic 101: System Architecture.....	26
101.1 Determine and Configure Hardware Settings.....	26
Introduction.....	26
BIOS.....	27
Enabling/Disabling Integrated Devices.....	27
IRQ, IO Addresses and DMA Addresses.....	28
Interrupt Requests.....	28
Input/Output Addresses.....	30
DMA Addresses.....	32
Configuring IRQs, IO Ports/Addresses and DMA.....	33
Linux Device Management Overview.....	34
Mass Storage Devices.....	35
Hot plug & Cold plug Devices.....	38
Device Drivers.....	40
101.2 Boot the System.....	43
Introduction.....	43
Boot Parameters.....	44
Troubleshooting The Boot Process.....	48
101.3 Runlevels, Rebooting & Shutting down the system.....	50

Introduction.....	50
System V init.....	51
Systemd init.....	51
Upstart init.....	52
Configuring Runlevels.....	53
Changing Runlevels.....	54
Starting & Stopping Services.....	55
Topic 102: Linux Installation and Package Management.....	57
102.1 Design hard disk layout.....	57
Introduction.....	57
Hard Drive Partitions.....	57
Common Partition Schemes.....	58
How to Partition Using fdisk.....	59
Logical Volume (LV).....	62
Basic LVM Commands:.....	63
What can it do for me?.....	63
102.2 Install a boot manager.....	64
Introduction.....	65
Grub Boot Loader.....	65
102.3 Manage shared libraries.....	71
Introduction.....	71
The ldd command.....	72
102.4 Debian Package Management.....	74
Introduction.....	74
Package Naming.....	74
Files.....	76
APT.....	77
The Alien Tool.....	78
102.5 RPM and YUM Package Management.....	79
Introduction.....	79
Yum Package Manager.....	82
Topic 103: GNU and Unix Commands.....	84
103.1 Working on the Command Line.....	84
Overview.....	85

The interactive shell.....	85
Shell Variables.....	86
Wildcards.....	87
The Command History.....	89
Manpages and the whatis database.....	90
103.2 Process text streams using filters.....	94
Text Processing Utilities.....	94
Formatting output with fmt and pr	97
103.3 Perform basic file management.....	99
Moving Around the Filesystem.....	99
File Archiving and Compression.....	102
103.4 Streams, Pipes and Re-directs.....	105
Input, Output, Redirection.....	105
Piped Commands.....	106
103.5 Create, Monitor and Kill Processes.....	108
Starting and Stopping Jobs.....	108
Viewing Running Processes.....	109
Sending Signals To Processes.....	111
103.6 Modify Process Execution Priorities.....	111
103.7 Using Regular Expressions.....	114
Overview.....	114
The grep family.....	115
103.8 Using Vi.....	118
Modes.....	118
Text Items.....	119
Running a Shell Command.....	121
Topic 104: Devices, Linux Filesystems, Filesystem Hierarchy Standard.....	123
104.1 Create Partitions and Filesystems.....	123
Linux File Systems.....	123
File System Formatting.....	124
104.2 Maintain the integrity of filesystems.....	127
Monitoring Disk Usage.....	127
File System Checking, Repair and Maintenance.....	127
104.3 Control mounting and unmounting of filesystems.....	132

Manually Mounting and Unmounting Filesystems.....	133
104.4 Manage Disk Quotas.....	135
104.5 Manage File Permissions and Ownership.....	137
Changing permissions and owners.....	139
SGID permissions.....	141
104.6 Create and change hard and symbolic links.....	143
104.7 Finding and Placing Files within the File Hierarchy Standard	145
The Linux File System.....	145
Finding Files and Directories.....	145
Chapter II: LPIC 102.....	147
Topic 105: Shells, Scripting and Data Management.....	148
105.1 Customize and use the shell environment.....	148
Introduction.....	148
Summary of Common Commands.....	149
Aliases.....	151
Functions.....	151
set.....	152
105.2 Customize or Write Simple Scripts.....	154
What is a shell script?.....	154
Special Parameters.....	154
Conditional statements.....	156
Shell functions.....	158
Conditional Expressions.....	160
105.3 SQL Data Management.....	162
Common SQL Implementations for Linux.....	162
Basic SQL Commands.....	162
Topic 106: User Interfaces and Desktops.....	169
106.1 Install and Configure X11.....	169
Introduction.....	169
Configuring X11R6.....	170
Controlling X clients.....	172
Choosing a Window Manager.....	174
106.2 Set Up a Display Manager.....	176
The Display Manager.....	176

106.3 Accessibility.....	182
Keyboard and Mouse Accessibility Options.....	182
Screen Display Settings.....	184
Using Additional Assistive Technologies.....	186
Topic 107: Administrative Tasks.....	188
107.1 Manage User and Group Accounts.....	188
Creating New Users.....	188
Working with Groups.....	189
Modifying accounts and default settings.....	193
107.2 Automate System Administration Tasks.....	196
Cron Jobs.....	196
107.3 Localization and Internationalization.....	201
Topic 108: Essential System Services	206
108.1: Maintain System Time.....	206
System Time and the Hardware clock.....	206
108.2 System Logging.....	209
Log Utilities.....	212
108.3 Mail Transfer Agent (MTA) basics.....	214
Mail system architecture.....	214
Sendmail.....	215
Aliases and mail forwarding.....	216
108.4 Managing Printers and Printing.....	220
Introducing CUPS.....	220
Legacy commands for printing.....	225
Topic 109: Networking Fundamentals.....	228
109.1 Fundamentals of Internet Protocols.....	228
IP addresses and the Dotted Quad Notation.....	228
Binary numbers.....	228
Broadcast Address, Network Address and Netmask.....	229
Routing.....	233
IPV6 Basics.....	234
Phase 1 – Link-Local Address.....	238
Phase 2 – Global Address.....	238
109.2 Basic Network Configuration.....	246

The Network Interface.....	246
Network configuration.....	247
Routing.....	250
109.3 Basic Network Troubleshooting.....	252
109.4 Configure client-side DNS.....	260
Some background on DNS.....	261
Topic 110 Security.....	264
110.1 Perform Security Administration Tasks.....	264
Best Practices.....	268
110.2 Setting Up Host Security.....	275
110.3 Securing data with encryption.....	282
Introducing ssh.....	282
User Authentication.....	284
Port forwarding.....	287
GnuPG.....	288
Digital Signatures.....	291
Chapter III: FOSS TRAINING.....	294
Introduction.....	294
Learning Objectives.....	294
Sessions and Timetable.....	295
Module III.1: How to be a FOSS Trainer.....	296
Duration:.....	296
III.1.1 FOSS Trainer Characteristics.....	296
III.1.2 Types of Training Interventions.....	296
Questions.....	297
Exercise.....	297
Module III.2: FOSS Training as a Business.....	297
Duration:.....	297
III.2.1 Identifying FOSS Business Opportunities.....	297
III.2.2 Case Study.....	298
III.2.3 Identifying Training Opportunities.....	299
III.2.4 Marketing of Training Courses.....	299
III.2.5 FOSS Certifications.....	299
Questions.....	302

Exercise 1.....	302
Exercise 2.....	302
Module III.3: Organising Trainings.....	302
Duration:.....	302
III.3.1 Course Design and Curriculum Development.....	303
III.3.2 Course Material Development.....	303
III.3.3 Licensing of Course Material.....	303
III.3.4 Preparing Yourself for Class.....	304
III.3.5 Preparing Your Training Room.....	306
III.3.6 Beginning the Training Session.....	306
III.3.7 Ending Your Training Session.....	307
Questions.....	307
Exercise.....	307
Module III.4: Open Educational Resources and Open Content.....	308
Duration:.....	308
6.4.1 Open Educational Resources.....	308
III.4.2 Open Content.....	309
Questions.....	310
Exercise.....	310
Module III.5: Communication Skills.....	311
Duration:.....	311
III.5.1 The Four Learning Styles.....	311
My Training Style.....	314
III.5.2 Presenting Information.....	322
III.5.3 Using Your Body Effectively.....	323
III.5.4 Building Rapport with Eye Contact.....	324
III.5.5 Enhancing Voice Quality.....	325
Questions.....	327
Exercise.....	328
Partners.....	334
Introducing ict@innovation.....	336
The community webportal.....	337
About this Training Guide on Linux System Administration, LPI Certification Level 1	338

Chapter I: LPIC 101

Topic 101: System Architecture

101.1 Determine and Configure Hardware Settings

Candidates should be able to configure fundamental system hardware by making the correct settings in the system BIOS in x86 based systems

Key Knowledge Areas

- Enable and disable integrated peripherals.
- Configure systems with or without external peripherals such as keyboards.
- Differentiate between the various types of mass storage devices.
- Set the correct hardware ID for different devices, especially the boot device.
- Know the differences between coldplug and hotplug devices.
- Determine hardware resources for devices.
- Tools and utilities to list various hardware information (e.g. `lsusb`, `lspci`, etc.)
- Tools and utilities to manipulate USB devices
- Conceptual understanding of `sysfs`, `udev`, `hal`, `dbus`

Introduction

A computer system consists of a central CPU, primary storage or memory, secondary or permanent storage such as a hard disk, and various input/output devices. In order to operate the CPU needs to be able to load instructions and data into the CPU from memory, possibly having to fetch it from the secondary storage first, execute the instructions and then store the results back in memory. It communicates with the various input/output or peripheral devices via a data path known as bus. A computer system may have more than one bus which it uses to communicate with different components.

This simplified model of the modern computer is known as the Von Neumann architecture after John von Neumann a Hungarian born mathematician who developed the basic architecture of modern computers in the 1940s.

With this basic conceptual model as a basis we can begin to understand how the CPU determines and configures the numerous peripherals and devices that make up a computer system, and how the operating system manages and coordinates activities for sharing system resources and access to its connected peripherals and devices.

These resources and peripherals include mass storage devices (secondary storage), and the input/output devices such as monitors, keyboards and network cards. Some peripherals are integrated into the motherboard of PCs, such as parallel and com ports, and even VGA or network cards in more modern ones while others are external devices such as USB sticks or bluetooth dongles. All these devices need a way to communicate with the CPU to provide it with information, to request services from the CPU or to receive instruction from the CPU.

When configuring a computer system you need to know how to find the current settings of these peripherals, what possible values their settings can have and how to change them in the BIOS or operating system if necessary. Configuring PC peripherals to work with an operating system is done in two places. The first place configuration is done is by the system firmware or BIOS (Basic Input/Output System); the second is by the operating system.

BIOS

The purpose of the BIOS (*Basic Input/Output System*) is to perform a Power On Self Test (POST), identify and initialize system devices and peripherals and start the process to load the operating system by loading the boot loader from the boot device. The BIOS also provides an abstraction layer to the operating system for accessing system devices. The intention of this layer is to insulate the operating system from having to deal with the wide variety of devices available today but it is ignored by most modern operating systems, including Linux, which access the devices with their own device drivers.

Most BIOS manufactures provide a console based user interface that allows you to configure the low-level system settings for devices. The BIOS configuration interface, and how it is accessed, varies from manufacturer to manufacturer but is usually accessed by pushing a key or key combination, such as delete or insert during system boot.

It is under the BIOS configuration console that you can:

- enable or disable devices,
- allocate resources such as IRQ and IO addresses,
- select the boot order of devices and,
- change settings that affect operating modes of devices such as disk drives and network cards

Besides the main BIOS many peripherals also have their own BIOS firmware and configuration consoles. Some network cards, and most SCSI host adapter cards come with their own BIOS which can be used to set their configuration. Like the main system BIOS these firmwares also perform some low level checking and initialization of their devices.

Enabling/Disabling Integrated Devices

Sometimes it may be necessary to disable a device which is preventing the operating system from installing or working properly. This is a rare occurrence, and can usually be remedied by passing the correct parameters to the Linux kernel at boot time or changing a parameter setting in the BIOS. Usually the setting which can give the most problems relate to hard disk access modes, power management and interrupt controller settings. Often the cause of these problems are bugs in the firmware itself.

Most BIOS console allow one to disable integrated peripherals such as COM ports, video or network cards. Even if these devices are not causing any installation or boot up issues you may wish to disable them if they are unused to free up resources that would otherwise be unavailable for other purposes.

Sometimes it is necessary to disable system checks for peripherals such as a keyboard or a mouse which, although necessary for the proper operation of desktop machines, are often not present for servers. Machines without keyboards, mouse or monitors are referred to as

“headless” systems. Some BIOSes will refuse to boot if these devices are not present unless these system checks are disabled. (You may wonder how headless machines are accessed at all if they don't have a keyboard, mouse or monitor. The answer is via the network with utilities such as SSH which is covered later in this book.) Of course to access the BIOS itself it is necessary to have peripherals such as a keyboard and monitor present!

IRQ No.	Hardware Assignment						
0	System timer	4	COM1	8	Real Time Clock	12	PS2 Mouse
1	Keyboard	5	LPT2 / Sound Card	9	Available	13	Floating Point Proc
2	Handles IRQ 8 - 15	6	Floppy Controller	10	Available	14	Primary IDE
3	COM2	7	Parallel Port	11	Available	15	Secondary IDE

Table 101.1.1: IRQ Default Allocation

IRQ, IO Addresses and DMA Addresses

To understand how peripheral devices communicate with the CPU you need to know about interrupt requests (IRQs), Input/Output (IO) addresses and direct memory access (DMA) addresses. The CPU is responsible for processing all instructions and events that occur in the system. To communicate with a device it needs to know when a device has an event for it to handle and it needs to be able to pass information to and from the device.

IRQs are the mechanism by which peripherals tell the CPU to suspend its current activity and to handle its event such as a key press or a disk read. IO addresses are regions of memory mapped to devices where the CPU can write information of the devices attention and read from the devices as well. This is a somewhat simplified explanation of how peripheral devices communicate with the CPU but suits our purposes for understanding IRQ and IO Addresses.

Interrupt Requests

When an event occurs on a device, such as a mouse movement or data arriving from a USB connected drive, the device signals to the CPU that it has data which it needs to handle by generating an interrupt on the bus. Before the advent of Plug and Play technology, which requires both a hardware (PCI bus) and software component to work, Intel PCs were limited to 16 possible IRQ settings and many of these IRQ lines were preset for devices, so that of those, only a few are freely available. The table below lists the IRQ's that cannot be used in red and the IRQ's that could be reassigned providing certain hardware does not exist in your system in orange, and those that you are free to assign as you please in white.

- IRQ 0 system timer (cannot be changed);
- IRQ 1 keyboard controller(cannot be changed);
- IRQ 2 cascaded signals from IRQs 8–15 devices configured to use IRQ 2 will use IRQ 9
- IRQ 3 serial port controller for COM2 (shared with COM4, if present);

- IRQ 4 serial port controller for COM1 (shared with COM3, if present);
- IRQ 5 LPT port 2 or sound card;
- IRQ 6 floppy disk controller;
- IRQ 7 LPT port 1 or sound card (8bit Sound Blaster and compatibles).
- IRQ 8 realtime clock;
- IRQ 9 open interrupt / available or SCSI host adapter;
- IRQ 10 open interrupt / available or SCSI or NIC;
- IRQ 11 open interrupt / available or SCSI or NIC;
- IRQ 12 mouse on PS/2 connector;
- IRQ 13 math coprocessor / integrated floating point unit or interprocessor interrupt (use
- IRQ 14 primary ATA channel (disk drive or CDROM);
- IRQ 15 secondary ATA channel

As the number and types of peripheral devices grew, the limited number of IRQ lines became a problem. Even if there was a free IRQ some devices were hardwired to use a specific IRQ that may already have been in use. Some devices allowed for manual configuration via jumper settings and could have IRQ, as well as IO addresses re-assigned in the BIOS.

To overcome this limitation “plug and play” technology was introduced which allows devices to share interrupt lines thus expanding the number of devices that could be accommodated. In addition “plug and play” did away with the need for manual configuration of devices. With Intel's Advanced Programmable Interrupt Controller (APIC) architecture there are now usually 24 interrupt lines available that can accommodate up to 255 devices.

In order to see the allocation of IRQs to devices in a Linux system, you can examine the contents of `/proc/interrupts` file.

From this point onward it becomes necessary to have access to a Linux PC. Although some theory is involved, we shall be interacting with Linux more and more. I advise that you attempt the commands as you come across them, testing your understanding as you go.

```

CPU0      CPU1
0:         526      739  IO-APIC-edge    timer
1:          2        2  IO-APIC-edge    i8042
3:          1        1  IO-APIC-edge
4:          1        1  IO-APIC-edge
7:          0        0  IO-APIC-edge    parport0
8:          1        0  IO-APIC-edge    rtc0
9:          0        0  IO-APIC-fasteoi acpi
12:         4        2  IO-APIC-edge    i8042
14:       183025    218782 IO-APIC-edge    ata_piix
15:       634399    658512 IO-APIC-edge    ata_piix
16:     11830673      373  IO-APIC-fasteoi uhci_hcd:usb5, HDA Intel, nvidia
18:          0        0  IO-APIC-fasteoi uhci_hcd:usb4
19:          0        0  IO-APIC-fasteoi uhci_hcd:usb3
23:         46    1311280 IO-APIC-fasteoi ehci_hcd:usb1, uhci_hcd:usb2
27:          3    1577694 PCI-MSI-edge    eth1
NMI:          0        0  Non-maskable interrupts
LOC:    29061412    9378470  Local timer interrupts
SPU:          0        0  Spurious interrupts
PMI:          0        0  Performance monitoring interrupts
PND:          0        0  Performance pending work
RES:    19421375    14071480 Rescheduling interrupts
CAL:         221        854  Function call interrupts
TLB:     38533      68238  TLB shootdowns
TRM:          0        0  Thermal event interrupts
THR:          0        0  Threshold APIC interrupts
MCE:          0        0  Machine check exceptions
MCP:         279      279  Machine check polls
ERR:          1
MIS:          0

```

Figure 101.1..1: Sample result of /proc/interrupts

What is the /proc file system?

The proc file system is a virtual or pseudo file system. The kernel sets up the /proc file system to export information about the running kernel, user processes and hardware devices that have been configured. It is a virtual file system in that it exists in memory only and does not represent any true physical files. Treating everything as a file is a cornerstone of UNIX design philosophy and many such pseudo file systems exist. Others are the /sys and /dev directories.

Input/Output Addresses

When the CPU and peripheral devices need to communicate and/or pass data between each other they make use of IO addresses. Essentially they communicate by reading and writing data to the reserved IO addresses of the device. There are two complementary methods for performing IO between the CPU and device. There is memory mapped IO (MMIO) and port IO (PIO) addressing.

In memory mapped IO, addresses are regions of memory reserved for communication between the CPU and a particular device. It is important that these memory regions are not used by any other processes. In port mapped IO addressing the CPU has a separate set of instructions for

performing IO with a devices having a separate address space.

The allocation of IO ports in Linux can be revealed by examining the contents of the `/proc/ioports` file.

```
0000-001f : dma1
0020-0021 : pic1
0040-0043 : timer0
0050-0053 : timer1
0060-0060 : keyboard
0064-0064 : keyboard
0070-0073 : rtc0
0080-008f : dma page reg
00a0-00a1 : pic2
00c0-00df : dma2
00f0-00ff : fpu
0170-0177 : 0000:00:1f.2
  0170-0177 : ata_piix
01f0-01f7 : 0000:00:1f.2
  01f0-01f7 : ata_piix
0290-029f : pnp 00:01
  0290-0294 : pnp 00:01
02f8-02ff : serial
0376-0376 : 0000:00:1f.2
  0376-0376 : ata_piix
0378-037a : parport0
03c0-03df : vga+
03f6-03f6 : 0000:00:1f.2
  03f6-03f6 : ata_piix
03f8-03ff : serial
0400-047f : 0000:00:1f.0
  0400-0403 : ACPI PM1a_EVT_BLK
  0404-0405 : ACPI PM1a_CNT_BLK
  0408-040b : ACPI PM_TMR
  0410-0415 : ACPI CPU throttle
  0428-042f : ACPI GPE0_BLK
0480-04bf : 0000:00:1f.0
04d0-04d1 : pnp 00:01
```

Figure 101.1.2: Sample of `/proc/ioports`

For allocation of IO memory you can look at `/proc/iomem`, an example of the contents of this file is given below.

```

00000000-00000fff : System RAM
00001000-00005fff : reserved
00006000-00009fff : System RAM
00009f800-00009ffff : reserved
0000d1800-0000d3fff : pnp 00:0c
0000e0000-0000effff : pnp 00:0c
0000f0000-0000ffff : reserved
00100000-bfedffff : System RAM
  01000000-0154baa8 : Kernel code
  0154baa9-018349cf : Kernel data
  0191e000-01a2fe63 : Kernel bss
bfee0000-bfee2fff : ACPI Non-volatile Storage
bfee3000-bfeeffff : ACPI Tables
bfef0000-bfefffff : reserved
bff00000-bfffffff : RAM buffer
c0000000-cfffffff : PCI MMCONFIG 0 [00-ff]
  c0000000-cfffffff : reserved
  c0000000-cfffffff : pnp 00:0b
d0000000-dfffffff : PCI Bus 0000:01
  d0000000-dfffffff : 0000:01:00.0
e0000000-e2ffffff : PCI Bus 0000:01
  e0000000-e0ffffff : 0000:01:00.0
  e0000000-e0ffffff : nvidia
  e1000000-e1ffffff : 0000:01:00.0
  e2000000-e201ffff : 0000:01:00.0
e3000000-e3ffffff : PCI Bus 0000:03
e4000000-e40ffffff : PCI Bus 0000:03

```

Figure 101.1.3: Sample of /proc/iomem

Prior to “plug and play” technology you would have to configure IO devices addresses in the BIOS and operating system but with the introduction of “plug and play” the allocation is now done automatically by the operating system and the bus.

DMA Addresses

DMA stands for direct memory access and is an optimization that allows devices to read and write directly to memory without having to go through the CPU. Traditionally when the CPU requests data to be read from a device into memory, the CPU needs to be involved in the process of transferring the data. Under this model, called Programmed IO (PIO), significant CPU time is spent simply copying data between the device and memory. With DMA devices can write directly to memory, by-passing the CPU. This greatly enhances system performance. To examine the allocation of DMA addresses on a Linux box you can examine the contents of /proc/dma

```

3: parport0
4: cascade

```

Figure 101.1.4: Sample of /proc/dma output

DMA is automatically configured by the operating system for most devices. A significant device where DMA may not be automatically configured is on the system parallel ata (PATA) disks. For PATA devices (see section below) DMA access can be enabled, assuming your device is /dev/hda (SATA drives which are popular today do not use DMA in the conventional sense) by running the command:

```
hdparm -d1 /dev/sda
```

To see the current setting on your hard disk you can run the command

```
hdparm -d /dev/sda
```

Configuring IRQs, IO Ports/Addresses and DMA

Configuration for device IRQ, IO addresses, DMA channels and memory regions happens automatically with today's plug and play PCI bus. Resources are allocated via the system BIOS, via the Linux kernel and possibly the device driver with resource conflicts being automatically resolved in most cases.

Linux provides various utilities to query devices to find out the resources that they use. A lot of these utilities make use of the information exported by the kernel in the /proc file system.

Two commands used to query PCI devices are the **lspci** and **setpci** commands. The **lspci** utility can provide verbose information on devices using the PCI bus, depending on which parameters you use.

“**lspci -vv**” provides detailed output on PCI devices. The output below shows typical output from the “**lspci**” command with no parameters.

```

~$ lspci
00:00.0 Host bridge: Intel Corporation Mobile 4 Series Chipset Memory Controller Hub (rev 07)
00:02.0 VGA compatible controller: Intel Corporation Mobile 4 Series Chipset Integrated Graphics Controller (rev 07)
00:02.1 Display controller: Intel Corporation Mobile 4 Series Chipset Integrated Graphics Controller (rev 07)
00:1a.0 USB Controller: Intel Corporation 82801I (ICH9 Family) USB UHCI Controller #4 (rev 03)
00:1a.1 USB Controller: Intel Corporation 82801I (ICH9 Family) USB UHCI Controller #5 (rev 03)
00:1a.7 USB Controller: Intel Corporation 82801I (ICH9 Family) USB2 EHCI Controller #2 (rev 03)
00:1b.0 Audio device: Intel Corporation 82801I (ICH9 Family) HD Audio Controller (rev 03)
00:1c.0 PCI bridge: Intel Corporation 82801I (ICH9 Family) PCI Express Port 1 (rev 03)
00:1c.1 PCI bridge: Intel Corporation 82801I (ICH9 Family) PCI Express Port 2 (rev 03)
00:1c.2 PCI bridge: Intel Corporation 82801I (ICH9 Family) PCI Express Port 3 (rev 03)
00:1c.4 PCI bridge: Intel Corporation 82801I (ICH9 Family) PCI Express Port 5 (rev 03)
00:1d.0 USB Controller: Intel Corporation 82801I (ICH9 Family) USB UHCI Controller #1 (rev 03)
00:1d.1 USB Controller: Intel Corporation 82801I (ICH9 Family) USB UHCI Controller #2 (rev 03)
00:1d.2 USB Controller: Intel Corporation 82801I (ICH9 Family) USB UHCI Controller #3 (rev 03)
00:1d.3 USB Controller: Intel Corporation 82801I (ICH9 Family) USB UHCI Controller #6 (rev 03)
00:1d.7 USB Controller: Intel Corporation 82801I (ICH9 Family) USB2 EHCI Controller #1 (rev 03)
00:1e.0 PCI bridge: Intel Corporation 82801 Mobile PCI Bridge (rev 93)
00:1f.0 ISA bridge: Intel Corporation ICH9M LPC Interface Controller (rev 03)
00:1f.2 SATA controller: Intel Corporation ICH9M/M-E SATA AHCI Controller (rev 03)
00:1f.3 SMBus: Intel Corporation 82801I (ICH9 Family) SMBus Controller (rev 03)
00:1f.6 Signal processing controller: Intel Corporation 82801I (ICH9 Family) Thermal Subsystem (rev 03)
04:00.0 Network controller: Atheros Communications Inc. AR928X Wireless Network Adapter (PCI-Express) (rev 01)
05:00.0 Ethernet controller: Realtek Semiconductor Co., Ltd. RTL8111/8168B PCI Express Gigabit Ethernet controller (rev 02)
07:00.0 System peripheral: JMicron Technology Corp. SD/MMC Host Controller
07:00.2 SD Host controller: JMicron Technology Corp. Standard SD Host Controller
07:00.3 System peripheral: JMicron Technology Corp. MS Host Controller
07:00.4 System peripheral: JMicron Technology Corp. xD Host Controller

```

Figure 101.1.5: Sample *lspci* output

All PCI devices are identified by a unique ID. This ID is made up of a unique vendor ID, device ID with potentially a sub-system vendor ID and sub-system device ID. When “**lspci**” is run the PCI ID is looked up in the systems ID database and translated into a human readable format, showing

the vendor name and device. The PIC ID database is found at `/usr/share/misc/pci.ids` on Ubuntu and on RedHat. When a PCI ID is not found in the database it is displayed as two numbers separated by a colon. In such cases it is a good bet that the correct device drivers for this peripheral are not loaded either. Searching the Internet for the PCI ID may reveal the manufacturer, device ID and chipset information with which it could be possible to identify the correct driver for the device. Alternatively no driver for Linux has been written for the device yet. `setpci` is a utility for configuring and querying PCI devices. It is an advanced utility that is beyond the scope of this manual but you should be aware of its existence.

How to reassign IRQ Allocation in a PnP BIOS/OS

Even though IRQ conflicts are now a thing of the past, as interrupts can be shared, it can still be a bad idea for some devices with a high frequency of interrupts to share the same IRQ. This is particularly true if the devices are hard disks or sound devices.

In this case it is recommended to try and move the PCI cards to different slots to try and get them assigned different IRQs.

Linux Device Management Overview

Before going on to discuss the various types of mass storage devices that you will typically encounter when running Linux, it is beneficial to have an overview of how Linux manages devices. Many of the concepts introduced here will be expanded on later in this manual, but a high level overview will provide you with a map, so to speak, of how it all fits together.

A device driver is a kernel space program that allows user space applications to interact with the underlying hardware. Kernel space refers to privileged code which has full access to hardware and runs in ring 0 which is a hardware enforced privileged execution mode. User space applications run in a less privileged mode, ring 3, and cannot access hardware directly. User space application can only interact with hardware by making system calls to the kernel to perform actions on their behalf.

The Linux kernel exports information about the devices from the device drivers to a pseudo filesystem mounted under `/sys`. This file system tells user space what devices are available. It is populated at system boot but when a hot plug device is inserted or removed the `/sys` file system is updated and the kernel fires events to let user space know that there has been a change.

One of the principles of UNIX design is to use the metaphor of a file where ever possible, providing a standard conceptual model for interacting with various UNIX components where its a real file, printer, disk or keyboard using standard input/output system calls. The file interface for device drivers is exported under the `/dev` directory.

One of these user space applications is the kernel device manager, `udev`, which creates a device node under the `/dev` directory to allow user space application access to the device. The name of the device node or file is determined by the device drivers naming convention or by user defined rules in `/etc/udev/rules.d/`. It is through the entries under `/dev` that user space application can interact with the device driver.

The `udev` daemon is also responsible for informing other user land applications of changes. The applications that are most important here are the HAL (hardware abstraction layer) daemon, and D-Bus (desktop bus). These applications are mainly used by desktop environment to carry out tasks when an event occurs such as open the file browser when a USB drive is inserted or image application when a camera is inserted.

While `udev` creates the relevant entries under the `/dev` file system, if anything useful need to happen when the event occurs HAL and D-Bus are needed. (Note: HAL is now deprecated as it is being merged into `udev`). HAL is a single daemon responsible for discovering, enumerating and mediating access to most of the hardware on the host computer for desktop applications to which it provides a hardware abstraction layer. Application register with the D-Bus daemon to receive notifications of events and also post event notifications that other applications may be interested in. D-Bus is used for example to launch media players when a audio CD is inserted and to notify other applications of the currently playing song for example.

From a practical point of view, the service which impacts on you the most will be `udev` as it created device nodes under the `/dev` directory. For LPI it is important to be able to identify which device are available.

Mass Storage Devices

The secondary storage or mass storage devices, as they are known today come in different types which are determined by their physical interface. An interface is the means by which the device physically attaches to the computer. Over time numerous interfaces for connecting mass storage devices have been developed but the four main types of disks encountered today are :

- **PATA** – Parallel Advanced Technology Attachment , also known as IDE
- **SATA** – Serial Advanced Technology Attachment, the latest standard replacing PATA especially on desktops and laptops,
- **SCSI** – Small Computer System Interface disks are used in servers and other high end machines. SCSI provides high speed access as well as the ability to connect a large number of devices
- **SAS** – serial SCSI, a new SCSI standard aimed at servers

Besides the interface, the type of mass storage is also determined by the device itself. For example you can have SATA optical drives (CDROMs/DVDROMS) as well as hard disk and SCSCI disks and SCSI tape drives.

PATA

PATA is an obsolete standard but you might still encounter it on older machines. Parallel refers to the manner in which data is transferred from the device to the CPU and memory. There are several variations on the PATA standard such as IDE and EIDE but since the introduction of SATA they are collectively referred to as PATA devices.

Motherboards come with two PATA connectors and PATA cables support up to two devices per cable, in a master/slave setup. Which device is master or slave depends on the location of the device on the cable and jumper settings on the hard disks themselves. Since ATA drives have been around for a long time they are well supported in Linux and most BIOSes usually have no problem in automatically identifying and configuring these devices.

Linux identifies ATA devices under the `/dev` file system with the naming convention `/dev/hd[a-z]` . The last letter of the device name is determined by the whether the disk is master or slave on the primary or secondary connection.

Gaps can also appear in the device node naming. For example it is often the case that the disk is identified as `/dev/hda`, being the master drive on the primary connector, with the CDROM drive being identified as `/dev/hdc` as the master on the secondary connector. This is quiet a common setup for desktop machines as it is better to have your two most frequently accessed mass storage devices on separate cables for improved performance rather than having them share a

cable where access contention may arise.

Partitions on PATA disks are identified by a number following the letter. For example the first primary partition on the `/dev/hda` drive is identified by `/dev/hda1` and the 2nd primary partition `/dev/hda2` etc. For more information on disk partition numbering please refer to section 102.1

What is important about understanding device node naming conventions at this point is being able to identify the type of hard disk device and its device name.

SATA

Serial ATA (SATA) drives have largely replaced PATA drives on desktops and laptops. SATA is a serial standard but offers higher throughput than the older PATA interface. SATA drives are not configured in a master/slave setup and each have their own dedicated controller or channel. The cables for SATA drives are considerably thinner than those for PATA devices saving space and cost. SATA controllers use the Advanced Host Controller Interface (AHCI) which allows for hotplugging and hotswapping of SATA disk.

As with PATA devices most BIOSes automatically detect SATA drives and the Linux kernel usually has no problem identifying and loading the correct drivers for SATA drives. A peculiarity of SATA under Linux is that SATA disks make use of the SCSI disk sub system and hence the naming convention of these devices follows that of SCSI devices.

SATA drive naming conventions uses the scsi sub system naming conventions with devices being labeled `/dev/sd[a-z]`. The final character of the device node name is determined by the order in which the Linux kernel discovers these devices. Partitions on SATA drives are named numbered from 1 upwards.

SCSI Devices

SCSI like PATA and SATA defines a physically interface as well as protocol and commands for transferring data between computers and peripheral devices. SCSI is most commonly used for hard disks and tape drives, but it can connect a wide range of other devices, such as scanners and CD drives. Usually SCSI devices connect to a host adapter that has its own BIOS. SCSI storage devices are faster and more robust than SATA or PATA devices but are also more expensive, hence they are used mostly in servers or high end workstations.

There are two types of SCSI interfaces: an 8-bit interface with a bus that supports 8 devices, this includes the controller, so there is only space for 7 devices and a 16-bit interface (WIDE) that supports 16 devices including the controller, so there can only be 15 block devices.

SCSI devices are uniquely identified using a set of 3 numbers called the SCSI ID.

- a. the SCSI channel
- b. the device ID number
- c. the logical unit number LUN

The SCSI Channel

Each SCSI adapter supports one data channel on which to attach SCSI devices (disc, CDROM, etc)

These channels are numbered from 0 onwards.

Device ID number

Each device is assigned a unique ID number that can be set using jumpers on the disk. The IDs range from 0 to 7 for 8-bit controllers and from 0 to 15 for 16-bit controllers.

Logical Units

The Logical Unit Number (LUN) is used to differentiate between devices within a SCSI target number. This is used, for example, to indicate a particular partition within a disk drive or a particular tape drive within a multi-drive tape robot. It is not seen so often these days as host adapters are now less costly and can accommodate more targets per bus.

Hardware Detection

All detected devices are listed in the `/proc/scsi/scsi` file. The example below is from the SCSI-2.4-HOWTO

`/proc/scsi/scsi`

```
Attached devices:
Host: scsi0 Channel: 00 Id: 02 Lun: 00
  Vendor: PIONEER Model: DVD-ROM DVD-303 Rev: 1.10
  Type:   CD-ROM          ANSI SCSI revision: 02
Host: scsi1 Channel: 00 Id: 00 Lun: 00
  Vendor: IBM Model: DNES-309170W Rev: SA30
  Type:   Direct-Access   ANSI SCSI revision: 03
```

Since SATA drives use the same scsi sub-system as real SCSI drives, the naming convention for SCSI devices is the same as that set out above for SATA drives. It is not only scsi drives that make use of the scsi sub system but USB drives as well, hence you will find USB drives following the same naming convention

The `scsi_info` tool uses the information in `/proc/scsi/scsi` to print out the SCSI_ID and the model of a specified device. From the file above `scsi_info` would produce the following output:



```
# scsi_info /dev/sda
SCSI_ID="0,0,0"
MODEL="IBM DNES-309170W"
FW_REV="SA30"
```

The system will boot from the device with SCSI ID 0 by default. This can be changed in the SCSI BIOS which can be configured at boot time. If the PC has a mixture of SCSI and SATA/PATA disks, then the boot order must be selected in the system's BIOS first.

SAS

Serial attached SCSI is the latest interface on the block and is an upgrade to the SCSI protocol and interface as SATA is to PATA. In general SCSI devices are faster and more reliable than SATA/PATA drives. The performance gap between the two technologies continues to close but the two technologies are generally targeted at two different markets, namely the consumer market for SATA and the enterprise for SCSI. The enterprise has a higher requirement for reliability and speed than does the consumer market and the price of the different drives reflect this.

Identifying the correct device ID for BOOT device

The purpose of listing the different type of mass storage devices and their naming conventions is to allow you to easily identify the device ID for your disks. This is important for being able to identify which of your drives is the boot device and which disk partitions contains the root and boot directories.

There is however a problem with the Linux naming convention, and not just for disk drives. The problem is that device names may change between system reboots when hardware configuration changes are made. Since the naming convention of hardware has a component which depends on the order the device is discovered by the kernel, adding, moving or removing devices may result in changes to device names. This was not such a problem a few years ago, as changing hard disks is not a regular occurrence; but with the advent of USB and the fact that it uses the SCSI sub-system for its device naming, the problem has become more severe. In order to uniquely identify a device, irrespective of when it is discovered or located, away has to be found to uniquely identify the device. For disks this is done by writing a universally unique id to the disk meta data and using the UUID in configuration files rather than the device node. (It is not only hard disks that need to be uniquely identifiable. Other devices need to be as well but may use different means of doing so. Network cards use their MAC address for example.) The **blkid** utility, which replaces **vol_id**, can be used to query the settings on a device or search for a device with a specific UUID.

blkid /dev/sda1

for example produces the following output: `/dev/sda1: UUID="c7d63e4b-2d9f-450a-8052-7b8929ec8a6b" TYPE="ext4"` showing that the first partition on the device sda has an ext4 filesystem and a UUID of `c7d63e4b-2d9f-450a-8052-7b8929ec8a6b`

blkid -U 75426429-cc4b-4bfc-beb9-305e1f7f8bc9

searches for the device with the specified ID and returns `/dev/sdb1`. Alternatively you could look under `/dev/disk/by-uuid` to see which ids map to which devices as seen by the Linux kernel.

Hot plug & Cold plug Devices

Hot plug and cold plug devices refer to the ability of a device to be inserted or removed while the computer is running (hot plug) or whether the machine has to be powered down before the device can be inserted or removed (cold plug).

Typical hot plug devices are USB data sticks, mice or keyboards and some SATA drives, while cold plug devices include video cards, network cards and CPUs, although in high end server machines even these components can be hot pluggable. The benefit of hot plug devices is the ability to swap out faulty, or add additional, components without having to take the machine offline, a requirement for many production servers that need to maintain maximum uptime.

A lot of what you need to know has been explained under "Linux Device Management Overview". When a hotplug device is inserted or removed the kernel updates the sys virtual filesystem and udev receives an event notification. Udev will create the device node under `/dev` and fire notifications to HAL which in turn will notify D-Bus of the changes.

USB Support

One of the most popular types of hotplug interfaces today is the Universal Serial Bus (USB) which is a communication architecture designed to connect devices to a PC. These devices are

divided into five classes:

- *Display Devices*
- *Communication Devices*
- *Audio Devices*
- *Mass Storage Devices*
- *Human Interface Devices (HID)*

The devices are plugged into a USB port which is driven by a USB controller. Support for USB controllers is present in the Linux kernel since version 2.2.7 (The Linux USB sub-system HOWTO)

Host Controllers

There are 3 types of USB host controllers:

<i>Host Controller</i>	<i>Kernel Module</i>
OHCI (Compaq)	usb-ohci.o
UHCI (Intel)	usb-uhci.o
EHCI (USB v2.0)	ehci-hdc.o

Once a USB device is plugged into a PC we can list the devices with `lsusb`:

```


# lsusb

Bus 001 Device 001: ID 0000:0000
Bus 001 Device 002: ID 04a9:1055 Canon, Inc.
    
```

```

File Edit View Terminal Help
:~$ lsusb
Bus 008 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 007 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 006 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 005 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 004 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 003 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 002 Device 005: ID 0d8c:0008 C-Media Electronics, Inc.
Bus 002 Device 004: ID 093a:2510 Pixart Imaging, Inc. Hama Optical Mouse
Bus 002 Device 003: ID 05e3:0608 Genesys Logic, Inc. USB-2.0 4-Port HUB
Bus 002 Device 002: ID 13fd:1e40 Initio Corporation
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub

```

Figure 101. 1.6: Sample `lsusb` output

Device Drivers

In our discussion of how Linux manages devices we mentioned that the kernel makes use of device drivers to export information to the `/sys` filesystem. In Linux device drivers most drivers are provided as part of the kernel source code and can be compiled as modules which can be loaded and unloaded dynamically into of the kernel or are compiled directly into the kernel. The modular system makes it possible for 3rd parties to write device drivers and provide them as modules that can be loaded and unloaded without having to recompile the kernel.

Most drivers are compiled as modules as this keep the kernel small and allows for far more drivers to be available that what would be practical if they were all compiled as part of the kernel. The commands you need to know to manage device drivers under Linux are:

- `lsmod` – lists currently install modules,
- `modinfo` – query a module for dependency, author and parameter information
- `insmod` – installs a module by providing a path to the driver,
- `modprobe` – installs a module via module name and handles dependency resolution,
- `rmmmod` – remove

In order to see what kernel modules are loaded by your system you can run the command **`lsmod`**. The first column of the output is the module name, the second is the size of the module and the third shows which other modules depend on this module.

Module	Size	Used by
binfmt_misc	6587	1
ppdev	5259	0
vboxnetadp	6326	0
vboxnetflt	15280	0
vboxdrv	190562	2 vboxnetadp,vboxnetflt
joydev	8708	0
snd_hda_codec_realtek	203310	1
fbcon	35102	71
tileblit	2031	1 fbcon
font	7557	1 fbcon
bitblit	4707	1 fbcon
softcursor	1189	1 bitblit
vga16fb	11385	0
vgastate	8961	1 vga16fb
snd_usb_audio	75765	2
snd_usb_lib	15801	1 snd_usb_audio
arc4	1153	2
snd_hda_intel	21941	2
snd_hda_codec	74201	2 snd_hda_codec_realtek,snd_hda_intel
snd_hwdep	5412	2 snd_usb_audio,snd_hda_codec

Figure 101.1.7: Sample `lsmod` output

If you require additional information about a module, such as its dependencies and configuration parameters you can use the `modinfo` command. Below is an example of output from the command “`modinfo psmouse`” – the mouse driver.

```

~-$ modinfo psmouse
filename:      /lib/modules/2.6.32-24-generic/kernel/drivers/input/mouse/psmouse.ko
license:      GPL
description:   PS/2 mouse driver
author:       Vojtech Pavlik <vojtech@suse.cz>
srcversion:   75F003305E7B43F23CFB5F2
alias:        serio:ty05pr*id*ex*
alias:        serio:ty01pr*id*ex*
depends:
vermagic:     2.6.32-24-generic SMP mod_unload modversions 586
parm:         tpdebug:enable debugging, dumping packets to KERN_DEBUG. (int)
parm:         recal_delta:packets containing a delta this large will cause a recalibration. (
int)
parm:         jumpy_delay:delay (ms) before recal after jumpiness detected (int)
parm:         spew_delay:delay (ms) before recal after packet spew detected (int)
parm:         recal_guard_time:interval (ms) during which recal will be restarted if packet rec
eived (int)
parm:         post_interrupt_delay:delay (ms) before recal after recal interrupt detected (int)
parm:         autorecal:enable recalibration in the driver (int)
parm:         proto:Highest protocol extension to probe (bare,imps,exps,any). Useful for KVM
switches. (proto_abbrev)
parm:         resolution:Resolution, in dpi. (uint)
parm:         rate:Report rate, in reports per second. (uint)
parm:         smartscroll:Logitech Smartscroll autorepeat, 1 = enabled (default), 0 = disabled.
(bool)
parm:         resetafter:Reset device after so many bad packets (0 = never). (uint)
parm:         resync_time:How long can mouse stay idle before forcing resync (in seconds, 0 = n
ever). (uint)

```

Figure 101.1.8: Sample `modinfo` output

To load a driver for a device you make use of the **insmod** or **modprobe** command. The **insmod** command takes the path to a kernel module as its parameter and will attempt to load the module into memory. The command below will attempt to load the xpad driver into memory.

```
insmod /lib/modules/2.6.32-22-  
generic/kernel/drivers/input/joystick/xpad.ko
```

The **insmod** command is inconvenient because it does not load any driver dependencies and you need to specify the full path to a driver module. The **modprobe** command will automatically load any dependencies which are not resident and allows one to use the module name rather than the module's file name.

To remove a module one can use the **rmmmod** command or **modprobe -r**. This will remove a module as long as there are no modules which depend on this module.

Used terms, files and utilities:

•/sys	•lsmod
•/proc	•lspci
•/dev	•lsusb
•modprobe	

101.2 Boot the System

Candidates should be able to design a disk partitioning scheme for a Linux system. Candidates should be able to select, install and configure a boot manager.

Key Knowledge Areas

- Provide common commands to the boot loader and options to the kernel at boot time.
- Demonstrate knowledge of the boot sequence from BIOS to boot completion.
- Check boot events in the log files.

Introduction

When an x86 computer starts up it follows a predefined set of steps to boot the operating system. On start up the CPU jumps to the address of the BIOS and proceeds to load it. The BIOS performs some checks and initializes hardware before locating the configured boot device. The boot device is configured in the BIOS user interface by setting the boot order of attached mass storage devices.

Once the boot device is located the BIOS proceeds to load the the Master Boot Record (MBR) of the boot device. The MBR is the first sector (512 bytes) of the boot device and contains the 1st stage boot loader with a partition table. The first stage boot loader can either directly boot the operating system in the case of a Linux 1st stage boot loader but usually the the 1st stage boot loader is responsible for locating and loading the 2nd stage boot loader which, in the case of Linux boot loaders, allows for more flexibility in selecting the kernel and operating system to boot.

The partition table is required for the 1st stage boot loader to be able to locate the offset to the 2nd stage boot loader. Due to the limited size of the MBR the partition table only contains the location of the primary partitions which therefore requires that the 2nd stage boot loader must be located on a primary partition.

The 1st stage boot loader locates the partition of the 2nd stage boot loader by looking at the boot sectors of the partition marked as active/bootable. (We will cover marking partitions (as opposed to entire devices) as bootable later when looking at disk partitioning.)

The second stage boot loader has the task of loading the operating system for Linux; this means the Linux kernel and initial ram disk. The second-boot loader may present the user with a menu to select which kernel to boot and may even allow the user to boot heterogeneous operating systems (aka dual boot).

Once the 2nd stage boot loader has loaded the kernel it passes control over to it, The kernel starts and configures the CPU type, interrupt handling, the rest of memory management such as paging tables and memory paging, device initialization, drivers, etc.

The kernel also loads any initial RAM disk image that may be present into memory and mounts it as a temporary root filesystem in ram. The initial ram disk (initrd) contains an image of system configuration files and modules that the kernel will need to be able to access system hardware. It is here that various file system and disk drivers are loaded that enable the kernel to find and mount the real root partition. For example if the real root partition is on a raid 1 devices, the module for raid1 will need to be loaded to enable the kernel to mount and read the root filesystem. The temporary root files system is later swapped out for the real root filesystem once the kernel has access to it.

Once the kernel is fully operational it starts an initial programme which by default is `/sbin/init`. The init programme sets up user space and starts the login shells and/or the graphical login. What services are started and the state of the machine after the init process has completed its initialization depends on the default runlevel and its configuration.

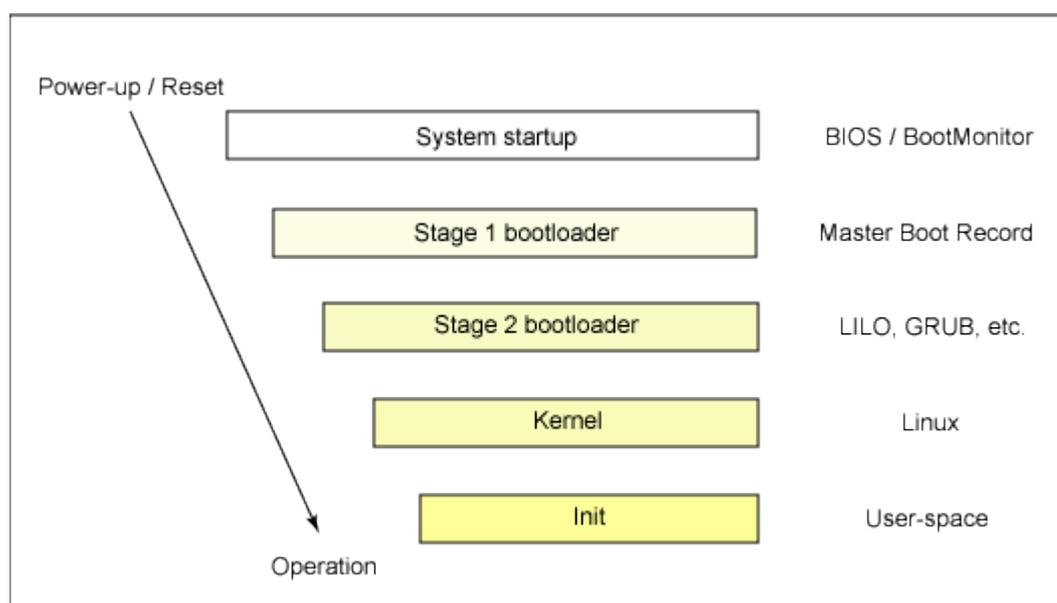


Figure 101.2.1: The Bootup Process

Boot Parameters

There are two widely used boot loaders for Linux, namely *GRUB* and *LILO*. Both boot loaders are broken into at least two stages. The first stage is a small machine code binary on the MBR. Its sole job is to locate and load the second stage boot loader. The 2nd stage boot loader then locates and loads the Linux kernel passing in any parameters with which it has been provided.

When the 2nd stage boot loader is running, you are presented with an opportunity to pass additional parameters to the Linux kernel via a text console that is part of the boot loader. Typical parameters that you may pass to the kernel include:

init – overrides the process that is run by the kernel after it has finished loading. “`init=/bin/bash`” is used to bypass the login prompt in cases where the root password had been forgotten. Because this gives access to a root shell, this also highlights why it is important to have good physical access controls to the computer console and why it is a good idea to secure your boot loader with a password. This will prevent users from modifying the boot-time kernel parameters.

root – informs the kernel which device to use as the root filesystem. Often used when troubleshooting an incorrectly configured boot loader. E.g `root = /dev/hda1` tells the kernel to use `/dev/hda1` as the root device filesystem rather than what it has been configured to use

noapic/nolapic – tells the kernel not to use the advanced programmable interrupt controller or local advanced programmable interrupt controller for assigning IRQ and resources. This effectively turns off pnp in the Linux kernel.

noacpi – turns off the advance configuration and power interface capabilities of the Linux kernel. Often needed in the case of a buggy BIOS.

There are many more parameters that can be passed to the kernel at boot time. Please consult the kernel documentation for more options.

Init Process Overview (SysV init style)

Once the kernel has finished loading it starts the init process. Init is responsible for checking and mounting file systems, and starting up configured services, such as the network, mail and web servers for example, it does this by entering its default runlevel. This is configured in the `/sbin/init` application configuration file `/etc/inittab`. An example of the `inittab` file is given below:

```

1.#
2.# inittab          This file describes how the INIT process should set up
3.#                  the system in a certain run-level.
4.
5.id:3:initdefault:
6.
7.# System initialization.
8.si::sysinit:/etc/rc.d/rc.sysinit
9.
10.l0:0:wait:/etc/rc.d/rc 0
11.l1:1:wait:/etc/rc.d/rc 1
12.l2:2:wait:/etc/rc.d/rc 2
13.l3:3:wait:/etc/rc.d/rc 3
14.l4:4:wait:/etc/rc.d/rc 4
15.l5:5:wait:/etc/rc.d/rc 5
16.l6:6:wait:/etc/rc.d/rc 6
17.
18.# Trap CTRL-ALT-DELETE
19.ca::ctrlaltdel:/sbin/shutdown -t3 -r now
20.
21.# When our UPS tells us power has failed, assume we have a few minutes
22.# of power left.  Schedule a shutdown for 2 minutes from now.
23.# This does, of course, assume you have powerd installed and your
24.# UPS connected and working correctly.
25.pf::powerfail:/sbin/shutdown -f -h +2 "Power Failure; System Shutting Down"
26.
27.# If power was restored before the shutdown kicked in, cancel it.
28.pr:12345:powerokwait:/sbin/shutdown -c "Power Restored; Shutdown Cancelled"
29.
30.# Run gettys in standard runlevels
31.1:2345:respawn:/sbin/mingetty tty1
32.2:2345:respawn:/sbin/mingetty tty2
33.3:2345:respawn:/sbin/mingetty tty3
34.4:2345:respawn:/sbin/mingetty tty4
35.5:2345:respawn:/sbin/mingetty tty5
36.6:2345:respawn:/sbin/mingetty tty6
37.# Run xdm in runlevel 5
38.x:5:respawn:/etc/X11/prefdm -nodaemon

```

A line in the `inittab` file has the following format:

id:runlevels:actions:process

- **id** – 1-4 characters that identify the function,
- **runlevels** – the runlevels for which the **process** will be executed,
- **action** – One of a defined list of events for which the process should be executed or an instruction to init on what to do when the process is executed. The most commonly used actions are:
 - **wait** – init will wait till the process it is starting has completed before continuing,
 - **respawn** – tells init to restart the process whenever it terminates, this is useful for login processes.
 - **Ctrl-Alt-Del** – this traps the ctrl-alt-delete key combination,
 - **initdefault** – set the default runlevel
 - **powerfail** and **powerokwait** – are used to respond to notifications from attached UPS devices in the event of power failure and restoration
- **process** - the command to execute

Line 5 sets the default run level for init. In order to determine the default runlevel the init process searches for the `initdefault` entry, if there is no such entry (or no `/etc/inittab` at all), a runlevel must be entered at the system console. There are 6 run levels. On a RedHat-style system, the default run level is 3 for a server (no GUI) or 5 for a desktop machine.

Line 8 is for the `sysinit` action, this process is run for every runlevel.

Line 10 – 16 define the scripts that should be run for each runlevel.

Line 19, 25, 28 define the processes to be run when the specified actions occur. These actions are:

- **Line 19** when ctrl-alt-delete combination is hit, the computer shuts down, ,
- **Line 25** when the a signal from the ups on a power outage is received, the computer is told to shutdown after 2 minutes,
- **Line 28** when power is restored and the machine has not shutdown the scheduled shutdown is canceled.

Line 32-37 spawn the console logins. They are set to automatically restart when the process dies.

Line 40 – starts the graphical login console for runlevel 5

After it has spawned all of the processes specified, init goes dormant, and waits for one of three events to happen:- processes it started to end or die, a power failure signal or a request via `/sbin/telinit` to further change the runlevel.

Init Process Overview (systemd init style)

Systemd is a system and service manager for Linux, backwards-compatible with SysV and LSB init scripts. So it can be installed on any Linux that uses `sysvinit` or as a replacement.

systemd brings efficiency by providing faster startups by parallelizing processes startup, this is achieved by activating D-Bus(Desktop Bus) which uses Unix domain sockets for inter-process communication, processes are able to run parallel to each other without acknowledging if the daemons they depend on are actually running.

It also offers on-demand starting of daemons and serves as the full-time dynamic resource manager to a running system rather than simply starting the system and then going to sleep until the next reboot as it is with sysvinit.

You will need to tell your kernel to run the init provided by systemd, this is achieved by changing the init kernel parameter to pint to /usr/bin/systemd:

```
FILE: /boot/grub/grub.cfg
. . .
title Desktop -- openSUSE 12.1 - 3.1.0-1.2
    root (hd0,0)
    kernel /vmlinuz-3.1.0-1.2-desktop root=/dev/sda1 init=/bin/systemd
splash=silent quiet showopts vga=0x317
    initrd /initrd-3.1.0-1.2-desktop
```

Init Process Overview (upstart init style)

Upstart is an **event-based** replacement for the/sbin/init daemon which handles starting of tasks and services during boot, stopping them during shutdown and managing them while the system is running. Upstart comes with a set of default jobs which it installs into /etc/init. These are based on the sysvinit configuration of Debian based systems, including running the /etc/init.d/rcscript.

Troubleshooting The Boot Process

During boot process the various modules that the kernel loads/probes for their supported hardware , producing copious amount of log output in the processes. These are the console messages that fly-by during system boot.

Since the logging service has not yet been started the kernel logs its messages to an in memory ring buffer. It is called a ring buffer as, once the log has reached a set memory size, earlier messages are overwritten by newer messages.

As this information in the ring buffer could be lost on reboot or by being overwritten most distributions write the ring buffer entries to disk either under /var/log/dmesg, /var/log/messages or /var/log/syslog depending on their logging service configuration.

The contents of the ring buffer can be read with the **dmesg** command. This is usually combined with a pagination utility, such as less, as the amount of data contain in the ring buffer is usually too large to fit on one screen.

dmesg | less

If you have hardware that is not being configured properly during boot the ring buffer is a good place to look for clues as to what could be wrong as usually the driver will write out some error message that can help in resolving the issue.

Beside the ring buffer you can also by examining the system log files configured by the logging service. These files are located under /var/log and may be /var/log/syslog or /var/log/messages.

Failure to boot to command prompt

In some cases you may be dumped to the command line during the boot process with a message that the root device could not be found or that you can type Ctrl-D plus the root password to perform maintenance. Usually this means that the root device has not been configured properly and you may be able to fix this from the command line.

In other cases you may get a message which says “kernel panic” and the machine automatically reboots, or waits in an unusable state until rebooted. The causes of this kind of problems can be many, from an incorrectly configured `initrd` image, to a missing root device. The output on the screen, before system failure can provide you with a clue as to the configuration error. This may be due to:

- configuration problems with boot loader, i.e. it is configured for the incorrect root device,
- missing modules in the `initrd` image,
- configuration error on the root file system.

In the cases of incorrectly configured boot loaders you can either enter the boot loader editor during the 2nd stage, by pressing 'e' for the grub loader, or at the `lilo` prompt, and passing in the correct parameters to get the computer to boot, editing the configuration file once the system has successfully started. In other cases it may be necessary to boot a live CD and then trouble shoot the root file system without mounting it.

Used files, terms and utilities:

- `/var/log/messages`
- `dmesg`
- BIOS
- `bootloader`
- `kernel`
- `init`

101.3 Runlevels, Rebooting & Shutting down the system

Candidates should be able to manage the runlevel of the system. This objective includes changing to single user mode, shutdown or rebooting the system. Candidates should be able to alert users before switching runlevel and properly terminate processes. This objective also includes setting the default runlevel.

Key Knowledge Areas

- Set the default runlevel.
- Change between run levels including single user mode.
- Shutdown and reboot from the command line.
- Alert users before switching runlevels or other major system event.
- Properly terminate processes.
- Knowledge of basic features of systemd and Upstart.

The following is a partial list of the used files, terms and utilities:

- /etc/inittab
- shutdown
- init
- /etc/init.d
- telinit

Introduction

Traditionally Linux has 7 runlevels numbered from 0- 6. A runlevel defines the state of the computer after boot. Except for runlevels 0 and 6, the the allocation of runlevel numbers to particular configurations differ from distribution to distribution. Run level 0 is the runlevel used to halt the system and run level 6 is used for rebooting the system. The remaining runlevels are typically assigned to the following configurations

- 1) single user mode
- 2) multi-user, text mode, no servers
- 3) multi-user, text mode, servers
- 4) multi-user, GUI mode, no servers
- 5) multi-user, GUI mode, servers.

Below are the runlevel for a Redhat server.

```
# Default runlevel. The runlevels used by RHS are:
# 0 - halt (Do NOT set initdefault to this)
```

```
# 1 - Single user mode
# 2 - Multiuser, without NFS (The same as 3, if you do not have networking)
# 3 - Full multiuser mode
# 4 - unused
# 5 - X11
```

System V init

What services and hardware is configured for the different runlevels is determined by a set of scripts configured for each level. Runlevel specific scripts are stored under `/etc/rc.d/rc?.d`, `/etc/init.d/rc?.d` or `/etc/rc?.d` or similar location where the ? is the number of the runlevel.

Under each of these directories there are symbolic links back to a common set of scripts, under the `/etc/rc.d`, `/etc/init.d`, or `/etc/rc.d/init.d` (note the missing number from the paths) , that all runlevels use.

The symbolic links have a name that starts with a *S* or *K*, a number and then the name of the service the script controls. The *S* means that this is a script run with 'start' when the runlevel is entered and *K* signifies a script which is run when the run level is exited. The number indicates the order in which the scripts are run while the name indicates the services. The order of the scripts is determined by system requirements, for example the network needs to be configured before the mail or web server can be started. If one were to take a long listing of the symbolic links you would see that they point to the same script, they only differ in the parameters based to the scripts by init. 'start' to start it, and 'stop' to stop it.

Startup scripts are simply bash scripts that call the appropriate application when run. These scripts take a standard set of parameters namely; start,stop,restart and reload. Some poorly written scripts may not implement all of the parameters.

Systemd init

systemd uses service files (`/lib/systemd/system/`) as its native form of configuration, and is compatible with traditional initscripts.

Example:

```
# ls /lib/systemd/system/
acpid.service
alsa-restore.service
cgroup.service
cleanup.service
clock.service
console-kit-daemon.service
cron.service
crypto-early.service
crypto.service
fsck-root.service
fsck@.service
getty@.service
```

```
network.target
nss-lookup.target
powerof
systemd-timedated.service
systemd-tmpfiles-clean.service
systemd-tmpfiles-clean.timer
systemd-tmpfiles-setup.service
systemd-update-utmp-runlevel.service f.service
udev.service
udev-settle.service
udev-trigger.service
```

On the system configured to use systemd init, where sys V initscripts are also available, service manipulation can be achieved by executing the relevant sys V initscripts which call systemd to handle the execution.

Example:

```
# /etc/init.d/network status
redirecting to systemctl
network.service - LSB: Configure the localfs depending network interfaces
   Loaded: loaded (/etc/init.d/network)
   Active: active (exited) since Tue, 31 Jul 2012 09:44:58 +0200;
         4min 52s ago
   Process: 964 ExecStart=/etc/init.d/network start (code=exited,
status=0/SUCCESS)
   CGroup: name=systemd:/system/network.service
```

Upstart init

Processes managed by upstart are known as jobs and are defined by the files in the /etc/init directory.

Example:

```
# ls /etc/init/
acpid.conf
alsa-restore.conf
anacron.conf
atd.conf
console-setup.conf
cron.conf
cups.conf
dbus.conf
dmesg.conf
mountall.conf
mounted-dev.conf
networking.conf
network-interface.conf
network-interface-security.conf
rc.conf
rsyslog.conf
ssh.conf
tty1.conf
udevmonitor.conf
udevtrigger.conf
upstart-socket-bridge.conf
wait-for-state.conf
```

The primary event is the startup event, generated when the init daemon has completed loading its configuration, and is the signal that the rest of the system may be started. The primary event is the startup event, emitted when the daemon has finished loading its configuration. In the default Upstart configuration, the primary task run on the startup event is the `/etc/init/rc-sysinit.conf` job responsible for generating the System V compatible runlevel event.

Configuring Runlevels

You can add or remove services from run levels by changing the symbolic links in the respective `rc` directory but various tools exists to enable easy management of these scripts.

Under Debian systems and their derivatives you can use the `update-rc.d` or `rc-update` scripts, passing in the name of the service you wish to add or remove with the appropriate parameter.

Example:



```
# update-rc.d disable 3 apache2
```

will remove the apache2 service from runlevel 3



```
# update-rc.d enable 3 apache2
```

will add apache to runlevel 3.

For rpm based distributions the command `chkconfig` does the same thing.

Example:



```
# chkconfig -del httpd
```

will remove the apache web server from all configured run levels



```
# chkconfig -add httpd
```

will add the apache web server to all default runlevel,



```
# chkconfig -level 3 httpd on
```

will add apache to runlevel 3

Besides adding and removing these services `chkconfig` can also list the runlevels for which a service has been configured with `chkconfig --list apache2`

For system with `systemd` installed services can be configured to start or stop on bootup for each runlevel.



```
# systemctl disable smartd.service
```

Disables `smartd` to not start on bootup



```
# systemctl enable smartd.service
```

Enables `smartd` to be started on bootup

Changing Runlevels

Runlevels can be changed by invoking the `init` binary with the run level number; for example `init 6` will reboot the machine, while `init 0` will halt or shutdown the machine. An alternative to `init` is `telinit`. `init` and `telinit` perform the same function and differ only in the parameters they accept.

`Teleinit/init` is often used to change the state of a computer without rebooting it. Sometimes it is necessary to change the state of a computer to single user mode for system maintenance. This can be done with `init 1`. Before changing run levels it is good practise to let currently logged in users know about it. This can be done with the `wall` command. E.G. `wall "users will be logged out in 5 minutes..."` will send the text message to all logged in users.

If you need to determine your current runlevel you can run the command `"runlevel"`. To change runlevels you use the `init` command as explained above. Runlevels can also be changed by the following commands:

- **halt** – stops the system,

- **reboot** – reboots the system,
- **shutdown** – will halt the system after a specified time or at a specified time interval. The command can also be used to schedule a reboot. The **shutdown** command can also take a text message that is displayed to any users logged in on a console on the system. For example:

- **shutdown -h now** - will halt the system immediately,
- **shutdown -h +10** - System shutdown in 10 minutes – will halt the system in 10 minutes time,
- **shutdown -r 14:30** - System will be rebooted at 14:30
- **shutdown -c** - will cancel any scheduled shutdown or reboot

Starting & Stopping Services

You may also stop and start services at runtime. On Debian-based systems, by either invoking the start-up script directly on debian based systems with the appropriate parameter i.e. start for starting the service and stop for halting the service.

On Ubuntu one can run:



```
# /etc/init.d/apache2 stop
# /etc/init.d/apache2 start
```

To start or stop apache server

On Redhat based systems you can use the service command to stop and start services.



```
# service httpd start
# service httpd stop
```

To start or stop apache server

These commands can also be used to reload service configuration files to get the service to implement any configuration changes that may have been made. e.g.



```
# /etc/init.d/apache2 reload
# service http reload
```

systemd init system

On OpenSUSE one can run:



```
# systemctl stop sshd.service
# systemctl start sshd.service
```

To start or stop ssh server immediately



```
# systemctl status sshd.service
sshd.service - LSB: Start the sshd daemon
   Loaded: loaded (/etc/init.d/sshd)
   Active: active (running) since Mon, 30 Jul 2012 13:28:11 +0200;
   8min ago
     Process: 1509 ExecStart=/etc/init.d/sshd start (code=exited,
status=0/SUCCESS)
    CGroup: name=systemd:/system/sshd.service
           └─ 1619 /usr/sbin/sshd -o PidFile=/var/run/sshd.init.pid
```

To check status of ssh server.

Upstart init system

On Ubuntu one can run:



```
# initctl stop cups
# initctl start cups
```

To start or stop cups server.

It should be noted that not all services support the ability to reload their configuration files.

Used files, terms and utilities:

- /etc/inittab
- shutdown
- init
- /etc/init.d
- telinit

Topic 102: Linux Installation and Package Management

102.1 Design hard disk layout

Candidates should be able to design a disk partitioning scheme for a Linux system

Key Knowledge Areas

- Allocate filesystems and swap space to separate partitions or disks.
- Tailor the design to the intended use of the system.
- Ensure the /boot partition conforms to the hardware architecture requirements for booting.
- Knowledge of basic features of LVM.

Introduction

When installing an operating system you will need to design the layout of your hard-disk partitions and choose a filesystem you will use to format them. The choices you make will depend on the number and size of mass storage devices you have and the purpose for which the computer will be used.

Linux creates filesystems on block devices. A device whose data can only be accessed sequentially, such as keyboards and mice are character devices but devices such as hard-disks, that can have their data accessed randomly are call block devices. Each type of device has different functionality to cater for the differences in data access methods. The Linux kernel's device-mapper framework allows for the creation of virtual block devices on which filesystems can be created.

The advantage of virtual block devices is that Linux can provide functionality such as encrypted drives and raid devices without the need for specialist hardware. Virtual block devices allow for the creation of LVM (Logical Volume Manager) drives and software raid with `mdadm` for example.

In this manual we will only cover block device drivers that map directly to the underlying hard-disk. i.e. we exclude virtual block device drivers and look at creating filesystems directly on physical disk partitions.

The word partition is used in several different contexts when discussing Linux systems. Sometimes it is used to refer to the partitions on a hard disk and in other context it is used to refer to the location of directories. For example the root partition, which contains the core Linux configuration files, may actually be spread across many disk partitions if it is created on a LVM block device for example.

Hard Drive Partitions

Originally X86 machines' hard disks only supported a maximum of 4 partitions as, at the time, this was considered more than enough for anyone's needs. Now-a-days it is common to have more than 4 partitions on a single disk. To overcome the 4 partition limitation, in a manner that was backwardly compatible, the standard was extended to enable one of the primary partitions to be created as an "extended partition" which allowed for the creation of many "logical partitions" on the extended partition.

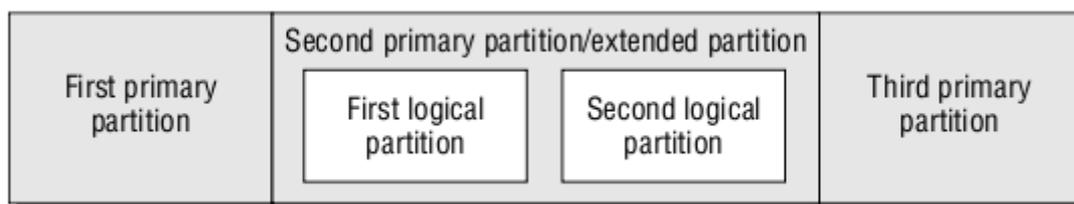


Figure 102.1-1: Hard Disk Partitions

The 4 partitions originally supported by the X86 architecture are called primary partitions. If you need more than 4 partitions, then at least one of the primary partitions needs to be created as an extended partition. The extended partition can then have multiple logical partitions created on it.

Under Linux primary partitions are assigned the number 1-4 depending on the primary partition being created. The extended partition, of which there can be only one, is assigned the number 4 with logical partitions being assigned numbers greater than 4.

So for a master PATA drive on the primary connector the 1st primary partition would be `/dev/hda1` and the 3 primary partition would be `/dev/hda3`, while the extended partition would be `/dev/hda4` and the first logical partitions numbered `/dev/hda5`. Primary partition numbers can be skipped so it is possible to have a disk layout such as `/dev/hda1`, `/dev/hda3` and `/dev/hda5` `/dev/hda6` `/dev/hda7` for the 1st primary partition, extended partition, and then two logical drives on the extended partition (hda3).

If you intend to have more than 4 partitions, it is important to ensure that you create an extended partition for one of the primary partitions, as it is not possible to convert a primary partition to an extended partition. If you need an extended partition, and all primary partitions are used you would have to delete one of the primary partitions to create the extended partition.

Common Partition Schemes

You may ask why would you need to partition a hard drive and in some cases, especially desktop computers it may not be necessary to provide for more than two partitions. You will always need at least two partitions to cater for how Linux handles virtual memory. Virtual memory is the amount of memory the kernel tells its applications is available for use and this can often exceed the amount of physical memory available. In order to cater for the extra memory which does not really exist, the kernel uses hard disk space as virtual memory. The algorithms used to manage virtual memory and swapping of memory to disk and back are beyond the scope of this manual. What is important to know is that the kernel needs a special disk partition, known as swap space, to use for virtual memory management.

Besides swap space there are other reasons for using extra partitions, especially in machines with more than one disk. These include: preventing log files and temporary files from consuming too much disk space, improving disk performance by splitting read and writes across partitions on different disks, planning for disk expansions and creating a flexible layout for the future. If you have multiple disk, splitting the filesystem across disks will help isolate parts of the system from disk failures in different drives. Even in the case of laptops, which can only take a single drive, there are good reasons for creating a flexible disk layout, for example you may need to potentially accommodate a multi-boot environment later or to make it easier to run virtual machines with virtual hard-disks.

One should be careful though not to create too many partitions, especially when you are not taking advantage of one of the Linux kernels virtual block devices, such as LVM, which enable

dynamic allocation of disk partitions to filesystems, as it is a common problem to be running out of disk space on one partition while there is still space available on another, so plan your disk partitioning strategy carefully.

Some common disk partitions for smaller systems

/boot partition	50-100MB usually on the 1 st partition to ensure all kernels are below the 1024 cylinder limit for older kernel BIOS.
Swap	Swap space allocation depends on the amount of physical ram available. There is much debate about the optimal size of the partition for swap space, especially as the amount of physical memory available in most systems is increasing, but a general rule of thumb is to use twice the amount of ram available but a maximum of around 1 G is recommended.
/	The root filesystem partition contains the rest of the Linux operating system and should usually be allocated the remaining disk space. Most distributions today will install the root partition on an logical partition to allow for the possibility of creating more than 4 partitions in future should this be necessary. If all the available space has been allocated to the root partition, it may be necessary to resize the root partition to free up space before this can be done.

Larger Systems Partition Strategy

/boot partition	100MB-250MB In more modern computers, that do not have the 1024 cylinder limit, you may want to allocate more space to the boot to accommodate additional kernels. Keeping the /boot separate may be necessary especially if the remaining filesystems are loaded on virtual block devices which the boot loader cannot access natively, for example they root filesystem is on a software raid5 block device.
swap	As above
/home	The /home directory is where all user data and documents are kept. A separate partition means that it will be easier to backup data and protects the kernel and other core system files from running out of disk space due to users storing too much data in their home directories.
/usr	The user directory contains is user application. Usually the /usr directory does not grow that much in size and is fairly static.
/var	The /var directory contains system log files. Logs files, especially if not maintained properly, can rapidly fill up disk space. By placing the /var directory on a different drive the rest of the system is protected from running out of disk space.
/tmp	The /tmp directory is used to house all temporary files. The directory should be cleaned up on reboot or when users log out but often files are left behind. Over time this can cause the system to run out of disk space. Placing /tmp in its own directory protects the rest of the system form this.
/	The remainder of the filesystem.

How to Partition Using fdisk

During installation you may be presented with a graphical interface to partition your disks but usually

when creating or editing your drives partitions you will use a utility such as **fdisk**.

The fdisk command takes the block device you wish to partition as a parameter. For example to partition the master drive on the primary connector for a PATA drive the command would be



```
# fdisk /dev/hda
```

The fdisk utility provides you with a command prompt to create and delete partitions on block devices. Pressing “m” at the prompt will provide you with a list of possible actions to perform on the block device.

```
Command action options
a  toggle a bootable flag
b  edit bsd disklabel
c  toggle the dos compatibility flag
d  delete a partition
l  list known partition types
m  print this menu
n  add a new partition
o  create a new empty DOS partition table
p  print the partition table
q  quit without saving changes
s  create a new empty Sun disklabel
t  change a partition's system id
u  change display/entry units
v  verify the partition table
w  write table to disk and exit
x  extra functionality (experts only)
```

To Create A Partition

Typing “n” at the command prompt will result in a prompt asking if you want to create an extended or primary partition.

```
Command action
e  extended
p  primary partition (1-4)
```

It is usually good practice to create at least one of your partitions as an extended partition. Once an extended partition has been created subsequent requests to create a new partition will result in a prompt asking if you want to create a logical or primary partition, assuming you have not created the maximum number of primary partitions.

```
Command action
l  logical (5 or over)
p  primary partition (1-4)
```

Once you have your drive partitioned you need to decide what type your partition will be. To get a list of partition types type “l” at the fdisk prompt. This will list a large number of potential partition types but the ones you will be interested in are:

```
82 Linux Swap
83 Linux
```

```
8e Linux LVM
85 Linux Extended
```

For the purposes of this manual we will consider only partition ids 82 and 83. Types 8e and 85 are examples of partition types that are used to create virtual block devices.

Formatting Partitions with Filesystems

A filesystem is a way of storing data in a computer-accessible form on the hard disk. Different filesystems have different algorithms and structures to determine where the data and indexing information, needed by the computer to find data, will be stored. Meta-data relating to the organisation of the filesystem is stored in areas of the disk called superblocks. Superblocks contain critical information for the computer to find files and data on disk. To protect the filesystem from becoming inaccessible when superblocks become corrupted Linux filesystems store multiple copies of the superblock at defined offsets. Some popular Linux filesystems include:

ext2 – the oldest and most well supported filesystem on Linux machines.

ext3 - an extension of the ext2 filesystem which adds journaling support

reiserfs - an enhanced journaling filesystem written by Hans Reiser

For more information on filesystems and how to create them see section 104.1

Managing Swap Space

We have covered swap space earlier in this chapter. Swap space does not contain a file system but is accessed in raw mode by the Linux kernel. By passing a filesystem has some speed advantages which is crucial for swap space. After creating the disk partition to be used as swap space, you will need to activate it with

```
swapon [device] e.g. swapon /dev/sda2
```

During system installation this will be done automatically for you. Information on the swap partition can be displayed with `swapon`.

```
swapon -s # Display the available swap partitions
```

Mounting filesystem Partitions

When you create a filesystem partition it needs to be mounted to be made available to the system. A mount point makes the filesystem partition available under a well defined location under root (/). Linux follows a well defined filesystem hierarchy and mounting the partition under the correct mount point is crucial if the system is to operate properly.

Mount points are defined under the `/etc/fstab` configuration file.

Knowledge of basic features of LVM.

What is LVM?

LVM is a Logical Volume Manager for the Linux operating system and it is a way or means of allocating hard drive space into logical volumes that can be easily resized instead of partitions..

There are two versions of LVM:

- LVM 2 – The latest version of LVM for Linux. LVM 2 is backward compatible with

volumes created with LVM 1. LVM 2 uses the device mapper kernel driver. Device mapper support is in the 2.6 kernel tree and there are patches available for current 2.4 kernels.

- LVM 1 – The version that is in the 2.4 series kernel.

One or more hard drives are allocated to one or more *physical volumes* but a physical volume can not span over more than one drive The physical volumes are combined into *logical volume groups* excluding the boot partition. The boot partition can not be include onto a logical volume group because it make it difficult to read by the the boot loader. Therefore If your root / partition is on a logical volume you should make sure that a separate /boot partition is created and not is not part of any volume group.

Since physical volumes can not span over more than one drive, to span over more than one drive, create one or more physical volumes per drive and then link them together through a logical Volume group. .

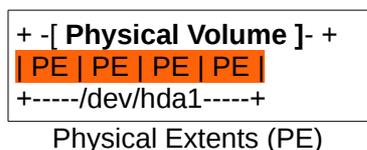
We start at the bottom, more or less.

The physical media

The word physical simply a hard disk, or a partition. Examples, /dev/hdc, /dev/hdc1, /dev/sda.

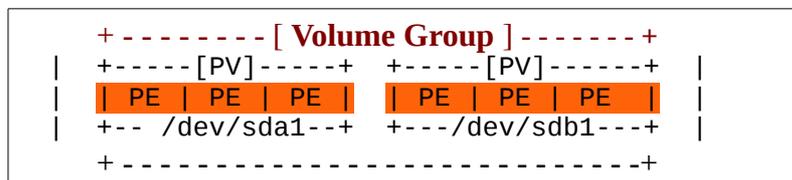
Physical Volume (PV)

A PV is nothing more than a physical medium or hard drive with some administrative data added that LVM will recognise.



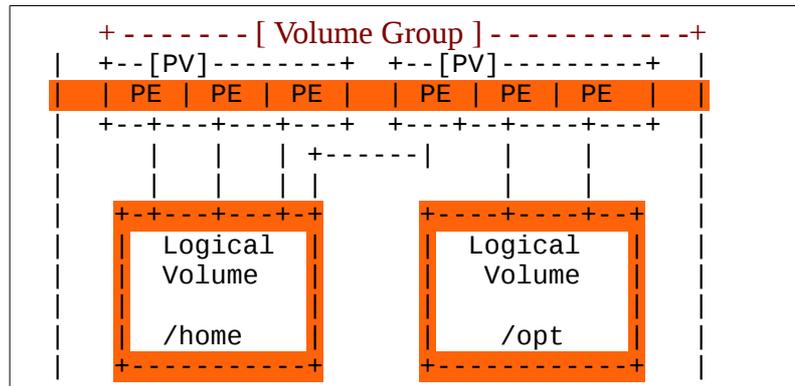
Physical Extents are just big blocks, often with a size of megabytes.

Volume Group (VG) :- A VG is made up of a number of Physical Extents (which may have come from multiple Physical Volumes or hard drives). One could generally think of a VG as being made up of several hard drives (/dev/hda and /dev/sda for example) but it's more accurate to say that it made up of PEs which may be provided by multiple hard drives



Logical Volume (LV)

A Logical Volume is the end result of our work, and it's there that we store our information. This is equivalent to the historic idea of partitions.



This shows us two filesystems, spanning two disks. The /home filesystem contains 4 Physical Extents, the /opt filesystem 2.

Basic LVM Commands:

The following commands can be used to commands to show the physical volume, volume group and logical volume status: **pvdisplay**, **vgdisplay** and **lvdisplay**.

A **Scenario** explaining how useful LVM could be: Say that you have a 20GB disc and you create the "/home" partition using 15GB and /var 1GB. Imagine that you run out of space and decide that you need 18GB in "/home". Using the old notion of partitions, you'd have to have another drive at least 20GB in size. You could then add the disc, create a new /home, and copy the existing data over.

However, with an LVM setup, you could simply extend with free space repartitioned or just add a 5GB (or larger) disc, and add it's storage units to the "/home" partition. Other tools allow you to resize an existing file-system, so you simply resize it to take advantage of the larger partition size and you're back in business. LVM can even make 'snapshots' of itself which enable you to make backups of a non-moving target.

What can it do for me?

- LVM == Logical Volume Manager
- Flexibility
- High availability
- Reliability

Key Benefits Of LVM:

- Arbitrary "partition" resizing (flexibility).
- Migrate data easily--swap drives on a running system (availability).

Filesystem

This filesystem is whatever you want it to be: the standard ext2, ReiserFS, NWFS, XFS, JFX, NTFS, etc... To the linux kernel, there is no difference between a regular partition and a Logical Volume.

Some LVM terminology and Commands:

To show LVM version:

lvm version

To create the physical disks:

```
pvcreate /dev/sda
pvcreate /dev/sdb
pvcreate /dev/sdc1
pvcreate /dev/sdc2
```

To show allocated Physical volumes:

```
pvscan
pvdisplay
lvm pvs
```

To setup a volume group:

```
vgcreate vg_name /dev/sda /dev/sdb /dev/sdc1 /dev/sdc2
```

To display a volume group:

```
vgdisplay
lvm vgs
```

To display a logical volume:

```
lvdisplay
lvm lvs
```

To create Logical Volume:

Sets the size to 1GB

```
lvcreate -L1G -n lv_name vg_name
```

Used files, terms and utilities:

- /(root) filesystem
- /var filesystem
- /home filesystem
- swap space
- mount points
- partitions

102.2 Install a boot manager

Candidates should be able to select, install and configure a boot manager.

Key Knowledge Areas

- Providing alternative boot locations and backup boot options.
- Install and configure a boot loader such as GRUB Legacy.
- Perform basic configuration changes for GRUB 2.
- Interact with the boot loader.
-

The following is a partial list of the used files, terms and utilities:

- /boot/grub/menu.lst, grub.cfg and other variations.
- grub-install.

- MBR.
- superblock.

Introduction

A boot loader is installed in the MBR. When a system starts, it loads what is in the MBR to RAM. Under Linux there are two main boot loaders:

- Lilo: LInux LOader(deprecated).
- Grub: GRand Unified Boot Loader.

A boot loader allows you to select the image that you would like to boot from. A system can contain multiple images (operating systems).

A boot loader allows you to interactively run commands and pass parameters to the image that you will boot. The initrd is the driver that will be used to build a filesystem on RAM to mount other filesystems and execute programs.

GRUB is today's default boot loader for many distributions. When installing Windows with Linux, install Windows first and Linux second, because Windows overwrites the MBR without asking.

As explained under the boot processes above, there are two stages to a boot loader. The first stage and second stage. The first stage is the boot sector application that is responsible for loading stage2.

Grub Boot Loader

Grub understands Linux filesystems and instead of using raw sectors to locate the Linux kernel GRUB can load from an ext2 or ext3 filesystems. GRUB is also capable of directly loading several non Linux operating systems. The GRUB boot loader has a three stages. Stage 1 has the usual job of a primary boot loader but it loads stage 1.5 which is a boot loader which understand a particular file system such as ext2/3, reiserfs etc and once loaded it loads the 2nd boot loader. The 1.5 stage boot loader is located on the first 30 kilobytes of hard disk immediately after the MBR.

Grub can be installed, either by editing the grub configuration file located at `/boot/grub/grub.conf` or `/boot/grub/menu.lst` and running `grub-install` or by invoking the grub shell. When referring to hard disks and partition grub uses the convention `(hdx)` for hard disks, irrespective of whether they are PATA or SATA disks, and `(fdx)` for floppy disks, where x refers to the disk number as seen by the BIOS. When a reference to a partition is need the conventions `(hdx,n)` or `(fdx,n)` is used, where n is the partition numbered from 0.

A sample of a grub configuration file is given below:

```
default=0
timeout=10
splashimage=(hd0,0)/grub/splash.xpm.gz
title    Linux (2.4.18-14)
        root (hd0,0)
        kernel /vmlinuz-2.4.18-14 ro root=/dev/hda5
```

```
initrd /initrd-2.4.18-14.img
```

The main sections of the grub.conf/menu.lst file are:

default	image that will boot by default (the first entry is 0)
timeout	prompt timeout in seconds
title	name of the image
root	where the 2 nd stage bootloader and kernel are e.g (hd0,0) is /dev/hda1
kernel	path for the kernel starting from the previous root e.g /vmlinuz
ro	read-only
root	the filesystem root
initrd	path to the initial root disk

To install the first stage MBR loader on /dev/hda with grub-install you would run the command **grub-install (hd0)**. Alternatively grub can be installed through the grub shell but still requires the grub.conf/menu.lst file. The grub shell can be entered by typing **grub**. To install the boot loader you need to run the following commands in the shell:

```
root (hd0,0)
setup (hd0)
```

The first line tells grub that the /boot partition, where the configuration file is located, is on the first hard disk on partition 1, and the 2nd command tells grub to install the 1st boot loader to the MBR of the first hard disk. An advantage of Grub is that each time the grub configuration file is updated with an new image for example, grub does not have to be reinstalled as with LILO.

GRUB 2

GRUB 2 is the next generation, a rewrite of what was formerly known as GRUB (i.e. Ver 0.9x), which has, in turn, become GRUB Legacy. GRUB Legacy is no longer being developed. GRUB 2 shares many characteristics with GRUB Legacy, but it also introduces many new changes, this includes better portability, memory management and modularity, supports non-ASCII characters, dynamic loading of modules, and more. There is three locations

The main configuration file has a new name, /boot/grub/grub.cfg, unlike menu.lst, this file can not be edited directly, typically the file is automatically generated by grub-mkconfig command or via update-grub command, which both read scripts located in the /etc/grub.d/ to build grub.cfg file.

Sample grub.cfg file

```
#
# DO NOT EDIT THIS FILE
#
# It is automatically generated by grub-mkconfig using templates
# from /etc/grub.d and settings from /etc/default/grub
#

### BEGIN /etc/grub.d/00_header ###
if [ -s $prefix/grubenv ]; then
  set have_grubenv=true
```

```

load_env
fi
set default="2"
if [ "${prev_saved_entry}" ]; then
    set saved_entry="${prev_saved_entry}"
    save_env saved_entry
    set prev_saved_entry=
    save_env prev_saved_entry
    set boot_once=true
fi
### END /etc/grub.d/05_debian_theme ###

### BEGIN /etc/grub.d/30_os-prober ###
menuentry 'Linux Mint 12 64-bit, 3.0.0-23-virtual (/dev/sda7)' --class
linuxmint --class gnu-linux --class gnu --class os {
    recordfail
    set gfxpayload=$linux_gfx_mode
    insmod gzio
    insmod part_msdos
    insmod ext2
    set root='(hd0,msdos7)'
    search --no-floppy --fs-uuid --set=root f97658d1-ca33-44fe-81fc-
6239f8278a23
    linux /boot/vmlinuz-3.0.0-23-virtual root=UUID=f97658d1-ca33-44fe-
81fc-6239f8278a23 ro quiet splash vt.handoff=7
    initrd /boot/initrd.img-3.0.0-23-virtual
}
menuentry "Failsafe -- openSUSE 12.1 - 3.1.0-1.2 (on /dev/sda3)" --class
gnu-linux --class gnu --class os {
    insmod part_msdos
    insmod ext2
    set root='(hd0,msdos1)'
    search --no-floppy --fs-uuid --set=root ee7d61f1-bb6b-4b95-adba-
602acb6684ca
    linux /vmlinuz-3.1.0-1.2-desktop root=/dev/disk/by-id/ata-
Hitachi_HTS725050A9A364_110227PCK404GLGZS20J-part3 showopts apm=off
noresume edd=off powersaved=off nohz=off highres=off
processor.max_cstate=1 nomodeset xllfailsafe vga=0x317
    initrd /initrd-3.1.0-1.2-desktop
}
### END /etc/grub.d/30_os-prober

### BEGIN /etc/grub.d/40_custom ###
# This file provides an easy way to add custom menu entries.  Simply type
the
# menu entries you want to add after this comment.  Be careful not to
change
# the 'exec tail' line above.
### END /etc/grub.d/40_custom #####

```

Sample /etc/grub.d/ directory

```
# ls -l /etc/grub.d/
-rwxr-xr-x 1 root root 6698 2011-10-01 14:40 00_header
-rwxr-xr-x 1 root root 5522 2011-10-01 14:19 05_debian_theme
-rwxr-xr-x 1 root root 1183 2011-10-23 18:05 06_mint_theme
-rwxr-xr-x 1 root root 7370 2011-10-22 21:01 10_linux
-rwxr-xr-x 1 root root 6518 2011-09-20 12:36 10_lupin
-rwxr-xr-x 1 root root 6344 2011-10-01 14:40 20_linux_xen
-rwxr-xr-x 1 root root 1588 2011-05-03 01:07 20_memtest86+
-rwxr-xr-x 1 root root 7545 2011-10-01 14:40 30_os-prober
-rwxr-xr-x 1 root root 214 2011-10-01 14:40 40_custom
-rwxr-xr-x 1 root root 95 2011-10-01 14:40 41_custom
-rw-r--r-- 1 root root 483 2011-10-01 14:40 README
```

Here the numbering in the script names defines precedence in the order of execution by the command grub-update or grub-mkconfig.

Let's review the scripts:

- 00_header – is the script that loads GRUB settings from /etc/default/grub, including timeout and default boot entry.
- 05_debian_theme – defines the background, colors and themes.
- 10_linux – loads the menu entries for installed operating systems.
- 20_memtest86+ loads the memtest utility.
- 30_os-prober – is the script that will probe hard disks for other operating systems and add them to the boot menu.
- 40_custom is a template that you can use to create additional entries to be added to the boot menu.

```
# If you change this file, run 'update-grub' afterwards to update
# /boot/grub/grub.cfg.
# For full documentation of the options in this file, see:
#   info -f grub -n 'Simple configuration'

GRUB_DEFAULT=2
#GRUB_HIDDEN_TIMEOUT=0
GRUB_HIDDEN_TIMEOUT_QUIET=true
GRUB_TIMEOUT=10
GRUB_DISTRIBUTOR=`lsb_release -i -s 2> /dev/null || echo Debian`
GRUB_CMDLINE_LINUX_DEFAULT="quiet splash"
GRUB_CMDLINE_LINUX=""

# Uncomment to enable BadRAM filtering, modify to suit your needs
# This works with Linux (no patch required) and with any kernel that
obtains
# the memory map information from GRUB (GNU Mach, kernel of FreeBSD ...)
#GRUB_BADRAM="0x01234567,0xfefefefe,0x89abcdef,0xefefefef"

# Uncomment to disable graphical terminal (grub-pc only)
#GRUB_TERMINAL=console

# The resolution used on graphical terminal
```

```
# note that you can use only modes which your graphic card supports via
VBE
# you can see them in real GRUB with the command `vbeinfo'
#GRUB_GFXMODE=640x480

# Uncomment if you don't want GRUB to pass "root=UUID=xxx" parameter to
Linux
#GRUB_DISABLE_LINUX_UUID=true

# Uncomment to disable generation of recovery mode menu entries
#GRUB_DISABLE_RECOVERY="true"

# Uncomment to get a beep at grub start
#GRUB_INIT_TUNE="480 440 1"
```

/etc/default/grub contains menu settings customization, such as default menu entry and boot default timeout.

New changes and features can be summarised as follows.

- grub.cfg is the main configuration
- scripts files located in /etc/grub.d/ are used to generate grub.cfg
- partition numbers in GRUB device names now start at 1, not 0.
- can read files directly from LVM and RAID devices.
- Many facilities are dynamically loaded as modules allowing flexible built of core image.
- Boot stages are no more, image files have been reorganised.
- Supports more file systems, including ext4, NTFS and HFS+.
- After adding or editing of script(s), update-grub command must be run to rebuild grub.cfg with updates.

Used files, terms and utilities:

- /boot/grub/menu.lst, grub.cfg and other variations.
- grub-install
- MBR
- superblock

102.3 Manage shared libraries

Candidates should be able to determine the shared libraries that executable programs depend on and install them when necessary.

Key Knowledge Areas

- Identify shared libraries
- Identify the typical locations of system libraries.
- Load shared libraries.

Introduction

A library is a set of functions that programs can use to implement their functionalities. When building (linking) a program, those libraries can be statically or dynamically linked to an executable. Static link means that the final program will contain the library function within its file. (lib.a) Dynamic link means that the needed libraries are loaded into RAM when the program executes (lib.so).

When writing an application developers make use of existing software libraries to provide functionality needed by their applications. When the final application is compiled there are two options available.

- Build the applications with all required libraries compiled into the application. This is known as statically linking the libraries to the application. This has the advantage that the application will have no dependency problems once installed, but has the negative side effect of increasing the size of the binary because the library is duplicated in every application that statically links the library. The most negative effect though is that the kernel cannot optimise memory by loading the duplicated code only once. Instead the same code is loaded for each application.
- Dynamically link the required dependencies at run time. Dynamically linked executables are much smaller than when the same application is statically linked programs as they do not contain the library code on which they depend. The use of dynamic linking allows running programs, which use the same library, to share one copy of that library, rather than occupy memory with duplicate copies. Because the library code is separate, it is possible for the Linux distributions package management system to update the library code independently of the application. For these reasons, most programs today use dynamic linking.

In order to find libraries, required by an application at runtime, Linux needs to know where to find them. By default Linux looks in the following trusted locations for library files:

- `/lib`: - Used mainly by `/bin` programs.
- `/usr/lib` - Used mainly by `/usr/bin` programs.

Additional locations for library files can specified in the `/etc/ld.so.conf` file. The `ld.so.conf` file may include all files under the `/etc/ld.so.conf.d` directory, depending on your distribution. Listing 5 shows the contents of `/etc/ld.so.conf` on a 64-bit Fedora 12 system.

Content of /etc/ld.so.conf

```

$ cat /etc/ld.so.conf
include ld.so.conf.d/*.conf
$ ls /etc/ld.so.conf.d/*.conf
/etc/ld.so.conf.d/kernel-2.6.31.12-174.2.19.fc12.x86_64.conf
/etc/ld.so.conf.d/kernel-2.6.31.12-174.2.22.fc12.x86_64.conf
/etc/ld.so.conf.d/kernel-2.6.31.12-174.2.3.fc12.x86_64.conf
/etc/ld.so.conf.d/mysql-x86_64.conf
/etc/ld.so.conf.d/qt-x86_64.conf
/etc/ld.so.conf.d/tix-x86_64.conf
/etc/ld.so.conf.d/xulrunner-64.conf

```

In order to optimise library location and loading, the directories in the `ld.so.conf` file and the trusted directories are parsed by the `ldconfig` command to create a fast caches at `/etc/ld.so.cache`. When an application is launched the dynamic loader uses the cached information from `ld.so.cache` to locate files. Any changes to the `ld.so.conf` files requires the `ldconfig` command to be run to update the `/etc/ld.so.cache` file.

The `ldd` command

If you wish to know whether an application is statically linked, or what the dependencies for a dynamically linked application are you can use the `ldd` command. For example `ldd /usr/sbin/apache` show the following:

```

ldd /usr/sbin/apache2
linux-vdso.so.1 => (0x00007ffff85ff000)
libpcre.so.3 => /lib/libpcre.so.3 (0x00007feeaf2e4000)
libaprutil-1.so.0 => /usr/lib/libaprutil-1.so.0
(0x00007feeaf0c1000)
libapr-1.so.0 => /usr/lib/libapr-1.so.0 (0x00007feeae8b000)
libpthread.so.0 => /lib/libpthread.so.0 (0x00007feeaec6e000)
libc.so.6 => /lib/libc.so.6 (0x00007feeae8eb000)
libuuid.so.1 => /lib/libuuid.so.1 (0x00007feeae6e5000)
librt.so.1 => /lib/librt.so.1 (0x00007feeae4dd000)
libcrypt.so.1 => /lib/libcrypt.so.1 (0x00007feeae2a4000)
libdl.so.2 => /lib/libdl.so.2 (0x00007feeae09f000)
libexpat.so.1 => /lib/libexpat.so.1 (0x00007feeade76000)
/lib64/ld-linux-x86-64.so.2 (0x00007feeaf7ab000)

```

This tells us that `apache2` is dynamically linked and relies on the libraries listed. If a library is missing `ldd` will out the dependency with the words “not found”. This can be useful when troubleshooting dependency issues.

The `LD_LIBRARY_PATH`

Sometime you may need to override the default search path for dynamic libraries. This can be for applications you have installed from source, for testing out latest version of a library or if the application relies on an older version of a library that you already have installed on your machine.

To add libraries to the search path you will need to define and export the `LD_LIBRARY_PATH` variable as follows:

```
# export LD_LIBRARY_PATH=/usr/lib/:/opt/someapp/lib;
```

LD_LIBRARY_PATH is a colon-separated list of directories that is searched before the system ones specified in ld.so.cache. This allows for the temporary overriding of libraries defined in the ld.so.cache and in the trusted directories.

Used files, terms and utilities:

- ldd
- ldconfig
- /etc/ld.so.conf
- LD_LIBRARY_PATH

102.4 Debian Package Management

Candidates should be able to perform package management using the Debian package manager.

Key Knowledge Areas

- Install, upgrade and uninstall Debian binary packages.
- Find packages containing specific files or libraries which may or may not be installed.
- Obtain package information like version, content, dependencies, package integrity and installation status (whether or not the package is installed).

Introduction

Most Linux distributions manage software using some form of package management to perform tasks such as installations, updates and queries. There are two main package management system in use today and what your distribution uses will depend on its heritage. Most distributions derived from Redhat use the rpm package manager, while those that are derived from Debian use the dpkg manager.

Debian Package Management

Systems using Debian based variants of Linux use the Debian Package Management system. The Debian system is more rigorous and configurable than the rpm system and is used by Debian derivatives such as Ubuntu. Under Debian system's packages are managed with '**dpkg**' but you may be more familiar with dpkg management tools such as "**apt**" and "**aptitude**".

Package Naming

Debian package names are formed as follows:

name_version-release_architecture.deb

e.g. *xxchat_2.8.6-4ubuntu5_amd64.deb*

The release number indicates which Debian release of the version of the software the package contains, while the architecture name specifies the computer architecture (i386, sparc, all). So the above deb package is for xchat version 2.8.6-4ubuntu5 for the AMD64 architecture.

dpkg

The **dpkg** command is controlled via command line parameters, which consist of an action and zero or more options. The action parameter tells dpkg what to do and options control the behaviour of the action in some way.

dpkg maintains some usable information about available packages. The information is divided in three classes: **states**, **selection states** and **flags**.

Package States

State	Description
-------	-------------

installed	The package is unpacked and configured OK.
half-installed	The installation of the package has been started, but not completed for some reason.
not-installed	The package is not installed on your system.
unpacked	The package is unpacked, but not configured.
half-configured	The package is unpacked and configuration has been started, but not yet completed for some reason.
config-files	Only the configuration files of the package exist on the system.

Package Flags

<i>Flag</i>	<i>Description</i>
hold	A package marked to be on hold is not handled by dpkg , unless forced to do that with option --force-hold .
reinst-required	A package marked reinst-required is broken and requires reinstallation. These packages cannot be removed, unless forced with option --force-reinstreq .

Actions

The heart of dpkg operation is the command line parameters specifying the action which should be performed. While there are a large number of these, the following table summarises the main actions you are likely to require on any regular basis.

<i>Action</i>	<i>Description</i>
-l	Prints a list of the packages installed on the system, or matching a pattern if any is given. The first three characters on each line show the state, selection state, and flags of the package
-s	Shows the status and information about particular installed package(s)
-l	Show information about a package in a .deb file
-L	List the files included in a package
-S	Show the package which includes the file specified
-i	Install (or upgrade) and configure a package from a .deb file
--unpack	Unpack (only) a package in a .deb file
--configure	Configure an unpacked package. With -a (or --pending) configures all packages requiring configuration
-r	Remove a package (but leave its configuration files)
-P	Purge – remove a package along with its configuration files
--get-selections	Get a list of package selections from a system (to stdout)
--set-selections	Set the list of package selections for a system (from stdin)

Options

All options can be specified both on the command line and in the **dpkg** configuration file `/etc/dpkg/dpkg.cfg`. Each line in the configuration file is either an option (exactly the same

as the command line option but without leading dashes) or a comment (if it starts with a #).

Option	Description
--force-thing	Forces dpkg to perform an action which it would normally not take (for example, to ignore dependency information - --force-depends , or to downgrade a package with -force-downgrade)
--refuse-thing	Refuse to do something which dpkg would normally automatically do
--ignore-depends	Ignore dependency checking for a package
--no-act	Show what dpkg would do, but don't do it (also: --simulate)
-R	Recurse through directories (using with -i or --unpack)

Files

dpkg

dpkg uses a number of files in its operation, including **/etc/dpkg/dpkg.cfg** which contains default configuration settings.

Lists of available packages along with their statuses are held in the files

/var/lib/dpkg/available and **/var/lib/dpkg/status**.

A **.deb** file, along with the files making up a packages programs, libraries and configuration, will also include a number of control files which allow the execution of scripts before and after installation and removal, along with lists of files and configuration files. These can be found in the **/var/lib/dpkg/info** directory once the packages are installed.

Use of dpkg

To install a package from a **.deb** file, you could use **dpkg** as follows:



```
# dpkg -i hello_2.1.1-4_i386.deb OR
# dpkg --unpack hello_2.1.1-4_i386.deb
# dpkg --configure hello
```

To remove the hello package along with its configuration, you could use:



```
# dpkg -P hello
```

While:



```
# dpkg -r hello
```

would remove only the package, leaving its configuration files installed.

The get a list of all the packages installed on the system, use the command:

```
# dpkg -l
```

Note that when dealing with a package file, the filename is given, while when dealing with an installed package, only the package name is given.

APT

The **dpkg** tool is fine for installing individual packages with no dependencies, but when installing a number of packages which may have dependencies, the APT tool is generally used instead.

APT is one of the strengths of dpkg, and provides an easy way of installing and updating a system. It is controlled by two files:

File	Description
/etc/apt/apt.conf	Contains general configuration options for APT, such as which release of Debian to install, whether/which proxy settings to use, etc
/etc/apt/sources	Lists sources of Debian files, which may be on CDs, or on the network

In general, to use APT you must first configure the sources it is to used. The main configuration file for apt is the `/etc/apt/source.list` file. This defines the repositories apt should use for installing new software or for updating existing applications. Below is an excerpt from an Ubuntu `/etc/apt/sources.list` file.

```
deb http://archive.ubuntu.com/ubuntu lucid main restricted
deb-src http://archive.ubuntu.com/ubuntu lucid main restricted
deb http://archive.ubuntu.com/ubuntu lucid-updates main restricted
deb-src http://archive.ubuntu.com/ubuntu lucid-updates main restricted
```

The type of archive can be either `deb` for binary packages or `deb-src` for source files. The rest of the line define repository location such by providing the url at which the repository can be found, and details needed to find the specific repository at the url location.

Once APT knows where the Debian packages are located, two command line tools are used for package management: **apt-cache** and **apt-get**.

apt-cache

apt-cache allows manipulation of the APT package cache (which is stored in files in `/var/cache/apt`). An action normally follows apt-cache on the command line, and common options include:

Action	Description
search	Search all the available package descriptions for the string given, and print a short description of the matching package

show	Shows a full description of the package specified
-------------	---

apt-get

While **apt-cache** is useful for finding out information about available packages, **apt-get** allows updating of package information, retrieval, installation and removal of packages, and even upgrading of an entire Debian distribution. apt-get expects an action to be provided on the command line, and the most common are listed below:

Action	Description
update	Update the list of packages from the sources in /etc/apt/sources.list
install package	Install the package(s) specified, along with any dependencies
upgrade	Upgrade any packages which have newer versions available
dist-upgrade	Upgrade entire distribution to the latest release (best to read the release notes first!)
remove	Remove the package(s) specified

Use of APT

One use of APT is for updating the system (for example if security-related updates have become available). This is normally done using the two commands:



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

The other main use of APT is to install required packages. This normally involves the following commands:

```
apt-get update           #update list of packages
apt-cache search frob   #find packages relating to frobbing
apt-cache show frobnicate #show information regarding a particular package
apt-get install frobnicate #install frobnicate package and its dependencies
```

The Alien Tool

The **alien** tool will change Debian packages into RedHat ones and vice versa. One can download it at: <http://kitenet.net/programs/>

Convert a debian package to an rpm:



```
#alien --to-rpm package.deb
```

Convert an rpm package to debian:



```
# alien --to-debian package.rpm
```

Aptitude

Aptitude is another text based front-end to the Debian package management system. Much like apt, aptitude can be used to install, remove and update software on Debian machines. Below is a list of commonly used parameters with aptitude:

- `aptitude update` – update the list of available packages.
- `aptitude install` – will install new software ,
- `aptitude reinstall` – will reinstall an existing package
- `aptitude remove` – will remove an existing package
- `aptitude purge` – will remove a package and its associated configuration files
- `aptitude search` – will allow you to search the list of available packages.

Used files, terms and utilities:

- `/etc/apt/sources.list`
- `dpkg`
- `dpkg-reconfigure`
- `apt-get`
- `apt-cache`
- `aptitude`

102.5 RPM and YUM Package Management

Candidates should be able to perform package management using RPM based tools.

Key Knowledge Areas

- Install, re-install, upgrade and remove packages using RPM and YUM.
- Obtain information on RPM packages such as version, status, dependencies, integrity and signatures.
- Determine what files a package provides, as well as find which package a specific file comes from.

Introduction

Some Linux distribution uses rpm the “Red Hat Package Manager” for all its distribution software. RPM maintains a detailed database of all software installed in the system.

RPM Package Naming

There is no strict convention but most rpm package names are formed as follows:

name-version-release.architecture.rpm

[ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification] Created during the initiative “ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software in Africa”, a programme of FOSSFA and GIZ. For more information see www.ict-innovation.fossfa.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.

The architecture name can either indicate which computer architecture the enclosed binaries are made for (e.g i386, ppc, ia64, noarch) or it can indicate that the package contains the source code (src).

Major and minor modes

Some short name options are similar but perform different actions depending on their position on the command line. A distinction is made between the first option and other options.

The first option given to **rpm** is in major mode. For example in `rpm -iv A.rpm` the option 'i' is a major option and will cause package A to be installed.

Similarly an option that is not in first position is in minor mode. For example in `rpm -qpi A.rpm` the option 'i' is a minor mode and will get information from the package A such as the author and the licence type.

These are the major mode options for **rpm**.

Short	Long	Description
-i	--install	Installs the package
-U	--update	Updates or installs a package
-F	--freshen	Updates only installed package
-V	--verify	file size, MD5, permissions, type ...
-q	--query	Queries installed/uninstalled packages, and files
-e	--erase	Uninstall package

These are the minor mode options for **rpm**.

Short	Description
a	applies to all installed packages
c	together with q lists configuration files
d	together with q lists documentation files
f	together with q queries which package installed a given file
h	adds hashes while processing
i	together with q lists information about a package
l	together with q lists all files and directories in a package
p	together with q specifies that the query is performed on the package file
v	verbose

Query modes

Three query types: uninstalled packages, installed packages and files

Query Type	Option
Package file	-qp
Installed package	-q
File	-qf

An extra option will allow you to get information on all installed files **-l**, documentation **-d** configuration files **-c**, etc ...

We consider for example the package `routed-0.17.i386.rpm`. We can query this package and list its contents before installation with the **l** option as follows:



```
# rpm -qpl routed-0.17.i386.rpm
```

Once this package is installed we can query the installed package with:



```
# rpm -ql routed-0.17          or
# rpm -ql routed
```

Finally if we want to find out which package installed the file `/usr/sbin/routed` the rpm database can be queried with:



```
# rpm -qf /usr/sbin/routed
```

Special Options

<code>--nodeps</code>	Install a package regardless of dependencies
<code>--force</code>	force an upgrade
<code>--test</code>	doesn't actually install or upgrade, just prints to stdout
<code>--requires PACKAGE</code>	together with <code>q</code> lists capabilities required by a package
<code>--whatrequires CAPABILITY</code>	together with <code>q</code> lists packages which require the capability

Package Signatures

You can check the signature of each package that is distributed as part of a project. For example to load the keys of all the developers involved with the Fedora project do the following (just once):



```
# rpm --import /usr/share/rhn/RPM-GPG-KEY-fedora
```

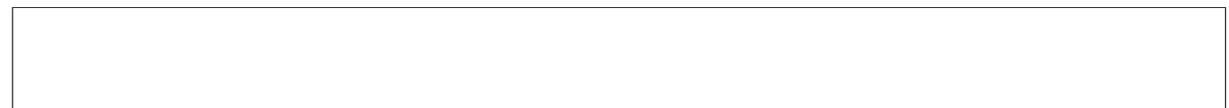
You can now download any package from an FTP site which mirrors the project's RPMs. For example we downloaded `zlib-1.2.1.1-2.1.i386.rpm` from `ftp.mirror.ac.uk` in the Fedora subdirectory. We next check the authenticity of the file:



```
# rpm --checksig /home/adrian/zlib-1.2.1.1-2.1.i386.rpm
# /home/adrian/zlib-1.2.1.1-2.1.i386.rpm: (sha1) dsa sha1 md5 gpg OK
```

Package Integrity

The next command checks the integrity of the package `bash`:



```
# rpm -V bash
```

This returns nothing. We next do the following as user root:



```
# chown bin /bin/bash
# chmod 775 /bin/bash
```

If we check the integrity of **bash** again this time we get:



```
# rpm -V bash
.M...U.. /bin/bash
```

The package manager has compared the current status of all files which are part of the **bash** package with the known original state of these files stored in a database. The changes made to **/bin/bash** have been identified.

It is possible to verify the integrity of all packages installed on the system by adding the 'a' (**--all**) option after 'v' (**--verify**)

The **--verify** option performs a number of tests on each file; when a test is positive a number of characters (listed below) are used to identify the errors:

<i>Returned character</i>	<i>Error description</i>
.	the test was successful
?	the test couldn't be performed
S	file size has changed
M	permission mode or file type has changed
5	the file's MD5 sum has changed
D	device major/minor number miss-match
L	broken symbolic link
U	the user owner of the file has changed
G	the group owner of the file has changed
T	the mtime (modified time) has changed

Yum Package Manager

Yum is the default package manager for many rpm based distributions that allows for the

installing, updating and removing of rpm packages without having to worry about resolving dependencies yourself. Yum uses external repositories to provide the meta-data, in the form of index files, about what packages are available and their dependencies as well as to store the rpm packages that will be automatically downloaded to fulfil an installation request. A repository may be an web site or directory that is formatted in the manner expected by yum.

The most common command for yum are:

install – install a package, automatically resolving and installing dependencies. The command below would install the tsclient package with its dependencies.



```
# yum install tsclient
```

This comand will install all software packages in the “mysql database group”, which will include the mysql server as well as admin and management tools



```
# yum groupinstall "mysql database"
```

update – update the list of available packages and will update all installed packages on your system to the latest available versions.



```
# yum update
```

search – search the list of available rpm packages. The command below will search for a package by name.



```
# yum list tsclient
```

If you do not know the name of the package, which is often the case, you can search for a package by keyword. The search command can also accepts wildcards in its search criteria.



```
# yum search PalmPilot
```

erase/remove – delete a package from your system. Erase is a safer option to use as remove may delete dependencies that are needed by other packages. Below we remove he tsclient package



```
# yum remove tsclient
```

We can also remove all packages in a group.



```
# yum groupremove "mysql database"
```

Yum is configured in the `/etc/yum.conf` file and repositories are managed via the `/etc/yum.repos.d/` directory. To add an extra repository, place a definition file in the `/etc/yum.repos.d/` directory on your system. Package providers make the definition files for their repositories available on their web sites. You may also create a repository file manually by following the correct file format. To disable a repository you can add the line `enable=0` to the repository definition file. If you wish to permanently remove the repository delete the definition file and clear the yum cache directory `/var/cache/yum/`. The cache directory is updated each time you run “**yum update**”

The **yumdownloader** is a program for downloading RPMs from Yum repositories. It is used to download the rpm packages without installing them on the system. This can be useful for downloading packages from a faster connection and then copying them to a machine with a slower connection for installation. The parameters that are commonly used with the yum downloader are:

- **destdir DIR** - specify a destination directory for the download. Defaults to the current directory.
- **resolve** When downloading RPMs, resolve dependencies and also download the required packages.

Used files, terms and utilities:

- | | |
|----------------|--------------------|
| •rpm | •/etc/yum.repos.d/ |
| •rpm2cpio | •yum |
| •/etc/yum.conf | •yumdownloader |

Topic 103: GNU and Unix Commands

103.1 Working on the Command Line

Candidates should be able to interact with shells and commands using the command line. The Objective assumes the bash shell.

Key Knowledge Areas

- Use single shell commands and one line command sequences to perform basic tasks on the command line.
- Use and modify the shell environment including defining, referencing and exporting environment variables.
- Use and edit command history.
- Invoke commands inside and outside the defined path.

Overview

A basic way to interact with a computer system is to use the command line. The shell interprets the instructions typed in at the keyboard. The shell prompt (ending with \$ or # for user root) indicates that it is ready for user input.

The shell is also a programming environment which can be used to perform automated tasks. Shell programs are called scripts.

<i>Most Common Shells</i>	
The Bourne shell	/bin/sh
The Bourne again shell	/bin/bas h
The Korn shell	/bin/ksh
The C shell	/bin/csh
Tom's C shell	/bin/tcs h

Since the bash shell is one of the most widely used shells in the Linux world the LPI concentrates mainly on this shell.

The interactive shell

Shell commands are often of the form **command [options] {arguments}**.

The the bash shell uses the echo command to print text to the screen.



```
$ echo "this is a short line"
```

Full/Relative path

The shell interprets the first "word" of any string given on the command line as a command. If the string is a full or relative path to an executable then the executable is started. If the first word has no "/" characters, then the shell will scan directories defined in the PATH variable and attempt to run the first command matching the string.

For example if the PATH variable only contains the directories **/bin** and **/usr/bin** then the command **xeyes** won't be found since it is stored in **/usr/X11R6/bin/xeyes** so the full path needs to be used



```
$ /usr/X11R6/bin/xeyes
```

An alternative to typing the full path to an executable is to use a **relative** path. For example, if the user

is in the directory where the **xeyes** program is stored then one can type

```


$ ./xeyes
    
```

The '.' in the above command means the current working directory. Since you are already in the current directory the command could also simply be written as: (only if "." is in your search path)

```


$ xeyes
    
```

Shell Variables

Shell variables are similar to variables used in any computing language. Variable names are limited to alphanumeric characters. For example CREDIT=300 simply assigns the value 300 to the variable named CREDIT.

1. initialise a variable:	Variable-Name=value (no spaces !!)
2. reference a variable:	Variable-Name

```


CREDIT=300
echo $CREDIT
    
```

The value of a variable can be removed with the **unset** command.

Export, Set and Env

There are two types of variables: local and exported.

Local variables will be accessible only to the current shell. On the other hand, exported variables are accessible by both the shell and any child process started from that shell.

The commands **set** and **env** are used to list defined variables

<i>The set and env commands</i>	
set	Lists all variables
env	Lists all exported variables

A global variable is global in the sense that any child process can reference it. The example below exports the variable credit and then checks to see if it has been exported as expected.

```


export CREDIT
env | grep CREDIT
    
```

List of common predefined variables

PREDEFINED VARIABLES	MEANING
DISPLAY	Used by X to identify where to run a client application
HISTFILE	Path to the user's .bash_history file
HOME	The path to the user's home
LOGNAME	The name used by the user to log in
PATH	List of directories searched by the shell for programs to be executed when a command is entered without a path.
PWD	The current working directory
SHELL	The shell used (bash in most Linux distributions)
TERM	The current terminal emulation

Special variables

The next few variables are related to process management.



\$! represents the PID value of the last child process
\$\$ represents the PID of the running shell
\$? is 0 if the last command was executed successfully and non-zero otherwise

Metacharacters and Quotes

Metacharacters are characters that have special meaning for the shell. They are mainly used for **globbing**, that is to match several files or directory names using a minimum of letters. The input (<), output (>) and pipe (|) characters are also special characters as well as the dollar (\$) sign used for variables. We will not list them here but note that these characters are seldom used to name regular files.

Wildcards

- The * wildcard can replace any number of characters.



```
$ ls /usr/bin/b*
```

lists all programs starting with a 'b'

- The ? wildcard replaces any one character.



```
$ ls /usr/bin/?b*
```

lists all programs having a 'b' as the second letter

- [] is used to define a range of values.

```

$ ls a[0-9]
$ ls [!Aa]*

```

First line lists all files starting with an 'a' and have a digit in second position.
The second line lists all files that don't start with and 'a' or an 'A'

- {string1,string2}; although not a file naming wildcard, it can be used to generate a list of names that have a common stem.

```

$ mkdir {mon, tues, wednes} day

```

Quotes and escape codes

The special meaning of metacharacters can be cancelled by *escape characters*, which are also metacharacters.

The backslash (\) is called the **escape character** and cancels the meaning of the following character, forcing the shell to interpret it literally.

The single quotes (' ') cancel the meaning of all metacharacters except the backslash.

The double quotes (" ") are the weakest quotes but cancel most of the special meaning of the enclosed characters except the pipe (|), the backslash (\) and a variable (\$var).

The Back Tick

Back quotes ` will execute a command enclosed and substitute the output back on the command line. The next example defines the variable TIME using the **date** command.

```

$ TIME="Today's date is `date +%a:%d:%b`"
echo $TIME
Today's date is Sun:15:Jul

```

Another way of executing commands (similar to the back ticks) is to use **\$()**. This will execute the enclosed command and treat it as a variable.

```

$ TIME=$(date)

```

The Command History

To view the list of previously typed commands you can use the **bash** built-in command history.

```

$ history
1      ls
2      grep  500 /etc/passwd
    
```

This has listed all the cached commands as well as the commands saved in `~/.bash_history`. When a user exits the shell cached commands are saved to `~/.bash_history`.

You can recall commands by using the Up-arrow and Down-arrow on your keyboard. There are also emacs key bindings that enable you to execute and even edit these lines.

<i>Emacs Key Bindings for Editing the Command History</i>	
Ctrl+P	Previous line (same as Up-arrow)
Ctrl+n	Next line (same as Down-arrow)
Ctrl+b	Go back one character on the line (same as Left-Arrow)
Ctrl+f	Go forward one character on the line (Same as Right-Arrow)
Ctrl+a	Go to the beginning of the line (Same as <Home>)
Ctrl+e	Go to the end of the line (Same as <End>)

The **bang (!)** key can be used to rerun a command.

Example

```

!x          executes the latest command in the history list starting with an 'x'
!2          runs command number 2 from the history output
!-2        runs the command before last
!!         runs the last command
^string1^string2  run previous command and replace string1 by string2
    
```

Other Commands

Aliases

You can create aliases for commands needing many arguments. The format to create an alias is

```

$ alias myprog='command [options]{arguments}'
    
```

By typing alias alone at the command line you will get a list of currently defined aliases.

Command completion

By pressing **TAB**, the shell will complete the commands you have started typing in.

Compound commands

command1; command2; command3	The three commands are run in sequence regardless of the success of the previous command
command1 && command2 && command3	Each command will execute only if the previous exit code is 0 (success)
command1 comand2 command3	The next command will execute only if the previous exit code is not 0 (failure)

The "exec" command

This command is not a binary but rather is part of the shell. It is used to start other commands. Ordinarily if a command is executed, a sub-process is started. If the exec command is used to initiate the new program, it reoccupies the process used to start it. It replaces the current shell (in a script or the interactive shell).

When the new command terminates, control is not passed back to the calling shell, but returns to the process that called the shell used to make the exec call.

```

$ echo $$
414

$ bash
$ echo $$
455

$ echo hello
hello
$ echo $$
455

$ exec echo hello
hello
$ echo $$
414

```

The above shows control falling back to the second shell (process 455) after a straight forward echo and the first shell (process 414) using an exec.

Manpages and the whatis database

<i>The manpages are organised in specific topics</i>	
NAME	the name of the item followed by a short one line description.
SYNOPSIS	the syntax for the command
DESCRIPTION	a longer description
OPTIONS	a review of all possible options and their function
FILES	files that are related to the current item (configuration files etc)

The manpages are organised in specific topics

SEE ALSO	other manpages related to the current topic
----------	---

These are the main topic sections one can expect to find in a manpage.

The **whatis** database stores the NAME section of all the manpages on the system. This is updated regularly through a daily **cron**. The **whatis** database has the following two entries:

name(key) - one line description

The syntax for **whatis** is:

```
whatis <string>
```

The output is the full NAME section of the manpages where *string* matched *named(key)*

One can also use the man command to query the **whatis** database. The syntax is

```
man -k <string>
```

This command is similar to **apropos**. Unlike **whatis** this will query both the “name” and the “one line description” entries of the database. If the string matches a word in any of these fields the above query will return the full *NAME* section.

Example: (the matching string has been highlighted)

```


whatis lilo

lilo                (8) - install boot loader
lilo.conf [lilo]    (5) - configuration file for lilo

```

```


man -k lilo

grubby              (8) - command line tool for configuring grub, lilo, and
elilo
lilo                (8) - install boot loader
lilo.conf [lilo]    (5) - configuration file for lilo

```

The filesystem hierachy standard, a recommended layout for Linux filesystems, recommends manpages to be kept in **/usr/share/man**. However additional locations can be searched using the **MANPATH** environment variable set in **/etc/man.config**. Each directory is further divided into subdirectories corresponding to manpage sections.

Manpage Sections	
Section 1	Information on executables
Section 2	System calls, e.g mkdir(2)

Manpage Sections	
Section 3	Library calls, e.g stdio(3)
Section 4	Devices (files in /dev)
Section 5	Configuration files and formats
Section 6	Games
Section 7	Macro packages
Section 8	Administration commands
Section 9	Kernel routines

Sometimes manpages with the same name are present in more than one section.

To access a specific section *N* one has to enter:

man *N* command

Examples:

```


$ man mkdir
$ man 2 mkdir

```

```


$ man crontab
$ man 5 crontab

```

file

`file` is used to try and detect what type a particular file is.

For example

```


$ file picture.png
picture.png: PNG image, 179 x 179, 8-bit/color RGBA, non-interlaced

```

The utility will identify files that have been incorrectly named so if `picture.png` had been named `readme.txt` the command “`file readme.txt`” would still identify the file as a png file.

uname

The `uname` command prints information relating to the kernel version, machine name, processor type and node name. It is most commonly used to identify which version of the kernel a machine is running.

```


$ uname -r

```

Prints the currently running kernel's version number.

pwd

This is a command which simply prints out the current working directory for the shell.

Used files, terms and utilities:

- bash
- echo
- env
- exec
- export
- pwd
- set
- unset
- man
- uname
- history

103.2 Process text streams using filters

Candidates should be able to apply filters to text streams.

Key Knowledge Areas

- Send text files and output streams through text utility filters to modify the output using standard UNIX commands found in the GNU textutils package.

Text Processing Utilities

Linux has a rich assortment of utilities and tools for processing and manipulating text files. In this section we cover some of them.

cat - cat is short for concatenate and is a Linux command used to write the contents of a file to standard output. Cat is usually used in combination with other command to perform manipulation of the file or if you wish to quickly get an idea of the contents of a file. The simplest format of the command is:

```
# cat /etc/aliases
```

Cat can take several parameters, the most commonly used being `-n` and `-b` which output line numbers on all lines and non-empty lines only respectively.

head and **tail** - The utilities head and tail are often used to examine log files. By default they output 10 lines of text. Here are the main usages.

List 20 first lines of `/var/log/messages` :

```
# head -n 20 /var/log/messages
# head -20 /var/log/messages
```

List 20 last lines of `/etc/aliases`:

```
# tail -20 /etc/aliases
```

The **tail** utility has an added option that allows one to list the end of a text starting at a given line.

List text starting at line 25 in `/var/log/messages`:

```
# tail +25 /etc/log/messages
```

Finally **tail** can continuously read a file using the **-f** option. This is most useful when you are examining live log files for example.

wc -The **wc** utility counts the number of *bytes*, *words*, and *lines* in files. Several options allow you to control **wc**'s output.

Options for wc

-l	count number of lines
-w	count number of words
-c or -m	count number of bytes or characters

nl - The **nl** utility has the same output as **cat -b**

Number all lines including blanks

```
# nl -ba /etc/lilo.conf
```

Number only lines with text

```
# nl -bt /etc/lilo.conf
```

expand/unexpand - The **expand** command is used to replace TABs with spaces. One can also use **unexpand** for the reverse operations.

od There are a number of tools available for this. The most common ones are **od** (octal dump) and **hexdump**.

split - splitting files - The **split** tool can split a file into smaller files using criteria such as size or number of lines. For example we can split */etc/passwd* into smaller files containing 5 lines each

```
# split -l 5 /etc/passwd
```

This will create files called *xaa*, *xab*, *xac*, *xad* ... each file contains at least 5 lines. It is possible to give a more meaningful prefix name for the files (other than 'x') such as *'passwd-5.'* on the command line

```
# split -l 5 /etc/passwd passwd-5
```

This has created files identical to the ones above (*aa*, *xab*, *xac*, *xad* ...) but the names are now *passwd-5aa*, *passwd-5ab*, *passwd-5ac*, *passwd-5ad* ...

Erasing consecutive duplicate lines

The `uniq` tool will send to `stdout` only one copy of consecutive identical lines.

Consider the following example:

```


# uniq > /tmp/UNIQUE
  line 1
  line 2
  line 2
  line 3
  line 3
  line 3
  line 1
^D

```

The file `/tmp/UNIQUE` has the following content:

```


# cat /tmp/UNIQUE
  line 1
  line 2
  line 3
  line 1

```

NOTE:

From the example above we see that when using `uniq` non consecutive identical lines are still printed to `STDOUT`. Usually the output is sorted first so that identical lines all appear together.

```


# sort | uniq > /tmp/UNIQUE

```

cut The `cut` utility can extract a range of characters or fields from each line of a text. The `-c` option is used to cut based on character positions.

Syntax:

cut {*range1*,*range2*}

Example

```


# cut -c5-10,15- /etc/password

```

The example above outputs characters 5 to 10 and 15 to end of line for each line in `/etc/passwd`. One can specify the field delimiter (a space, a comma etc ...) of a file as well as the fields to output. These options are set with the `-d` and `-f` flags respectively.

Syntax:

```
{delimiter} -f {fields}
```

Example:



```
# cut -d: -f 1,7 --output-delimiter=" " /etc/passwd
```

This outputs fields 1 and 7 of `/etc/passwd` delimited with a space. The default `output-delimiter` is the same as the original input delimiter. The `--output-delimiter` option allows you to change this.

paste/join - The easiest utility is `paste`, which concatenates two files next to each other.

Syntax:

```
paste text1 text2
```

With `join` you can further specify which fields you are considering.

Syntax:

```
join -j1 {field_num} -j2{field_num} text1 text2 or
```

```
join -1 {field_num} -2{field_num} text1 text2
```

Text is sent to stdout only if the specified fields match. Comparison is done one line at a time and as soon as no match is made the process is stopped even if more matches exist at the end of the file.

sort - By default, `sort` will arrange a text in alphabetical order. To perform a numerical sort use the `-n` option.

Formatting output with `fmt` and `pr`

fmt is a simple text formatter that reformats text into lines of a specified length.

You can modify the number of characters per line of output using `fmt`. By default `fmt` will concatenate lines and output 75 character lines.

fmt options

-w number of characters per line

-s split long lines but do not refill

-u place one space between each word and two spaces at the end of a sentence

Long files can be paginated to fit a given size of paper with the `pr` utility. Text is broken into pages of a specified length and page headers are added. One can control the page length (default is 66 lines) and page width (default 72 characters) as well as the number of columns.

pr can also produce multi-column output.

When outputting text to multiple columns each column will be evenly truncated across the defined page width. This means that characters are dropped unless the original text is edited to avoid this.

tr The tr utility translates one set of characters into another.

Example changing uppercase letters into lowercase

```
tr 'A-Z' 'a-z' < file.txt
```

Replacing delimiters in **/etc/passwd**:



```
# tr ':' ' ' < /etc/passwd
```

NOTE: tr has only **two arguments!**.

sed sed stands for stream editor and is used to manipulate text stream **tr** will not read from a file, it only reads standard input. It is most commonly used to transform text input generated by other commands in bash scripts. sed is a complex tool that can take some time to master. It's most common use case is to find and replace text in an input stream. sed's output is written to standard out, with the original file left untouched, and needs to be redirected to a file to make the changes permanent.

The command:



```
# sed 's/linux/Linux/g' readme.txt > ReadMe.txt
```

will replace every occurrence of the word linux with Linux in the readme.txt file. The g at the end of the command is used to make the replacement global so sed will process the entire line and not stop at the first occurrence of the word linux. For more information on sed refer to section 103.7

Used files, terms and utilities:

- cat
- cut
- expand
- fmt
- head
- od
- join
- nl
- paste
- pr
- sed
- sort

- split
- tail
- tr
- unexpand
- uniq
- wc

103.3 Perform basic file management

Candidates should be able to use basic Linux commands to manage files and directories.

Key Knowledge Areas

- Copy, move and remove files and directories individually.
- Copy multiple files and directories recursively.
- Remove files and directories recursively.
- Use simple and advanced wildcard specifications in commands.
- Using find to locate and act on files based on type, size, or time.
- Usage of tar, cpio and dd.

Moving Around the Filesystem

Absolute and relative paths

A directory or a file can be accessed by giving its full pathname, starting at the root (/) or its relative path, starting from the current directory.

Absolute path: independent of the user's current directory, starts with /

Relative path: depends on where the user is, doesn't start with /

As in any structured filesystem there are a number of utilities that can help you navigate through the system.

pwd: Gives your actual position as an absolute path.

cd: The 'change directory' command

ls: List the contents of a directory.

The command can take several parameters the most common of which are:

- l – use the long listing format,
- a – list all files and directories including hidden files and directories,
- h – show file sizes in human readable format, ie. Formatted for easy reading
- d – list directories only and does not list their contents.

Finding Files and Directories

We will describe the **find**, **which**, **whereis** and **locate** utilities.

find

Syntax:

```
find <DIRECTORY> <CRITERIA> [-exec <COMMAND> {} \;]
```

The *DIRECTORY* argument tells **find** where to start searching and *CRITERIA* can be a combination of serial selection criteria, including the name of a file or directory we are looking for.

Examples:



```
# find /usr/X11R6/bin -name "x*".
# find / -user 502
```

The names of matching files are listed to standard output. Alternatively, a specific operation can be performed on each file found. For example to delete the file, or change the permission. The **find** tool has the built-in option **-exec** which allows you to do that. For example, remove all files belonging to user 502:



```
# find / -type f -user 502 -exec rm -f {} \;
```

Common criteria switches for find

-type	specify the type of file
-name	name of the file (can include wildcards)
-user	user owner
-atime, ctime, mtime	access, creation and modified times (multiples of 24 hrs)
-amin, cmin, mmin	access, creation and modified times (multiples of 1 min)
-newer FILE	files newer than FILE

Handling directories

Creating a directory

When making a directory you can set the permission mode with the **-m** option. Another useful option is **-p** which creates all subdirectories automatically as needed.

Example:



```
# mkdir -p docs/programs/versions
```

Removing directories:

To remove a directory use either **rmdir** or **rm -r**. **rmdir** will only remove empty directories. Specify **-f** to force the deletion of files on which you do not have write permission..

Notice: **rm -rf /dir1/*** removes all files and subdirectories leaving dir1 empty

rm -rf /dir1/ removes all files and subdirectories including dir1

Using cp and mv

cp

Syntax:

```
cp [options] file1 file2
cp [options] files directory
```

It is important to notice that **cp file1 file2** makes a new copy of *file1* and leaves *file1* unchanged. You can also copy several files to a directory, using a list or wildcards. The following table lists the most used options.

Most common options for cp	
-d	do not follow symbolic link (when used with -R)
-f	force
-i	interactive, prompt before overwrite
-p	preserve file attributes
-r	recursively copy directories

Note: `cp -r /mydir/* /dir2/` will copy all files and subdirectories omitting *mydir*
`cp -r /mydir/ /dir2/` will copy all files and subdirectories including *mydir*

mv

Syntax:

```
mv [options] oldname newname
mv [options] source destination
mv [options] source directory
```

The **mv** command can both *move* and *rename* files and directories. If *oldname* is a file and *newname* is a directory then the file *oldname* is moved to that directory.

If the source and destination are on the same filesystem, then the file isn't copied but the the link is simply moved to the new location. Most common options are **-f** force overwrite and **-i** query interactively.

touch and dd

touch

Another way of creating or modifying a file is to use **touch**.

Syntax: `touch {options} file(s)`

If *file* doesn't exist it is created. You can also update the access time of a file to the current time using the **-a** option, **-m** changes the modification time and **-r** is used to apply the time attributes of another file.

Example:

```
touch file1.txt file2.txt          creates new files
touch myfile -r /etc/lilo.conf     myfile gets the time attributes of lilo.conf
```

dd

This command copies a file with a changeable I/O block size. It can also be used to perform conversions (similar to **tr**). Main options are **if=** (input file) **of=** (output file) **conv=** (conversion)

The conversion switch can be: **lcase** **ucase** **ascii**

Example:

```
# dd if=/dev/sda1 of=/dev/sda2
```

Notice that unlike **cp** the **dd** tool will copy portions of a device and preserve the underlying filesystem. On the other hand **cp** only deals with the data and will transfer it from one filesystem to another:

File Archiving and Compression

Linux has several utilities for compressing and archiving files. Some of these tools have their origins in tape archiving and backup solutions and the parameters and names reflect this.

tar

The **tar** (tape archive) command is used to archive directories and optionally compress the archive. Originally the **tar** command was used to archive to tape but can now also archive to disk, which is its most common use. An archive is created as follows:



```
# tar -cvjf backup.tar.bz /home/user1
```

This will create a bzip compressed archive of user1's home directory. The options provided to **tar** are:

- **c** – create the archive,
- **v** – show verbose output during archive creation,
- **j** – compress the archive with bzip compression, alternatively you could stipulate **z** which would use gzip compression
- **f** – the name of the file to created, in this case **backup.tar,bz**

To extract the **backup.tar.bz** archive you would use the following command



```
# tar -xvjf backup.tar.bz
```

This would extract the archive to the current directory. The command line parameters are mostly the same as the above example except for the **-x** (extract) parameter which replaces **-c** (create) To list the contents of an archive without extracting it you would use the **-t** parameter:



```
# tar -tf backup.tar.bz
```

cpio

`cpio` is an older archive utility that does not support compression natively. `cpio` stands for copy in/out. Although `cpio` has been largely superseded by `tar` it is still used in Linux. In particular the `initrd` image file is a `cpio` archive. `cpio` expects a list of files to archive on standard input and so is usually used in combination with the `find` or `ls` command.



```
$ ls | cpio -ov > backup.cpio
```



```
$ find / | cpio -ov > backup.cpio
```

The above two commands create an archive using `cpio`. The `v` parameters tells `cpio` to provide verbose output during archive creation. To extract an archive you will use a command such as:



```
$ cpio -iv < backup.cpio
```

This will extract the `cpio` archive to the current directory. One of the tricky things to remember with `cpio` is its parameters. With archiving we usually talk of archiving and extracting which suggest the parameters `-c` for creating and `-x` or `-e` for extraction. The easiest way to remember that `-o` is for creating and `-i` is for extraction is to remember `cpio` stands for copy in/ copy out. You will create an archive by copying out from the filesystem and extract an archive by copying in from an archive.

gzip/gunzip

`gzip` is used to compress files using Lempel–Ziv coding. As with most Linux commands it can take a plethora of parameters but is most commonly used as follows:



```
$ gzip largefile.txt
```

By default `gzip` creates an output file with the same name as the input file but with the extension `.gz` added. The above command would create a compressed file with the name `largefile.txt.gz`. To uncompress the file you would run the command



```
$ gunzip largefile.txt.gz      or  
$ gzip -d largefile.txt.gz
```

bzip/bzip2

bzip compresses files using the Burrows-Wheeler block sorting text compression algorithm, and Huffman coding, which is considered more efficient than the Lempel-Ziv file. Its most commonly used format for compressing a file take a number from 1 – 9 as a parameter. This number is used to tell bzip2 to use the least efficient but fastest compression block size for 1 and use the most efficient but slowest compression block size for 9. If no number is specified the default value of 9 is used.



```
$ bzip2 -9 largefile.txt
```

It will create the smallest file with the original file name and .bz appended. In our case the file would be largefile.txt.bz. To uncompress the file you would use



```
$ bzip2 -d largefile.txt.bz2    or  
$ bunzip2 largefile.txt.bz2
```

Used files, terms and utilities:

- cp
- find
- mkdir
- mv
- ls
- rm
- rmdir
- touch
- tar
- cpio
- dd
- file
- gzip
- gunzip
- bzip2
- file globbing

103.4 Streams, Pipes and Re-directs

Candidates should be able to redirect streams and connect them in order to efficiently process textual data. Tasks include redirecting standard input, standard output and standard error, piping the output of one command to the input of another command, using the output of one command as an argument for another command and sending output to both standard output and a file.

Key Knowledge Areas

- Redirecting standard input, standard output and standard error.
- Pipe the output of one command to the input of another command.
- Use the output of one command as arguments to another command.
- Send output to both `stdout` and a file.

Input, Output, Redirection

UNIX processes use streams to get input, (standard input stream) send output to, (standard output stream) and a stream to send error messages to (the standard error stream). These streams can be redirected for any process. In most cases standard input (`stdin`) is the keyboard, and the two output descriptors, standard out (`stdout`) and standard error (`stderr`), go to the screen. Sometimes it is convenient to redirect these standard streams so that the process received input from a file and/or sends output to file.

<i>Numerical values for <code>stdin</code>, <code>stderr</code> and <code>stdout</code></i>	
<code>stdin</code>	0
<code>stdout</code>	1
<code>stderr</code>	2

When redirecting or interacting with these streams we refer to them by their numerical values.

Standard Out Redirection

To redirect standard output from the screen to a file for example you use the ">" symbol.

For example



```
$ find / -iname *.txt > textfiles.txt
```

This will run the `find` utility and output the result to the `textfiles.txt` file. No output is visible on the screen.. The `textfiles.txt` file will be created first if it doesn't exist and overwritten if not. To append to a file rather than create a new one the '>>' operator can be used.

Standard Error Redirect

Standard error redirect uses the same format as for standard input redirect but you need to specify that the `stderr` stream and not `stdout` is to be redirected. This is done via placing the `stderr` stream id before the redirect symbol.



```
$ myapp 2> error.txt
```

As per the `stdout` example above this will create a new file. To append to an existing file you would use “2>>” to redirect standard error.

Redirect `stdout` and `stderr`

To redirect `stdout` and `stderr` at the same time you would use the “&>” or “&>>” operator. This will direct both standard out and standard error to the same file.

Standard In Redirection

To have the process read input from a file rather than get its input from the keyboard you would use the “<” symbol as in the example below:



```
$ mysql -u root -p < createtable.sql
```

Here the `mysql` command line interface is told to take its standard input from a file called `createtable.sql` rather than read input from the keyboard. This file would contain SQL statements necessary to create a table for example.

Piped Commands

The pipe command is used to redirect the standard output from one process to the standard input of another.

```
program1 | program2
```

Pipes are represented by the “|” symbol. The data stream goes from the left to the right. The next figure illustrates how the `stdout` for one process is redirected to the `stdin` for another process.



```
# ls-l | less
```

The tee Command

```
command | tee FILENAME
```

This command is used after a pipe and takes a filename as an argument. The standard output from the previous command is then sent to the file given as an argument but **tee** also lets the stream through to **stdout**. The **stdout** has been duplicated in this way.

xargs

This tool is often thought of as a companion tool to **find**. In fact **xargs** will process each line of standard output as an *argument* for another tool. We could use **xargs** to delete all files belonging to a user with:



```
$ find / -type f -user 502 | xargs rm -f
```

If the list of filenames is very long, `xargs` will split it into pieces and invoke the `rm` command several times, one for each piece. This is sometimes useful if the argument list would otherwise be too long to handle.



```
$ ls |xargs rm -f
```

Used files, terms and utilities:

- tee
- xargs

103.5 Create, Monitor and Kill Processes

Candidates should be able to perform basic process management.

Key Knowledge Areas

- Run jobs in the foreground and background.
- Signal a program to continue running after logout.
- Monitor active processes.
- Select and sort processes for display.
- Send signals to processes.

When the shell runs a command, it normally waits and will not prompt for further input until that command has completed. The command is said to run in the foreground.

When a program is running in the foreground it is possible to recover the shell prompt but only by interrupting the program for while. The interruption signal is Ctrl Z.

Starting and Stopping Jobs

A process started from a shell is also called a *job*. Once the job receives the **^z** signal it is stopped and the shell prompt is recovered. To restart the program in the background simply type: **bg**.

Example

	<pre>\$ xclock</pre>	xclock running in foreground, shell prompt lost
	<pre>[1]+ Stopped xclock</pre>	xclock received ^Z signal
	<pre>\$ bg</pre>	shell prompt recovered, issue the bg command
	<pre>[1]+ xclock &</pre>	xclock is running in the background

Notice the [1]+ symbol above. The integer is the process' *job number*, which it can be referred to as.

The '+' sign indicates the last modified process. A '-' sign would indicate the second last modified process. One can start a process in the background by appending a **&** to the command.

	<pre>\$ xclock&</pre>
	<pre>[1] 6213</pre>

The numbers reported here are the job numbers (in square brackets), and the process ID.

Listing jobs

The `jobs` utility lists all running processes started from the current shell. The *job number*, the job's state (running/stopped), as well as the two last modified processes, will be listed.

Output for jobs	
[1]- Stopped	xclock
[2] Running	xman &
[3]+ Stopped	xload

The job number

One can conveniently stop and start a selection of jobs using the *job number*. This is achieved with the `fg` command.

Calling job 2 to the foreground and killing job 1



```
fg 2          or          kill -9 %1
fg %2        or
fg %?xma
```

Avoiding HUP with nohup

There is a program called `nohup` which acts as a parent process independently from the user's session. When a user logs off, the system sends a HUP signal to all processes owned by that process group. For example, to avoid this HUP signal a script called `bigbang` which attempts to calculate the age of the Universe should be started like this:



```
$ nohup bigbang &
```

Viewing Running Processes

Processes have a unique Process ID the PID. This number can be used to modify a process' priority or to stop it. A process is any *running* executable. If process_2 has been spawned by process_1, it is called a *child* process. The spawning process_1 is called the *parent* process.

The `ps` command gives a good illustration of *parent* and *child* process hierarchy.

```

bash(1046)---xinit(1085)-+-X(1086)
      ^-xfwm(1094)-+-xfce(1100)---xterm(1111)---bash(1113)-+-
pstree(1180)
      |
soffice.bin(1139)---soffice.bin(1152)-+-
      |
-soffice.bin(1153)
      |
|-soffice.bin(1154)
      |
|-soffice.bin(1155)
      |
|-soffice.bin(1156)
      |
^-soffice.bin(1157)
      |
xclock(1138)
      |
      | -xfgnome(1109)
      | -xfpager(1108)
      | -xfsound(1107)
      ^-xscreensaver(1098)
  
```

Figure 103.5.1: Part of the *ps* tree output

In the above figure all the process' PIDs are shown; these are clearly incremental. The most common used options are **-p** to display PIDs and **-h** to highlight a users processes only.

A more direct way to determine which processes are running is to use *ps*. Most users learn a favourite combination of options which work for most situations.

Here are three such options:

- ps ux** all processes run by the user
- ps T** processes run under the current terminal by the user
- ps aux** all processes on the system

It is recommended you read the **ps manpage** and choose your own best options!

ps accommodates UNIX-style and BSD-style arguments

```

usage: ps -[Unix98 options]
       ps [BSD-style options]
       ps --[GNU-style long options]
       ps --help for a command summary
  
```

Summary of options

- a** show all processes for the current user linked to a tty (*except* the session leader)
- e** or **-A** show all processes
- f** gives the PPID (Parent Process ID) and the STIME (Start Time)
- l** is similar to **-f** and displays a long list
- a** show all processes linked to a tty, including other users
- x** show all processes without a controlling tty as well

Sending Signals To Processes

The **kill** command can be used to send *signals* to processes. There are 63 signals available. The default signal terminates a process and is called **SIGTERM** with value 15.

kill

Syntax

```
kill SIGNAL process_PID
```

Unless you are root, you can only send signals to processes that you own. Every process can choose whether or not to catch a signal except for the **SIGKILL** which is dealt with by the kernel. Most daemons use **SIGHUP** to mean “re-read configuration file”.

Most Common Signals

- 1 or **SIGHUP** hangup or disconnect the process
- 2 or **SIGINT** same as Ctrl+C interrupt
- 3 or **SIGQUIT** quit
- 9 or **SIGKILL** kill the process through a kernel call
- 15 or **SIGTERM** terminate a process 'nicely'. This is the **DEFAULT** signal.

One can also stop processes without knowing the process' PID using **killall**.

killall

Syntax

```
killall SIGNAL process_NAME
```

Used files, terms and utilities:

- &
- bg
- fg
- jobs
- kill
- nohup
- ps
- top
- free
- uptime
- killall

103.6 Modify Process Execution Priorities

Candidates should be able to manage process execution priorities.

Key Knowledge Areas

- Know the default priority of a job that is created.

- Run a program with higher or lower priority than the default..
- Change the priority of a running process.

Process Priority

When a process is started by a user it has a default priority, or **nice** number of 0. Nice numbers (NI) alter the CPU priority and are used to balance the CPU load in a multiuser environment. Nice numbers range from **19** [lowest] to **-20** [highest].

Only root can decrease the nice number of a process. Since all processes start with a default nice number of zero as a consequence negative nice numbers can only be set by root!

To modify a process' priority that is already running use **renice**. To set a process' priority use **nice**.

Syntax



```
# nice -<NI> <process>
# renice <+/-NI> -p <PID>
```

Notice that **renice** works with PIDs and handles lists of processes at a time. A useful option to **renice** is the **-u** option which affects all processes run by a user.

Set nice number 1 for processes 234 and 765:



```
# renice +1 -p 234 765
```

Set nice number -5 for **xclock**:



```
# nice --5 xclock
```

Continuously updating process information

The **top** utility will update information on processes at an adjustable rate. While **top** is running you can type **h** for a list of commands. The space bar will update information instantly. You can also use **top** to change a process' priority as we shall see in the next section. **top** provides a convenient summary of number of processes, users, length of time the machine has been up and the load average for the past 5, 10 and 15 minutes. You can also obtain this information once off with the **uptime** command. Below is an example of the output you can expect from running this command.

```
13:21:48 up 5:54, 1 user, load average: 0.92, 0.70, 0.66
```

Here 13:21 is the current time, 5:54 is the length of time the machine has been up, 1 user is

logged in and the load average for the past 5,10 and 15 minutes is 0.82, 0.70 and 0.66.

Used files, terms and utilities:

- nice
- ps
- renice
- top

103.7 Using Regular Expressions

Candidates should be able to manipulate files and text data using regular expressions. This objective includes creating simple regular expressions containing several notational elements. It also includes using regular expression tools to perform searches through a filesystem or file content.

Key Knowledge Areas

- Create simple regular expressions containing several notational elements.
- Use regular expression tools to perform searches through a filesystem or file content.

Overview

Finding a word or multiple words in a text is achieved using **grep**, **fgrep** or **egrep**. The keywords used during a search are a combination of letters called *regular expressions*. Regular expressions are recognised by many other applications such as **sed**, and **vi**.

Regular Expressions

Traditional Regular Expressions (regex)

A regular expression is a sequence of characters (or atoms) used to match a pattern. Characters are either constants (treated literally) or metacharacters.

Table1: Main metacharacters

Characters	Search Match
\<KEY	Words beginning with 'KEY'
WORD\>	Words ending with 'WORD'
^	Beginning of a line
\$	End of a line
[Range]	Range of ASCII characters enclosed
[^c]	Not the character 'c'
\[Interpret character '[' literally
"ca*t"	Strings containing 'c' followed by no 'a' or any number of the letter 'a' followed by a 't'
"."	Match any single character

Extended regex:

The main regex's are: +, ?, () and |

Table2: List of main regex

Characters	Search Match
"A1 A2 A3 "	Strings containing 'A1' or 'A2' or 'A3'
"ca+t"	Strings containing a 'ca' followed by any number of the letter 'a' followed by a 't'
"ca?t"	Strings containing 'c' followed by no 'a' or exactly one 'a' followed by a 't'

The grep family

The grep utility supports regular expressions *regex* such as those listed in Table1.

Working with basic grep

Syntax for grep:

grep PATTERN FILE

Options for grep include:

grep	Main Options
-c	count the number of lines matching PATTERN
-f	obtain PATTERN from a file
-i	ignore case sensitivity
-n	Include the line number of matching lines
-v	output all lines except those containing PATTERN
-w	Select lines only if the pattern matches a whole word.

For example list all non blank lines in /etc/lilo.conf :

```
$ grep -v "^$" /etc/lilo.conf
```

egrep

The egrep tool supports extended regular expressions *eregex* such as those listed in Table2.

The egrep utility will handle any modern regular expressions. It can also search for several keywords if they are entered at the command line, separated by the vertical bar character.

For example;

```
$  egrep 'linux|^image' /etc/lilo.conf
```

fgrep

fgrep stands for *fast grep* and fgrep interprets strings literally (no regex or eregex support). The **fgrep** utility does not recognise the special meaning of the regular expressions.

For example



```
$ fgrep 'cat*' FILE
```

will only match words containing 'cat*'. The main improvement came from **fgrep**'s ability to search from a list of keywords entered line by line in a file, say LIST. The syntax would be



```
$ fgrep -f LIST FILE
```

The Stream Editor - sed

sed performs automatic, non-interactive editing of files. It is often used in scripts to search and replace patterns in text. It supports most regular expressions.

Syntax for sed:

```
sed [options] 'command' [INPUTFILE]
```

The input file is optional since **sed** also works on file redirections and pipes. Here are a few examples assuming we are working on a file called MODIF.

Delete all commented lines:



```
$ sed '/^#/ d' MODIF
```

Notice that the search pattern is between the double slashes.

Substitute /dev/hda1 by /dev/sdb3:



```
$ sed 's/\/dev\/hda1/\/dev\/sdb3/g' MODIF
```

The **s** in the command stands for 'substitute'. The **g** stands for "globally" and forces the substitution to take place throughout each line. You can also specify which line numbers the substitutions should occur on, either using line numbers or regular expression match.

If the line contains the keyword KEY then substitute ':' with ';' globally:



```
$ sed '/KEY/ s/:/;/g' MODIF
```

More Advanced sed

You can issue **several commands** each starting with **-e** at the command line. For example, (1) delete all blank lines then (2) substitute 'OLD' by 'NEW' in the file MODIF



```
$ sed -e '/^$/ d' -e 's/OLD/NEW/g' MODIF
```

These commands can also be written to a file, say `COMMANDS`. Then each line is interpreted as a new command to execute (no quotes are needed).

An example `COMMANDS` file

```
1 s/old/new/
/keyword/ s/old/new/g
23,25 d
```

The syntax to use this `COMMANDS` file is:

```
sed -f COMMANDS MODIF
```

This is much more compact than a very long command line !

Summary of options for `sed`

Command line flags

-e	Execute the following command
-f	Read commands from a file
-n	Do not printout unedited lines

`sed` commands

d	Delete an entire line
r	Read a file and append to output
s	Substitute
w	Write output to a file

Used files, terms and utilities:

- grep
- egrep
- fgrep
- sed
- regex (7)

103.8 Using Vi

Candidates should be able to edit text files using *vi*. This objective includes *vi* navigation, basic *vi* modes, inserting, editing, deleting, copying and finding text.

Key Knowledge Areas

- Navigate a document using *vi*.
- Use basic *vi* modes.
- Insert, edit, delete, copy and find text.

In most Linux distributions **vi** is the text editor of choice. It is considered an essential admin tool such as **grep** or **cat**.

Modes

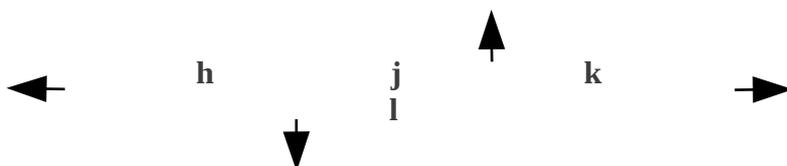
In order to perform complex operations such as copy/paste, **vi** can operate in different modes.

Command mode

This is the editing and navigation mode. Commands are often just a letter. For example use **j** to jump to the next line.

As a rule of thumb if you want to perform an operation several times you can precede the command by a number. For example **10j** will jump 10 lines.

In some situations the arrow keys on the keyboard are not mapped properly, it is still possible to navigate using the commands **h j k l** with the following effect:



Last Line (or column) Mode

You enter this mode from the command mode by typing a colon. The column will appear at the bottom left corner of the screen. In this mode you can perform a simple search operation, save, quit or run a shell command.

Insert Mode

The easiest way to enter this mode while in command mode is to use **i** or **a**. This mode is the most intuitive and is mainly used to interactively enter text into a document.

□ The *Esc* key will exit the *insert mode* and return to *command mode*

Text Items

Items such as words and paragraphs are defined in *command mode* to allow editing commands to be applied to text documents without using a mouse.

Word, sentences and paragraphs

e resp. b	Move to the end/beginning of the current word
(resp.)	Move to the beginning/end of the current sentence
{ resp. }	Move to the beginning/end of the current paragraph
w	Similar to e but includes the space after the word

Beginning and End

^	Beginning of line
\$	End of line
1G	Beginning of file
G	End of file

All these text items can be used to navigate through the text one word (**w**) or paragraph (**}**) at a time, go to the beginning of a line (**^**) the end of the file (**G**) etc. One can also use these text items to execute commands such as deleting and copying.

Inserting Text

When in command mode typing **i** will allow you to enter text in the document interactively. As with all other features in **vi** there are many other ways of doing this. The table below lists the commands used to enter insert mode.

Insert commands

a	Append text with cursor on the last letter of the line
A	Append text with cursor after last letter at the end of the line
i	Insert text at the current position
o	Insert text on a new line below
O	Insert text on a new line above
s	Delete the current letter and insert
S	Delete current line and insert

A very useful option when modifying a document is to delete a section of text you wish to replace just before entering insert mode. This is done by the *change* **c** command. As the other commands in this section **c** will put you into *INSERT* mode but you can specify which portion of the text needs to be deleted before. For example:

c\$

will delete all the text from the current cursor position to the end of the line.

Another command used to replace a single character (nothing else!) is **x**. First choose which character needs to be replaced and put the cursor on this character. Next press **x** followed by a new character. The new character will replace the old one. This command will leave the editor in *COMMAND* and not *INSERT* mode!

Cut and Paste

If you want to delete a single character while in command mode you would use **x**. Use **dd** to delete the current line. One can then paste the deleted item with the command **p**.

Remark: Nearly all **vi** commands can be repeated by specifying a number in front of the command. You can also apply the command to a text item (such as word., sentence, paragraph ...) by placing the entity after the command.

Examples:

dw Delete a word:

d\$ Delete text from here to the end of the current line

d} Delete text from here to the end of the current paragraph

One can simultaneously delete an item and switch to insert mode with the **c** command. As usual you can use this command together with a text item such as **w** or **{**.

Copying and Pasting

The copy action in **vi** is the command **y** (for yank, the letter **c** was already taken for *change*), and the paste action is still **p**.

If an entire line is yanked the pasted text will be inserted on the next line below the cursor.

The text selection is made with the familiar text items **w**, **l**, **}**, **\$** etc ... There are a few exceptions such as the last example.

Examples:

y\$ Copy the text from here to the end of the current line

yy Copy the entire current line

3yy Copy 3 lines

▮ The latest deleted item is always buffered and can be pasted with the **p** command. This is equivalent to a cut-and-paste operation.

Search and Replace

Since searching involves pattern matching we find ourselves once again dealing with regular expressions (regex). Like many UNIX text manipulation tools such as **grep** or **sed**, **vi** recognises regular expressions too.

To perform a search one must be in *COMMAND* mode. The **/** (forward slash) command searches forward and the **?** command searches backwards.

One can also perform search and replace operations. The syntax is similar to **sed**.

Example:

/\<comp Downward search for words beginning with 'comp' in all the text
?^z Upward search for lines starting with the letter z
:% s/VAR/var Search in the whole text for the keyword 'VAR' and replace it by 'var'

Undo and Redo

At this stage it is worth mentioning that one can always undo changes! This must be done in **COMMAND** mode with the **u** command (works as long as one hasn't yet saved the file). The redo command is **^R**.

Running a Shell Command

While in **LASTLINE** mode everything following an exclamation mark **!** is interpreted as a shell command.

For example while editing `lilo.conf` or `grub.conf` you may need to find out the name of the root device. This can be done with:

```
!:df /
```

Save and Quit

The command for saving is **:w**. By default the complete document is saved. In some cases **vi** will refuse to save changes made to a document because of insufficient rights. In such cases one can attempt to force a write with **:w!**

One can also specify an alternative name for the file. Portions of the text can be saved to another file while other files can be read and pasted in the current document. Here are the examples which illustrate this.

Examples:

:w newfile Save the current document as 'newfile'
:w 15,24 extract Save lines 15 to 24 in a file called 'extract'
:r extract Read from file 'extract'. The text will be pasted at the cursor

Warning: In the *column mode* context we have the following

. is the current line
\$ is the end of the document

The following are different ways available to quit **vi**:

:wq save and quit
:q! quit but do not save changes
:x exit and save when changes exist
:quit same as **:q**
:exit Or **:e** same as **:x**
ZZ same as **:x**

Used files, terms and utilities:

- vi
- /, ?
- h,j,k,l
- i, o, a
- c, d, p, y, dd, yy
- ZZ, :w!, :q!, :e!

Topic 104: Devices, Linux Filesystems, Filesystem Hierarchy Standard

104.1 Create Partitions and Filesystems

Candidates should be able to configure disk partitions and then create filesystems on media such as hard disks. This includes the handling of swap partitions.

Key Knowledge Areas

- Use various mkfs commands to set up partitions and create various filesystems such as:
 - *ext2/ext3/ext4*
 - *xfs*,
 - *reiserfs v3* and
 - *vfat*.

The following is a partial list of the used files, terms and utilities:

- fdisk
- mkfs
- mkswap

Linux File Systems

In order to persist data on a disk you need to create a file system on it first. At installation time you will be asked which type of file system should be used to format your configured block device before installation can occur.

Linux supports many filesystems the most popular being the extended (*ext2,ext3,ext4*) file system family. The *ext3* filesystem has been the default Linux filesystem over the last several years but has recently begun to be replaced by *ext4*, the latest iteration of the extended filesystem.

Ext4 is backwardly compatible with *ext3* and *ext3* and *ext4* are both backwardly compatible with *ext2*. *Ext4* is considered an interim solution while a more modern filesystem, called *Btrfs*, is developed. The features required from a modern filesystem include, pooling, snapshots, checksums and integral multi-device spanning, all of which *btrfs* will deliver in due course.

Ext2 File System

The Second Extended File System (*ext2fs* or *ext2*), sometimes referred to as the Linux native filesystem, was developed in 1993 and became the dominant filesystem used on Linux. Due to its widespread use over several years, *ext2* is regarded as a well tested and reliable filesystem. It is also consequently the best supported filesystem on Linux with a range of management tools and utilities. The utilities are part of the *e2fsprogs* package available on your Linux distribution and are often installed by default. The utilities and tools included in the *e2fsprogs* package include:

- **e2fsck** a fsck program that checks for and corrects inconsistencies
- **mke2fs** used for creating *ext2* file systems
- **tune2fs** used to modify file system parameters
- **dumpe2fs** which prints superblock and block group information.
- **debugfs** used to manually view or modify internal structures of the file system

Ext3 File System

The ext3 filesystem is an ext2 filesystem with journaling capabilities. [Journaling](#) improves reliability and eliminates the need to check the file system after an unclean shutdown, which results in faster start ups. One of the advantages of having ext3 being backwardly compatible with ext2 is that it allow well-tested and mature file system maintenance utilities, such as fsck and tune2fs for monitoring, checking and repairing ext2 file systems to also be used with ext3 without major changes.

Ext4 File System

The ext4 filesystem is an extension to ext3, meant to extend storage limits and add other performance improvements. Features include large volume and file size support, backward compatibility with ext3 and ext2 which allows use of maintenance utilities available, use of checksums in the journal to improve reliability, faster file system checking and snapshot support.

XFS File System

XFS is a high-performance journaling file system created by Silicon Graphics, originally for their IRIX operating system. The code was donated by Silicon Graphics and ported to the Linux kernel. XFS has a good reputation for speed and robustness and is particularly good at handling large files. XFS is much less popular than ext and consequently there are fewer tools and utilities to monitor and maintain XFS filesystems. The XFS tools are found as part of the `xfsprogs` package on most Linux distributions and include:

- **xfs_fsr** - Used to defragment mounted XFS file systems. When invoked with no arguments, xfs_fsr defragments all regular files in all mounted XFS file systems. This utility also allows users to suspend a defragmentation at a specified time and resume from where it left off later.
- **xfs_bmap** - Prints the map of disk blocks used by files in an XFS filesystem. This map list each extent used by a specified file, as well as regions in the file with no corresponding blocks (i.e. holes).
- **xfs_info** - Prints XFS file system information.
- **xfs_admin** - Changes the parameters of an XFS file system. The xfs_admin utility can only modify parameters of unmounted devices/file systems.
- **xfs_copy** Copies the contents of an entire XFS file system to one or more targets in parallel.
- **xfs_metadump** - Copies XFS file system metadata to a file. The xfs_metadump utility should only be used to copy unmounted, read-only, or frozen/suspended file systems; otherwise, generated dumps could be corrupted or inconsistent.

Reiserfs File System

ReiserFS is another general-purpose, journalled file system designed by Hans Reiser. It is particularly efficient at handling a large number of small files. It was one of the first journaling file systems for Linux and is often used in situations where the drive will store a large number of small files. Other filesystems that are not as efficient at handling small files, end up wasting a lot of space as their minimum block allocation often exceeds the size of the file. Reiserfs, like the XFS system, also has a fewer number of tools and utilities than the extended file system. Utilities for reiserfs are found in the `reiserfsprogs` package and include `reiserfsck`.

File System Formatting

In order to create a filesystem you need to format the partition. To create a file systems while running a Linux system you need to install the associated formatting tools which should be

available as a package. The formatting tools follow the naming convention of the filesystem type preceded with the **mkfs**, for make filesystem.

The formatting tool for ext2 for example is `mke2fs`. Similarly the formatting tool for the **xfs** file system is **mkfs.xfs** while for reiserfs it is **mkfs.reiserfs**. The `mkfs` command is a wrapper around each filesystem specific tool and acts as a front for all the different file system types. The syntax for using the **mkfs** command is:

```
mkfs -t <fstype> <DEVICE>
```

The ext3 filesystem is created by specifying an **ext2** file system and passing in the `-j` parameter to enable journaling support.

Example 1: Making a xfs filesystem



```
# mkfs -t xfs /dev/hda12
```

Example 2: Making a ext2 filesystem



```
# mke2fs /dev/hda11 [or mkfs -t ext2 /dev/hda11]
```

Example 3: Making a ext3 filesystem



```
# mke2fs -j /dev/hda11 [or mkfs -t ext2 -j /dev/hda11]
```

Example 4: Making a ext4 filesystem



```
# mkfs.ext4 /dev/hda1 [or mkfs -t ext4 /dev/hda11]
```

Formatting Swap Space

Swap space is made with the `mkswap` command. Swap space does not have a real filesystem as the kernel does raw read and writes to swap space to enhance the speed with which it can access cached memory pages. To create swap space you would run



```
# mkswap /dev/sda2
```

To activate the swap space you would run the command `swapon`; for example:



```
# swapon /dev/sda2
```

Used files, terms and utilities:

- fdisk (covered in section 102.1)
- mkfs
- mkswap

104.2 Maintain the integrity of filesystems

Candidates should be able to maintain a standard filesystem, as well as the extra data associated with a journalling filesystem.

Key Knowledge Areas

- Verify the integrity of filesystems.
- Monitor free space and inodes.
- Repair simple filesystem problems.

Once a filesystem has been created on your block device you will want to know how to monitor the filesystem and check it for errors, recovering from errors where possible. Fortunately the filesystem provide several command and utilities to aid in this process.

Monitoring Disk Usage

The **df** (disk free) and disk usage (**du**) commands can be used to report on the amount of free disk space and query how much space directories and files are using. **df** works on a device level, as opposed to a directory level.

The **df** tool shows used and available disk space. By default this is given in blocks of 1K.

```
$ df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/hda9       289M  254M   20M  93% /
/dev/hda2       23M   7.5M   14M  35% /boot
none           62M    0    61M   0% /dev/shm
/dev/hda5       1.4G  181M  1.1G  13% /share
/dev/hda7       787M   79M  669M  11% /tmp
/dev/hda3       4.3G  3.4G  813M  81% /usr
/dev/hda6       787M  121M  627M  17% /var
//192.168.123.2/share 12G  8.8G  3.7G  71% /mnt/smb
```

The **du** command will display disk usage. This is done on a per directory basis. **du** cannot display available space since this information is only available at a device level.

The following command will list the current usage of the **/etc** directory in human readable units (using the **-h** switch) and will only print the grand total (using the **-s** switch):



```
# du -sh /etc
62M    /etc/
```

File System Checking, Repair and Maintenance

As with filesystem monitoring there are numerous tools for maintaining a Linux filesystem which are provided by the creators of the respective filesystem.

If the file system is damaged or corrupt, then the **fsck** utility should be run against the partition (the

minimum requirement is that the file system be unmounted or mounted read-only).

fsck acts as a front that automatically detects the file system type of a partition. Then as with **mkfs**, the tools **fsck.ext2**, **fsck.ext3**, **fsck.ext4** or **fsck.xfs** will be called accordingly to carry out the system check and, if necessary, repair. Since ext3 is the main filesystem type for Linux there is a **e2fsck** command that only handles this filesystem type.

You invoke the filesystem check as follows explicitly specify a file system type with the following syntax:

```
fsck -t <fstype> <device>
```

Example: Checking a *reiserfs* filesystem on the `/dev/sdb10` device:



```
# fsck -t reiserfs /dev/sdb10
# fsck.reiserfs /dev/sdb10
```

Ext File System Maintenance utilities

As the extended filesystem is the most widely used and deployed filesystem on Linux, the tools for ext filesystem support are more numerous and comprehensive than for other filesystems.

Ext File System Debug Commands

The **debugfs** and **dumpe2fs** are seldom used but can be useful in providing low level information about an ext2, ext3 or ext4 filesystem.

```
debugfs [-b blocksize ] [-s superblock ] [-f cmd_file ] [-R request ] [-V] [[-w] [-c] [-i] [device ]]
```

The **debugfs** program is an interactive file system debugger. It can be used to examine and change the state of an ext2/3/4 file system.

Once in the **debugfs** shell, internal commands can then be used to change directory, examine inode data, remove files, create links, dump the ext3 journal logs etc. While this is a very powerful command, it should be used with caution, generally only after the **fsck** command has failed to make any headway.

```
dumpe2fs [-bfhixV] [-ob superblock ] [-oB blocksize ] device
```

dumpe2fs prints the super block and block group information for the filesystem present on *device*.

```
dumpe2fs /dev/hda1
dumpe2fs 1.35 (28-Feb-2004)
Filesystem volume name: /boot1
Last mounted on: <not available>
Filesystem UUID: d741042c-3eaf-49ee-94c1-7dd8c5459764
Filesystem magic number: 0xEF53
Filesystem revision #: 1 (dynamic)
Filesystem features: has_journal ext_attr resize_inode dir_index filetype
needs_recovery sparse_super
Default mount options: (none)
Filesystem state: clean
```

```

Errors behavior:          Continue
Filesystem OS type:      Linux
Inode count:             25584
Block count:             102280
Reserved block count:    5114
Free blocks:             80564
Free inodes:             25537
First block:             1
Block size:              1024
Fragment size:          1024
Reserved GDT blocks:     256
Blocks per group:        8192
Fragments per group:    8192
Inodes per group:        1968
Inode blocks per group:  246
Filesystem created:      Sat May  7 10:40:51 2005
Last mount time:         Sun May 29 04:08:01 2005
Last write time:         Sun May 29 04:08:01 2005
Mount count:             10
Maximum mount count:     -1
Last checked:            Sat May  7 10:40:51 2005
Check interval:          0 (<none>)
Reserved blocks uid:     0 (user root)
Reserved blocks gid:     0 (group root)
First inode:             11
Inode size:              128
Journal inode:           8
Default directory hash:  tea
Directory Hash Seed:     50108791-6a0a-41ff-9608-0485c993eaf9
Journal backup:          inode blocks

Group 0: (Blocks 1-8192)
  Primary superblock at 1, Group descriptors at 2-2
  Block bitmap at 259 (+258), Inode bitmap at 260 (+259)
  Inode table at 261-506 (+260)
  0 free blocks, 1942 free inodes, 2 directories
  Free blocks:
  Free inodes: 27-1968

[....]

```

tune2fs

tune2fs allows you to adjust various filesystem parameters on Linux extended filesystems. The following is a list of the most common parameters used to adjust extended filesystem settings:

- c sets the number of times a filesystem will be mounted before a filesystem check is forced. This is usually at next boot but can be run when a filesystem is manually unmounted.
- C sets the number of times the filesystem was mounted since it was last checked.
- L sets the volume label, this used to be used to uniquely identify hard disk partitions but is being replaced with UUIDs.
- i sets the maximum time between filesystem checks. A filesystem check will be forced when either the time expires or the the maximum number of mounts has been exceeded, which ever comes first.
- j adds journaling to an ext2 filesyseem making it an ext3 filesystem.

Running `tune2fs -l` will print out the current settings for a filesystem.

```
tune2fs 1.41.11 (14-Mar-2010)
Filesystem volume name: <none>
Last mounted on: /
Filesystem UUID: 6f4746c6-777e-4937-92ee-de98cf8f5aa4
Filesystem magic number: 0xEF53
Filesystem revision #: 1 (dynamic)
Filesystem features: has_journal ext_attr resize_inode dir_index
filetype needs_recovery extent flex_bg sparse_super large_file huge_file
uninit_bg dir_nlink extra_isize
Filesystem flags: signed_directory_hash
Default mount options: (none)
Filesystem state: clean
Errors behavior: Continue
Filesystem OS type: Linux
Inode count: 26558464
Block count: 106205707
Reserved block count: 5310285
Free blocks: 62912050
Free inodes: 25721258
First block: 0
Block size: 4096
Fragment size: 4096
Reserved GDT blocks: 998
Blocks per group: 32768
Fragments per group: 32768
Inodes per group: 8192
Inode blocks per group: 512
RAID stride: 32747
Flex block group size: 16
Filesystem created: Sat Feb 13 00:48:21 2010
Last mount time: Tue Sep 28 07:27:39 2010
Last write time: Tue Sep 21 09:14:03 2010
Mount count: 22
Maximum mount count: 25
Last checked: Tue Sep 21 09:14:03 2010
Check interval: 15552000 (6 months)
Next check after: Sun Mar 20 09:14:03 2011
Lifetime writes: 709 GB
Reserved blocks uid: 0 (user root)
Reserved blocks gid: 0 (group root)
First inode: 11
Inode size: 256
Required extra isize: 28
Desired extra isize: 28
Journal inode: 8
First orphan inode: 404853
Default directory hash: half_md4
Directory Hash Seed: 982f8e6c-db49-49b0-8f65-bce3725b5196
Journal backup: inode blocks
```

XFS File System Maintenance Utilities

The tools that come with XFS for filesystem integrity checking are `xfs_info` and `xfs_metadump`.

`xfs_metdump` is a filesystem debugging utility, that dumps xfs filesystem meta-data to a file. The file

can then be used to debug the files or as a backup. Later the meta-data can be restored with the `xfstool` utility.

Used files, terms and utilities:

- `du`
- `df`
- `fsck`
- `e2fsck`
- `mke2fs`
- `debugfs`
- `dumpe2fs`
- `tune2fs`
- `xfstool` (such as `xfstool`, `xfsmetadump` and `xfstool`)

104.3 Control mounting and unmounting of filesystems

Candidates should be able to configure the mounting of a filesystem

Key Knowledge Areas

- Manually mount and unmount filesystems.
- Configure filesystem mounting on bootup.
- Configure user mountable removable filesystems.

Mounting Filesystems at Bootup

At boot time the */etc/fstab* file assigns mount points for block devices.

The /etc/fstab format

device	mount-point	fstype	options	dump-number	fsck-number
--------	-------------	--------	---------	-------------	-------------

Sample /etc/fstab

LABEL=/	/	ext2	defaults	1	1
LABEL=/boot	/boot	ext2	defaults	1	2
LABEL=/home	/home	ext3	defaults	1	2
/dev/fd0	/mnt/floppy	auto	noauto,owner		0 0
LABEL=/usr	/usr	ext2	defaults	1	2
LABEL=/var	/var	ext3	defaults	1	2
none	/proc	proc	defaults	0	0
none	/dev/shm	tmpfs	defaults	0	0
none	/dev/pts	devpts	gid=5,mode=620	0	0
/dev/hdc9	swap,pri=-1	swap	defaults	0	0
/dev/cdrom	/mnt/cdrom	iso9660	noauto,owner,kudzu,ro	0	0

The options that are available for use with *fstab* file:

•**atime / noatime / relatime** The Unix stat structure records when files are last accessed (*atime*), modified (*mtime*), and created (*ctime*). One result is that *atime* is written every time a file is read, which has been heavily criticized for causing performance degradation and increased wear. However, *atime* is used by some applications and desired by some users, and thus is configurable as *atime* (update on access), *noatime* (do not update), or (in Linux) *relatime* (update *atime* if older than *mtime*). Through Linux 2.6.29, *atime* was the default; as of 2.6.30 (9 June 2009), *relatime* is the default,

•**auto / noauto** - With the *auto* option, the device will be mounted automatically at bootup or when the *mount -a* command is issued. *auto* is the default option. If you don't want the device to be mounted automatically, use the *noauto* option in */etc/fstab*. With *noauto*, the device can be only mounted explicitly.

•**dev / nodev** - Interpret/do not interpret block special devices on the filesystem.

- **exec / noexec** - `exec` lets you execute binaries that are on that partition, whereas `noexec` doesn't let you do that. `noexec` might be useful for a partition that contains no binaries, like `/var`, or contains binaries you don't want to execute on your system, or that can't even be executed on your system. Last might be the case of a Windows partition.

- **ro** - Mount read-only.

- **rw** - Mount the filesystem read-write. Again, using this option might alleviate confusion on the part of new Linux users who are frustrated because they can't write to their floppies, Windows partitions, or other media,

- **sync / async** -How the input and output to the filesystem should be done. `sync` means it's done synchronously. If you look at the example `fstab`, you'll notice that this is the option used with the floppy. In plain English, this means that when you, for example, copy a file to the floppy, the changes are physically written to the floppy at the same time you issue the copy command.

- **suid / nosuid** - Permit/Block the operation of `suid`, and `sgid` bits.

- **user / users / nouser** - `user` permits any user to mount the filesystem. This automatically implies `noexec`, `nosuid`, `nodev` unless overridden. If `nouser` is specified, only root can mount the filesystem. If `users` is specified, every user in group `users` will be able to unmount the volume.

- **owner** - Permit the owner of device to mount,

- **defaults** - Use default settings. Default settings are defined per file system at the file system level. For ext3 file systems these can be set with the `tune2fs` command. The normal default for Ext3 file systems is equivalent to `rw,suid,dev,exec,auto,nouser,async(no acl support)`. Modern Red Hat based systems set `acl` support as default on the root file system but not on user created Ext3 file systems. Some file systems such as XFS enable `acls` by default. Default file system mount attributes can be over ridden in `/etc/fstab`.

Manually Mounting and Unmounting Filesystems

The `mount` command is used to make a particular device available on a specific directory (mount point). The syntax is:

```
mount -t [FSTYPE] -o [OPTIONS] DEVICE DIRECTORY
```

For example we can mount a CDROM on the mount point `/media/cdrom` with:



```
# mount -t iso9660 /dev/cdrom /media/cdrom
```

`Mount` maintains the list of mounted filesystems in `/etc/mtab`. Typing `mount` with no options will show all filesystems currently mounted. The output is similar to `/etc/mtab`. The kernel also keeps track of mounted filesystems in `/proc/mounts`.

User Mountable Configurable Systems

On a running system the `/etc/fstab` file also acts as a *shortcut* for assigning a resource to a specific directory. For example:



```
# mount /dev/cdrom
```

The **mount** utility reads **fstab** and deduces where to mount the resource. Notice that some of the devices are accessed using a label. Labels are assigned to devices with the **tune2fs** tool:



```
#tune2fs -L /usr/local /dev/hdb12
```

Mount can take most of the **fstab** options that have been listed above. This allows a user to mount a filesystem read-only or with **noexec** for example. Beside mounting filesystems that have been defined in the **/etc/fstab** file, **mount** can also be used to mount new devices not defined in **fstab**. If an external SATA drive is plugged into the machine for example, and given the device node of **/dev/sdc**, its first partition can be mounted as



```
# mount /dev/sdc1 /mnt
```

This will make the 1st partition of the device available under the **/mnt** directory.

NOTICE

Remember that **mount -a** will mount all filesystems in **/etc/fstab** that have not been mounted and do not have the option **noauto**

The **umount** command will unmount a device. Notice that the command is misspelled! The syntax is:

umount DEVICE or MOUNT-POINT

For example the following commands will both unmount the CDROM device:



```
# umount /dev/cdrom
```

or



```
# umount /media/cdrom
```

Conventionally, mount points under **/media** are used for removable devices such as CDs or USB storage drives.

Used files, terms and utilities:

- /etc/fstab
- /media
- mount
- umount

104.4 Manage Disk Quotas

Candidates should be able to manage disk quotas for users

Key Knowledge Areas

- Set up a disk quota for a filesystem.
- Edit, check and generate user quota reports.

Quota Setup

Disk quotas allow system administrators to allocate the maximum amount of disk space a user or group's files and data may take up on a machine. This functionality can be crucial on a Linux machine that is used as a file server to prevent the hard disks from filling up and causing system instability.

The quota tools allow administrators to set up quotas without having to reboot. Here are the steps.

Edit `/etc/fstab` and add `usrquota` to the options

Remount the partition:

```
# mount -o remount <device>
```

Start the quota stats:

```
# quotacheck -ca
```

The preliminary `aquota.user` database file is generated at the top of the directory.

Editing and Quota Reporting

Edit quotas for each user:

```
# edquota -u <user>
```

Here a soft/hard limit must be set for both the number of blocks and inodes available for each user.

The system will allow the user to exceed the soft limit during a certain `grace` period. After the grace period has expired the soft limit will be enforced as a hard limit.

Start enforcing quotas:

```
# quotaon -a
```

Users can query the quota status with `quota`. The system administrator can generate reports with

repquota or quotastats.

Used files, terms and utilities:

- quota
- edquota
- repquota
- quotaon

104.5 Manage File Permissions and Ownership

Candidates should be able to control file access through the proper use of permissions and ownerships.

Key Knowledge Areas

- Manage access permissions on regular and special files as well as directories.
- Use access modes such as `suid`, `sgid` and the sticky bit to maintain security.
- Know how to change the file creation mask.
- Use the group field to grant file access to group members.

File and Directory Permissions

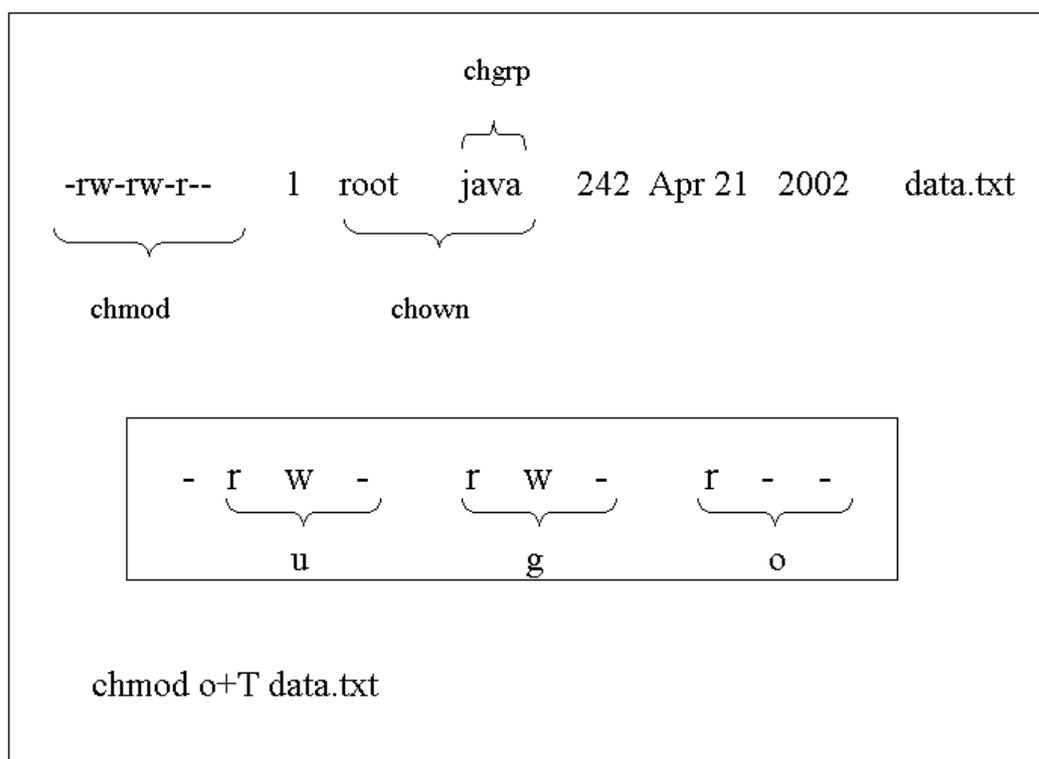
Access to directories and files on Linux is controlled by a simple file permissions systems. Every file/directory has permissions for the file owner, the group to which the file belongs and other, that is users who are not the owner and do not belong to the group to which the file belongs. The permissions are known as the access mode of file/directory and can be viewed by running the `ls -l` command.

File access modes are displayed symbolically as group of 3 letters or numerically as a set of 3 octal digits, but represent a 9 bit number, with each bit representing an access right.

<code>drwxr-xr-x</code>	3	root	root	4.0K	2009-10-27	20:03	hal
<code>-rw-r--r--</code>	1	root	root	4.7K	2009-10-06	22:45	hdparm.conf
<code>-rw-r--r--</code>	1	root	root	92	2009-04-27	11:56	host.conf
<code>-rw-r--r--</code>	1	root	root	4	2010-02-13	01:03	hostname
<code>-rw-r--r--</code>	1	root	root	292	2010-06-24	11:57	hosts
<code>-rw-r--r--</code>	1	root	root	579	2009-10-27	20:12	hosts.allow

The extract above is from running the `ls -l` command on the `/etc` directory. When a file is created it is owned by the user who created the file and assigned to the default group of the owner.

Symbolic and Octal Notation



Permissions can be read=r, write=w and execute=x. The octal values of these permissions are listed in the next table.

Octal and symbolic permissions.

Symbolic	Octal	Binary
read	4	'100'
write	2	'010'
execute	1	'001'

Permissions apply to the user, the group and to others. An item has a set of 3 grouped permissions for each of these categories.

How to read a 755 or -rwxr-xr-x permission

user	group	other
rwx	r-x	r-x
4+2+1=7	4+1=5	4+1=5

The Standard Permissions & UMASK

UNIX system create files and directories with standard permissions as follows:

Standard permission for:

Files	666	-rw-rw-rw-
Directories	777	-rwxrwxrwx

Every user has a defined **umask** that alters the standard permissions. The **umask** applies only at the point at which the file is created. The **umask** has an octal value and is subtracted(*) from the octal *standard permissions* to give the file's permission (this permission doesn't have a name and could be called the file's *effective permission*).

(*) While subtraction works in most cases, it should be noted that technically the standard permissions and the umask are combined as follows:

Final Permissions = Standard Permissions (logical AND) (NOT) umask

On systems where users belong to separate groups, the umask can have a value of 002. For systems which place all users in the *users* group, the umask is likely to be 022 so that files do not have group write access by default.

Changing permissions and owners

From the previous figure we see that permissions can be acted upon with **chmod**. There are 3 categories of ownership for each file and directory:

- u**: user
- g**: group
- o**: other

Example:

```
-rw-rw-r-- 1 jade sales 24880 Oct 25 17:28 libcgic.a
```

Changing Permissions with **chmod**:



```
#chmod g=r,o-r libcgic.a
#chmod g+w libcgic.a
```

Changing user and group with **chown** and **chgrp** :



```
#chown root libcgic.a
#chgrp apache libcgic.a
```

NOTE:

A useful option for **chmod**, **chown** and **chgrp** is **-R** which recursively changes ownership and permissions through all files and directories indicated.

Special Permissions

SUID Permissions

An executable can be assigned a special permission which will always make it run as the owner of this file. This permission is called **SUID** meaning 'set user ID'. It has a symbolic value **s** or a numerical value **4000**.

Administrative tools may have the SUID bit set in order to allow non-root users to change system files.

For example the **passwd** command can be run by any user and will interactively change his or her current password. This password will be saved to **/etc/shadow**. However this file belongs to user root with typical permissions of 600.

This problem has been solved by setting the SUID bit on **passwd** hence forcing it to run as user root with the correct permissions to modify **/etc/shadow**.

The SUID on **passwd**



```
# ls -l $(which passwd)
-r-s--x--x      1 root          root          18992 Jun   6   2003
/usr/bin/passwd
```

NOTE:

The SUID bit is shown in symbolic form in the command above. It is possible to get more information about a file using **stat** as well as seeing the octal representation of the permissions as follows:



```
# stat /usr/bin/passwd

File: '/usr/bin/passwd'
Size: 18992    Blocks: 40    IO Block: 4096   regular file
Device: 305h/773d    Inode: 356680    Links: 1
Access: (4511/-r-s--x--x)  Uid: ( 0/ root)  Gid: ( 0/ root)
```

WARNING! WARNING! WARNING!

The SUID permission is often associated with security issues. Here is an example that illustrates this.

1. A user would like to read user root's mail. For this he changes the MAIL environment variable as follows:



```
# export MAIL=/var/spool/mail/root
```

2. The user then uses the command **mail**, hoping to see something!



```
# mail

/var/spool/mail/root: Permission denied
```

So far it doesn't work. This would be too easy!

But if root can be convinced to set the SUID bit on **mail** the previous commands would allow any user to read anybody's mail (including root).

The next examples are dangerous. Why?



```
#chmod 4755 /bin/cat  
#chmod u+s /bin/grep
```

SGID permissions

The SGID is a permission similar to SUID that is set for group members. The symbolic value is **s** and the octal value of **2000**.

Setting **SGID** on a directory changes the group ownership used for files subsequently created in that directory to the directory's group ownership. No need to use `newgrp` to change the effective group of the process prior to file creation.

Examples:



```
#chmod 2755 /home/data  
#chmod g+s /bin/wc
```

The sticky bit

The sticky bit permission with value **1000** has the following effect:

- Applied to a directory it prevents users from deleting files unless they are the owner (ideal for directories shared by a group, or for **/tmp**)
- Applied to a file this used to cause the file or executable to be loaded into memory and caused later access or execution to be faster. The symbolic value for an executable file is **t**. It was supported in some versions of Unix but is not used in Linux.

Examples:



```
#chmod 1666 /data/store.txt  
#chmod o+t /home/students
```

Used files, terms and utilities:

- chmod
- umask
- chown
- chgrp

104.6 Create and change hard and symbolic links

Candidates should be able to create and manage hard and symbolic links to a file

Key Knowledge Areas

- Create links.
- Identify hard and/or softlinks.
- Copying versus linking files.
- Use links to support system administration tasks.

When we copy a file with the cp command, a duplicate of that file is created, sometimes however we want to provide an link to an existing file but want that path to point to the exact same file as the original. In this case we will use symbolic links.

Symbolic Links

A soft link to a file or a directory is a special file type that simply contains the name of the file that it “points to”.



```
# ln -s mytext.txt myext.sym
```

Soft links can be created across filesystems. By running **ls -l** we can identify whether a directory entry is a soft link or just an ordinary file from the output. A symbolic link is shown as follows when we run the **ls -l** command

mytext.txt -> mytext.sym. Notice that the reference count is **1** for both files.

```
-rw----- 1 root root 223 Sep 29 09:10 mytext.txt
lrwxrwxrwx 1 root root 9 Sep 29 09:10 mytext.sym -> mytext.txt
```

To find all symbolic links to a file you can use the find command for example

```
find / -lname mytext.txt
```

will find all symbolic links to the file mytext.txt.

Hard Links

A hard link is an additional name for the same inode and as such the reference count of the file increases by one for every new hard link.



```
# ln mytextfile.txt mytextfile.link
```

In the listing notice that the reference count is **2** and that both files have the same size. In fact they are identical.

```
-rw----- 2 mark mark 223 Sep 26 09:06 mytextfile.txt
```

```
-rw----- 2 mark mark 223 Sep 26 09:06 mytextfile.link
```

Hard links can only be created within the same filesystem. Using ls, a hard link can be identified by the reference count shown in the output, as in the above example. Another way of finding files with hard links is to obtain the file's inode number and then run the find command with the inode number as a parameter. To find the inode of a file run the command:



```
# ls -li mytextfile.txt  
8652338 mytextfile.txt
```

This will output the inode number of the file, next run the find command as follows:



```
# find / -inum 8652338
```

Used files, terms and utilities:

- ln

104.7 Finding and Placing Files within the File Hierarchy Standard

Candidates should be thoroughly familiar with the Filesystem Hierarchy Standard (FHS), including typical file locations and directory classifications.

Key Knowledge Areas

- Understand the correct locations of files under the FHS.
- Find files and commands on a Linux system.
- Know the location and purpose of important file and directories as defined in the FHS.

The Linux File System

In general Linux filesystem layout is consistent across distributions, with minor variations. The consistency is largely due to the existence of the Filesystem Hierarchy Standard (FHS) project, which aims to provide a recommended standard layout for Linux and Unix like operating systems.

Below is a listing of the most important directories, and a brief explanation of their purpose, commonly found on Linux systems:

- **/bin** and **/sbin** Contain binaries needed to boot up the system and essential commands.
- **/dev** Location for device or special files
- **/etc** Host specific configuration files
- **/lib** Shared libraries for binaries in **/bin** and **/sbin**. Also contains kernel modules
- **/mnt/** or **/media** Mount point for external filesystems
- **/proc** Kernel information. Read-only except for **/proc/sys/**
- **/boot** Contains the Linux kernel, the system maps and the “second stage” bootloaders.
- **/home** The directories for users. Initially contains the contents from **/etc/skel/**
- **/root** The directory for user root
- **/tmp** Temporary files
- **/usr** User Specific Resource. Mainly static and shareable content
- **/usr/local** or **/opt (optional)** Add-on software applications. Can also contain shared libraries for add-on software.
- **/var/www, /var/ftp/** Location for HTML pages and anonymous FTP directories.
- **/var** Variable data, such as spools and logs. Contains both shareable (eg. **/var/spool/mail**) and non-shareable (eg. **/var/log/**) subdirectories.

Finding Files and Directories

We will describe the **find**, **which**, **whereis** and **locate** utilities. (

The **find** command was covered in topic 103.3)

locate

Syntax:

locate <STRING>

When using **locate** all files and directories that match the *expression* are listed.



```
# locate X11R
```

The search is much faster than **find**. In fact **locate** queries the `/var/lib/slocate/slocate.db` database. This database is kept up to date via a daily cron job which runs `updatedb`.

When running **updatedb** from the command line the `/etc/updatedb.conf` file is read to determine pruned file systems (e.g NFS) and directories (e.g `/tmp`)

which

Syntax:

which string

This tool will return the full path to the file called **string** by scanning the directories defined in the user's PATH variable only. As a result **which** is only used to find commands.

whereis

Syntax

whereis string

This tool will return the full path to source or binaries as well as documentation files matching **string** by scanning the PATH variable as well as a number of well known locations

Getting the most from ls

Most common options for ls

-I	show inode
-h	print human readable sizes
-n	list UIDs and GIDs
-p	append descriptor (<code>/=</code>) to list
-R	recursively display content of directories
-S	sort by file size
-t	sort by modification time (similar to <code>-c</code>)
-u	show last access time

Used files, terms and utilities:

- find
- locate
- updatedb
- whereis
- which
- type

Chapter II: LPIC 102

Topic 105: Shells, Scripting and Data Management

105.1 Customize and use the shell environment

Candidates should be able to customize existing scripts, or write simple new BASH scripts.

Key Knowledge Areas

- Set environment variables (e.g. PATH) at login or when spawning a new shell.
- Write BASH functions for frequently used sequences of commands.
- Maintain skeleton directories for new user accounts.
- Set command search path with the proper directory.

Introduction

The command line interface (CLI or terminal) may seem intimidating at first, but it's important to remember that the command line is really, truly your friend. An army of tools are at your disposal that can take what would be a tedious and lengthy job (like removing the last four characters from every line of a lengthy file) and turn it into a two minute job.

For every Linux distribution the command line prompt will look a little different. For example, on one system you might see your username, the '@' symbol, the machine name, your current directory and the prompt.

```
user@linux ~/ $
```

This is a very common prompt. You may also see your username, a space, the fully qualified domain name of the computer, the full path to your present working directory followed by the prompt

```
user linux.box.com /home/user $
```

The prompt varies from system to system based on a number of things. For example, it may be the default configuration set by the creators of your particular Linux distribution. It could also have been configured by the person who administers the computer or by yourself.

The way you configure the look of your command prompt depends on what shell you use and the shell is the piece that most people commonly refer to as "the command line" when, in reality, it is simply a piece of software that provides an interface to the services of a kernel. The distinction between a 'shell' and the 'command line' is simply that a shell refers to a specific piece of software (e.g BASH, tcsh, ksh, etc) that provides a command line interface. Most modern Linux systems use BASH (Bourne Again SHell) as their default shell.

A Little History

The commands used at the command line may seem a little cryptic due to their tendency to be very short. This is because the roots of the Linux command line are from systems where a single letter entry could take a significant amount of time to travel from a terminal, to a central server and back to the terminal where it was printed onto a roll of paper. In those old systems, the shorter the input was, the better as it meant less time waiting to issue your command and receive output. The best thing you can do to remember what commands stand for is to find out what word the command is an abbreviation for. This can go a long way to remembering the command later.

Summary of Common Commands

- `ls` - This command 'lists' the contents of your present working directory.
- `pwd` - Shows you what your present working directory is.
- `cd` - Lets you change directories.
- `rm` - removes one or more files.
- `rmdir` - Remove an empty directory.
- `mkdir` - Make a directory.
- `ps` - Provides a list of currently running processes.
- `cp` - Copy a file.
- `mv` - Move a file (this is also used to rename a file, "moving" it from one file name to another.)
- `grep` - The global regular expression print program lets you search through a file or output of another program.
- `find` - Find a file on the filesystem
- `man` - Displays the manual for most commands (including 'man').

TIP

For help about a command, use `man command` which will bring up the manual for it. Note that some commands are built into your shell and do not have a `man` page, use your interpreter internal command (should be `help <command>`).

Login vs. Non-login shell

Login shell: Shell started with `login`, `bash -l` or `su` command

Non-login shell: Shell started any other way

Reason for 2 types of shell

The login shell reads a series of configuration file as it is started.

The non-login shells inherit settings (environment variables) from the parent program which started it.

Variable inheritance

Variables declared inside a shell are inherited by child processes if the variable has been exported.

If a child process changes its own copy of an exported variable, the parent shell's copy is not changed. The changed value is exported to any sub-child processes.

All exported shell variables keep their export settings in the child process.

If a shell script is called from within a shell a new child non-login shell is started.

If a shell script is started with the `'.'` command within a shell, then the script is run within that current shell.

Example: `/home/joe/bin/myscript`

Warning: If the called script runs the command `exit`, the current shell will be terminated!

Interactive and non-interactive shells

Interactive shell: Provides a prompt where the user can type commands.

Non-Interactive shell: No shell prompt – started by calling a shell script or by the command:

```
sh -c 'command...'
```

Sequence of events when bash starts

Interactive-login bash

```
bash --login or su - username or from login
```

`/etc/profile` Executed first from interactive login shell. It contains system-wide environment settings.

`~/.bash_profile` Individual user's shell settings.

`~/.bash_login` Executed if `~/.bash_profile` doesn't exist.

`~/.profile` Executed if `~/.bash_login` or `~/.bash_profile` doesn't exist.

Interactive non-login bash

```
su username or bash -c command
```

`~/.bashrc` The only script executed when started. Inherits from parent bash environment.

Non-Interactive non-login bash (forked when scripts are run)

The above scripts are not executed but inherit environment from their parent.

`BASH_ENV` Reads file in the variable `BASH_ENV`.

`ENV` Reads file in the variable `ENV` if `BASH_ENV` doesn't exist.

Extra files

`/etc/inputrc` - System bash line editing (readline) configuration file

`~/.inputrc` - Individual bash line editing (readline) configuration file

`~/.bash_logout` - Executed (if exists) when a login shell exits.

Commands for shell/environment variables

Variablename=Value - Assigns a value to a set (existing) or non-set variable.

export Variablename or

declare -x Variablename - Sets the export tag ON for an existing shell var.

export **Variablename=value** or

declare -x Variablename=value - Assign a value to a set (existing) or non-set variable and sets its export tag ON, all in one command.

env - Displays all the environment variables (export tag ON)

export - Same as env command except the display format is different eg.
declare -x PAGER="less"

Aliases

Aliases are normally used to create command shortcuts (short names).

Aliases are NOT exportable: not passed-on to sub-shells or child process.

Aliases are not recognized in scripts.

An alias can call another alias within a command.

Example `alias li="ls -l"; alias al="li -a"` al calls the alias 'li'

Parameters added to alias will be added at the end of the real command.

The parameter variables (\$1, \$2, \$3 ...etc) cannot be used within aliases.

Aliases are often defined in a file run within a script like `~/ .bashrc` or `~/ .profile` with the dot '.' command.

Alias commands:

`alias` - Displays all the current shell aliases.

`alias AliasName="command(s)..."` - Sets a new alias value.

Example:

```
# alias cp="cp -i"
```

Replaces the original command `cp` with `cp -i` for interactive copying (asks before overwriting files).

unalias AliasName - Unsets (deletes) the alias.

Functions

They are normally used like fast local mini-scripts within a shell which need to be called more than once within the interactive shell or script.

Variables can be passed-on to functions and will be recognized as \$1 \$2 \$3 etc. In fact the following variables are local within a function:

\$1 - \$9 - Positional parameters

\$# - Number of positional parameters

\$* "\$1 \$2 \$3 ..."

\$_ "\$1" "\$2" "\$3" ...

The positional parameter \$0 and all other variables stay global within the shell unless the command `local Variablename` is given within the function. Within a function, the variable `FUNCNAME` is used instead of the \$0.

Global shell or exported variables can be changed within the function.

Functions do not return variables except for the return number, eg. Return 5. The return command will also terminate the function immediately. The return number can then be read as a normal exit value using \$?.

In scripts functions are normally included at the top so that they are read in first.

Environment functions can be put into a file and read in with the `.` command.

Functions may be recursive. No limit is imposed on the number of recursive calls.

Functions can be exported, using the command: `export -f FunctionName`

Function syntax

```
FunctionName ()
```

```
{  
command1 ;  
command2 ;  
}
```

Command Search Priority

When a command is run, bash tries to find the command in the following sequence:

- Aliases
- Functions
- Builtin commands
- searching the PATH

The first command found is the one which is run.

To force using a builtin command instead of an alias or a function (in the case the same command name exists as alias or function), use the command `builtin`.



```
# builtin cat /etc/fstab
```

set

Syntax: `set [--abefhkmnptuvxBCHP] [-o option] [arg ...]`

The set command is used to:

-Display all bash variables and their values as well as the functions.

Example:

set [bash operating attributes] (using options).

-Assign values to positional parameters:

Example:

```
# set aaa bbb ccc ($1 $2 $3)
```

The above assigns the value aaa to \$1, bbb to \$2 and ccc to \$3.

unset

Syntax: unset [-fv] [name ...]

For each name, remove the corresponding variable or function.

Each unset variable or function is removed from the environment passed to subsequent commands. If any of RANDOM, SECONDS, LINENO, HISTCMD, FUNCNAME, GROUPS, DIRSTACK are unset, they lose their special properties, even if they are subsequently reset. The exit status is true unless a name does not exist or is readonly.

If no options are supplied, or the -v option is given, the name refers to a shell variable. Read-only variables may not be unset.

```
# unset -v
# unset -f
```

The following examples delete the variable DISPLAY, and the function startx respectively.

```
# unset DISPLAY
# unset -f startx
```

The following is a partial list of the files, terms and utilities that were used.

- /etc/profile
- env
- export
- set
- unset
- ~/.bash_profile
- ~/.bash_login
- ~/.profile
- ~/.bashrc
- ~/.bash_logout
- function
- alias
- lists

105.2 Customize or Write Simple Scripts

Candidates should be able to customize existing scripts, or write simple new BASH scripts.

Key Knowledge Areas

- Use standard sh syntax (loops, tests).
- Use command substitution.
- Test return values for success or failure or other information provided by a command.
- Perform conditional mailing to the superuser.
- Correctly select the script interpreter through the shebang (#!) line.
- Manage the location, ownership, execution and suid-rights of scripts.

What is a shell script?

A shell script is a text file that tells the shell what to do.

It contains the name of the program that is used as the interpreter for the rest of the content of the script. The line starting with `#!/ProgramPath+Name` (normally the first line) designates the interpreter to be used:

```
#!/bin/bash      or      #!/bin/sh      or      #!/usr/bin/perl -w
```

In reality when the system is asked to start a script, the line starting with `#!` is read and the appropriate script interpreter is started which in turns reads the script and executes the commands included in it.

Conditions for running a script

The script file must be runnable by the user running it (`chmod`) The interpreter must be where the script says it is: the default is to call bash.

Language used in shell scripts

The script language depends on the script interpreter used. bash has its own syntax which can be used interactively or in a script.

Passing parameters to a script

Scripts can be given up to 9 positional parameters (for all interpreters) or up to 99 parameters with bash.

Inside the script each parameter will be identified as `$1` to `$9` or `${10}` to `${99}`

```
scriptname param1 param2 param3 param4..... param47.....$0 $1 $2 $3 $4
${47} .....
```

Some special parameters are automatically set by the Bourne shell, and usually cannot be directly set or modified.

Parameter `$n` can be modified by the `set` command inside the script.(where `n` is 1-99 for bash)

```
set aaa bbb ccc ... $1 $2 $3
```

Special Parameters

`$n` - Positional parameter `n` max. `n=9` (`$0` is the name of the shell script)

`${nn}` - Positional parameter `nn` (for `nn>9`)

- \$# - Number of positional parameters (not including the script program)
- \$_, \$* - All positional parameters
- "\$@" - Same as "\$1" "\$2" . . . "\$n"
- "\$*" - Same as "\$1c\$2c . . . \$n" c = content of \$IFS (default is space)
- \$? - Exit status of the last command
- \$\$ - Process ID of the current shell
- \$- - Current options in effect
- #! - Process ID of the last background command
- \$ - Name of the current shell (in this case 'bash')

The shift command

The shift command moves the assignment of the positional parameters to the left. If a script is called like this:

```
script1 aaa bbb ccc ddd
```

And the following commands are run inside the script



```
# echo $1 $2 $3
# shift
# echo $1 $2 $3
```

The result of the first echo command is:

```
aaa bbb ccc
```

The result of the second echo command is:

```
bbb ccc
```

The set and unset commands

The unset command is normally used to unset values of variables, and the set command to assign values to positional parameters from inside a script. Very useful if a script has been started without positional parameters and after verifying this the script assigns default values to them.

set aa bb cc dd \$1 \$2 \$3 \$4 - assigns aa to \$1, bb to \$2, cc to \$3 and dd to \$4

The set command is also useful for changing properties of bash's behaviour.

One important option of set is:



```
# set -o noclobber
```

The command causes the redirection symbol (>) to fail to overwrite the contents of an existing file.

Conditional statements

Below is a list of the most used conditional directives

The if conditional branching directive

`if` - allows certain commands to execute only if certain conditions are met.

Syntax: (see also the section 'CONDITIONAL EXPRESSIONS' later in this topic):

```
if <condition_is_true> ; then
run_these_commands
```

.....

```
elseif <condition_is_true> ; then
```

if first condition is not met and this one is met then:

```
run_these_commands
```

.....

```
else
```

if all conditions above are not met then:

```
run_these_commands_instead
```

.....

```
fi
```

end of if directive block

<condition_is_true> can be of the following types:

(1) Test the status of files or directories.

```
if test -e /etc/fstab ; then
```

or

```
if [ -e /etc/fstab ] ; then
```

(2) Command or script exit code.

```
if (ifconfig | grep 'ppp0') ; then
```

(3) The contents of a variable corresponding to a certain value:

```
if $1 ; then
```

- true if \$1 has a value in it

```
if [ "$net" = "eth0" ] ; then
```

- testing strings

```
if test ["$#" -eq 5] ; then
```

(integer testing)

The case conditional branching directive

`case` is normally used for conditionally branching to one of several choices depending on

the content of a variable.

Syntax:

```

case <Variable> in
<choice1>)
commands to run
;;
choice2)
commands to run
;;
choice3)
commands to run
;;
*)
commands to run if none of the above conditions are met.
;;
case

```

end of case directive block

Looping in scripts

Used whenever a sequence of commands must be executed more than once.

The while conditional loop directive

The while directive keeps looping and running the commands in its block for as long as its condition(s) (defined in the while statement) is/are met.

Syntax:

```

while <condition_is_true> ; do
run_these_commands
done
end of while directive block

```

Note: While is often used to ask the user for a keyboard entry of some sort and if the response is not adequate then the request is repeated until the proper information is entered. The while loop is then exited and the program resumes its execution.

The until conditional loop directive

The until loop works exactly the same way as the while loop except that the logic is the opposite:

The loop continues until condition(s) is/are met.

Syntax:

```

until <condition_is_true> ; do
run_these_commands
done
end of until directive block

```

The for loop directive

The for directive allows a sequence of commands to be executed as many times as there are items in a given list. Each time the loop runs through, the content of a specific variable becomes value of the current item in the given list.

Syntax:

```
for variable in list ; do
run_these_commands
done
end of for directive block
```

variable =the variable name which will have its content become the current item on each loop round in the given list. The list can also be a variable which contains a list of items.

```
for item in ~/file1 ~/file2 ~/file3 ; do
echo "----- Content of $item -----"
cat $item >> ~/allfiles
done
```

Shell functions

Shell functions are a series of commands stored in one place that can be used from several points in the script. Parameters can be passed to functions via positional parameters.

The positional parameters (\$1, \$2, \$3 ...), which will become local to the function. They use the same syntax as for a script except that the first (\$0) stays global.

The variable FUNCNAME is used similarly to, for the same purpose as, \$0.

Special variables like \$#, \$*, \$@, are also local within the function.

All other variables are global to the script and can be modified by the functions.

The command `return x` (x=return code) can be used as a function exit command and to assign a function return code.

Syntax:

```
FunctionName () {
command ;
command ;
}
or
function FunctionName () {
command ;
command ;
}
```

(See functions in the previous section Customize and use the shell environment for more details on shell Functions.)

Exit codes and the variable \$?

All programs, including scripts, return an exit code when their process ends. The exit code helps determine the success or failure of the program or the script. This exit code can be read via the special variable \$? and be used to make decisions further in the calling script.

Generally the exit code of '0' means success and any other code (1-255) means some sort of failure. It is also often referred as the error code.

The && and || conditional branching

The exit code can be used to execute another command (only one) depending upon its success or its failure. The double ampersand '&&' is used to designate the command to run if the exit code is success (0). The double pipe '||' designates the command to run if the exit code is not a success (1-255).

Example



```
# ifconfig ppp0 && echo "pppd running" || echo "pppd not running"
```

If the command `ifconfig ppp0` succeeds then the command `echo "pppd running"` will be executed (&&) otherwise the command `echo "pppd not running"` will be executed.

Mailing messages to root from a script

Sometimes it is useful to mail a message to root or to other users announcing some anomalies or success in the running of an automated script. The program normally used is 'mail'. See `man mail` for all the options it uses.

Syntax1:



```
# mail -s "subject" destination_mail_address "message.."
```

Syntax2:



```
# program | mail -s "subject" destination_mail_address
```

Syntax3:



```
# mail -s "subject" destination_mail_address <<EOM
message body.....
EOM
```

Example:



```
# df | mail -s "HD Space on $(date)" root
```

Mails the result of the command `df` to the local root user.

Location and security for bash scripts

Administration scripts are normally stored in the PATH which is either `/usr/local/bin` or `/root/bin`. The normal access rights are `755(rwx r-x r-x)` or for more protection by preventing any other user than root to run it: `700(rwx --- ---)`.

Although the SUID doesn't have any effect on scripts, very old versions of Linux may be affected by SUID being set.

Conditional Expressions

The `test` and `[...]` commands are used to evaluate conditional expressions with file attributes, strings, and integers. The basic format is: `test expression` or `[expression]`, where `expression` is the condition you are evaluating. There must be whitespace after the opening bracket, and before the closing bracket. Whitespace must also separate the expression arguments and operators. If the expression evaluates to true, then a zero exit status is returned, otherwise the expression evaluates to false and a non-zero exit status is returned.

Test File Operators

- a** <file> True if file exists.
- b** <file> True if file exists and is a block special file.
- c** <file> True if file exists and is a character special file.
- d** <file> True if file exists and is a directory.
- e** <file> True if file exists.
- f** <file> True if file exists and is a regular file.
- g** <file> True if file exists and is set-group-id.
- h** <file> True if file exists and is a symbolic link.
- k** <file> True if file exists and its ``sticky" bit is set.
- p** <file> True if file exists and is a named pipe (FIFO).
- r** <file> True if file exists and is readable.
- s** <file> True if file exists and has a size greater than zero.
- t** <fd> True if file descriptor `fd` is open and refers to a terminal.
- u** <file> True if file exists and its SUID bit is set.
- w** <file> True if file exists and is writable.
- x** <file> True if file exists and is executable.
- O** <file> True if file exists and is owned by the effective UID.
- G** <file> True if file exists and is owned by the effective GID.
- L** <file> True if file exists and is a symbolic link.
- S** <file> True if file exists and is a socket.
- N** <file> True if file exists and has been modified since it was last read.
- file1 -nt file2** True if file1 is newer (according to the modification date) than file2, or if file1 exists and file2 does not.

file1 -ot file2 True if file1 is older than file2, or if file2 exists and file1 does not.
file1 -ef file2 True if file1 and file2 refer to the same device and inode numbers.

Test String Operators

-n string True if length of string is not zero
-z string True if length of string is zero
string True if string is not set to null
string1 = string2 True if string1 is equal to string2
string1 = string2 True if string1 is equal to string2
string1 != string2 True if string1 is not equal to string2
string1 < string2 True if string1 sorts before string2 lexicographically in the current locale.
string1 > string2 True if string1 sorts after string2 lexicographically in the current locale.
string = pattern True if string matches pattern
string != pattern True if string does not match pattern

Test Integer Operators

exp1 -eq exp2 True if exp1 is equal to exp2 eg. ["\$#" -eq 4]
exp1 -ne exp2 True if exp1 is not equal to exp2 eg. test "\$#" -ne 3
exp1 -le exp2 True if exp1 is less than or equal to exp2
exp1 -lt exp2 True if exp1 is less than exp2
exp1 -ge exp2 True if exp1 is greater than or equal to exp2
exp1 -gt exp2 True if exp1 is greater than exp2

Other test Operators

! exp True if the given expression is false eg. [! -r /etc/motd]
exp1 -a exp2 True if both exp1 and exp2 evaluate to true (see example below)
exp1 -o exp2 True if either exp1 or exp2 evaluate to true
\(exp \) True if exp is true; used to group expressions

The \ used to escape parentheses. Use spaces before and after this character

["\$A" = "\$B" -a \("\$C" = "\$D" -a "\$E" = "\$F" \)]

The following is a partial list of files, terms and utilities that were used.

- for
- while
- test
- if
- read
- seq

105.3 SQL Data Management

Candidates should be able to query databases and manipulate data using basic SQL commands. This objective includes performing queries involving joining of 2 tables and/or subselects.

Key Knowledge Areas

- Use of basic SQL commands.
- Perform basic data manipulation.

The Structured Query Language (SQL) is a database management programming language. SQL is a tool for accessing databases, and more specifically, relational databases, and can be used with different database products. This chapter will prepare you to learn basic database management using this language.

Common SQL Implementations for Linux

Some of the more common choices in Linux include the following:

MySQL - This SQL Implementation is owned by Sun, and is released under the GPL. Most major Linux distributions will include MySQL in their package databases.

PostgreSQL - Released under the BSD license, PostgreSQL was evolved from Ingres software. (PostgreSQL= post-Ingres SQL). It's available as multiple packages in most Linux distributions.

SQLite – To implement SQL as a library, you need SQLite. SQLite is intended to provide users and programs a way to store data using a SQL interface within the program. SQLite3 can be used to manipulate SQLite databases for major Linux distros.

SQL Basics

SQL is used to access relational databases. Each database contains more or less tables which in turn contain more or less rows and columns. Hereby a single row is seen as a separate object with features represented by the tables' columns. To access a table's data you first have to connect to its database.

1. `mysql -u USERNAME -p PASSWORD`
2. `use DATABASE`

Basic SQL Commands

SELECT

A SELECT statement retrieves zero or more rows from one or more database tables or database views. In most applications, SELECT is the most commonly used Data Manipulation Language (DML) command. As SQL is a declarative programming language, SELECT queries specify a result set, but do not specify how to calculate it. The database translates the query into a "query plan" which may vary between executions, database versions and database software. This functionality is called the

"query optimizer" as it is responsible for finding the best possible execution plan for the query, within applicable constraints.

The SELECT statement has many optional clauses:

- WHERE specifies which rows to retrieve.
- GROUP BY groups rows sharing a property so that an aggregate function can be applied to each group.
- HAVING selects among the groups defined by the GROUP BY clause.
- ORDER BY specifies an order in which to return the rows.

Examples:

Given a table T, the *query* SELECT * FROM T will result in all the elements of all the rows of the table being shown.

With the same table, the query SELECT C1 FROM T will result in the elements from the column C1 of all the rows of the table being shown.

With the same table, the query SELECT * FROM T WHERE C1 = 1 will result in all the elements of all the rows where the value of column C1 is '1' being shown.

WHERE

A WHERE clause specifies that a SQL statement should only affect rows that meet specified criteria. The criteria are expressed in the form of predicates. WHERE clauses are not mandatory clauses of SQL statements, but should be used to limit the number of rows affected by a SQL DML statement or returned by a query.

The following query returns only those rows from table *mytable* where the value in column *mycol* is greater than 100.

```
SELECT *  
FROM mytable  
WHERE mycol > 100
```

DISTINCT

DISTINCT will eliminate all duplicate rows from the selection. DISTINCT ON *column* will eliminate all duplicates in the specified column; this is equivalent to using GROUP BY *column*. ALL will return all candidate rows, including duplicates.

```
SELECT DISTINCT jobTitle FROM employees;
```

+-----+

```
| jobTitle          |
+-----+
| President         |
| VP Sales          |
| VP Marketing      |
| Sales Manager     |
| (APAC)           |
+-----+
```

GROUP BY

The GROUP BY clause is used to project rows having common values into a smaller set of rows. GROUP BY is often used in conjunction with SQL aggregation functions or to eliminate duplicate rows from a result set. The WHERE clause is applied before the GROUP BY clause.

The example below demonstrates a query of multiple tables, grouping, and aggregation, by returning a list of books and the number of authors associated with each book.

```
SELECT Book.title,
       count(*) AS Authors
FROM   Book JOIN Book_author
       ON Book.isbn = Book_author.isbn
GROUP BY Book.title;
```

Example output might resemble the following:

Title	Authors
SQL Examples and Guide	4
The Joy of SQL	1
An Introduction to SQL	2
Pitfalls of SQL	1

HAVING

A HAVING clause in SQL specifies that an SQL SELECT statement should only return rows where aggregate values meet the specified conditions. It was added to the SQL language because the WHERE keyword could not be used with aggregate functions.

To return a list of department IDs whose total sales exceeded \$1000 on the date of January 1, 2000, along with the sum of their sales on that date:

```
SELECT DeptID, SUM(SaleAmount)
FROM   Sales
WHERE  SaleDate = '01-Jan-2000'
GROUP BY DeptID
```

```
HAVING SUM(SaleAmount) > 1000
```

ORDER BY

The ORDER BY clause allows a user to specify that he/she wishes the rows sorted according to the ASCending or DESCending mode operator.

```
SELECT * FROM Employees
```

```
ORDER BY LastName, FirstName
```

IN

IN will find any values existing in a set of candidates.

```
SELECT ename WHERE ename IN ('value1', 'value2', ...)
```

All rows match the predicate if their value is one of the candidate set of values. This is the same behavior as

```
SELECT ename WHERE ename='value1' OR ename='value2'
```

except that the latter could allow comparison of several columns, which each IN clause does not. For a larger number of candidates, IN is less verbose.

BETWEEN

BETWEEN will find any values within a range.

```
SELECT ename WHERE ename BETWEEN 'value1' AND 'value2'
```

All rows match the predicate if their value is between 'value1' and 'value2', inclusive.

LIKE

LIKE will find a string fitting a certain description.

- Ending Wildcard

- Find any string that begins with the letter 'S'

```
SELECT ename FROM emp WHERE ename LIKE 'S%';
```

- Leading Wildcard

- Find any string that ends with the letter 'S'

```
SELECT ename FROM emp WHERE ename LIKE '%S';
```

- Multiple Wildcards

- Find any string that contains, anywhere, the letter 'S'

```
SELECT ename FROM emp WHERE ename LIKE '%S%';
```

- Single Character Wildcard

- Find any string that contains the letter 'A' followed by any single character followed by

the letter 'E'

```
SELECT ename FROM emp WHERE ename LIKE '%A_E%';
```

SQL programmers need to be aware that the LIKE predicate typically performs a search without the normal performance benefit of indexes. Using '=', '<>', etc.. instead will increase performance. Users of the LIKE predicate should be aware that case sensitivity (e.g., 'S' versus 's') may be different based upon database product or configuration.

UNION

In SQL the UNION clause combines the results of two SQL queries into a single table of all matching rows. The two queries must result in the same number of columns and compatible data types in order to unite. Any duplicate records are automatically removed unless UNION ALL is used.

Note that UNION does not guarantee the order of rows. Rows from the second operand may appear before, after, or mixed with rows from the first operand. In situations where a specific order is desired, ORDER BY must be used.

Note that UNION ALL may be much faster than plain UNION.

```
sales2005
```

person amount

```
Joe 1000
```

```
Alex 2000
```

```
Bob 5000
```

```
sales2006
```

person amount

```
Joe 2000
```

```
Alex 2000
```

```
Zach 35000
```

Executing this statement:

```
SELECT * FROM sales2005
```

```
UNION
```

```
SELECT * FROM sales2006;
```

yields this result set, though the order of the rows can vary because no ORDER BY clause was

supplied:

person amount

Joe	1000
Alex	2000
Bob	5000
Joe	2000
Zach	35000

Note that there are two rows for Joe because these rows are distinct across their columns. There is only one row for Alex because these rows are not distinct for both columns.

JOIN

A SQL join clause combines records from two or more tables in a database. It creates a set that can be saved as a table or used as is. A JOIN is a means for combining fields from two tables by using values common to each. ANSI standard SQL specifies four types of JOINS: INNER, OUTER, LEFT, and RIGHT. In special cases, a table (base table, view, or joined table) can JOIN to itself in a self-join.

A programmer writes a JOIN predicate to identify the records for joining. If the evaluated predicate is true, the combined record is then produced in the expected format, a record set or a temporary table.

```
SELECT column_list FROM table_1 [INNER | LEFT | RIGHT] table_2 ON conditions_2 [INNER | LEFT | RIGHT] table_3 ON conditions_3 ... WHERE conditions
```

LEFT JOIN can be used when you want to retrieve the data from the main table (table_1) even if there is no match in other tables (table_2, table_3...). While RIGHT JOIN is used to retrieve the data from all other tables (table_2, table_3...) even if there is no match in the main table.

INNER JOIN

An inner join is the most common join operation used in applications and can be regarded as the default join-type. Inner join creates a new result table by combining column values of two tables (A and B) based upon the join-predicate. The query compares each row of A with each row of B to find all pairs of rows which satisfy the join-predicate. When the join-predicate is satisfied, column values for each matched pair of rows of A and B are combined into a result row. The result of the join can be defined as the outcome of first taking the Cartesian product (or cross-join) of all records in the tables (combining every record in table A with every record in table B)—then return all records which satisfy the join predicate. Current SQL implementations normally use other approaches like a hash join or a sort-merge join where possible, since computing the Cartesian product is very inefficient.

SQL specifies two different syntactical ways to express joins: "explicit join notation" and "implicit join notation".

The "explicit join notation" uses the JOIN keyword to specify the table to join, and the ON keyword to specify the predicates for the join, as in the following example:

```
SELECT *
```

```
FROM employee INNER JOIN department
ON employee.DepartmentID = department.DepartmentID;
```

The "implicit join notation" simply lists the tables for joining (in the FROM clause of the SELECT statement), using commas to separate them. Thus, it specifies a cross-join, and the WHERE clause may apply additional filter-predicates (which function comparably to the join-predicates in the explicit notation).

INSERT

```
INSERT INTO table [ ( column [, ...] ) ]
{ VALUES ( expression [, ...] ) | SELECT query }
```

INSERT allows one to insert new rows into a table. One can insert a single row at a time or several rows as a result of a query. The columns in the target list may be listed in any order. In every column not present in the target list, its default value will be inserted. If a column has no declared default value it will be assumed as NULL. If the expression for each column is not of the correct data type, automatic type coercion will be attempted.

```
INSERT INTO tbl_movies(title,year) VALUES('Alexander','2004')
```

UPDATE

```
UPDATE table SET column = expression [, ...]
[ FROM fromlist ]
[ WHERE condition ]
```

UPDATE changes the values of the columns specified for all rows which satisfy condition. Only the columns to be modified need appear as column. Array references use the same syntax found in SELECT. That is, either single array elements, a range of array elements or the entire array may be replaced with a single query. You must have write access to the table in order to modify it, as well as read access to any table whose values are mentioned in the WHERE condition.

```
UPDATE tbl_movies SET genre = 'Dramatic' WHERE genre = 'Drama';
```

The following is a partial list of the files, terms and utilities that were used.

- insert
- update
- select
- delete
- from
- where
- group by
- order by
- join

Topic 106: User Interfaces and Desktops

106.1 Install and Configure X11

Candidates should be able to install and configure X11.

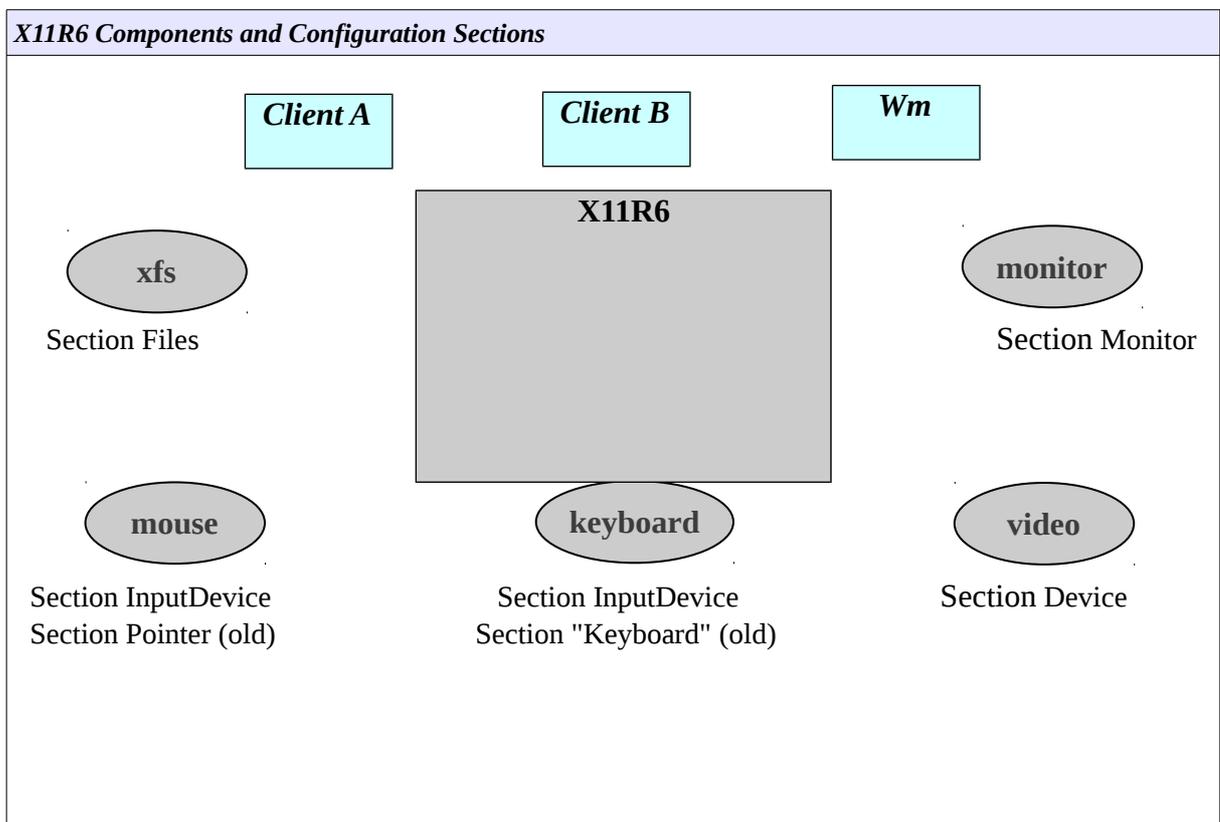
Key Knowledge Areas

- Verify that the video card and monitor are supported by an X server.
- Awareness of the X font server.
- Basic understanding and knowledge of the X Window configuration file.

Introduction

The X Windows system was developed as the display component of Project Athena at the Massachusetts Institute of Technology. It is the graphical environment for UNIX. The X Window system for Linux is based on the freely distributable port of X Window version 11 release 6 (Commonly referred to as **X11R6**).

This freely distributable port is commonly known as **xfree86** for the 80386/80486 and Pentium processor families. Since its initial port, Xfree86 has been ported to other computing platforms, including System V/386 and 386BSD.



The above diagram shows the components of the X11R6 server. The “Section” names refer to configuration sections in the **XF86Config** configuration file (covered in the next section).

The two clients depicted on top of the server are so-called *x-applications* (e.g. xclock or xterm). The window manager is also a client. Window managers add “windowing” facilities around the other x-application clients, allowing functionalities such as window dragging, focus, iconification, etc.

NOTICE:

The X11R6 server is independent from the clients that run on top. Clients are configured using specific configuration files or global files usually called **Xdefaults** or **Xresources**. The X server configuration file will only configure components such as the font server and font directories, mouse, keyboard, monitor resolution and color depth.

Configuring X11R6

Two of the configuration utilities provided with the Xfree86 software are the **XF86Setup** and **xf86config** scripts. Other vendors have specific utilities such as:

- Xconfigurator**, **redhat-config-xfree86** (RedHat)
- XFdrake** (Mandrake)
- sax** (Suse)

Once the server has been configured one can change the horizontal and vertical settings for the monitor with **xvidtune**.

All the above mentioned configuration utilities will create and edit the **XF86Config** configuration file. This file is read at start up by the X Server and determines its behavior. This file is typically found in the `/etc/X11` directory, and this is its' full path: **`/etc/X11/XF86Config`**.

There are 11 configuration sections in the config file, they are listed below:

ServerFlags	Modes
Module	Screen
InputDevice	ServerLayout
Device	DRI
VideoAdapter	Vendor
Monitor	

NOTICE:

The obsolete section names *Keyboard* and *Pointer* are still recognized for compatibility reasons, the new section name is now *InputDevice*

One of the first sections is the Section “Files”. The `FontPath` keyword tells whether to get fonts from a local directory or from a font server. The `RgbPath` keyword is used to indicate the full path to rgb text file used to map color names to RGB notation:

```
Section "Files"
    FontPath "/path/to/fonts/dir/"
    FontPath "trans/hostname: port"
    RgbPath  "/path/to/rgb"
EndSection
```

Where trans is the transport type **unix**, hostname is the fully qualified domain name of the font server, and port is the port to connect to, usually port 7100.

Example:

```
FontPath "unix/:7100" # Local Font Server
FontPath "unix/myfontserver.mydomain.com:7100"
```

Below is a sample **XF86Config** file:

```
Section "Files"
    RgbPath  "/usr/X11R6/lib/X11/rgb"
    FontPath
"/usr/X11R6/lib/X11/fonts/misc:unscaled,/usr/X11R6/lib/X11/fonts/75dpi:unscaled,/usr/X11R6/lib/X11/fonts/100dpi:unscaled,/usr/X11R6/lib/X11/fonts/misc/"
EndSection
```

```
Section "InputDevice"
    Identifier "Keyboard0"
    Driver     "keyboard"
EndSection
```

```
Section "InputDevice"
    Identifier "Mouse0"
    Driver     "mouse"
    Option    "Protocol" "IMPS/2"
    Option    "Device"  "/dev/psaux"
    Option    "ZAxisMapping" "4 5"
EndSection
```

```
Section "Monitor"
    Identifier      "Primary Monitor"
    VendorName     "Unknown"
    ModelName      "Unknown"
    HorizSync      31.5-37.9
    VertRefresh    55-90
    Modeline       "800x600" 40.00 800 840 968 1056 600 601 605 628 +hsync
+vsync
EndSection
```

```
Section "Device"
    Identifier     "Primary Card"
    VendorName     "Unknown"
    BoardName      "None"
    VideoRam      2048
```

EndSection

```

Section "Screen"
    Driver      "Accel"
    Device      "Primary Card"
    Monitor     "Primary Monitor"
    DefaultColorDepth 24
    BlankTime   0
    SuspendTime 0
    OffTime     0

    SubSection "Display"
        Depth    24
        Modes    "800x600"
    EndSubSection
    SubSection "Display"
        Depth    32
        Modes    "800x600"

```

Controlling X clients

Setting Fonts and Colours

X clients are configured using the **.Xresources** or **.Xdefaults** file. These file are kept in the users home directory. It is not automatically created by default, as system-wide defaults are also available for each program.

Below is an extract from a **.Xresources**:

```

xterm_color*background: Black
xterm_color*foreground: Wheat
xterm_color*cursorColor: Orchid
xterm_color*reverseVideo: false
xterm_color*scrollBar: true
xterm_color*saveLines: 5000
xterm_color*reverseWrap: true
xterm_color*font: fixed
xterm_color.geometry: 80x25+20+20
xterm_color*fullCursor: true
xterm_color*scrollTtyOutput: off
xterm_color*scrollKey: on
term_color*VT100.Translations: #override\n\
    <KeyPress>Prior : scroll-back(1,page)\n\
    <KeyPress>Next : scroll-forw(1,page)
xterm_color*titleBar: false

```

Each of these directives is a system default directive that describes how a client will be displayed. Each line consists of the client name followed by an asterisk and the X Window parameter. Through a carefully configured **.Xresources** file the user can define the way a client will look each time it is started.

The DISPLAY Variable

When an x-application (or X client) is started it needs to know which X server to run on. An X server is referred to as a display. For example the first X server you start (using **startx** for example) is called : **0** the second would be called **:1** and so on. The first X server (or display) running on the host 192.168.1.99 is called **192.168.1.99:0**

Most native X clients such as **xterm** or **xclock** have a **-display** switch which can be used to set the display. But the easiest method is to set the environment variable called **DISPLAY!**

The next two commands are equivalent:



```
xclock -display 192.168.1.99:0
```



```
DISPLAY=192.168.1.99:0 xclock
```

However the X server on the host 192.168.1.99 will not allow this x-application to run. The user that started the X server on the remote host (192.168.1.99) needs to run the **xhost** command. This tool can selectively add or remove hosts to an access control list.

Example: Allow remote x-applications from host 192.168.1.7 to run on local server



```
xhost + 192.168.1.7  
192.168.1.7 being added to access control list
```

NOTICE

The **xhost** mechanism must be used in conjunction with **xauth** (not part of the LPI objectives). For a remote x-client from 192.168.1.7 to run on our local server we still need to run the following on the local host:



```
xauth extract - $DISPLAY | ssh 192.168.1.7 xauth merge -
```

(Assuming that the user names are the same and that the host name contained in \$DISPLAY can be resolved)

Starting X

An X session can be started using 2 methods:

Method 1: From the command line, after logging in onto a virtual terminal the user launches the X Server using a script called **startx**

Method 2: A Display Manager is running prompting the user with a graphical login, this is available for a specific runlevel (on RedHat type distributions this is **runlevel 5**).

From the Command Line

The **startx** script starts **xinit**. The **xinit** script has two main arguments (a) the X server and (b) the **xinitrc** script. The **xinitrc** script will source (read) the files **Xresources** (controlling the x-applications) and the **Xclients** (choosing a window manager). So we can trace the startup sequence as follows:

```
startx --> xinit --> X -> xinitrc -> Xclients
```

Troubleshooting X Clients

Occasionally X Clients won't terminate properly leaving *zombie* processes. A zombie process is one whose parent processes has terminated, and cannot clear references to the child process. When a child process' parent exits leaving the child process still running, this is usually visible by running **ps** which will reveal the child process being owned by PID 1 (init). These processes should be killed because they may be using CPU resources. Killing such a process requires the user to be the user who owns the process, or root. It might be necessary to use the **-9** option to actually kill these processes.

Choosing a Window Manager

The area that is commonly referred to as the desktop is also known in the X Window world as the screen. It covers the entire area of your monitor display. The root window is the background of your screen, typically used to display a colour or picture. The window manager provides an interface between the user and the X server. It is virtually impossible to use X without a window manager, because it provides the title bar and the familiar buttons with which you manipulate the display.

Information on available window managers is available from the Window Managers website at <http://www.PliG.org/xwinman>. Many of the Linux versions of these window managers are available at <ftp://metalab.unc.edu/pub/Linux/X11/window-managers>.

In addition to the different window managers there are also various desktop environments, among which the most common are KDE, GNOME and XFCE.

Below is a brief list of integrated window managers:

Enlightenment

fvwm

icewm

amiWM

m1vwm

dfm

olwm

olvwm

mwm

Window Maker

The following is a partial list of files, terms and utilities that were used.

- /etc/X11/xorg.conf
- xhost
- DISPLAY
- xwininfo
- xdpinfo
- X

106.2 Set Up a Display Manager

Candidates should be able set up and customize a display manager. This objective covers the display managers XDM (X Display Manger), GDM (Gnome Display Manager) and KDM (KDE Display Manager).

Key Knowledge Areas

- Turn the display manager on or off.
- Change the display manager greeting.
- Change default color depth for the display manager.
- Configure display managers for use by X-stations.

The Display Manager

There are three main display managers, xdm (generic), gdm (GNOME) and kdm (KDE). A display manager will automatically be started if the system is running in a given runlevel (e.g runlevel 5). We first describe the login process; the next section covers more advanced functionalities of a Display Manager. The login process follows the following steps:

```
xdm --> xlogin --> Xsession --> (optionally) Xclients or ~/.Xclients
```

Different versions of display managers as well as different Linux distributions may use slightly different steps. In general however, note that **startx** uses **xinit** whereas **xdm** uses **Xsession**.

CUSTOMISING

Each user can further customise their environment by using a **.xinitrc** file. This file will be merged into the system **xinitrc**.

The **switdesk** tool allows users to define a custom **.Xclients** file

KDM

This display manager is installed with the KDE desktop environment. It is based on the generic **xdm** display manager and shares many common configuration files. These configuration files for are in **/usr/share/config/kdm**. The file that controls most functionalities is **kdmrc**.

The path to the **kdm** binary is **/usr/bin/kdm**. And the KDM Configuration files:

kdmrc Xaccess (same as xdm) **Xservers** (same as xdm) **Xsession** (same as xdm) **Xsetup Xstartup**

GDM

This display manager is distributed with the GNOME desktop environment. The main configuration file is **gdm.conf**

The path to the **gdm** binary is **/usr/bin/gdm**. GDM Configuration Files (**/etc/X11/gdm**):

[ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification] Created during the initiative "ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software in Africa", a programme of FOSSFA and GIZ. For more information see www.ict-innovation.foossfa.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.

Sessions/ `gdm.conf`

XDM

The `xdm` display manager is part of the Xfree86 application. The main configuration file is `xdm-config`

The path to the `xdm` binary is `/usr/bin/xdm`. XDM Configuration Files:

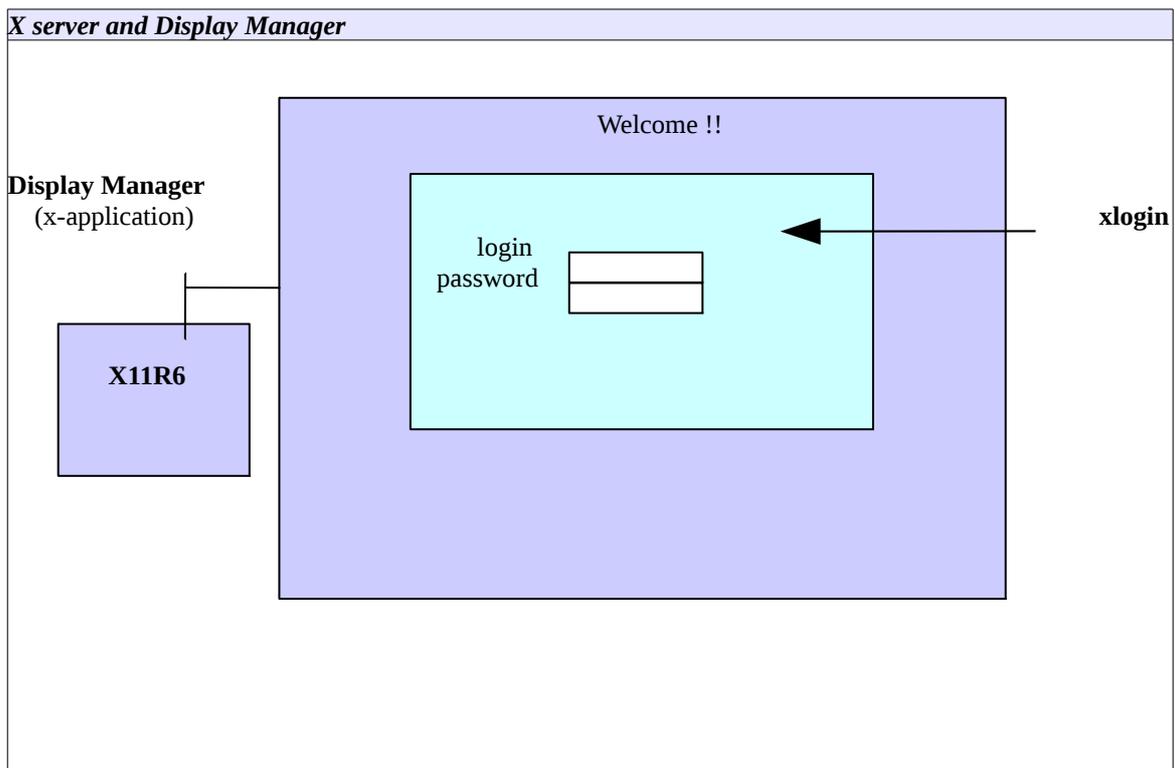
Xaccess Xresources Xsession xdm-config Xservers

We will look at the `xdm` configuration files in more detail later in this section.

Display Managers are used mainly in run level 5:

```
Set default runlevel in /etc/inittab
id:5:initdefault:
```

Display managers allow local users to log onto the system using the graphical interface. They can also be used to provide a graphical login interface over the network. For this they use a protocol called **XDMCP** or X Display Manager Control Protocol. By default XDMCP is disabled (we will enable XDMCP as an exercise).



Configuration Files

`/etc/X11/xdm/Xresources`

Since the Display Manager is also an x-application, the fonts, the background colors and **xlogin** can be configured with the **Xresources** file in `/etc/X11/xdm/`. When using **gdm**, the `/etc/X11/gdm/Init/Default` script will source **Xresources**.

`/etc/X11/xdm/Xservers`

This file simply maps the name of a display with an X server. For example display: 0 is understood to be the local X server. Remember that X always runs on the first free `/dev/tty`.

`/etc/X11/xdm/xdm-config`

This is the main configuration file for **xdm**. It is also used to enable XDMCP (see exercises)

`/etc/X11/xdm/Xaccess`

This file is used to enable XDMCP, allowing remote hosts to directly connect to the local server (using **-query**) or query about other display

The Xaccess file

```
# $XConsortium: Xaccess,v 1.5 91/08/26 11:52:51 rws Exp $
#
# Access control file for XDMCP connections
# To control Direct and Broadcast access:
#
#     pattern
#
# To control Indirect queries:
#
#     pattern          list of hostnames and/or macros ...
#
# To use the chooser:
#
#     pattern          CHOOSER BROADCAST
#
# or
#
#     pattern          CHOOSER list of hostnames and/or macros ...
#
# To define macros:
#
#     %name            list of hosts ...
#
# The first form tells xdm which displays to respond to itself.
# The second form tells xdm to forward indirect queries from hosts
matching
# the specified pattern to the indicated list of hosts.
# The third form tells xdm to handle indirect queries using the
chooser;
# the chooser is directed to send its own queries out via the broadcast
# address and display the results on the terminal.
# The fourth form is similar to the third, except instead of using the
# broadcast address, it sends DirectQuerys to each of the hosts in the
list
#
```

```
# In all cases, xdm uses the first entry which matches the terminal;
# for IndirectQuery messages only entries with right hand sides can
# match, for Direct and Broadcast Query messages, only entries without
# right hand sides can match.
#
*                               #any host can get a login window
#
# To hardwire a specific terminal to a specific host, you can
# leave the terminal sending indirect queries to this host, and
# use an entry of the form:
#
#terminal-a-host-a
#
# The nicest way to run the chooser is to just ask it to broadcast
# requests to the network - that way new hosts show up automatically.
# Sometimes, however, the chooser can't figure out how to broadcast,
# so this may not work in all environments.
#
*           CHOOSER BROADCAST           #any indirect host can get a chooser
#
# If you'd prefer to configure the set of hosts each terminal sees,
# then just uncomment these lines (and comment the CHOOSER line above)
# and edit the %hostlist line as appropriate
#
#%hostlist host-a host-b
#*           CHOOSER %hostlist           #
```

The Xservers file

```
# $XConsortium: Xserv.ws.cpp,v 1.3 93/09/28 14:30:30 gildea Exp $
#
#
# $XFree86: xc/programs/xdm/config/Xserv.ws.cpp,v 1.1.1.1.12.2
1998/10/04 15:23:14 hohndel Exp $
#
# Xservers file, workstation prototype
#
# This file should contain an entry to start the server on the
# local display; if you have more than one display (not screen),
# you can add entries to the list (one per line).  If you also
# have some X terminals connected which do not support XDMCP,
# you can add them here as well.  Each X terminal line should
# look like:
#     XTerminalName:0 foreign
#
#:0 local /usr/X11R6/bin/X
```

Since the Display Manager is also an *x-application* the **Xresources** file is similar to the **.Xresources** file except that it controls how the login screen is displayed.

Sample Xresources file

```
! $XConsortium: Xresources /main/8 1996/11/11 09:24:46 swick $
xlogin*borderWidth: 3
xlogin*greeting: CLIENTHOST
xlogin*namePrompt: login:\040
xlogin*fail: Login incorrect
#ifdef COLOR
xlogin*greetColor: CadetBlue
xlogin*failColor: red
*Foreground: black
*Background: #ffffff0
#else
xlogin*Foreground: black
xlogin*Background: white
#endif

XConsole.text.geometry:      480x130
XConsole.verbose:           true
XConsole*iconic: true
XConsole*font:              fixed
```

Sample xdm-config file

```
! $XFree86: xc/programs/xdm/config/xdm-conf.cpp,v 1.1.1.2.4.2 1999/10/12 18:33:29 hohndel Exp
$
!
DisplayManager.servers:      /etc/X11/xdm/Xservers
DisplayManager.accessFile:   /etc/X11/xdm/Xaccess
! All displays should use authorization, but we cannot be sure
! X terminals will be configured that way, so by default
! use authorization only for local displays :0, :1, etc.
DisplayManager._0.authorize: true
DisplayManager._1.authorize: true
!
DisplayManager*resources:    /etc/X11/xdm/Xresources
DisplayManager*session:      /etc/X11/xdm/Xsession
DisplayManager*authComplain: false
! SECURITY: do not listen for XDMCP or Chooser requests
! Comment out this line if you want to manage X terminals with xdm
DisplayManager.requestPort:  0
```

The following is a partial list of the files, terms and utilities that were used.

- startx
- xinit
- gdm
- xdm
- kdm
- /etc/X11/xdm/Xresources

- xdm-config

106.3 Accessibility

Candidates should be able to demonstrate knowledge and awareness of accessibility technologies.

Key Knowledge Areas

- Keyboard Accessibility Settings
- Visual Settings and Themes
- Assistive Technology (ATs)

Keyboard and Mouse Accessibility Options

Most computers have been designed for use by able bodied individuals. With the current rate of global computerization, persons with disabilities need access to computers. Linux Distributions are changing the face of computers for persons with various physical disabilities. For example, individuals who may press certain keys longer than usual can have the **Keyboard Repeat rate** adjusted accordingly. Some options might be available only in specific desktop environments (such as KDE or GNOME).

Some Linux Distributions have **Onscreen Keyboards**, and these can come in handy for people who can move the mouse but cannot actually type keys. In this case, the mouse is used to press the keys on the keyboard image, making it work much like a real one.

One example of an On-screen keyboard is the **GNOME On-Screen Keyboard (GOK)**. In addition to its keyboard capabilities, it provides shortcuts for the Mouse, Toolbar and Menu features of other programs, as well as tools to navigate a GNOME desktop.



```
$ sudo apt-get install gok
```

Almost all graphical Linux desktop environments have control panels for the keyboard and mouse. Normally, these might be accessible from: System > Preferences > Keyboard (or Mouse) or Hardware > Keyboard (Mouse).

Figure 106.3-1 Linux desktop environments provide control panels with accessibility options.

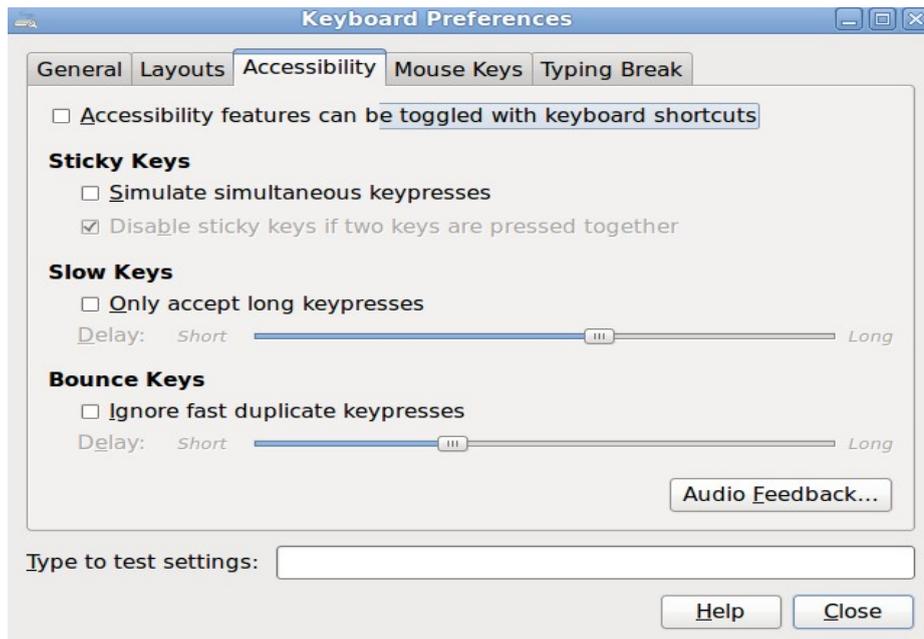


Figure 106.3-1 shows the Keyboard Preferences control panel.

Keyboard and mouse accessibility features include:

Keyboard Repeat Rate - As mentioned earlier, it may be necessary to disable keyboard repeat or set a very long delay for some users. Note that these settings override those set in the X configuration file.

Sticky Keys – For users who have difficulty pressing multiple keys simultaneously, Sticky keys allow modifier keys (Ctrl, Alt, and Shift) to “stick” when pressed. It then makes it easy to press key combinations that involve use of these modifiers. Usually, a Toggle Keys option will make a beeping sound to notify you if sticky keys are enabled or not.

Slow Keys – If a user accidentally presses a certain key more often than not, then this key can be activated as a slow key, so that in future, accidental presses do not result in that character being typed on the screen. The key will have to be pressed for a longer time before it registers.

Bounce and Debounce keys – When a user presses a single key and its registered multiple times, this is called the bounce effect. Linux distributions can allow a user to activate or deactivate bounce effect, as is normally seen in older keyboards.

Mouse Tracking and click options – As with the keyboard, the mouse options too can be adjusted to suit a specific user. Double Click Speed, Pointers, and mouse speed all can be adjusted to a specific desire.

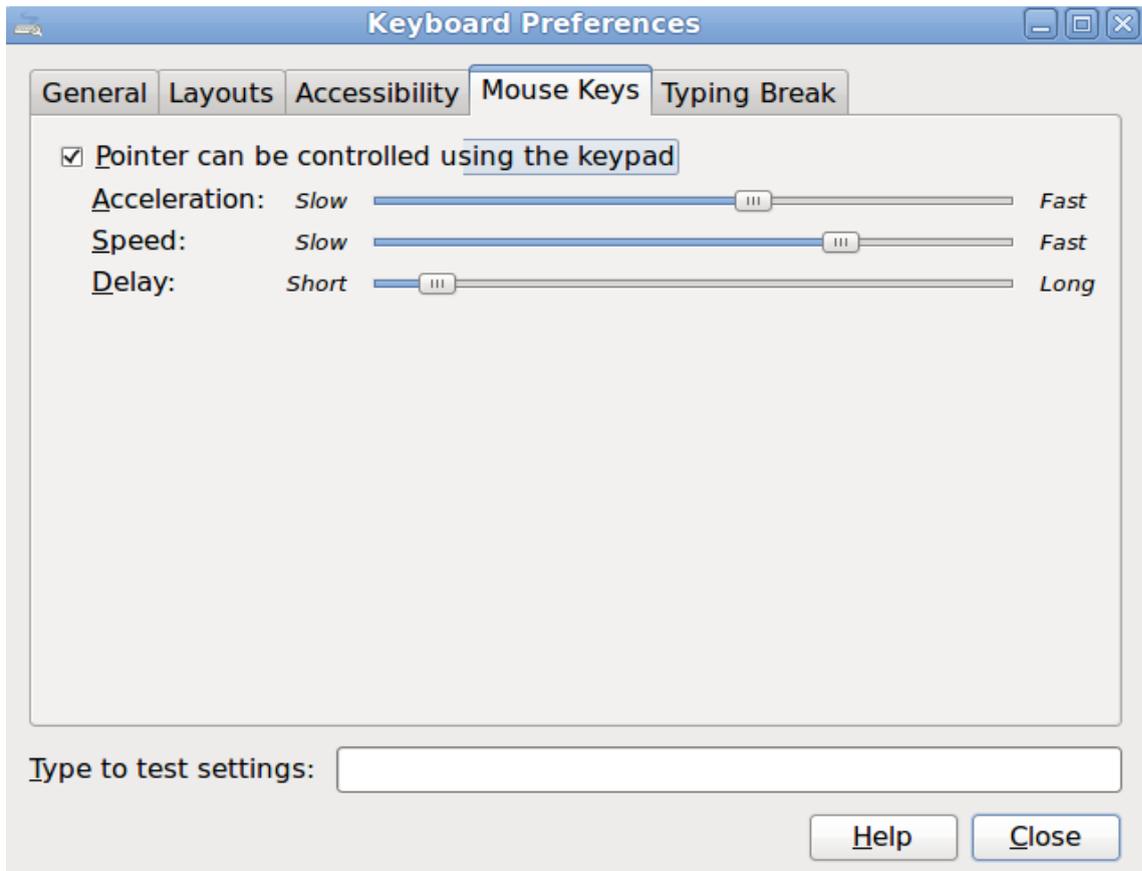


Figure 106.3-2 Linux desktop environments provide controls for Mouse Keys.

Simulated Mouse clicks - when a mouse stays over a certain area - and simulated double clicks - whenever the mouse button is pressed for an extended period this can also be achieved in some Linux environments.

Mouse Gestures - Just like keyboard shortcuts, you can permit your mouse to activate certain program options by moving it in a specific gesture.

Screen Display Settings

Sometimes, Users with poor eyesight may need to perform adjustments to screen settings in order to see the Applications and Menus more easily. Screen Magnification, as well as default Font options are available in this regard.

Fonts

Linux Desktop environments provide options to set the default fonts a user would like to see on the screen. Figure 106.3-3 shows a dialog box provided with GNOME (System>Preferences>Appearance menu on Ubuntu 10.04).

To adjust, simply click the Fonts tab, the result is a font selection dialog box, in which you can select the font (Sans, Times, and so on), the font style (normal, bold, and so on), and size in points. You can adjust these options to suit your liking.

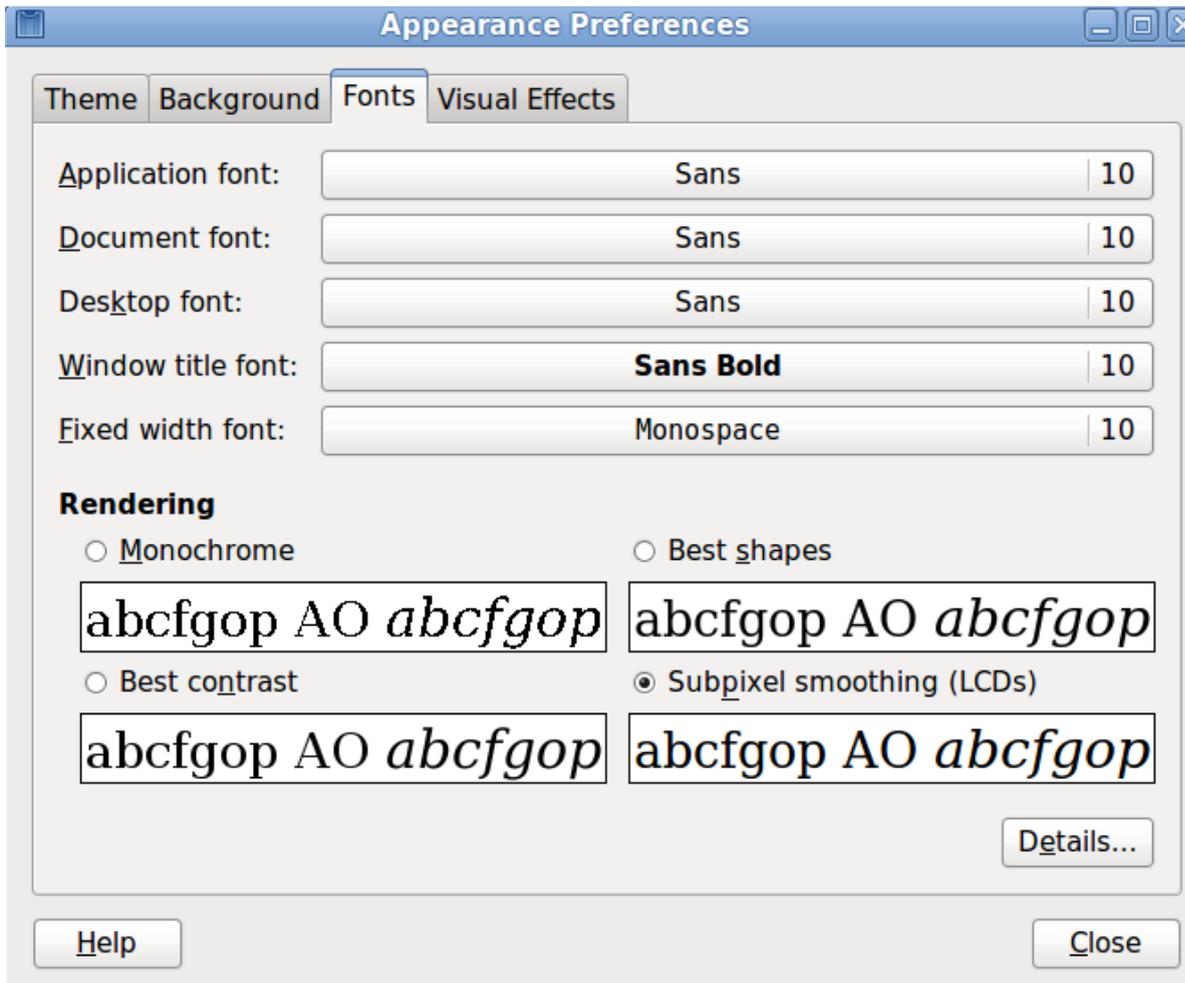


Figure 106.3-3 Linux desktop environments provide control panels with accessibility options.

NOTE: Not all applications will take their cues on fonts from the GNOME and KDE settings. You may have to adjust these within the individual applications.

Contrast

Linux Desktop environments provide various themes, with varying differences in their lay out— colors, window decorations, and so on. Computer Monitors also have their own contrast standards. All these can be adjusted for the best visibility.

NOTE - Adjustments afforded by desktop environments are independent of the physical monitor's contrast settings.

In GNOME, you can set themes in the same Appearance preferences dialog box in which you set the fonts (Figure 106.3-3); you click the Theme tab and select the theme you want to use.

Screen Magnification

A screen magnifier application typically enlarges part of the screen, especially the area immediately surrounding the mouse. One common screen magnifier is KMag, which is part of the KDE suite.

You may have to install kmag in order to use it.



```
$ sudo apt-get install kmag
```

To get kmag to run, just type `kmag` or select it from your desktop menus. A *KMag* window will open on the screen, and will typically enlarge the area around the cursor.

Using Additional Assistive Technologies

There are additional assistive technologies that can help those with special needs, such as screen readers and Braille displays. Perhaps the commonest computer speech synthesis products for Linux are Orca and Emacspeak.

Orca

Orca is a free open source tool that is available in GNOME 2.16 and later. It will allow you to read the parts of the screen using combinations of speech applications, Screen Magnifiers, and Braille.

Orca as an application is called `gnome-orca`. You may have to install Orca, as it's not default on most Linux distributions.



```
$ sudo apt-get install gnome-orca
```

Because Orca is in itself a set of other applications, not just an application, it does have dependencies such as:

- The Screen Magnifier – `gnome-mag`.
- Python programming language packages.

Orca can be configured using a text based wizard,



```
$ orca --text-setup
```

Or



```
$ orca --setup
```

Emacspeak aims to provide a great Linux Computer experience visually impaired users. Emacspeak is primarily a screen reader for `emacs`, and can be installed as below:



```
$ sudo apt-get install emacspeak
```

Braille Displays

Braille is a special type of computer which creates a tactile display of text-based information in Braille. You can use a Braille display with Linux text-mode console for anyone who is visually impaired. The Linux package required in order to use a Braille display, is called The **BRLTTY**. It provides a daemon that redirects text-mode console output to a Braille display, and can support speech synthesis, scroll back, and multiple virtual terminals.

Linux Kernels from 2.6.26 onwards add some support for Braille displays.

The following is a partial list of the files, terms and utilities that were used.

- Sticky/Repeat Keys
- Slow/Bounce/Toggle Keys
- Mouse Keys
- High Contrast/Large Print Desktop Themes
- Screen Reader
- Braille Display
- Screen Magnifier
- On-Screen Keyboard
- Gestures (used at login, for example gdm)
- Orca
- GOK
- Emacspeak

Topic 107: Administrative Tasks

107.1 Manage User and Group Accounts

Candidates should be able to add, remove, suspend and change user accounts.

Key Knowledge Areas

Add, modify and remove users and groups.

Manage user/group info in password/group databases.

Create and manage special purpose and limited accounts.

Linux is a multi-user environment. Each user belongs to one primary group (normally created as default) and possibly to additional groups. Ownership of files in Linux is closely related to user ids and groups. *Topic 107.1 prepares you to learn how to add, delete, and manage users and groups.* You will also learn about the files in `/etc`, where user and group information is stored.

Creating New Users

The `/usr/sbin/useradd` command adds new users to the system and the symbolic link `adduser` points to it.

Syntax: `useradd [options] login-name`

Example: add a user with login-name brian

```
useradd brian
```

Default values will be used when no options are specified. You can list these values with `useradd -D`.

Default options listed with `useradd -D`

```
GROUP=100  
HOME=/home  
INACTIVE=-1  
EXPIRE=  
SHELL=/bin/bash  
SKEL=/etc/skel
```

NOTE: This information is also available in the file `/etc/default/useradd`

The `id` command can be used to display basic information about user, such as their user ID, their group and any other group they may be part of.



```
# id brian  
uid=503(brian) gid=503(brian) groups=503(brian)
```

To allow a user to access his or her account the administrator must allocate a password to the user using the **passwd** tool.

Syntax: **passwd login-name**

When you press enter, the system will give you a prompt onto which to enter the password. Now the user brian has a password. Because the user was created using the default settings for the **useradd** command, the user's environment such as a *home directory*, *default shell*, and his *primary* group have all been set.

useradd (options)

-c	comment (Full Name)
-d	path to home directory
-g	initial group (GID). The GID must already exist
-G	comma separated list of supplementary groups
-u	user's UID
-s	user's default shell
-p	password (md5 encrypted, use quotes!)
-e	account expiry date
-k	the skel directory
-n	switch off the UPG group scheme

Working with Groups

Every new user is assigned to an initial (or *primary*) group. Two conventions exist.

Traditionally this *primary* group is the same for all users and is called **users** with a group id (GID) of **100**. Many Linux distributions adhere to this convention such as Suse and Debian.

The User Private Group scheme (UPG) was introduced by RedHat and changes this convention without changing the way in which UNIX groups work. With UPG each new user belongs to their own *primary* group. The group has the same name as the login-name (default), and the GID is in the 500 to 60000 range (same as UIDs).

NOTE: When using the traditional scheme for groups the user's **umask** (see LPI 101) is set to **022**, whereas in the UPG scheme the **umask** is set to **002**.

Belonging to Groups

A user can belong to any number of groups. However at any one time (when creating a file for example) only one group is the *effective* group.

The list of all groups a user belongs to is obtained with either the **groups** or **id** commands.

Example for user root:



```
uid=0(root) gid=0(root) groups=0(root), 1(bin), 2(daemon), 3(sys), 4(adm), 6(disk), 10(wheel), 600(sales)
```



```
groups
```

```
root bin daemon sys adm disk wheel sales
```

Joining a group

Joining a group changes the user's *effective* group and starts a new session from which the user can then logout. This is done with the **newgrp** command. The group should exist already.

Example: joining the *ict* group



```
newgrp ict
```

If the **groups** command is issued, the first group on the list would no longer be *root* but *ict*.

Creating and deleting groups

The **groupadd** tool is used to add new groups. It will add an entry in the `/etc/group` file.

Example: Create the group *devel*



```
groupadd devel
```

groupadd (options)

-g	assign a GID
-----------	--------------

The **groupdel** tool is used to delete groups. This will remove relevant entries in the `/etc/group` file.

Example: Delete the group *devel*



```
groupdel devel
```

Adding a user to a group

Administration tasks can be carried out with the **gpasswd** tool. One can add (**-a**) or remove (**-d**) users from a group and assign an administrator (**-A**). The tool was originally designed to set a single password on a group, allowing members of the same group to login with the same password. For security reasons this feature no longer works.

Example: Add andrew to the group devel

```
 gpasswd -a andrew devel
```

Configuration files

The /etc/passwd and /etc/shadow files:

The names of all the users on the system are kept in **/etc/passwd**. This file has the following structure:

- 1.Login name
- 2.Password (or x if using a shadow file)
- 3.The UID
- 4.The GID
- 5.Text description for the user
- 6.The user's home directory
7. The user's shell

These 7 fields are separated by colons. As in the example below.

```
george:$1$K05gMb0v$b7ryoKGTd2hDrw2sT.h:Dr G Micheal:/home/georges:/bin/bash
```

Shadow Passwords

In order to hide the encrypted passwords from ordinary users you should use a shadow file. The **/etc/shadow** file then holds the user names and encrypted passwords and is readable only by root.

If you don't have a shadow file in **/etc** then you should issue the following command:

```
 /usr/sbin/pwconv (passwd -> shadow)
```

This will leave an 'x' in the 2nd field of **/etc/passwd** and create the **/etc/shadow** file. If you don't wish to use shadow passwords you can do so using;



`/usr/sbin/pwunconv` (shadow -> passwd)

Caution: When using a shadow password file the `/etc/passwd` file may be world readable (644) and the `/etc/shadow` file must be more restricted (600 or even 400). However, when using `pwunconv` make sure to change the permissions on `/etc/passwd` (600 or 400).

The `/etc/group` and `gshadow` files:

In the same way, information about groups is kept in `/etc/group`. This file has 4 fields separated by colons.

1. Group name
2. The group password (or x if `gshadow` file exists)
3. The GID
4. A comma separated list of members

Example `/etc/group` entry:

```
java:x:550:jade, eric, rufus
```

As for users there is a `/etc/gshadow` file that is created when using shadow group passwords. The utilities used to switch backwards and forward from shadow to non-shadow files are as follows:



`/usr/sbin/grpconv` creates the `/etc/gshadow` file



`/usr/sbin/grpunconv` deletes the `gshadow` file

The `/etc/login.defs` and `/etc/skel` files

The `/etc/login.defs` file contains the following information:

- the mail spool directory: `MAIL_DIR`
- 1. password aging controls: `PASS_MAX_DAYS`, `PASS_MIN_DAYS`,
`PASS_MAX_LEN`, `PASS_WARN_AGE`
- max/min values for automatic UID selection in `useradd`: `UID_MIN`,
`UID_MAX`
- 0 max/min values for automatic GID selection in `groupadd`:
`GID_MIN`, `GID_MAX`
- automatically create a home directory with `useradd`: `CREATE_HOME`

The `/etc/skel` directory contains default files that will be copied to the home directory of newly

[ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification] Created during the initiative "ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software in Africa", a programme of FOSSFA and GIZ. For more information see www.ict-innovation.fossfa.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.

created users: `.bashrc`, `.bash_profiles`, ... these can be viewed by Viewing hidden files, in the user's home directory.

Modifying accounts and default settings

All available options while creating a user or a group can be modified. The `usermod` utility has the following main options:

usermod (options)

-d	the users directory
-g	the users initial GID
-l	the user's login name
-u	the user's UID
-s	the default shell.

Notice these options are the same as for `useradd`.

In the example below, User brian has been renamed to user2, and his shell is now the TurboC shell, and his home directory has also been changed.

```
# usermod -l user2 -s /bin/tcsh -d /home/user2 brian
```

Likewise, you can change details about a group with the `groupmod` utility. There are mainly two options:

-g	the GID
-----------	---------

Locking an account

A user's account can be locked by prefixing an exclamation mark to the user's password. This can also be done with the following command line tools:

Lock	Unlock
<code>passwd -l</code>	<code>passwd -u</code>
<code>usermod -L</code>	<code>usermod -U</code>

When using shadow passwords, replace the `x` with a `*` A less useful option is to remove the password entirely with `passwd -d`. Finally, one can also assign `/bin/false` to the user's default shell in `/etc/passwd`.

Changing the password expiry dates:

By default a user's password is valid for 99999 days, that is 273,9 years (default `PASS_MAX_DAYS`). The user is warned for 7 days that his password will expire (default `PASS_WARN_AGE`) with the following message as he logs in:

Warning: your password will expire in 6 days

There is another password aging policy number that is called `PASS_MIN_DAYS`. This is the minimum number of days before a user can change his password; it is set to zero by default.

The **chage** tool allows an administrator to change all these options.

Usage: `chage [-l] [-m min_days] [-M max_days] [-W warn] [-I inactive] [-E expire] [-d last_day] user`

The first option `-l` lists the current policy values for a user. We will only discuss the `-E` option. This locks an account at a given date. The date is either in UNIX days or in YYYY/MM/DD format.

Notice that all these values are stored in the `/etc/shadow` file, and can be edited directly.

Removing an account:

A user's account may be removed with the **userdel** command. To make sure that the user's home directory is also deleted use the `-r` option.



```
userdel -r jade
```

Special-purpose accounts

Root has a user id of 0, whereas other some users have user id's starting at the `UID_MIN` value set in `/etc/login.defs`, usually set to 500 or 1000.

Root is not the only special purpose account on a Linux system. There are others for daemons such as mail, SSH, FTP, news, and others. These accounts can be viewed from the `/etc/passwd` file.

```
ntp:x:38:38::/etc/ntp:/sbin/nologin
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
news:x:9:13:news:/etc/news:
nobody:x:99:99:Nobody:/:/sbin/nologin
uucp:x:10:14:uucp:/var/spool/uucp:/sbin/nologin
apache:x:48:48:Apache:/var/www:/sbin/nologin
gopher:x:13:30:gopher:/var/gopher:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
```

As you can imagine, special purpose accounts cannot be accessed via regular login. They help to control files and daemons, and as such have a special login shell : `/sbin/nologin`, or `/bin/false`. Any login attempts on these shells will automatically fail.

The following is a partial list of the files, terms and utilities that were used.

- `/etc/passwd`
- `/etc/shadow`
- `/etc/group`
- `/etc/skel`
- `chage`
- `groupadd`
- `groupdel`
- `groupmod`



- passwd
- useradd

- userdel
- usermod

107.2 Automate System Administration Tasks

Candidates should be able to use cron or anacron to run jobs at regular intervals and to use at to run jobs at a specific time.

Key Knowledge Areas

- Manage cron and at jobs.
- Configure user access to cron and at services.

1. Cron Jobs

A Cron is a time scheduled jobs on the UNIX system. For any administrative tasks that have to be run regularly, such as Back Ups and Network Services, then the cron facility is the best way to do it. This section is about the cron, anacron and other such facilities.

The cron facility, consists of the crond daemon and a set of tables – crontabs - describing what work is to be done, when and how frequently. The daemon, which is started by init, wakes up every minute and checks the crontabs to determine what is to be done. Users manage crontabs using the crontab command.

Cron jobs can be run as shell scripts, and are better designed to accept no parameters.

crontabs

To create a crontab, the crontab command with the -e (for "edit") option will open a text editor where your specifications of the cron job can be specified. Every crontab entry will contain six fields:

Minute, hour, day of the month, month of the year, day of the week and String to be executed by sh.

The respective ranges for the time fields are: 0-59, 0-23, 1-31 and 1-12, 0-6 (Sunday=0).

NOTE: For all time fields, you can specify a range, or a single unit of time. You can specify the day of the week, using its short name: sun, mon, tue, and so on.

The final field will always be interpreted as a string to pass to the Bash.

Crontabs can use special characters, like % = newline, but these have to be preceded with a backslash (\). The line up to the first % is passed to the shell, while any line(s) after the % are passed as standard input.

Crontab Example:



```
0,30 20-23 * 8 mon-fri /home/tux/backup.sh
```

Here, the back up shell script is executed at the 0th and 30th minutes (every 30 minutes), for the hours between 8 P.M and 11 P.M. every monday to Friday, during August. See the man page for crontab(5)

for details on additional ways to specify times./ For further details on how to specify times please see the man page for crontab.

Cron jobs output can is usually sent as an email to the user who set up the cron job. Normally, like other system processes, the syslog facility will capture whatever the cron job did. Below is an example of a mailed report.



```
From tux@it.northpole.com Mon Jul  2 23:00:02 2010
Date: Mon, 2 Jul 2010 23:00:01 -0400
From: root@it.northpole.com (Cron Daemon)
To: tux@it.northpole.com
Subject: Cron <tux@it> /home/tux/backup.sh
Content-Type: text/plain; charset=UTF-8
Auto-Submitted: auto-generated
X-Cron-Env: <SHELL=/bin/sh>
X-Cron-Env: <HOME=/home/tux>
X-Cron-Env: <PATH=/usr/bin:/bin>
X-Cron-Env: <LOGNAME=tux>
X-Cron-Env: <USER=tux>
```

BackUp was successful!

Location of the crontabs

All crontabs are stored in /etc/spool/cron/*name of the user who created it*. Thus crontab is a suid program it can only run by a user with root authority.

The cron facility checks some more directories as well i.e. /etc/crontab and /etc/cron.d. As these are not user but system crontabs they have one more field between "day" and "command". This field just defines for which user the command should be run (root in normal case).

/etc/crontab example



```
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
HOME=/
# run-parts
01 * * * * root run-parts /etc/cron.hourly
02 4 * * * root run-parts /etc/cron.daily
22 4 * * 0 root run-parts /etc/cron.weekly
42 4 1 * * root run-parts /etc/cron.monthly
```

In this case, the cron jobs are done by the run-parts command, whereas the /etc/crontab simply controls the timing.

NOTE: Note also that the crontab can include shell variables that have been set before the commands are run. All these cronjobs are run for the user root.

Anacron - for non 24/7 machines

[ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification] Created during the initiative "ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software in Africa", a programme of FOSSFA and GIZ. For more information see www.ict-innovation.fossfa.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.

Whereas the cron facility is ideal on Computer systems that run continuously, and as such jobs that run hourly, some systems may infact require to be turned off every now and then. Consequently, you can only schedule jobs that run daily, weekly or monthly. A facility to do this is called *anacron* (for "anachronistic cron").

Anacron will keep timestamp files in `/var/spool/anacron` as a way to record when jobs are run. Anacron works by checking to see if the required number of days has passed since the job was last run and then determines to run it if necessary. The anacrontab is stored in `/etc/anacrontab`, and may contain environment variables. Every anacron job has four fields.

Time – Delay – Job-identifier - Command

The time is always a number of days, but can be specified as `@weekly`, `@monthly` to ensure that a job runs only then. The delay is the number of minutes to wait after the job is due to run before actually starting it. This feature can be useful if you do not want to flood the system with multiple jobs! You can use this to prevent a flood of jobs when a system first starts.

NOTE – Unlike the specific jobs contained in them, both `/etc/crontab` and `/etc/anacrontab` – can only be updated by direct editing, not by `crontab -e`

Sometimes you just want to run a job once. Linux provides the `at` command. The instructions to be executed are read from a file specified with the `-f` option, or from `stdin` if `-f` is not used. To display the time for the job to run, you can use the `-v` option. Below is an example:



```
$ at -f backup.sh -v 10:25
Wed Jul 7 10:25:00 2010

job 5 at Wed Jul 7 10:25:00 2010
```

Job output from `at`



```
From tux@it.northpole.com Wed Jul 7 10:25:00 2010
Date: Wed, 7 Jul 2010 10:25:00 -0400
From: Tux <tux@it.northpole.com>
Subject: Output from your job 5
To: tux@it.northpole.com

BackUp was successful!
```

The `at` command also has a `-q` option. When used, this option increases the nice value for the job. Be sure to look at the man pages for more details on these features.

Managing Jobs

You can use the `crontab` command with the `-l` option to list your `crontab`, and use the `atq` command to display the jobs you have queued using the `at` command, as shown in below.

Displaying scheduled jobs



```
$ crontab -l
```

```
0,30 20-23 * 8 mon-fri /home/tux/backup.sh
$ atq
13      Wed Jul  7 02:00:00 2010 a tux
14      Sat Jul 10 02:00:00 2010 a tux
15      Sun Jul 11 22:00:00 2010 a tux
16      Tue Jul 13 02:00:00 2010 a tux
```

NOTE – For details on the actual command scheduled for execution by `at`, you can use the `-c` option and the job number. You will notice that most of the environment that was active at the time the `at` command was issued is saved with the scheduled job.

Deleting Jobs

You can delete all scheduled cron jobs using the `crontab` command with the `-r` option as shown below:



```
$ crontab -l
0,30 20-23 * 8 mon-fri /home/tux/backup.sh
$ crontab -r
$ crontab -l
no crontab for tux
```

You can delete system cron or anacron jobs, by editing the `/etc/crontab`, `/etc/anacrontab`, or by editing and/or deleting files in the `/etc/cron.d` directory.

For jobs scheduled using the `at` command, you can delete them using the `atrm` command with the job number. Multiple jobs have to be separated by a space.



```
$ atq
13      Wed Jul  7 02:00:00 2010 a tux
14      Sat Jul 10 02:00:00 2010 a tux
15      Sun Jul 11 22:00:00 2010 a tux
16      Tue Jul 13 02:00:00 2010 a tux
$ atrm 16 14 15
$ atq
13      Wed Jul  7 02:00:00 2010 a tux
```

Manging non-root user access

There may be situations when a non root user needs to have access to the crontab and the cron facility. In such a case, all you have to do is create the file `/etc/cron.allow` (if it does not already exist), and any non-root user must be listed in it. In contrasting effect, if `/etc/cron.deny` does exist, a non-root user who is listed in it cannot use `crontab` or the cron facility. If the `/etc/cron.deny` file is empty (and this is the default), all users are allowed to use the cron facility.

The following is a partial list of the files, terms and utilities that were used.

- `/etc/cron.{d,daily,hourly,monthly,weekly}`
- `/etc/at.deny`

- /etc/at.allow
- /etc/crontab
- /etc/cron.allow
- /etc/cron.deny
- /var/spool/cron/*
- crontab
- at
- atq
- atrm

107.3 Localization and Internationalization

Candidates should be able to localize a system in a different language than English. As well, an understanding of why LANG=C is useful when scripting.

Key Knowledge Areas

- Locale settings.
- Timezone settings.

Time is very important on the Linux System. Many facilities such as Anacron, Crontab, Backup, Restore, Update Managers all need accurate time to be able to perform their executions well. Developments in computing have resulted in computer systems that are able to keep time even when the computer is switched off.

There are 2 kinds of clocks on the Linux system;
Hardware Clock aka RTC, RealTimeClock, CMOS Clock, BIOS Clock. This clock runs independent of the Operating System and runs even when the computer is turned OFF, as long as the CMOS battery lasts.

Software Clock aka System Clock. This clock runs via the system timer interrupt. Normally, it will count the number of seconds since 1st Jan 1970. Is the main clock under Linux. At boot time it reads the hardware clock and continues alone from there.

Under Linux 2 main programs are used to control the 2 clocks. The hardware clock can be directly changed with the hwclock utility. The main options are:

- r or --show prints the current times
- w or --systohc set the hardware clock to the current system time
- s or --hctosys set the system time to the current hardware clock time



```
# date; hwclock; hwclock -s; date
Mon Jul  9 22:20:23 EDT 2007
Mon 09 Jul 2007 11:19:01 PM EDT  -0.414881 seconds
Mon Jul  9 23:19:02 EDT 2007
```

The System Clock is controlled by the date facility. It can also be used to show the System time.

Syntax: date [options]

Options include: +text_and_metacodes

To display System time, using the date utility, you can issue a command like:
date "+It is now %H Hours and %M Minutes"

Will have the following result: It is now 14 Hours and 33 Minutes

Setting the Time and Date in Linux

You can set the time using 2 relatively simple procedures:

- 1) Set the Hardware clock to UTC via the BIOS setup.
- 2) Set the environment variable TZ to the proper time zone using the script: tzselect

The System date on the other hand can be changed with the date command. The syntax is:

```
date MMDDhhmmCCYY[.ss]
```

Note: In the file /etc/adjtime the correction factor can be saved to keep the clock as accurate as possible.



```
# cat /etc/adjtime
0.000990 1184019960 0.000000
1184019960
LOCAL
# cat /etc/adjtime
-0.003247 1182889954 0.000000
1182889954
LOCAL
```

Time Zone Configuration

The time zone is a simply a measure of the difference of your local time from UTC. The /usr/share/zoneinfo file will store information on all available time zones. On many modern systems, this file the /usr/share filesystem needs to be mounted when the local time zone information is needed early in the boot process.

Some countries, in addition to UCT (used interchangeably with UTC) time, apply “day light saving”, a phenomenon where clocks are moved an hour ahead or behind at a specific date every year. Such policies are available on a Linux system in /usr/share/zoneinfo/. As long as the appropriate zone file is copied to /etc/localtime, a user can ensure that Time Zone's policies are in effect.

For example if we copy /usr/share/zoneinfo/Nairobi to /etc/localtime the next time we run date this will give us the time in Nairobi. This is because date will read /etc/localtime each time it is run.



```
# cat /etc/timezone
France/Paris
# cat /etc/sysconfig/clock
# The ZONE parameter is only evaluated by system-config-date.
# The timezone of the system is defined by the contents of
/etc/localtime.
ZONE="France/Paris"
UTC=false
ARC=false
```

Setting System time with tzconfig

You can use the tzconfig utility to modify your time zone.

```
# tzconfig
Your current time zone is set to France/Paris
Do you want to change that? [n]:
Your time zone will not be changed
```

Using NTP

NTP stands for Network Time Protocol. It is defined in RFC1305 to provide for the transfer and maintenance of time functions over multiple network systems. One of the commonest NTP servers in use is ntpd (ntp.isc.org) It works by correcting with delicate accuracy, any differences of the system clock, as opposed to large adjustments at once or within very few hours.

The computer synchronizes its time by sending messages to the *time server*. The time returned is adjusted by an offset of half the round-trip delay. As a result, the accuracy of the time is therefore dependent on the network latency in both directions. Faster networks (the shorter the path to a time server), are likely to have a more accurate time.

Graphically, you should be able to set your NTP time servers using a dialog similar to that in the figure below.

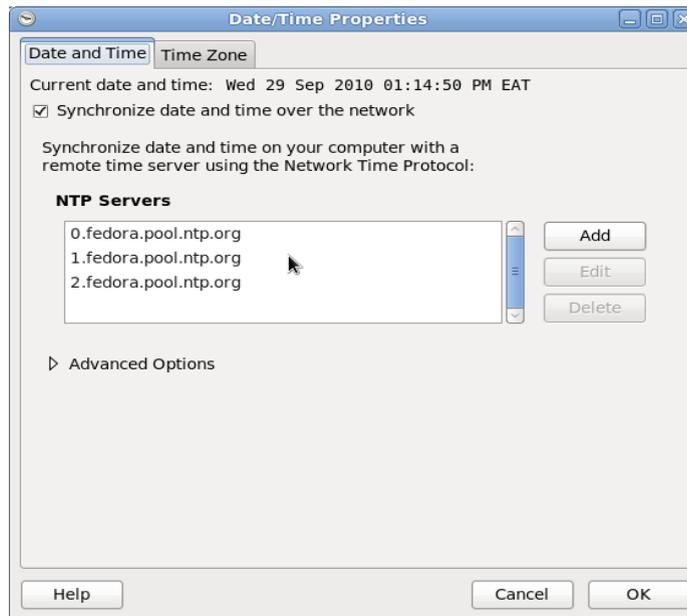


Figure 107.3-2. Setting NTP servers

NTP configurations are kept in `/etc/ntp.conf`, but the ntpd daemon has to be restarted after you save any changes to the configuration file.

Configuring a client to query an NTP server:

An NTP daemon called ntpd is used to regularly query a remote time server. All that is needed is a

[ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification] Created during the initiative "ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software in Africa", a programme of FOSSFA and GIZ. For more information see www.ict-innovation.fossfa.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.

server entry in `/etc/ntp.conf` pointing to a public or corporate NTP server. Public NTP servers can be found online.

The NTP protocol can also estimate the frequency errors of the hardware clock from a sequence of queries, this estimate is written to a file referred to by the `driftfile` tag.

NTP Commands

The `ntpdate` command can be used to set system time from an NTP time server as shown below

```
# ntpdate 0.us.pool.ntp.org
10 Jul 10:27:39 ntpdate[15308]: adjust time server 66.199.242.154 offset -0.007271 sec
```

NTP servers run in round robin mode, so its possible that the next time you run `ntpdate`, a different server will be queried. The pool from which a server is selected can be viewed when u dig your DNS, as shown below.

Round robin NTP server pool

```
# dig 0.pool.ntp.org +noall +answer | head -n 5
0.pool.ntp.org. 1062 IN A 217.116.227.3
0.pool.ntp.org. 1062 IN A 24.215.0.24
0.pool.ntp.org. 1062 IN A 62.66.254.154
0.pool.ntp.org. 1062 IN A 76.168.30.201
0.pool.ntp.org. 1062 IN A 81.169.139.140
```

The `ntpdate` command can be effected using `ntpd` with the `-q` option, as shown below.

Setting system time using `ntpd -q`

```
# ntpd -q
ntpd: time slew -0.014406s
```

Note: `ntpd` command uses the time server information from `/etc/ntp.conf`. If the `ntpd` daemon is running, `ntpd -q` will quietly exit, leaving a failure message in `/var/log/messages`. The NTP daemon itself can be queired using the `ntpq` command.

The following is a partial list of the files, terms and utilities that were used

- `/etc/timezone`
- `/etc/localtime`
- `/usr/share/zoneinfo`
- Environment variables:
 - `LC_*`

- LC_ALL
- LANG
- /usr/bin/locale
- tzselect
- tzconfig
- date
- iconv
- UTF-8
- ISO-8859
- ASCII
- Unicode

Topic 108: Essential System Services

108.1: Maintain System Time

Candidates should be able to properly maintain the system time and synchronise the clock via NTP

Key Knowledge Areas

- Set the system date and time.
- Set the hardware clock to the correct time in UTC.
- Configure the correct time zone.
- Basic NTP configuration.
- Knowledge of using the `pool.ntp.org` service

System Time and the Hardware clock

There are two separate clocks in a Linux system. The *hardware clock* (on the motherboard) runs independently of the operating system, and even when the computer is switched off. The *system time* is maintained by the Linux kernel and is incremented by a timer interrupt. When Linux starts up it initialises the system time from the hardware clock. Linux does not normally refer to the hardware clock again while it is running.

The hardware clock can be examined and modified with the `hwclock` utility. The main options are:

<code>-r</code> or <code>--show</code>	prints the current times
<code>-w</code> or <code>--systohc</code>	set the hardware clock to the current system time
<code>-s</code> or <code>--hctosys</code>	set the system time to the current hardware clock time
<code>--set</code>	set the hardware clock to a specified time and date

Only the superuser can change the hardware clock setting

Example:

```
# hwclock --set --date="10/14/2010 16:55:05"
# hwclock --show
Thu 14 Oct 2010 04:55:09 PM BST -0.673528 seconds
```

The system time is maintained by the kernel as the number of seconds since the beginning of 1 January 1970. This moment of time is known as “the epoch”.

Display the system time using the `date` command:

```


$ date
Tue Sep 14 10:18:33 BST 2010

```

The system time can also be changed with the date command. Only the superuser can do this. The syntax is:

```
date MMDDhhmmCCYY[.ss]
```

Example:

```


# date 101216552010
Tue Oct 12 16:55:00 BST 2010

```

Time Zones

The computer's hardware clock is usually set to UTC regardless of the computer's geographic location. However, the times reported (and set) by the **date** command are usually in the local time zone of the computer. The *timezone* defines the offset between local time and UTC. On some linux distributions the timezone is defined in the file `/etc/timezone`.

In addition to UCT time some countries apply “daylight saving” policies which add or remove an hour at a given date every year. These policies are defined by files in `/usr/share/zoneinfo/`. These are binary files. By copying the appropriate zone file to `/etc/localtime` one can establish a particular zone policy.

For example if we copy `/usr/share/zoneinfo/Hongkong` to `/etc/localtime` the next time we run `date` this will give us the time in Hongkong. This is because `date` will read `/etc/localtime` each time it is run.

Using NTP

Both the hardware clock and the system time can drift due to small errors in clock frequencies. Over time these drifts can accumulate. Clock skew between computers may cause problems with tools such as `make`, which examines the timestamps on files, and with authentication protocols that use timestamps to limit the validity of authentication messages.

NTP (**Network Time Protocol**) is a protocol that can synchronise a computer's clock with a time server, to an accuracy of a few milliseconds.

Computers that are directly updated (by an atomic clock, say) are called *primary* time servers and are used to update a larger number of *secondary* time servers. This forms a tree structure similar to the DNS structure. The root servers are on the first level or *stratum*, the secondary servers on the second and so on.

Configuring a client to query an NTP server:

An NTP daemon called `ntpd` is used to regularly query one or more remote time servers. It adjusts the rate of the kernel's clock so that it is gradually adjusted to the correct time. Abrupt changes, especially adjusting the clock backwards, are avoided. The daemon is very easy to configure. All that is needed is one or more server entries in `/etc/ntp.conf` pointing to a public

[ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification] Created during the initiative "ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software in Africa", a programme of FOSSFA and GIZ. For more information see www.ict-innovation.fossfa.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.

or corporate NTP server. A list of public NTP servers can be found online at <http://support.ntp.org>. The time servers at pool.ntp.org use DNS “round robin” to make a random selection from a pool of time servers that have volunteered to be in the pool. This is good enough for many end users. A simple ntp.conf file to use the pool might look like this:

A simple ntp.conf file

```
server 0.pool.ntp.orgserver 1.pool.ntp.org
driftfile /var/lib/ntp/ntp.drift
```

The NTP protocol can also estimate the frequency errors of the hardware clock from a sequence of queries; this estimate is written to a file referred to by the driftfile tag.

Once **ntpd** is started it will itself be an NTP server providing services on port 123 using UDP.

One off adjustments:

The **ntp** package also provides the ntpdate tool which can be used to set the time from the command line. The -q option simply queries the server, without -q the command will set the system time.

Example:



```
$ ntpdate -q ntp.ubuntu.com
server 91.189.94.4, stratum 2, offset 0.009249, delay 0.05046
14 Sep 11:25:23 ntpdate[9493]: adjust time server 91.189.94.4 offset
0.009249
```

The following is a partial list of the used files, terms and utilities:

- /usr/share/zoneinfo
- /etc/timezone
- /etc/localtime
- /etc/ntp.conf
- date
- hwclock
- ntpd
- ntpdate
- pool.ntp.org

108.2 System Logging

Candidates should be able to configure the `syslog` daemon. This objective also includes configuring the logging daemon to send log output to a central log server or accept log output as a central log server.

Key Knowledge Areas

- `syslog` configuration files
- `syslog`
- standard facilities, priorities and actions

The `/var/log/` directory

Many services report their activities by writing messages to log files. Some services such as Apache, Samba and the X server write their own log files directly. Many others do their logging through a logging service called `syslog`. This service is provided through two daemons – `syslogd` handles application logs and `klogd` handles kernel logs. The advantage of using `syslog` is that it provides a central place to control where the logged messages will go.

Most log files are in the directory `/var/log`. The exact set of files that you will find here varies between Linux distributions but those that you will commonly find include:

<code>cron</code>	keeps track of messages generated when cron executes
<code>mail</code>	messages relating to mail
<code>messages</code>	logs all messages except private authentication <code>authpriv</code> , <code>cron</code> , <code>mail</code> and <code>news</code>
<code>secure</code>	logs all failed authentications, users added/deleted etc.

The most important log file is `messages` where most activities are logged.

The `/etc/syslog.conf` file

When `syslogd` is started it reads the configuration file `/etc/syslog.conf`. (It is possible to start `syslogd` with the `-f` option to specify an alternative config file.) This file contains a list of rules. The first part of each rule identifies which messages the rule applies to and the second part of each rule specifies an action (what to do with the message).

`Syslog` can send messages to:

- A file
- A logged-in user
- Another `syslog` running on a remote machine

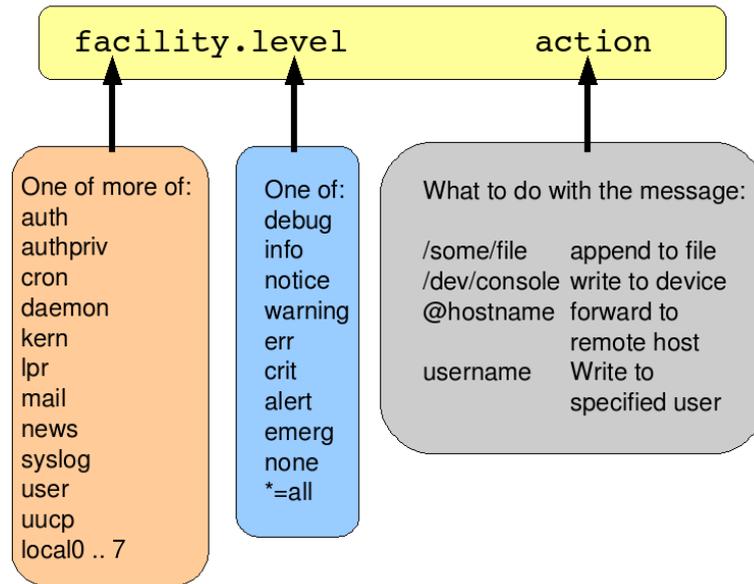


Figure 108.2-1: Format of rules in /etc/syslog.conf

Messages are identified using a *facility* and a *priority level*. The facility indicates approximately where the message came from and the priority indicates how important the message is.

The available facilities and priorities are shown in the figure. They are predefined and cannot be extended; however note that facilities local0 to local7 are provided for general use.

Valid priorities are: (from highest to lowest)

```
emerg
alert
crit
err
warning
notice
info
debug
```

In addition, the wild-card '*' can be used to mean "all facilities" or "all priorities".

Consider the following four example lines in `syslog.conf`. (These are not meant to represent a working configuration but simply to provide examples of the syntax. Also the line numbers are for reference, they are not part of the file):

```
1. cron.notice           /var/log/cron
2. *.*;authpriv.none    /var/log/messages
3. mail.*                ~/var/log/mail
4. *.crit                *
5. *.crit                @neptune
6. lpr.=info            /var/log/lpr
```

Line 1 matches messages from the `cron` facility at the notice priority or higher. These messages are appended to `/var/log/cron`

Line 2 matches all messages except those from the `authpriv` facility (note the use of 'none' here).

Line 3 matches all messages from the mail facility. These messages are appended to `/var/log/mail`. The '~' in front of the file name suppresses syslog's normal behaviour which is to flush the output file after every message is written. Flushing degrades I/O performance and is probably not appropriate for low-priority, high-volume messages such as a mailer daemon might generate.

Line 4 matches all messages at level `crit` (or above) and writes them to all logged in users. (On a server this is unlikely to be an effective way of getting anyone's attention!)

Line 5 shows how messages can be forwarded to another machine ("neptune" in the example).

Line 6 shows the use of '=' to match just a single priority level.

Note that a single message can match more than one rule, in which case, more than one action will be taken.

Listing of `/etc/syslog.conf`

```
# Log all kernel messages to the console.
# Logging much else clutters up the screen.
# kern.*                               /dev/console
# Log anything (except mail) of level info or higher.
# Don't log private authentication messages!
*.info;mail.none;news.none;authpriv.none /var/log/messages

# The authpriv file has restricted access.
authpriv.*                               /var/log/secure

# Log all the mail messages in one place.
mail.*                                    /var/log/maillog

# Log cron stuff
cron.*                                    /var/log/cron

# Everybody gets emergency messages, plus log them on another
# machine.
*.emerg                                   *
*.emerg                                   @10.1.1.254

# Save boot messages also to boot.log
local7.*                                   /var/log/boot.log
#
news.=crit                                /var/log/news/news.crit
news.=err                                  /var/log/news/news.err
news.notice                               /var/log/news/news.notice
```

Note that if the `syslog.conf` file is changed, the `syslogd` daemon should be signalled to re-read it:



```
# pkill -HUP syslogd
```

Using a central logging server

On large networks it may be useful to forward messages to a secure central server. This has two advantages. First, it is easier to conduct log file analysis if the messages are all in one place. Second, it makes it much harder for an intruder to “cover his tracks” by doctoring the log files on a compromised machine. If the messages have also been forwarded to another machine the intruder will not be able to delete them unless he is able to compromise that machine also.

To do this, on the “upstream” machines, in the `syslog.conf` file simply specify an action of the form “@10.1.1.254” or “@loghost” for those messages that you want to forward. (“loghost” must be a resolvable name for your central logging server). On the “downstream” machine, you must start `syslogd` with the `-r` flag to tell it to listen (on UDP port 514) for forwarded messages. This is not the default, so you may need to modify the start-up script (probably `/etc/init.d/syslog`) to add this flag.

Log Utilities

The `logger` command

The command-line utility `logger` may be used to send messages to `syslogd`. It can be used interactively (usually to test your syslog configuration), or it can be used within a shell script.

Example:



```
# logger "invalid request from client"
```

By default, `logger` sends messages with a priority of `user.notice` and on a typical system these messages will end up in `/var/log/messages`:



```
# tail -1 /var/log/messages
Sep 15 07:07:04 neptune chris: invalid request from client
```

In this message, “neptune” is the host name and “chris” is the user name.

The `logger` utility logs messages with a priority of `user.notice` by default. You can use the `-p` option to specify a different priority. In the example below, we log with a priority of `local4.notice`. The

local4 facility is one of eight (local0 to local7) defined for general use. These can be used to create your own log files. (Though note that RedHat uses local7 to log boot-time information in /var/log/boot.log).

First, we add the following line to /etc/syslog.conf:

```
local4.*          /dev/tty1
```

Restart the syslogd or force it to re-read its' configuration file as follows:



```
#pkill -HUP syslogd
```

Now, messages from the local4 facility will be written to the terminal /dev/tty1:



```
#logger -p local4.notice "This script is writing to /dev/tty1"
```

The following is a partial list of the used files, terms and utilities:

- syslog.conf
- syslogd
- klogd
- logger

108.3 Mail Transfer Agent (MTA) basics

Candidates should be aware of the commonly available MTA programs and be able to perform basic forward and alias configuration on a client host. Other configuration files are not covered.

Key Knowledge Areas

- Create e-mail aliases.
- Configure e-mail forwarding.
- Knowledge of commonly available MTA programs (postfix, sendmail, qmail, exim) (no configuration)

Mail system architecture

Mail systems are built from a number of components as shown in the figure.

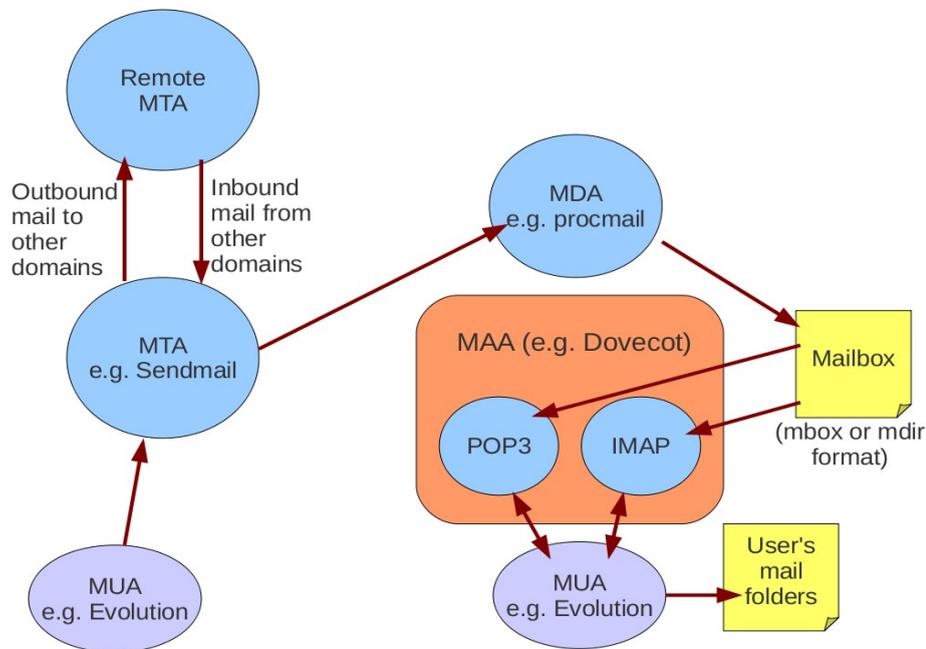


Figure 108.3-1: Mail System Architecture

Mail user agents (MUA) are the programs the end-user interacts with directly to compose or to read email. The MUA usually allows the user to organise mail into folders on his local machine. Examples include Kmail, Evolution, pine, mutt, elm, and many others.

Mail transfer agents (MTA) are the programs that perform the "long haul" of mail delivery. They deliver messages to the mail server for the destination domain. MTAs listen on port 25 and use the SMTP (Simple Mail Transfer Protocol) to transfer messages. Usually, an MTA is configured to accept inbound mail destined for one or more "local" domains and to accept outbound mail from a MUA on the local network. MTAs should not be configured to relay mail (that is, to forward it from

one remote MTA to another) or to accept mail from remote, unauthenticated MUAs. Such "open relays" provide the perfect cover for a spammer to hide behind, and will likely get your server black-listed.

Historically, `sendmail` has been the most popular mail transfer agent and it is still widely used. It has a notoriously complex configuration file. Other MTAs include postfix, exim and qmail.

Mail delivery agents (MDA) handle the local delivery of mail. Messages can be delivered to a person, a mailing list, a file, or even to a program. There are several MDAs in use. For example, Postfix can act as an MDA, delivering mail direct to the users' mailboxes. Other MDAs include procmail and the "deliver" component of Dovecot.

Mail access agents (MAA) deliver messages from the mailbox on the mail server to the user's MUA. Two protocols are in common use: POP3 and IMAP. POP3 is popular with Internet Service Providers; usually messages are downloaded from the mail server to the user's local machine. The messages are then deleted from the server so that long-term storage is the responsibility of the user. IMAP is often used by corporate mail servers. With IMAP, the mail server is typically responsible for the long-term storage of a user's mail. Examples of MAAs include `imapd`, `qpopper` and `dovecot`.

Sendmail

Historically, `sendmail` has been the most widely used MTA. The `sendmail` script which stops or starts the `sendmail` daemon is usually located in the `/etc/rc.d/init.d/` directory.

The main configuration file for `sendmail` is `/etc/mail/sendmail.cf` (or `/etc/sendmail.cf`). Here you can specify the name of the server as well as the names of the hosts from which and to which mail relay is allowed. The file contains a complex series of rules that govern the re-writing of mail addresses and the selection of a delivery mechanism. Some key options include:

sendmail.cf options	
Cw	the mailer hostname. Can also contain a list of hostnames or domain names the mailer will assume but it is better to use Fw for this
Fw	path to the file containing domain names <code>sendmail</code> will receive mail for
Ds	address for 'smart host', this is a mailer that will relay our outgoing mail

For the LPI certification you do not need to know how to write `sendmail.cf` rules. In fact, the rules are usually generated using the `sendmail.m4` or `sendmail.mc` macro file to produce a `sendmail.cf` file by running the command:

```
m4 sendmail.mc > sendmail.cf
```

This process is not part of the LPI objectives.

Sendmail uses the file `/etc/mail/local-host-names` to determine which mail domains it should receive mail for. This file must list all the machines and domain names for which this

[ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification] Created during the initiative "ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software in Africa", a programme of FOSSFA and GIZ. For more information see www.ict-innovation.fossfa.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.

server will handle (incoming) mail. If you change the file you must send sendmail a HUP signal for it to notice the change.

Sendmail usually runs as a daemon listening on port 25 for SMTP messages. It can also be invoked from the command line to inject mail messages or to examine and administer the mail queues. Alternative MTAs such as postfix emulate the sendmail behaviour by providing a compatible command-line interface. This includes the commands `mailq`, `newaliases`, and the `sendmail` command itself.

Aliases and mail forwarding

Aliases allow mail to be re-routed either by the system administrator or by individual users. Aliases let you define mailing lists or allow users to be referred to by more than one name.

The file `/etc/aliases` defines system-wide aliases. (Actually, the path name to this file is specified by the "AliasFile" parameter in `sendmail.cf` so it might be different on your system. `/etc/mail/aliases` is a common alternative location.) Each line in this file contains two fields as follows:

```
alias: address_1,address_2,address_3, ...
```

The first field is the name to alias. The aliased addresses can be:

- A local user name (example: `benjamin`)
- A local file name (example: `/tmp/maildrop`)
- A command (example: `| someprogram`)

A simple use of aliases is to define an alternative mail name for a user. For example if the user Fred Flintstone has a Linux account called `fred`, a line in `/etc/aliases` such as:

```
flintstone: fred
```

will allow `fred` to receive mail with the name `flintstone`.

You can even redirect mail to a remote domain with an alias like this:

```
flintstone: fred@somewhere.else.com
```

(but be aware that if there is a local account called `flintstone`, this line would prevent him from receiving any email!)

command is used to examine and modify the parameters defined in this file. Postfix has a good reputation for security and is relatively straightforward to configure. After `sendmail` it is probably the most popular MTA and is used by default in Ubuntu.

Mail delivery and DNS

MTAs need to find the mail servers for the domains they are trying to deliver mail to. They do this by querying DNS for an MX (mail exchanger) record for the domain they are trying to reach. For example if an MTA is delivering mail to `fred@example.com` it will query DNS for an MX record for the `example.com` domain. This record indicates the name of the domain's mail server. If you are configuring a mail server, you will need to add (or ask your ISP to add) the DNS MX record for your domain to point to your machine.

Finally in order to use a domain name as a valid email address an MX record needs to be added on an authoritative name server for your domain (usually your ISP).

You will also need to add the domain name to `/etc/mail/local-host-names`

The following is a partial list of the used files, terms and utilities:

- `~/ .forward`
- `sendmail` emulation layer commands
- `newaliases`
- `mail`
- `qmail`
- `mailq`
- `postfix`
- `sendmail`
- `exim`

108.4 Managing Printers and Printing

Candidates should be able to manage printer queues and user print jobs using CUPS and the LPD compatibility interface

Key Knowledge Areas

- Basic CUPS configuration (for local and remote printers).
- Manage user print queues.
- Troubleshoot general printing problems.
- Add and remove jobs from configured printer queues.

Introducing CUPS

CUPS (Common Unix Print System) is the standard print system on Linux. The cups server is cupsd; it listens on port 631 and accepts print jobs using IPP (the Internet Printing Protocol). IPP is layered over HTTP, in the sense that an IPP request is an HTTP request with a specific type of content.

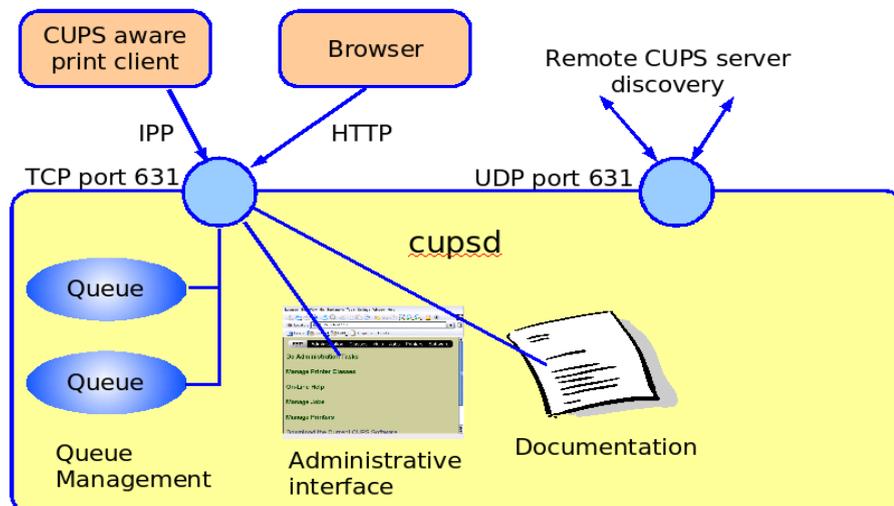


Illustration 108.4-1: CUPS Printing System

The CUPS server handles printer queue management and also provides a web server (also on port 631) that supports a browser-based configuration interface allowing the addition and deletion of printers, printer queue management, and so on.

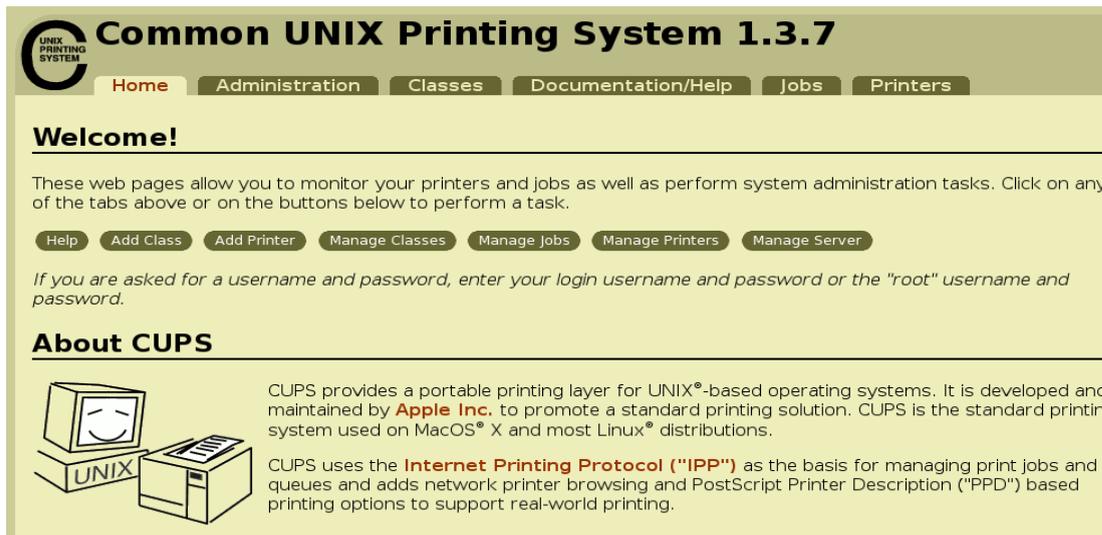


Figure 108.4-2: CUPS Web Interface: Home Page

CUPS filters

The internal work-flow within cups can be relatively complex:

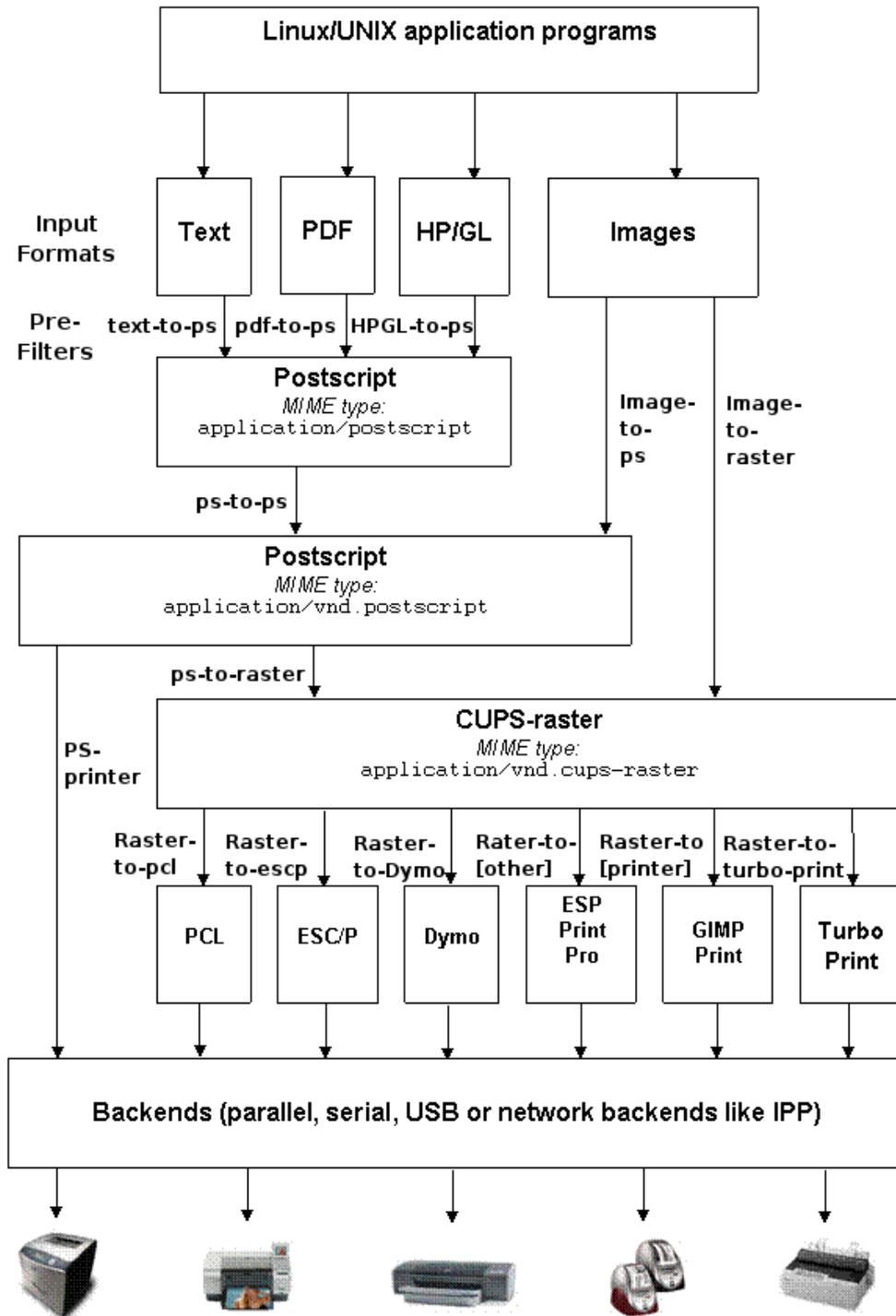


Figure 108.4-3: Linux Printing Architecture

The front-end of CUPS is the actual server that handles the queuing of print jobs and provides the web-based configuration interface. Beyond that, at the heart of the work-flow are the *filters*, which provide format conversion from the initial input file (plain text, jpeg image, etc) to the language understood by the actual printer. These filters are based on *Ghostscript* (a GNU project). They

[ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification] Created during the initiative "ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software in Africa", a programme of FOSSFA and GIZ. For more information see www.ict-innovation.fossfa.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.

consult PPD (Postscript Printer Description) files which specify the printer's capabilities. The filters are found in (for example) `/usr/share/ghostscript`.

Ghostscript can also be invoked from the command line with the name `gs`. This command takes postscript or PDF files as input and generates an output file for a specific printer type. It has a database of printer drivers it can handle (this list is reasonably up to date, for example many USB printers are supported) and converts the postscript directly into PCL for these known models. Run the command `gs -h` to see a list of supported devices:



```
# gs -h
GPL Ghostscript 8.71 (2010-02-10)
Copyright (C) 2010 Artifex Software, Inc. All rights reserved.
Usage: gs [switches] [file1.ps file2.ps ...]
Most frequently used switches: (you can use # in place of =)
-dNOPAUSE          no pause after page   | -q          `quiet', fewer
messages
-g<width>x<height> page size in pixels   | -r<res>    pixels/inch
resolution
-sDEVICE=<devname> select device         | -dBATCHEX  exit after last
file
-sOutputFile=<file> select output file: - for stdout, |command for
pipe,
                                     embed %d or %ld for page #
Input formats: PostScript PostScriptLevel1 PostScriptLevel2
PostScriptLevel3 PDF
Default output device: x11alpha
Available devices:
  alc1900 alc2000 alc4000 alc4100 alc8500 alc8600 alc9100 ap3250
appledmp
  atx23 atx24 atx38 bbox bit bitcmyk bitrgb bitrgbtags bj10e bj10v
bj10vh
  bj200 bjc600 bjc800 bjc880j bjccmyk bjccolor bjcgray bjcmmono bmp16
bmp16m
  bmp256 bmp32b bmpgray bmpmono bmpsep1 bmpsep8 ccr cdeskjet cdj1600
cdj500
  ... many lines deleted ...
```

CUPS back-ends

CUPS supports a variety of back-ends. The term back-end refers to the technology or protocol used to connect to the printer. The table below gives some examples:

Table: CUPS Back-end Technologies

Back-end	URI Syntax	Example
Parallel port	parallel:/dev/lpx	parallel:/dev/lp0
USB	usb://make/model? serial=number	usb://hp/officejet123? serial=108442
ipp	ipp://host/printers/queue	ipp://neptune/printers/ xerox1
LPD	lpd://host/queue	lpd://neptune/xerox1
socket	socket://host:port	socket://neptune:9100
CIFS	smb://user:password@workgrou p/host/share	

CUPS printers can be added using the web-based interface. The figure below shows two of several screens in this sequence:

Figure 108.4-4: Adding a Printer - I

Figure 108.4-5: Adding a Printer - II

Printers can also be assigned to classes. This is mainly useful on heavy duty print servers that have

several printers of the same type connected. Users send print jobs to a class; CUPS will print them on the first printer that becomes available.

Legacy commands for printing

CUPS replaces two earlier printing systems used in UNIX and Linux. One comes from System V Unix, and the other comes from BSD (Berkeley Software Distribution). These two systems were principally intended for printing of plain text files from the command line (and not, for example, printing of office documents with embedded images from a graphical application such as Open Office). The two systems, whilst conceptually similar, used different commands, different configuration files, and a different protocol to talk to the print server.

For backward compatibility, CUPS provides "work-alike" versions of many of these command-line tools. These include:

Command	Description
lp, lpr	submit files for printing
lpq	print printer queue status
lprm	remove a queued print job
accept	Allow jobs to be sent to a print queue
reject	Prevent jobs from being sent to a print queue
cupsenable	Allow jobs to be sent from a queue to a printer
cupsdisable	Stop jobs being sent from the queue to the printer
lpstat	Show the status of the CUPS printer queues
lpadmin	Configure cups printers and classes

When using CUPS, these command submit jobs to the CUPS server which performs its usual filtering and back-end processing.

lp and lpr

The lpr and lp utilities are used to submit jobs to a printer. Note that they use different command options. For example lpr uses the -P flag to specify the printer, lp uses the -d flag.

lpq

A user can monitor the status of print queues with lpq. In the example below, demo-1 is the name of the printer.

```


# lpr -Pdemo-1 anaconda-ks.cfg
# lpq -Pdemo-1
demo-1 is not ready
Rank   Owner   Job           File(s)                               Total Size

```

1st	root	4	install.log	39936 bytes
2nd	root	5	anaconda-ks.cfg	1024 bytes

lprm

The `lprm` command is used to delete jobs from the print queue. In the example below we delete job 4 from the `demo-1` print queue then redisplay the queue:



```
# lprm -Pdemo-1 4
# lpq -Pdemo-1
demo-1 is not ready
Rank   Owner   Job    File(s)                Total Size
1st    root    5      anaconda-ks.cfg        1024 bytes
```

lpstat

`lpstat` displays status information about the current classes, jobs, and printers. When run with no arguments, `lpstat` will list jobs queued by the current user.

Example:



```
# lpstat -t
scheduler is running
no system default destination
device for demo-1: parallel:/dev/lp0
device for demo-2: ipp://printhost/ipp
demo-1 accepting requests since Sun 26 Sep 2010 12:38:10 PM BST
demo-2 accepting requests since Sun 26 Sep 2010 12:34:53 PM BST
printer demo-1 disabled since Sun 26 Sep 2010 12:38:10 PM BST -
    Paused
printer demo-2 is idle.  enabled since Sun 26 Sep 2010 12:34:53 PM BST
demo-1-5                root            1024    Sun 26 Sep 2010 12:39:30 PM BST

# lpstat
demo-1-5                root            1024    Sun 26 Sep 2010 12:39:30 PM BST
```

cupsenable and cupsdisable

These commands are used to enable and disable specified printer queues. `cupsdisable` stops jobs being taken from the queue and sent to the printer. It does not prevent new jobs being sent to the queue. `cupsenable` re-starts the servicing of the queue.

accept and reject

These commands are used to enable and disable the filling of specified printer queues. In the example below we use reject to disable the demo-1 queue, then attempt to print a file to it. We then re-enable the printer, and successfully send a job to the queue:



```
# reject demo-1
# lpr -Pdemo-1 scsconfig.log
lpr: Destination "demo-1" is not accepting jobs.

# accept demo-1
# lpr -Pdemo-1 scsconfig.log
```

lpadmin

The lpadmin command performs a variety of administrative operations on CUPS printers. For example it can be used to set the default printer. In the command sequence shown below there is initially do default destination set. After running lpadmin, we can print to the default printer by using lpr without specifying a printer name:



```
# lpr install.log
lpr: Error - no default destination available.

# lpadmin -d demo-1
# lpr install.log
```

lpadmin can also be used to delete CUPS printers. In the example below we first list the available queues, then use lpadmin to remove one of them, then list the queues again:



```
# lpstat -a
demo-1 accepting requests since Sun 26 Sep 2010 12:38:10 PM BST
demo-2 accepting requests since Sun 26 Sep 2010 12:34:53 PM BST

# lpadmin -x demo-1

# lpstat -a
demo-2 accepting requests since Sun 26 Sep 2010 12:34:53 PM BST
```

The following is a partial list of the used files, terms and utilities:

- CUPS configuration files, tools and utilities
- /etc/cups
- lpd legacy interface (lpr, lprm, lpq)

Topic 109: Networking Fundamentals

109.1 Fundamentals of Internet Protocols

Candidates should demonstrate a proper understanding of TCP/IP network fundamentals.

Key Knowledge Areas

- Demonstrate an understanding network masks.
- Knowledge of the differences between private and public "dotted quad" IP-Addresses.
- Setting a default route.
- Knowledge about common TCP and UDP ports (20, 21, 22, 23, 25, 53, 80, 110, 119, 139, 143, 161, 443, 465, 993, 995).
- Knowledge about the differences and major features of UDP, TCP and ICMP.
- Knowledge of the major differences between IPv4 and IPV6.
- Knowledge of the basic features of IPv6.

IP addresses and the Dotted Quad Notation

Binary numbers

$10 = 2^1$	$100 = 2^2$	$101 = 2^2 + 1$	$111 = 100 + 010 + 001$
------------	-------------	-----------------	-------------------------

This means that a binary number can easily be converted into a decimal as follows:

10000000	=	2^7	=	128
01000000	=	2^6	=	64
00100000	=	2^5	=	32
00010000	=	2^4	=	16
00001000	=	2^3	=	8
00000100	=	2^2	=	4
00000010	=	2^1	=	2
00000001	=	2^0	=	1

The Dotted Quad notation:

Each network interface connected to an IP network is assigned a 32-bit *IP address*. This address is normally written down by breaking it into 4 bytes, writing each byte value in decimal, and separating them with dots. This is called "dotted quad" or "dotted decimal" notation. For example:

Decimal	Binary
192.168.1.1	11000000.10101000.00000001.00000001

Broadcast Address, Network Address and Netmask

An IP address is split into two parts. The top (left-hand) part is the *network address*. It identifies the network that the interface is connected to. It is this part of the IP address that is used to make routing decisions if an IP packet needs to be routed through intervening routers. The bottom (right-hand) part is the *host address*. This identifies a specific machine (host) on the network.

The Netmask

A netmask (also called a subnet mask) is used to define which part of the IP address is used as the network address. Netmasks can also be written using dotted quad notation, for example:

A 16 bit and 17 bit netmask:

255.255.0.0	16-bit	1 1 1 1 1 1 1 1 . 1 1 1 1 1 1 1 1 . 0 0 0 0 0 0 0 0 . 0
255.255.128.0	17-bit	1 1 1 1 1 1 1 1 . 1 1 1 1 1 1 1 1 . 1 0 0 0 0 0 0 0 . 0

If you know the netmask that's in use, you can determine if two IP addresses are on the same network. As an example, consider the IP addresses 32.128.1.1 and 32.128.0.11. In binary these look like this:

00100000	.	10000000	.	00000001	.	00000001
00100000	.	10000000	.	00000000	.	00000011

With a 16-bit netmask (255.255.0.0) these IP addresses are on the same network (because their top 16 bits are identical). So, a packet being sent between these two IP addresses would not need to be routed; it can be sent directly to its destination.

However, with a 24-bit netmask (255.255.255.0) the above two addresses would be on different networks:

00100000	.	10000000	.	00000001	.	00000001
00100000	.	10000000	.	00000000	.	00000011

So, a packet being sent between these two IP addresses would need to be routed; that is, it would need to pass through one or more intervening routers (gateways) to reach the correct network.

The Network Address

Every network has a number which is needed when setting up routing. The network number is a portion of the dotted quad. The host address portion is replaced by zero's.

Typical network address: 192.168.1.0

Network Classes

In the early days of IP networking there were only three places that the division between network part and host part of an IP address could be placed. These were called Class A , Class B and Class C. In detail:

Class A: 8-bit network address and 24-bit host address

The first byte of the IP number is used for the network address. So the default subnet mask would be **255.0.0.0**. The 3 remaining bytes are available to set host interfaces.

Since 255.255.255 and 0.0.0 are invalid host numbers there are $2^{24} - 2 = 16\,777\,214$ possible hosts.

IP numbers have the first byte ranging from **1** to **127**. This corresponds to a binary range of 00000001 to 01111111. The first two bits of a class A address can be set to “**00**” or “**01**”.

Class B: 16-bit network address and 16-bit host address

The two first bytes of the IP number are used for the network address. The default subnet mask is **255.255.0.0**. There are $2^{16} - 2 = 65\,534$ possible hosts.

The first byte ranges from **128** to **191**. Notice that the binary range of the first byte is 10000000 to 10111111. That is the first two bits of a class B address are always set to “**10**”.

Class C: 24-bit network address and 8-bit host address

The three first bytes are used for the network address. The default subnet mask is **255.255.255.0**. There are $2^8 - 2 = 254$ possible hosts.

The first byte ranges from **192** to **223**. This corresponds to a binary range from 11000000 to 11011111. From this we conclude that the first two bits of a class C address is always set to “**11**”.

The Broadcast Address

The broadcast address for a network is obtained by setting the host address to "all ones". For example, the broadcast address for the class C network 192.168.1.0 is 192.168.1.255. The broadcast address for the class B network 150.66.0.0 is 150.66.255.255. Packets sent to the broadcast address will be received by all hosts on the network. Usually, you can only send to the broadcast address of your own network because routers are usually configured to not forward directed broadcast packets. For example a machine on the 192.168.1.0 network cannot use the address 192.168.7.255 to broadcast to the 192.168.7.255 network.

Given an IP address and a netmask, simple logical operations can be applied to calculate the network number and the broadcast address.

To retrieve the network address from an IP number simply AND the IP with the netmask..

Network Address	=	IP	AND	Netmask
-----------------	---	----	-----	---------

Similarly the broadcast address is found with the network address OR 'not MASK'.

Broadcast Address	=	Network	OR	not[Netmask]
-------------------	---	---------	----	--------------

Here AND and OR are logical operations on the binary form of these addresses. 'not' means 'ones complement'

Example:

Take the IP **192.168.3.5** with a net mask **255.255.255.0**. We can do the following operations:

<u>Network address</u>	=	IP	AND	MASK
AND		11000000.10101000.00000011.00000101		(192.168.3.5)
		11111111.11111111.11111111.00000000		(255.255.255.000)
<hr/>				
		11000000.10101000.00000011.00000000		(192.168.3.0)

<u>Broadcast Address</u>	=	IP	OR	NOT-MASK
OR		11000000. 10101000.00000011.00000101		(192.168.3.5)
		00000000.00000000.00000000.11111111		(000.000.000.255)
<hr/>				
		11000000.10101000.00000011.11111111		(192.168.3.255)

Reserved IP addresses

Three IP address blocks (one each for class A, B, and C) are reserved for use on private (internal) networks. These addresses are never used on the Internet. The following table shows the three private address blocks.

Class A	10.x.x.x
Class B	172.16.x.x -- 172.31.x.x
Class C	192.168.0.x

Around the world, large numbers of private internal networks (intranets) re-use these address blocks. Typically, a small corporate network consumes only one "real" IP address (that is, one that can be routed on the Internet). This address is assigned to the external-facing interface of the company's external gateway. This re-use of private addresses has greatly reduced the rate of allocation of IP address space that would otherwise have occurred.

Subnets

In the early days of IP networking, a technique called *subnetting* was used to allow an organisation to divide its allocated IP address block amongst multiple physical networks. For example, consider an organisation that has the class B address block 140.3.0.0. In theory this identifies a single network that can accommodate 65,534 hosts as noted above. This number is much too large for practical use. Sub-netting is used to split the address block into multiple

networks. Essentially, bits allocated in the IP address for hosts are actually used to select the network. The boundary between the network ID and host ID parts of an IP address is determined by the netmask.

For example, our example organisation with the 140.3.0.0 address block might choose to use a 24-bit netmask (255.255.255.0) to split its address block into 256 networks of up to 254 hosts each – a much more practical solution.

As another example, a company with a class C address block 199.48.6.0 might choose to use a 26-bit netmask (255.255.255.192), allowing it to split its address space into 4 networks of up to 62 hosts each.

CIDR notation

From 1993 the internet moved away from Class A, B, and C addressing to use Classless Inter-domain Routing (CIDR). Using CIDR the boundary between network and host address can be placed in any bit position. Using CIDR notation, a network address is written down in dotted-quad form followed by the number of bits that define the network ID. For example, the following notations are equivalent:

10.0.0.0/9

network 10.0.0.0, netmask 255.128.0.0

We will take the example of the class C address block **192.168.1.0**. We investigate a 25-bit then a 26-bit network.

25-bit network

Netmask: 11111111.11111111.11111111.**10000000** or 255.255.255.128

Since Network = IP AND Netmask, we see from the netmask that two network addresses can be formed depending on the hosts range:

1. Host addresses in the 192.168.1.**0xxxxxxx** range result in a 192.168.1.**0** network. In CIDR notation this would be written as 192.168.1.0/25
2. Host addresses in the 192.168.1.**1xxxxxxx** range result in a 192.168.1.**128** network. In CIDR notation this would be written as 192.168.1.128/26

In both cases substitution of the x's by zeros or ones yield the network number and the broadcast address:

Network address	Substitute with 0's	Substitute with 1's
0	Network: 0	Broadcast: 127
128	Network: 128	Broadcast: 255

We can also determine the maximum number of hosts on each network. Since the host address is 7-bit long and we exclude 2 values (all 1's and all 0's) we have $2^7 - 2 = 126$ hosts on each network or a total of 252 hosts.

Notice that if the default subnet mask 255.255.255.0 is used we have 254 available host addresses. In the above example 192.168.1.127 and 192.168.1.128 are taken for the first broadcast and second network respectively, this is why only 252 host addresses can be used.

26-bit network

Netmask: 11111111.11111111.11111111.**11000000** or 255.255.255.192

Here again depending on the host's address 4 different network addresses can be determined with the AND rule.

1. Host addresses in the 192.168.1.**00xxxxxx** range result in a 192.168.1.**0** network.
2. Host addresses in the 192.168.1.**01xxxxxx** range result in a 192.168.1.**64** network.
3. Host addresses in the 192.168.1.**10xxxxxx** range result in a 192.168.1.**128** network.
4. Host addresses in the 192.168.1.**11xxxxxx** range result in a 192.168.1.**192** network.

Substituting the x's with 1's in the numbers above give us the corresponding broadcast addresses: 192.168.1.63, 192.168.1.127, 192.168.1.191, 192.168.1.255

Each subnet has $2^6 - 2 = 62$ possible hosts or a total of 248.

Routing

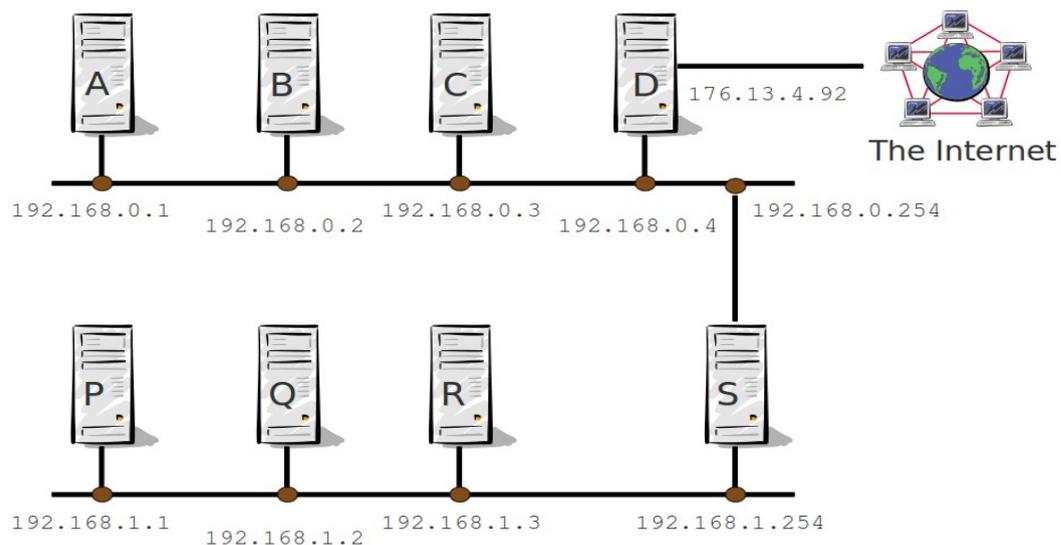


Figure 109.1-1: Sample Network

Part of the task of the Internet Protocol (IP) is to route packets to the correct machine. On each machine Linux maintains a *routing table*. For example, machine B in the picture above would contain an entry that says, in effect, "to get to network 192.168.1.0, go via the router at 192.167.0.254" In topic 109.2 we will see how to configure these routes. This machine would also contain a routing entry for the *default route*; this tells it where to send packets to reach other

networks for which it does not have a more specific route. For machine B the default route would be 192.168.0.4. The default route is often called the *default gateway*. For many so-called stub networks, which have only one gateway connected, the default route is all that's needed. For example machines P, Q and R simply need a default route, pointing to 192.168.1.254

IPv6 Basics

IPv6 is an Internet-layer protocol that provides end-to-end datagram transmission across multiple IP networks. IPv6 comes with more features not present in IPv4. It is described in Internet standard document RFC 2460, published in December 1998. IPv6 prides itself as a protocol that supports many dynamic plug and play functionalities. As a result, there are multiple ways to assign IP addresses in IPv6. This guide will go over two methods of IPv6 address assignment: stateless and stateful.

IPv6 Address Allocation

Internet Assigned Numbers Authority (IANA) assigned Regional Internet Registrars 23/12 bit blocks . Regional Internet registrars RIR (Afrinic) assign blocks 19/32 to local Internet registrars

Local Internet registries (ISP) assign IPv6 address to end users.

Recommended home users get 46 or 56 bit blocks.- meaning multiple subnets are possible.

There is provision for individuals to apply for own provider independent IPv6 address block with Regional Internet Registrar (RIR)– A **provider-independent address space** is a block of ip addresses assigned by a (RIR) directly to an end-user. The user must contract with an ISP to obtain routing of the address block within the internet.

Goals and benefits of IPv6.

Goals of IPv6	Benefits of IPv6	Benefits of IPv6
Simplify address allocation	Built in multi-casting	Stateless auto configuration
Simplify network administration,	No need to renumber network as number of devices grow e.g from class C to class A.	User can keep session while moving from location to location – e.g. wireless and mobile networking in bus, airplane
Simplify routing	No need to renumber network when location changes	Simplified IPv6 headers means faster processing even though larger than IPv4
Resolve security and mobility issues	No need for NAT	Better route aggregation,
Increase address space	No need for address re-use	Built in multi-casting

IP Address Notation

The primary difference between IPv4 and IPv6 addresses is length. IPv4 addresses are 32 bits long and IPv6 addresses are 128 bits long. This means that an IPv4 address is made up of 32 1s and 0s while an IPv6 address is made up of 128 of them – 128 binary digits. This massive length forces IPv6 addresses to be written using a different notation than IPv4 addresses and thus makes them very easy

to distinguish from IPv4 addresses.

As with IPv4, an IPv6 address serves as an identifier for an interface or group of interfaces. Also like IPv4, IPv6 addresses come in several types, based on how they represent those interfaces. IPv6 has three types of addresses. This post covers all three, plus some special purpose addresses as well.

IPv4 - Dotted quad notation : Addresses written as 4 groups of 3 digit decimal values separated by a .

192.168.253.018
Abbreviation rule

–
Drop leading 0 194.168.253.18

IPV4 DECIMAL: BINARY NOTATION

194.	168.	253.	18
xxxxxxxx	xxxxxxxx	xxxxxxxx	xxxxxxxx
byte	byte	byte	byte

x represents bits either a 1 or 0

IPv6 addresses, as commonly displayed as eight groups of four hexadecimal digits separated by colons, for example 2001:0dbb:fe01:fe01:0000:0000:0000:2000 “case insensitive”.

IPV6 HEX: BINARY NOTATION

2001:	0DBB:	FE01:	FE01:	0DBB:	0000:	0000:	2000
xxxxxxx							
xx:							
2bytes							

x represents bits either a 1 or 0

128 bit address, $2^{128} = 3,402823669 \times 10^{38}$ unique addresses.

Abbreviation rules

- 1.Drop leading zeros in a 16 bit value
e.g. 2001:0dbb:fe01:fe01:0:0:0:2000
2. Replace a group of sequential 0 with a double colon ::
e.g. 2001:0dbb:fe01:fe01::2000

EXAMPLE 2

2001:0db8:0000:0000:8a2e:0000:0000:7334
After rule 1
– 2001:db8:0:0:8a2e:0:0:7334
After rule 2 done only once in any address
– 2001:db8:85a3::8a2e:0:0:7334

All devices that connects to the Internet requires a globally unique addresses. IPv4 uses 32 bits for an IP address that allows about 4 billion unique IP addresses. When IPv4 was started as the

protocol for the Inter-networking, they did not anticipate an explosion in usage or the extent to which online technologies would become popular. An escalating demand for IP addresses was the main driving force behind the development of IPv6.

Some technologies were implemented in IPV4 as a interim measure but still that wasn't enough, some of these technologies includes natting, DHCP and sub-netting. According to estimates, in the wireless domain, more than a billion mobile phones, Personal Digital Assistants (PDA), and other wireless devices will require Internet access, and each device will need its own unique IP address.

IPv6 supports 128-bit address space and theoretically there are 340,282,366,920,938,463,374,607,431,768,211,456 addresses available. With this large address-space scheme, IPv6 has the capability to provide unique addresses to each and every device or node attached to the Internet.

Simplified Header

The **improved routing**, or movement of information from a source to a destination, is more efficient in IPv6 because it incorporates a hierarchical addressing structure and has a simplified header. The large amount of address space allows organizations with large numbers of employees to obtain blocks of contiguous address space. Contiguous address space allows organizations to aggregate addresses under one prefix for identification on the Internet. This structured approach to addressing reduces the amount of information Internet routers must maintain and store and promotes faster routing of data. In addition, as shown in above, IPv6 has a simplified header because of the elimination of six fields from the IPv4 header. The simplified header also contributes to faster routing.

IPv6 Address types

Address Type	Description	Binary Pref	Prefix
Unspecified	Only used during hist initialisaion	0000...0 (128 bits)	::/128
loopback	Address used when when talks to itself	0000...01(128 bits)	::1/128
Link-Local Unicast	Used on a single link or non routed LAN eq to 169.254.0.0/16 in IPV4	1111 1110 10	FE80::/10
Site local	Can not be used on the internet, sim to priv ranges in IPV4	1111 1110 1100	FEC0::/10
Multicast	Identifies multicast grps, used as dest add never as source	1111 1111	FF00::/8
IPV4-Mapped	Used to embed IPV4 in an IPV6 address	::FFFF:192.168.0.1	::FFFF/96

Unique Local(ULAs)	These are reserved for home and private enterprise not public	1111 1100	FC00::/7
Global unicast	All other except those reserved		2000::/3

IPv6 ADDRESS ASSIGNMENT

IPv6 Neighbour Discovery combines and improves upon the functionality found in Address Resolution Protocol ([ARP](#)), Internet Control Message Protocol ([ICMP](#)), Router Discovery and ICMP-Redirects in IPv4, and adds some new features as well.

Neighbour Discovery Services includes four main functions:

- **Router Discovery:**

Routers periodically send out router advertisement messages to announce their presence, advertise prefixes that are on-link, assist in address configuration and share other information about the link (MTU, hop limit, etc.) Router Advertisements support multiple prefixes on the same link. Hosts can learn on-link prefixes from router advertisements or, when the router is configured to withhold them, from redirects as needed.

- **Neighbour Discovery:**

IPv6 nodes communicate their link-layer addresses to each other using neighbor solicitation and neighbor advertisement messages. These messages are also used to detect duplicate addresses and test reachability. Neighbor Discovery has many improvements and new features when compared to the corresponding IPv4 protocols. Some of the most notable differences are:

- Neighbour Discovery moves address resolution to the ICMP layer which makes it much less media dependant than ARP as well as adding the ability to use IP layer security when needed.
- Neighbour Discovery uses link-local addresses. This allows all nodes to maintain their router associations even when the site is renumbered to a new global prefix.
- All Neighbour Discovery messages carry link-layer address information so a single message (or pair of messages) is all that is needed for nodes to resolve the other's addresses; no additional address resolution is needed.

- **Neighbour Unreachability Detection:**

IPv6 nodes rely on positive confirmation of packet delivery. This is accomplished in two ways. First, nodes "listen" for new acknowledgements being returned or similar upper-layer protocol confirmation that packets sent to a neighbour are in fact reaching their destination. When such confirmation is absent, the node sends unicast neighbour solicitation messages to confirm next-hop reachability. Neighbour Unreachability Detection is built in, making packet delivery much more robust in a changing network. Using Neighbour Unreachability Detection, Neighbour Discovery will detect router failures, link failures and most notably partial link failures such as one-way communication.

- **Redirects:**

Very similar to the ICMPv4 redirect feature, the ICMPv6 Redirect message is used by routers to inform on-link hosts of a better next-hop for a given destination. The intent is to allow the router(s) to help hosts make the most efficient local routing decisions possible.

SLAAC

In addition to everything discussed above, Neighbour Discovery also enables address autoconfiguration – namely Stateless Address Autoconfiguration (SLAAC). SLAAC provides plug-and-play IP connectivity in two phases: Phase 1 – Link-Local address assignment and then in Phase 2 –

[ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification] Created during the initiative "ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software in Africa", a programme of FOSSFA and GIZ. For more information see www.ict-innovation.fossf.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.

Global address assignment.

IPv6 still maintains the capability for stateful address assignment through DHCPv6 (and static assignment) but SLAAC provides a very lightweight address configuration method that may be desirable in many circumstances. In the stateful autoconfiguration model, a host obtains the interface addresses as well as other required information such as the configuration information and parameters from a server. The DHCP server maintains a manually administered list of hosts and keeps track of which addresses have been assigned to which hosts. DHCP is necessary at sites where central management of hosts is important.

Phase 1 – Link-Local Address

Phase 1 steps for local connectivity:

1. **Link-Local Address Generation:** Any time that a multicast-capable IPv6-enabled interface is turned up, the node generates a link-local address for that interface. This is done by appending an interface identifier to the link-local prefix (FE80::/10).
2. **Duplicate Address Detection:** Before assigning the new link-local address to its interface, the node verifies that the address is unique. This is accomplished by sending a neighbor solicitation message destined to the new address. If there is a reply than the address is a duplicate and the process stops (requiring operator intervention).
3. **Link-Local Address Assignment:** If the address is unique, the node assigns it to the interface it was generated for.

At this point, the node has IPv6 connectivity to all other nodes on the same link. Only hosts move on to Phase 2; a router's interface addresses must be configured by other means.

Phase 2 – Global Address

And phase 2 steps for global connectivity:

1. **Router Advertisement:** The node sends a router solicitation to prompt all on-link routers to send it router advertisements. When the router is enabled to provide stateless autoconfiguration support, the router advertisement will contain a subnet prefix for use by neighbouring hosts.
2. **Global Address Generation:** Once it receives a subnet prefix from a router, the host generates a global address by appending the interface id to the supplied prefix.
3. **Duplicate Address Detection:** The host again performs duplicate detection, this time for the new global address.
4. **Global Address Assignment:** Assuming that the address is not a duplicate, the host assigns it to the interface.

Use of Stateless Address Auto-Configuration (SLAAC) where a Router Advertisement message announces the on-link prefixes as well as the link-local address of the router. The prefix is then combined with either the node link-layer address to form a EUI-64 address or with a random number to form a privacy extension address..

Better Security

The Internet has functioned for the last three decades with IPv4 as the underlying protocol. However, because of this end-to-end model, IPv4 was designed with almost no security in mind and assumes that the required security will be provided at the end nodes. For example, consider an application such

as email that may require encryption services - under IPv4, it is the responsibility of the email client at the end nodes to provide those services. Today, the Internet faces threats such as Denial of Service Attacks, Malicious code distribution, Man-in-the-middle attacks, Fragmentation attacks and Reconnaissance attacks.

Better Quality of Service

IPv6 provides ways to use features like flow label and the traffic fields for applications to request special handling of certain packets without delay “low latency” throughout the WAN. This enables the support of multi media or real-time applications that requires good degree of consistent throughput, delay, or jitter. These types of applications are known as **multi media** or **real-time** applications. The term often used to describe this is low latency. Streaming audio and video requires low latency through high priority.

Improved Connectivity.

Without Network Address Translation (NAT), true end-to-end connectivity at the IP layer is restored, enabling new and valuable services. Peer-to-peer networks and applications such as multi-player online games, video-conferencing (streaming media), file sharing are easier to create and maintain, and services such as VoIP and Quality of Service (QoS) becomes much better as group of computers can communicate directly with each other without need a central server.

Better Optimization

In a multicast technique a packet is copied from one stage down to another in a hierarchical tree-like structure, instead of sending it from the source directly. This means that there are fewer packets in the network thereby optimizing bandwidth utilization and also reducing the resources required at each network node. This multicast technique is particularly useful when streams of information have to be made available to a wide variety of connected devices and not just one single destination. For example multicast technique is used to relay audio data, video data, news feeds, financial data feeds and so on.

Mobility In-built

Mobility in IPv4 is optional and could be made available through a set of extensions. With IPv6, mobility support is mandatory by the use of Mobile IPv6 (MIPv6). Route optimization is a built-in feature for mobile IPv6. Auto discovery and configuration services allows mobile nodes to work in any location without needing the services of any special router. When a mobile node is not on the home network, it sends information about its present location “care-of-address” to the home agent. Routing of packets to this host occurs like this: if any host or node wants to communicate with this mobile node, it will first send the information packets to the home address. The home agent receives these packets and using a routing table, sends these packets to the care-of-address of the mobile node.

Better Administration

When an existing network is to be expanded or two networks to be merged, or when service providers are changed the network no longer needs to be renumbered. IPv6 provides capabilities so that network renumbering can happen automatically. It simplifies aspects of address assignment stateless address autoconfiguration network renumbering and router announcements when changing network connectivity providers. Network address management in IPv6 will no longer requires manual configuration for IP devices, hosts and routers. Multihoming techniques have been made easy to implement. If simultaneous connections are established to two ISPS, when service goes down from one ISP there is a back-up connection to the Internet. This ensures far greater reliability of services.

Easy Transition

IPv6 uses many design features that made IPv4 so successful. This enable smooth transition from IPv4 to IPv6. IPv6 transition will be a gradual even though it offer many competitive advantages to IPv4. Surely many application will operate faster and soother on the IPv6 platform compared to the current IPv4 on.

Commands to use with IPv6

```
trust@trust-desktop:~$ ifconfig -a
eth0      Link encap:Ethernet  HWaddr 00:a1:b0:01:09:90
          inet addr:192.168.10.19  Bcast:192.168.10.255  Mask:255.255.255.0
          inet6 addr: fe80::2a1:b0ff:fe01:990/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:26710 errors:0 dropped:0 overruns:0 frame:0
          TX packets:19011 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:32377321 (32.3 MB)  TX bytes:2365746 (2.3 MB)
          Interrupt:17 Base address:0xde00

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:141 errors:0 dropped:0 overruns:0 frame:0
          TX packets:141 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:11796 (11.7 KB)  TX bytes:11796 (11.7 KB)

trust@trust-desktop:~$

trust@trust-desktop:~$ netstat -r -6
Kernel IPv6 routing table
Destination                                Next Hop                                Flag Met Ref Use If
fe80::/64                                  ::                                       U    256 0    0 eth0
fe80::/64                                  ::                                       U    256 0    0 virbr0
::/0                                        ::                                       !n   -1 1    3 lo
::1/128                                    ::                                       Un   0 1    46 lo
fe80::2a1:b0ff:fe01:990/128              ::                                       Un   0 1    0 lo
fe80::c8a5:5aff:fe06:7adc/128            ::                                       Un   0 1    0 lo
ff00::/8                                   ::                                       U    256 0    0 eth0
ff00::/8                                   ::                                       U    256 0    0 virbr0
::/0                                        ::                                       !n   -1 1    3 lo

trust@trust-desktop:~$

trust@trust-desktop:~$ route -A inet6
Kernel IPv6 routing table
Destination                                Next Hop                                Flag Met Ref Use If
fe80::/64                                  ::                                       U    256 0    0 eth0
fe80::/64                                  ::                                       U    256 0    0 virbr0
::/0                                        ::                                       !n   -1 1    3 lo
::1/128                                    ::                                       Un   0 1    46 lo
fe80::2a1:b0ff:fe01:990/128              ::                                       Un   0 1    0 lo
fe80::c8a5:5aff:fe06:7adc/128            ::                                       Un   0 1    0 lo
ff00::/8                                   ::                                       U    256 0    0 eth0
ff00::/8                                   ::                                       U    256 0    0 virbr0
::/0                                        ::                                       !n   -1 1    3 lo

trust@trust-desktop:~$

trust@trust-desktop:~$ ping6 ::1
PING ::1(::1) 56 data bytes
```

```
64 bytes from ::1: icmp_seq=1 ttl=64 time=0.022 ms
64 bytes from ::1: icmp_seq=2 ttl=64 time=0.027 ms
64 bytes from ::1: icmp_seq=3 ttl=64 time=0.032 ms
64 bytes from ::1: icmp_seq=4 ttl=64 time=0.029 ms
^C
--- ::1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2998ms
rtt min/avg/max/mdev = 0.022/0.027/0.032/0.006 ms
trust@trust-desktop:
```

```
To see ping6 options $ ping6 -?
trust@trust-desktop:~$ ping6 -?
Usage: ping6 [-LUdfnqrvVaA] [-c count] [-i interval] [-w deadline]
        [-p pattern] [-s packetsize] [-t ttl] [-I interface]
        [-M mtu discovery hint] [-S sndbuf]
        [-F flow label] [-Q traffic class] [hop1 ...] destination
trust@trust-desktop:~$
```

```
trust@trust-desktop:~$ traceroute6 ::1
traceroute to ::1 (::1) from ::1, port 33434, from port 62840, 30 hops max, 60 byte
packets
 1 localhost (::1) 0.004 ms 0.001 ms 0.001 ms
trust@trust-desktop:~$
```

In ubuntu you must have application ndisc6 installed to run tracert6 command.

```
trust@trust-desktop:~$ sudo apt-get install ndisc6

trust@trust-desktop:~$ tracert6 ::1
traceroute to ::1 (::1) from ::1, 30 hops max, 60 byte packets
 1 localhost (::1) 0.004 ms 0.001 ms 0.001 ms
trust@trust-desktop:~$
```

To verify you have Ipv6 running enter command

```
trust@trust-desktop:~$ cat /proc/net/if_inet6
00000000000000000000000000000001 01 80 10 80      lo
fe8000000000000002alb0fffe010990 02 40 20 80      eth0
fe80000000000000c8a55afffe067adc 03 40 20 80      virbr0
trust@trust-desktop:~$
```

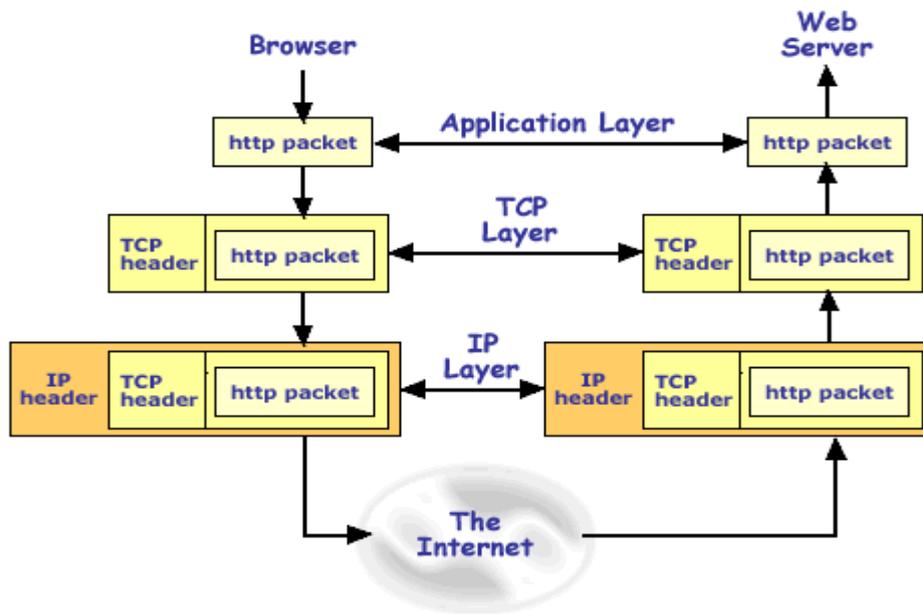


Figure 109.1-2: The 4 Layer TCP/IP Model

The 4 layer TCP/IP model:

Application	application level (FTP, SMTP, SNMP, HTTP, ...)
Transport	Delivery to specific end-points (TCP, UDP)
Internet	Packet delivery and routing (IP, ICMP, IGMP, ARP)
Network Access	network cards, e.g Ethernet, token ring ...

The table below shows some of the key protocols in the TCP/IP suite

Protocol	Purpose
Network Layer	
IP	Internet Protocol. Responsible for machine-to-machine packet delivery (including routing) around the Internet
Transport Layer	
UDP	User Datagram Protocol. Provides connectionless, non-guaranteed, end-to-end delivery
TCP	Transmission Control Protocol. Provides connection-oriented, guaranteed, correctly sequenced, end-to-end delivery
ICMP	Internet Control Message Protocol. Provides a range of low-level services concerned with controlling the IP layer
Application Layer	
SMTP (Port 25)	Simple Mail Transfer Protocol, delivers mail to a mail server
DNS (Port 53)	Domain Name System. Provides machine name to IP address translation, lookup of mail servers, etc
FTP (Ports 20, 21)	File Transfer Protocol. Provides capability to transfer files between computers
POP3 (Port 110)	Post Office Protocol. Allows download of mail from a mail server
IMAP (Port 143)	Internet Message Access Protocol. Allows access to mail messages that are stored on a mail server
HTTP (Port 80)	Hyper Text Transfer Protocol. This is the protocol that web browsers use to request pages from web servers
SNMP (Port 161)	Simple Network Management Protocol. Provides the capability to monitor the status of network-attached devices
NNTP (Port 119)	Network News Transfer Protocol. Used for reading and posting Usenet articles.

TCP/IP Services and Ports

At the application layer, each service listens for connections on a specific *port number*. This is a 16-bit number that identifies a specific transport endpoint in the machine. The transport-layer protocols (TCP and UDP) are responsible for delivering a packet to the correct port.

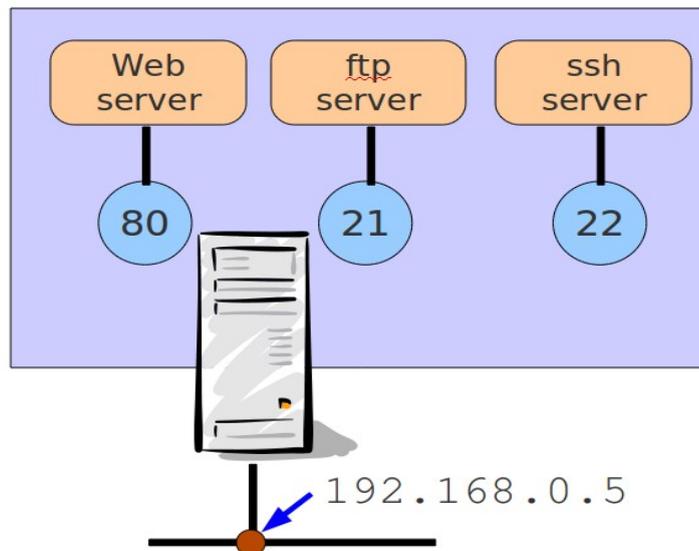


Figure 109.1-3: Services on Host 192.168.0.5

The list of known services and their ports is found in `/etc/services`. The official list of services and associated ports is managed by the IANA (Internet Assigned Numbers Authority).

Since the port number is 16 bits there are 65535 available numbers. Numbers from 1 to 1023 are known as *privileged* ports and are reserved for services run by root. Most standard services use port numbers in this range.

The `/etc/services` main ports:

```
ftp-data      20/tcp
ftp           21/tcp
ssh           22/udp
ssh           22/tcp
telnet        23/tcp
smtp          25/tcp      mail
domain        53/tcp
domain        53/udp
http          80/tcp      # www is used by some broken
pop-3         110/tcp     # PostOffice V.3
sunrpc        111/tcp
sftp          115/tcp
uucp-path     117/tcp
nntp          119/tcp     usenet      # Network News Transfer
ntp           123/tcp     # Network Time Protocol
netbios-ns    137/tcp     nbns
netbios-ns    137/udp     nbns
netbios-dgm   138/tcp     nbdgm
netbios-dgm   138/udp     nbdgm
netbios-ssn   139/tcp     nbssn
imap          143/tcp     # imap network mail protocol
NeWS          144/tcp     news        # Window System
```

snmp	161/udp
snmp-trap	162/udp

The following is a partial list of the used files, terms and utilities:

- | | |
|----------------|-------------|
| •/etc/services | •ping |
| •ftp | •dig |
| •telnet | •traceroute |
| •host | •tracert |

109.2 Basic Network Configuration

Candidates should be able to view, change and verify configuration settings on client hosts

Key Knowledge Areas

- Manually and automatically configure network interfaces
- Basic TCP/IP host configuration.

The Network Interface

The network interface card (NIC) must be supported by the kernel. You may be able to determine which card you are using (and whether your kernel recognises it) by examining the output from `dmesg`, which shows the content of the kernel's message buffer:

Example:



```
$ dmesg | grep eth0
e1000: eth0: e1000_probe: Intel(R) PRO/1000 Network Connection
e1000: eth0: e1000_watchdog_task: NIC Link is Up 1000 Mbps Full
Duplex, Flow Control: None
eth0: no IPv6 routers present
```

You can probe the hardware directly with `lspci`:



```
$ lspci | grep -i ethernet
09:00.0 Ethernet controller: Marvell Technology Group Ltd. 88E8040
PCI-E Fast Ethernet Controller (rev 12)
```

`lspci -v` gives more detail. Here is the relevant part of the output:



```
09:00.0 Ethernet controller: Marvell Technology Group Ltd. 88E8040
PCI-E Fast Ethernet Controller (rev 12)
Subsystem: Dell Device 022e
Flags: bus master, fast devsel, latency 0, IRQ 28
Memory at f9ffc000 (64-bit, non-prefetchable) [size=16K]
I/O ports at de00 [size=256]
Capabilities: <access denied>
Kernel driver in use: sky2
Kernel modules: sky2
```

From the example above we see the Ethernet controller type, the memory and I/O address assignments, the IRQ in use, and the relevant kernel driver module (sky2 in this example). This information can be useful either if the wrong module is being used or if the resources (I/O or IRQ) are not available.

Occasionally, this information can either be used to insert a module with a different i/o address (using the `modprobe` or `insmod` utilities) or can be saved in `/etc/modules.conf`. This file is consulted by `modprobe`; placing entries here will save the settings for the next system boot. However, the kernel will normally auto-detect the card and load the correct module.

Name resolution

A machine must be able to perform host name resolution; that is, to translate a machine name such as `neptune.example.com` to an IP address such as `144.11.20.101`.

The file `/etc/hosts` is one way to perform name resolution. At minimum you will see an entry in this file for the loopback address (traditionally called `localhost`). The file may also contain entries for other machines, but usually only machines in the local network:

```
# Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1      m1530-rhel localhost localhost.localdomain
::1           localhost6
# other hosts
192.168.1.108 mesa mesa.domain.org
192.168.1.119 pico
```

(Note also the IPV6 entry for `localhost6`)

Name resolution can also be performed by consulting DNS. There's more detail on this in topic 109.4: Configure client-side DNS.

The file `/etc/nsswitch.conf` is used to tell the resolvers in the standard C library where to look for their information. A resolver is a routine that looks up something (like a user name or a host name) and returns some corresponding information (such as the UID or the IP address). The line in `nsswitch.conf` that controls host name resolution looks something like this:

```
hosts:      files  dns
```

This entry tells the resolver to first look at the local file (`/etc/hosts`) then to consult DNS. Other sources for host name lookups that can be specified here include `nis` and `nisplus`.

Network configuration

The kernel's current idea of the machine's host name is reported by the `hostname` command:

```

# hostname
```

```
m1530-rhel.example.com
```

The machine's host name is set in various files. On some distributions (e.g. Ubuntu) the file `/etc/hostname` is used. Others (e.g. RedHat) set it in `/etc/sysconfig/network`. The following command might help you to find out where the host name is defined on your distribution. This example is on RedHat:



```
# find /etc -type f -exec grep -l $(hostname) {} \;
/etc/sysconfig/network-scripts/ifcfg-eth0
/etc/sysconfig/network
/etc/hosts
/etc/dhclient-eth0.conf
```

The file `/etc/sysconfig/network` defines if networking must be started. On a RedHat system it also defines the host name and the default gateway. These are settings that apply to the machine as a whole (i.e. they are not specific to one network interface):

```
NETWORKING=yes
HOSTNAME=mesa.domain.org
GATEWAY=192.168.1.254
```

On a RedHat system the file `/etc/sysconfig/network-scripts/ifcfg-eth0` contains the configuration parameters for `eth0`. This example shows how the file might look for an interface with a statically-assigned IP address:

```
DEVICE=eth0
BOOTPROTO=static
BROADCAST=192.168.1.255
IPADDR=192.168.1.108
NETMASK=255.255.255.0
ONBOOT=yes
```

For a card that's configured using DHCP the file would be simpler:

```
DEVICE=eth0
BOOTPROTO=dhcp
ONBOOT=yes
```

Stop and Start Networking

- From the command line

The main tool used to bring up the network interface is `/sbin/ifconfig`. Once initialised, the kernel module aliased to `eth0` in `/etc/modules.conf` (e.g. `tulip.o`) is loaded and assigned an IP and netmask value. As a result the interface can be stopped and restarted without losing this information as long as the kernel module is inserted.

ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification] Created during the initiative "ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software in Africa", a programme of FOSSFA and GIZ. For more information see www.ict-innovation.fossfa.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.

Examples: Using ifconfig.

```
/sbin/ifconfig eth0 192.168.10.1 netmask 255.255.128.0
/sbin/ifconfig eth0 down
/sbin/ifconfig eth0 up
```

Running ifconfig with no parameters will show the current settings:



```
# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:0C:29:76:13:68
          inet addr:192.168.81.130  Bcast:192.168.81.255
Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe76:1368/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:5623 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8037 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:418797 (408.9 KiB)  TX bytes:1338985 (1.2 MiB)
          Base address:0x2000 Memory:d8920000-d8940000
```

Another tool is /sbin/ifup. This utility reads the system's configuration files in /etc/sysconfig/ and assigns the stored values for a given interface. The script for eth0 is called ifcfg-eth0 and has to be configured. If a boot protocol such as DHCP is defined then ifup will start the interface with that protocol.

Examples: Using ifup.



```
/sbin/ifup eth0
/sbin/ifup ppp0
/sbin/ifdown eth0
```

RedHat has no man pages for ifup and ifdown (Ubuntu does!) but on RedHat they are shell scripts so you can examine the code if you wish.

- Using the network script

At boot time the network interfaces are initialised with the /etc/rc.d/init.d/network script. All the relevant networking files are sourced in the /etc/sysconfig/ directory. This script can also be run manually.

In addition the script also reads the sysctl options in /etc/sysctl.conf, this is where you can configure the system as a router (allow IP forwarding in the kernel). For example the line:

```
net.ipv4.ip_forward = 1
```

will enable ip forwarding. You would need to do this if the machine is being used as a gateway /

router.

Using this script, networking can be started, stopped or restarted like this:

```
# /etc/rc.d/init.d/network start
# /etc/rc.d/init.d/network stop
# /etc/rc.d/init.d/network restart
```

The last command is useful if you have changed any network settings under `/etc/sysconfig`. Note however that this command will stop and restart all network interfaces, whereas `ifconfig`, `ifup` and `ifdown` can be used to start and stop individual interfaces.

RedHat also provides a "wrapper" program called `service` that provides a slightly easier way to run these scripts, for example:



```
# service network restart
```

Routing

A noticeable difference when using a system script such as `ifup` rather than `ifconfig` on its own, is that the system's routing tables are set in one case and not in the other.

This is because either the `/etc/sysconfig/network` file is read, where a default gateway is stored, or the DHCP server has sent this information together with the IP number.

The routing tables are configured, checked and changed with the `/sbin/route` tool.

Routing examples:

Add a static route to the network 10.0.0.0 through the device `eth1` and use 192.168.1.108 as the gateway for that network:



```
#!/sbin/route add -net 10.0.0.0 gw 192.168.1.108 dev eth1
```

Add a default gateway:



```
/sbin/route add default gw 192.168.1.1 eth0
```

Routes added in this way are only "for here and now" – they will not survive a reboot.

Listing the kernel routing table:

```


/sbin/route -n
▶ Kernel IP routing table
  Destination      Gateway            Genmask           Iface
  192.168.1.0      0.0.0.0           255.255.255.0    eth0
  10.1.8.0         192.168.1.108    255.0.0.0        eth1
  127.0.0.0        0.0.0.0           255.0.0.0        lo
  0.0.0.0          192.168.1.254    0.0.0.0          eth0

```

Here, the first entry defines the directly connected network for eth0. It says that no gateway is needed to reach the network 192.168.1.0. The second entry defines a specific route to the network 10.1.8.0. The last entry defines the default route. Since this rule has a Genmask of 0.0.0.0 and a Destination of 0.0.0.0 it always matches; however it will only be used if a more specific route (one with a longer Genmask) is not found.

If you belong to the 192.168.10.0 network and you add a route to the 192.168.1.0 network you may find that machines in the latter network are not responding. This may be because no route has been set from the 192.168.1.0 network back to your host!! This problem is solved with dynamic routing, using the daemons `gated` and `routed` to dynamically update routing tables across a network. However, dynamic routing is not necessary on "stub networks" (where most machines are found) and it is not part of the LPI level 1 objectives.

Permanent Static Routes

Permanent static routes can be defined in the files under `/etc/sysconfig`. Depending on the version, RedHat takes interface-specific routes from `/etc/sysconfig/network-scripts/route-eth0` (for eth0), and non-interface-specific routes from `/etc/sysconfig/static-routes`. These routes will be added at boot time by the network script.

Naming Networks

Using the `/etc/networks` file it is possible to assign names to network numbers. A typical entry might look like this:

```
office    192.168.3.0
```

It is then possible to use network names with tools like `route`, for example:

```


route add -net office.org netmask 255.0.0.0

```

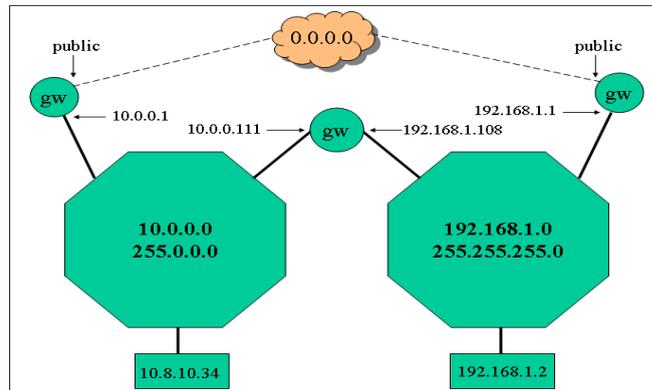


Figure 109.2-1: A Routing Scenario:

The following is a partial list of the used files, terms and utilities:

- /etc/hostname
- /etc/hosts
- /etc/resolv.conf
- /etc/nsswitch.conf
- ifconfig
- ifup
- ifdown
- route
- ping

109.3 Basic Network Troubleshooting

Candidates should be able to troubleshoot networking issues on client hosts

Key Knowledge Areas

- Manually and automatically configure network interfaces and routing tables to include adding, starting, stopping, restarting, deleting or reconfiguring network interfaces.
- Change, view, or configure the routing table and correct an improperly set default route manually.
- Debug problems associated with the network configuration.

The figure summarises various points at which networking can fail, and may serve as a basis for fault-finding.

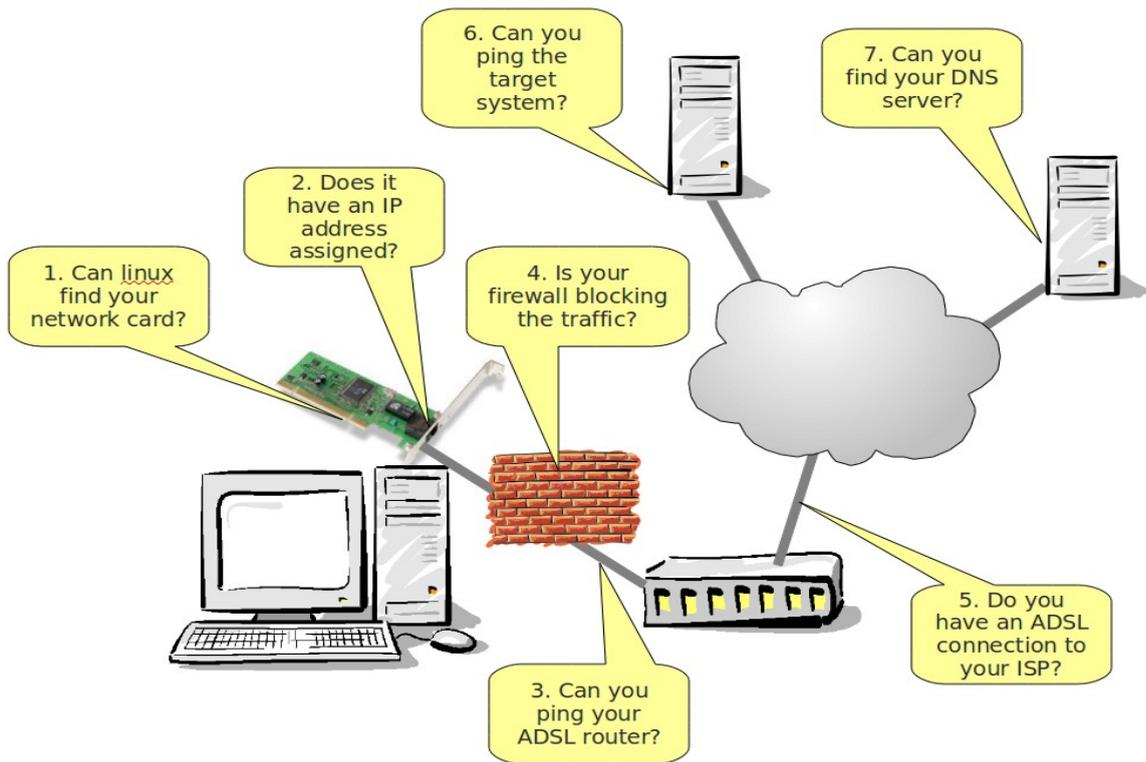


Figure 109.3-1: Potential Points of Failure in a Network

Can Linux find your network card?

The network interface card (NIC) must be supported by the kernel. Topic 109.2 provides some suggestions to help you determine if the kernel has detected your network card and loaded an appropriate driver.

Does it have an IP address assigned?

Use the command `ifconfig` to determine the network settings of your card:

Example:

```

# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:0C:29:76:13:68
          inet addr:192.168.81.130  Bcast:192.168.81.255
Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe76:1368/64  Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:6240 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8402 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:475800 (464.6 KiB)  TX bytes:1392930 (1.3 MiB)
          Base address:0x2000  Memory:d8920000-d8940000
    
```

In particular verify that the IP address and netmask are correct.

Can you ping other machines on the network?

The `ping` command can be used to test reachability of a machine by sending an ICMP echo request packet and awaiting a reply. A simple test is to ping another machine on the same network. In the example below the `-c` flag is used to limit the number of pings. By default, ping will continue indefinitely at 1 second intervals.



```
# ping -c 4 192.168.81.129
PING 192.168.81.129 (192.168.81.129) 56(84) bytes of data.
64 bytes from 192.168.81.129: icmp_seq=1 ttl=64 time=0.822 ms
64 bytes from 192.168.81.129: icmp_seq=2 ttl=64 time=1.15 ms
64 bytes from 192.168.81.129: icmp_seq=3 ttl=64 time=0.812 ms
64 bytes from 192.168.81.129: icmp_seq=4 ttl=64 time=0.745 ms

--- 192.168.81.129 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3000ms
rtt min/avg/max/mdev = 0.745/0.884/1.157/0.160 ms
```

Does DNS name resolution work?

If you can connect to a machine by specifying its IP address but not by specifying its name you should suspect that name resolution is not working. First, verify that the correct DNS servers are specified in `/etc/resolv.conf`. See topic 109.4 for more detail on DNS client-side configuration.

Verify that you can ping your DNS servers by IP address.

Use the `dig` utility to manually test DNS lookups:



```
# dig www.lpi.org

; <<>> DiG 9.3.4-P1 <<>> www.lpi.org
;; global options: printcmd
;; Got answer:
;; ->HEADER<<- opcode: QUERY, status: NOERROR, id: 24846
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;www.lpi.org.                IN      A

;; ANSWER SECTION:
www.lpi.org.                5      IN      A      24.215.7.162
```

[ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification] Created during the initiative "ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software in Africa", a programme of FOSSFA and GIZ. For more information see www.ict-innovation.fossfa.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.

```
;; Query time: 32 msec
;; SERVER: 192.168.81.2#53(192.168.81.2)
;; WHEN: Thu Sep 16 15:09:10 2010
;; MSG SIZE rcvd: 45
```

Verify that an answer (an A record) was received, and that it came from the DNS server you expect (192.168.81.2 in the example above).

The nslookup command can also be used but gives slightly less information:



```
# nslookup www.lpi.org
Server:          192.168.81.2
Address:         192.168.81.2#53

Non-authoritative answer:
Name: www.lpi.org
Address: 24.215.7.162
```

Are your servers listening?

If other machines cannot connect to your services, verify that the services are running; for example:



```
# service sshd status
sshd (pid 5851 5849 5220) is running...
```

Or you can look for them in the output from ps:



```
# ps -ef | grep sshd
root      5220      1  0 08:41 ?          00:00:00 /usr/sbin/sshd
root      11440 11415  0 15:25 pts/2    00:00:00 grep  sshd
```

Verify that the service is listening on the expected port. The command netstat can be used to examine active ports; for example:



```
$ netstat -ant
```

```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 192.168.122.1:53       0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22            0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:631         0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:23            0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:25          0.0.0.0:*               LISTEN
tcp        0      0 192.168.81.1:60864    192.168.81.130:22      ESTABLISHED
tcp        0      0 192.168.1.78:43011    174.129.193.12:443     ESTABLISHED
tcp6       0      0 :::22                 :::*                    LISTEN
tcp6       0      0 :::1:631              :::*                    LISTEN
```

Here, the first three lines of output represent:

- 1.A local DNS server listening on an internal network 192.168.122.0
- 2.A secure shell server listening on port 22
- 3.A CUPS print server listening on port 631 (but only on the loopback address)

The command `netstat -i` will show network interfaces. It provides similar information to running `ifconfig` with no arguments:



```
$ netstat -i
Kernel Interface table
Iface  MTU Met  RX-OK RX-ERR RX-DRP RX-OVR  TX-OK TX-ERR TX-DRP TX-
OVR Flg
eth0   1500 0     0     0     0 0     0     0     0     0
BMU
lo     16436 0     60    0     0 0     60    0     0     0
LRU
virbr0 1500 0     0     0     0 0     34    0     0     0
BMRU
vmnet1 1500 0     0     0     0 0     34    0     0     0
BMRU
vmnet8 1500 0     8858  0     0 0     6676  0     0     0
BMRU
wlan0  1500 0     66642 0     0 0     20490 0     0     0
```

BMRU

In the output above, you see a wired network interface (`eth0`), a wireless interface (`wlan0`), the loopback interface (`lo`) and three interfaces to support virtualization (`virbr0`, `vmnet1` and `vmnet8`)

Other useful options for the `netstat` command include:

- `-r, --route:` Show the routing table (similar to `route -n`)
- `-t, --tcp:` Show TCP endpoints
- `-u, --udp:` Show UDP endpoints
- `-a, --all:` Show both listening and connected endpoints. By default only connected endpoints are shown
- `-n, --numeric:` Show numeric values instead of trying to determine symbolic names for hosts or ports

You can also examine active TCP and UDP endpoints with the command `lsof -i`:



```
# lsof -i
COMMAND      PID  USER  FD  TYPE  DEVICE  SIZE  NODE  NAME
portmap      4977  rpc   3u  IPv4  9744    UDP  *:sunrpc
portmap      4977  rpc   4u  IPv4  9745    TCP  *:sunrpc (LISTEN)
rpc.statd    5006  root  3u  IPv4  9807    UDP  *:945
rpc.statd    5006  root  6u  IPv4  9794    UDP  *:942
rpc.statd    5006  root  7u  IPv4  9814    TCP  *:948 (LISTEN)
hpiod        5200  root  0u  IPv4  10272   TCP  m1530-rhel.example.com:2208
(LISTEN)
python       5205  root  4u  IPv4  10302   TCP  m1530-rhel.example.com:2207
(LISTEN)
sshd         5220  root  3u  IPv6  10327   TCP  *:ssh (LISTEN)
cupsd        5231  root  3u  IPv4  10362   TCP  m1530-rhel.example.com:ipp
(LISTEN)
cupsd        5231  root  5u  IPv4  10365   UDP  *:ipp
sendmail     5251  root  4u  IPv4  10440   TCP  m1530-rhel.example.com:smtp
(LISTEN)
avahi-daemon 5370  avahi 13u  IPv4  10721   UDP  *:mdns
avahi-daemon 5370  avahi 14u  IPv6  10722   UDP  *:mdns
avahi-daemon 5370  avahi 15u  IPv4  10723   UDP  *:filenet-rpc
avahi-daemon 5370  avahi 16u  IPv6  10724   UDP  *:filenet-nch
dhclient     11325  root  5u  IPv4  17296   UDP  *:bootpc
```

As you can see, the `lsof` output is more informative – it tells you the name and the PID of the process that is using the endpoint.

Is your firewall blocking access?

If other machines are unable to connect to your services, check that your firewall is not blocking access. A quick way to make this check is to briefly disable the firewall and repeat the test. However, do not forget to re-enable it immediately afterwards. From the command line you can flush the firewall rules with the command:



```
# iptables -F
```

Other diagnostic tools

The command `traceroute` can be used to trace the path that a packet takes to a specific destination. For example:



```
# traceroute www.lpi.org
traceroute to www.lpi.org (24.215.7.162), 30 hops max, 40 byte packets
 1  192.168.81.2 (192.168.81.2)  0.326 ms  0.207 ms  0.212 ms
 2  BThomehub.home (192.168.1.254)  81.405 ms  78.430 ms  77.940 ms
 3  217.47.111.122 (217.47.111.122)  20.152 ms  21.611 ms  23.679 ms
 4  217.47.111.161 (217.47.111.161)  23.903 ms  26.248 ms  27.741 ms
 5  213.1.69.38 (213.1.69.38)  29.059 ms  31.459 ms  33.284 ms
 6  213.120.180.197 (213.120.180.197)  33.977 ms  36.439 ms  38.711 ms
 7  213.120.179.26 (213.120.179.26)  41.249 ms  25.029 ms  25.639 ms
 8  213.120.179.178 (213.120.179.178)  25.474 ms  26.261 ms  25.895 ms
 ... lines deleted ...
21  clark.lpi.org (24.215.7.162)  149.154 ms  123.495 ms  123.828 ms
```

The times shown in the output above are the round-trip times of the probes to each gateway. (Each gateway is probed three times.)

The `tracert` utility gives similar information with a slightly different format:



```
# tracert www.lpi.org
 1:  192.168.81.130 (192.168.81.130)           0.161ms  pmtu 1500
 1:  192.168.81.2 (192.168.81.2)             0.379ms
 2:  BThomehub.home (192.168.1.254)          asymm  1  98.572ms
 3:  217.47.111.122 (217.47.111.122)         asymm  1  54.991ms
 4:  217.47.111.161 (217.47.111.161)         asymm  1  49.023ms
 5:  213.1.69.38 (213.1.69.38)               asymm  1  48.824ms
 6:  213.120.180.197 (213.120.180.197)       asymm  1  48.795ms
 7:  213.120.179.26 (213.120.179.26)        asymm  1  48.786ms
 8:  213.120.179.178 (213.120.179.178)      asymm  1  48.361ms
 ... lines deleted ...
21:  clark.lpi.org (24.215.7.162)           asymm  1 278.213ms
```

reached

Usually it is only the first couple of hops in this route that are on your own network. Anything beyond that is likely outside of your administrative control.

The following is a partial list of the used files, terms and utilities:

- ifconfig
- ifup
- ifdown
- route
- host
- hostname
- dig
- netstat
- ping
- traceroute

109.4 Configure client-side DNS

Candidates should be able to configure DNS on a client host

Key Knowledge Areas

- Demonstrate the use of DNS on the local system.
- Modify the order in which name resolution is done.

When host name lookups are performed, two configuration files are consulted to determine where to get the information from. The first is `/etc/nsswitch.conf` (the "name service switch" file). This file tells the resolver (the lookup routine) what data source to consult for the information. It is used to configure several types of lookup including user name, group, and host name lookups. In this topic we are concerned only with host name lookups.

All that the `nsswitch` file really does is to tell the resolver which library to call to do the work. There is a simple mapping between the entries in the file and the library names. For example, given this line in `nsswitch.conf`:

```
hosts: dns nis files
```

the resolver will attempt to use the libraries `libnss_dns`, `libnss_nis` and `libnss_files`, in that order.

Additional notations in the file control what action to take if a particular type of lookup fails. For example, this entry:

```
hosts: dns [NOTFOUND=return] files
```

tells the resolver to consult DNS first. If the resolver was able to perform a DNS lookup but DNS did not find the name, the resolver will immediately return failure. The local file (`/etc/hosts`) will only be consulted in the event that DNS cannot be contacted at all.

Common "database" names in `nsswitch.conf`

Keyword	Description
passwd	user names
group	group names
hosts	host names
networks	network names

Common information sources in `nsswitch.conf`

Keyword	Description
---------	-------------

files	flat files, generally in /etc
nis	a map from a NIS server
dns	a DNS server
ldap	an LDAP server

Sample /etc/nsswitch.conf	
hosts:	files dns
networks:	files nis ldap

In the case that DNS is being used for name resolution, a second file, `/etc/resolv.conf`, is consulted. This specifies the IP addresses of one (or preferably two or three) DNS servers.

Sample /etc/resolv.conf	
search	example.com
nameserver	192.168.1.254
nameserver	24.215.7.126
options	timeout:2

Here, two name servers are specified. The first is, presumably, on the local network (since it's a private IP address). In the case of networks that connect through a broadband modem/router to an ISP, the router itself is probably providing a caching DNS service. The second (to be used if the first is unavailable) might be a name server maintained by your ISP, or any other DNS server you trust.

The `search` directive specifies a default domain. For example if the resolver is looking up a simple name like "neptune" it will append the default domain (so in this example it will look up `neptune.example.com`).

The `options` directive is not often used. It can be used to configure specific resolver settings. The example shown sets the amount of time (in seconds) the resolver will wait for a response from a remote name server before trying the next one. (The default is 5 seconds). For other options, see the man page for `resolv.conf(5)`

Some background on DNS

DNS (Domain Name System) is a distributed hierarchical naming system. A primary use of DNS is to map host names (such as `www.lpi.org`) onto IP addresses (such as `24.215.7.162`).

ict@innovation

Names are organised within a hierarchical structure. At the top of this tree are a number of pre-defined names. Early assignments of top-level domains included the following:

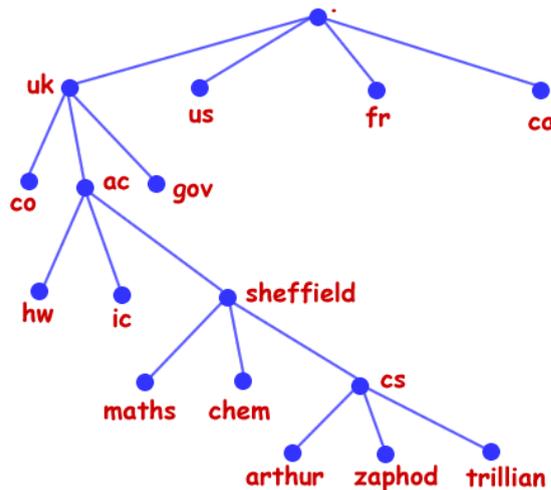
com	Commercial organisations
edu	US educational institutions
gov	US government institutions
mil	US military institutions
net	Gateways and network providers
org	Non commercial sites

Later, two character top-level domains were added, corresponding to sites in individual countries; for example:

uk	United Kingdom
ca	Canada
jp	Japan
ke	Kenya
za	South Africa

For a complete list, see http://en.wikipedia.org/wiki/List_of_Internet_top-level_domains.

The figure below shows a fragment of the DNS name-space for machines in the Computer Science department at Sheffield University, an academic institution in the UK:



Names in DNS are written "little endian", that is, starting from the bottom of the tree and working up. An example from the figure above is `zaphod.cs.sheffield.ac.uk`. Such a name is known as a *fully qualified domain name (FQDN)*.

[ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification] Created during the initiative "ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software in Africa", a programme of FOSSFA and GIZ. For more information see www.ict-innovation.fosssa.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.

DNS servers hold several types of record. These include:

<i>Record types in DNS</i>	
Record type	Description
A	These records map host names onto IPV4 addresses. These are the most commonly used records in DNS
MX	Mail exchanger records specify the name of the mail server for a specified domain. For example the MX record for lpi.org tells us where to send mail for the user jim@lpi.org
PTR	These records map IP addresses back to host names, supporting "reverse DNS lookups". A complete name-space (with a top-level domain at inaddr.arpa) exists to support these lookups
NS	Name server records specify the name server for a given domain. For example the NS records for lpi.org tell us which servers are able to resolve the name www.lpi.org.

The *primary* or *master* DNS server for a domain is the one that is ultimately responsible for resolving queries for records in that domain. On these servers, *zone files* (plain text files) contain the definitions of the records for that domain. These zone files are the raw materials of DNS.

Secondary (or *slave*) DNS servers are used to share the load with the primary servers (and to avoid having a single point of failure in case the primary server goes down). Secondary DNS servers maintain copies of the zone files but these are obtained by synchronising with the primary server. This synchronisation is called a *zone transfer*.

Caching DNS servers do not hold their own zone files. They simply forward DNS queries to other servers but retain (cache) the results they receive so that if a query is repeated it can be answered rapidly from the local cache. Caching DNS servers are easy to set up and even if you run only a small corporate network it is probably worth setting up a caching DNS server to service it.

The following is a partial list of the used files, terms and utilities:

- /etc/hosts
- /etc/resolv.conf
- /etc/nsswitch.conf

Topic 110 Security

110.1 Perform Security Administration Tasks

Candidates should know how to review system configuration to ensure host security in accordance with local security policies.

Key Knowledge Areas

- Audit a system to find files with the suid/sgid bit set.
- Set or change user passwords and password aging information.
- Being able to use nmap and netstat to discover open ports on a system.
- Set up limits on user logins, processes and memory usage.
- Basic sudo configuration and usage.

Prevent Root access at Boot time.

Hardening init

Many systems can (in their default configuration) be booted into single-user mode to obtain a root shell without supplying the root password. This is true, for example, for RHEL5. To prevent this, edit `/etc/inittab` and add the line:

```
ss:S:respawn:/bin/sulogin
```

This line tells the system that to get into state "S" (single-user) it must successfully run the program `/sbin/sulogin`, which asks for the root password. This change will prevent users gaining root access through a simple single-user boot.

Hardening GRUB

There are other boot-time exploits that involve modifying the options passed to the kernel by GRUB. You can prevent these by setting a GRUB password. To do this:

1. Boot Linux and run `grub`
2. At the grub prompt, enter the command `md5crypt`
3. At the password prompt, entered the desired password
4. You will be shown the password hash (beginning "\$1...")
5. Edit the file `/boot/grub/grub.conf` and add a line like this:

```
password --md5 $1$c1Vc1/$R1F6Wm6XF3am1hOcDTCdv.
```

(here you should copy and paste the password hash you got at step 4).

The effect of this line depends on where within `grub.conf` you put it. If you put it within one of

the stanzas that correspond to specific boot menu selections (i.e. after a "title" line) then grub will always ask for the password if you choose that item from the boot menu. However, if you put it outside of these stanzas, grub will only ask for the password if you try to edit the commands at boot time. In other words, a standard boot using the existing items from the grub configuration does not require the password.

Here's an example of `grub.conf` where the password is inside a "title" stanza:

```
default=0
timeout=5
splashimage=(hd0,0)/boot/grub/splash.xpm.gz
hiddenmenu
title Red Hat Enterprise Linux Client (2.6.18-92.el5)
    root (hd0,0)
    kernel /boot/vmlinuz-2.6.18-92.el5 ro root=LABEL=/ rhgb quiet
    initrd /boot/initrd-2.6.18-92.el5.img
    password --md5 $1$c1Vc1/$R1F6Wm6XF3amlhOcDTCdv.
```

Hardening the BIOS

Would-be intruders that have physical access to the computer can boot from a rescue disk (or any "live" Linux CD), then mount and access the file systems on the hard drive. To prevent this the BIOS should be configured to boot only off the hard drive. When this is done, set a password on the BIOS.

Intrusion detection

A user with a legitimate account on your system who gains root access for a while (perhaps because you walked away from the computer with a root login still active) can plant a "back door" by creating an executable program that's owned by root and has the "setuid" bit turned on. This special mode bit causes the executable to run with the effective permissions of the file's owner.

You can see an example of such a program by listing the password-changing program:



```
# ls -l /usr/bin/passwd
-rwsr-xr-x 1 root root 27768 Jul 17 2006 /usr/bin/passwd
```

Notice the 's' in the fourth character position that indicates that the set-user-id bit is on. This program runs "setuid to root" so that ordinary (non-root) users can use it to change their passwords (which requires updating their password hash in `/etc/shadow`).

There are a number of legitimate `setuid` programs on the system. However, unexpected `setuid` programs (especially if owned by root) are an indication that a back door may have been created by an intruder. You can find such programs with the `find` command; for example:



```
# find / -user root -perm +4000 2> /dev/null
/bin/ping
/bin/su
/bin/mount
/usr/bin/passwd
/usr/bin/sudo
/usr/bin/crontab
/usr/bin/sudoedit
/usr/bin/chage
/usr/bin/rlogin
/usr/bin/rsh
/usr/bin/at
/usr/bin/chfn
/usr/bin/newgrp
... many lines of output deleted ...
```

In this example, the value "+4000" represents the octal value of the setuid mode bit.

You could implement a simple intrusion detection system by first capturing a list of these files when the system is in a pristine state:



```
# find / -user root -perm +4000 2> /dev/null > /root/good-setuid-list
```

Subsequently you could build a new list then compare the two. In the example below a new setuid copy of bash was deliberately created:



```
# find / -user root -perm +4000 2> /dev/null > /root/new-setuid-list
# diff /root/good-setuid-list /root/new-setuid-list
36a37
> /home/chris/bash
```

Intrusion detection tools such as tripwire and AIDE (Advanced Intrusion Detection Environment) provide a more thorough and more systematic way to detect unexpected changes to the file system. Their use is beyond the scope of LPI-1

There is also a "set group ID" bit (octal value 2000) that causes an executable to run with the effective group identity of its group. For example:



```
# ls -l /usr/bin/wall
-r-xr-sr-x 1 root tty 14792 Oct 13 2006 /usr/bin/wall
```

Though less common (and less dangerous) you can also search for setgid programs with find:



```
# find / -perm +2000 2> /dev/null
/usr/bin/ssh-agent
/usr/bin/xterm
/usr/bin/crontab
/usr/bin/locate
... lines deleted ...
/usr/bin/wall
```

Password Management

Users can set their own password with the passwd command:



```
$ passwd
Changing password for user chris.
Changing password for chris
(current) UNIX password:
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
```

Root can set or change the password of any user:



```
# passwd chris
Changing password for user chris.
New UNIX password:
BAD PASSWORD: it is based on a dictionary word
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
```

Note that when a non-root user sets his password, a password strength check is enforced. However when root sets a user's password, the password strength check merely issues a warning; it does not prevent root from setting a weak password.

Password strength checking policies are controlled by PAM modules. For additional control over password strength, consider using the modules `pam_passwdqc` or `pam_cracklib`.

Passwords are not stored anywhere in the system as plain text. Instead, their hash is written into the file `/etc/shadow`. Here is a sample line:



```
# grep chris /etc/shadow
chris:$1$z5b2XDJ7$LL0h4VX5us3UMP2PMFXFp0:14869:0:99999:7:::
```

When a user logs in and supplies his password, it is hashed, and the hash is compared with what's in `/etc/shadow`. Passwords may be *locked* and *unlocked* to temporarily disable and re-enable an account. Root can lock any user's password using `passwd -l` and unlock it using `passwd -u`. Locking the password simply places "!" in front of the password hash so that it is no longer a valid hash. Unlocking the password simply removes the "!" characters:



```
# passwd -l chris
Locking password for user chris.
passwd: Success
# grep chris /etc/shadow
chris:!!$1$z5b2XDJ7$LLOh4VX5us3UMP2PMFXFp0:14869:0:99999:7:::
# passwd -u chris
Unlocking password for user chris.
passwd: Success.
# grep chris /etc/shadow
chris:$1$z5b2XDJ7$LLOh4VX5us3UMP2PMFXFp0:14869:0:99999:7:::
```

A *password aging* policy can be implemented to force a user to choose a new password every so often, or to force expiry of a user's account on a specified date. The password aging parameters are stored in fields 3 – 8 in `/etc/shadow` and are managed using the `chage` command. Options for the `chage` command include:

- E Set an account expiry date
- m Set the minimum number of days between password changes (0 means "no minimum")
- M Set the maximum lifetime of a password
- W Set the warning period (number of days) for a user whose password will soon expire
- l List current password aging values

Example:



```
# chage -E 31-12-2011 -M 30 -W 7 chris
# chage -l chris
Last password change           : Sep 17, 2010
Password expires               : Oct 17, 2010
Password inactive              : never
Account expires                : Jun 02, 2037
Minimum number of days between password change : 0
Maximum number of days between password change : 30
Number of days of warning before password expires : 7
```

Best Practices

System administrators should minimise the amount of time they spend logged in as root. They should have a normal account which they use for general work, and should transition to root only when they need to do something that requires root privilege. Transitioning to root is performed

[ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification] Created during the initiative "ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software in Africa", a programme of FOSSFA and GIZ. For more information see www.ict-innovation.fossfa.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.

using the `su` (substitute user) command.

The general form of the command is:

```
su [-l] [user]
```

`su` will prompt for the password of the user then start a shell running with that user's identity. If no user name is specified the command starts a root shell. This is the most common usage. The `-l` option tells `su` to start a login shell. This shell will perform the same start-up processing as if the user was initially logging in; that is, it will set up the user's environment, including his search path. Normally, this is what you want to do. (You can also just use `-` instead of `-l`)

The `id` command can be used to display a user's identity. and group memberships. In the example below we use it to check the identity before and after using `su`:



```
$ id
uid=500(chris) gid=500(chris) groups=500(chris)
$ su - l
Password:
# id
uid=0(root) gid=0(root)
groups=0(root),1(bin),2(daemon),3(sys),4(adm),6(disk),10(wheel)
```

The `sudo` command can be used to grant non-root users the ability to run *specific* commands as root without telling them the root password. It is configured by the file.

Here's a simple example of a rule from that file:

```
chris    ALL=(ALL) /sbin/iptables
```

with this rule in place, `chris` can elevate his privilege to run the `iptables` command as `root`:



```
$ /sbin/iptables -F
iptables v1.3.5: Permission denied (you must be root)
$ sudo /sbin/iptables -F
[sudo] password for chris:
$
```

The password that's being requested here is the user's own password, *not* the root password.

Here's a more complex rule from `sudoers`:

```
harry williambox=(william) /bin/ls, /home/william/bin/testscript.sh
```

This rule allows `harry` to run `/bin/ls` and `/home/william/bin/testscript.sh` as `william`, only on the machine `williambox`. (Restricting a `sudo` rule to one specific machine is useful if the `sudoers` file is shared network-wide). Notice that commands in a `sudoers` file must be specified using absolute path names.

There is a "grace period" within which `sudo` will not prompt for the password again. This can be set by a line in the `sudoers` file like this:

```
Defaults    timestamp_timeout=10
```

Setting it to 0 will force sudo to prompt the a password every time. Users will probably find this annoying!

Larger, more complex `sudoers` files can be made easier to maintain by defining *user aliases* (named lists of users), *host aliases* (named lists of hosts) and *command aliases* (named lists of commands). Here's a longer example:

```
User_Alias  ADMINS = jsmith, mikem

Cmnd_Alias  SOFTWARE = /bin/rpm, /usr/bin/up2date, /usr/bin/yum

Cmnd_Alias  SERVICES = /sbin/service, /sbin/chkconfig

Cmnd_Alias  STORAGE = /sbin/fdisk, /sbin/sfdisk, /sbin/parted,
/sbin/partprobe, /bin/mount, /bin/umount

Host_Alias  FILESERVERS = fs1, fs2

Host_Alias  MAILSERVERS = smtp, smtp2

ADMINS      FILESERVERS = STORAGE
chris       ALL = SOFTWARE
%wheel      ALL = (ALL) ALL
```

In this example, the command aliases effectively define roles, in terms of the set of commands which that role needs to be able to execute. The last three lines of the file are the actual rules. Taken in turn:

- 1.Members of the user alias ADMINS can execute any of the commands in the command alias STORAGE on any of the machines listed in the host alias FILESERVERS
- 2.chris can execute commands in the command alias SOFTWARE on all machines
- 3.Members of the Linux wheel group can execute any command as root on any machine

The `sudoers` file must be edited using the command `visudo`. It should not be edited directly. The `visudo` command will verify that the file is syntactically valid before saving it.

Using `sudo` has a number of advantages:

- 1.The number of people who need to know the root password is kept to a minimum
- 2.Users spend the absolute minimum amount of time running as root (just the duration of one command) and immediately revert to non-root command prompt
- 3.sudo logs all its actions, usually in `/var/log/secure`. Thus, an audit trail of all root activity is kept.

Ubuntu uses `sudo` exclusively for root commands. In the default configuration the root account is locked so that it is not possible to log in directly as root or to become root using `su`. Instead, `sudo` is used for everything. The key line in Ubuntu's `sudoers` file looks like this:

```
%admin      ALL = (ALL) ALL
```

The initial account created when Ubuntu is installed is automatically made a member of the *admin* group.

Find and Close Ports

A machine should only run those services it's supposed to be running. Un-needed services and their associated open ports can potentially weaken security. If a service should not be running, make sure it isn't configured to start. For a RedHat style distribution (or other distribution that uses the traditional "System-V" style service management, use the `chkconfig` command to check and modify the status of a service. The following command sequence shows how to check the current status of the `httpd` daemon (web server) and to disable it.



```
# chkconfig --list httpd
httpd          0:off 1:off 2:on  3:on  4:on  5:on  6:off
# chkconfig httpd off
# chkconfig --list httpd
httpd          0:off 1:off 2:off 3:off 4:off 5:off 6:off
```

In the example above we see that `httpd` is initially configured to start in run levels 2, 3, 4 and 5.

Note that `chkconfig` only affects the boot-time behaviour (or more accurately, the behaviour when the run-level changes). Turning a service off with `chkconfig` will not actually stop it running. For that you should run the service's control script:



```
# /etc/init.d/httpd stop
Stopping httpd:          [ OK ]
```

Another way to examine which services are running is to list the open ports on the machines. You can use `netstat` or `lsof` for this; these were discussed in topic 109.3

You can also determine which ports are open using `nmap`. Nmap is a "network reconnaissance" tool, more often called a *port scanner*. Nmap will examine your open ports "from the outside", whereas `netstat` and `lsof` examine them "from the inside" of the machine. Nmap is also useful for verifying that your firewall is behaving the way you expect.

Be aware, though, that `nmap` can also be used in the initial stages of attacking a machine, to determine what ports are running and thus what vulnerabilities might be exploited. Because of this, running port scans on your machines (even your own machines) might be considered a hostile act, and you should obtain management permission before doing so.

The command syntax for `nmap` is summarised in the figure below:

Range of port numbers.
You can also add a protocol specifier such as T:21-25 to scan a range of TCP ports

Range of IP addresses.
You can also use wildcards such as: 192.168.*.*
or CIDR-style netmasks such as: 192.168.0.0/16
or full domain names such as: foo.example.com

nmap -sT -p 20-100 192.168.0.1-50

Scan type. Other types include:

- sS: TCP SYN scan ("half-open")
- sP: Ping scanning
- sV: Version detection scan
- sF: Stealth FIN scan
- sX: "Xmas Tree" scan
- sM: "Null" scan

Here's an example of an nmap scan:

```

$ nmap 192.168.81.130

Starting Nmap 5.00 ( http://nmap.org ) at 2010-09-17 16:09 BST
Interesting ports on 192.168.81.130:
Not shown: 997 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
111/tcp   open  rpcbind

Nmap done: 1 IP address (1 host up) scanned in 0.20 seconds
    
```

User Limits

When the `/etc/nologin` file is present (it can be empty) it will prevent all users from login in to the system (except user root). If the **nologin** file contains a message this will be displayed after a successful authentication.

In the `/etc/security/` directory are a collection of files that allow administrators to limit user CPU time, maximum file size, maximum number of connections, etc on a per-user basis.

The file `/etc/security/access.conf` can be used to disallow logins for groups and users from specific locations.

The file `/etc/security/limits.conf` can be used to set hard and soft limits for a number of resources. Each line in this file is of the form:

```
<domain>          <type> <item> <value>
```

domain	a user name, a group name (with @group)	
type	hard or soft	
item	<i>core</i>	- limits the core file size (KB)
	<i>data</i>	- max data size (KB)
	<i>fsize</i>	- maximum filesize (KB)
	<i>memlock</i>	- max locked-in-memory address space (KB)
	<i>nofile</i>	- max number of open files
	<i>cpu</i>	- max CPU time (MIN)
	<i>nproc</i>	- max number of processes
	<i>as</i>	- address space limit
	<i>maxlogins</i>	- max number of simultaneous logins for this user
	<i>priority</i>	- the priority to run user process with
	<i>locks</i>	- max number of file locks the user can hold

This is not a complete list – see the man page for `limits.conf(5)` for the full list.

Here are some sample entries from `limits.conf`, with comments:

```
# User ellie can never create a file larger than 100 Mbytes
ellie hard fsize 102400

# Ellie must increase her soft limit to create file larger than 50
Mbytes
ellie soft fsize 51200

# Users in the student group cannot run more than 50 simultaneous
processes
@student hard nproc 20

# By default, no users can create core files unless they up their soft
limit
* soft core 0
```

The soft limit for a resource is the limit that's currently in force. The hard limit is the maximum that the user can increase the soft limit to.

A user can use the `ulimit` command (a shell built-in) to change the limits currently in force. A (non-root) user can set a soft limit to any value that's not greater than the hard limit. He can also reduce the hard limit (but not put it back up again). These limits are held within the context of that shell and are inherited by any programs started from that shell. However if a user logs out and back in again the limits will be re-established from `limits.conf`.

The `ulimit` command takes many options, including:

- a All current limits are reported
- c The maximum size of core files created
- d The maximum size of a process's data segment
- e The maximum scheduling priority ("nice")
- f The maximum size of files written by the shell and its children
- i The maximum number of pending signals

- l The maximum size that may be locked into memory
- n The maximum number of open file descriptors
- p The pipe size in 512-byte blocks (this may not be set)
- q The maximum number of bytes in POSIX message queues
- r The maximum real-time scheduling priority
- s The maximum stack size
- t The maximum amount of cpu time in seconds
- u The maximum number of processes available to a single user
- v The maximum amount of virtual memory available to the shell
- x The maximum number of file locks

In the example below we set the maximum amount of CPU time to 100 seconds then report all values:

```


$ ulimit -t 100
$ ulimit -a
core file size          (blocks, -c) 0
data seg size          (kbytes, -d) unlimited
scheduling priority    (-e) 0
file size              (blocks, -f) unlimited
pending signals        (-i) 8192
max locked memory      (kbytes, -l) 32
max memory size        (kbytes, -m) unlimited
open files             (-n) 1024
pipe size              (512 bytes, -p) 8
POSIX message queues   (bytes, -q) 819200
real-time priority     (-r) 0
stack size             (kbytes, -s) 10240
cpu time               (seconds, -t) 100
max user processes     (-u) 8192
virtual memory         (kbytes, -v) unlimited
file locks             (-x) unlimited

```

Establishing ulimit settings at login time from the settings in `limits.conf` is performed by the PAM module `pam_limits`. Some systems also support the directory `/etc/security/limits.d`. The contents of any files found there are processed in turn as if they were appended to `/etc/security/limits.conf`

The following is a partial list of the used files, terms and utilities:

- find
- passwd
- lsof
- nmapchage
- netstat
- sudo
- /etc/sudoers
- su
- usermod
- ulimit

110.2 Setting Up Host Security

Candidates should know to set up a basic level of host security

Key Knowledge Areas

- Awareness of shadow passwords and how they work.
- Turn off network services not in use.
- Understand the role of TCP wrappers.

Many network services, such as `sshd` and `httpd`, are started at boot time and run continuously. They handle their own network connections, accepting connections from clients and servicing them.

However, some services rely on a super-server to listen for and accept connection requests on their behalf. When a connection is accepted, the super-server starts up the required service and passes the network connection to it via a pair of file descriptors.

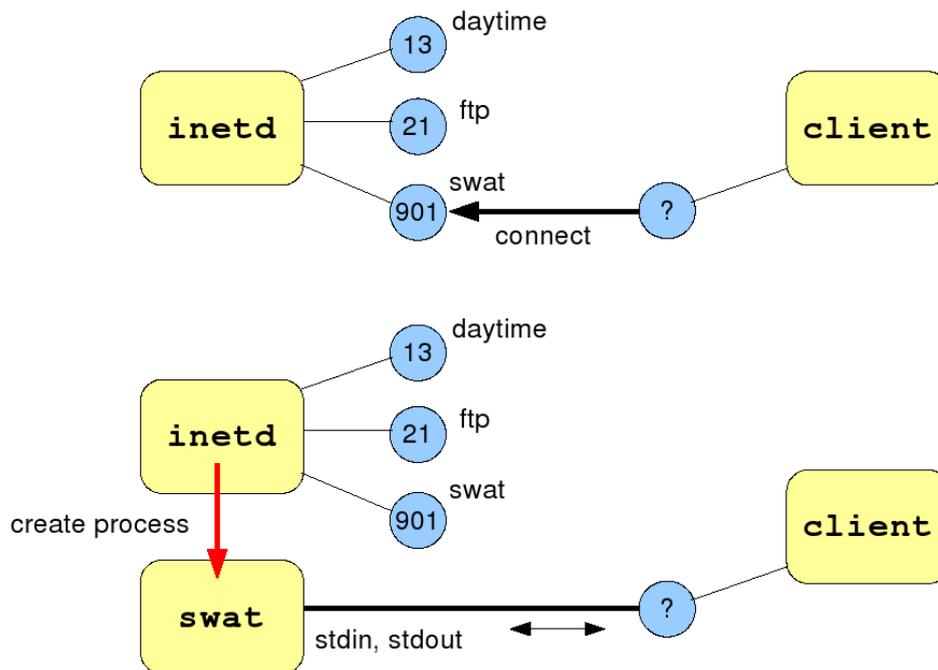
This way of working reduces the number of processes in the system which are blocked, awaiting client connections. It can also provide a central place to do access control and logging when service requests are received. There is, however, a small overhead in creating the service processes on demand, so this technique is usually not used for services that can experience a high rate of connection requests.

There are two super-servers. The first is `inetd`. This is quite old and is not used on any current Linux distributions; however it is included in the LPI-1 objectives. The more modern super-server is `xinetd`. This adds access controls and logging capabilities.

The `inetd` daemon (old)

This daemon is started at boot time and listens for connections on specific ports. This allows the server to run a specific network daemon only when needed. For example, the old `telnet` service has a daemon `in.telnetd` which handles telnet sessions. Instead of running this daemon all the time `inetd` is instructed to listen on port 23. As another example, `swat` (a browser-based configuration tool for samba) also relies on a super-server to listen for connections.

The process of accepting a connection and starting the service is shown below.



inetd is configured in the file `/etc/inetd.conf`. Each line defines one service for which inetd should listen.

The fields of `/etc/inetd.conf` contain the following:

service-name	valid name from <code>/etc/services</code>
socket type	stream for TCP and dgram for UDP
protocol	valid protocol from <code>/etc/protocols</code>
flag	nowait if multithreaded and wait if single-threaded
user/group	run application as user or group.
program	The name of the program to run
argument	The arguments to be passed to the program (if any)

Examples:

```
daytime stream tcp nowait root internal
telnet stream tcp nowait root /usr/sbin/in.telnetd
pop-3 stream tcp nowait root /usr/sbin/tcpd ipop3d
```

The first line shows a simple TCP service that's implemented internally by `inetd`. There is no external server to start. The second line starts an "inet-aware" version of the telnet server. The third line shows a service that's started via an intermediate program called TCP Wrappers (`tcpd`). This program adds an access control layer, and is discussed later in this topic.

The service names that appear in the first field of this file are looked up in the `/etc/services` file to find the associated port numbers. The fields in `/etc/services` are as follows:

```
service-name  port/protocol  [aliases]
```

For example, the entry for `pop3` appears as follows:

```
pop3          110/tcp        pop-3
```

If the `/etc/inetd.conf` file is changed, send a HUP signal to `inetd` to make it re-read the file:



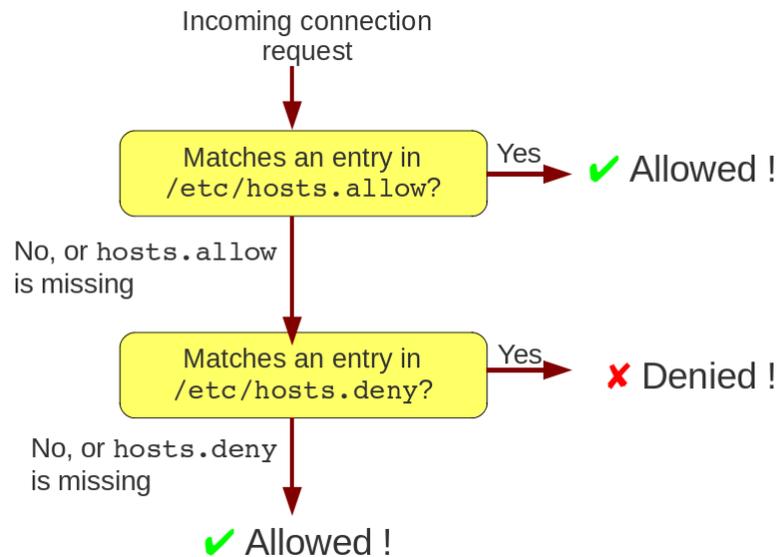
```
# pkill -HUP inetd
```

TCP wrappers

TCP wrappers is a program that adds an access control layer to services, under control of the files `/etc/hosts.allow` and `/etc/hosts.deny`. Often, TCP wrappers (the `tcpd` daemon) is used in conjunction with `inetd` to add access control. It works like this:

1. The `inetd` server listens for and accepts connections to a specified service (`pop3` for example)
2. `inetd` starts up the program specified in its config file for that service. To use TCP wrappers, that program is specified as `/usr/sbin/tcpd`
3. The name of the real server (`ipop3d` in our example) is passed as an argument to `tcpd`
4. `tcpd` consults the files `/etc/hosts.allow` and `/etc/hosts.deny` (discussed below) to decide whether or not to allow the connection.
5. If the connection is allowed, `tcpd` starts up the real server and passes it the file descriptors relating to the network connection from the client.

The figure below shows the logic used in checking entries in the `hosts.allow` and `hosts.deny` files.



Note that the default situation (if the `hosts.allow` and `hosts.deny` files are empty) is to allow access.

The format of the lines in both these files is:

```
daemon-list : client-host-list [ : shell-command ]
```

A typical configuration places a single line in `hosts.deny`:

```
ALL: ALL
```

With this file in place, all connections are denied unless they match a rule in `hosts.allow`.

Here are some sample entries from `hosts.allow`:

```
sshd: ALL except 192.168.1.11
pop3: 192.168.1.0/24
```

In these examples the client hosts are identified by IP address (or IP address block). You can also specify a host name here, or use the keyword `ALL` to match all hosts.

TCP wrappers will also log each connection to a service. Examining the logs for evidence of unauthorised attempts to connect may provide evidence of an attack on your system.

TCP wrappers is often invoked through the `tcpd` daemon, via `inetd`. However, some services such as `sshd` and `vsftpd` are linked against the `libwrap` library (the library that handles the access control checks against `hosts.allow` and `hosts.deny`) and therefore honour these access controls.

You can check if a service uses the `libwrap` library by examining the output from `ldd`, which shows which dynamic linked libraries (`.so` files) an executable requires. For example:



```
$ ldd /usr/sbin/sshd
```

[ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification] Created during the initiative "ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software in Africa", a programme of FOSSFA and GIZ. For more information see www.ict-innovation.fossfa.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.

```
libwrap.so.0 => /usr/lib64/libwrap.so.0 (0x00002b4c09fe3000)
libpam.so.0 => /lib64/libpam.so.0 (0x00002b4c0a1ec000)
libdl.so.2 => /lib64/libdl.so.2 (0x00002b4c0a3f7000)
libselinux.so.1 => /lib64/libselinux.so.1 (0x00002b4c0a5fc000)
libaudit.so.0 => /lib64/libaudit.so.0 (0x00002b4c0a814000)
libcrypto.so.6 => /lib64/libcrypto.so.6 (0x00002b4c0aa29000)
... lines deleted ...
```

A little command line cunning can find all the files in a directory that use libwrap:



```
$ find /usr/sbin -type f -exec grep -l libwrap {} \; 2> /dev/null
/usr/sbin/mailstats
/usr/sbin/makemap
/usr/sbin/stunnel
/usr/sbin/conmand
/usr/sbin/sendmail.sendmail
/usr/sbin/smrsh
/usr/sbin/rpc.rquotad
/usr/sbin/praliases
/usr/sbin/sshd
```

The xinetd Daemon

In modern Linux distributions `xinetd` has replaced `inetd`. It performs essentially the same job, but it's also able to enforce access control restrictions, based on the client machine, the time of day, the load on the machine, and so on. It also logs each connection so that the `tcpd` daemon is no longer used, instead `xinetd` does everything. Configuration is done either through a single file `/etc/xinetd.conf` and `/` or by individual files in `/etc/xinetd.d/` named after the services being monitored by `xinetd`.

In most distributions, the file `/etc/xinetd.conf` provides default settings which may be overridden by the entries in the service-specific files in `/etc/xinetd.d`

Structure of service file in `xinetd.d`

```
Service-name {
    disable = yes/no
    socket_type = stream for TCP and dgram for UDP
    protocol = valid protocol from /etc/protocols
    wait = <yes or no>
    user= the user the application runs as
    group= the group the application runs as
    server= the name of the program to be run for this service
}
```

Here's an example of the "top level" file, `/etc/xinetd.conf`

```
defaults
{
    log_type          = SYSLOG daemon info
    log_on_failure    = HOST
    log_on_success    = PID HOST DURATION EXIT

    cps               = 50 10
    instances         = 50
    per_source        = 10

    v6only           = no

    groups           = yes
    umask            = 002
}

includedir /etc/xinetd.d
```

In this example, we set the facility and priority at which xinetd will send messages to syslog and specify what information will be included in those messages. We also implement some simple access controls that limit the load on the machine. In this example the `cps` directive limits the rate of incoming connections to 50 per second. If this rate is exceeded the service will be temporarily disabled for 10 seconds. (You might also think of this as a way for an attacker to mount a denial-of-service attack on the machine!) The `instances` directive determines the maximum number of servers that can be simultaneously active for a service.

Here's an example of a service-specific file:

```
service rsync
{
    disable          = no
    socket_type      = stream
    wait            = no
    user            = root
    server          = /usr/bin/rsync
    server_args     = --daemon
    log_on_failure  += USERID
    only_from       = 192.168.0.0/24
}

```

The field you're most likely to need to change here is the "disable" field. Set "disable = no" to turn the service on, or set "disable = yes" to turn it off. After making a change to the xinetd configuration, send it a SIGHUP signal:



```
# pkill -HUP xinetd
```

For more detail, see the man page for `xinetd.conf(5)`

The following is a partial list of the used files, terms and utilities:

- /etc/nologin
- /etc/passwd
- /etc/shadow
- /etc/xinetd.d/*
- /etc/xinetd.conf
- /etc/inetd.d/*
- /etc/inetd.conf
- /etc/inittab
- /etc/init.d/*
- /etc/hosts.allow
- /etc/hosts.deny

110.3 Securing data with encryption

The candidate should be able to use public key techniques to secure data and communication

Key Knowledge Areas

- Perform basic OpenSSH-2 client configuration and usage
- Understand the role of OpenSSH-2 server host keys
- Perform basic GnuPG configuration and usage
- Understand SSH port tunnels (including X11 tunnels)

Introducing ssh

The secure shell (SSH) is a popular replacement for earlier remote login tools such as `telnet` and `rlogin`. The biggest advantages of SSH over these earlier tools is that all traffic between client and server is encrypted, and the server's identity is securely verified. The implementation of secure shell used on Linux is called OpenSSH.

The OpenSSH server (`sshd`) listens on port 22 by default. On some Linux distributions the server is installed and running by default; on others it will need to be installed using the package management tools. The client-side pieces of `ssh` (commands such as `ssh`, `ssh-keygen`, `ssh-agent`, and `ssh-add`) are usually installed by default.

SSH uses public/private key encryption technology to securely authenticate the server when a connection is made. This authentication handshake also performs a secure exchange of a shared secret key that is used to encrypt all subsequent traffic between the client and the server. The figure below illustrates the difference between *symmetric* cryptography (using a shared secret key) and *asymmetric* cryptography (using a public and private key pair).

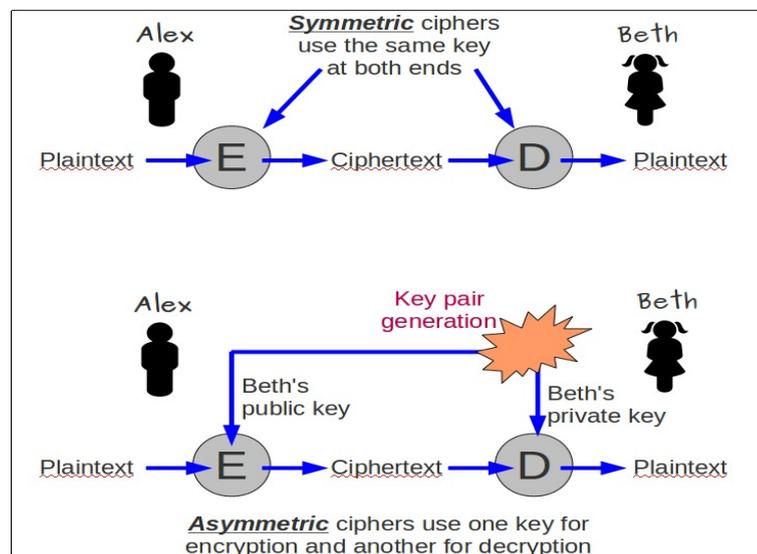


Figure 110.3-1: Symmetric and asymmetric cryptography

SSH Host Authentication

[ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification] Created during the initiative "ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software in Africa", a programme of FOSSFA and GIZ. For more information see www.ict-innovation.fossfa.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.

When the openssh server is installed, a public/private key pair is generated for that machine. Actually, two key pairs are generated: a DSA key pair and an RSA key pair. (DSA and RSA are just the names of two asymmetric cryptography technologies.) The private keys are stored in `/etc/ssh/ssh_host_rsa_key` and `/etc/ssh/ssh_host_dsa_key`; these files are readable only by root. The public keys are stored in `/etc/ssh/ssh_host_rsa_key.pub` and `/etc/ssh/ssh_host_dsa_key.pub`. These keys are summarised in the figure below.

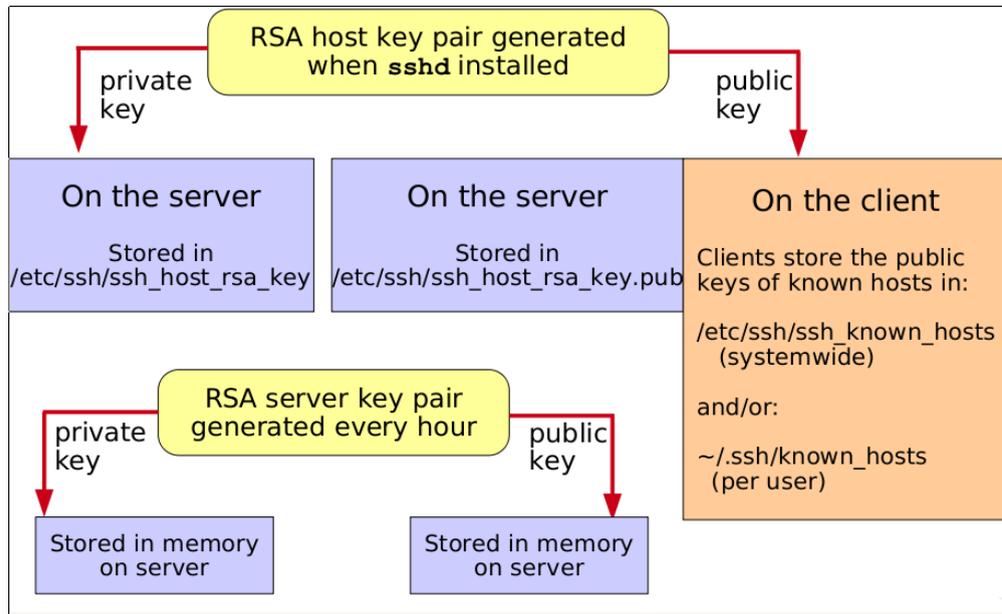


Figure 110.3-2: Public and private server keys

These keys are used to securely verify the identity of the server machine, and to set up an encrypted channel via a key exchange protocol. The server's public key needs to be available on any client that needs to connect. It can be copied into place on the client by the system administrator (into the file `/etc/ssh/ssh_known_hosts`). Your distribution may have a utility called `ssh-keyscan` to help with this. Alternatively, when an ssh client connects to a server, the server will offer the host's public key. At this stage the user will be prompted with something like this:

```


# ssh chris@192.168.81.1
The authenticity of host '192.168.81.1 (192.168.81.1)' can't be
established.
RSA key fingerprint is
a9:77:96:9f:13:24:59:90:92:f6:e2:da:f3:91:e5:06.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.81.1' (RSA) to the list of known
hosts.
chris@192.168.81.1's password:

Last login: Tue Sep  7 21:00:02 2010
$

```

A security-minded individual might choose to verify this key fingerprint against the same fingerprint generated on the server using the command `ssh-keygen`. For example:



```
# ssh-keygen -l
Enter file in which the key is (/root/.ssh/id_rsa):
/etc/ssh/ssh_host_rsa_key.pub
2048 a9:77:96:9f:13:24:59:90:92:f6:e2:da:f3:91:e5:06
/etc/ssh/ssh_host_rsa_key.pub
```

However, most users simply trust that they are connecting to the correct server (and not an imposter) and simply answer yes to accept and continue the connection. At this point, the server's public key will be added to the local `~/.ssh/known_hosts` file. Obviously, this file is "per-user" – it only applies to the user who logged in.

Once the server's public key is known to the client, the "are you sure you want to continue" message will not appear again.

User Authentication

OpenSSH supports several mechanisms to authenticate the user. By default, on most Linux distributions `ssh` is configured to support simple password-based authentication – the user is prompted for the password for his account on the remote server and logs in. Note, however, that this password travels across an encrypted channel to the server and is not subject to being stolen by an eavesdropper.

A user can also be authenticated using a public/private key pair, in much the same way that an SSH server authenticates to the client when the initial connection is made. For this to work, the user must first generate himself a key pair, then he must copy the public key of the pair onto any server on which he needs to authenticate.

Key generation is performed using `ssh-keygen`:



```
$ ssh-keygen -t dsa
Generating public/private dsa key pair.
Enter file in which to save the key (/home/chris/.ssh/id_dsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/chris/.ssh/id_dsa.
Your public key has been saved in /home/chris/.ssh/id_dsa.pub.
The key fingerprint is:
14:af:92:52:d5:38:ba:82:88:11:44:ee:f8:b1:66:1a chris@m1530-
rhel.example.com
```

will generate a DSA key. By default these keys will be saved in `~/.ssh/id_dsa` and `~/.ssh/id_dsa.pub`. If RSA keys are generated they would be stored in `~/.ssh/id_rsa` and `~/.ssh/id_rsa.pub`. Note that the user has the option to protect the private key with a pass-phrase. He may choose not to use a pass-phrase by simply pressing 'Enter' at this prompt.

The file containing the public key will need to be copied onto the server (using `scp` for example) and added to the file `~/.ssh/authorized_keys` in the user's home directory on the server, as shown in the figure below. Your distribution may include a handy script called `ssh-copy-id` which automates

this process:

```
$ ssh-copy-id -i .ssh/id_dsa.pub chris@192.168.81.1

chris@192.168.81.1's password:
Now try logging into the machine, with "ssh 'chris@192.168.81.1'", and
check in:

    .ssh/authorized_keys
to make sure we haven't added extra keys that you weren't expecting.
```

This script will create the `~/.ssh` directory and the `authorized_keys` file on the remote machine if necessary, and set their permissions correctly. If the `authorized_keys` file already exists, the key will be appended to it.

The management of user keys is summarised in the figure below.

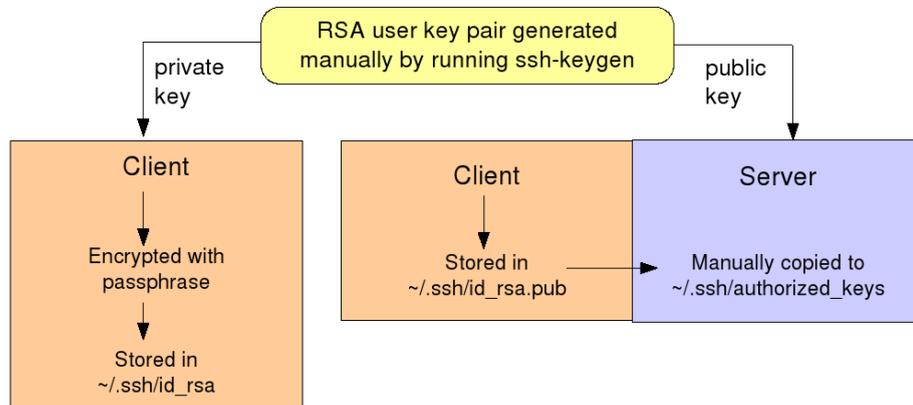


Figure 110.3-3: SSH user public and private keys

With this key in place, if the user did not set a pass-phrase on his private key he can now perform a password-less ssh login to the server. However, if he did set a passphrase, he will be prompted for it at this point:

```


$ ssh 192.168.81.1
Enter passphrase for key '/home/chris/.ssh/id_dsa':

Last login: Tue Sep 28 12:01:42 2010 from m1530-rhel.local
```

Note: The permissions on the files are important. All files holding private keys should be mode 600. The `~/.ssh` directory should have mode 700. ssh will refuse to use these files if the permissions are too lax.

To avoid the inconvenience of having to supply a passphrase to unlock your private key during

ssh login, the program `ssh-agent` can be used to hold the passphrase and supply it automatically when required. You supply your passphrase to the agent just once, and the agent will give the keys out to other processes owned by you on request.

The following dialog starts `ssh-agent` and adds your identities to it:



```

$ eval $(ssh-agent)
Agent pid 22305

$ ssh-add
Enter passphrase for /home/chris/.ssh/id_dsa:
Identity added: /home/chris/.ssh/id_dsa (/home/chris/.ssh/id_dsa)

```

Now, you should be able to perform a secure but password-free login to any server that has a copy of your public user key.

The strange use of command substitution and the `eval` command to start `ssh-agent` arises because `ssh-agent` reports some environment variables on its standard output which the process needs to know in order to contact the agent. This 'trick' allows us to propagate these variable back into the shell. Note that the older command substitution syntax uses back-quotes like this: `eval `ssh-agent``

Client-side ssh commands

Once configured, the end-user experience of SSH is straightforward. Some sample commands are shown in the table below.

Table: Sample ssh commands

Command	What it does
<code>\$ ssh neptune</code>	Log in on machine neptune using the local user account name
<code>\$ ssh brian@neptune</code>	Log in on machine neptune as user brian
<code>\$ ssh neptune date</code>	Run a single command on neptune
<code>\$ scp foo neptune:</code>	Copy the local file foo into your home directory on neptune
<code>\$ scp foo neptune:/tmp/foo</code>	Copy local file foo to the specified path name on neptune
<code>\$ scp neptune:/tmp/foo .</code>	Copy /tmp/foo from neptune into your home directory on the local machine

Port forwarding

ssh has a useful feature called *remote port forwarding*, that allows the remote ssh server to listen for connections on any specified port and forward the traffic through an encrypted ssh tunnel to a specified port on the local machine. In the top half of the figure shown below, a user on machine mercury has run the command:

```
$ ssh -R 4023:mercury:23 venus
```

Then on machine earth, a user runs `telnet`, connecting to port 4023 on venus. Port forwarding means that the traffic is secure, and that the only port that needs to be open for connections on the server is port 22. In practice, 'venus' and 'earth' are often the same machine.

Local port forwarding is similar but it's the client that forwards the traffic, not the server. In the bottom half of the figure a user on mercury has run the command:

```
$ ssh -L 8080:earth:80 venus
```

(Again, 'venus' and 'earth' are often the same machine.) We can then browse securely through an encrypted channel by directing our browser to localhost:8080.

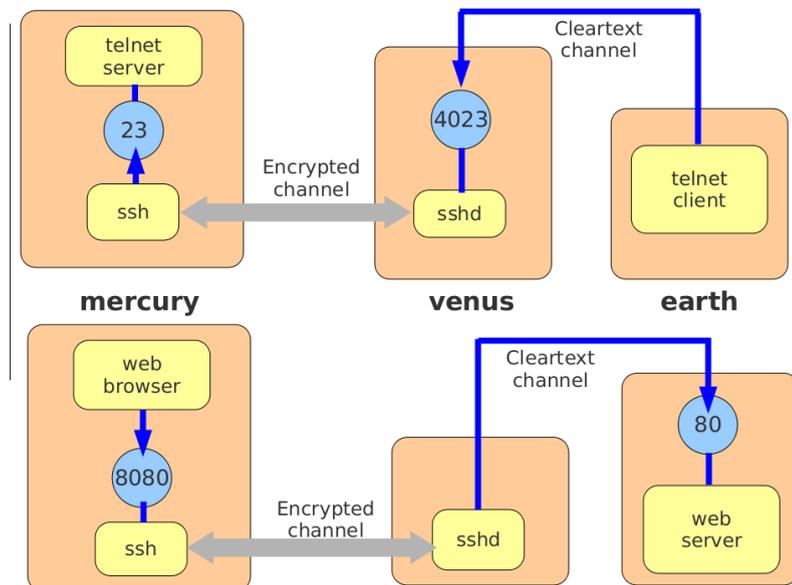


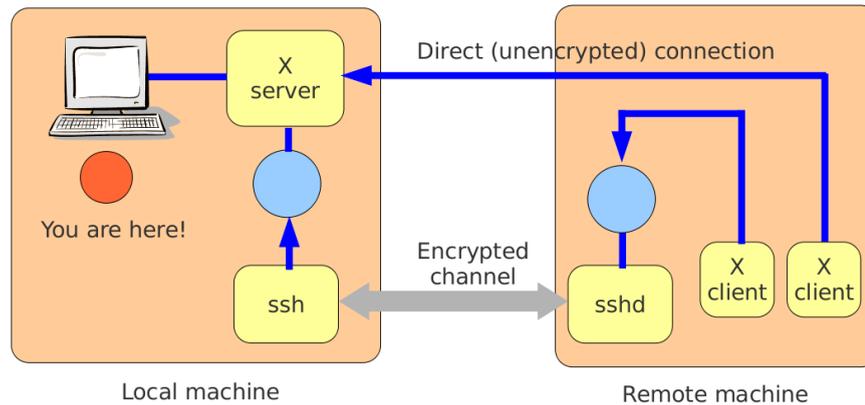
Figure 110.3-4: Local and remote port forwarding

As a "special case" of port forwarding, ssh can arrange to carry X traffic from a remote graphical application back to the X server on the local machine. This is known as *X11 forwarding*. The following command:

```
$ ssh -X neptune xcalc
```

runs `xcalc` on the machine `neptune` and arranges for it to display on the X server on the local machines. Behind the scenes, `sshd` acts as a proxy X server and forwards all X traffic down the encrypted channel back to the client. In addition, the `DISPLAY` environment variable is set to direct the `xcalc` program to the proxy server. The operation is illustrated below:

Figure 110.3-5: X-11 forwarding using ssh



GnuPG

GPG (Gnu Privacy Guard) is a re-write of the original PGP software. It uses a combination of symmetric and asymmetric cryptography to provide secure communication between two parties. GPG also provides tools for key management.

At the heart of GPG is a command line tool called `gpg`, that handles the basic tasks of signing, encrypting and decrypting messages, and also handles key management. (Many email programs have plugins to encrypt, sign and decrypt email that use this command-line tool behind the scenes.) To illustrate its use, suppose Alex wants to communicate securely with Beth.

First, Alex generates his public/private key pair with the command `gpg--gen-key`:



```
$ gpg --gen-key
gpg (GnuPG) 1.4.5; Copyright (C) 2006 Free Software Foundation, Inc.
This program comes with ABSOLUTELY NO WARRANTY.
This is free software, and you are welcome to redistribute it
under certain conditions. See the file COPYING for details.

gpg: directory `/home/alex/.gnupg' created
gpg: new configuration file `/home/alex/.gnupg/gpg.conf' created
gpg: WARNING: options in `/home/alex/.gnupg/gpg.conf' are not yet active
during this run
gpg: keyring `/home/alex/.gnupg/secring.gpg' created
gpg: keyring `/home/alex/.gnupg/pubring.gpg' created
Please select what kind of key you want:
  (1) DSA and Elgamal (default)
  (2) DSA (sign only)
  (5) RSA (sign only)
Your selection? 1
DSA keypair will have 1024 bits.
ELG-E keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048) 2048
Requested keysize is 2048 bits
Please specify how long the key should be valid.
    0 = key does not expire
    <n> = key expires in n days
```


comment which were used to construct a "User ID" which identifies whose key this is in a human-friendly way.

The public and private keys are placed in files in the `.gnupg` directory:

```
$ ls -l .gnupg
total 32
-rw----- 1 alex alex 9207 Sep 28 13:28 gpg.conf
-rw----- 1 alex alex 1178 Sep 28 13:35 pubring.gpg
-rw----- 1 alex alex 1178 Sep 28 13:35 pubring.gpg~
-rw----- 1 alex alex 600 Sep 28 13:35 random_seed
-rw----- 1 alex alex 1327 Sep 28 13:35 secring.gpg
-rw----- 1 alex alex 1280 Sep 28 13:35 trustdb.gpg
```

Alex then exports his public key to an ascii file like this:

```
$ gpg --export --armor -o alex.pub Alex
```

"Armoring" a key simply means representing it in an ascii form that could (for example) be printed out or transmitted as the text of an email. The final argument of the command ('Alex') is a fragment of the key's User ID, sufficient to identify the key. The file he ended up with, `alex.pub`, is the one he plans to give to Beth.

Beth goes through the same process, generating her own key pair and her own passphrase, and exporting the public key into a file call `beth.pub` ready to give to Alex.

Alex and Beth now exchange their public GPG keys. Alex copies Beth's file to his computer and imports it into his GPG keyring like this:

```
$ gpg --import beth.pub
```

Beth similarly imports Alex's public key into her keyring.

At this point, if Alex lists his keys he will see the public half of his own key, and the one he imported from Beth:

```
$ gpg --list-keys
/home/alex/.gnupg/pubring.gpg
-----
pub 2048D/B4DF979C 2010-08-02 [expires: 2010-08-30]
uid                               Alex Example (Demo User A) <alex@example.com>
```

```
sub 2048g/626C246D 2010-08-02 [expires: 2010-08-30]
pub 2048D/F6CA4978 2010-08-02 [expires: 2010-08-30]
uid Beth Example (Demo User B) <beth@example.com>
sub 2048g/07DFB736 2010-08-02 [expires: 2010-08-30]
```

There's one more thing Alex needs to do before he can use Beth's public key to send her a confidential message. He needs to sign Beth's key to say that he believes it really is hers:



```
$ gpg --sign-key Beth
```

Alex can now use Beth's public key to send her an encrypted message:



```
$ gpg --encrypt --armor --recipient Beth secret.txt
```

In this command `Beth` is any part of the key ID, sufficient to identify this key uniquely. At this point Alex has an encrypted file called `secret.txt.asc` which he can email to Beth. He knows that only Beth can read it, because only she knows the private key corresponding to the public key Alex encrypted it with.

Beth can decrypt this file like this:



```
$ gpg --decrypt -o poetry.txt secret.txt.asc
```

You need a passphrase to unlock the secret key for
user: "Beth Example (Demo User B) <beth@example.com>"
2048-bit ELG key, ID 07DFB736, created 2010-08-02 (main key ID F6CA4978)

```
gpg: encrypted with 2048-bit ELG key, ID 07DFB736, created 2010-08-02
    "Beth Example (Demo User B) <beth@example.com>"
gpg: Signature made Mon 02 Aug 2010 03:25:00 BST using DSA key ID B4DF979C
gpg: Good signature from "Alex Example (Demo User A) <alex@example.com>"
```

Alex's key pair is also important. He has already used his private key to sign (declare his trust in) Beth's public key. His public key, of course, would be used to encrypt a reply from Beth back to Alex.

Digital Signatures

Alex signs his message by creating a hash of the message and encrypting it with his private key. On receipt, Beth can decrypt the hash (using Alex's public key), compute her own hash of the message, and compare the two. If they match, Beth has high confidence that the message came from Alex and that it has not been modified since it was signed. Note that digital signatures do not provide confidentiality – that is, they do not encrypt the message. `gpg` hides the complexity behind a simple command option (`--sign`). It's possible for Alex to both encrypt and sign the

message:

```

$ gpg --encrypt --sign --armor --recipient Beth secret.txt
    
```

Then, when Beth decrypts it, the signature will automatically be checked. (At the end of the earlier example, note the report of whose public key the message was encrypted with, whose private key it was signed with, and that the signature matches the message.

GPG also supports its own public key infrastructure using a "web of trust" but that is beyond our present scope.

How PGP works

PGP uses a hybrid of symmetric and asymmetric cryptography. A random session key (used only for this one message) is chosen and is encrypted with the recipient's public key. The result is sent as part of the message. The session key is then used to encrypt the sender's plain text using a symmetric cipher (IDEA, by default) and the result is attached to the message. The recipient decrypts the session key using her private key, then uses it to decrypt the body of the message.

This approach has several advantages: first, symmetric ciphers are much faster to compute, and don't need such long keys. Also, if an eavesdropper does crack the session key, it won't help him read any subsequent messages. Finally, because the asymmetric cipher is used only to encrypt a short (and random) value, a cryptanalyst would have very little hope of cracking it.

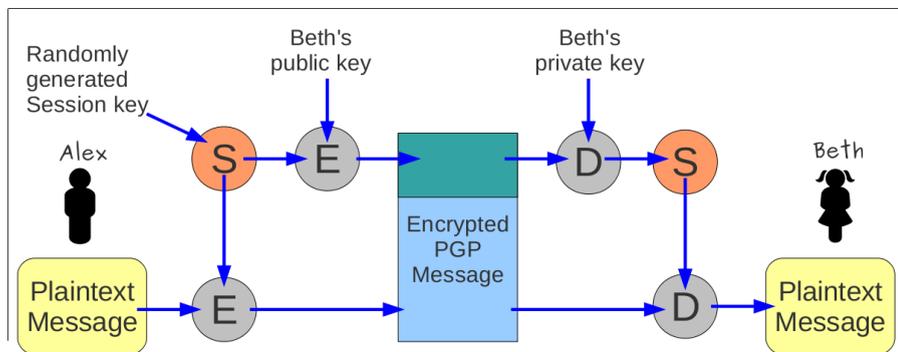


Figure 110.3-6: How PGP works

The following is a partial list of the used files, terms and objectives:

- ssh
- ssh-keygen
- ssh-agent
- ssh-add
- ~/ .ssh/id_rsa and id_rsa.pub
- ~/ .ssh/id_dsa and id_dsa.pub
- /etc/ssh/ssh_host_rsa_key
- ssh_host_rsa_key.pub
- /etc/ssh/ssh_host_dsa_key
- ssh_host_dsa_key.pub

[ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification] Created during the initiative "ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software in Africa", a programme of FOSSFA and GIZ. For more information see www.ict-innovation.fossfa.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.

• ~/.ssh/authorized_keys

• gpg

• /etc/ssh_known_hosts

• ~/.gnupg/*

Chapter III: FOSS TRAINING

Created by: Content Creation Community for the ict@innovation training material

Free your IT-Business in Africa! Advanced Training Material on African Free and Open Source Software (FOSS) Business Models for IT-SMEs

Version / Date : ICT_INNO_17/ May, 2010

Introduction

FOSS Training has become a novel business opportunity for new and existing businesses as FOSS is becoming more and more mainstream. As more and more mature FOSS applications emerge, businesses who want to use them require training. Growing investment in FOSS applications are expected to continue in following years leading to growing training needs and opportunities. The Open World Forum road-map for 2020¹ predicts that 40% of jobs in IT will be FOSS related and highlight the challenges associated with the shortage of skilled FOSS professionals. The authors assume a 2% growth rate of IT employment annually, which translates directly into 1.5 million jobs in Europe i.e. the creation of 1.2 million completely new jobs (Open World Forum Roadmap, p 69). Thus, the modules in this topic will build on previous topics on basic FOSS concepts in Module 1, African FOSS business experiences as demonstrated by the case studies in Module 2 and some FOSS business and management skills (Modules 3 - 5) to addresses the skills and knowledge required to organise FOSS training in the African context.

Learning Objectives

1. Understand some of the requirements for becoming a FOSS trainer.
2. Be able to identify and seize the opportunities that exist for FOSS training as a business.
3. Gain the knowledge and skills required to organise and provide FOSS training.
4. Appreciate the benefits of peer production of Open Educational Resources

¹ <http://www.2020flossroadmap.org/>

[ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification] Created during the initiative "ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software in Africa", a programme of FOSSFA and GIZ. For more information see www.ict-innovation.fossfa.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.

and Open Content.

5. Understand the crucial role of communication.

Sessions and Timetable

The entire content in this topic is estimated to be delivered in **2** days, with some variations within the modules. For instructional purposes, the content of this module can be delivered as proposed in the summarized table below.

Time	Session
9:00 – 10:30	<i>Completion of Module</i>
10:30 - 10:45	Coffee Break
10:45 - 12:15	<ul style="list-style-type: none"> • HOW TO BE A FOSS TRAINER
12:15- 13:30	Lunch
13:30 – 15:00	<ul style="list-style-type: none"> • FOSS TRAINING AS A BUSINESS
15:00 – 15.15	Coffee Break
15:15-17:00	Invited talks: Discussion of FOSS training Experience, FOSS Business in Africa, FOSS in Government, FOSS in Education
	<i>Next day</i>
9:00 – 10:30	<ul style="list-style-type: none"> • Organising Training
10:30 - 10:45	Coffee Break
10:45 - 12:15	<ul style="list-style-type: none"> • Training Material Development
12:15- 13:30	Lunch
13:30 - 15:00	<ul style="list-style-type: none"> • Open Educational Resources and Open Content
15:00 - 15:15	Coffee Break
15:15 - 17:00	<ul style="list-style-type: none"> • Training Communication Skills • Training Communication Skills • End of Module

Module III.1: How to be a FOSS Trainer

Duration:

1 hour

III.1.1 FOSS Trainer Characteristics

Any good trainer is natural communicator with strong technical skills to provide his/her students with the right tools to work in the industry. The winning combination of the 'ideal' FOSS instructor includes:

- Winning personality and ability to communicate.
- Practical experience in FOSS applications.
- IT training experience.
- Ability to engage participants and relate content to their situation.
- Be inquisitive and have a passion for training.
- They should have the ability to teach themselves a program and then use their skills, knowledge and attitude to facilitate the transfer of that knowledge.
- Have an understanding of the philosophical underpinnings of the FOSS movement.

Currently, no specific certification roadmap exists for a FOSS trainer.

Trainers should continuously act and reflect. They should always look back on how to improve delivery of the training, learn from course evaluations and aspire to have a better understanding of the applications they teach. FOSS trainers should not see the software as different from proprietary software.

It is also important that trainers use the software they are training, in one way or another. In so doing, they become conversant with features and functionalities that the software provides and can transmit this knowledge to their trainees. It also provides the trainer with a sense of authority and confidence.

Trainers should see training as a stage production:

- There is the audience (participants).
- There is a stage (training room).
- The performance by the actor (trainer conducting the course).
- The applause (the feedback).

III.1.2 Types of Training Interventions

Potential trainers may need their skills to be upgraded in different ways. We can broadly distinguish between three types of training interventions for FOSS trainers:

- There are those who have the technical knowledge of FOSS packages and have been using certain packages, but have not conducted any training of any nature. This type of person needs to acquire the knowledge and skills on how to conduct FOSS training for

adults.

- There are those who have the knowledge and skills on how to conduct ICT training for adults which they might have acquired through self-learning or having had attended a course, but have not conducted any FOSS training. Instead, they have conducted training in proprietary software. This type of trainer will need to be trained on FOSS packages.
- There are those who have conducted FOSS training courses and have the knowledge and skills on how to conduct training, but have not trained other trainers. This type will have to be trained on how to train other trainers.

Ideally, a new trainer should follow a trainer development program that may include delivering the content to a colleague and/or to a group of peers.

Part of the training curriculum for a train-the-trainer course may include topics of the [CompTIA's Certified Technical Trainer \(CTT+\)](#) curriculum or a similar programme.

Questions

1. List the winning combination of an "ideal" FOSS trainer
2. Name a certificate which exists for trainers
3. Discuss the three types of training interventions
4. Why is it important that trainers use the software they will be training?
5. Is there is difference between a FOSS trainer and proprietary software trainer?

Exercise

Trainers are going to brainstorm the qualities a trainer should possess. The trainer will write the contributions on a flipchart.

Module III.2: FOSS Training as a Business

Duration:

2 hours

III.2.1 Identifying FOSS Business Opportunities

FOSS training can be undertaken as part of an existing business function, academic pursuit in educational institutions or as a sponsored group activity. The type of training method chosen will influence the revenue, steps to be taken, facilities and content.

If FOSS training is undertaken as part of an existing business function, then it could:

- leverage the company's competitive position in the industry.
- access a ready pool of participants from the company.
- co-share facilities with other courses, thus the investment is low.

If academic institutions are involved in FOSS training, then

- it could target students who might not yet have loyalty to proprietary software.
- it could be incorporated in Computer Science courses.
- cost might be reduced in the acquisition of licenses for proprietary software

- If FOSS training is pursued as sponsored group activities, then
- it could include workshops, seminars, exhibitions
- it should be targeted
- the training duration should be short
- cost should be undertaken by the sponsor
- Selection of candidates for FOSS training can be done using
- role/function in organisation
- educational background
- identified need

FOSS training curriculum should be comprehensive and detailed. It should include all topics covered for the equivalent proprietary software.

III.2.2 Case Study

Arnold Pietersen (CECS) provides some practical examples of FOSS training he conducted or intend to conduct, below.

In 2006, Arnold came across Open Workbench (which is the same type of programme as MS Project). He started using it for CECS' project and was impressed by the programme. It dawned upon him that this might be a useful tool for NGOs. Arnold visited the websites of some of the major training companies in South Africa to see what they offer with regard to MS Project 2003 Level 1. He then modelled the Open Workbench course based on the MS Project 2003 course outline. This he thought would provide for benchmarking or comparability. Since then he has conducted numerous courses by sending e-mails to NGOs and CBOs to announce course dates. Participants are now requesting for the Level 2 course. The last course was conducted with 16 participants. With improved marketing he surmise more course and regular courses could be conducted. Participants attending are from across the board: NGOs, CBOs, government, individuals, schools. The course is being charged at about 80% of what a MS Project course would cost. CECS now want to target students at universities who studies project management.

CECS received some money from OSISA to develop an open source course for entrepreneurs and latched onto TurboCASH. CECS have been conducting numerous courses for the past three years. A contracted trainer is conducting the course who install, train, supports TurboCASH as a business. The organisation have had requests from individuals and organisations especially Cape Town and Durban to attend the training course. There are very few companies offering TurboCASH courses, let one NGOs.

Last year July CECS started conducting Web Design Training Using Joomla courses which proves to be very popular. The courses are well-attended. CECS also derive other opportunities from conducting this course such as organisations wanting to contract the organisation to migrate their websites, to conduct on-site training, individuals and organisations wanting to purchase manuals. The manual is still very much work in progress. It was a question of do we spend a year or two trying to develop a "perfect" manual or do we start with some material and then built upon that as we gain experience. A scan was undertaken regarding the Joomla training environment in South Africa before CECS embarked on the training.

Arnold is now in the process of putting together a course for Ubuntu Linux for absolute beginners. Participants will bring their laptops to the course. The course will map to a certain

extent to say a Windows XP Level 1/Beginners. He thinks that critical for the course is showing people how to install Ubuntu. When people go wrong at the partition stage, they then blame Ubuntu Linux for all their woes. Thus, we need to give them a solid understanding regarding installing Ubuntu Linux. This will be a pure end-user course.

It is difficult to estimate the demand for FOSS training. FOSS training can attract learners if the application has been widely adopted in the industry. As an example FREEBSD has been promoted by AfNOG over the years thus FREEBSD training is likely to attract more users. People are prepared to pay for courses, whether it is FOSS or proprietary, provided that these courses address their real business needs.

III.2.3 Identifying Training Opportunities

Training opportunities can be identified in the following ways:

- Surveys should be done to identify training needs.
- Identifying popular applications (e.g. by looking at downloads from sourceforge.net and freshmeat.net).
- Subscribing to newsletters, mailing lists and participating in relevant forums.
- Attending (either actively or passively) ICT conferences, workshops and other events such as Software Freedom Day.
- Accessing market reports, e.g. Gartner, Government, etc.
- Identify FOSS applications that may satisfy market needs.
- Participating in relevant tenders, request for proposals, requested for interest, pre-qualification exercises, etc.

III.2.4 Marketing of Training Courses

- Direct advertising through local press, magazines and professional publications.
- Register and contribute in forums, mailing lists, blogs, etc.
- Taking advantage of ICT conferences, workshops and other events such as Software Freedom Day as a marketing opportunity.
- Contributing articles to the local press and other media houses.
- Maintaining a presence on website portals that bring together trainers and potential trainees (e.g. <http://www.flosslit.org.za/>).
- Organise computer literacy events in schools, educational institutions, etc.

III.2.5 FOSS Certifications

What follows below are courses with an international repute. We provide a brief description of the courses.

OpenICDL

<http://www.icdl.org.za>

OpenICDL refers to the International Computer Driving Licence based on open source software.

OpenICDL is a test of practical skills and competencies and consists of seven separate modules covering computer theory and practice. To achieve OpenICDL certification, a Candidate must successfully pass a test in all seven modules.

OpenICDL Module 1 is a theoretical test of computing knowledge at a general level and modules

2-7 are practical skills tests. The following are the modules:

- Concepts of Information Technology
- Using the Computer and Managing Files (Ubuntu Linux)
- Word Processing (OpenOffice.org Writer)
- Spreadsheets (OpenOffice.org Calc)
- Database (OpenOffice.org Base)
- Presentation (OpenOffice.org Impress)
- Information and Communication (Mozilla Firefox & Mozilla Thunderbird)

You must be registered with the [ICDL Foundation](#) in order to offer the OpenICDL.

Linux Professional Institute Certification (LPIC)

<http://www.lpi.org>

The Linux Professional Institute Certification (LPIC) program is designed to certify the competency of IT professionals using the Linux operating system and its associated tools. It is designed to be distribution neutral, following the Linux Standard Base and other relevant standards and conventions.

The LPIC program is designed in multiple levels. Determining which tasks were suitable to each level was done using a "Job Task Analysis" (JTA) survey. As with all of the LPIC exam development processes, the JTA was developed and executed using recognized psychometric processes, to ensure its relevance and high quality.

The LPIC program consists of three levels of certification: LPIC-1, LPIC-2 and LPIC-3.

Junior Level Linux Professional (LPIC-1)

- Pre-Requisites: None
- Requirements: Passing Exams 101 and 102
- Overview of Tasks: To pass Level 1 someone should be able to:
- Work at the Linux command line
- Perform easy maintenance tasks: help out users, add users to a larger system, backup & restore, shutdown & reboot
- Install and configure a workstation (including X) and connect it to a LAN, or a stand-alone PC via modem to the Internet.



Advanced Level Linux Professional (LPIC-2)

- Pre-Requisites: You must have an active LPIC-1 certification to receive LPIC-2 certification, but the LPIC-1 and LPIC-2 exams may be taken in any order.

- Requirements: Passing Exams 201 and 202
- Overview of Tasks: To pass Level 2 someone should be able to:
- Administer a small to medium-sized site
- Plan, implement, maintain, keep consistent, secure, and troubleshoot a small mixed (MS, Linux) network, including a:
 - LAN server (samba)
 - Internet Gateway (firewall, proxy, mail, news)
 - Internet Server (webserver, FTP server)
 - Supervise assistants
 - Advise management on automation and purchases



Senior Level Linux Professional (LPIC-3)

The LPIC-3 Certification program represents the culmination of LPI's Certification Program.

LPIC-3 is designed for the "enterprise-level" Linux professional. The program has been developed with the input of hundreds of Linux professionals from around the globe and with input from some of the world's leading technology companies. It also represents the highest level of professional, distribution-neutral Linux certification within the industry.

The LPIC-3 program consists of a single exam for LPIC-3 "Core" designation, with several electives.



Ubuntu Certifications

<http://www.ubuntu.com>

The following are the Ubuntu Certifications:

- Ubuntu Certified Professional
- Deploying Ubuntu Server in an Enterprise Environment
- Ubuntu Desktop Training

Ubuntu Certified Professional

The Ubuntu Certified Professional (UCP) is a training certification based on the LPI level 1 certification. To earn the UCP, candidates are required to pass the LPI 101, LPI 102 and the Ubuntu 199 exams. Exams can be taken in any order. Two, five day courses, Ubuntu Professional Courses 1 & 2, will assist System Administrators to pass the required exams and achieve the Ubuntu Certified Professional certification.

The certification tests student's ability to:

- Install and configure Ubuntu systems
- Perform routine administration tasks: boot and shut down the system, manage user accounts and file systems, and maintain system security
- Configure network connectivity and key network services
- Work productively at the Linux command line

Deploying Ubuntu Server in an Enterprise

This hands-on course will provide participants with the skills they need to deploy, configure and maintain secure Ubuntu Server Edition within the enterprise infrastructure. The course is based

on Ubuntu 8.04 LTS and will help system administrators to implement services at an advanced level. Extensive lab exercises in a multi-server virtual machine environment will help attendees put their new skills into practice.

If you are an experienced Linux or Unix system administrator working in an organisation, which is about to, or has already, deployed Ubuntu desktop and servers in the office, this course is for you!

After completing this course, you will be able to:

- Install and deploy an Ubuntu Server in an enterprise environment
- Use Debian package management tools to:
- Install, configure, update and upgrade packages
- Set up a repository
- Manage a mirror service
- Automate updates
- Monitor server status remotely
- Define and implement a Backup strategy
- Create and deploy virtual Machines using KVM and libvirt
- Manage directory services and authentication using OpenLDAP and Kerberos
- Keep servers as secure as possible

Ubuntu Desktop Training

This course provides both home and office users with hands on training on Ubuntu. No prior knowledge of Ubuntu is required, although computer literacy is assumed and is a pre-requisite. Ubuntu 8.04 LTS must be installed on the computer hard disk before starting this course. The Ubuntu desktop course is designed to be modular. If all lessons are studied in a classroom, it should be completed within two full days. However, topics and lessons can be selected as required and a day's content designed to suit the key learning objectives.

Questions

1. List companies, organisations or schools who is conducting FOSS education and training in your country
2. Critique or support the case study in 6.2.2
3. Make a list of other FOSS certifications which might be available
4. What are the obstacles, if any, you might perceive in people not wanting to attend FOSS training courses?
5. Which FOSS certification is more recognisable in your country?

Exercise 1

Participants will engage in an individual exercise the list the benefits of training. Their responses will result in a plenary/discussion.

Exercise 2

Participants will list their knowledge of FOSS certifications/training. Participants with similar lists will be grouped to discuss and present the benefits of each certification/training.

Module III.3: Organising Trainings

Duration:

2 hours

III.3.1 Course Design and Curriculum Development

- Explore curricula for equivalent software (e.g. Open Workbench, OpenProj and MS Project) and use these as a benchmark (aim higher)
- Provide a benchmark for comparability
- Tailor the curriculum to what is termed as the unit standards (smallest element of learning – a case of South Africa)
- Develop lesson plans
- Curricula should have clear learning objectives
- Adapt where feasible from existing commercial curriculum

III.3.2 Course Material Development

Professionally designed and pedagogically sound course material will be important. Some of the issues that should be considered are:

- The course material should be peer-reviewed by placing it on a wiki for comments and input.
- Exercises should be relevant
- The course materials should be graphically-rich and be of a step-by-step nature
- The layout of the course material should be done in FOSS desktop publishing (DTP) or graphic design program
- The licensing of the course material depending on the available types of licenses
- The continuous updating of the material in relation to new versions of the software being released
- The course material should be translated in the mainstream languages in Africa.
- A facilitators guide should also be developed
- The development of the course material should take lessons from existing courseware for similar types of software.

III.3.3 Licensing of Course Material

Creative Commons [<http://www.creativecommons.com>]

The Creative Commons licenses provide everyone from individual authors and artists to large companies and institutions a simple, standardised way to grant copyright permissions to their creative work such as the development of course material. The Creative Commons licenses enable people to easily change their copyright terms from the default of “*all rights reserved*” to “*some rights reserved*.”

For example, a developed training manual could be licensed under a Creative Commons Attribution-ShareAlike licence. This license lets others remix, tweak, and build upon your work even for commercial reasons, as long as they credit you and license their new creations under the identical terms. This license is often compared to free software licenses. All new works based on yours will carry the same license, so any derivatives will also allow commercial use.

GNU Free Documentation License (GNU FDL or simply GFDL)

<http://www.gnu.org/copyleft/copyleft.html>

The purpose of this License is to “*make a manual, textbook, or other functional and useful document “free” in the sense of freedom*”: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or non-commercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

Using the above two URLs and your experience, copy and complete the table below:

Type of creative commons Licence	Usefulness for	
	FOSS business	FOSS Training
1)		
2)		
3)		

III.3.4 Preparing Yourself for Class

Preparations can be the most important part of your instructional day. Time spent before trainees arrive often has a direct effect on all aspects of the day.

As a trainer, you should prepare in three keys areas:

- yourself
- the classroom, and
- the trainees

As you prepare yourself for a class, consider the following areas for preparation:

1. Materials

[ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification] Created during the initiative “ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software in Africa”, a programme of FOSSFA and GIZ. For more information see www.ict-innovation.fossfa.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.

Checklists are an excellent way to guarantee that you have all of the necessary materials for the day. A checklist contains two types of items: The first category includes obvious items that you'll never forget, such as your trainer's manual. The second category includes unique supplies, like extra kokis. When preparing to teach, trainers often forget materials from the second category.

2. Instruction

There are two areas in which you should be prepared: knowledge of the content and presenting the materials effectively. The most common challenge that you will face is trying to learn the content fluently of the course is to focus on the material that must be learned.

To prepare yourself for a teaching a new course:

1. Work through the course as many times as needed. Write any questions you may have about the content, but do not look for answers yet. Many of your questions may be answered later in the course.
2. Networking with another trainer(s). Set up a time to meet with another trainer(s) who also teaches the course and bring the unanswered questions you have.
3. Research. Even after meeting with another trainer(s), you might not feel comfortable with all the areas. Now it's time to research those areas.

Work on Your Presentation

The following are four approaches to preparing your presentation. Unlike the steps mentioned above, these are not sequential order, but rather in order of effectiveness from least to most:

- Mirroring - Alone, observe yourself in a mirror.
- Verbal - Informally, with a friend or co-worker (preferably someone who is not familiar with the course content) as an audience.
- Desert Run - Alone, in an empty classroom.
- Dry Run - In a classroom with friends or co-workers acting as trainee.

Rehearsal

You need to know your materials thoroughly before you start to train. A problem might arise where you focus on the training materials but not on how you are going to present it or how the trainees are going to use them. By looking at aims, objectives and purpose you will avoid this trap.

Timing

Timing is one of the main problems with new or inexperienced trainers. How long does it take? To some extent it takes as long as you have got - but this is an unsatisfactory answer. How then do you plan a day's training?

Start with a page of A4 paper and put the start time at the top and the end time at the bottom.

Then you need to work on the following estimates; depending on the size of the group:

- Allow 15 - 30 minutes at the start for introductions and housekeeping
- Allow 15 - 20 minutes for expectations and fears
- Allow 15 - 30 minutes at end for final review (and evaluation).
- Slot in TEA, LUNCH and COFFEE BREAKS.
- Examine AIMS and OBJECTIVES.
- Write these exercises and allocate approximate timings. Do this by doing the exercise

yourself and multiplying the time you took by 5 to get to a realistic time for your trainees

- Prepare a spare exercise for every session (for those who go faster than everyone else)
- Fit exercises and handouts (including reading time) into plan.
- Fit topic/content explanations into "missing" gaps!!
- Set one exercise per objective.

Tip: For training sessions of less than one day use the same technique but reduce the introduction and review times. Obviously you may not lose so much time with breaks, but remember that everyone needs to have a break every 90 minutes!

With a carefully written aim, objective and purpose for every session you will be able to rehearse your sessions in advance.

Notes

Ensure that you have brief notes on pages or card that you can refer to where necessary. Do not write out a total script: unlike an actor you are in control of what you say, not merely repeating someone else's words.

Materials and Props

Just as an actor has materials and props, you too need these as a trainer. Ensure that you specify what equipment you need in advance and that you have back-up if something fails.

Always get to the room 30 minutes before the trainees arrive. You will need to set out the training manuals, check if the equipment works, write up some flip charts, find out where the toilets are and about fire drills and emergency exits, lunch and break times and to settle yourself before your 'performance'.

Additional props that most trainers carry are such things as:

- Flip chart pens
- White board pens
- Spare exercises
- Tent card with the trainees names on it
- Pair of scissors

www.cecs.org.za

III.3.5 Preparing Your Training Room

Physical Setup

How you arrange the furniture in your classroom can effect both the learning environment, and the type of interaction that can take place among the trainees. Seating can affect the availability of an instructor to a trainee, and can also influence the effectiveness of media, such as overheads or trainer screen. Instruction can be facilitated or hindered depending on the amount of interaction allowed between trainees.

An effective classroom setup and tear down checklist helps guarantees a successful classroom for the next class. Trainees and instructors expect occasional hardware problems, but a trainer should do everything possible to control the classroom environment.

III.3.6 Beginning the Training Session

Addressing Trainees Expectations

It's important that you address trainees' expectations as early in the day as possible. What you do in those first few moments of the day can have significant impact on the rest of the day. The following are sum of examples on how you can begin your day.

- Discuss the facilities - This lowers trainees anxiety about the new environment bar outside pressure (phone for outside calls, rest room and so forth). Be sure to discuss the environment both outside and inside the classroom.
- Write your schedule on board - This allows trainees to see a direction for a day and to get feel that they are in the right place.
- Preview training manual - This shows trainees the backup support materials for the information about to be taught and can lower anxiety. Be sure to review the topics to be covered as well as the way in which the training manual is to be used.
- Introduction - These help to create an open environment. Encourage the trainees to introduce themselves by sharing such information as their names, the schools they from, the grades they are teaching, their computer experience and their expectations. This will also help them to relate to their peers who come to class with similar abilities.
- Take this opportunity to encourage questions and to establish a friendly, relaxed atmosphere. Indicate whether they should take notes or move ahead in the course manual, and hat their primary focus for the day should be.
- You might also suggest your preference about how the equipment is treated and whether it's appropriate to bring refreshment to the workstation. Indicate when it is appropriate to interact with another trainee.

III.3.7 Ending Your Training Session

Your training day should not just stop; it should be end with closure. Just as your initial statements set the tune for the day, your closing statement should complete the impression of the successful day of training. In addition to exiting from the software, answering final questions, and complementing the evaluation form you can:

- Discuss "What's next". Encourage trainee to arrange practice time and recommend that they find a job-relevant task to practice with. Review outlines for advanced-level courses.
- Advertise continuing support service, if available. Encourage learners by reinforcing the use of books and online services as effective help system.

Questions

1. List some of the steps involved in the design of a course/curriculum
2. List and discuss two licenses available for course material
3. Why should the development of FOSS training content take cognisance of the commercial world for proprietary software
4. Why is it important to have course material in indigenous languages?
5. Discuss how you should prepare yourself for class.

Exercise

Participants will brainstorm in groups of 4 preparing a checklist for organising training. The responses of the participants will be captured on a flipcart, which they will put on the wall.

Module III.4: Open Educational Resources and Open Content

Duration:

2 hours

6.4.1 Open Educational Resources

http://en.wikipedia.org/wiki/Open_educational_resources

Open educational resources (OER) are an Internet empowered worldwide community effort to create an education commons.

The term "open educational resources" was first adopted at UNESCO's 2002 Forum on the Impact of Open Courseware for Higher Education in Developing Countries funded by the William and Flora Hewlett Foundation. Open educational resources are educational materials and resources offered freely and openly for anyone to use and under some licenses to re-mix, improve and redistribute. Open educational resources include:

- Learning content: full courses, course materials, content modules, learning objects, collections, and journals.
- Tools: Software to support the creation, delivery, use and improvement of open learning content including searching and organisation of content, content and learning management systems, content development tools, and on-line learning communities.
- Implementation resources: Intellectual property licenses to promote open publishing of materials, design-principles, and localisation of content.

History

From 24 October to 2 December 2005 the UNESCO on-line Forum Open course content for higher education took place.

In September 2006, the Third Annual Open Education Conference (Community, Culture and Content) was held in Logan, Utah. The last conference was held on September 24-27, 2007 in Logan, Utah.

In June 2007, educators at the iCommons iSummit in Dubrovnik joined the open movement worldwide to showcase emerging open education initiatives and to explore ways to better create, share and evolve open educational materials.

In January 2008 The Cape Town Open Education Declaration was published.

OER and Open Source

Since 2005 there has been a marked increase in the Open Educational Resource (OER) movement and in Open Educational Licenses (like Creative Commons). Many of the projects on OER were funded by the William and Flora Hewlett Foundation, and partly also by the Shuttleworth Foundation that focuses on projects concerning collaborative content creation. There has been a strong international debate on how to apply OER in practice and the UNESCO chaired a vivid discussion on this through its International Institute of Educational Planning (IIEP).

Alignment With Open Source Software Community

By the second half of 2006 it also became clear to some of the forerunners that OER and

Free/Libre Open Source Software (FLOSS) do somehow belong together. As a result, the discussion groups of IIEP on OER and FOSS were merged and forces were further joined through mergers with a related OECD campaign.

What has still not become clear by now to most actors in the OER domain is that there are further links between the OER and the Free / Libre Open Source Software (FLOSS) movements, beyond the principles of “FREE” and “OPEN”. The FOSS model stands for more than this and, like e.g. Wikipedia, shows how users can become active “resource” creators and how those resources can be re-used and freely maintained. In OER on the other hand a focus is still on the traditional way of resource creation and participant roles.

Best Practices and Communities for OER Contributors

FOSS communities are today known for producing good quality software using a different development approach than proprietary software producers. FOSS is built by a community of volunteers and might be backed by companies that generate their revenues by providing services related to the software. In more recent years FOSS communities also gained attention for their community production and support models and regarding their way of knowledge creation and learning. FOSS communities possess many characteristics that educational communities could benefit by adopting:

1. Open and inclusive ethos: everyone can participate, no charges, no deadlines, life long participation
2. Up to date content; everyone can add, edit and update the content
3. Materials are usually the product of many authors with many contributions from people other than authors
4. Frequent releases and updates where product features and community structures are the result of a continuous re-negotiation / reflection process within a continuous development cycle
5. Prior learning outcomes and processes are systematically available through mailing lists, forums, commented code and further instructional materials (re-use)
6. A large support network; provided voluntarily by the community member in a collaborative manner nearly 24/7
7. Free Riders (lurker) welcome paradox – the more the better
8. New ICT solutions are adapted early by the community

Education professionals may be aware that FOSS-like principles can benefit education, but there has been no systematic and comprehensive approach to map and transfer those principles, or to develop new educational models and scenarios around them. The European Union funded FLOSSCom project is likely to be the first attempt to map the open source landscape from an educational point of view, but further research and work still remains to be done.

However, Teachers Without Borders, a non-profit based in Seattle, is currently developing a new OER website where members can take courses, discuss their findings with people around the world, and publish their work, all on the same website. Their goal is to connect educators and learners from around the world and give free access to a wide variety of courses, thus helping to close the education divide.

III.4.2 Open Content

<http://www.wikipedia.org>

Open content, a neologism coined by analogy with "open source", describes any kind of creative work published in a format that explicitly allows copying and modifying of its information by anyone, not exclusively by a closed organization, firm or individual. The largest open content

project is Wikipedia.

Technical Definition

Work on a technical definition for open content has been undertaken by the Open Knowledge Foundation. The Open Knowledge Definition (OKD) gives a set of conditions for openness in knowledge - much as the Open Source Definition does for open-source software. Content can be either in the public domain or under a license which allows re-distribution and re-use, such as Creative Commons Attribution and Attribution-Sharealike licenses or the GFDL. It is worth noting that the OKD covers open data as well as open content.

History

It is possible that the first documented case of open content was the Royal Society, which aspired to share information across the globe as a public enterprise. The term "open content" was first used in the modern context by David Wiley, then a graduate student at Brigham Young University, who founded the Open Content Project and put together the first content-specific (non-software) license in 1998, with input from Eric Raymond, Tim O'Reilly, and others.

Questions

1. Define Open Educational Resources.
2. Define Open Content.
3. What does Open Educational Resources include?
4. What can educational communities learn from FOSS communities?
5. Contrast Open Educational Resources with Open Content.

Exercise

Participants should brainstorm in groups of 5 the importance of open educational resources open content for the advancement of free/libre and open source software. A rapporteur will provide feedback in a plenary.

Module III.5: Communication Skills

Duration:

4 hours

III.5.1 The Four Learning Styles

As a trainer, you will be working with trainees of a variety of learning styles different from your own. Knowing your learning style means you can work with it to deliver a training program that uses your strengths and meets the needs of your trainees.

If you are the Divergent Learning Style...

You are best at using the Concrete Experience (CE) and Reflective Observation (RO) steps in learning. If this is your style, you probably have the ability to view specific situations from many perspectives. For example, you may enjoy brainstorming and small group discussions. You also like to gather information and probably have broad interests. Your tendency may be to watch events rather than participate in them.

To increase your learning power you also need to place emphasis on the Abstract Conceptualization (AC) and Active Experimentation (AE) steps in the learning process. This means forming conclusions from your information, planning the application of these conclusions and actually implementing them.

For example, after watching a role play or listening to a discussion, summarise your observations into clear conclusions. Then decide how and when to test these conclusions in your own situations. Establish criteria to evaluate if the new idea really worked. Do this at the end of every activity in which you are an observer.

To further increase your learning power, take a more active role in the workshop than you might normally choose. Volunteer to be in the role plays, or to lead group discussions. This may be uncomfortable at first but it will give you an opportunity to experiment with your conclusions. It will also give you more experience with trial-and-error learning, something you may tend to avoid in real-life situations.

You may find it useful to discuss workshop topics with someone who has a Converger learning style. This person will help you see possible conclusions and applications you might overlook. You in turn may help them see information they might overlook, and develop more perspective.

You may have a tendency to concentrate on the human side of problems or topics or exercises. This reflects your ability to understand or to sympathise with others' feelings or points of view, but you may also have a tendency to avoid drawing conclusions about the quantitative or technical aspects of the situation.

Try to develop these skills:

- Collecting and analyzing numerical data.
- Looking for overall patterns in any feedback you get.
- Putting your own feelings aside for a moment and taking a more objective look.

If you are the Assimilative Style...

You are best at using the Reflective Observation (RO) and Abstract Conceptualization (AC) steps in the learning process. If this is your style, you have the ability to create theoretical models

(ideas that predict outcomes and descriptions of how different factors interact). You most likely enjoy inductive reasoning and distilling disparate observations into logical explanations. To increase your learning power, you also need to place more emphasis on the Active Experimentation (AE) and Concrete Experience (CE) steps in the learning process. This involves speeding up your learning cycle by moving into action sooner.

For example, after watching a role play or listening to a discussion, think about ways to immediately apply your conclusions. Look for opportunities to test your new idea during the workshop and personally experience the results. This may require you to conceptualize smaller scale experiments, not the large scale efforts you may prefer. To further increase your learning, be more aware of the feelings and reactions of individuals (including yourself). You may have a tendency to discount intuitive or emotional information. However, much can be learned from a person's tone of voice, facial expressions, and other body language. Much of this data is preliminary in nature and hard to analyse in a logical fashion, but it provides an early warning about how things are going or if an idea has been understood.

You may have a preference for examining the quantitative or "thing" aspects of a situation. Your conclusions may be based primarily on policies, official relations, or formulas developed in other situations. This can cause you to be over-cautious about experimenting and miss opportunities for learning. No two situations are exactly alike. Put more effort into trying ideas, skills, or concepts. Then pay attention to the way things actually happen. It is often different than the way things are "supposed" to happen. Your ability to deal with non-quantitative data will increase if you get involved in interpersonal activities (role plays, simulations, discussions) more frequently. Take an active role and express your feelings. Others will do the same and this will give you experience handling this data.

Enter into discussions with people whose primary learning style is Accommodative. Note the value they place on intuition as a decision-making device. Research shows that in many situations intuition is more effective than logic. Try to implement their suggestions even if they can't provide a supporting rationale, or perhaps you can help them think through the rationale.

Try to add these learning skills:

- Seeking and exploring possibilities
- Influencing others
- Being personally involved
- Dealing with the people side of issues you work on, particularly how to get the support of key individuals whose help you will need

If you are the Convergent Style...

You are best at using the Abstract Conceptualization and Active Experimentation steps in the learning process. If this is your style, you have the ability to find practical application for ideas, concepts, and theories. In particular, you enjoy situations in which there is a single of best answer to a question or problem. You may usually assume there is one best answer and use technical analysis to reveal it. You too may prefer to deal with technical issues rather than people issues.

To increase your learning power you need to place even more emphasis on the Concrete Experience and Reflective Observation steps in the learning process. This means placing a higher value on gathering and understanding non-quantitative information by looking at a situation from different perspectives. The result may seem to slow your learning process. In fact, it will speed the long-term accuracy by ensuring you are learning the most important things.

For example, while watching a role play or listening to a lecture, you may be thinking about how the topic or technique applies to your situation. Before making a decision, however, try to get other people's perspectives. Listen to their ideas, comments, and questions. You may discover the situation has elements you weren't considering. This may influence how you apply your

learning.

To further increase your learning, try to take a less active role in the workshop than you might usually take. Spend some time really listening to others' ideas. Try to see the world as they see it, to understand their feelings and values. Play an observer role from time to time and avoid making judgments or decisions about how well others are doing. Instead, try to understand why they are saying or doing something. This may lead you to new and eventually useful information.

You will find it important to discuss workshop topics with someone who has a Divergent learning style. This person will see both questions and possibilities you might tend to ignore or avoid. You may help them see how to apply some of their ideas.

You may have a tendency to concentrate on the "things" side of problems, topics, or exercises. You may underestimate the impact people's values and emotions have on the way systems actually work. Avoid coming to quick conclusions.

Try to add these skills:

- Listening with an open mind
- Gathering information
- Imagining the implications of situations

If you are the Accommodative Style...

You are best at using the Active Experimentation (AE) and Concrete Learning (CL) steps in the learning process. If this is your style, you have the ability to learn primarily from hands-on experience. You probably enjoy carrying out plans and involving yourself in new and challenging experiences.

Your tendency may be to act on intuition and gut feel rather than careful analysis. When a thoughtful approach does not seem to be working out, you will be quick to discard it and improvise.

To increase your learning power, you need to place even more emphasis on the Reflective Observation (RO) and Abstract Conceptualization (AC) steps in the learning process. This means collecting and analyzing more information about the results of your efforts. Your batting average in the trial and error method of learning will increase if you learn more than you currently do from each of your trials.

When watching a role play, you may feel frustrated and prefer to be doing the plan yourself. Your tendency might be to think of how you would do the same activity. However, to develop your Reflective and Abstract skills, you should examine other, less personal aspects of the situation. Here are questions you might ask: What basic point does the exercise prove or disprove? What other information aside from your personal experience do you have that relates to the same topic? Does this exercise help you understand why certain techniques work (not just what the techniques are or how to use them)? To further increase your learning power try to take a less physically active part in the workshop than you might normally choose. Be more mentally active. Volunteer to be an observer in some exercises, not a doer. This will give you an opportunity to reflect on other people's experiences and learn from their trial and error.

You will find it useful to discuss workshop topics with someone who has an Assimilative learning style. This person will help you see information you might otherwise miss. They will also help you see the hidden logic and patterns in situations. You can often use this perspective to guide your intuition. You in turn can help them see new possibilities and opportunities to try out their ideas.

You may have a tendency to concentrate on the urgent aspects of a situation, favouring immediate utility over long-term understanding. To increase your learning, keep notes on your

experiences, analyze them, and look for patterns. In other words, look for the forest as well as the trees. Take more time to get other people’s perspective on what has happened (or what you are about to do) during the workshop.

The particular skills you want to add are:

1. Organizing information
2. Building conceptual models
3. Testing theories and ideas

Similarities and Preference Patterns in your Group

Group	Ways to include this group in training
Accommodative Style	
Convergent Style	
Divergent Style	
Assimilative Style	

My Training Style

Look at what you plan to do in your 15-minute workshop. Have you chosen something that fits in well with your own learning style?

Here is a review of each style.

The Converger (AC / AE)

- Practical application of ideas
- Good at closed-ended, “thing” problems

- Can focus on specific problems
- Can apply concepts
- Relatively unemotional (engineers/accountants)

The Diverger (CE / RO)

- Imaginative ability
- Views concrete situations from many points of view
- Brainstormer
- People-oriented
- Emotional (personnel managers)

The Assimilator (AC / RO)

- Creator of theoretical models
- Inductive reasoner
- Likes abstract concepts
- Can assimilate separate observations into an integrated
- Explanation (research and planning departments)

The Accommodator (CE / AE)

- Doer
- Carries out plans and experiments
- Risk-taker
- Adapter
- Likes to go by the seat of the pants (marketing and sales)

Individual Exercise

As I reflect on my most successful experience as a trainer, I remember...

What I like best about being a trainer is...

My favourite instructional technique is...

What I find most difficult about being a trainer is...

About The Trainer Type Inventory

Each of us is influenced not just by our own learning style, but also by our training type.

As agents of change, most trainers are continually aware of changes in themselves. As you facilitate growth and development in others, you struggle to improve yourself, and to become a more effective leader, planner, presenter, and facilitator.

Once you have recognized that learners have preferences for the way they learn, you become more motivated to help them:

- Learn even better in their own preference, where they are comfortable.
- Become more willing to expand their comfort level.
- Try other new techniques and new behaviours to enhance their own learning. The Trainer Type Inventory

The Trainer Type Inventory (TTI) is designed to help you as a trainer identify your preferred training methods in order to:

[ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification] Created during the initiative "ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software in Africa", a programme of FOSSFA and GIZ. For more information see www.ict-innovation.fossfa.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.

- Identify the areas in which you have the greatest skill and expertise, so you can share this expertise with other trainers in this workshop.
- Identify the areas where you will want to increase your skills, thereby increasing your ability to address all aspects of the learning cycle.
- Change and growth can become more meaningful, more useful, and more exciting for everyone involved when we grow as trainers, right along with the people we are training.

Malcolm Knowles (1984) says that adults will learn “no matter what.” Learning is as natural as rest or play. With or without workbooks, visual aids, inspiring trainers or classrooms, adults will manage to learn. Trainers can however make a difference in what people learn and how well they learn. If adults (and, many believe, children as well) know when they are learning and if the reason fits their needs as they perceive them (the “So what?”) they will learn quickly and deeply. There have been other attempts to categorize how trainers train. At first it was thought that trainers would prefer to train others in the style they preferred for learning. However, research has since discovered that there is very little significant relationship between a trainer’s own learning style and training–style preferences.

Introduction to the TTI

The Trainer Type Inventory identifies four different training types: a Listener, a Director, an Interpreter, and a Coach. Generally we have a preference for one type or another, even though we need all four types to be a successful trainer.

The Training Type Inventory (TTI) has often been administered in conjunction with Kolb’s Learning Style Inventory. It has been used often enough to have some validity for trainers. It is not a psychological tool, but an exercise to help us recognize our own specific trainer development needs.

Completing Trainer Type Inventory

There are twelve sets of four words or phrases listed below. Rank order the words or phrases in each set by assigning a 4 to the word or phrase that most closely applies to or reflects your personal training style, a 3 to the word or phrase that next best applies to your training style, a 2 to the one that next applies to your training style, and a 1 to the word or phrase that is least descriptive of your training style.

You may find it difficult to rank the items. Be assured that there is no right or wrong answers; the purpose of the inventory is to describe the style in which you train most often, not how effectively you train.

Question	Choices	Your Ranking
1.	a) Subgroups	
	b) Lectures	
	c) Readings	
	d) Lecture discussions	
2.	a) Showing	
	b) Perceiving	
	c) Helping	
	d) Hearing	

3.	a) Symbols	
	b) Actions	
	c) People	
	d) Instructions	
4.	a) Small group discussions	
	b) Free expression	
	c) Little participation	
	d) Time to think	
5.	a) Immediate personal feedback	
	b) Objective tests	
	c) Subjective tests	
	d) Personal evaluation	
6.	a) Expert	
	b) Scholar	
	c) Advisor	
	d) Friend	
7.	a) Theory	
	b) Practical skills	
	c) Application to real life	
	d) New ways of seeing things	
8.	a) Coach	
	b) Listener	
	c) Directory	
	d) Interpreter	
9.	a) Seeing “who”	
	b) Telling “how”	
	c) Finding “why”	
	d) Asking “what”	

10.	a) Processing	
	b) Generalizing	
	c) Doing	
	d) Publishing	
11.	a) Lead them to understand it	
	b) Leave them to do it	
	c) Let them enjoy it	
	d) Get them to think about it	
12.	a) It's yours	
	b) It's ours	
	c) It's mine	
	d) It's theirs	

Scoring

Each word or phrase in each of the twelve sets on the TTI corresponds to one of four training styles, which will be described on the TTI Interpretation Sheet. To compute your scale scores for each type, transfer your numerical ranking for each item on the inventory to the appropriate space in the columns below. Then add up the numbers in each column and enter the totals in the spaces below the columns. The totals are your scores for the four training types.

TOTALS:	L	D	I	C
	1a:	1b:	1c:	1d:
	2b:	2a:	2b:	2c:
	3c:	3d:	3a:	3b:
	4b:	4c:	4d:	4a:
	5a:	5b:	5c:	5d:
	6d:	6a:	6b:	6c:
	7c:	7d:	7a:	7b:
	8b:	8c:	8d:	8a:
	9a:	9b:	9c:	9d:
	10d:	10a:	10b:	10c:
	11c:	11d:	11a:	11b:
	12b:	12c:	12d:	12a:

Interpreting Trainer Type Inventory

Your lowest score is your least preferred training type, and offers you the greatest opportunity for growth and development. Your highest score is your most preferred type. On possible implication

here, if this score is too high, is that you may be using your preferred style to excess. You may need to develop your skill in the other training styles in order to present information in ways that make sense to a greater range of participants.

The Trainer Type Inventory describes four training approaches: Listener, Director, Interpreter, or Coach. Each of the four training styles identified by the TTI is characterized by a certain training approach, way of presenting content, and relationship between the trainer and the trainees. The following are the primary characteristics of the trainer for each of the four training types.

Listener (L)	Director (D)
<ul style="list-style-type: none"> •Creates an effective learning environment •Trains the Concrete Experienter most effectively •Encourages learners to express personal needs freely •Assures that everyone is heard •Shows awareness of individual group members •Reads nonverbal behaviour •Prefers that trainees talk more than the trainer •Wants learners to be self-directed and autonomous •Exposes own emotions and experiences •Shows empathy •Feels comfortable with all types of expression (words, gestures, hugs, music, art etc.) •Does not seem to worry about the training •Stays in the here-and-now •Is practical (goes with the flow) •Appears relaxed and unhurried 	<ul style="list-style-type: none"> •Creates a perceptual learning environment •Trains the Reflective Observer most effectively •Takes charge •Gives directions •Prepares notes and outlines •Appears self-confident •Is well organized •Evaluates with objective criteria •Is the final judge of what is learned •Uses lectures •Is conscientious (sticks to the announced agenda) •Concentrates on a single item at a time •Tells participants what to do •Is conscious of time •Develops contingency plans •Provides examples •Limits and controls participation
Interpreter (I)	Coach (C)
<ul style="list-style-type: none"> •Creates a symbolic leaning environment •Trains the Abstract Conceptualiser most effectively •Encourages learners to memorize and master terms and rules •Makes connections (ties past to the present, is concerned with the flow of the training design) •Integrates theories and events •Shares ideas but not feelings •Acknowledges others' interpretations as well as own •Uses theory as a foundation 	<ul style="list-style-type: none"> •Creates a behavioural learning environment •Trains the Active Experimenter most effectively •Allows learners to evaluate their own progress •Involves trainees in activities, discussions •Encourages experimentation with practical applications •Puts trainees in touch with one another •Draws on the strengths of the group •Uses trainees as resources •Helps trainees to verbalize what they

<ul style="list-style-type: none"> •Encourages generalizations •Presents well-constructed interpretations •Listens for thoughts; often overlooks emotions •Wants trainees to have a thorough understanding of facts, terminology •Uses case studies, lectures, readings •Encourages learners to think independently •Provides information based on objective data 	<p>already know</p> <ul style="list-style-type: none"> •Acts as facilitator to make the experience more comfortable and meaningful •Is clearly in charge •Uses activities, projects and problems based on real life •Encourages active participation
--	--

Each type also trains in a different way.

- The Listener trains the Concrete Experiencer most effectively, and is very comfortable in the activity and publishing steps of the Experiential Learning Cycle.
- The Director obtains the best results from the Reflective Observer, and is usually very comfortable during Step 3, which is processing (particularly in helping trainees make the transition from “How do I feel about this?” to “Now what?”).
- The Interpreter trains in the style favoured by the Abstract Conceptualiser (Step 4, generalizing).
- The Coach trains in the style favoured by the Active Experimenter (Step 5, applying)
- These relationships are indicated in the table below.

	L Listener	D Director	I Interpreter	C Coach
Learning Environment	Affective	Perceptual	Symbolic	Behavioural
Dominant Learning Style	Concrete Experiencer	Reflective Observer	Abstract conceptualiser	Active Experimenter
Means of evaluation	Immediate personal feedback	Discipline based; External criteria	Objective criteria	Learner's own judgment
Means of Learning	Free expression of personal needs	New ways of seeing things	Memorization; knowing terms and rules	Discussion with peers
Instructional Techniques	Real-life applications	Lectures	Case studies, theory, reading	Activities, homework, problems
Contact with Learners	Self-directed Autonomous	Little participation	Opportunity to think alone	Active participation
Focus	"Here and now"	"How and why"	"There/then"	"What/How"
Transfer of Learning	People	Images	Symbols	Actions
Sensory	Touching	Seeing and	Perceiving	Motor Skills

Perception		hearing		
------------	--	---------	--	--

III.5.2 Presenting Information

Coaching

In most technical training situations, the objective is to train the trainees to use a particular software program or computer system. This typically involves hands-on practise. The trainer can be more effective in these situations if he/she acts like a “coach” rather than a “trainer” in the traditional sense.

Following are some tips:

1. Think like a coach

Be committed to everyone’s success; don’t think about “bell curves”. You are a successful trainer only if the trainees have a successful experience.

2. Prepare the trainees

Let them know the “rules of the game”. Tell them what they will be doing and point out the “pitfalls” – ahead of time.

3. Focus on the basics

Reinforce basic skills such as reading the screen, knowing the keyboard and using the mouse, understanding the general concepts and context

4. Don’t give away answers

Make trainees think. Forward the action by asking “show me what you did”. Try taking different approaches. Let them experience the solution.

5. Don’t press the keys!

Never press the keys. This is one of the biggest sins a trainer can commit. Let them do the driving, except when it distracts from the training

6. Reinforce strengths and build on success

Point out to the things that they already know and what is being done correctly, than encourage and help them move the ball down the field.

Interactive Lecturing

Interactive lecturing is the use of questioning, discussion and lecturing to stimulate understanding, direct discussions, and provide information. Its purpose is to change the roles of both trainer and the trainee from passive to active.

Following are some tips:

1. Have a plan

Have a plan or information flow in mind is, of course the most important first step. This serves as a “road map” to help keep the trainer focused. This plan also has the information divided into manageable units.

2. Use your eyes

Look at the trainees. Use eye contact to create involvement. Check understanding and “control” the room

3. Use your voice

Speak clearly and strongly. Don't use "filler" language. It indicates that you are not sure what you are talking about. Use inflection in the voice to keep interest and emphasise key pieces of information. Remember to breathe.

4. Use your presence

Move around the room. Use the back as well as the sides and front. Use your presence to promote involvement and discouraged distractions

5. Use questioning

Questions are the most important tool. They are used to stimulate the trainees thinking and involvement in the content, moving them from passive to an active role.

6. Build on what they already know

Use analogies, metaphors, stories, graphics and "real world" examples to illustrate both verbally and visually the information you are providing.

7. Set a context

Make sure that you present the "big picture" and point out where the trainees are focused at the moment. Also be sure to let them know what is coming.

8. Stay conscious

Read the trainees' body language. Make eye contact, breathe; move around. Use variety and humour. Keep to the timing. Don't get off track. Keep in mind whose needs are being met.

III.5.3 Using Your Body Effectively

Effective communication involves more than talking to your audience. Your body language plays an important role in communication. Research shows that what you say accounts for only 7% of the effectiveness of a presentation, while 93% are based on non-verbal communication. Body language, proximity, and eye contact are three main areas of focus in non-verbal communication. Remember it's not what you say, but how you say it that often matters the most in communication.

Some areas to consider while presenting include:

1. Facial expressions

Smiling is a powerful cue that transmits friendliness, warmth, and approachability. Smiling is often contagious and others will react favourably. They will be more comfortable around you and more open to the information you are offering.

2. Posture

You communicate numerous messages by the way you hold yourself while presenting. A person who is slouching or leaning with arms across their chest may be perceived as being uninterested or unapproachable. Standing erect, facing the audience with an open stance, and leaning forward communicates that you are receptive and friendly. Speaking with your back turned or looking at the floor or ceiling should be avoided as it communicates disinterest.

3. Gestures

[ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification] Created during the initiative "ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software in Africa", a programme of FOSSFA and GIZ. For more information see www.ict-innovation.fossfa.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.

A lively speaking style captures attention, makes the material more interesting, and facilitates understanding. Use natural movements to emphasise topics and free, easy arm and hand movements to add personality to your presentation. If you fail to gesture while speaking, you may be perceived as boring and stiff. Gesturing too often can also be distracting for some learners.

4. Movement

Moving naturally around the classroom increases interaction, adds interest, and draws attention to the presentation. Staying frozen in the front of the room can be distracting and boring for people to watch. Shuffling feet and pacing can convey nervousness and lack of confidence.

5. Proximity

Cultural norms dictate a comfortable distance for interaction with others. When interacting with adults in the classroom, a presenter needs to be aware of people's defined levels of personal space. Signals of discomfort caused by invading other's space may include rocking, leg swinging, tapping, and gaze aversion. Do not invade a student's intimate space. Most adults will feel uncomfortable, even if rapport has been established.

III.5.4 Building Rapport with Eye Contact

Steady eye contact helps to regulate the flow of communication, encourages participation, and can be used to develop rapport with the audience. When students feel that you see them as individuals, they are more likely to trust you as a trainer and be more open to the learning experience.

Some tips for using eye contact to build rapport include:

1. Length of eye contact

Try to maintain eye contact with one person at a time for at least 3 – 5 seconds or until you complete a thought. This helps to establish a connection with people and helps you to avoid darting eyes, which can be distracting and communicate nervousness.

2. Movement of eyes

Try to establish direct eye contact towards different parts of the audience throughout the course of your presentation. Staring too long in one direction may cause you to miss important information and can make certain audience members feel less important.

3. Search for friendly eyes

If you are nervous, look for a friendly trainee and establish eye contact with that trainee. Gradually, work to establish eye contact with everyone.

Some habits to avoid include:

1. Talking to the ceiling

Don't lecture to a spot over the top of the trainee's heads. They may think you don't care or they may feel that you are "above them." Adults learn better with colleagues.

2. Talking to the board

Don't talk to your desk, to the whiteboard, or to your visuals. Trainees may not be able to

hear you and may become disinterested.

3. Clutching your training manual

Be familiar with your training material. Being tied to your notes or a manual keeps you from establishing eye contact and may cause trainees to question your knowledge, preparedness, and confidence.

III.5.5 Enhancing Voice Quality

Voice is another area of communication that can affect the quality of learning in a classroom. An interesting and audible voice will engage trainees, while a soft or monotone voice can cause boredom or disinterest among trainees. While it may be difficult to listen to and change our own voice, with awareness and practice, it is possible to use one's voice effectively. The first step to refining your voice is to understand the components of voice and identify common voice problems. Once identified, most voice problems can be improved by being aware of the problem, altering some habits, and practicing new behaviours on a regular basis.

1. Pace

How long a sound lasts. Talking too fast causes words and syllables to be short, while talking slowly may lengthens them. Varying pace helps to maintain the audience's interest.

Suggestions for improvement:

- Be aware of your normal conversational pace and keep in mind how tension affects the speed in which you talk.
- Use breathing and natural pauses to slow down your pace
- Constantly vary your pace in order to maintain audience interest.

2. Projection

The direction of the voice so that it can be plainly heard at a distance is considered effective. Problems with projection are often the result of tension and breathing from your throat.

Suggestions for improvement:

- Avoid projecting from your throat which can lead to sore throats, coughing, and loss of your voice.
- Take slow, deep breaths, initiated from your abdomen
- Open your mouth fully and speak to the people in the back of the room.

3. Articulation

The ability to pronounce words distinctly. It often reflects our attitude towards the words we are speaking. Clear enunciation reflects self-confidence and interest, while slurred or mumbled speech indicates insecurity or indifference.

Suggestions for improvement:

- Speak at a slower pace than your normal conversational tone.
- Take the time to pronounce each letter or sound within a word.
- Listen for common articulation problems, such as dropping the "g" at the end of words

such as finding or going.

4. Pitch

Pitch describes the normal range of the voice – its highness or lowness. Everyone is capable of a wide voice range. Stress and poor breathing can greatly alter the pitch of your voice.

Suggestions for improvements:

- Adjust pitch to convey different meanings throughout a presentation.
- To alter pitch, control your breathing; breathe from your abdomen and slow your rate of speech.
- Take pauses to relax between pitch changes

5. Inflection

Inflection refers to the manner in which pitch varies as we speak. Inflection serves as verbal punctuation and involves changing pitch to convey meaning. Upward inflections ask a question, suggest uncertainty or doubt, and communicate hesitancy. Downward inflections give information and convey strength and authority to the audience.

Suggestions for improvement:

- Use upward and downward inflections appropriately.
- Avoid constant middle inflection where the voice neither rises nor falls but just drones on and on.

Module III.5.6 Questioning Techniques

Questioning is the power tool to use in training. It has many uses, from testing trainees on their knowledge of the subject matter, to get information to helping a trainer maintaining classroom control. Trainers often state concept when the class could be actively involve if more questioning were used.

Types of Questions

1. Whole group

This type of question is directed to the entire group.

2. Individual

This type of question is directed to a trainee. You should use this questioning method carefully. You can start by asking a whole-group question. Then, and only after evaluating the group and identifying a trainee who will clearly be able to answer, redirect the question to particular trainee.

3. Pass

This technique is used to direct a question asked by a trainee, to the group. It can also be used get a trainee “off the hook” if he or she is unable to answer an individual question.

4. Reword and ask again

This technique can be used when you have a poorly worded question and you need to restate for better understanding, or when you’ve receive an answer that is “close” but not quite correct.

5. Rhetorical

[ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification] Created during the initiative “ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software in Africa”, a programme of FOSSFA and GIZ. For more information see www.ict-innovation.fossfa.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.

A rhetorical question is usually asked solely for thought-provoking purposes. An answer is not expected.

6. Testing Questions

A testing question is asked by a trainer to test knowledge, something that a trainee already knows or can be reasonably expected to know. It is also used to:

Handling Responses to Question

How you handle responses from trainees can be equated with doctors "bedside manner." The ways in which question are addressed can either encourage interaction or end it for the rest of the day.

Giving trainees nine second to respond may seem long and at first quite uncomfortable. However, it takes the average adult about three second to process the questions; another three second to see if someone else will answer the question for them, and an additional three seconds to find the courage to respond.

- Accept at any time, but take one at a time
- Deal with those which are relevant now and others later
- Indicate degree of correctness
- Build on trainee's words
- If you do not know the answer - let them know; note questions and find the answer
- Do not bluff; you will be caught out
- Tell the group as well as the individual

Level of Questioning

1. Low level

Low-level questions are the most commonly used questions in a classroom (50%-90%). These questions are highly convergent, and they typical check for concrete knowledge learned. These questions often start with words like, what, when, where, and who. They work well early in the day because they are safe questions with clear right and wrong answers.

2. High level

High level questions involve some type of personal value judgement on the part of person answers. High-level questions tend to promote divergent thought. They typically start with words like who and why. Typical, a low-level understanding of situation is needed to give an answer to the high-level questions.

Questions

1. List and briefly describe the four learning styles.
2. List and briefly describe the four different training types.
3. Discuss coaching and interactive lecturing as techniques to present information
4. What are the areas you should consider when presenting information
5. List some tips for using eye contact to build rapport.
6. List and briefly describe the components of voice.

7. Why is questioning important?
8. How should one handle responses to questions?

Exercise

Design a programme for (a) business people; (b) trainers; and (c) graduates in a group. The outcomes should be presented by a group.

Assignments and Answers for Chapter III FOSS TRAINING

TASK

It is required that you design a national roll **plan**. The content of the plan should include:

- strategies and ideas on how to achieve this plan
- how you intend to get your participants
- qualifying participants assessment criteria
- duration of the program
- cost of training or participation fee
- number of trainers needed and their cost
- number of modules to be included in your training
- any sponsorship opportunities
- how you intend to sustain further trainings

Discuss the Case study based on the core questions discussed under FLOSS training business opportunities.

Rules: Please state (**3 maximum**) points briefly how you would have approached the same training opportunities and any two training opportunities you would like to explore as a group of trainers.

Some examples from participants

1 Objectives and Vision Statement

The basic objectives of the national roll-out training are:

- To raise FOSS awareness in the Tanzanian Community
- To develop sustainable FOSS-based IT business
- To strengthen FOSS communities in Tanzania

The vision is to build capacities in African small and medium IT enterprises to make FOSS based business. It aims to encourage the growth of African IT industries.

2 Participants

As the objective of the training is focusing on FOSS business models, the training will call for participants from IT companies wishing to diversify their business, start-up IT entrepreneurs, local FOSS communities, educators and graduates in IT related subjects as well as other participants interested in FOSS business models in Tanzania.

A call for participants will therefore be advertised through different marketing strategies. Different media through which this course can get very good publicity in Tanzania, are: the newspapers, e-mail to target groups and other prospective IT companies. Other strategies will be development of promotional materials such as posters and place them in high traffic areas.

The Training will be delivered in a Workshop style to allow maximum interaction and discussions. The target is to train 20 participants each quarter (three months) and therefore about 80 participants per year. Initial trainings will be concentrated in regional towns and cities where IT infrastructure is well established

It is expected that all participants will be selected from those who have at least knowledge of basic computers, experience in FOSS or potential FOSS collaborator to allow uniform workshop interactions.

3. Output

The target is to achieve the set objectives; so we can expect the Training outcome to be:

- Awareness on FOSS causes more people to demand for FOSS applications and solutions and as such provide opportunities for any FOSS businesses to flourish
- Growth of a FOSS Community, which comes with it, more FOSS minds, students, trainers, and advocacy and so on.
- With these and more benefits, it is possible to plan an annual FOSS event, such as FOSS Clinics, Free FOSS Training, FOSS certification and so on.

4 The Training Content

The training shall consider the local conditions and environment. It will comprise three modules, namely Module 1, Module 2 and any one of the other modules as listed below:

- Module 1: Introduction to emerging FOSS business models
- Module 2: African business models: Case studies
- Module 3: Communicating FOSS
- Module 4: Introduction to General business skills
- Module 5: FOSS specific business knowledge and skills
- Module 6: FOSS Training
- Module 7: E-learning Platform

The selection of the modules will depend on the mission of the target groups and their levels of understanding. The content materials will be availed to the participants at least one week before the workshop.

5 Resource Persons (Trainers)

As a FOSS ToT member I will be the principal trainer. Supporting trainers will be sourced from within FOSS related institutions and FOSS - ToT Alumni. Shortfall of trainers will be requested from InWent and FOSSFA. Qualified trainers will need to have knowledge of FOSS, proven training Experience and proven excellence in their field of profession

6 Time Schedules and Venues

A workshop session will be done for a period of 5 days (not less than 40 hours). This time is just long enough for participants to understand the materials content and practices of the FOSS modules without being bored.

Information on the workshop venue, schedules for starting and end times and dates for the workshop sessions will be set and communicated to participants at least four weeks in advance. Arrangements for hotel accommodation will be arranged for participants in need. The workshop organization will establish a support through help desk for post training questions, accommodations reservations, training registrations etc.

7 Fees and Costs

Participants shall contribute an TSH. xxx for the workshop. Trainers would be paid TSH xxx per day.

8 Sponsorship & Marketing

InWent and FOSSFA have promised funding and support, It is possible to look for Local Support for this Training workshop, especially from Companies and institutions that will likely benefit from adding their name and profile to such an event. So a workshop prospectus together with a call for support and sponsorship would go a long way in acquiring some support. This needs to be done in time.

9 Media

Media Coverage is crucial for such a workshop and this would help to inform a larger group of people about the Workshop. A Newspaper article, a phone in Radio announcement, a community announcement could all add to the buzz.

10 Course ware evaluation

Evaluation forms for the course ware and trainers will be prepared and availed to the participants after every module. At closing of the workshop there will be general discussion on the evaluation results and process to ways for improvement. This will also give opportunity to discuss issues that are of interest or were omitted for future improvements.

11 Budget

#	Activity	No	Rate/day	Tsh/day	TOTAL / 5 days
1	INCOME				
1.1	Participation fee	20			
1.2	Total Income				
2	EXPENDITURE				
2.1	Venue with computers	1			
2.2	Refreshments	23			
2.3	Lunch	23			
2.4	Transport for ppts	20			
2.5	Stationary	20			
2.6	Trainers fees	3			
2.7	Advertising and marketing	1			
2.7	Total Expenditure				
	Contingencies / accidentals 10%				
2.8	TOTAL				

Aflent's national roll out plan will focus on providing training initially to university students, BEE entrepreneurs and other established black businesses in South Africa. The aim is to empower these groups with FLOSS for two major reasons:

- Impart skills to use FOSS applications to enable and capacitate internal business processes in order to gain competitive advantage
- To train the groups on how to move into FOSS-based businesses (FOSS consultancy and training)

Strategies

The key to achieving successful trainings lie in the ability to attract attention of participants, offering a highly accredited and internationally recognised African FOSS Business Model course establish and maintain continuous online learning and networking platform for trainees. The trainings will take the form of a strategy-driven process.

Conduct a preliminary FLOSS workshop to market the concept and the training.

Pitch for short course raining deals with local institutions and universities.

Clear the issue of certification and accreditation of the course.

Develop a solid sustainability model.

Getting participants

Participants will be sourced from the corporate industry, universities, NGOs and the public sector including technologists in government departments.

- Marketing the FOSS concept
- Marketing the FOSS training
- Networking with colleagues, former workmates and clients interested in Open

Participants' assessment criteria

1. Attitude
2. Eagerness to learn
3. Reasons why participants want to get the training
4. Skills
5. A qualification in IT and/ or business
6. Training skills
7. Previous advocacy work
8. Knowledge
9. Basic understanding of FOSS and/or IT's role in business

Duration of the program

2 weeks for 2 months

Cost of training or participation fee

Trainers needed and their cost

As a certified trainer I will choose a number of modules 2 and 4. The remainder of the modules will be delivered by local and regional trainers subject to their availability. In total four trainers are required for the course. Cost of the trainers will be determined by FOSSFA and InWent.

Estimated rate per trainer per module will be obtained from the FOSSFA/InWent.

Modules to be included in your training

Provisionally all modules are to be included:

1. Introduction to emerging FLOSS Business Models
2. African Business Models Case Studies
3. Communicating FLOSS
4. Business Skills
5. FOSS Business Knowledge and Skills
6. FOSS Training Business

Sponsorship

1. Established FOSS companies (Sun Microsystems)
2. First National Bank
3. IT companies
4. Government departments (DST, SITA, Local Councils)

Sustainability Model

The training will be offered as a short course through The Business Place and Monash and North West University.

TEST Chapter III: FOSS TRAINING

1. Below are some of the winning combinations of an "ideal" FOSS trainer. Which one is not? Ability to engage participants and relate content to their situation.

- (a)Winning personality and ability to communicate.
- (b)Practical experience in FLOSS applications.
- (c)Ability to program in Java and Visual Basics

2. Which one of the following is not one of the three types of training interventions for FOSS Trainers discussed in module 6.1?

- (a)Those who have the technical knowledge of FLOSS packages and have been using certain packages, but have not conducted any training of any nature.
- (b)Those who have the knowledge and skills on how to conduct ICT training for adults which they might have acquired through self-learning or having had attended a course, but have not conducted any FLOSS training. Instead, they have conducted training in proprietary software.
- (c)Those who have no jobs and they want to occupy themselves
- (d)Those who have conducted FLOSS training courses and have the knowledge and skill on how to conduct training, but have not trained other trainers

3. Why is it important that trainers use the software they will be training?

- (a)Helps them become conversant with features and functionalities that the software provides and can transmit this knowledge to their trainees.
- (b) Helps them boast of the knowledge they have about the software
- (c) Helps them deliver the training within the shortest time possible
- (d) Helps them be liked by the trainees

4. Trainers should see training as a stage production: which of the following is not part of the stage production aspects?

- (a)There is the audience (participants).
- (b)There is a stage (training room).
- (c)The performance by the actor (trainer conducting the course).
- (d)The performance by the audience (Shouting)

5. Selection of candidates for FOSS training can be done using the following criteria except.....

Please contribute to the material! Follow this link to contribute: <http://www.ict->

innovation.fossfa.net/node/4255

6. Below are some of FOSS certification recognisable in many countries. Which one is not very recognizable when it comes to FOSS training?

Please contribute to the material! Follow this link to contribute: <http://www.ict-innovation.fossfa.net/node/4255>

7. Below is a list of some of the steps involved in the design of a course/curriculum. Which one is weak compared to the others in the list?

Please contribute to the material! Follow this link to contribute: <http://www.ict-innovation.fossfa.net/node/4255>

8. Why should the development of FLOSS training content take cognisance of the commercial world for proprietary software?

9. The Diverger is described by the following except:

Please contribute to the material! Follow this link to contribute: <http://www.ict-innovation.fossfa.net/node/4255>

10. The Converger is described by the following except:

Please contribute to the material! Follow this link to contribute: <http://www.ict-innovation.fossfa.net/node/4255>

APPENDIX

Chapter III: FOSS TRAINING: Authors and Trainers

Pool of African ict@innovation Trainers and Experts

	Name Module	Name
Module 6	FOSS Training	Arnold Pietersen, Celso Timana, Paschalia Ouma, Shirley Akasreku, Frederick Yeboah, Kofi Kwarko More Trainers per country in full Pool of Trainers here: http://www.ict-innovation.fossfa.net/wiki/public-wiki/course-advanced-african-foss-business-models/FBMTrainers

Main contributors

Module 6	Sulayman K. Sowe (Facilitator), Arnold Pietersen, Glenn McKnight, Paschalia Ndungwa Ouma, Derek Lakudzala
----------	---

Partners

Implementing Partners

FOSSFA – Free Software and Open Source Foundation for Africa (FOSSFA)

FOSSFA is the premier pan African organization promoting the use of free and open source source and the open source development model for African development.

FOSSFA partners with GIZ to implement the ict@innovation programme. FOSSFA is the premier African FOSS organization. The vision of FOSSFA is to promote the use of FOSS and the FOSS model in African development, and the organization supports the integration of FOSS in national policies. FOSSFA also coordinates, promotes, and adds value to African FOSS initiatives, creativity, industry, expertise, efforts and activities at all levels. For more information visit <http://www.fossfa.net>

GIZ – Deutsche Gesellschaft für Internationale Zusammenarbeit GmbH Broad-based expertise for sustainable development

The services delivered by the Deutsche Gesellschaft für Internationale Zusammenarbeit (GIZ) GmbH draw on a wealth of regional and technical expertise and tried and tested management know-how. As a federal enterprise, we support the German Government in achieving its objectives in the field of international cooperation for sustainable development. We are also engaged in international education work around the globe.

Tailored services

We offer demand-driven, tailor-made and effective services for sustainable development. To ensure the participation of all stakeholders, we apply a holistic approach based on the values and principles upheld in German society. This is how we facilitate change and empower people to take ownership of their own sustainable development processes. In doing this, we are always guided by the concept of sustainable development, and take account of political, economic, social and ecological factors. We support our partners at local, regional, national and international level in designing strategies and meeting their policy goals.

Developing solutions

GIZ operates in many fields: economic development and employment promotion; governance and democracy; security, reconstruction, peacebuilding and civil conflict transformation; food security, health and basic education; and environmental protection, resource conservation and climate change mitigation. We also support our partners with management and logistical services, and act as an intermediary, balancing diverse interests in sensitive contexts. In crises, we carry out refugee and emergency aid programmes. As part of our services, we also second development advisors to partner countries.

Through programmes for integrated and returning experts, we place managers and specialist personnel in key positions in partner countries. We also promote networking and dialogue among actors in international cooperation. Capacity development for partner-country experts is a major component of our services, and we offer our programme participants diverse opportunities to benefit from the contacts they have made. We also give young people a chance to gain professional experience around the world – exchange programmes for young professionals lay the foundations for successful careers in national and international markets.

Who we work for

Most of our work is commissioned by the German Federal Ministry for Economic Cooperation and Development. GIZ also operates on behalf of other German ministries – including the Federal Foreign Office, the Federal Ministry for the Environment, Nature Conservation and Nuclear Safety, the Federal Ministry of Defence, the Federal Ministry of Economics and Technology and the Federal Ministry of Education and Research – as well as German states and municipalities, and public and private sector clients in Germany and abroad. These include the governments of other countries, the European Commission, the United Nations and the World Bank. We work closely with the private sector and promote synergies between the development and foreign trade sectors. Our considerable experience with networks in partner countries and in Germany is a key factor for successful international cooperation, not only in the

[ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-Enterprises to get Open Source Skills and Certification on Level 1 of the Linux Professional Institute (LPI) Certification] Created during the initiative "ict@innovation – Creating Business and Learning Opportunities with Free and Open Source Software in Africa", a programme of FOSSFA and GIZ. For more information see www.ict-innovation.fossfa.net. Provided under a Creative Commons Attribution-Share Alike 3.0 Germany License. Copyright: FOSSFA & GIZ.



business, research and cultural spheres, but also in civil society.

Global reach – the company at a glance

GIZ operates throughout Germany and in more than 130 countries worldwide. Our registered offices are in Bonn and Eschborn. We have more than 17,000 staff across the globe, some 70 per cent of whom are employed locally as national personnel. There are also around 1,000 development advisors working for GIZ. In addition, CIM – which is jointly run by GIZ and the German Federal Employment Agency – places experts with local employers. At the end of 2011, almost 600 integrated experts had employment contracts with organisations and companies in the field, while 450 returning experts were receiving financial support and advice. Last year just under 500 young people were assigned abroad through the 'weltwärts with GIZ' programme. At year-end GIZ's business volume stood at around EUR 2 billion. (Figures as at 31 December 2011)

For more information visit <http://www.giz.de>

Funding / Strategic Partners

BMZ – German Federal Ministry for Economic Cooperation and Development

ict@innovation is implemented by GIZ on behalf of and funded by the German Federal Ministry for Economic Cooperation and Development (BMZ). The BMZ represents GIZ's shareholder, the Federal Republic of Germany. The BMZ develops the guidelines and the fundamental concepts on which German development policy is based. It devises long-term strategies for cooperation with the various players concerned and defines the rules for implementing that cooperation. These are the foundations for developing shared projects with partner countries and international development organisations. All efforts are informed by the United Nations' Millennium Development Goals, which ambitiously aim to halve extreme poverty in the world by 2015. For more information visit <http://www.bmz.de/en/>

OSISA – Open Society Initiative for Southern Africa

OSISA supports ict@innovation on a strategic level. The Open Society Initiative for Southern Africa (OSISA) is a leading Johannesburg-based foundation established in 1997, working in ten Southern Africa countries: Angola, Botswana, DRC, Lesotho, Malawi, Mozambique, Namibia, Swaziland, Zambia and Zimbabwe. As a foundation, OSISA provides African leadership in the definition and development, within the specificities of Southern African realities, of the concept and ideals of an open society. For more information visit <http://www.osisa.org>

Creating Business and Learning Opportunities with Free and Open Source Software

What is Free and Open Source Software?

Free and Open Source Software (FOSS) is software which can be freely used, modified and distributed. FOSS offers a number of different opportunities. Developers are able to customize, change or add to open source software and join in global open production processes. This can help stimulate local innovation and growth in the IT sector.

With FOSS, small and medium-sized IT businesses can create locally adapted IT solutions, independent of foreign software vendors. FOSS allows local value chains to be tapped, instead of forcing customers to rely on foreign software vendors.

Free and Open Source Software technologies are used all over the world. FOSS is often the technology of choice to run servers, networks, or content management systems, but also operating systems such as Linux, or business and office applications such as OpenOffice. As FOSS is adaptable and does not entail license fees, it is particularly useful when applications need to be adapted to a specific context.

The use of FOSS is spreading – governments and businesses are increasingly employing FOSS. This means that the business market around FOSS solutions is growing. Local businesses, in particular IT-SME can benefit from FOSS as users, but more importantly they can generate business models around FOSS such as offering high-value IT services, software development, training and qualification.

Free and Open Source Software creates business opportunities!

FOSS technologies offer opportunities particularly for small and medium sized IT enterprises to provide IT services for local IT markets.

What do small and medium-sized enterprises (SME) in Southern and East Africa need to work with FOSS? Many IT-SME are not yet aware of how they can use FOSS in their business models – knowledge sharing and training is needed to qualify employees. In addition, trust in FOSS needs to be improved, for instance by spreading quality standards. ict@innovation addresses these topics by:

- Sharing knowledge on African Business Models and Skills in FOSS
- Building trust and business through FOSS
- Certification
- Developing innovative local FOSS Applications

ict@innovation is an international capacity building programme, implemented in partnership by **FOSSFA** (Free Software and Open Source Foundation for Africa) and **GIZ** (Gesellschaft für Internationale Zusammenarbeit GmbH).

The **ict@innovation** programme offers **training courses** for training institutions and trainers on

- business models and business development for IT SME – how to integrate FOSS services in your training portfolio
- how to get certified in basic FOSS technical skills

Main objective of ict@innovation is to foster small and medium-sized enterprises (SME) in the field of Free and Open Source Software in Southern and East Africa. Through advanced training and networking in FOSS skills, the programme contributes to qualify African IT SME in providing localized and adapted FOSS applications and services to public administration and private sector.

The programme focuses on Free and Open Source Software (FOSS) as a key technology to drive innovation, add local value and create sustainable and affordable ICT-solutions.

Region of Implementation	East and Southern Africa (particularly Ethiopia, Kenya, Malawi, Mozambique, Namibia, Rwanda, South Africa, Tanzania, Uganda, and Zambia) West Africa (particularly Cameroon, Benin, Burkina Faso, Ivory Coast, Niger, Mali, Niger, Nigeria, Senegal and Togo)
Duration	2008 – 2012 East and Southern Africa 2011 – 2012 West Africa
Funding/ Strategic Partners	German Federal Ministry for Economic Cooperation and Development (BMZ) & Open Society Initiative for Southern Africa (OSISA)
Website	www.ict-innovation.fossfa.net

Site Features:

Profiles – Create your own profile, advertise your FOSS skills and share your interests in FOSS and the ict@innovation programme

Forums and Wiki – Read and contribute to discussions and work processes by engaging in online discussions in the forum and collaborative text editing using the wikis

Blogs – read and discuss about current news and developments on FOSS related topics and the ict@innovation programme in the blogs

Groups – Join in online organizational activities by engaging in topic specific groups

Online Community Membership

Do you have a FOSS project or initiative you wish to interest others in? Do you want to contribute to the ict@innovation programme in a specific way?

The ict@innovation portal offers functions for those of you, who want to engage more actively in the community or programme – The blog is open for all community members to share relevant news and views. Also, all community members can create groups to share files, manage events, and engage in private discussions on relevant topics. The different training programmes will later on all have their own group to collaborate online. All services of this site are free of charge.

www.ict-innovation.fossfa.net the community webportal

The web portal is designed to keep its members in touch with current developments of the ict@innovation programme and other FOSS projects in Africa. It is also designed to host a community of persons interested in or working in the area of FOSS in Africa and to enable participatory management of the ict@innovation programme. We invite you to become a community member!

Joining the ict@innovation Web Community

The Benefits: Why join? This website offers you the opportunity to get to know a range of persons working in related areas, as well as to advertise your skills and interests, to share and learn from other experts using the range of tools that enable active knowledge exchange for community members.

Becoming a Member: You can become a member of this community by registering on the site and creating a profile, indicating your interests and skills in FOSS. This will enable you to contribute to the website fora and wikis, to join in knowledge exchange with other persons interested in FOSS and/ or involved in the ict@innovation programme.

Staying Informed: Receive all blog posts and other website updates by email or RSS feed! In order for everyone to stay in touch with the programme and community activities, we offer an email-subscription and RSS feed service on all website services and news items (including blog posts, events, forum messages etc.). You find RSS icons to subscribe to feeds at the bottom of the sections that offer subscriptions. You can manage email subscriptions as a registered user under "My account" > "Subscriptions".

Sign up on ict@innovation to join the community and stay updated: <http://www.ict-innovation.fossfa.net/>

About this Training Guide on Linux System Administration, LPI Certification Level 1

The "ict@innovation: Training Guide on Linux System Administration – LPI Certification Level 1. Supporting African IT-enterprises to get Open Source skills and certification on level 1 of the Linux Professional Institute (LPI) Certification" supports the building of knowledge and capacities in African small and medium ICT enterprises. The guide is part of "Linux Certification in Africa - The ict@innovation Training of Trainers Programme". It aims to help African IT-enterprises involved in Free and Open Source Software (FOSS) to improve the quality of their services through certification of their FOSS skills. The open training material is part of the initiative ict@innovation, a partnership of FOSSFA (Free Software and Open Source Foundation for Africa) and GIZ (Deutsche Gesellschaft für Internationale Zusammenarbeit GmbH).

This Training Guide was mainly written for African experts and institutions wanting to become certified trainers on the subject of Linux system administration. It is also geared towards learners, who want to go through the first essential steps in becoming a Linux System Administrator charged with installing, supporting, and maintaining Linux-based computer systems. It builds on the Linux Professional Institute (LPI) Certification as a world-wide recognized distribution and vendor-neutral standard for evaluating the competency of Linux professionals with the possibility to hold low-cost paper-based examinations. The LPIC-1 (101 and 102 exams) level taught in this guide cover the following fundamental Linux system administration skills:

- **LPI 101 Module:** Install Linux, making appropriate choices for disk partitioning / Boot the system, change run levels, shut-down and reboot / Work effectively at the shell command prompt / Install and manage packages using both RedHat and Debian tools / Manage, find, copy, delete, archive and compress files and directories / Process text streams using pipes, filters and redirection / Manage processes and modify process execution priorities / Search text files with regular expressions and edit files with vi / Create partitions and filesystems, and maintain their integrity / Control file access permissions.
- **LPI 102 Module:** Customise the shell and write simple shell scripts / Query databases and manipulate data using SQL / Install and configure the X server and set up a display manager / Manage user accounts and groups / Schedule jobs at regular intervals using cron / Localise the system for a language other than English / Keep your system clock correct / Manage printers and printing / Understand IP networking and set up a basic network configuration / Maintain host security and enable secure login with ssh.
- **FOSS Trainer Module:** Understand some of the requirements for becoming a FOSS trainer / Be able to identify and seize the opportunities that exist for FOSS training as a business / Gain the knowledge and skills required to organise and provide FOSS training / Appreciate the benefits of peer production of Open Educational Resources and Open Content.

<p>ict@innovation contact FOSSFA</p> <p>Mr Mawusee Komla FOLI-AWLI Community Empowerment Manager (CEM)</p> <p>Email: cem@fossfa.net</p>	<p>Free Software and Open Source Foundation for Africa</p> <p>Secretariat hosted at Advanced Information Technology Institute (AITI) of the The Ghana-India Kofi Annan Centre of Excellence in ICT PMB, State House, Accra Ghana www.fossfa.net</p>
<p>ict@innovation contact GIZ</p> <p>Mr Thorsten SCHERF GIZ Division Economic Development & Employment, ICT Advisor, Sector Project ICT4D</p> <p>Phone: +49 6196 79-1286 Fax: + 49 6196 79-801286 Email: thorsten.scherf@giz.de</p>	<p>Deutsche Gesellschaft für Internationale Zusammenarbeit</p> <p>Friedrich-Ebert-Allee 40 53113 Bonn Germany</p> <p>www.giz.de</p>

www.ict-innovation.fossfa.net