KARPAGAM ACADEMY OF HIGHER EDUCATION
(Deemed to be University)
(Established Under Section 3 of UGC Act, 1956)
(For the candidates admitted from 2017 onwards)
DEPARTMENT OF CS, CA & IT

SUBJECT NAME : OPEN SOURCE TECHNOLOGIES
SEMESTER : III
SUBJECT CODE : 17CSP302 **CLASS: II M.Sc CS**

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

Scope: 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.

Objectives:

- To understand fundamental operating system abstractions such as Processes, Threads, files Semaphores, IPC, abstractions, shared memory regions etc.
- To Understand how the operating system abstractions can be used in the development of application programs or to build higher level abstractions
- To Understand how the operating system abstractions can be implemented
- To 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.

SUGGESTED READINGS

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

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

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

Web Sites:

1. http://www.oreilly.com/catalog/ open sources/book/toc.html

2. http://dsl.org/cookbook/cookbook_toc.html

3 http://cvsbook.red-bean.com/

4. http://www.tldp.org/guides.html

5. http://developer.gnome.org/doc/GGAD

Journals :

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

KARPAGAM ACADEMY OF HIGHER EDUCATION
(Deemed to be University)
(Established Under Section 3 of UGC Act, 1956)
(For the candidates admitted from 2017 onwards)
DEPARTMENT OF CS, CA & IT
LESSON PLAN
SUBJECT NAME : OPEN SOURCE TECHNOLOGIES (17CSP302)
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	R1:33
8	1	Trends and Potential— Indian	R1:37
9	1	Recapitulation and Dicussion 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.

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.

POSSIBLE QUESTIONS

PART – A (20 X 1 = 20 MARKS)

ONLINE 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.

PART - B (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.

PART - C (10 Marks)

1. Write a Linux program to display the allocated memory.
2. Discuss about the CPU scheduling process.

Karpagam Academy of Higher Education

Department of CS, CA & IT

Part –A-Multiple Choice Questions

II M.Sc(CS) (BATCH 2017-2019)

OPEN SOURCE TECHNOLOGIES

Class: II M.Sc. CS **Subject Code:17CSP302**

ONLINE EXAMINATIONS	ONE MARK QUESTIONS				
Questions	**opt1**	**opt2**	**opt3**	**opt4**	**Answer**
Unit I					
What command is used to count the total number of lines, words, and characters contained in a file?	countw	wcount	wc	count p	wcount
What command is used to remove files?	dm	rm	delete	erase	rm
What command is used to remove the directory?	rdir	remove	rd	rmdir	rmdir
What command is used with vi editor to delete a single character?	x	y	a	z	x
What hardware architectures are not supported by Red Hat?	SPARC	IBM-compatible	Alpha	Macintosh	Macintosh
How can you add Amit, a new user, to your system?	Using useradd	Using adduser	Using linuxconf	Using admin	Using adduser
What file specifies the order in which to use specified name services?	/etc/services	/etc/nsorder	/etc/nsswitch.conf	/etc/hosts	/etc/nsswitch.conf

How many primary partitions can exist on one drive?	16	4	2	1	4
In which directory can you store system user default files used for creating user directories?	/usr/tmp	/etc/default	/etc/skel	/etc/users	/etc/skel
What does FSF stand for?	Free Software	File Server First	First Serve First	Free Software	Free Software Foundation
Which of the following is a valid format for mounting a CD-ROM drive?	mount -t iso9660	mount /dev/cdrom	mount /mnt/cdrom	All of the above	All of the above
What command do you use to create Linux file systems?	fdisk	mkfs	fsck	mount	mkfs
Which of the following command can you execute to count the number of lines in a file?	lc	wc - l	cl	count	wc - l
Which of the following is not a communication command?	grep	mail	mesg	write	grep
What command is used to display the characteristics of a process?	au	ps	du	pid	ps
What command is not used to list the files chap01, chap02 and chap04?	ls chap*	ls chap[124]	ls - x chap0[124]	ls chap0[124	ls chap[124]
What command is used with vi editor to replace text from cursor to right	S	s	R	r	R
What sign is used to back up over typing errors in vi?	!	$	#	@	#
What sign is used to erase or kill an entire line you have typed and start you are on a new line (but not display a new prompt)?	!	$	#	@	@
To increase the response time and throughput, the kernel minimizes the frequency of disk access by keeping a pool of	Pooling	Spooling	Buffer cache	Swapping	Buffer cache
At start of process execution, STDOUT & STDERR	Point to current	Are closed	Point to special files	Point to files	Point to current
wtmp and utmp files contain:	Temporary system	User login-logout log	The user's command	The user's su and	User login-logout log
Which is the core of the operating system?	Shell	Kernel	Commands	Script	Kernel

ILP32 stands for	32 bit Integer,	32 bit Integrated	32 bit Intelligent	32 bit Long &	32 bit Integer, Long &
Single Unix Specification Version 2 provides enhanced support for	16 bit Unix	32 bit Unix	64 bit Unix	8 bit Unix	64 bit Unix
Under UNIX the key board is the default input device and the monitor is the default output device	TRUE	FALSE	0	1	TRUE
Which among the following interacts directly with system hardware?	Shell	Commands	Kernel	Applications	Kernel
Applications communicate with kernel by using:	System Calls	C Programs	Shell Script	Shell	System Calls
Solaris is the name of a flavor of UNIX from	HP	IBM	Digital Equipment	Sun Microsyst	Sun Microsystems
Which of the following is “NOT” a UNIX variant ?	Solaris	AIX	IRIX	AS400	AS400
The system calls in UNIX is written using which language	C	C++	Assembly Language	Fortran	C
Which of the following enables multi-tasking in UNIX?	Time Sharing	Multi programming	Multi user	Modularity	Time Sharing
Which of the following is considered as the super daemon in Unix?	sysinit	init	inetd	proc	init
Unix is which kind of Operating System?	Multi User	Multi Processes	Multi Tasking	All the above	All the above
SVR4 stands for?	Standard Version	System Version	Standard Five Release 4	System Five	Standard Version
Lp0 device file is used to access:	Floppy	Cdrom	Printer	Tape drive	Printer
Syntax of any Unix command is:	command [options]	command options	command [options]	command options	command [options]
SVR4 was developed by	Sun Microsyste	AT&T	University of Berkeley	Sun and AT&T	Sun and AT&T jointly
Which of these is not a Unix Flavor?	BSD	MAC	AIX	IRIX	MAC

Which of the following statement is FALSE ?	Unix supports	Linux is an open source	Shell takes care of inter	Shell provides	Shell takes care of inter
Which of the following UNIX flavor is from IBM?	BSD	Solaris	HP-UX	AIX	AIX
x86-32 uses which programming model?	IP16	IP32	ILP16	ILP32	ILP32
What are the sizes of (Integer/Long/Pointer) in LP64 programming model?	8/8/2008	4/4/8	4/8/8	4/8/2004	4/8/8
What control character signals the end of the input file?	ctrl + a	ctrl + b	ctrl + c	ctrl + d	ctrl + d
How do you get help about the command “cp”?	help cp	man cp	cd ?	can cp	man cp
The dmesg command	Shows user login	Shows the syslog file for	kernel log messages	Shows the daemon	kernel log messages
The command “mknod myfifo b 4 16”	Will create a block	Will create a block device	Will create a FIFO if user	Will create a	Will create a block device if
Which command is used to set terminal IO characteristic?	tty	ctty	ptty	stty	stty
Which command is used to record a user login session in a file	macro	read	script	write	script
Which command is used to display the operating system name	os	Unix	kernel	uname	uname
Which command is used to display the unix version	uname -r	uname -n	uname -t	kernel	uname -r
Which command is used to print a file	print	ptr	lpr	none of the above	lpr
Using which command you find resource limits to the session?	rlimit	ulimit	setrlimit	getrlimit	ulimit
Which option of ls command used to view file inode number	–l	-o	–a	–i	–i
find / -name ‘*’ will	List all files and	List a file named * in /	List all files in / directory	List all files and	List all files and

Which command is used to display the octal value of the text	octal	text_oct	oct	od	od
Which command is used to view compressed text file contents	cat	type	zcat	print	zcat
Which command changes a file’s group owner	cgrp	chgrp	change	group	chgrp

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.

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
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

Architecture	Debian Designation	Subarchitecture	Flavor
		MIPS Malta (64 bit)	5kc-malta
IBM/Motorola PowerPC	powerpc	PowerMac	pmac
		PReP	prep
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 muondatasub-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 crondeamon.
/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 nfssystems.

/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 SndSystem: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 &&sudosh 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→ControlPanel→Hardware and Sound→Device Manager.

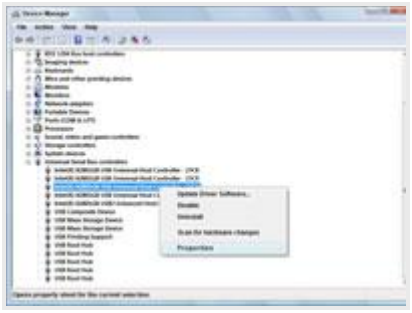
The Device Manager shows all the hardware associated with your computer as well as the health of that hardware.

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:

sudoiwlist scan | more

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

Type the following command:

sudoiwlist 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.

POSSIBLE QUESTIONS

PART – A (20 X 1 = 20 MARKS)

ONLINE 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.

PART - C (10 Marks)

1. Write a Linux program for CPU scheduling of processes.

Karpagam Academy of Higher Education

Department of CS, CA & IT

Part –A-Multiple Choice Questions

II M.Sc(CS) (BATCH 2017-2019)

OPEN SOURCE TECHNOLOGIES

Subject Code:17CSP302

UNIT II

Questions	opt1	opt2	opt3	opt4	Answer
Which command is used to extract intermediate result in a pipeline	tee	extract	exec	tree	tee
Which command is used to extract a column from a text file	paste	get	cut	tar	cut
Which command is used to display disk consumption of a specific directory	du	ds	dd	dds	du
Which command is used to perform backup in unix?	backup	cpio	zip	gzip	cpio
Which command creates an empty file if file does not exist?	cat	touch	ed	read	touch
Which option of rm command is used to remove a directory with all its subdirectories	–b	–o	–p	–r	–r
Which command is used to identify file type?	Type	File	Finfo	Info	File
. Command used to determine the path of an executable file is	which	where	wexec	what	which

Command used to count number of character in a file is	grep	wc	count	cut	wc
Which of these commands could you use to show one page of output at a time?	less	sed	pause	grep	less
Which commands will give you information about how much disk space each file in the current directory uses?	ls -l	ls -la	du	ls -a	du
Which of the following command output contains userid?	ls	help	date	ls –l	ls –l
Which command is used to display all the files including hidden files in your current and its subdirectories ?	ls –aR	ls –a	ls –R	ls –l	ls –aR
Which of the following commands can be used to copy files across systems?	ssh	telnet	rsh	ftp	ftp
pwd command displays	user password	password file content	present working	password	present working directory
Which of the following commands can be used to change default permissions for files and directories at the time of	Chmod	Chown	Umask	Chgrp	Umask
Which tar command option is used to list the files in a tape archive format?	cvf	tvf	xvf	ovf	tvf
Which of the following commands will allow the user to search contents of a file for a particular pattern	touch	grep	find	ls	grep
Write the command to display the current date in the form dd/mm/yyyy	date +%d/%m/	date +"%d/%m/	date +/%d/%m/20	date +"/%d/%m/2	date +%d/%m/%Y
The command syntax to display the file 'sample.txt' one page at a time is	man sample.txt	cat sample.txt	cat sample.txt|mo	cat sample.txt|les	cat sample.txt|more
Which one shows the name of the operating system?	uname -n	uname -r	uname -o	uname –m	uname -o
How do you add (append) a file "file1″ to the example.tar file	no you cannot add	tar -cvf example.ta	tar -rvf file1 example.tar	tar -evf file1 example.tar	tar -rvf file1 example.tar
How to execute ls command inside a vi editor?	!ls	:ls	:!ls	we can't execute	:!ls
Which command gives the first byte where the difference is in the file1 & file2?	diff	cmp	comm	ls -a	cmp

To open a file file1 with cursor at line number 4	vi +num file1	vi +set num file1	vi + “set num” file1	vi +/se nu file1	vi +num file1
sed is a command typically used for	Perform complex	Perform FIFO	Modify/print selective	Modify the file	Modify/print selective contents
What communication command provides communication to another user logged on by writing to the bottom of their	talk	write	chat	transmit	talk
Which screen manipulation command sets the screen back to normal?	tput cup	tput smso	tput rmso	tput blink	tput rmso
Which command will you use to see the available routes?	show route	route status	netstat -r	none of the mentioned	netstat -r
Which of the following time stamps need not exist for a file on traditional unix file system	Access Time	Modification Time	Creation Time	ChangeTime	Creation Time
Which command is used to set limits on file size	fsize	flimit	ulimit	usize	ulimit
Which option of rmdir command will remove all directories a, b, c if path is a/b/c	–b	–o	–p	–t	–p
Which represents the user home directory	/	.	..	~	~
If a file is removed in Unix using ‘rm’ then	The file can be	The file cannot be	The file can be fully	The file will be moved to	The file cannot be recovered by a
Executing the ‘cd ..’ command when at the root level causes	Error message	Behavior is unix-flavor	Results in changing to	Nothing happens	Nothing happens
How do you rename file “new” to file “old”?	mv new old	move new old	cp new old	rn new old	mv new old
What command is used to copy files and directories?	copy	cp	rn	cpy	cp
When mv f1 f2 is executed which file’s inode is freed?	f1	f2	new inode will be used	implementation dependent	f2
Any file’s attribute information is stored in which structure on the disk	Inode	Data blocks	File blocks	Directory file	Inode
By default if any regular file is created, the number of link is displayed as 1 ?	TRUE	FALSE	0	1	TRUE

How many links are created when we creat a directory file?	1	2	3	4	2
A user creates a link to a file file1 using the following command “ln file1 file2”. Which of the following is not	file1 and file2 have	The number of	The number of links for	The number of links for	The number of links for file1 is
There are two hard links to the “file1″ say hl and h2 and a softlink sl. What happens if we deleted the “file1″?	We will still be able	We will not be able	We will be able to access	We will not be able to	We will still be able to access the
If two files on same partition point to the same inode structure they are called	Soft links	Hard links	Alias	Special files	Hard links
Deleting a soft-link	Deletes the destination	Deletes both the	Deletes just the softlink	backup of the destination is	Deletes just the softlink
Creation of hardlinks that point across partitions	is allowed only to	Can be done by all	The effects are	is not allowed	is not allowed
Which command is used to change permissions of files and directories?	mv	chgrp	chmod	set	chmod
Where can I find the printer in the file structure?	/etc	/dev	/lib	/printer	/dev
Which of the following statement is true?	The cp command	The sort command	The mv command will	The command ps	The mv command will preserve the
What UNIX command is used to update the modification time of a file?	time	modify	cat	touch	touch
The soft link will increase the link counter of the file.(T/F)	TRUE	FALSE	0	1	FALSE
When you use the ln command, which of the following occurs?	a file is created	a file is created	a file is moved from one	a file is renamed	a file is created that points to an
srwxr-xrw- is a	internet socket file	unix domain	symbolic link	shared file	unix domain socket file
Binary or executable files are:	Regular files	Device files	Special files	Directory files	Regular files
The directory file contains:	File names & File	File names & Inode	File names & Address	File names & Permissions	File names & Inode Numbers
Which directory contain device special files?	/etc	/etc/dev	/root/bin	dev	dev

Which of the following is not a valid file type on Linux	Socket	Softlink	Inode	FIFO	Inode
Which of the following is not correct statement regarding file types?	Hard links share same	Soft links cannot be	Socket files are Unix	Character file is a special	Soft links cannot be created across
Which are the two types of device files?	Character & Block	Character & Socket	Block & FIFO	Input & output	Character & Block

UNIT III

SYLLABUS

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.

SERVER SETUP AND CONFIGURATION

SETTING UP EMAIL SERVERS USING POSTFIX

Install and Configure a Postfix Mail Server

Installation

1) Open up a terminal window (or, if you are using a GUI-less server just log in).

2) Issue the command *sudo apt-get install postfix*.

The installation will also automatically start the Postfix daemon for you. So as soon as installation is complete you can test to make sure you can connect to your Postfix server with the command:

telnet localhost 25

You should see something like this:

Trying 127.0.0.1...
Connected to www.mymail.com.

Escape character is '^]'.
220 localhost.localdomain ESMTP Postfix (Ubuntu)

telnet www.mymail.com 25

Configuring Postfix

The Postfix mail server has one main configuration file **/etc/postfix/main.cf**. This is where you will do the bulk of your configurations. Open this file up in your favorite text editor (mine is Nano) and look for the following section:

myhostname =
alias_maps = hash:/etc/aliases
alias_database = hash:/etc/aliases
myorigin = /etc/mailname
mydestination =
relayhost =
mynetworks =
mailbox_command = procmail -a "$EXTENSION"
mailbox_size_limit = 0
recipient_delimiter = +
inet_interfaces = all

myhostname: This is the hostname of your machine. But don't put the full hostname. If your machine hostname is *mail.mydomain.com* you will only use *mydomain*.

mydestination: This parameter specifies what destinations this machine will deliver locally. The default is:

mydestination = $myhostnamelocalhost.$mydomain localhost

mydomain.com mydomainlocalhost.localdomain localhost

mynetworks: This line is a bit trickier. This entry will define authorized destinations that mail can be relayed from. You would think that adding your subnet here would work. Sometimes that is the case; sometimes not. You could go with a **mynetworks** entry that looks like:

mynetworks = 127.0.0.1/8

The above entry is a safe entry and defines local machines only.

You could also have an entry that looks like:

mynetworks = 127.0.0.1/8 192.168.100.1/24

The above entry would authorize local machines and your internal network addresses.

Test Your Server

Go to an external source and send an email to one of your users on your new mail server. To find out if it worked you can log on as that user and use the Alpine command line email reader (you might have to install that first with the command *sudo apt-get install alpine*). If you do not see an email show up you will want to check the log file **/var/log/mail.err** which should give you some clues as to what is going wrong.

Setup services using IMAP

Install the packages

```
sudo apt-get install dovecot-imapd dovecot-pop3d
```

Configure the protocol you need to be used by appending the protocol in the file /etc/dovecot/dovecot.conf:

```
protocols = pop3 pop3s imapimaps
```

Choose the mailbox you would like to use. Dovecot supports maildir and mbox formats. Edit the file /etc/dovecot/dovecot.conf and change the line

```
mail_location = maildir:~/Maildir # (for maildir)
mail_location = mbox:~/mail:INBOX=/var/spool/mail/%u # (for mbox)
```

Restart the service

```
sudo /etc/init.d/dovecot restart
```

Use telnet to check that dovecot is working properly.

```
telnet localhost imap
```

Requirements for squirrelmail.

• A web server with PHP is installed. PHP needs to be at least 4.1.0. PHP 4, PHP 5 and PHP 6 are all supported.
• Access to an IMAP server which supports IMAP 4 rev 1.
• PHP gettext extension for better performance.
• The PHP mbstring extension is required for translations that uses multibytes or character sets but ISO-8859-1. Without the PHP mbstring extension the interface will remain usable, but some internationalization features and fixes won't be enabled.
• The PHP XML extension is required if the DIGEST-MD5 authentication is used.
• SquirrelMail is shipped with some Perl scripts. One of the most useful is config/conf.pl, which will help you configure your squirrelmail installation.
First two are the mandatory requirements for squirrelmail. Following PHP settings are recommended for Squirrelmail.
• register_globals off - This is a dangerous setting when enabled, and is not generally needed for most recent PHP applications.
• magic_quotes_{runtime, gpc, sybase} off - Squirrelmail may work with any of these turned on, but if you experience stray backslashes in your mail or other strange behaviour, it may be advisable to turn them off.

• file_uploads on - This is needed if your users want to attach files to their emails.
• safe_mode on or off - Turning safe mode on in squirrelmail's case is not much more secure than having it off. When it is enabled, incompatibilities with some functionality may arise.

Installation of squirrelmail.

Now, we can go through the installation of squirrelmail.

Download the squirrelmail package.

You can get the latest squirrelmail package from http://www.squirrelmail.org/. The latest stable version when I am writing this article is 1.4.22.

Unpack the squirrelmail package.

```
cd /usr/src
tar –jxvf squirrelmail-1.4.22.tar.bz2
mkdir /usr/local/squirrelmail
mv /usr/src/squirrelmail-1.4.22 /usr/local/squirrelmail/www
```

3. Change the ownership of squirrelmail directory.

You need to change the group ownership of the directory to the user/group of Apache web server. You can find it from Apache configuration file 'httpd.conf' (check for 'User' and 'Group' directives).

```
chown –R root.apache /usr/local/squirrelmail
```

4. Create the data and attachment directories.

You need to create data and attachment folders for squirrelmail. The 'data' directory is default location for squirrelmail users' preference files.

```
cd /usr/local/Squirrelmail
mkdir data attachments
chgrp apache data attachments
chmod 0730 data attachments
```

5. Configure squirrelmail.

The configuration perl script is available under the directory 'config'. So, you need to cd to 'config' directory and execute,

./conf.pl

This configuration utility provides menu based configuration options:

6. Configure squirrelmail In Apache.

Now, you need to modify the Apache configuration file, httpd.conf to make squirrelmail available through web browser. Add the following to httpd.conf:

```
Alias /squirrelmail /usr/local/squirrelmail/www

Options None
AllowOverride None
DirectoryIndexindex.php
Order Allow,Deny
Allow from All
```

7. Restart Apache.

Now, you need to restart Apache for the changes to take effect.

```
/usr/local/apache/bin/apachectl restart
```

Now, you will be able to access **"http://domain.com/squirrelmail"**.

Setting up Web server - Apache server

httpd

mod_ssl

elinks

httpd package install Apache web server.

mod_ssl is the additional package which required to create secure websites

elinks is the additional package for text based web browser.

If you have yum repository configured use following command to install Apache web server with additional package

yum install –y httpdmod_ssl

```
[root@Server ~]# yum install -y httpd mod_ssl
Loaded plugins: product-id, refresh-packagekit, subscription
```

yum install elinks

```
[root@Server ~]# yum install elinks
Loaded plugins: product-id, refresh-packagekit, subscription
```

Or you can do it in more simpler way by using groupinsatall. With following command you can install mandatory and all default packages.

yum groupinstall "Web Server"

```
[root@Server ~]# yum groupinstall "Web Server"
Loaded plugins: product-id, refresh-packagekit, subscription
```

If yum repository is not configured use rpm command to install necessary RPM. Mount installation disk of RHEL6 in media folder and move in Packages folder.

```
[root@Server ~]# cd /media/RHEL_6.1\ x86_64\ Disc\ 1/Packages/
[root@Server Packages]# ls httpd*
httpd-2.2.15-9.el6.x86_64.rpm        httpd-manual-2.2.15-9.el6.
httpd-devel-2.2.15-9.el6.i686.rpm    httpd-tools-2.2.15-9.el6.x
httpd-devel-2.2.15-9.el6.x86_64.rpm
[root@Server Packages]# ls mod_ssl*
mod_ssl-2.2.15-9.el6.x86_64.rpm
[root@Server Packages]# _
```

Run following command to install httpd

```
#rpm -ivhhttpd* --nodeps --force
```

```
[root@Server Packages]# rpm -ivh httpd* --nodeps --force
warning: httpd-2.2.15-9.el6.x86_64.rpm: Header V3 RSA/SHA256
d431d51: NOKEY
Preparing...                ########################################
   1:httpd-tools            ########################################
   2:httpd                  ########################################
   3:httpd-devel            ########################################
   4:httpd-devel            ########################################
   5:httpd-manual           ########################################
[root@Server Packages]# _
```

Verify that the packages were installed correctly

```
[root@Server ~]# rpm -qa httpd*
httpd-devel-2.2.15-9.el6.x86_64
httpd-tools-2.2.15-9.el6.x86_64
httpd-2.2.15-9.el6.x86_64
httpd-devel-2.2.15-9.el6.i686
httpd-manual-2.2.15-9.el6.noarch
[root@Server ~]# rpm -qa mod_ssl*
mod_ssl-2.2.15-9.el6.x86_64
[root@Server ~]# _
```

Run following command to start service when the system boots

```
[root@Server ~]# chkconfig httpd on
[root@Server ~]# _
```

SETTING UP WEB SERVERS – USING APACHE

Install Apache

To start off we will install Apache.

1. Open up the Terminal (*Applications* > *Accessories* > *Terminal*).
2. Copy/Paste or type the following line of code into Terminal and then press enter:

sudo apt-get install apache2

3. The Terminal will then ask you for you're password, type it and then press enter.

Testing Apache

To make sure everything installed correctly we will now test Apache to ensure it is working properly.

1. Open up any web browser and then enter the following into the web address:

http://localhost/

You should see a folder entitled *apache2-default/*. Open it and you will see a message saying "It works!" , congrats to you! or something like that!

Install PHP

In this part we will install PHP 5.

Step 1. Again open up the Terminal (*Applications* > *Accessories* > *Terminal*).
Step 2. Copy/Paste or type the following line into Terminal and press enter:

sudo apt-get install php5 libapache2-mod-php5

Step 3. In order for PHP to work and be compatible with Apache we must restart Apache. Type the following code in Terminal to do this:

sudo /etc/init.d/apache2 restart

Test PHP

To ensure there are no issues with PHP let's give it a quick test run.

Step 1. In the terminal copy/paste or type the following line:

```
sudogedit /var/www/testphp.php
```

This will open up a file called *testphp.php*.

Step 2. Copy/Paste this line into the phptest file:

```
<?php phpinfo(); ?>
```

Step 3. Save and close the file.

Step 4. Now open you're web browser and type the following into the web address:

```
http://localhost/testphp.php
```

(It will show you the page that has all information about your php. If you have prior experience of installing php in some other OS, you must have seen this page.)

Congrats you have now installed both Apache and PHP!

Install MySQL

To finish this guide up we will install MySQL.

Step 1. Once again open up the amazing Terminal and then copy/paste or type this line:

```
sudo apt-get install mysql-server
```

Step 2 (optional). In order for other computers on your network to view the server you have created, you must first edit the "Bind Address". Begin by opening up Terminal to edit the *my.cnf* file.

```
gksudogedit /etc/mysql/my.cnf
```

Change the line

```
bind-address = 127.0.0.1
```

And change the *127.0.0.1* to your IP address.
(In Linux Mint 11, terminal itself asked to the set password, But if it doesn't follow the step 3.)

Step 3. This is where things may start to get tricky. Begin by typing the following into Terminal:

```
mysql -u root
```

Following that copy/paste or type this line:

```
mysql> SET PASSWORD FOR 'root'@'localhost' = PASSWORD('yourpassword');
```

(Make sure to change *yourpassword* to a password of your choice.)
Step 4. We are now going to install a program called phpMyAdmin which is an easy tool to edit your databases. Copy/paste or type the following line into Terminal:

sudo apt-get install libapache2-mod-auth-mysql php5-mysql phpmyadmin

After that is installed our next task is to get PHP to work with MySQL. To do this we will need to open a file entitled *php.ini*. To open it type the following:

gksudogedit /etc/php5/apache2/php.ini

Now we are going to have to uncomment the following line by taking out the semicolon (;). Change this line:

;extension=mysql.so

To look like this:

extension=mysql.so

Now just restart Apache and you are all set!

sudo /etc/init.d/apache2 restart

If you get a 404 error upon visiting http://localhost/phpmyadmin: You will need to configure apache2.conf to work with Phpmyadmin.

sudogedit /etc/apache2/apache2.conf

Include the following line at the bottom of the file, save and quit.

Include /etc/phpmyadmin/apache.conf

Then just restart Apache

sudo /etc/init.d/apache2 restart

PERL CONFIGURATION IN LINUX

The Apache HTTP Server Project is an effort to develop and maintain an open-source HTTP server for modern operating systems including UNIX and Windows NT.

Install Apache2 in Ubuntu

sudo aptitude install apache2

This will complete the installation.

After installation Type the server's IP address (or alias if you added the server to your /etc/hosts file) in your browser's address bar or, if you are browsing on the server itself, type 127.0.0.1 or localhost. If an error occurs, then you will have to edit the apache2.conf file to ensure that Apache can fully resolve the server's name.If you have any problem then you have to edit the apache2 configuration file using the following command

sudonano /etc/apache2/apache2.conf

Add the following line somewhere

ServerName localhost

or

ServerNameyourserverip

Save and exit the file

Now you need to restart Apache server using the following command.

sudo apache2ctl restart

Enable PHP support for apache2 webserver

If you want to enable php5 or php4 support to your apache webserver use the following commands to install require packages

For PHP5

sudoaptitiude install php5 libapache2-mod-php5

For PHP4

sudoaptitiude install php4 libapache2-mod-php4

You also make sure the php5 and php4 modules are enabled using the following commands

sudo a2enmod php5

sudo a2enmod php4

After installing php support you need to restart apache webserver using the following command

sudo apache2ctl restart

Test your PHP Support foe apache webserver

To check the status of your PHP installation

sudonano /var/www/testphp.php

and insert the following line

<?php phpinfo(); ?>

Save and exit the file

Now open web browser at http://yourserveripaddress/testphp.php and check.

Enable CGI and perl support for apache2 server

You need to install the following package

sudo aptitude install libapache2-mod-perl2

Configure a cgi-bin directory

You need to create a cgi-bin directory using the following command

sudomkdir /home/www/cgi-bin

Configuring Apache to allow CGI program execution is pretty easy. Create a directory to be used for CGI programs and add the following to the site configuration file (again between the <VirtualHost> tags).

ScriptAlias /cgi-bin/ /home/www/cgi-bin/
<Directory /home/www/cgi-bin/>
Options ExecCGI
AddHandlercgi-script cgi pl
</Directory>

The first line creates an alias that points to the directory in which CGI scripts are stored. The final line tells Apache that only files that end with the *.cgi and *.pl extensions should be considered CGI programs and executed.

Test your Perl Program

cd /home/www/cgi-bin

sudonano perltest.pl

Copy and paste the following section save and exit the file.

```
###Start###
#!/usr/bin/perl -w
print "Content-type: text/html\r\n\r\n";
print "Hello there!<br />\nJust testing .<br />\n";

for ($i=0; $i<10; $i++)
{
print $i."<br />";
}

###End###
```

make sure you change permissions on it

sudochmoda+x perltest.pl

Installing Samba

1. Use yum to install the Samba package:

yum -y install samba

Creating Samba Test Directory and Files

You're going to create a new directory containing three empty files which you'll share using Samba.

While logged on as root, create the new directory /smbdemo with the following command:

mkdir /smbdemo

Change the permissions on the new directory to 770 with the following command:

chmod 770 /smbdemo

Navigate to the new directory with the following command:

cd /smbdemo

Add three empty files to the directory with the following command:

touch file1 file2 file3

```
[root@LinuxServer01 ~]# mkdir /smbdemo
[root@LinuxServer01 ~]# chmod 770 /smbdemo
[root@LinuxServer01 ~]# cd /smbdemo
[root@LinuxServer01 smbdemo]# touch smb1 smb2 smb3
[root@LinuxServer01 smbdemo]# _
```

USING NFS

NFS (Network File System) is basically developed for sharing of files and folders between Linux/Unix systems by Sun Microsystems in 1980. It allows you to mount your local file systems over a network and remote hosts to interact with them as they are mounted locally on the same system. With the help of NFS, we can set up file sharing between Unix to Linux system and Linux to Unix system.

Benefits of NFS

- NFS allows local access to remote files.

- It uses standard client/server architecture for file sharing between all *nix based machines.
- With NFS it is not necessary that both machines run on the same OS.
- With the help of NFS we can configure centralized storage solutions.
- Users get their data irrespective of physical location.
- No manual refresh needed for new files.
- Newer version of NFS also supports acl, pseudo root mounts.
- Can be secured with Firewalls and Kerberos.

NFS Services

Its a System V-launched service. The NFS server package includes three facilities, included in the portmap andnfs-utils packages.

- portmap : It maps calls made from other machines to the correct RPC service (not required with NFSv4).
- nfs: It translates remote file sharing requests into requests on the local file system.
- rpc.mountd: This service is responsible for mounting and unmounting of file systems.

Important Files for NFS Configuration

- /etc/exports :Its a main configuration file of NFS, all exported files and directories are defined in this file at the NFS Server end.
- /etc/fstab : To mount a NFS directory on your system across the reboots, we need to make an entry in/etc/fstab.
- /etc/sysconfig/nfs : Configuration file of NFS to control on which port rpc and other services are listening.

Setup and Configure NFS Mounts on Linux Server

To setup NFS mounts, we'll be needing at least two Linux/Unix machines. Here in this tutorial, I'll be using two servers.

- NFS Server: nfsserver.example.com with IP-192.168.0.100
- NFS Client : nfsclient.example.com with IP-192.168.0.101

Installing NFS Server and NFS Client

We need to install NFS packages on our NFS Server as well as on NFS Client machine. We can install it via "yum" (Red Hat Linux) and "apt-get" (Debian and Ubuntu) package installers.

```
[root@nfsserver~]# yum install nfs-utilsnfs-utils-lib

[root@nfsserver~]# yum install portmap (not required with NFSv4)

[root@nfsserver~]# apt-get install nfs-utilsnfs-utils-lib
```

Now start the services on both machines.

```
[root@nfsserver~]# /etc/init.d/portmap start

[root@nfsserver~]# /etc/init.d/nfs start

[root@nfsserver~]#chkconfig --level 35 portmap on

[root@nfsserver~]#chkconfig --level 35 nfs on
```

Important commands for NFS

Some more important commands for NFS.

- showmount -e : Shows the available shares on your local machine
- showmount -e <server-ip or hostname>: Lists the available shares at the remote server
- showmount -d : Lists all the sub directories

- exportfs -v : Displays a list of shares files and options on a server
- exportfs -a : Exports all shares listed in /etc/exports, or given name
- exportfs -u :Unexports all shares listed in /etc/exports, or given name
- exportfs -r : Refresh the server's list after modifying /etc/exports

SETTING UP PRINTER SERVICES – USING CUPS

The command-line tools of the CUPS printing system and their manual pages are included in cups-client. Further documentation is provided by cups and installed in /usr/share/doc/packages/cups, in particular the *CUPS Software Users Manual*, found at /usr/share/doc/packages/cups/sum.htmland the *CUPS Software Administrators Manual* at /usr/share/doc/packages/cups/sam.html. If a CUPS daemon runs locally on your host, you should also be able to access the documentation at http://localhost:631/documentation.html.

Managing Local Queues

Printing Files

To print a file, enter the *System V style* print command **lp -d *queuename file*** or a *Berkeley style* command like **lpr -P*queuename file***.

Additional information can be obtained with **man lpr** and **man lp** as well as in the section *Using the Printing System* of the *CUPS Software Users Manual*(/usr/share/doc/packages/cups/sum.html#USING_SYSTEM).

The -o parameter allows specification of a number of important options, some of which directly influence the type of printout. More information is available in the manual page of **lpr** and **lp** as well as in the section *Standard Printer Options* of the *CUPS Software Users Manual*(/usr/share/doc/packages/cups/sum.html#STANDARD_OPTIONS).

Checking the Status

To check the status of a queue, enter the *System V style* command **lpstat -o *queuename* -p *queuename*** or the *Berkeley style* command **lpq -P*queuename***. If you do not specify a queue name, the commands display information about all queues.

With **lpstat -o**, the output shows all active print jobs in the form of a *queuename-jobnumber* listing. With **lpstat -l -o *queuename* -p *queuename***, the output is more verbose. **lpstat -t** or **lpstat -l -t** displays the maximum amount of available information.

For additional information, consult the manual page of **lpq**, **lpstat**, and the section *Using the Printing System* of the *CUPS Software Users Manual*(/usr/share/doc/packages/cups/sum.html#USING_SYSTEM).

Removing Jobs from the Queue

Enter the *System V style* command **cancel *queuename-job number*** or the *Berkeley style* command **lprm -P*queuename job number*** to remove the job with the specified number from the specified queue. For additional information, consult the manual page of **lprm**, **cancel**, and the section *Using the Printing System* of the *CUPS Software Users Manual* (/usr/share/doc/packages/cups/sum.html#USING_SYSTEM).

Specifying Options for Queues

To see how to specify hardware-independent options that affect the type of printout, read the section *Standard Printer Options* in the *CUPS Software Users Manual* (/usr/share/doc/packages/cups/sum.html#STANDARD_OPTIONS).

Printer-specific options affecting the type of printout are stored in the PPD file for the queue in question. List them with the command **lpoptions -p *queuename*-l**. The output has the following form:

option/text: value valuevalue ...

The currently active setting is marked with an asterisk (*) to the left, for example:

PageSize/Page Size: A3 *A4 A5 Legal Letter
Resolution/Resolution: 150 *300 600

According to the above output, the PageSize is set to A4 and the Resolution to 300 dpi.

The command **lpoptions -p *queuename* -o option=value** changes the value for the given option. With the above sample settings in mind, use the following command to set the paper size for the specified queue to Letter:

lpoptions -p <queuename> -o PageSize=Letter

The system administrator can change the defaults of a PPD file with a command like:

lpadmin -p <queuename> -o PageSize=Letter

For more information, refer to the Support Database article *Print Settings with CUPS*.

Managing Remote Queues

For each of the commands explained below, replace *printserver* with the name or IP address of your print server. *queuename* must be a queue on the print server.

Printing Files

You can use the *System V style* command **lp -d *queuename* -h *printserver file*** to generate a print job for the specified queue on the specified print server.

Checking the Status

Check the status of a queue on the print server with the *System V style* command **lpstat -h *printserver* -o *queuename* -p *queuename***.

Removing Jobs from the Queue

The *System V style* command **cancel -h *printserver queuename-jobnumber*** removes the print job with the specified job number from the specified queue on the print server.

Using Command-Line Tools for CUPS Troubleshooting

Print jobs will be kept in the printer queue if you shut down the system while a job is being processed. This means a broken print job will still be there even after rebooting and you need to remove it from the queue manually with the commands mentioned above.

Other problems occur if there is some fault in the physical data link between the computer and the printer. The printer may then be unable to make sense of the data it receives and start spitting out lots of pages with garbage on them.

1. To make sure the printer stops working, first remove all paper from it (in the case of inkjet printers) or open the paper trays (laser printers).
2. At this point, the print job will often still be in the queue, because jobs are only removed from the queue when all data has been sent to the device. Check which queue is currently printing by entering **lpstat -o** (or **lpstat -h *printserver* -o**) then remove the problematic print job with **cancel*queuename-jobnumber*** (or with **cancel -h *printserver queuename-jobnumber***).
3. Some data might still find their way to the printer in spite of the job having been deleted. To stop this, enter the command **fuser -k /dev/lp0** (for a printer at the parallel port) or **fuser -k /dev/usb/lp0** (for a USB printer). This kills any processes still using the printer device.
4. Do a complete reset of the printer by disconnecting it from power for some time. Then put in the paper and switch the printer back on.

Foomatic (Printer Database)

Foomatic is a configurable printing filter. It uses PPD files as configuration to generate appropriate output for a given printer. It is spooler independent which means it can be used with Common Unix Printing System (CUPS), LPRng and others.[1] It usesGhostscript in the background, using options according to the PPD file of the printer. Currently it is developed by the OpenPrinting workgroup of the Linux Foundation.is a configurable printing filter. It usesGhostscript in the background, using options according to the PPD file of the printer. Currently it is developed by the OpenPrinting workgroup of the Linux Foundation.

Like CUPS, foomatic supposes that applications will produce output in PostScript. If the output spools to a PostScript printer, no further action is needed. Otherwise, the most generic way to act is:

- Create a raster file from the PostScript (ps2raster, usually using Ghostscript in the background)
- Create a printer-language file from the raster data (raster2xxx, using the raster driver of the target printer)
- Send the printer-language file to the printer

But if foomatic-rip "knows" about the available printer, it will translate the PostScript data directly to the printer's language, without creating the intermediate raster file.

The components of the package are:

foomatic-filters (or "foomatic-rip")

It transforms PostScript data to raster (or to the printer's native language), using the PPD as configuration. It needs a low level driver (specific to each printer) to generate the final code.

foomatic-tools

foomatic-db-engine: A tool that generates PPD files from the data in Foomatic's database. It also contains scripts to directly configure print queues and handle jobs.

foomatic-db: The collected knowledge about printers, drivers, and driver options in XML files, used by foomatic-db-engine to generate PPD files.

foomatic-db-hpijs: Foomatic XML data generators for HP's HPIJS driver.

Free drivers that can interface with foomatic[edit]

The following free drivers were specifically developed to work with foomatic:

- pxlmono and pxlcolor, to work with HP LaserJets
- ljet4, also for LaserJet printers
- hpijs, for PCL inkjet printers
- SpliX, for Samsung Printer Language
- gdi, for Samsung SmartGDI
- ptouch-driver, for Brother P-touch series of label printers

Spoolers that can interact with foomatic[edit]

- CUPS
- LPRng
- LPD
- GNUlpr (see its SourceForge page)
- Solaris LP
- PPR
- CPS
- Direct printing (no spooler)

Setting up a firewall -Using netfilter and iptables

iptables is Linux's firewall which has been a part of the kernel since version 2.4. It is often referred to as a *packet filter* as it examines each packet transferred in every network connection to, from, and within your computer. iptables replaced ipchains in the 2.4 kernel and added many

new features including *connection tracking* (also known as stateful packet filtering). In this article we will use iptables to build simple but effective firewalls for the following scenarios using allow/disallow rules based on IP addresses, ports, and states:

1. a standard home computer;
2. a home/small office network with a single Internet connection;
3. port forwarding for a home/small office network.

Rules, Targets, Chains, Tables, States, and all that jazz

iptables makes decisions on what to do with a packet based on *rules* that the system administrator creates. Data is passed through the Internet in the form of *packets of information*; connecting from your computer to a website will cause many packets to be exchanged in both directions. A rule specifies the criteria necessary for a packet to match it. A decision is known as a *target* and it can be a user-defined chain (not covered in this article) or one of the following:

ACCEPT

Allow the packet through the firewall.

DROP

Drops the packet; the packet is not allowed through the firewall and the sender of the packet is not notified.

There a number of other possible targets and we will cover some of these later.

Rules are grouped into *chains* which in turn are contained in *tables*. There are three default tables which the packets may traverse; we are only concerned with one of these right now:
the filter table. This is the default table and contains three chains:

OUTPUT

For packets generated by and leaving your computer; for example when you connected to the Linux Gazette's web site your browser created a packet and sent it out of your computer to the Gazette's server.

INPUT

Any packets coming into your computer; for example the packets containing the Gazette's web page sent back by its server to your browser.

FORWARD

For packets being routed through your computer; for exampleentering one network card and leaving through the other.

The two other tables available by default are the nat table and the mangle table

NEW

The packet is trying to start a new connection; for example when you first connected to the Linux Gazette website your browserattempted to create a new connection with the Gazette's web server.

ESTABLISHED

A connection that has seen packets travel in both directions; once the Gazette's web server replied to your browser the connection isestablished.

RELATED

A packet that is starting a new connection but is related to anexisting connection. An example of this is downloading a file overFTP. When you first connect to an FTP server you are creating a newconnection to its FTP port. However, when you download a file fromthe FTPserver using this connection a second new connection ismade between your computer and the FTP server for the fileownload. Although it is a new connection it is related to thefirst. This stateful packet filtering is useful as this newconnection does not use the FTP port and simple port based rulesare not appropriate for this.

INVALID

This packet is associated with no known connection. These packetsshould be dropped.

Creating and Storing Rules

Rules can be appended to the chains directly by using the iptables command. For example, to add a new rule to allow new connections to a web server running on your computer from anywhere we would execute the following:

$ iptables -A INPUT -s 0/0 -d 1.2.3.4 -m state --state NEW -p tcp --dport 80 -i eth0 -j ACCEPT
where:
-s (or --src or --source) and -d (or --dst or --destination)

is the source and destination specification of the packet. It isusually an IP address with an optional mask. 0/0 is shorthand for 0.0.0.0/0.0.0.0 meaning that the source can be **any** IP address.

POSSIBLE QUESTIONS

PART – A (20 X 1 = 20 MARKS)

ONLINE QUESTIONS

(2 Marks)

1. Define NFS
2. What is the purpose of SMTP server
3. Define Samba
4. Write a note on SQUID
5. Define IMAP
6. What is the need of IPTables
7. What is the purpose of netfilter.

PART - B (6 Marks)

1. Discuss about setting up and Configuring of E-Mail server.
2. Write about setting of file services using SAMBA in Linux.

3. Describe configuration of SMTP server.
4. Write about setting of Proxy services using SQUID.
5. Describe configuration of IMAP and POP3 server in Linux.
6. Write about using NFS file services for gnu/Linux / Unix networks.
7. Explain configuring and setting up of any two web servers in Linux.
8. Write about setting up of printer services using print spooler.
9. Describe configuring of apache and PHP web server in Linux.
10. Explain setting up a firewall using net filter and iptables.

PART - C (10 Marks)

1. Write a Linux program to display date & time using TCP sockets.

Karpagam Academy of Higher Education

Department of Computer Science

Part –A-Multiple Choice Questions

II M.Sc(CS) (BATCH 2017-2019)

OPEN SOURCE TECHNOLOGIES

Class: II M.Sc. CS **Subject Code:17CSP302**

ONLINE EXAMINATIONS **ONE MARK QUESTIONS**

UNIT III

Questions	opt1	opt2	opt3	opt4	Answer
All device files are stored in which directory?	/etc	/bin	/dev	/usr	/dev
The file permission 764 means:	Every one can read, group can	Every one can read and	Every one can read,	Every one can read	Every one can read,
The permission -rwxr–r– represented in octal expression will be	777	666	744	711	744
Effective user id can be set using following permission	777	2666	4744	1711	4744
Effective group id can be set using following permission	777	2666	4744	1711	2666
Sticky bit can be set using following permission	777	2666	4744	1711	1711
The permission -rwSr–r– represented in octal expression will be	777	2666	4744	4644	4644

The permission -rwxr-sr– represented in octal expression will be	777	2766	2744	2754	2754
If user tries to remove (rm) a readonly file (444 permission), what will happen?	The file is removed	The rm command	The rm command	The rm command	The rm command
A user does a chmod operation on a file. Which of the following is true?	The last accessed time of the file is	The last modification	The last change time	None of the above	The last change time
If the umask value is 0002. what will be the permissions of new directory	777	775	774	664	775
What is the command to set the execute permissions to all the files and subdirectories within the directory	chmod –r +x /home/user1/direc	chmod –R +x	chmod –f –r +x	chmod –F +x	chmod –R +x
The permission -rwxr-xr-t represented in octal expression will be	777	1755	1754	2754	1755
With a umask value of 112, what is the default permission assigned to newly created regular file?	—x–x-wx	-rw-rw-r–	-r-xr-x-r–	-rw-rw-r–	-rw-rw-r–
Which command is used to assign read-write permission to the owner?	chmod a+r file	chmod o+r file	chmod u=rw file	chmod og-r file	chmod u=rw file
Given the command $ chmod o-w datafile	sets write permission to	sets write permission	clears write permission	clears write	clears write permission
Which of these commands will set the permissions on file textfile to read and write for the owner, read for the	chmod 046 textfile	chmod 640 textfile	chmod 310 textfile	chmod rw r nil textfile	chmod 640 textfile
If you are a root user, how can you grand execute permission only for the owner of the file project1?	chmod +x project1	chmod u+x project1	chmod a+x project1	chmod U+X	chmod u+x project1
A user executes the following command successfully: $ chmod +x file1.txt	The command results in adding	The command	The command	The command	The command
What does chmod +t do?	wrong syntax	set effective userid for	set effective groupid for	set the sticky bit	set the sticky bit
Which of the following umask settings doesn't allow execute permission to be set by default on directory files	222	111	0	444	0
Which of the following umask settings allow execute permission to be set by default on regular files	222	111	0	None of the given	None of the given choices
The command chmod 4777 a.out	will set the suid bit of a.out	will set the suid bit of	is not a valid	will set the sticky bit	will set the suid bit of

What protocol(s) is(are) allowed a user to retrieve her/his mail from the mail server to her/his mail reader?	POP3	FTP	MAP	All of the above	POP3
What project is currently developing X server support?	XFree86 Project, Inc.	RHAD Labs	GNOME Project	All of the above	XFree86 Project, Inc.
What X-based tool is available for configuring the X Window system?	XConfigurator	XF86Setup	xf86config	xf76config	XF86Setup
Which of the following server is used with the BIND package?	httpd	shttp	dns	named	named
What port does squid listen, by default?	4322	2314	7334	3128	3128
Which of the following is the main Apache configuration file?	/etc/apache.conf	/etc/httpd/config.ini	/etc/httpd/conf/httpd.con	/etc/srm.conf	/etc/httpd/conf/httpd.conf
Which of the following command is used to access an SMB share on a Linux system?	NFS	SMD	smbclient	smbserver	smbclient
Which of the following command is used to see the services running in NFS server?	rpcinfo	serverinfo	NFSinfo	infserv	rpcinfo
In which tcp_wrappers file can you specify to allow all connections from all hosts?	/etc/hosts.allow	/etc/hosts.deny	/etc/hosts	/etc/tcp.conf	/etc/hosts.allow
What does GNU stand for?	GNU's not Unix	Greek Needed Unix	General Unix	General Noble Unix	GNU's not Unix
Simple mail transfer protocol (SMTP) utilizes ____ as the transport layer protocol for electronic mail transfer.	TCP	UDP	DCCP	SCTP	TCP
SMTP connections secured by SSL are known as	SMTPS	SSMTP	SNMP	FTP	SMTPS
SMTP uses the TCP port	22	23	24	25	25
Which one of the following protocol is used to receive mail messages?	SMTP	FTP	TCP	HTTP	SMTP
What is on-demand mail relay (ODMR)?	protocol for SMTP security	an SMTP extension	protocol for web pages	an FTP extension	an SMTP extension
An email client needs to know the ____ of its initial SMTP server.	IP address	MAC address	url	port address	IP address

A SMTP session may include	zero SMTP transaction	one SMTP transaction	more than one SMTP	all of the mentioned	all of the mentioned
SMTP defines	message transport	message encryption	message content	message address	message transport
Which one of the following is an SMTP server configured in such a way that anyone on the internet	open mail relay	wide mail reception	open mail reception	open relay	open mail relay
SMTP is used to deliver messages to	user's terminal	user's mailbox	user kernel	user file	user's mailbox
What is the initial stage of Linux boot process?	System Startup	Stage 1 boot loader	Kernel	init	System Startup
Which of the following stages included GRUB boot loader?	Stage 1 boot loader	Stage 2 boot loader	Kernel	init	Stage 2 boot loader
The job of the _____ is to find and load the secondary boot loader.	Primary boot loader	Secondary boot loader	Grub boot loader		Primary boot loader
In stage 1 boot loader, the first _____ bytes are the primary boot loader.	450	449	448	446	446
What is the final stage of Linux boot process?	Kernel	Stage 2 boot loader	Stage 1 boot loader	init	init
If you use the default partitioning scheme then which partitions gets created?	boot, /, home	boot, /, home, var	boot, /, home, swap	boot, /, home,	boot, /, home, swap
What is the mount point of SWAP?	SWAP	ext3	NTFS	Not Acceptable	Not Acceptable
What is the minimum length of root password required?	10 Char	8 Char	6 Char	12 Char	6 Char
The tool allows to configure the printer is-	Print Configuration	CUPS	PTR Configuratio	Printer Configurati	Print Configuration
_____ is the initial stage of Linux boot process.	System Startup	Stage1 Boot Loader	Kernel	init	System Startup
The _____ serves as a validation check of the MBR.	System Number	System Counter	Magic Number	Magic Counter	Magic Number
The call to _____, a long list of initialization functions are called to set up interrupts, perform memory	Stop_Kernel	Start_Kernel	Suspend_Kernel	Resume_Kernel	Start_Kernel

The su command is used to	Exit from user session	Switching the user	Both a and b	None of the above	Switching the user
The exit command is used to	Exit from user session	Switch	Both a and b	None of these	Exit from user session

UNIT IV
SYLLABUS

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 Libc 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 Practicies and Guidelines for GNU/Linux and FOSS ; Introduction to Bash , sed & awk scripting.

PROGRAMMING TOOLS

USING GNU COMPILER COLLECTION – GNU COMPILER TOOLS

C preprocessor (cpp)

The C preprocessor is a *macro processor* that is used automatically by the C compiler to transform your program before actual compilation. It is called a macro processor because it allows you to define *macros*, which are brief abbreviations for longer constructs.

The C preprocessor provides four separate facilities that you can use as you see fit:

- Inclusion of header files. These are files of declarations that can be substituted into your program.

- Macro expansion. You can define *macros*, which are abbreviations for arbitrary fragments of C code, and then the C preprocessor will replace the macros with their definitions throughout the program.
- Conditional compilation. Using special preprocessing directives, you can include or exclude parts of the program according to various conditions.
- Line control. If you use a program to combine or rearrange source files into an intermediate file which is then compiled, you can use line control to inform the compiler of where each source line originally came from.

Transformations Made Globally

Most C preprocessor features are inactive unless you give specific directives to request their use. (Preprocessing directives are lines starting with `#'; All C comments are replaced with single spaces.

- Backslash-Newline sequences are deleted, no matter where. This feature allows you to break long lines for cosmetic purposes without changing their meaning.
- Predefined macro names are replaced with their expansions

Preprocessing Directives

Most preprocessor features are active only if you use preprocessing directives to request their use.Preprocessing directives are lines in your program that start with `#'. The `#' is followed by an identifier that is the *directive name*. For example, `#define' is the directive that defines a macro. Whitespace is also allowed before and after the `#'.

Header Files

A header file is a file containing C declarations and macro to be shared between several source files. You request the use of a header file in your program with the C preprocessing directive `#include'.

Uses of Header Files

Header files serve two kinds of purposes.

- System header files declare the interfaces to parts of the operating system. You include them in your program to supply the definitions and declarations you need to invoke system calls and libraries.
- Your own header files contain declarations for interfaces between the source files of your program. Each time you have a group of related declarations and macro definitions all or most of which are needed in several different source files, it is a good idea to create a header file for them.

The `#include' Directive

Both user and system header files are included using the preprocessing directive `#include'. It has three variants:

#include <*file*>

This variant is used for system header files. It searches for a file named *file* in a list of directories specified by you, then in a standard list of system directories.

#include "*file*"

This variant is used for header files of your own program. It searches for a file named *file* first in the current directory, then in the same directories used for system header files.

#include *anything else*

This variant is called a *computed #include*. Any `#include' directive whose argument does not fit the above two forms is a computed include.

Macros

A macro is a sort of abbreviation which you can define once and then use later. There are many complicated features associated with macros in the C preprocessor.

Simple Macros

A *simple macro* is a kind of abbreviation. It is a name which stands for a fragment of code. Some people refer to these as *manifest constants*.

Before you can use a macro, you must *define* it explicitly with the `#define' directive. `#define' is followed by the name of the macro and then the code it should be an abbreviation for. For example,

```
#define BUFFER_SIZE 1020
```

defines a macro named `BUFFER_SIZE' as an abbreviation for the text `1020'. If somewhere after this `#define' directive there comes a C statement of the form

Predefined Macros

Several simple macros are predefined. You can use them without giving definitions for them. They fall into two classes: standard macros and system-specific macros.

C compiler (gcc)

1. GNU C and C++ compiler collection
2. Development tools
3. Development libraries
4. IDE or text editor to write programs

Step #1: Install C/C++ compiler and related tools

```
$ sudo apt-get install build-essential manpages-dev
```

Step #2: Verify installation

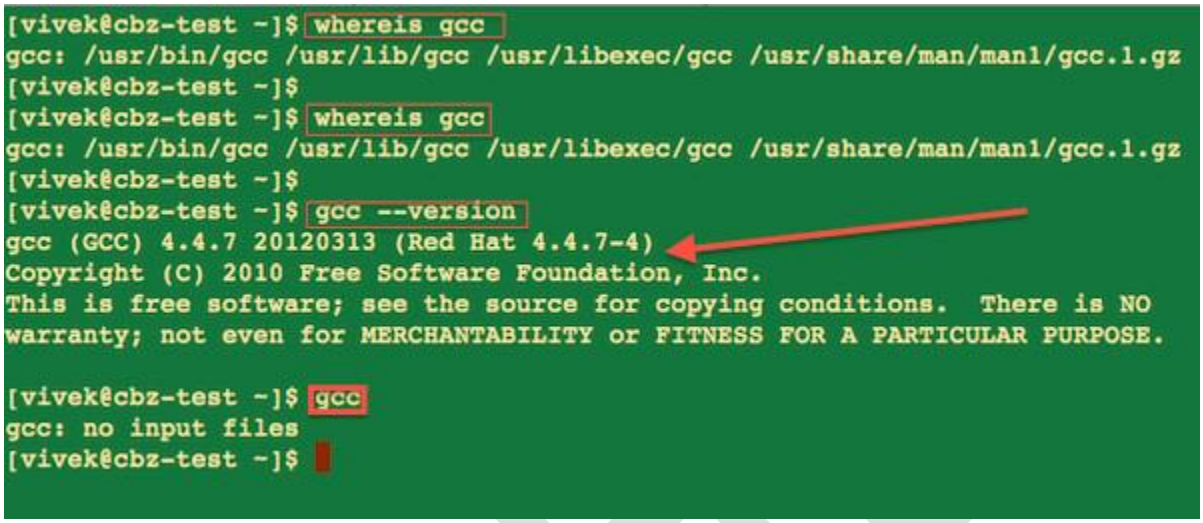
Type the following command to display the version number and location of the compiler on Linux:

```
$ whereis gcc
```

$ which gcc

$ gcc --version

Sample outputs:



How to Compile and Run C/C++ program on Linux

Create a file called demo.c using a text editor such as vi, emacs or joe:

```
#include<stdio.h>
/* demo.c:  My first C program on a Linux */
int main(void)
{
 printf("Hello! This is a test prgoram.\n");
 return 0;
}
```

```
## assuming that executable-file-name.c exists ##
make executable-file-name
```

In this example, compile demo.c, enter:

```
cc    demo.c   -o demo
```

OR

```
## assuming demo.c exists in the current directory ##
make demo
```

If there is no error in your code or C program then the compiler will successfully create an executable file called demo in the current directory, otherwise you need fix the code. To verify this, type:

```
$ ls -l demo*
```

Compiling and running a simple C++ program

Create a program called demo2.C as follows:

```
#include "iostream"

// demo2.C - Sample C++ prgoram

int main(void)

{

   std::cout << "Hello! This is a C++ program.\n";

   return 0;

}
```

To compile this program, enter:

```
g++ demo2.C -o demo2
## or use the following syntax ##
make demo2
```

To run this program, type:

```
./demo2
```

Installing C++ Compiler (g++)

C++ is one of the most popular programming language, mostly used in system programming (system software development e.g Most of the modern operating systems are programmed (primarily) either in C language or C++). C is a High Level language as compared to Assembly program, so it comes in the middle of Assembly and Java like High Level Language

Compiling and Running C++ program on Ubuntu 11.04

For C++ programs you can use G++ compiler an advanced compiler with lot of features.Toinstall G++ type following command at command line.

> sudo apt-get install g++

STEP#1 : Write a program, better use gedit (The Default Text editor). Right click on Desktop and create a new file – 'hello.cpp' and copy the code, given below and save it (CTRL+S). (C++ programs, have .cpp, although it's a different fact that Linux doesn't recognize file based on their extension but some applications may do that and it also avoid confusion, so better use proper extension)

Example (hello.world in c++)

```
// my first program in C++
#include <iostream>
using namespace std;
int main ()
{
```

cout << "Hello World!";

return 0;

}

STEP#2 : Now you compile the program using G++ compiler .Open the Terminal (CTRL+ALT+T) and type the command (First move on to the directory where your file is located, I assume you have created the file on Desktop.

> cd Desktop

> g++ hello.cpp -o helloworld

The -o option (specifies the Output File Name) in the following command is optional, but it's a good practice, because if you won't specify that – then a default a.out file will be created (which will eventually overwrite older a.out file in that directory).

STEP #3 : Execute/Run the program. Type –

> ./helloworld

GAS, the GNU Assembler

GAS, the GNU Assembler, is the default assembler for the GNU Operating System. It works on many different architectures and supports several assembly language syntaxes. These examples are only for operating systems using the Linux kernel and an x86-64 processor, however.

Here is the traditional Hello World program that uses Linux System calls, for a 64-bit installation:

hello.s

```
# ----------------------------------------------------------------------------
# Writes "Hello, World" to the console using only system calls. Runs on 64-bit Linux only.
# To assemble and run:
#
```

```
#     gcc -c hello.s && ld hello.o && ./a.out
#
# or
#
#     gcc -nostdlib hello.s && ./a.out
# ----------------------------------------------------------------------------

.global _start


        .text
_start:
        # write(1, message, 13)
        mov     $1, %rax                # system call 1 is write
        mov     $1, %rdi                # file handle 1 is stdout
        mov     $message, %rsi          # address of string to output
        mov     $13, %rdx               # number of bytes
        syscall                 # invoke operating system to do the write

        # exit(0)
        mov     $60, %rax               # system call 60 is exit
        xor     %rdi, %rdi              # we want return code 0
        syscall                 # invoke operating system to exit
message:
        .ascii  "Hello, world\n"
```

Constructing make files and using make using autoconf and autogen to automatically generate make files tailored for different development environments ;

Autoconf and Automake provide an effective build system to maintain your software, usually on someone else's system. Automake examines source files, determines how they

depend on each other, and generates a Makefile so the files can be compiled in the correct order. Autoconf permits automatic configuration of software installation, handling a large number of system quirks to increase portability. Libtool (not discussed here) is a command-line interface to the compiler and linker that makes it easy to generate static and shared libraries.

The Essential Files

The smallest project requires you provide only two files:

- Makefile.am - an input file to automake that specifies a projects build requirements: what needs to be built, and where it goes when installed.
- configure.in - an input file to autoconf that provides the macro invocations and shell code fragments autoconf uses to build a *configure* script.

The GNU Autotools will generate the rest of the files needed to build the project.

The Directory Structure

Before writing any code for a new project you need to decide on the directory structure the project will use.

- The *top-level* directory is used for configuration files, such as configure.in, and other sundry files like ChangeLog, COPY (a copy of the project license) and README.
- Any unique library should have its own subdirectory containing all headers and sources, a Makefile.am, and any other library specific files.
- The headers and sources for the main application should be in another subdirectory, typically called *src*.

- Other directories can include: *config* for intermediate files, *doc* for the project documentation and *test* for the project self-test suite.

The following steps will take you through creating and building the HelloWorld project. The top-level directory for HelloWorld is <tests/project>. You will find the project's headers and sources in the *src* subdirectory. There are three files: helloworld.cc, helloworld.h and main.cc.

Makfile.am

You must provide a Makefile.am file for each directory in your source tree. Makefile.am for the top-level directory is simple. Create a new text file called Makefile.am in the <tests/project> directory. Add the following line to the file and save it:

```
SUBDIRS = src
```

Using cvs to manage source code revisions patch & diff

CVS is primarily used as a source code control system for text files. Programmers will generate revisions to individual source code files. A collection of these files may define a specific software release. CVS aims to manage the collection of these files and the respective revisions of the individual files that make up the collection. CVS is a command driven file checkout, update, compare and management system. Front end web and desktop GUI systems are available to ease the use of CVS.

Setting up your environment for CVS:

- Set environment variables: (add to your .bashrc file)

  Environment variables:

```
export CVSROOT='/home/Project/CVS_root'      - directory for CVS source code repository

export CVSEDITOR=/bin/vi
```

- Set environment variables: (add to your .cshrc file) (for csh users)

  Environment variables:

```
setenv CVSROOT  '/home/Project/CVS_root'
setenv CVSEDITOR /bin/vi
```

- **CVSROOT**: Location of CV source code repository.

Example use of CVS:

CVS commands are used with directives and command line options to create a repository, check-out, check-in and update code and interrogate changes between versions. Typically one will use a CVS repository which has already been generated, if not, one must be generated and populated with source code text files.

Creating CVS repository for the first time:

Create the CVS "root" of a new CM repository:

```
cvs -d /home/Project/CVS_root init
```

Importing a new project:

To put your project under CVS control:

Check in all files and directories from within the current working directory. The directory referenced is the tree structure for CVS not your current path.

```
cvs import -m "Put text description here" ProgABC CorpABC start
```

Linking against object archives (.a libraries) and dynamic shared object libraries (.so libraries) generating statically linked binaries and libraries generating dynamically linked libraries.

The C standard libraries and C++ STL are examples of shared components which can be linked with your code. The benefit is that each and every object file need not be stated

when linking because the developer can reference the individual library. This simplifies the multiple use and sharing of software components between applications. It also allows application vendors a way to simply release an API to interface with an application. Components which are large can be created for dynamic use, thus the library remain separate from the executable reducing it's size and thus disk space used. The library components are then called by various applications for use when needed.

Linux Library Types:

There are two Linux C/C++ library types which can be created:

1. Static libraries (.a): Library of object code which is linked with, and becomes part of the application.
2. Dynamically linked shared object libraries (.so): There is only one form of this library but it can be used in two ways.
    1. Dynamically linked at run time but statically aware. The libraries must be available during compile/link phase. The shared objects are not included into the executable component but are tied to the execution.
    2. Dynamically loaded/unloaded and linked during execution (i.e. browser plug-in) using the dynamic linking loader system functions.

Library naming conventions:

Libraries are typically names with the prefix "lib". This is true for all the C standard libraries. When linking, the command line reference to the library will not contain the library prefix or suffix.

Thus the following link command: gcc src-file.c -lm -lpthread

The libraries referenced in this example for inclusion during linking are the math library and the thread library. They are found in /usr/lib/lib**m**.a and /usr/lib/lib**pthread**.a.

Note: The GNU compiler now has the command line option "-pthread" while older versions of the compiler specify the pthread library explicitly with "-lpthread". Thus now you are more likely to see gcc src-file.c -lm -pthread

Static Libraries: (.a)

How to generate a library (object code archive file):

- Compile: cc -Wall -c ctest1.c ctest2.c

  Compiler options:
  - -Wall: include warnings. See man page for warnings specified.
- Create library "libctest.a": ar -cvq libctest.a ctest1.o ctest2.o
- List files in library: ar -t libctest.a
- Linking with the library:
  - cc -o *executable-name* prog.c libctest.a
  - cc -o *executable-name* prog.c -L/path/to/library-directory -lctest
- Example files:
  - ctest1.c

```
void ctest1(int *i)
{
  *i=5;
}
```

  - ctest2.c

```
void ctest2(int *i)
{
  *i=100;
}
```

  prog.c

```
#include <stdio.h>
void ctest1(int *);
```

```
void ctest2(int *);
int main()
{
  int x;
  ctest1(&x);
  printf("Valx=%d\n",x);
  return 0;
}
```

Historical note: After creating the library it was once necessary to run the command: ranlib ctest.a. This created a symbol table within the archive. Ranlib is now embedded into the "ar" command.

Note for MS/Windows developers: The Linux/Unix ".a" library is conceptually the same as the Visual C++ static ".lib" libraries.

Dynamically Linked "Shared Object" Libraries: (.so)

How to generate a shared object: (Dynamically linked object library file.) Note that this is a two step process.

1. Create object code
2. Create library
3. Optional: create default version using a symbolic link.

Library creation example:

```
gcc -Wall -fPIC -c *.c
gcc -shared -Wl,-soname,libctest.so.1 -o libctest.so.1.0   *.o
mv libctest.so.1.0 /opt/lib
```

```
ln -sf /opt/lib/libctest.so.1.0 /opt/lib/libctest.so.1

ln -sf /opt/lib/libctest.so.1.0 /opt/lib/libctest.so

This creates the library libctest.so.1.0 and symbolic links to it.
```

It is also valid to cascade the linkage:

```
ln -sf /opt/lib/libctest.so.1.0 /opt/lib/libctest.so.1

ln -sf /opt/lib/libctest.so.1   /opt/lib/libctest.so

Compiler options:
```

- -Wall: include warnings. See man page for warnings specified.
- -fPIC: Compiler directive to output position independent code, a characteristic required by shared libraries. Also see "-fpic".
- -shared: Produce a shared object which can then be linked with other objects to form an executable.
- -Wl,*options*: Pass options to linker.
  In this example the options to be passed on to the linker are: "-soname libctest.so.1". The name passed with the "-o" option is passed to gcc.
- Option -o: Output of operation. In this case the name of the shared object to be output will be "libctest.so.1.0"

Library Links:

- The link to /opt/lib/libctest.so allows the naming convention for the compile flag -lctestto work.
- The link to /opt/lib/libctest.so.1 allows the run time binding to work. See dependency below.

Compile main program and link with shared object library:

Compiling for runtime linking with a dynamically linked libctest.so.1.0:

gcc -Wall -I*/path/to/include-files* -L*/path/to/libraries* prog.c -lctest -o prog

Use:

gcc -Wall -L/opt/lib prog.c -lctest -o prog

Where the name of the library is libctest.so. (This is why you must create the symbolic links or you will get the error "/usr/bin/ld: cannot find -lctest".) The libraries will NOT be included in the executable but will be dynamically linked during runtime execution.

gdb to debug programs

Using GNU Debugging Tools - GDB to debugger Programs

Firstly, in order to successfully use debuggers like GDB, you have to compile your program in such a way that the compiler also produces debugging information that's required by debuggers. For example, in case of the gcc compiler, which we'll be using to compile the example C program later on this tutorial, you need to use the -g command line option while compiling your code.

.

Next step is to make sure that you have GDB installed on your system. If that's not the case, and you're on a Debian-based system like Ubuntu, you can easily install the tool using the following command:

```
sudo apt-get install gdb
```

For installation on any other distro, head here.

Now, once you've compiled your program in a way that it's debugging-ready, and GDB is there on your system, you can execute your program in debugging mode using the following command:

```
gdb [prog-executable-name]
```

While this will initiate the GDB debugger, your program executable won't be launched at this point. This is the time where you can define your debugging-related settings. For example, you can define a breakpoint that tells GDB to pause the program execution at a particular line number or function.

Moving on, to actually launch your program, you'll have to execute the following gdb command:

```
run
```

It's worth mentioning here that if your program requires some command line arguments to be passed to it, you can specify them here. For example:

```
run [arguments]
```

GDB provides many useful commands that come in handy while debugging. We'll discuss some of them in the example in next section.

GDB usage example

Now we have a basic idea about GDB as well as its usage. So let's take an example and apply the knowledge there. Here's an example code:

```
#include <stdio.h>

int main()
{
   int out = 0, tot = 0, cnt = 0;
   int val[] = {5, 54, 76, 91, 35, 27, 45, 15, 99, 0};
```

```
    while(cnt < 10)
    {
        out = val[cnt];
        tot = tot + 0xffffffff/out;
        cnt++;
    }

    printf("\n Total = [%d]\n", tot);
    return 0;
}
```

So basically, what this code does is, it picks each value contained in the 'val' array, assigns it to the 'out' integer, and then calculates 'tot' by summing up the variable's previous value and the result of '0xffffffff/out.'

The problem here is that when the code is run, it produces the following error:

```
$ ./gdb-test
Floating point exception (core dumped)
```

So, to debug the code, the first step would be to compile the program with -g. Here's the command:

```
gcc -g -Wall gdb-test.c -o gdb-test
```

Next up, let's run GDB and let it know which executable we want to debug. Here's the command for that:

```
gdb ./gdb-test
```

Graphical Debuggers like ddd

DDD can do four main kinds of things (plus other things in support of these) to help you catch bugs in the act:

- Start your program, specifying anything that might affect its behavior.
- Make your program stop on specified conditions.
- Examine what has happened, when your program has stopped.
- Change things in your program, so you can experiment with correcting the effects of one bug and go on to learn about another.

Technically speaking, DDD is a front-end to a command-line debugger (called *inferior debugger*, because it lies at the layer beneath DDD). DDD supports the following inferior debuggers:

- To debug *executable binaries*, you can use DDD with GDB, DBX, *Ladebug*, or XDB.
    - GDB, the GNU debugger, is the recommended inferior debugger for DDD. GDB supports native executables binaries originally written in C, C++, Java, Modula-2, Modula-3, Pascal, Chill, Ada, and FORTRAN.
    - As an alternative to GDB, you can use DDD with the DBX debugger, as found on several UNIX systems. Most DBX incarnations offer fewer features than GDB, and some of the more advanced DBXfeatures may not be supported by DDD. However, using DBX may be useful if GDB does not understand or fully support the debugging information as generated by your compiler.
    - As an alternative to GDB and DBX, you can use DDD with *Ladebug*, as found on Compaq and DEC systems. Ladebug offers fewer features than GDB, and some of the more advanced Ladebug features may not be supported by DDD. However, using Ladebug may be useful if GDB or DBX do not understand or fully support the debugging information as generated by your compiler.[1]

- As another alternative to GDB, you can use DDD with the XDB debugger, as found on HP-UX systems.[2].

The Data Display Debugger (DDD) is a popular graphical user interface to UNIX debuggers such as GDB, DBX, XDB, JDB and others. Besides typical front-end features such as viewing source texts and breakpoints, DDD provides an interactive graphical data display, where data structures are displayed as graphs. Using DDD, you can reason about your application by watching its data, not just by viewing it execute lines of source code.

Other DDD features include: debugging of programs written in Ada, Bash, C, C++, Chill, Fortran, Java, Modula, Pascal, Perl and Python; machine-level debugging; hypertext source navigation and lookup; breakpoint, backtrace, and history editors; preferences and settings editors; program execution in terminal emulator window; debugging on remote host; on-line manual; interactive help on the Motif user interface; GDB/DBX/XDB

command-line interface with full editing, history, and completion capabilities.

Profiling Libraries mpatrol and Valgrind

Valgrind is a memory mismanagement detector. It shows you memory leaks, deallocation errors, etc. Actually, Valgrind is a wrapper around a collection of tools that do many other things (e.g., cache profiling); however, here we focus on the default tool, memcheck. Memcheck can detect:

- Use of uninitialised memory
- Reading/writing memory after it has been free'd
- Reading/writing off the end of malloc'd blocks
- Reading/writing inappropriate areas on the stack
- Memory leaks -- where pointers to malloc'd blocks are lost forever
- Mismatched use of malloc/new/new [] vs free/delete/delete []
- Overlapping src and dst pointers in memcpy() and related functions
- Some misuses of the POSIX pthreads API

To use this on our example program, test.c, try

gcc -o test -g test.c

This creates an executable named test. To check for memory leaks during the execution of test, try

valgrind --tool=memcheck --leak-check=yes --show-reachable=yes --num-callers=20 --track-fds=yes ./test

This outputs a report to the terminal like

==9704== Memcheck, a memory error detector for x86-linux.

Mpatrol

After you link the program from Example 7-6 with the mpatrol debug library, you should set theMPATROL_OPTIONS environment variable to the desired debugging action:

$ gcc leak.c o leak g lmpatrol lbfd

$ export MPATROL_OPTIONS="LOGALL USEDEBUG"

$ ./leak

$ cat mpatrol.log . . . ALLOC: malloc (127, 4 bytes, 4 bytes) [main|/tmp/leak.c|15] 0x0804854B main+31 at /tmp/leak.c:15 0x40102500 __libc_start_main+224 0x08048491 _start+33 at ../sysdeps/i386/elf/start.S:102 ...

In the preceding example, the options LOGALL and USEDEBUG were used to log memory allocations and display line number information where memory was allocated. Note that the test program was also linked with libbfd.so, because mpatrol requires symbols from libbfd.

Review of Common Programming Practices and Guidelines for GNU/Linux and FOSS

Linux kernel coding style

1) Indentation

Tabs are 8 characters, and thus indentations are also 8 characters. There are heretic movements that try to make indentations 4 (or even 2!) characters deep, and that is akin

In short, 8-char indents make things easier to read, and have the added benefit of warning you when you're nesting your functions too deep. Heed that warning.

The preferred way to ease multiple indentation levels in a switch statement is to align the switch and its subordinate caselabels in the same column instead of double-indenting the case labels. E.g.:

```
switch (suffix) {
case 'G':
case 'g':
        mem <<= 30;
        break;
case 'M':
case 'm':
        mem <<= 20;
        break;
case 'K':
case 'k':
        mem <<= 10;
        /* fall through */
default:
        break;
}
```

Don't put multiple statements on a single line unless you have something to hide:

```
if (condition) do_this;
  do_something_everytime;
```

Don't put multiple assignments on a single line either. Kernel coding style is super simple. Avoid tricky expressions.

Outside of comments, documentation and except in Kconfig, spaces are never used for indentation, and the above example is deliberately broken.

Get a decent editor and don't leave whitespace at the end of lines.

2) Breaking long lines and strings

Coding style is all about readability and maintainability using commonly available tools.

The limit on the length of lines is 80 columns and this is a strongly preferred limit.

3) Placing Braces and Spaces

The other issue that always comes up in C styling is the placement of braces. Unlike the indent size, there are few technical reasons to choose one placement strategy over the other, but the preferred way, as shown to us by the prophets Kernighan and Ritchie, is to put the opening brace last on the line, and put the closing brace first, thusly:

```
if (x is true) {
        we do y
}
```

This applies to all non-function statement blocks (if, switch, for, while, do). E.g.:

```
switch (action) {
case KOBJ_ADD:
        return "add";
case KOBJ_REMOVE:
        return "remove";
```

```
case KOBJ_CHANGE:
        return "change";
default:
        return NULL;
}
```

However, there is one special case, namely functions: they have the opening brace at the beginning of the next line, thus:

```
int function(int x)
{
        body of function
}
```

```
do {
        body of do-loop
} while (condition);
```

and

```
if (x == y) {
        ..
} else if (x > y) {
        ...
} else {
        ....
}
```

Rationale: K&R.

Do not unnecessarily use braces where a single statement will do.

```
if (condition)
        action();
```

and

```
if (condition)
        do_this();
else
        do_that();
```

This does not apply if only one branch of a conditional statement is a single statement; in the latter case use braces in both branches:

```
if (condition) {
        do_this();
        do_that();
} else {
        otherwise();
}
```

Spaces

Linux kernel style for use of spaces depends (mostly) on function-versus-keyword usage. Use a space after (most) keywords. The notable exceptions are sizeof, typeof, alignof, and __attribute__, which look somewhat like functions (and are usually used with parentheses in Linux, although they are not required in the language, as in: sizeof info afterstruct fileinfo info; is declared).

So use a space after these keywords:

```
if, switch, case, for, do, while
```

but not with sizeof, typeof, alignof, or __attribute__. E.g.,

```
s = sizeof(struct file);
```

Do not add spaces around (inside) parenthesized expressions. This example is **bad**:

```
s = sizeof( struct file );
```

When declaring pointer data or a function that returns a pointer type, the preferred use of * is adjacent to the data name or function name and not adjacent to the type name. Examples:

```
char *linux_banner;
unsigned long long memparse(char *ptr, char **retptr);
char *match_strdup(substring_t *s);
```

Use one space around (on each side of) most binary and ternary operators, such as any of these:

```
= + - < > * / % | & ^ <= >= == != ? :
```

but no space after unary operators:

```
& * + - ~ ! sizeof typeof alignof __attribute__ defined
```

no space before the postfix increment & decrement unary operators:

```
++ --
```

no space after the prefix increment & decrement unary operators:

```
++ --
```

and no space around the . and -> structure member operators.

4) Naming

C is a Spartan language, and so should your naming be. Unlike Modula-2 and Pascal programmers, C programmers do not use cute names like ThisVariableIsATemporaryCounter. A C programmer would call that variable tmp, which is much easier to write, and not the least more difficult to understand.

GLOBAL variables (to be used only if you **really** need them) need to have descriptive names, as do global functions. If you have a function that counts the number of active users, you should call that count_active_users() or similar, you should**not** call it cntusr().

Encoding the type of a function into the name (so-called Hungarian notation) is brain damaged - the compiler knows the types anyway and can check those, and it only confuses the programmer. No wonder MicroSoft makes buggy programs.

LOCAL variable names should be short, and to the point. If you have some random integer loop counter, it should probably be called i. Calling it loop_counter is non-productive, if there is no chance of it being mis-understood. Similarly, tmp can be just about any type of variable that is used to hold a temporary value.

5) Typedefs

Please don't use things like vps_t. It's a **mistake** to use typedef for structures and pointers. When you see a

vps_t a;

in the source, what does it mean? In contrast, if it says

struct virtual_container ***a;**

6) Functions

Functions should be short and sweet, and do just one thing. They should fit on one or two screenfuls of text (the ISO/ANSI screen size is 80x24, as we all know), and do one thing and do that well.

In source files, separate functions with one blank line. If the function is exported, the **EXPORT** macro for it should follow immediately after the closing function brace line. E.g.:

```
int system_is_up(void)
{
        return system_state == SYSTEM_RUNNING;
}
EXPORT_SYMBOL(system_is_up);
```

In function prototypes, include parameter names with their data types. Although this is not required by the C language, it is preferred in Linux because it is a simple way to add valuable information for the reader.

8) Commenting

Comments are good, but there is also a danger of over-commenting. NEVER try to explain HOW your code works in a comment: it's much better to write the code so that

The preferred style for long (multi-line) comments is:

```
/*
 * This is the preferred style for multi-line
 * comments in the Linux kernel source code.
 * Please use it consistently.
 *
 * Description:  A column of asterisks on the left side,
 * with beginning and ending almost-blank lines.
 */
```

12) Macros, Enums and RTL

Names of macros defining constants and labels in enums are capitalized.

#define CONSTANT 0x12345

Enums are preferred when defining several related constants.

CAPITALIZED macro names are appreciated but macros resembling functions may be named in lower case.

Generally, inline functions are preferable to macros resembling functions.

Macros with multiple statements should be enclosed in a do - while block:

```
#define macrofun(a, b, c)                \
    do {                                 \
        if (a == 5)                      \
            do_this(b, c);               \
    } while (0)
```

Things to avoid when using macros:

1. macros that affect control flow:

```
#define FOO(x)                           \
    do {                                 \
        if (blah(x) < 0)                 \
            return -EBUGGERED;           \
    } while (0)
```

is a **very** bad idea. It looks like a function call but exits the calling function; don't break the internal parsers of those who will read the code.

2. macros that depend on having a local variable with a magic name:

```
#define FOO(val) bar(index, val)
```

might look like a good thing, but it's confusing as hell when one reads the code and it's prone to breakage from seemingly innocent changes.

3) macros with arguments that are used as l-values: FOO(x) = y; will bite you if somebody e.g. turns FOO into an inline function.

4) forgetting about precedence: macros defining constants using expressions must enclose the expression in parentheses. Beware of similar issues with macros using parameters.

```
#define CONSTANT 0x4000
#define CONSTEXP (CONSTANT | 3)
```

5) namespace collisions when defining local variables in macros resembling functions:

```
#define FOO(x)                  \
({                              \
      typeof(x) ret;            \
      ret = calc_ret(x);        \
      (ret);                    \
})
```

ret is a common name for a local variable - __foo_ret is less likely to collide with an existing variable.

The cpp manual deals with macros exhaustively. The gcc internals manual also covers RTL which is used frequently with assembly language in the kernel.

14) Allocating memory

The kernel provides the following general purpose memory allocators: kmalloc(), kzalloc(), kmalloc_array(), kcalloc(), vmalloc(), and vzalloc(). Please refer to the API documentation for further information about them.

The preferred form for passing a size of a struct is the following:

```
p = kmalloc(sizeof(*p), ...);
```

Casting the return value which is a void pointer is redundant. The conversion from void pointer to any other pointer type is guaranteed by the C programming language.

The preferred form for allocating an array is the following:

```
p = kmalloc_array(n, sizeof(...), ...);
```

The preferred form for allocating a zeroed array is the following:

```
p = kcalloc(n, sizeof(...), ...);
```

Both forms check for overflow on the allocation size n * sizeof(...), and return NULL if that occurred.

INTRODUCTION TO BASH,SED AND AWK SCRIPTING

AWK

A text pattern scanning and processing language, created by Aho, Weinberger & Kernighan (hence the name). It can be quite sophisticated so this is NOT a complete guide, but should give you a taste of what awk can do. It can be very simple to use, and is strongly recommended. It can have an optional BEGIN{} section of commands that are done before processing any content of the file, then the main {} section works on each line of the file, and finally there is an optional END{} section of actions that happen after the file reading has finished:

BEGIN { …. initialization awk commands …} { …. awk commands for each line of the file…} END { …. finalization awk commands …}

For each line of the input file, it sees if there are any pattern-matching instructions, in which case it only operates on lines that match that pattern, otherwise it operates on all lines. These 'pattern-matching' commands can contain regular expressions as for grep. The awk commands can do some quite sophisticated maths and string manipulations, and awk also supports associative arrays. AWK sees each line as being made up of a number of fields, each being separated by a 'field separator'. By default, this is one or more space characters, so the line:

this is a line of text

contains 6 fields. For example, if I list all the files in a directory like this:

[mijp1@monty RandomNumbers]$ ls -l

total 2648

-rw------- 1 mijp1 mijp1 12817 Oct 22 00:13 normal_rand.agr

I can see the file size is reported as the 5th column of data. So if I wanted to know the total size of all the files in this directory I could do:

[mijp1@monty RandomNumbers]

$ ls -l | awk 'BEGIN {sum=0} {sum=sum+$5}

END {print sum}'

2668269

AWK Pattern Matching AWK is a line-oriented language. The pattern comes first, and then the action. Action statements are enclosed in { and }. Either the pattern may be missing, or the action may be missing, but, of course, not both AWK control statements include:

if (condition) statement [else statement]

while (condition) statement

do statement while (condition)

for (expr1; expr2; expr3) statement for (var in array) statement

break

continue exit [expression] AWK input/output

AWK input/output statements include:

close(file [, how]) Close file, pipe or co-process.

getline Set $0 from next input record.

AWK numeric functions include:

atan2(y, x) Returns the arctangent of y/x in radians. cos(expr) Returns the cosine of expr, which is in radians.

exp(expr) The exponential function. int(expr)

Truncates to integer. log(expr) The natural logarithm function.

Rand() Returns a random number N, between 0 and 1, such that 0 <= N < 1.

sin(expr) Returns the sine of expr, which is in radians.

sqrt(expr) The square root function. srand([expr])

Uses expr as a new seed for the random number generator.

If no expr is provided, the time of day

SED basics

sed = stream editor

sed performs basic text transformations on an input stream (a file or input from a pipeline) in a single pass through the stream, so it is very efficient. However, it is sed's ability to filter text in a pipeline which particularly distinguishes it from other types of editor.

sed -e 's/input/output/' my_file

will echo every line from my_file to standard output, changing the first occurrence of 'input' on each line into 'output'. NB sed is line-oriented, so if you wish to change every occurrence on each line, then you need to make it a 'greedy' search & replace like

so: sed -e 's/input/output/g' my_file

The expression within the /.../ can be a literal string or a regexp. NB By default the output is written to stdout. You may redirect this to a new file, or if you want to edit the existing file in place you should use the '-i' flag:

sed -e 's/input/output/' my_file > new_file

sed -i -e 's/input/output/' my_file

SED and regexps What if one of the characters you wish to use in the search command is a special symbol, like '/' (e.g. in a filename) or '*' etc? Then you must escape the symbol just as for grep (and awk). Say you want to edit a shell scripts to refer to /usr/local/bin and not /bin any more, then you could do this

sed -e 's/\/bin/\/usr\/local\/bin/' my_script > new_script What if you want to use a wildcard as part of your search – how do you write the output string? You need to use the special symbol '&' which corresponds to the pattern found. So say you want to take every line that starts with a number in your file and surround that number by parentheses:

sed -e 's/[0-9]*/(&)/' my_file where [0-9] is a regexp range for all single digit numbers, and the '*' is a repeat count, means any number of digits. You can also use positional instructions in your regexps, and even save part of the match in a pattern buffer to re-use elsewhere. Other SED commands

The general form is

sed -e '/pattern/ command' my_file

where 'pattern' is a regexp and 'command' can be one of 's' = search & replace, or 'p' = print, or 'd' = delete, or 'i'=insert, or 'a'=append, etc. Note that the default action is to print all lines that do not match anyway, so if you want to suppress this you need to invoke sed with the '-n' flag and then you can use the 'p' command to control what is printed. So if you want to do a listing of all the sub-directories you could use

ls -l | sed -n -e '/^d/ p' as the long-listing starts each line with the 'd' symbol if it is a directory, so this will only print out those lines that start with a 'd' symbol. Similarly, if you wanted to delete all lines that start with the comment symbol '#' you could use

sed -e '/^#/ d' my_file i.e. you can achieve the same effect in different ways!

You can also use the range form sed -e '1,100 command' my_file to execute 'command' on lines 1,100.

You can also use the special line number '$' to mean 'end of file'.

So if you wanted to delete all but the first 10 lines of a file, you could use

sed -e '11,$ d' my_file You can also use a pattern-range form, where the first regexp defines the start of the range, and the second the stop. So for instance, if you wanted to print all the lines from 'boot' to 'machine' in the a_file example you could do this:

Bash Programming

The UNIX shell program interprets user commands, which are either directly entered by the user, or which can be read from a file called the shell script or shell program. Shell

scripts are interpreted, not compiled. Apart from passing commands to the kernel, the main task of a shell is providing a user environment, which can be configured individually using shell resource configuration files.

Shell types

Just like people know different languages and dialects, your UNIX system will usually offer a variety of shell types:

- **sh** or Bourne Shell: the original shell still used on UNIX systems and in UNIX-related environments. This is the basic shell, a small program with few features. While this is not the standard shell, it is still available on every Linux system for compatibility with UNIX programs.
- **bash** or Bourne Again shell: the standard GNU shell, intuitive and flexible. Probably most advisable for beginning users while being at the same time a powerful tool for the advanced and professional user. On Linux, **bash** is the standard shell for common users. This shell is a so-called *superset* of the Bourne shell, a set of add-ons and plug-ins. This means that the Bourne Again shell is compatible with the Bourne shell: commands that work in **sh**, also work in **bash**. However, the reverse is not always the case. All examples and exercises in this book use **bash**.
- **csh** or C shell: the syntax of this shell resembles that of the C programming language. Sometimes asked for by programmers.
- **tcsh** or TENEX C shell: a superset of the common C shell, enhancing user-friendliness and speed. That is why some also call it the Turbo C shell.
- **ksh** or the Korn shell: sometimes appreciated by people with a UNIX background. A superset of the Bourne shell; with standard configuration a nightmare for beginning users.

The file /etc/shells gives an overview of known shells on a Linux system:

```
mia:~> cat /etc/shells
/bin/bash
/bin/sh
/bin/tcsh
/bin/csh
```

Your default shell is set in the /etc/passwd file, like this line for user *mia*:

```
mia:L2NOfqdlPrHwE:504:504:Mia Maya:/home/mia:/bin/bash
```

To switch from one shell to another, just enter the name of the new shell in the active terminal. The system finds the directory where the name occurs using the PATH settings, and since a shell is an executable file (program), the current shell activates it and it gets executed. A new prompt is usually shown, because each shell has its typical appearance:

```
mia:~> tcsh
[mia@post21 ~]$
```

Advantages of the Bourne Again SHell

Bash is the GNU shell

The GNU project (GNU's Not UNIX) provides tools for UNIX-like system administration which are free software and comply to UNIX standards.

Bash is an sh-compatible shell that incorporates useful features from the Korn shell (ksh) and C shell (csh). It is intended to conform to the IEEE POSIX P1003.2/ISO 9945.2 Shell and Tools standard. It offers functional improvements over sh for both programming and interactive use; these include command line editing, unlimited size command history, job control, shell functions and aliases, indexed arrays of unlimited size, and integer

arithmetic in any base from two to sixty-four. Bash can run most sh scripts without modification.

Like the other GNU projects, the bash initiative was started to preserve, protect and promote the freedom to use, study, copy, modify and redistribute software. It is generally known that such conditions stimulate creativity. This was also the case with the bash program, which has a lot of extra features that other shells can't offer.

Bash startup files

Startup files are scripts that are read and executed by Bash when it starts. The following subsections describe different ways to start the shell, and the startup files that are read consequently.

Invoked as an interactive login shell, or with `--login'

Interactive means you can enter commands. The shell is not running because a script has been activated. A login shell means that you got the shell after authenticating to the system, usually by giving your user name and password.

Files read:

- /etc/profile
- ~/.bash_profile, ~/.bash_login or ~/.profile: first existing readable file is read
- ~/.bash_logout upon logout.

Error messages are printed if configuration files exist but are not readable. If a file does not exist, bash searches for the next.

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

1. Define SED

2. What is the need of awk?
3. List GNU debugging tools
4. What is G++
5. Define GAS.
6. What is the need of MPATROL

PART - B (6 Marks)

1. Brief about GNU compiler tools.
2. Write about Graphical debuggers tool in Linux.
3. Explain C preprocessor and compiler in Linux.
4. Brief about GNU debugging tools.
5. Describe C++ compiler (g++) and assembler (gas) in Linux environment.
6. Write about profiling libraries mpatrol and valgrind in Linux.
7. Differentiate between archives libraries and dynamic shared object libraries.
8. Give a note on bash, sed and awk scripting.

PART - C (10 Marks)

1.Write a Linux program for Priority scheduling of processes.

2. Compare source code versioning and management tools in Linux.

3. Give a review on common programming practices and guidelines Linux.

Karpagam Academy of Higher Education
Department of Computer Science
Part –A-Multiple Choice Questions
II M.Sc(CS) (BATCH 2017-2019)
OPEN SOURCE TECHNOLOGIES

Class: II M.Sc. CS **Subject Code:17CSP302**

ONLINE EXAMINATIONS **ONE MARK QUESTIONS**

UNIT IV

Questions	opt1	opt2	opt3	opt4	Answer
What command is used to remove the directory?	rdir	remove	rd	rmdir	rmdir
What command is used with vi editor to delete a single character?	x	y	a	z	x
What hardware architectures are not supported by Red Hat?	SPARC	IBM-compatible	Alpha	Macintosh	Macintosh
How can you add Amit, a new user, to your system?	Using useradd	Using adduser	Using linuxcon	Using admin	Using adduser
What file specifies the order in which to use specified name services?	/etc/services	/etc/nsorder	/etc/nsswitch.conf	/etc/hosts	/etc/nsswitch.conf
How many primary partitions can exist on one drive?	16	4	2	1	4
In which directory can you store system user default files used for creating user directories?	/usr/tmp	/etc/default	/etc/skel	/etc/users	/etc/skel
What does FSF stand for?	Free Software File	File Server First	First Serve	Free Software Foundation	Free Software Foundation
Which of the following is a valid format for mounting a CD-ROM drive?	mount -t iso9660 /dev/cdrom /	mount /dev/cdrom	mount /mnt/cdr	All of the above	All of the above

What command do you use to create Linux file systems?	fdisk	mkfs	fsck	mount	mkfs
Which of the following command can you execute to count the number of lines in a file?	lc	wc - l	cl	count	wc - l
Which of the following is not a communication command?	grep	mail	mesg	write	grep
What command is used to display the characteristics of a process?	au	ps	du	pid	ps
What command is not used to list the files chap01, chap02 and chap04?	ls chap*	ls chap[124]	ls - x chap0[1	ls chap0[124]	ls chap[124]
What command is used with vi editor to replace text from cursor to right	S	s	R	r	R
What sign is used to back up over typing errors in vi?	!	$	#	@	#
What sign is used to erase or kill an entire line you have typed and start you are on a new line (but not display a	!	$	#	@	@
To increase the response time and throughput, the kernel minimizes the frequency of disk access by keeping a pool	Pooling	Spooling	Buffer cache	Swapping	Buffer cache
At start of process execution, STDOUT & STDERR	Point to current terminal device	Are closed	Point to special	Point to files	Point to current terminal device
wtmp and utmp files contain:	Temporary system data	User login-logout log	The user's	The user's su and sudo	User login-logout log
Which is the core of the operating system?	Shell	Kernel	Comma nds	Script	Kernel
ILP32 stands for	32 bit Integer, Long & Pointer	32 bit Integrated	32 bit Intelligen	32 bit Long & Pointer	32 bit Integer, Long & Pointer
Single Unix Specification Version 2 provides enhanced support for	16 bit Unix	32 bit Unix	64 bit Unix	8 bit Unix	64 bit Unix
Under UNIX the key board is the default input device and the monitor is the default output device	TRUE	FALSE	0	1	TRUE
Which among the following interacts directly with system hardware?	Shell	Commands	Kernel	Applications	Kernel

Applications communicate with kernel by using:	System Calls	C Programs	Shell Script	Shell	System Calls
Solaris is the name of a flavor of UNIX from	HP	IBM	Digital Equipme	Sun Microsystems	Sun Microsystems
Which of the following is “NOT” a UNIX variant ?	Solaris	AIX	IRIX	AS400	AS400
The system calls in UNIX is written using which language	C	C++	Assembl y	Fortran	C
Which of the following enables multi-tasking in UNIX?	Time Sharing	Multi programming	Multi user	Modularity	Time Sharing
Which of the following is considered as the super daemon in Unix?	sysinit	init	inetd	proc	init
Unix is which kind of Operating System?	Multi User	Multi Processes	Multi Tasking	All the above	All the above
SVR4 stands for?	Standard Version Release 4	System Version	Standard Five	System Five Release 4	Standard Version Release 4
Lp0 device file is used to access:	Floppy	Cdrom	Printer	Tape drive	Printer
Syntax of any Unix command is:	command [options]	command options	comman d	command options	command [options]
SVR4 was developed by	Sun Microsystems	AT&T	Universit y of	Sun and AT&T jointly	Sun and AT&T jointly
Which of these is not a Unix Flavor?	BSD	MAC	AIX	IRIX	MAC
Which of the following statement is FALSE ?	Unix supports multiple users	Linux is an open source	Shell takes	Shell provides the feature of	Shell takes care of inter process
Which of the following UNIX flavor is from IBM?	BSD	Solaris	HP-UX	AIX	AIX
x86-32 uses which programming model?	IP16	IP32	ILP16	ILP32	ILP32
What are the sizes of (Integer/Long/Pointer) in LP64 programming model?	8/8/2008	4/4/8	4/8/8	4/8/2004	4/8/8

What control character signals the end of the input file?	ctrl + a	ctrl + b	ctrl + c	ctrl + d	ctrl + d
How do you get help about the command “cp”?	help cp	man cp	cd ?	can cp	man cp
The dmesg command	Shows user login logoff attempts	Shows the syslog file for	kernel log	Shows the daemon log	kernel log messages
The command “mknod myfifo b 4 16”	Will create a block device if	Will create a block device	Will create a	Will create a file	Will create a block device if user is
Which command is used to set terminal IO characteristic?	tty	ctty	ptty	stty	stty
Which command is used to record a user login session in a file	macro	read	script	write	script
Which command is used to display the operating system name	os	Unix	kernel	uname	uname
Which command is used to display the unix version	uname -r	uname -n	uname -t	kernel	uname -r
Which command is used to print a file	print	ptr	lpr	none of the above	lpr
Using which command you find resource limits to the session?	rlimit	ulimit	setrlimit	getrlimit	ulimit
Which option of ls command used to view file inode number	–l	-o	–a	–i	–i
find / -name ‘*’ will	List all files and directories	List a file named * in /	List all files in /	List all files and	List all files and directories
Which command is used to display the octal value of the text	octal	text_oct	oct	od	od
Which command is used to view compressed text file contents	cat	type	zcat	print	zcat
Which command changes a file’s group owner	cgrp	chgrp	change	group	chgrp

UNIT V
SYLLABUS

Application Programming

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

APPLICATION PROGRAMMING

BASICS OF THE X WINDOWS SERVER ARCHITECTURE

The X Window System Architecture: overview

X was designed with a client-server architecture. The applications themselves are the clients; they communicate with the server and issue requests, also receiving information from the server.

The X server maintains exclusive control of the display and services requests from the clients. At this point, the advantages of using this model are pretty clear. Applications (clients) only need to know how to communicate with the server, and need not be concerned with the details of talking to the actual graphics display device. At the most basic level, a client tells the server stuff like "draw a line from here to here", or "render this string of text, using this font, at this position on-screen".

This would be no different from just using a graphics library to write our application. However the X model goes a step further. It doesn't constrain the client being in the same computer as the server. The protocol used to communicate between clients and server can work over a network, or actually, any "inter-process communication mechanism that provides a reliable octet stream". Of course, the preferred way to do this is by using the TCP/IP protocols. As we can see, the X model is really powerful; the classical example of this is running a processor-intensive application on a Cray computer, a database monitor on a Solaris server, an e-mail application on a small BSD mail server, and a

visualization program on an SGI server, and then displaying all those on my Linux workstation's screen.

So far we've seen that the X server is the one handling the actual graphics display. Also, since it's the X server which runs on the physical, actual computer the user is working on, it's the X server's responsibility to perform all actual interactions with the user. This includes reading the mouse and keyboard. All this information is relayed to the client, which of course will have to react to it.

X provides a library, aptly called Xlib, which handles all low-level client-server communication tasks. It sounds obvious that, then, the client has to invoke functions contained within Xlib to get work done.

At this point everything seems to be working fine. We have a server in charge of visual output and data input, client applications, and a way for them to communicate between each other. In picturing a hypothetical interaction between a client and a server, the client could ask the server to assign a rectangular area on the screen. Being the client, I'm not concerned with where i'm being displayed on the screen. I just tell the server "give me an area X by Y pixels in size", and then call functions to perform actions like "draw a line from here to there", "tell me whether the user is moving the mouse in my screen area" and so

Qt Programming

Qt Programming is a portable cross platform application user interface framework which works on the Windows, Linux and Mac OS X operating systems. Qt SDK helps you create graphical user interfaces (GUI's) for your applications that will run on Windows, Linux and Mac OS X.

For this article we are going to use the following simple steps to construct our first Qt HelloWorld program.

1. Create the directory QtHelloWorld in order to hold your Qt program

2. Change into your directory QtHelloWorld
3. Create the Qt source file main.cpp within the QtHelloWorld directory
4. Compile and run your QtHelloWorld program

Note: This document presumes you have the Qt SDK successfully installed on your operating system. If you do not have the Qt SDK installed on your system please see the following document for more information How to Install Qt SDK on Ubuntu Linux. This document also presumes you have basic knowledge of the C++ programming language. Essentially, the Qt SDK is programmed in C++ and relies heavily on C++ design and functions. **Note:** There are some compilation changes with Qt SDK 4.8 and Qt SDK 5.0, hopefully this article will resole the compilation issues between the two different Qt SDK versions.

For this exercise we are going to open up a terminal on Ubuntu Linux and issue the following command which will create the main directory for a Qt program.

- **Type/Copy/Paste:** mkdir QtHelloWorld
- **Type/Copy/Paste:** cd QtHelloWorld
- This is very important to make sure you are in the correct directory when creating your Qt program.

While we are in the QtHelloWorld directory, we are going to create our Qt Program source code file

- **Type/Copy/Paste:** nano main.cpp
- **or**
- **Type/Copy/Paste:** gedit main.cpp
- This command will create the main.cpp file for the Qt program

1. **Now add the following lines in the code box below to your main.cpp source code file.**

- **Type/Copy/Paste:**

[[Image:Create Your First Qt Program on Ubuntu Linux Step 4.jpg|center]]

```
#include <QApplication>
#include <QLabel>
#include <QWidget>
int main(int argc, char *argv[ ])
{
QApplication app(argc, argv);
QLabel hello("<center>Welcome to my first WikiHow Qt program</center>");
hello.setWindowTitle("My First WikiHow Qt Program");
hello.resize(400, 400);
hello.show();
return app.exec();
}
```

- **Save the file as main.cpp and exit**
  - Make sure you are in the **QtHelloWorld** directory before you enter the following commands below to build and compile the file.
- **Type/Copy/Paste:** qmake -project
  - This will create the Qt project file
- **Type/Copy/Paste:** qmake
  - This will create the Qt make file
- **Type/Copy/Paste:** make
  - This will compile the Qt make file on your system into an executable program. At this point, providing that there are no errors the file should compile into an executable program.
- Finally execute your program by running the Qt executable. Use the command ./ to run your executable file or type the name of the executable program on the terminal line.
- **Type/Copy/Paste:** ./QtHelloWorld

GtK+ Programming

GTK+ is a widget toolkit. Each user interface created by GTK+ consists of widgets. This is implemented in C using GObject, an object-oriented framework for C. Widgets are organized in a hierachy. The window widget is the main container. The user interface is then built by adding buttons, drop-down menus, input fields, and other widgets to the window. If you are creating complex user interfaces it is recommended to use GtkBuilder and its GTK-specific markup description language, instead of assembling the interface manually. You can also use a visual user interface editor, like Glade.

GTK+ is event-driven. The toolkit listens for events such as a click on a button, and passes the event to your application.

This chapter contains some tutorial information to get you started with GTK+ programming. It assumes that you have GTK+, its dependencies and a C compiler installed and ready to use. If you need to build GTK+ itself first, refer to the Compiling the GTK+ libraries section in this reference.

Basics

To begin our introduction to GTK, we'll start with a simple signal-based Gtk application. This program will create an empty 200 × 200 pixel window.

reate a new file with the following content named example-0.c.

```
#include <gtk/gtk.h>

static void

activate (GtkApplication* app,

gpointer        user_data)

{

GtkWidget *window;
```

```
window = gtk_application_window_new (app);

gtk_window_set_title (GTK_WINDOW (window), "Window");

gtk_window_set_default_size (GTK_WINDOW (window), 200, 200);

gtk_widget_show_all (window);

}

int

main (int    argc,

char **argv)

{

GtkApplication *app;

int status;

app = gtk_application_new ("org.gtk.example", G_APPLICATION_FLAGS_NONE);

g_signal_connect (app, "activate", G_CALLBACK (activate), NULL);

status = g_application_run (G_APPLICATION (app), argc, argv);

g_object_unref (app);

return status;

}
```

You can compile the program above with GCC using:

```
gcc `pkg-config --cflags gtk+-3.0` -o example-0 example-0.c `pkg-config --libs gtk+-3.0`
```

All GTK+ applications will, of course, include gtk/gtk.h, which declares functions, types and macros required by GTK+ applications.

Even if GTK+ installs multiple header files, only the top-level gtk/gtk.h header can be directly included by third party code. The compiler will abort with an error if any other header is directly included.

In a GTK+ application, the purpose of the main() function is to create a GtkApplication object and run it. In this example aGtkApplication pointer named app is called and then initialized using gtk_application_new().

When creating a GtkApplication you need to pick an application identifier (a name) and input to gtk_application_new() as parameter. For this example org.gtk.example is used but for choosing an identifier for your application see this guide. Lastlygtk_application_new() takes a GApplicationFlags as input for your application, if your application would have special needs.

Next the activate signal is connected to the activate() function above the main() functions. The activate signal will be sent when your application is launched with g_application_run() on the line below. The gtk_application_run() also takes as arguments the pointers to the command line arguments counter and string array; this allows GTK+ to parse specific command line arguments that control the behavior of GTK+ itself. The parsed arguments will be removed from the array, leaving the unrecognized ones for your application to parse.

Within g_application_run the activate() signal is sent and we then proceed into the activate() function of the application. Inside the activate() function we want to construct our GTK window, so that a window is shown when the application is launched. The call to gtk_application_window_new() will create a new GtkWindow and store it

inside the window pointer. The window will have a frame, a title bar, and window controls depending on the platform.

A window title is set using gtk_window_set_title(). This function takes a GtkWindow* pointer and a string as input. As our windowpointer is a GtkWidget pointer, we need to cast it to GtkWindow*. But instead of casting window via (GtkWindow*), window can be cast using the macro GTK_WINDOW(). GTK_WINDOW() will check if the pointer is an instance of the GtkWindow class, before casting, and emit a warning if the check fails. More information about this convention can be found here.

Finally the window size is set using gtk_window_set_default_size and the window is then shown by GTK via gtk_widget_show_all().

When you exit the window, by for example pressing the X, the g_application_run() in the main loop returns with a number which is saved inside an integer named "status". Afterwards, the GtkApplication object is freed from memory with g_object_unref(). Finally the status integer is returned and the GTK application exits.

While the program is running, GTK+ is receiving *events*. These are typically input events caused by the user interacting with your program, but also things like messages from the window manager or other applications. GTK+ processes these and as a result,*signals* may be emitted on your widgets. Connecting handlers for these signals is how you normally make your program do something in response to user input.

The following example is slightly more complex, and tries to showcase some of the capabilities of GTK+.

In the long tradition of programming languages and libraries, it is called *Hello, World*.

Python Programming

Python is this thing called a **programming language**. It takes text that you've written (usually referred to as **code**), turns it into instructions for your computer, and runs those instructions. We'll be learning how to write code to do cool and useful stuff. No longer will you be bound to use *others'* programs to do things with your computer - you can make your own!

Practically, Python is just another program on your computer. The first thing to learn is how to use and interact with it. There are in fact many ways to do this; the first one to learn is to interact with python's interpreter, using your **operating system's** (OS) console.

A **console** (or 'terminal', or 'command prompt') is a *textual* way to interact with your OS, just as the 'desktop', in conjunction with your mouse, is the *graphical* way to interact your system.

Opening a console on Mac OS XOS X's standard console is a program called **Terminal**. Open Terminal by navigating to Applications, then Utilities, then double-click the **Terminal** program. You can also easily search for it in the system search tool in the top right.

The command line Terminal is a tool for interacting with your computer. A window will open with a command line prompt message, something like this:

```
mycomputer:~ myusername$
```

Opening a console on Linux

Different linux distributions (e.g Ubuntu, Fedora, Mint) may have different console programs, usually referred to as a terminal. The exact terminal you start up, and how, can depend on your distribution. On Ubuntu, you will likely want to open **Gnome Terminal**. It should present a prompt like this:

```
myusername@mycomputer:~$
```

Opening a console on Windows

Window's console is called the Command Prompt, named **cmd**. An easy way to get to it is by using the key combination Windows+R (Windows meaning the windows logo button), which should open a Run dialog. Then type **cmd** and hit Enter or click Ok. You can also search for it from the start menu. It should look like:

```
C:\Users\myusername>
```

Window's Command Prompt is not quite as powerful as its counterparts on Linux and OS X, so you might like to start the Python Interpreter (see just below) directly, or using the **IDLE**program that Python comes with. You can find these in the Start menu.

Using Python

The python program that you have installed will by default act as something called an **interpreter**. An interpreter takes text commands and runs them as you enter them - very handy for trying things out.

Just type **python** at your console, hit Enter, and you should enter Python's Interpreter.

To find out which version of python you're running, instead type python -V in your console to tell you.

Interacting With Python

After Python opens, it will show you some contextual information similar to this:

```
Python 3.5.0 (default, Sep 20 2015, 11:28:25)
[GCC 5.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Note

The prompt >>> on the last line indicates that you are now in an interactive Python interpeter session, also called the "Python shell". **This is different from the normal terminal command prompt!**

You can now enter some code for python to run. Try:

```
print("Hello world")
```

Press Enter and see what happens. After showing the results, Python will bring you back to the interactive prompt, where you could enter another command:

```
>>> print("Hello world")
Hello world
>>> (1 + 4) * 2
10
```

An extremely useful command is help(), which enters a help functionality to explore all the stuff python lets you do, right from the interpreter. Press q to close the help window and return to the Python prompt.

To leave the interactive shell and go back to the console (the *system* shell), press Ctrl-Z and then Enter on Windows, or Ctrl-D on OS X or Linux. Alternatively, you could also run the python command exit()!

Programming GUI Applications with Localisation Support

Just above we demonstrated entering a command to figure out some math. Try some math commands of your own! What operations does python know? Get it to tell you what 239 and 588 added together, and then squared is.

Solution

Show

Running Python files

When you have a lot of python code to run, you will want to save it into a file, so for instance, you can modify small parts of it (fix a bug) and re-run the code without having to repeatedly re-type the rest. Instead of typing commands in one-by-one you can save your code to a file and pass the file name to the **python** program. It will execute that file's code instead of launching its interactive interpreter.

Let's try that! Create a file hello.py in your current directory with your favorite code editor and write the print command from above. Now save that file. On Linux or OS X, you can also run touch hello.py to create an empty file to edit. To run this file with python, it's pretty easy:

```
$ python hello.py
```

Note

Make sure you are at your system command prompt, which will have $ or > at the end, **not** at python's (which has >>> instead)!

On Windows you should also be able to double-click the Python file to run it.

When pressing Enter now, the file is executed and you see the output as before. But this time, after Python finished executing all commands from that file it exits back to the system command prompt, instead of going back to the interactive shell.

And now we are all set and can get started with turtle!

Note

Not getting "Hello world" but some crazy error about "can't open file" or "No such file or directory?" Your command line might not be running in the directory that you saved the file in. You can change the working directory of your current command line with

the **cd** command, which stands for "change directory". On Windows, you might want something like:

```
> cd Desktop\Python_Exercises
```

On Linux or OS X, you might want something like:

```
$ cd Desktop/Python_Exercises
```

This changes to the directory Python_Exercises under the Desktop folder (yours might be somewhere different). If you don't know the location of the directory where you saved the file, you can simply drag the directory to the command line window.

POSSIBLE QUESTIONS

PART – A (20 X 1 = 20 MARKS)

ONLINE QUESTIONS

(2 Marks)

1. Define XClient
2. Define Xserver
3. What is the need of GTK+
4. What is python?
5. What are the protocols needed for X Windows server architecture.

PART - B (6 Marks)

1. Explain gtk+ programming in Linux.
2. Describe Qt programming in Linux.
3. Explain the architecture of gtk+ programming in Linux.
4. Explain programming of GUI applications with localization support in Linux.
5. Write a note on XClient and XServer in XWindows architecture.
6. Explain programming GUI applications in Linux.
7. Brief about Qt programming in Linux.

PART - C (10 Marks)

1. Write a Linux program for communication between Client and Server.
2. Write a review on communication of Xclient and Xserver in Xwindows architecture.
3. Compare protocols in X Windows server architecture.

Karpagam Academy of Higher Education
Department of CS, CA & IT
Part –A-Multiple Choice Questions
II M.Sc(CS) (BATCH 2017-2019)
OPEN SOURCE TECHNOLOGIES

Class: II M.Sc. CS **Subject Code:17CSP302**

ONLINE EXAMINATIONS **ONE MARK QUESTIONS**

UNIT V

Questions	opt1	opt2	opt3	opt4	Answer
What command is used to remove the directory?	rdir	remove	rd	rmdir	rmdir
What command is used with vi editor to delete a single character?	x	y	a	z	x
What hardware architectures are not supported by Red Hat?	SPARC	IBM-compatible	Alpha	Macintosh	Macintosh
How can you add Amit, a new user, to your system?	Using useradd	Using adduser	Using linuxconf	Using admin	Using adduser
What file specifies the order in which to use specified name services?	/etc/services	/etc/nsorder	/etc/nsswitch.conf	/etc/hosts	/etc/nsswitch.conf
How many primary partitions can exist on one drive?	16	4	2	1	4
In which directory can you store system user default files used for creating user directories?	/usr/tmp	/etc/default	/etc/skel	/etc/users	/etc/skel
What does FSF stand for?	Free Software File	File Server First	First Serve First	Free Software Foundation	Free Software Foundation
Which of the following is a valid format for mounting a CD-ROM drive?	mount -t iso9660	mount /dev/cdrom	mount /mnt/cdrom	All of the above	All of the above

What command do you use to create Linux file systems?	fdisk	mkfs	fsck	mount	mkfs
Which of the following command can you execute to count the number of lines in a file?	lc	wc - l	cl	count	wc - l
Which of the following is not a communication command?	grep	mail	mesg	write	grep
What command is used to display the characteristics of a process?	au	ps	du	pid	ps
What command is not used to list the files chap01, chap02 and chap04?	ls chap*	ls chap[124]	ls - x chap0[124]	ls chap0[124]	ls chap[124]
What command is used with vi editor to replace text from cursor to right	S	s	R	r	R
What sign is used to back up over typing errors in vi?	!	$	#	@	#
What sign is used to erase or kill an entire line you have typed and start you are on a new line (but not display a	!	$	#	@	@
To increase the response time and throughput, the kernel minimizes the frequency of disk access by keeping a pool	Pooling	Spooling	Buffer cache	Swapping	Buffer cache
At start of process execution, STDOUT & STDERR	Point to current	Are closed	Point to special files on the	Point to files	Point to current
wtmp and utmp files contain:	Temporary system data	User login-logout log	The user's command	The user's su and sudo attempts	User login-logout log
Which is the core of the operating system?	Shell	Kernel	Commands	Script	Kernel
ILP32 stands for	32 bit Integer, Long &	32 bit Integrated	32 bit Intelligent Long & Pointer	32 bit Long & Pointer	32 bit Integer, Long & Pointer
Single Unix Specification Version 2 provides enhanced support for	16 bit Unix	32 bit Unix	64 bit Unix	8 bit Unix	64 bit Unix
Under UNIX the key board is the default input device and the monitor is the default output device	TRUE	FALSE	0	1	TRUE
Which among the following interacts directly with system hardware?	Shell	Commands	Kernel	Applications	Kernel

Applications communicate with kernel by using:	System Calls	C Programs	Shell Script	Shell	System Calls
Solaris is the name of a flavor of UNIX from	HP	IBM	Digital Equipment Corp	Sun Microsystems	Sun Microsystems
Which of the following is “NOT” a UNIX variant ?	Solaris	AIX	IRIX	AS400	AS400
The system calls in UNIX is written using which language	C	C++	Assembly Language	Fortran	C
Which of the following enables multi-tasking in UNIX?	Time Sharing	Multi programming	Multi user	Modularity	Time Sharing
Which of the following is considered as the super daemon in Unix?	sysinit	init	inetd	proc	init
Unix is which kind of Operating System?	Multi User	Multi Processes	Multi Tasking	All the above	All the above
SVR4 stands for?	Standard Version	System Version	Standard Five Release 4	System Five Release 4	Standard Version
Lp0 device file is used to access:	Floppy	Cdrom	Printer	Tape drive	Printer
Syntax of any Unix command is:	command [options]	command options	command [options]	command options arguments	command [options]
SVR4 was developed by	Sun Microsystems	AT&T	University of Berkeley	Sun and AT&T jointly	Sun and AT&T jointly
Which of these is not a Unix Flavor?	BSD	MAC	AIX	IRIX	MAC
Which of the following statement is FALSE ?	Unix supports multiple users	Linux is an open source	Shell takes care of inter process	Shell provides the feature of I/O	Shell takes care of inter
Which of the following UNIX flavor is from IBM?	BSD	Solaris	HP-UX	AIX	AIX
x86-32 uses which programming model?	IP16	IP32	ILP16	ILP32	ILP32
What are the sizes of (Integer/Long/Pointer) in LP64 programming model?	8/8/2008	4/4/8	4/8/8	4/8/2004	4/8/8

What control character signals the end of the input file?	ctrl + a	ctrl + b	ctrl + c	ctrl + d	ctrl + d
How do you get help about the command “cp”?	help cp	man cp	cd ?	can cp	man cp
The dmesg command	Shows user login logoff	Shows the syslog file	kernel log messages	Shows the daemon log	kernel log messages
The command “mknod myfifo b 4 16”	Will create a block device	Will create a block device	Will create a FIFO if user is	Will create a file	Will create a block device if
Which command is used to set terminal IO characteristic?	tty	ctty	ptty	stty	stty
Which command is used to record a user login session in a file	macro	read	script	write	script
Which command is used to display the operating system name	os	Unix	kernel	uname	uname
Which command is used to display the unix version	uname -r	uname -n	uname -t	kernel	uname -r
Which command is used to print a file	print	ptr	lpr	none of the above	lpr
Using which command you find resource limits to the session?	rlimit	ulimit	setrlimit	getrlimit	ulimit
Which option of ls command used to view file inode number	–l	-o	–a	–i	–i
find / -name ‘*’ will	List all files and	List a file named * in /	List all files in / directory	List all files and directories in /	List all files and directories
Which command is used to display the octal value of the text	octal	text_oct	oct	od	od
Which command is used to view compressed text file contents	cat	type	zcat	print	zcat
Which command changes a file’s group owner	cgrp	chgrp	change	group	chgrp