KARPAGAM ACADEMY OF HIGHER EDUCATION

Coimbatore-641 021

(For the candidates admitted from 2017 onwards)

DEPARTMENT OF CS, CA & IT

SUBJECT NAME : OPEN SOURCE TECHNOLOGIES
SEMESTER : III
SUBJECT CODE : 18CSP302 **CLASS: II M.Sc CS**

Instruction Hours / week: L: 4 T: 0 P: 0 Marks: Int : 40 Ext : 60 Total: 100

Program Outcome: A student who successfully completes this course should be able to learn the concepts and principles that underlie modern operating systems and a practice component to relate theoretical principles with operating system implementation.

Program learning Outcomes:

- Understand fundamental operating system abstractions such as Processes, Threads, files Semaphores, IPC, abstractions, shared memory regions etc.
- Understand how the operating system abstractions can be used in the development of application programs or to build higher level abstractions
- Understand how the operating system abstractions can be implemented
- Understand the principles of concurrency and synchronization and apply them to write correct concurrent programs/software

UNIT-I

History and Overview Of GNU/Linux And FOSS 3

Definition of FOSS & GNU, History of GNU/Linux and the Free Software Movement, Advantages of Free Software and GNU/Linux FOSS Usage. Trends and Potential—Global and Indian.

UNIT-II

System Administration

GNU/Linux OS Installation, Detect Hardware, Configure Disk Partitions & File Systems and Install a GNU/Linux Distribution; Basic shell commands -Logging in, Listing Files, Editing Files, Copying/Moving Files, Viewing File Contents, Changing File Modes and Permissions, Process Management; User and Group Management, File Ownerships and Permissions, PAM Authentication; Introduction to Common System Configuration Files & Log Files;

Configuring Networking Basics of TCP/IP Networking and Routing connecting to the Internet (through dialup DSL Ethernet leased line); Configuring Additional Hardware -Sound Cards, Displays & Display Cards, Network Cards, Modems, USB drives, CD Writers;

Understanding the OS Boot up Process; Performing everyday tasks using GNU/Linux -- Accessing the Internet, Playing Music, Editing Documents and Spreadsheets, Sending and Receiving Email, Copy Files from disks and over the Network, Playing games, Writing CDs;

X Window System Configuration and Utilities, Configure X Windows, Detect Display Devices; Installing software from source code as well as using binary packages.

UNIT-III

Server Setup And Configuration

Setting up Email servers, Using Postfix (SMTP services) Courier (IMAP & POP3 services) Squirrel Mail (web mail services) ; Setting up Web Servers --Using Apache (HTTP services) PHP (server-side scripting), Perl (CGI support) ; Setting up File Services -Using Samba (file and authentication services for windows networks), Using NFS (file services for gnu/Linux / Unix networks) ; Setting up Proxy Services, Using Squid (http / ftp / https proxy services) ; Setting up Printer Services -Using CUPS (print spooler), Foomatic (printer database) ; Setting up a Firewall -Using netfilter and iptables.

UNIT-IV

Programming Tools

Using the GNU Compiler Collection, GNU compiler tools, C preprocessor (CPP), C compiler (GCC) and the C++ compiler (g++) assembler (gas) ; Understanding build systems -Constructing make files and using make, using autoconf and autogen to automatically generate make files tailored for different development environments ; Using source code versioning and management tools --Using CVS to manage source code revisions patch & diff ; Understanding the GNU Lib Libraries and Linker –Linking against Object Archives (.a libraries) and Dynamic Shared Object libraries (.so libraries), Generating Statically Linked Binaries and Generating Dynamically Linked Libraries.

Using the GNU Debugging Tools -GDB to Debug Programs, Graphical Debuggers like DDD Memory Debugging / Profiling Libraries mpatrol and valgrind ; Review of Common Programming Practices and Guidelines for GNU/Linux and FOSS ; Introduction to Bash , SED & AWK scripting.

UNIT-V

Application Programming

Basics of the X Windows server architecture; Qt Programming ; Gtk+ Programming ; Python Programming ; Programming GUI applications with localisation support.

TEXT BOOK

1.Venkateshwarlu, N. B. (2012) Introduction to Linux: Installation and Programming. New Delhi: BPS Publishers.

REFERENCES

1. William E. Shotts, Jr. (2012). The Linux Command Line (4th ed.). O'Reilly Publishers.

2. Christopher Negus .(2015). Linux Bible (9th ed.). Wiley Publications.

Web Sites:

1. http://dsl.org/cookbook/cookbook_toc.html
2. http://www.tldp.org/guides.html
3. http://sources.redhat.com/autobook/

Journals :

1. www.linux-mag.com
2. https://www.linuxjournal.com/
3. man7.org/linux/man-pages/man1/journalctl.1.html

ESE MARKS ALLOCATION

S.No	Category	Marks
1.	**Part A** 20 X1 = 20 Online Examination	20
2.	**Part B** 5 x 6 =30 Either 'A' or 'B' Choice	10
3.	**Part C** 1 x 10 = 10 Compulsory	30
4.	**Total**	**60**

KARPAGAM ACADEMY OF HIGHER EDUCATION
(Deemed to be University)
(Established Under Section 3 of UGC Act, 1956)
(For the candidates admitted from 2018 onwards)
DEPARTMENT OF CS, CA & IT
LESSON PLAN
SUBJECT NAME : OPEN SOURCE TECHNOLOGIES (18CSP302)
SEMESTER : III

		UNIT I	
SI.NO	**Lecture Duration (Hr)**	**Topics to be covered**	**Support Materials**
1	1	Definition of FOSS , GNU	T1:6
2	1	History of GNU/Linux	T1:10
3	1	Free Software Movement	W1
4	1	Advantages of Free Software and GNU/Linux	T1:13
5	1	Advantages of Linux	w1
6	1	FOSS usage	T1:31
7	1	Trends and Potential— Global	R1:31
8	1	Trends and Potential— Indian	R1:33
9	1	Recapitulation and Discussion of Important Questions	
Total no. of Hours Planned for Unit I			**9**

		UNIT II	
SI.NO	**Lecture Duration (Hr)**	**Topics to be covered**	**Support Materials**
1	1	GNU/Linux OS Installation-Detect Hardware, Configure Disk Partitions, File Systems and Install a GNU/Linux Distribution	T1:117
2	1	Basic Shell Commands - Logging In, Listing Files, Editing files, Copying/Moving Files	T1:76
3	1	Viewing File Contents, Changing File Modes and Permissions, Process Management	T1:91
4	1	User and Group Management, File Ownerships and Permissions, PAM Authentication	T1: 95
5	1	Introduction to Common System Configuration Files & Log Files	T1:261
6	1	Configuring Networking, Basics of TCP/IP Networking and Routing, Connecting to the Internet (through dialup, DSL, Ethernet, leased line), Configuring Additional Hardware - Sound Cards	T1: 275
7	1	Displays & Display Cards, Configuring Additional Hardware - Sound Cards, Displays & Display Cards	W3
8	1	Understanding the OS boot up process ; Performing every day tasks using gnu/Linux -- accessing the Internet,	R2:219
9	1	Playing Music, Editing Documents and Spreadsheets, Sending and Receiving Email, Copy files from Disks and over the Network, Playing Games	R2: 223

10	1	X Window System Configuration and Utilities--Configure X Windows, Detect Display Devices ; Installing Software from Source code as well as using binary packages	T1:560
11	1	Recapitulation and Dicussion of Important Questions	
Total Periods Planned for Unit II			**11**
		UNIT III	
SI.NO	**Lecture Duration (Hr)**	**Topics to be covered**	**Support Materials**
1	1	Setting up Email Servers-Using Postfix (SMTP services), Courier (IMAP & POP3 services)	T1:175
2	1	Squirrel Mail (web mail services), Setting up Web Servers -Using Apache (HTTP services),	T1: 177
3	1	PHP (server-side scripting), PERL (CGI support)	T1:153
4	1	Setting up File Services -Using Samba (File and Authentication Services for Windows Networks)	T1:156,w 1
5	1	Using NFS (File Services for GNU/Linux / Unix Networks)	R2: 223
6	1	Setting up Proxy Services --Using Squid (http / ftp / https proxy services)	T1:167
7	1	Setting up Printer Services -Using CUPS (print spooler), Foomatic (Printer Database), Setting up a Firewall -Using Netfilter and iptables	T1:180
8	1	Recapitulation and Dicussion of Important Questions	
Total Periods Planned for Unit III			**8**

		UNIT IV	
SI.NO	**Lecture Duration (Hr)**	**Topics to be covered**	**Support Materials**
1	1	Using the GNU Compiler Collection, GNU Compiler Tools	T1:288
2	1	C Preprocessor (cpp), C Compiler (gcc), C++ Compiler (g++), Assembler (GAS)	T1:320
3	1	Understanding Build Systems -Constructing Make Files and using make using autoconf and autogen to automatically generate make files tailored for different development environments	T1:385
4	1	Using Source Code Versioning and Management Tools --Using cvs to manage source code revisions, patch & diff	T1:394
5	1	Understanding the GNU Libc Libraries and Linker – Linking against Object Archives (.a libraries) and Dynamic Shared Object Libraries (.so libraries)	T1:385
6	1	Generating Statically Linked Binaries and Libraries, Generating Dynamically Linked Libraries, Using the GNU debugging tools --GDB to Debug Programs, Graphical Debuggers like ddd	T1:394
7	2	Memory debugging / profiling libraries mpatrol and valgrind, Review of common programming practises and guidelines for GNU/Linux and FOSS , Introduction to Bash, sed & awk scripting.	T1:351
8	1	Recapitulation and Discussion of Important Questions	

Total Periods Planned for Unit IV			**8**
UNIT V			
SI.NO	**Lecture Duration (Hr)**	**Topics to be covered**	**Support Materials**
1	1	Basics of X window Server Architecture	T1:560
2	1	(Contd) Basics of X window Server Architecture	T1:562
3	1	Qt Programming	T1:578, W2
4	1	(Contd) QT Programming	T1:582
5	1	GTK++ Programming	T1:558
6	1	(Contd)GTK++ Programming	T1:558
7	1	Python Programming	T1 : 473
8	1	Programming with Localisation support	T1:478
9	1	Recapitulation and Dicussion of Important Questions	
10	1	Discussion of Previous ESE Question Papers	
11	1	Discussion of Previous ESE Question Papers	
12	1	Discussion of Previous ESE Question Papers	
Total Periods Planned for Unit V			**12**
Total Periods			**48**

Text Book

T1	Venkateshwarlu, N. B. (2012) Introduction to Linux: Installation and Programming. New Delhi: BPS Publishers.

References

R1	William E. Shotts, Jr. (2012). The Linux Command Line (4th ed.). O'Reilly Publishers.
R2	Christopher Negus .(2015). Linux Bible (9th ed.). Wiley Publications.

Web Sites

w1	http://dsl.org/cookbook/cookbook_toc.html
w2	http://www.tldp.org/guides.html
w3	http://sources.redhat.com/autobook/

Journals :

1. www.linux-mag.com
2. https://www.linuxjournal.com/
3. man7.org/linux/man-pages/man1/journalctl.1.html

UNIT I
SYLLABUS

History and Overview Of GNU/Linux And FOSS 3

Definition of FOSS & GNU, History of GNU/Linux and the Free Software Movement, Advantages of Free Software and GNU/Linux FOSS Usage Trends and Potential—Global and Indian.

UNIT I

History and Overview of GNU/Linux and FOSS 3

Definition of FOSS

Overview of Free/Open Source Software-- Definition of FOSS & GNU

Free and open source software, also F/OSS, FOSS, or FLOSS (free/libre/open source software) is software which is liberally licensed to grant the right of users to study, change, and improve its design through the availability of its source code. This approach has gained both momentum and acceptance as the potential benefits have been increasingly recognized by both individuals and corporate players.

'F/OSS' is an inclusive term generally synonymous with both free software and open source software which describe similar development models, but with differing cultures and philosophies. 'Free software' focuses on the philosophical freedoms it gives to users and 'open source' focuses on the perceived strengths of its peer-to-peer development model. Many people relate to both aspects and so 'F/OSS' is a term that can be used without particular bias towards either camp.

Free software licenses and Open-source licenses are used by many software packages. The licenses have important differences, which mirror the differences in the ways the two kinds of software can be used and distributed and reflect differences in the philosophy behind the two.

Today the terms "free software" and "free open source software" (FOSS) and "free libre open source software" (FLOSS) generally mean the same thing. Despite disagreements about independently important but relatively minor differences, the simple term "open source" originally had the same meaning as FOSS/FLOSS for several years, nicely guarded, but not trademarked, by the Open Source Initiative. However, by mid 2007 enough companies were opening some source to hop on the open source bandwagon, while keeping other advanced functionality closed, that the common meaning of "open source" came to include what is now called "Commercial Open Source Software" (COSS) as well. Today either the specific terms free software/FOSS/FLOSS or COSS are often used instead of the more general term "open source" in order to differentiate between the two different models and preserve the original meaning of the free software/FOSS/FLOSS space.

In January 2008 Hewlett-Packard (HP) announced software governance initiative to address the legal, financial and security risks connected with the adoption of FOSS. The project is supported by other companies like OpenLogic , Google and Novell. FOSSology is a tool for tracking and monitoring the use of free and open-source software within an IT environment and is available under the terms of the GNU General Public License (version 2),FOSSBazaar is a web site that hosts discussion groups and information resources on how to adopt and manage free/open source code.

History of GNU/Linux and the Free Software Movement

The plan for the GNU operating system was publicly announced on September 27, 1983, on the net.unix-wizards and net.usoft newsgroups by Richard Stallman.[4] Software development began on January 5, 1984, when Stallman quit his job at the Massachusetts Institute of Technology's Artificial Intelligence laboratory so that they could not claim ownership or interfere with distributing GNU as free software. Richard Stallman chose the name by using various plays on words, including the song The Gnu. The goal was to bring a wholly free software operating system into existence. Stallman wanted computer users to be free, as most were in the 1960s and 1970s — free to study the source code of the software they use, free to share the software with other people, free to modify the behavior of the software, and

free to publish their modified versions of the software. This philosophy was later published as the GNU Manifesto in March 1985.

Richard Stallman's experience with the Incompatible Timesharing System (ITS), an early operating system written in assembly language that became obsolete due to discontinuation of PDP-10, the computer architecture for which ITS was written, led to a decision that a portable system was necessary. It was thus decided that GNU would be mostly compatible with Unix. At the time, Unix was already a popular proprietary operating system. The design of Unix had proven to be solid, and it was modular, so it could be reimplemented piece by piece.

Much of the needed software had to be written from scratch, but existing compatible free software components were also used such as the TeX typesetting system, and the X Window System. Most of GNU has been written by volunteers; some in their spare time, some paid by companies, educational institutions, and other non-profit organizations. In October 1985, Stallman set up the Free Software Foundation (FSF). In the late 1980s and 1990s, the FSF hired software developers to write the software needed for GNU.

As GNU gained prominence, interested businesses began contributing to development or selling GNU software and technical support. The most prominent and successful of these was Cygnus Solutions, now part of Red Hat.

What is free software and why is it so important for society?

Free software is software that gives you the user the freedom to share, study and modify it. We call this free software because the user is free.

To use free software is to make a political and ethical choice asserting the right to learn, and share what we learn with others. Free software has become the foundation of a learning society where we share our knowledge in a way that others can build upon and enjoy.

Currently, many people use proprietary software that denies users these freedoms and benefits. If we make a copy and give it to a friend, if we try to figure out how the program works, if we put a copy on more than one of our own computers in our own home, we could be caught and fined or put in jail. That's what's in the fine print of the license agreement you accept when using proprietary software.

The corporations behind proprietary software will often spy on your activities and restrict you from sharing with others. And because our computers control much of our personal information and daily activities, proprietary software represents an unacceptable danger to a free society.

The GNU Operating System and the Free Software Movement

What if there were a worldwide group of talented ethical programmers voluntarily committed to the idea of writing and sharing software with each other and with anyone else who agreed to share alike? What if anyone could be a part of and benefit from this community even without being a computer expert or knowing anything about programming? We wouldn't have to worry about getting caught copying a useful program for our friends—because we wouldn't be doing anything wrong.

In fact, such a movement exists, and you can be part of it. The free software movement was started in 1983 by computer scientist Richard M. Stallman, when he launched a project called GNU, which stands for "GNU is Not UNIX", to provide a replacement for the UNIX operating system—a replacement that would respect the freedoms of those using it. Then in 1985, Stallman started the Free Software

Foundation, a nonprofit with the mission of advocating and educating on behalf of computer users around the world.

There are now many variants or 'distributions' of this GNU operating system using the kernel Linux. We recommend those GNU/Linux distributions that are 100% free software; in other words, entirely freedom-respecting.

Today, free software is available for just about any task you can imagine. From complete operating systems like GNU, to over 5,000 individual programs and tools listed in the FSF/UNESCO free software directory. Millions of people around the world — including entire governments — are now using free software on their computers.

Our Core Work

The FSF maintains the Free Software Definition - to show clearly what must be true about a particular software program for it to be considered free software.

The FSF sponsors the GNU project the ongoing effort to provide a complete operating system licensed as free software. We also fund and promote important free software development and provide development systems for GNU software maintainers, including full email and shell services and mailing lists. We are committed to furthering the development of the GNU Operating System and enabling volunteers to easily contribute to that work, including sponsoring Savannah the source code repository

and center for free software development.

The FSF holds copyright on a large proportion of the GNU operating system, and other free software. We hold these assets to defend free software from efforts to turn free software proprietary. Every year we collect thousands of copyright assignments from individual software developers and corporations working on free software. We register these copyrights with the US copyright office and enforce the license under which we distribute free software - typically the GNU General Public License. We do this to ensure that free software distributors respect their obligations to pass on the freedom to all users, to share, study and modify the code. We do this work through our Free Software Licensing and Compliance Lab.

The FSF publishes the GNU General Public License (GNU GPL), the worlds most popular free software license, and the only license written with the express purpose of promoting and preserving software freedom. Other important licenses we publish include the GNU Lesser General Public License (GNU LGPL), the GNU Affero General Public License (GNU AGPL) and the GNU Free Document License (GNU FDL). Read more about our free software licensing and related issues.

Advantages of Free Software

Advantage Of GNU OpenSource Free Software

Open Source's proponents often claim that it offers significant benefits when compared to typical commercial products. Commercial products typically favour visible features (giving marketing advantage) over harder-to measure qualities such as stability, security and similar less glamorous attributes. As shorthand, we shall describe this phenomenon as ***quality vs. features***.

Open Source Software developers are evidently motivated by many factors but favouring features over quality is not noticeable amongst them. For many developers, peer review and acclaim is important, so

it's likely that they will prefer to build software that is admired by their peers. Highly prized factors are clean design, reliability and maintainability, with adherence to standards and shared community values preeminent.

"The Open Source community attracts very bright, very motivated developers, who although frequently unpaid, are often very disciplined. In addition, these developers are not part of corporate cultures where the best route to large salaries is to move into management, hence some Open Source developers are amongst the most experienced in the industry. In addition all users of Open Source products have access to the source code and debugging tools, and hence often suggest both bug fixes and enhancements as actual changes to the source code. Consequently the quality of software produced by the Open Source community sometimes exceeds that produced by purely commercial organisations."

Reliability

Reliability is a loose term. Broadly, we can take it to mean the absence of defects which cause incorrect operation, data loss or sudden failures, perhaps what many people would mean when they use the term `bug'. Strictly, a bug would also mean failure to meet the specification, but since most Open Source projects dispense with the concept of anything easily recognisable as a formal specification, it's hard to point to that as good way of defining what is a bug and what is a feature. Determining what constitutes a

`bug'. Strictly, a bug would also mean failure to meet the specification, but since most Open Source projects dispense with the concept of anything easily recognisable as a formal specification, it's hard to point to that as good way of defining what is a bug and what is a feature. Determining what constitutes a bug is usually by agreement amongst the developers and users of the software (an overlapping community in many cases). Obvious failure to perform is easily recognised as a bug, as is failure to conform to appropriate published standards. Security related failings (exploits or vulnerabilities) are clearly bugs too. Each of these kinds of bugs is usually addressed with speedy fixes wherever possible and Open Source advocates will claim very rapid time-to-fix characteristics for software.

Severe defects tend to be fixed within hours of their being detected, a process which is undoubtedly assisted by the availability of the source code. Able developers who discover a bug will commonly also fix it and then report it to the maintainers as well as issuing an updated version of the software on their own authority. Users of the software can choose whether to use the unofficial fix or wait for an `official'

it and then report it to the maintainers as well as issuing an updated version of the software on their own authority. Users of the software can choose whether to use the unofficial fix or wait for an `official' version. By `official' we mean a release blessed by the project team itself or a trusted authority such as one of the main distributors of Open Source packages. This mechanism clearly works very well in practice. The pattern with closed-source software is typically that a defect report needs to be filed and then there will be a delay before the vendor determines when or whether to issue an updated release. Users of the software are much more at the mercy of the vendor's internal processes than with the Open Source arrangement and the personal experience of the authors is that it can be extremely frustrating to move from the Open Source to the closed model.

"The market greatly values robustness, and the Open Source model, particularly as practiced by Linux, encourages a large market of early adopters (compared to the size of the early market for commercial products) who actively help debug the software. Consequently much Open Source software becomes

Stability

In a business environment software is mostly a necessary evil, a tool to do a job. Unless the job changes or more efficient processes are discovered then there is rarely pressure or need to alter the software that is being used to assist the task. This is more or less directly counter to what motivates a software vendor who are in the unenviable position of supplying a commodity that does not wear out or age much. The vendors need a stable revenue stream to be able to keep their business going whilst their customers have not the slightest desire to change or upgrade any product that is working well enough to suit their needs. If a software supplier can establish a virtual monopoly and then force upgrades onto its audience (as has been the history of the software industry since the mid 1960s) then the profits can be very high.

Software vendors can apply a number of tactics to persuade their customers to upgrade more or less willingly. Typical tactics include moving to allegedly new and improved file formats (which require the new and improved software to read them) or to withdraw support and bug fixes for older versions after a short period.

The problem for users of the software is that they rarely have much control over that process and are left isolated if they choose to remain with older versions that they consider to be acceptable. This has cost and control implications for the business.

Linux FOSS Usage

FOSS Usage

FOSS Manifesto

The Free and Open Source Software community in India calls upon political parties to make FOSS usage and promotion a central part of the IT, e-government and education plans in their election manifestos. FOSS is software which is liberally licensed to grant the right of users to study, change, and improve its design through the availability of its source code. The open, inclusive and participatory nature of FOSS is a natural fit for the vibrant traditions of Indian democracy and its emphasis on sharing knowledge. Since software is the foundation of the digital economy, India's IT infrastructure should be built on FOSS and not on closed, proprietary software systems that enforce restrictive licenses, limit the freedom of users and encourage monopolistic behavior.

We believe that encouragement of FOSS will result in

- ✓ Development of the domestic IT industry
- ✓ Creation of jobs
- ✓ Encouragement of skills development and upgradation
- ✓ Enable localization of software to Indian languages
- ✓ Reduction of India's dependence on monopolistic proprietary software vendors
- ✓ Encourage the usage of open standards
- ✓ Bridging the digital divide
- ✓ Rapid modernization and computerization of India's education system
- ✓ Technology upgradation of India's Small and Medium Enterprises
- ✓ Efficient usage of budget outlays for e-government
- ✓ Faster technology development through Collaborative Innovation

We call upon political parties in India to support the Indian FOSS community by

1. Encouraging the use of FOSS in Indian education system. This will inculcate the virtues of collaboration, sharing and participation in children from a very young age and make computerization of schools affordable.
2. Eliminating proprietary software from the education syllabus and making the syllabus vendor-neutral, thus giving teachers and students the choice of software that suits their budgets and needs. The education system must teach principles and not products. For example, it must teach word processing skills and not endorse specific brands of word-processors.
3. Using FOSS in e-government to the maximum possible extent and ensuring that government tenders are open and do not favor proprietary software vendors. All software developed with tax-payers

3. Using FOSS in e-government to the maximum possible extent and ensuring that government tenders are open and do not favor proprietary software vendors. All software developed with tax-payers money should be released under a FOSS license to encourage collaboration; and the sharing of code and best practices.
4. Mandating the usage of open standards that are free from royalties and vendor lock-in so that the interaction between the government and citizens happens in a free and open manner befitting a democracy.
5. Encouraging freely shareable, FOSS based knowledge repositories like Wikipedia in Indian languages.

6. Encouraging the usage of the collaborative model of FOSS in scientific research. Science thrives on collaboration and the sharing of knowledge. The current trend of privatizing knowledge leads to secrecy in science and reduces collaboration. We must use the FOSS model based on collaboration, community and shared ownership of knowledge to spark a renaissance of knowledge in India.
7. Eliminating ambiguities in Indian Patent Law that allow the surreptitious grant of software and business method patents. Such patents have lead to huge amounts of litigation in developed countries. Indian traditions have held that knowledge grows by sharing and diminishes when hoarded. Patents on software and business methods grant undue monopolies on ideas and prevent independent invention and the sharing of knowledge.

India has one of the most youthful populations in the world and it is important that they have access to the tools with which our modern digital society is built. The freedom to freely distribute FOSS tools, modify the source code, the ability to share knowledge and build communities make Free and Open Source Software the best, long-term model for India's digital future. We therefore urge all political parties to encourage the usage of FOSS for India's development.

Trends and Potential – Global and Indian

Trends and Potential—Global and Indian

FOSS in India

Organisations working on FOSS

- ✓ Open Source Software Resource Centre (OSSRC) - the partnership between Indian Institute of Technology, Mumbai (IIT-B) ,the Center for Development of Advanced Computing (CDAC) and IBM Corporation
- ✓ National Informatics Centre (NIC)
- ✓ It mission.org - Kerala - GNU/Linux support,help and resources
- ✓ Satyam Computer Services Ltd. - Chennai
- ✓ TWINCLING a "not-for-profit" organization in Hyderabad and Secunderabad, A.P. that develops, promotes and showcases opensource software.
- ✓ DeepRoot Linux a company specialising in developing gnu/Linux based products, services and solutions.

FOSSCOMM: Free and Open Source COMMunity in India

FOSS (Free and Open Source Software) activists are getting together to promote the awareness, influence Govt policy, defend the software freedom, bring in changes in IT education curriculum, work towards free information infrastructure.

Principles and Objectives

1. Belief in boundless knowledge, free/open software and free/open standards and their advantages for society
2. Work for the adoption and promotion of FOSS in Government policies as well as Industry
3. Promotion of FOSS only software in syllabus from primary to higher education
4. Encouragement for student commitment and contribution to FOSS
5. Promotion of accessing publicly funded software and creative works under free software or equivalent licenses
6. Resistance for the hardware and software monopolies by using anti-competition laws (Hardware drivers and import policy) + free/open specifications + anti-bundling .

The network is taking shape at http://fosscomm.in. One of the first tasks the network will be working is to get the open standards policy document for eGovernance in India approved without yielding to those groups who promote the interests of proprietary software groups (like NASSCOM, MAIT, who are influencing the Indian Govt to make room for RAND and multiple standards). Currently, the apex committee does not contain members who represent the FOSS community. Therefore one of the immediate objectives of the new network of FOSS advocates is to request the Govt to include FOSS representatives in the apex body.

Introduction to Opensource & Advantage

When it was formed in 1998, the OSI consisted of a small number of dedicated individuals with a shared aim of furthering the goals of open software. Although the composition of this board of directors changed over the years, it wasn't until 2005 that the size was increased from 5 to 9, and actively began expanding the organisation's focus beyond licence evaluation. Also in 2005 several initiatives were dentified, among them this one: to investigate the feasibility of changing the OSI to be a broad-based membership organisation. That's what this forum is all about: **should** OSI become member-based? If so, who can be a member? How does a member benefit by *being* a member? What about membership tiers, and corporate membershipd? Who's the actual target demographic for the OSI? Developers? End users? Commercial repackagers and integrators? These are all issues to be investigated in this forum.

The Open Source Definition

Introduction

Open source doesn't just mean access to the source code. The distribution terms of open-source software must comply with the following criteria:

1. Free Redistribution

The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license shall not require a royalty or other fee for such sale.

2. Source Code

The program must include source code, and must allow distribution in source code as well as compiled form. Where some form of a product is not distributed with source code, there must be a well-publicized means of obtaining the source code for no more than a reasonable reproduction cost

3. Derived Works

The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.

4. Integrity of The Author's Source Code

The license may restrict source-code from being distributed in modified form *only* if the license allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time. The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software.

5. No Discrimination Against Persons or Groups

The license must not discriminate against any person or group of persons.

6. No Discrimination Against Fields of Endeavor

The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.

7. Distribution of License

The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.

8. License Must Not Be Specific to a Product

The rights attached to the program must not depend on the program's being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software distribution.

9. License Must Not Restrict Other Software

The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be open-source software.

10. License Must Be Technology-Neutral

No provision of the license may be predicated on any individual technology or style of interface.

PART – A (20 X 1 = 20 MARKS)
ONLINE QUESTIONS

POSSIBLE QUESTIONS (2 Marks)

1. Define Foss.
2. What is Free Software?
3. Write the difference between Free and Open Source Software
4. List few Open Source Software.
5. Write the Difference between Linux and Unix.

POSSIBLE QUESTIONS (6 Marks)

1. Write about history of GNU/Linux and Free Software Movement.
2. Describe Trends and Potential in Global and Indian for Linux Software.
3. What are the advantages of Free Software and GNU/Linux.
4. Give a brief description about FOSS usage in Linux software.
5. What are the merits of Free Software Movement.

UNIT II

SYLLABUS

System Administration

GNU/Linux OS Installation, Detect Hardware, Configure Disk Partitions & File Systems and Install a GNU/Linux Distribution ; Basic shell commands -Logging in, Listing Files, Editing Files, Copying/Moving Files, Viewing File Contents, Changing File Modes and Permissions , Process Management ; User and Group Management, File Ownerships and Permissions, PAM Authentication ; Introduction to Common System Configuration Files & Log Files ;

Configuring Networking Basics of TCP/IP Networking and Routing connecting to the Internet (through dialup DSL Ethernet leased line) ; Configuring Additional Hardware -Sound Cards, Displays & Display Cards, Network Cards, Modems, USB drives, CD writers ;

Understanding the OS Boot up Process ; Performing every day tasks using GNU/Linux -- Accessing the Internet, Playing Music, Editing Documents and Spreadsheets, Sending and Receiving Email, Copy Files from disks and over the Network, Playing games, Writing CDs ;

X Window System Configuration and Utilities, Configure X Windows, Detect Display Devices ; Installing software from source code as well as using binary packages.

UNIT II

SYSTEM ADMINISTRATION

What is GNU/Linux?

Linux is an operating system: a series of programs that let you interact with your computer and run other programs.

An operating system consists of various fundamental programs which are needed by your computer so that it can communicate and receive instructions from users; read and write data to hard disks, tapes, and printers; control the use of memory; and run other software. The most important part of an operating system is the kernel. In a GNU/Linux system, Linux is the kernel component. The rest of the system consists of other programs, many of which were written by or for the GNU Project.

Supported Hardware

Debian does not impose hardware requirements beyond the requirements of the Linux or kFreeBSD kernel and the GNU tool-sets. Therefore, any architecture or platform to which the Linux or kFreeBSD kernel, libc, **gcc**, etc. have been ported, and for which a Debian port exists, can run Debian. Rather than attempting to describe all the different hardware configurations which are supported for 32-bit PC, this section contains general information and pointers to where additional information can be found.

GNU/Linux OS installation

Detect Hardware

CPU, Main Boards, and Video Support

Nearly all x86-based (IA-32) processors still in use in personal computers are supported, including all varieties of Intel's "Pentium" series. This also includes 32-bit AMD and VIA (former Cyrix) processors, and processors like the Athlon XP and Intel P4 Xeon.

However, Debian GNU/Linux jessie will *not* run on 486 or earlier processors. Despite the architecture name "i386", support for actual 80386 and 80486 processors (and their clones) was dropped with the Sarge (r3.1) and Squeeze (r6.0) releases of Debian, respectively. The Intel Pentium and clones, including those without an FPU (Floating-Point Unit or math coprocessor), are supported. The Intel Quark is *not* supported, due to hardware errata.

I/O Bus

The system bus is the part of the motherboard which allows the CPU to communicate with peripherals such as storage devices. Your computer must use the ISA, EISA, PCI, PCIe, PCI-X, or VESA Local Bus (VLB, sometimes called the VL bus). Essentially all personal computers sold in recent years use one of these.

Laptops

From a technical point of view, laptops are normal PCs, so all information regarding PC systems applies to laptops as well. Installations on laptops nowadays usually work out of the box, including things like automatically suspending the system on closing the lid and laptop specfic hardware buttons like those for disabling the wifi interfaces (“airplane mode”).

Multiple Processors

Multiprocessor support — also called “symmetric multiprocessing” or SMP — is available for this architecture. The standard Debian 8 kernel image has been compiled with *SMP-alternatives* support. This means that the kernel will detect the number of processors (or

Graphics Hardware Support

Debian's support for graphical interfaces is determined by the underlying support found in X.Org's X11 system, and the kernel. Basic framebuffer graphics is provided by the kernel, whilst desktop environments use X11. On modern PCs, having a graphical display usually works out of the box. In very few cases there have been reports about hardware on which installation of additional graphics card firmware was required even for basic graphics support, but these have been rare exceptions. For quite a lot of hardware, 3D acceleration also works well out of the box, but there is still some hardware that needs binary blobs to work well.

Network Connectivity Hardware

Almost any network interface card (NIC) supported by the Linux kernel should also be supported by the installation system; drivers should normally be loaded automatically. This includes most PCI/PCI-Express cards as well as PCMCIA/Express Cards on laptops. Many older ISA cards are supported as well.

ISDN is supported, but not during the installation.

Wireless Network Cards

Wireless networking is in general supported as well and a growing number of wireless adapters are supported by the official Linux kernel, although many of them do require firmware to be loaded.

Wireless NICs that are not supported by the official Linux kernel can generally be made to work under Debian GNU/Linux, but are not supported during the installation.

Braille Displays

Support for braille displays is determined by the underlying support found in **brltty**. Most displays work under **brltty**, connected via either a serial port, USB or bluetooth.

Hardware Speech Synthesis

Support for hardware speech synthesis devices is determined by the underlying support found in **speakup**. **speakup** only supports integrated boards and external devices connected to a serial port (no USB, serial-to-USB or PCI adapters are supported).

Peripherals and Other Hardware

Linux supports a large variety of hardware devices such as mice, printers, scanners, PCMCIA/CardBus/ExpressCard and USB devices. However, most of these devices are not required while installing the system.

Architecture	Debian Designation	Subarchitecture	Flavor

Architecture	Debian Designation	Subarchitecture	Flavor
Intel x86-based	i386		
AMD64 & Intel 64	amd64		
ARM	armel	Intel IXP4xx	ixp4xx
		Marvell Kirkwood	kirkwood
		Marvell Orion	orion5x
		Versatile	versatile
ARM with hardware FPU	armhf	multiplatform	armmp
		multiplatform for LPAE-capable systems	armmp-lpae
64bit ARM	arm64		
MIPS (big endian)	mips	SGI IP22 (Indy/Indigo 2)	r4k-ip22
		SGI IP32 (O2)	r5k-ip32
		MIPS Malta (32 bit)	4kc-malta
		MIPS Malta (64 bit)	5kc-malta
MIPS (little endian)	mipsel	MIPS Malta (32 bit)	4kc-malta
		MIPS Malta (64 bit)	5kc-malta
IBM/Motorola PowerPC	powerpc	PowerMac	pmac
		PReP	prep

Architecture	Debian Designation	Subarchitecture	Flavor
Power Systems	ppc64el	IBM POWER8 or newer machines	
64bit IBM S/390	s390x	IPL from VM-reader and DASD	generic

Configure Disk Partitions and File Systems

You must have at least 80MB of memory and 680MB of hard disk space to perform a normal installation. Note that these are fairly minimal numbers.

Here's a road map for the steps you will take during the installation process.

1. Back up any existing data or documents on the hard disk where you plan to install.
2. Gather information about your computer and any needed documentation, before starting the installation.
3. Locate and/or download the installer software and any specialized driver or firmware files your machine requires.
4. Set up boot media such as CDs/DVDs/USB sticks or provide a network boot infrastructure from which the installer can be booted.
5. Boot the installation system.
6. Select the installation language.
7. Activate the ethernet network connection, if available.
8. If necessary, resize existing partitions on your target harddisk to make space for the installation.
9. Create and mount the partitions on which Debian will be installed.
10. Watch the automatic download/install/setup of the *base system*.

11. Install a *boot loader* which can start up Debian GNU/Linux and/or your existing system.
12. Load the newly installed system for the first time.

Basic Shell Commands

Logging in Commands

man This command brings up the online Unix manual. Use it on each of the commands below.

For Example:

man pwd You will see the manual for the pwd command.

pwd Shows what directory (folder) you are in.

cd Changes directories.

cd muondata Moves down from your current directory into the muondata sub-directory

cd .. Moves up one directory (yes, include the two little dots)

cd /home/particle/muondata Moves from ANY directory into the muondata sub-directory of your home directory.

cd ~ Takes you back to your home directory(/home/particle)

mkdir dirName Creates a directory with name dirName.

For Example:

mkdir temp Creates the directory temp.

rmdir dirName Removes a directory dirName.

For Example:

rmdir temp Removes the directory temp

Listing Files

ls Lists files.

ls al

ls -al |more Shows one screen of file names at a time.

less data1 Dumps the contents of the data1 file to your screen with a pause

whereis data1 Shows you the location of the data1 file.

Editing Files

rm data1 Deletes the file data1 in the current directory.

rm -i muon* Removes all of your muon data files

Copying / Moving Files

mv data1 newdata/ moves the file data1 to the folder newdata and deletes the old one.

cp data1 newdata/ will copy the file data1 to the directory newdata (assuming it has already been created)

Changing File Modes and Permissions

chown [option(s)] username.group file(s)

Transfers the ownership of a file to the user with the specified user name.

RChanges files and directories in all subdirectories.

chgrp [option(s)] groupname file(s)

Transfers the group ownership of a given file to the group with the specified group name. The file owner can only change group ownership if a member of both the existing and the new group.

chmod [options] mode file(s) Changes the access permissions.

File Systems

mount [option(s)] [<device>] mountpoint

This command can be used to mount any data media, such as hard disks, CD-ROM drives, and other drives, to a directory of the Linux file system.

umount [option(s)] mountpoint

This command unmounts a mounted drive from the file system. To prevent data loss, run this command before taking a removable data medium from its drive.

USER AND GROUP MANAGEMENT

What Users and Groups Are

The control of users and groups is a core element of Red Hat Enterprise Linux system administration. The user of the system is either a human being or an account used by specific

applications identified by a unique numerical identification number called *user ID* (UID). Users within a group can have read permissions, write permissions, execute permissions or any combination of read/write/execute permissions for files owned by that group.

Red Hat Enterprise Linux supports *access control lists* (ACLs) for files and directories which allow permissions for specific users outside of the owner to be set. A group is an organization unit tying users together for a common purpose, which can be reading permissions, writing permission, or executing permission for files owned by that group. Similar to UID, each group is associated with a group ID (GID).

Red Hat Enterprise Linux reserves user and group IDs below 500 for system users and groups. By default, the **User Manager** does not display the system users. Reserved user and group IDs are documented in the setup package. To view the documentation, use this command:

Adding a New User

If there is a new user you need to add to the system, follow this procedure:

- Click the **Add User** button.
- Enter the user name and full name in the appropriate fields
- Type the user's password in the **Password** and **Confirm Password** fields. The password must be at least six characters long.

1. Select a login shell for the user from the **Login Shell** drop-down list or accept the default value of**/bin/bash**.
2. Clear the **Create home directory** check box if you choose not to create the home directory for a new user in **/home/*username*/**.

- You can also change this home directory by editing the content of the **Home Directory** text box. Note that when the home directory is created, default configuration files are copied into it from the **/etc/skel/ eate a private group for the user** check box if you do not want a unique group with tdirectory.

3. Clear the **Cr**he same name as the user to be created. User private group (UPG) is a group assigned to a user account to which that user exclusively belongs, which is used for managing file permissions for individual users.
4. Specify a user ID for the user by selecting **Specify user ID manually**. If the option is not selected, the next available user ID above 500 is assigned to the new user.
5. Click the **OK** button to complete the process.

PAM (Pluggable Authentication Modules)

Pluggable authentication modules are at the core of user authentication in any modern linux distribution.

Distributions that support pam.

Nearly all popular distributions have supported PAM for some time. Here's an incomplete list of distributions that support PAM:

- Redhat since version 5.0
- Mandrake since 5.2
- Debian since version 2.1 (partial support in 2.1 -- complete support in 2.2)
- Caldera since version 1.3
- Turbolinux since version 3.6

- SuSE since version 6.2

Installing PAM

PAM configuration files

PAM configuration files are stored in the /etc/pam.d/ directory. (If you don't have /etc/pam.d/ directory, don't worry, I'll cover that in the next section) Let's go over there and take a look.

```
~$ cd /etc/pam.d
/etc/pam.d/$ ls
chfn    chsh    login    other    passwd  su      xlock
/etc/pam.d/$
```

Let's take a look the PAM configuration file for login (I've condensed the file for the sake of simplicity):

```
/etc/pam.d/$ cat login
# PAM configuration for login
auth       requisite  pam_securetty.so
auth       required   pam_nologin.so
auth       required   pam_env.so
auth       required   pam_unix.so nullok
account    required   pam_unix.so
session    required   pam_unix.so
```

```
session    optional    pam_lastlog.so
password   required    pam_unix.so nullok obscure min=4 max=8
```

Configuration syntax

PAM configuration files have the following syntax:

```
type  control  module-path  module-arguments
```

Using the login configuration file (see above) as an example let's take a look a the syntax for PAM configuration files:

PAM configuration tokens

type

The type token tells PAM what type of authentication is to be used for this module. Modules of the same type can be "stacked", requiring a user to meet multiple requirements to be authenticated. PAM recognizes four types:

account

Determines whether the user is allowed to access the service, whether their passwords has expired, etc.

auth

Determines whether the user is who they claim to be, usually by a password, but perhaps by a more sophistcated means, such as biometrics.

password

Provides a mechanism for the user to change their authentication. Again, this usually their password.

session

Things that should be done before and/or after the user is authenticed. This might included things such as mounting/unmounting the user home directory, logging their login/logout, and restricting/unrestricting the services available to the user.

control

The control token tells PAM what should be done in if authentication by this module fails. PAM recognizes four control types:

requisite

Failure to authenticate via this module results in immediate denial of authentication.

required

Failure also results in denial of authentication, although PAM will still call all the other modules listed for this service before denying authentication.

sufficient

If authentication by this module is successful, PAM will grant authentication, even if a previous required module failed.

optional

Whether this module succeeds or fails is only significant if it is the only module of its type for this service.

Introduction to common system Linux Configuration Files & Log Files

profile	System wide environment and startup script program.
/dev/MAKEDEV	The /dev/MAKEDEV file is a script written by the system administrator that creates local only device files or links such as device files for a non-standard device driver.
/etc/aliases	Where the user's name is matched to a nickname for e-mail.
/etc/bootptab	The configuration for the BOOTP server daemon.
/etc/crontab	Lists commands and times to run them for the cron deamon.
/etc/dhcpd.conf	The configuration file for the DHCP server daemon.
/etc/ethers	File for RARP mapping from hardware addresses to IP addresses. See the man page ethers(5).
/etc/exports	The file describing exported filesystems for NFS services.
/etc/fdprm	The floppy disk parameter table. Describes the formats of different floppy disks. Used by setfdprm.
/etc/filesystems	Can be used to set the filesystem probe order when filesystems are mounted with the auto option. The nodev parameter is specified for filesystems that are not really locally mounted systems such as proc, devpts, and nfs

	systems.
/etc/fstab	Lists the filesystems mounted automatically at startup by the mount -a command (in /etc/rc or equivalent startup file).
/etc/group	Similar to /etc/passwd but for groups rather than users.
/etc/groups	May contain passwords that let a user join a group.
/etc/gshadow	Used to hold the group password and group administrator password information for shadow passwords.
/etc/host.conf	Specifies how host names are resolved.
/etc/hosts	List hosts for name lookup use that are locally required.
/etc/HOSTNAME	Shows the host name of this host. Used for support of older programs since the hostname is stored in the /etc/sysconfig/network file.
/etc/inittab	Configuration file for init, controls startup run levels, determines scripts to start with.
/etc/inetd.conf	Sets up the services that run under the inetd daemon.
/etc/issue	Output by getty before the login prompt. Description or welcoming message.

/etc/issue.net	Output for network logins with LINUX version
/etc/ld.so.conf	Configuration file for ld.so, the run time linker.
/etc/lilo.conf	Configuration file for LILO.
/etc/limits	Limits users resources when a system has shadow passwords installed.
/etc/localtime	In Debian the system time zone is determined by this link.
/etc/login.defs	Sets user login features on systems with shadow passwords.
/etc/logrotate.conf	Configures the logrotate program used for managing logfiles.
/etc/magic	The configuration file for file types. Contains the descriptions of various file formats for the file command.
/etc/motd	The message of the day, automatically output by a successful login.
/etc/mtab	A list of currently mounted file systems. Setup by boot scripts and updated by the mount command.
/etc/named.conf	Used for domain name servers.
/etc/networks	Lists names and addresses of your own and other networks, used by the route command.

/etc/nologin	If this file exists, non-root logins are disabled. Typically it is created when the system is shutting down.
/etc/nsswitch.conf	Name service switch configuration file.
/etc/passwd	The user database with fields giving the username, real name, home directory, encrypted password and other information about each user.
/etc/printcap	A configuration file for printers.
/etc/profile, /etc/cshlogin, /etc/csh/cshrc	Files executed at login or startup time by the Bourne or C shells. These allow the system administrator to set global defaults for all users.
/etc/protocols	Describes DARPA internet protocols available from the TCP/IP subsystem. Maps protocol ID numbers to protocol names.
/etc/rc or /etc/rc.d or /etc/rc?.d	Scripts or directories of scripts to run at startup or when changing run level.

Configuring networking basics of TCP/IP networking and Routing Connecting to the Internet

Linux and other Unix operating systems use the TCP/IP protocol. It is not a single network protocol, but a family of network protocols that offer various services. TCP/IP was developed

based on an application used for military purposes and was defined in its present form in an RFC in 1981. RFC stands for *Request for Comments*. They are documents that describe various Internet protocols and implementation procedures for the operating system and its applications. Since then, the TCP/IP protocol has been refined, but the basic protocol has remained virtually unchanged.

IP Addresses

Every computer on the Internet has a unique 32-bit address. These 32 bits (or 4 bytes) are normally written as illustrated in the second row in Table 14.1. "How an IP Address is Written".

How an IP Address is Written

IP Address (binary): 11000000 10101000 00000000 00010100
IP Address (decimal): 192. 168. 0. 20

In decimal form, the four bytes are written in the decimal number system, separated by periods. The IP address is assigned to a host or a network interface. It cannot be used anywhere else in the world. There are certainly exceptions to this rule, but these play a minimal role in the following passages.

Netmasks and Routing

Netmasks were conceived for the purpose of informing the host with the IP address 192.168.0.0 of the location of the host with the IP address 192.168.0.20. To put it simply, the netmask on a host with an IP address defines what is internal and what is external. Hosts located internally (professionals say, "in the same subnetwork") respond directly. Hosts located externally ("not in

the same subnetwork") only respond via a gateway or router. Because every network interface can receive its own IP address, it can get quite complicated.

Before a network packet is sent, the following runs on the computer: the IP address is linked to the netmask via a logical AND and the address of the sending host is likewise connected to the netmask via the logical AND. If there are several network interfaces available, normally all possible sender addresses are verified. The results of the AND links will be compared. If there are no discrepancies in this comparison, the destination, or receiving host, is located in the same subnetwork. Otherwise, it must be accessed via a gateway. The more "1" bits are located in the netmask, the fewer hosts can be accessed directly and the more hosts can be reached via a gateway. Several examples are illustrated in Table 14.2. "Linking IP Addresses to the Netmask".

Linking IP Addresses to the Netmask

IP address (192.168.0.20): 11000000 10101000 00000000 00010100
Netmask (255.255.255.0): 11111111 11111111 11111111 00000000

Result of the link: 11000000 10101000 00000000 00000000
In the decimal system: 192. 168. 0. 0

IP address (213.95.15.200): 11010101 10111111 00001111 11001000
Netmask (255.255.255.0): 11111111 11111111 11111111 00000000

Result of the link: 11010101 10111111 00001111 00000000
In the decimal system: 213. 95. 15. 0

Configuring Additional Hardware

Configuring Sound Cards

Linux needs a driver to control the sound card. The initial Linux kernel (after you install Linux from the companion CD-ROMs) does not load the sound driver. The sound drivers are provided as loadable modules you can load after booting Linux. You will find the sound drivers in the /lib/modules/2.4.*/kernel/drivers/sound directory, where VERSION is the kernel version number. If you look at the names of the module files, you see that each filename ends with the extension .o.gz (the .o extension identifies an object file—a file containing binary instructions that implements the driver and .gz means that the files are compressed).

Checking Information about a Sound Card

Typically, you can find out information about your PC's sound card from what kudzu finds as it probes the hardware. Kudzu stores the information about all detected hardware in the following file:

```
/etc/sysconfig/hwconf
```

You can open that file in a text editor and look for a line that looks like this:

```
class: AUDIO
```

The information about the sound card is in the lines that follow, up to the next line that has a lone hyphen. For example, here is the information for an ISA bus Yamaha OPL3SA2 sound card:

```
class: AUDIO
```

bus: ISAPNP

detached: 0

driver: ad1848

desc: "OPL3-SA3 Snd System:Unknown"

deviceId: YMH0021

pdeviceId: YMH0030

native: 1

active: 0

cardnum: 0

logdev: 0

dma: 0,0

Configuring Display Cards

The commands to do that are:

$ sudo apt-get remove nvidia* && sudo apt-get autoremove

Next, reboot and when you're back at the login screen press CTRL + Alt + F1 to switch to command console. Login here with your username and password. When you're at the text console, you'll have to kill the current graphics session by running **$ sudo stop lightdm**.

Finally, give permissions to the downloaded driver package and run it with:

cd /Downloads && chmod +x NVIDIA-Linux-*-346.35.run && sudo sh NVIDIA-Linux-*-361.42.run

Configuring Network Card

The "ifconfig" command is used for displaying current network configuration information, setting up an ip address, netmask or broadcast address to an network interface, creating an alias for network interface, setting up hardware address and enable or disable network interfaces.

Using the "ifconfig" command with "netmask" argument and interface name as (eth0) allows you to define an netmask to an given interface. For example, "ifconfig eth0 netmask 255.255.255.224" will set the network mask to an given interface eth0.

To assign an IP address, Netmask address and Broadcast address all at once using "ifconfig" command with all arguments as given below.

```
[root@tecmint ~]# ifconfig eth0 172.16.25.125 netmask 255.255.255.224 broadcast 172.16.25
```

View All Network Setting

The "ifconfig" command with no arguments will display all the active interfaces details. The ifconfigcommand also used to check the assigned IP address of an server.

```
[root@tecmint ~]# ifconfig
```

Display Information of All Network Interfaces

The following ifconfig command with -a argument will display information of all active or inactive network interfaces on server. It displays the results for eth0, lo, sit0 and tun0.

Configuring Modems

Understanding modem commands and setup strings

To create a new dialer, you need to understand how modem commands are used. You can enable or disable desired features by sending commands in a setup string to the modem. For example, the following setup string is used in the *dialHA24* dialer:

```
ATQ0E0T&D2&C1S0=0X4S2=043
```

You can change these strings to suit a different communications protocol or modem-specific commands by consulting the documentation for your modem. Though the setup commands may seem confusing because they are concatenated, there are two basic types of AT commands:

- basic modem commands (for example, Q0 and &D2)
- modem S-registers (for example, S0=0)

AT-compatible command strings always begin with the **AT** (attention) command.

Creating a new atdial dialer

You can create a new *atdial* dialer without using a development system. An *atdial* dialer is actually a link to the binary */usr/lib/uucp/atdialer* that calls a configuration file in

the */usr/lib/uucp/default* directory. The configuration file contains all the commands specific to that modem. For example, *atdialHAY* is linked to *atdialer* and the configuration file is in */usr/lib/uucp/default/atdialHAY*.

To create a new *atdial* dialer, *atdialMINE* for example, follow these steps:

1. Log in as *root*.
2. Copy one of the *atdial** files in */usr/lib/uucp/default* to use as a template for the new configuration file *atdialMINE* in the same directory. For example, to use *atdialW96* as the template, enter:

   cp /usr/lib/uucp/default/atdialW96 /usr/lib/uucp/default/atdialMINE
3. Edit */usr/lib/uucp/default/atdialMINE* to add and alter the parameters that are appropriate for your modem. See the **atdialer**(C) manual page for more information.
4. Create a symbolic link */usr/lib/uucp/atdialMINE* to */usr/lib/uucp/atdialer* using the command:

   ln -s /usr/lib/uucp/atdialer /usr/lib/uucp/atdialMINE

Configuring USB Drives

USB devices (peripherals) include flash drives, printers, routers, scanners, and drafting pens. Most computer peripherals use USB (Universal Serial Bus) technology to connect to the computer. Configuring a USB device for Windows Vista is easy if you know how to set up the correct drivers for your USB device.

Choose Start→Control Panel→Hardware and Sound→Device Manager.

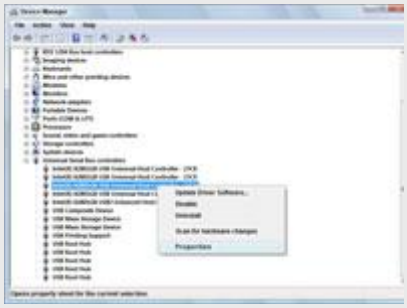
The Device Manager shows all the hardware associated with your computer as well as the health of that hardware.

Click the plus sign to the left of the Universal Serial Bus Controllers item.

This opens a list of all devices that either are plugged into a USB drive or have previously been set up on your computer.

Right-click an item and choose Enable to enable it.

Or click Disable to disable it.



Right-click the item you want to configure, choose Properties, and then click the Driver tab.

Click the buttons on the Driver sheet to manage the driver; you can view details about it, upgrade it to a newer version, or uninstall it.

Click OK.

This will save your updated USB device settings.

Configuring CD writers

How to use your CD writer under Linux

First, you need a compatible CD burner. Linux works with SCSI CD-burners, IDE CD-burners (through ide-scsi emulation), and USB CD burners. In all cases, the device appears as a SCSI device. If you can see your device with the command 'cdrecord -scanbus', then you are ready to burn. If so, skip to section 2.

Section 1: Configuring your CD writer

Often, people have a IDE cd-writer, but don't have the ide-scsi emulation turned on. This is best done by editing the*/etc/lilo.conf* on your computer, and adding an entry to your current boot option:

```
image=/boot/vmlinuz-STD-current
  label=linux
  read-only
  root=/dev/device
  append="hdb=ide-scsi"
```

(NOTE: this is just an example, your root partition and CD-writer device might be different. If you have any problems identifying or configuring your CD-writer, contact help@cs.cmu.edu). Once you have configured lilo, re-run */sbin/lilo* to install the new bootloader, and reboot.

Section 2: Using cdrecord

The output of 'cdrecord -scanbus' will look something like this:

```
# cdrecord -scanbus
Cdrecord 1.9 (i686-pc-linux-gnu) Copyright (C) 1995-2000 Jörg Schilling
Linux sg driver version: 3.1.22
Using libscg version 'schily-0.1'
scsibus0:
    0,0,0    0) 'TOSHIBA ' 'DVD-ROM SD-R1202' '1020' Removable CD-ROM
    0,1,0    1) *
    0,2,0    2) *
    0,3,0    3) *
    0,4,0    4) *
    0,5,0    5) *
    0,6,0    6) *
    0,7,0    7) *
```

The CD writer is the device labeled '0,0,0' in this example. You can now use cdrecord to burn an audio or data CD, using that device number. Here's an example of how to burn an ISO image to a cd, using cdrecord:

```
# cdrecord -v speed=2 dev=0,0,0 -data cdimage.iso
```

Accesing Internet in Linux

Determine Your Wireless Network Interface

From within the terminal enter the following command:

iwconfig

The most common wireless network interface is wlan0 but can be other things such as in my case it is wlp2s0.

Turn The Wireless Interface On

The next step is to make sure the wireless interface is turned on.

Use the following command to do this:

sudo ifconfig wlan0 up

Replace the wlan0 with the name of your network interface.

Scan For Wireless Access Points

Now that your wireless network interface is up and running you can search for networks to connect to.

Type the following command:

sudo iwlist scan | more

Now that your wireless network interface is up and running you can search for networks to connect to.

Type the following command:

sudo iwlist scan | more

Sending and Receiving emails

Task: Compose mail

Use following format:
mail -s <subject> <mailaddress>

For example write mail to boss@yahoo.com:

```
$ mail - "Hello" boss@yahoo.com
```

You are then expected to type in your message, followed by an â€˜control-Dâ€™ at the beginning of a line. To stop simply type dot (.):
Output:

Hi,

This is a test

Cc:

Task: Mail a whole text file

Mail a whole text file called demo.txt to boss@yahoo.com:

```
$mail -s "Report 05/06/07" boss@yahoo.com < demo.txt
```

You can also type mutt or pine to read and send mail:

```
$ mutt
```

OR

```
$ pine
```

Copy Files from disks and over the network

PCManFM file manager can be launched from the menu within the LXDE desktop environment.

This file manager is fairly basic along the lines of Thunar.

You can copy files by selecting them with the mouse. To copy the file press the CTRL and C key at the same time or right click on the file and choose "copy" from the menu.

To paste the file press CTRL and V in the folder you wish to copy the file to. You can also right-click and choose "paste" from the menu.

Dragging and dropping a file does not copy a file, it moves it.

There is an option when right clicking on a file called "copy path". This is useful if you want to paste the URL of the file in a document or on the command line for any reason.

Playing Games in Linux

First, install the correct graphics drivers. This depends on your video card.

Getting GeForce drivers for Linux

Open a terminal and enter the following commands (you can copy and paste by pressing *Ctrl+Shift+V*), remembering to press *Enter* after each one:

sudo add-apt-repository ppa:ubuntu-x-swat/x-updates

sudo apt-get update

sudo apt-get install nvidia-current

Reboot, and you're good to go. You'll automatically get updates for this driver in the future.

X window system configuration

For Linux systems, the graphical user interface of choice is the X Window System.

In order to run X, you need to have the necessary packages installed. If you selected the ``X Window System" component to be installed when you originally installed Red Hat Linux, everything should be ready to go.

XFree86 Configuration

There are three methods for configuring XFree86 on your machine:

- Xconfigurator
- xf86config
- by hand

Xconfigurator and xf86config are functional equivalents and should work equally well. If you are unsure of anything in this process, a good source of additional documentation is:

http://www.xfree86.org

Xconfigurator is a full-screen menu driven program that walks you through setting up your X server. xf86config is a line oriented program distributed with XFree86. It isn't as easy to use asXconfigurator, but it is included for completeness.

The X Server

Provided you selected the proper video card at install time, you should have the proper X server installed. When later running Xconfigurator or xf86config, you need to make sure you select the same video card or the autoprobe will fail.

For instance, if the CD is mounted on /mnt/cdrom, and you need to install the S3 server, enter the following commands:

```
cd /mnt/cdrom/RedHat/RPMS
rpm -ivh XFree86-S3-3.1.2-1.i386.rpm
ln -sf ../../usr/X11R6/bin/XF86S3 /etc/X11/X
```

This will install the S3 server and make the proper symbolic link.

Xconfigurator

To configure X Windows you must first select your video card. Scroll down the list of supported cards until you locate the card in your machine. Figure 52 may help you determine the video server that matches your hardware. If your card is not listed it may not be supported by XFree86. In this case you can try the last card entry on the list (Unlisted Card) or a commercial X Windows server.

The next step is to select your monitor. If your monitor is not listed you can select one of the generic monitor entries or ``Custom'' and enter your own parameters. Custom monitor configuration is recommended only for those who have a sound understanding of the inner workings of CRT displays. The average user should probably use one of the generic selections from the list. After selecting a monitor you need to tell Xconfigurator how much video memory you have.

PART – A (20 X 1 = 20 MARKS)

ONLINE QUESTIONS

POSSIBLE QUESTIONS(2 Marks)

1. What commands are used for logging?
2. Define Routing.
3. What are the types of Modems?
4. What are the commands for Process Management?
5. Define Windows Architecture
6. What is Sound Card?

POSSIBLE QUESTIONS(6 Marks)

1. Explain configuring and installation of Linux Operating System.
2. Write about configuring sound card in Linux.
3. Explain configuring, disk partition, installation of Linux operating System.
4. Write about configuring modems and CD writers in Linux.
5. Explain Linux commands for Logging and Files with example.
6. Write about TCP/IP networking and Routing in Linux.
7. Explain Process and Group Management commands with example.
8. Write about X Window system Configuration and Utilities.

9. Explain File ownership and PAM authentication in Linux.
10. Write about Utilities and Configuring of X Window System.