

EX.NO:01

DATE:

LINUX INSTALLATION

Aim:

To install the Linux operating system (FEDORA-open source).

INTRODUCTION TO OPEN SOURCE SOFTWARE

Open-source software (OSS) is computer software that is available in sourcecode form for which the source code and certain other rights normally reserved for copyright holders are provided under a software license that permits users to study,change, and improve the software. Open source licenses meet the requirements of the Open Source Definition.

Popular distributions:

Well-known Linux distributions include:

- CentOS, a distribution derived from the same sources used by Red Hat, maintained by a dedicated volunteer community of developers with both 100% Red Hat-compatible versions and an upgraded version that is not always 100% upstream compatible Debian, a non-commercial distribution maintained by a volunteer developer
- community with a strong commitment to free software principles Fedora, a community distribution sponsored by Red Hat
- Gentoo, a distribution targeted at power users, known for its FreeBSD Ports-like automated system for compiling applications from source code Knoppix, the first Live CD distribution to run completely from removable
- media without installation to a hard disk, derived from Debian

INTRODUCTION TO LINUX OPERATING SYSTEM

Linux refers to the family of Unix-like computer operating systems using the Linux kernel.

Linux can be installed on a wide variety of computer hardware, ranging from mobile phones, tablet computers and video game consoles, to mainframes and supercomputers. Linux is a leading server operating system, and runs the 10 fastest supercomputers in the world. Use of Linux by end-users or consumers has increased in recent years, partly owing to the popular Ubuntu, Fedora, and openSUSE distributions and the emergence of net books with pre-installed Linux systems and smartphones running embedded Linux.

The development of Linux is one of the most prominent examples of free and open source software collaboration; typically all the underlying source code can be used, freely modified, and redistributed, both commercially and non-commercially, by anyone under licenses such as the GNU General Public License. Typically Linux is packaged in a format known as a Linux distribution for desktop and server use.

Linux distributions include the Linux kernel and all of the supporting software required to run a complete system, such as utilities and libraries, the X Window System, the GNOME and KDE desktop environments, and the Apache HTTP Server. Commonly used applications with desktop Linux systems include the Mozilla Firefox web-browser, the OpenOffice.org office application suite and the GIMP image editor.

The name "Linux" comes from the Linux kernel, originally written in 1991 by Linus Torvalds. The main supporting user space system tools and libraries from the GNU Project (announced in 1983 by Richard Stallman) are the basis for the Free Software Foundation's preferred name GNU/Linux.

STANDARD INSTALLATION OF FEDORA 14

Fedora is an RPM-based, general purpose collection of software including an operating system based on the Linux kernel, developed by the community-supported Fedora Project and sponsored by Red Hat. The Fedora Project's mission is to lead the advancement of free and open source software and content as a collaborative community.

One of Fedora's main objectives is not only to contain software distributed under a free and open source license, but also to be on the leading edge of such technologies. Fedora developers prefer to make upstream changes instead of applying fixes specifically for Fedora—this ensures that their updates are available to all Linux distributions.

Distribution:

Package Kit, the default package manager front-end on Fedora.

The Fedora Project distributes Fedora in several different ways:

- Fedora DVD/CD set – a DVD or CD set of all major Fedora packages at time of shipping;
- Live images – CD or DVD sized images that can be used to create a Live CD or boot from a USB flash drive and optionally install to a hard disk;
- Minimal CD – used for installing over HTTP, FTP or NFS.

Features of Fedora 13 include:

- Automatic printer-driver & Automatic language pack installation
- Redesigned user-account tool
- A new way to install Fedora over the Internet
- Updates to NFS
- Inclusion of Zarafa Open Source edition
- KDE PulseAudio Integration
- New command-line interface for NetworkManager

Features of Fedora 14 include:

Fedora 14, codenamed Laughlin, was released on November 2, 2010.

- Updated Boost to the upstream 1.44 release
- Addition of the D compiler (LDC) and D standard runtime library (Tango)
- Updated Fedora's Eclipse stack to Helios releases
- Inclusion of virt-v2v tool
- Inclusion of Spice framework for VDI deployment
- NetBeans IDE updated to the 6.9 release
- Inclusion of a tech preview of the GNOME Shell environment

Basic Installation Steps:

1. Boot from the DVD.



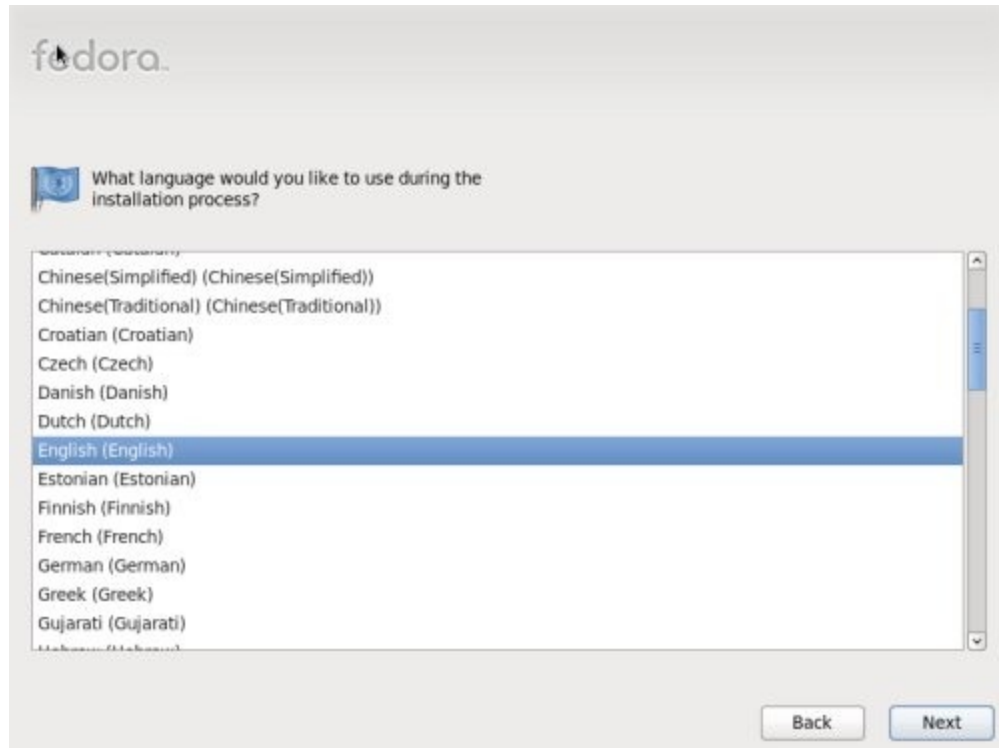
2. Skip the media test by pressing the tab and return keys on the keyboard.



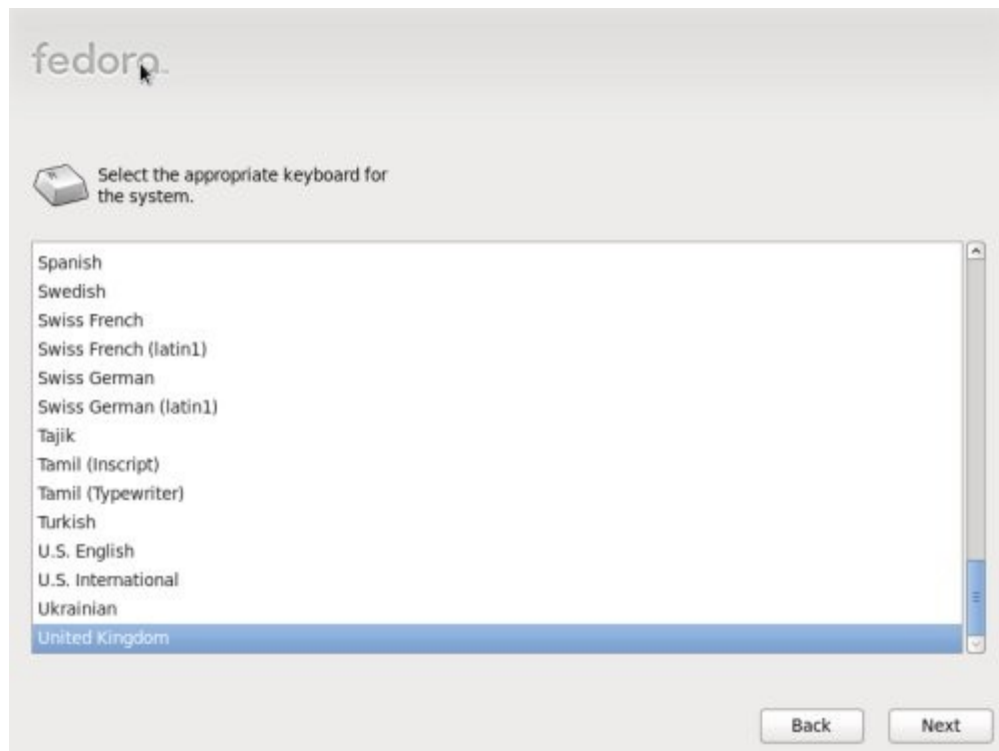
3. Click the "Next" button on the welcome screen.



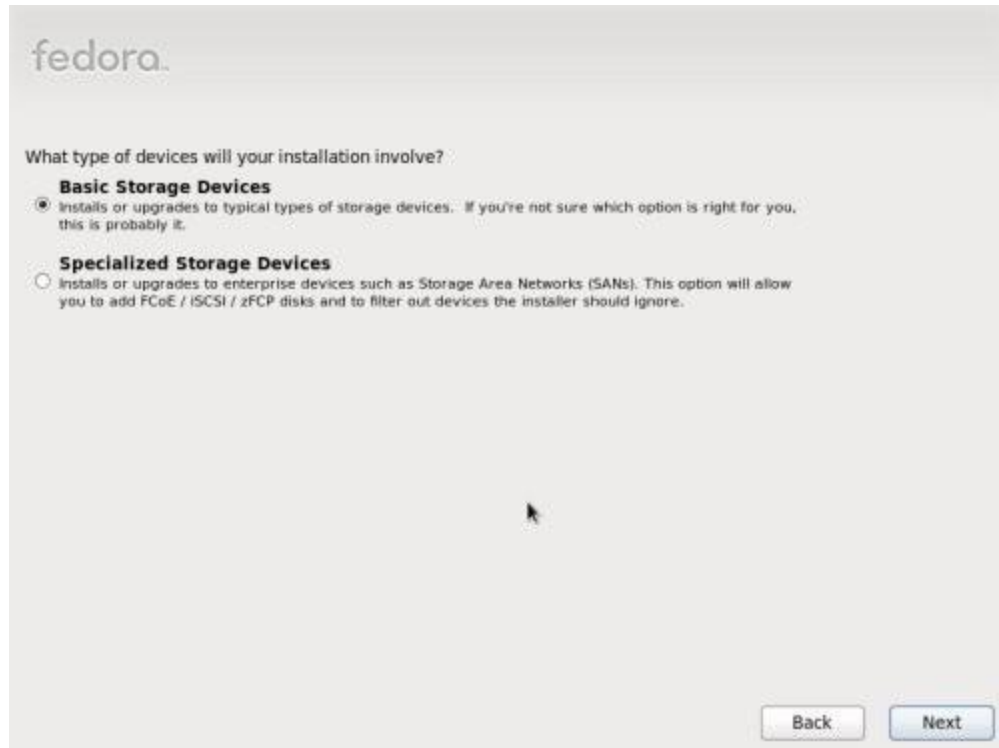
4. Select the appropriate language, then click the "Next" button.



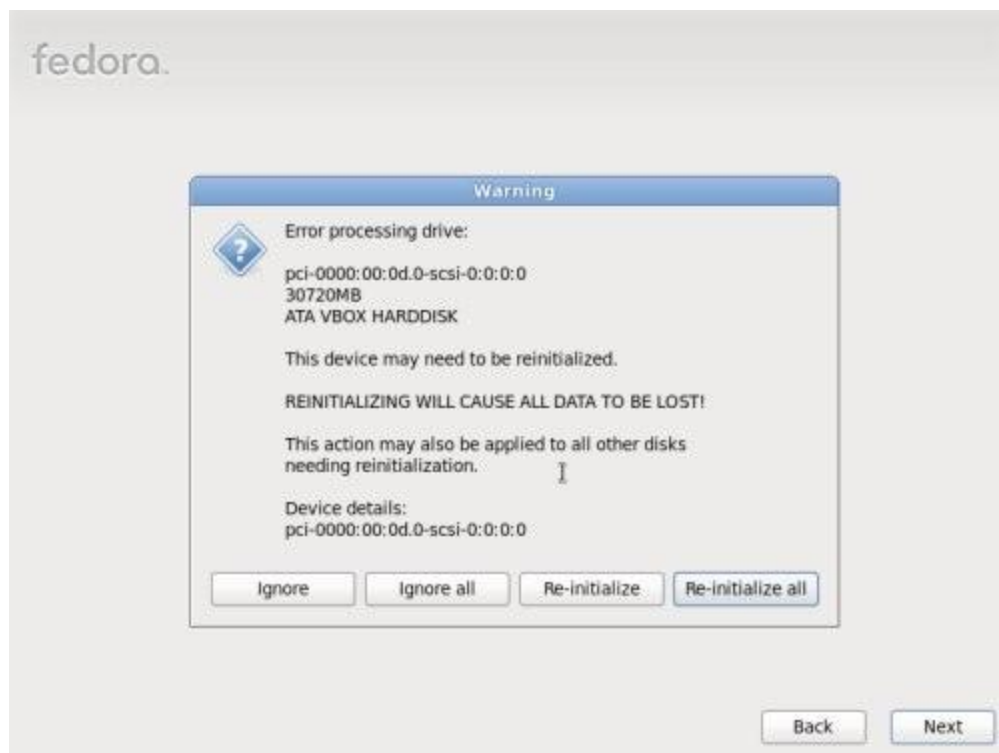
5. Select the appropriate keyboard layout, then click the "Next" button.



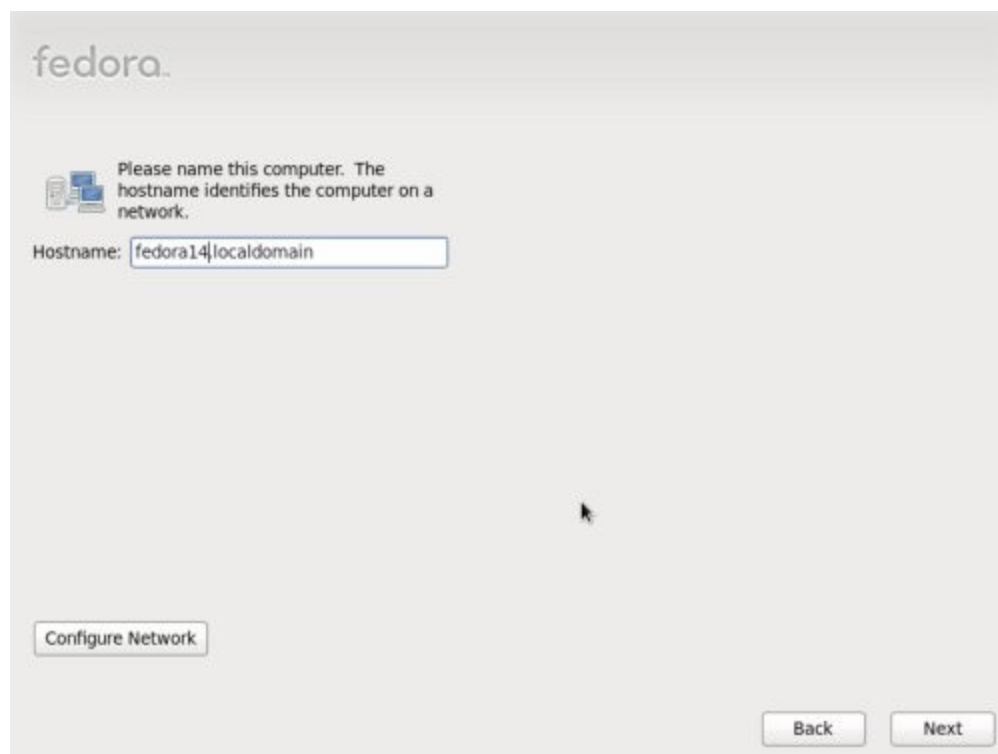
6. When prompted, click the "Re-initialize drive" button.



7. When prompted, click the "Re-initialize" button.



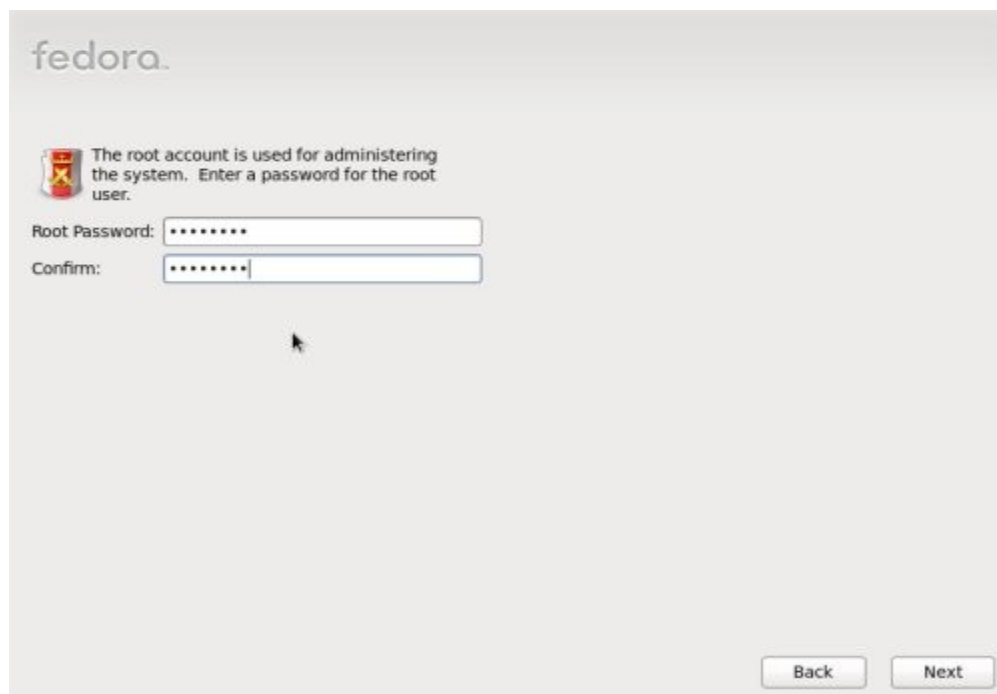
8. Enter a fully qualified hostname, then click the "Next" button.



9. Select the appropriate timezone by clicking on the nearest city on the map, then click the "Next" button.



10. Enter the root password, then click the "Next" button.

A screenshot of the Fedora root password setup window. The window has a light gray background. At the top left, the word "fedora." is displayed in a lowercase, sans-serif font. Below it, on the left, is the Fedora logo (a red shield with a yellow sun and a red cross). To the right of the logo, the text reads: "The root account is used for administering the system. Enter a password for the root user." Below this text, there are two password input fields. The first field is labeled "Root Password:" and contains seven asterisks. The second field is labeled "Confirm:" and contains seven asterisks. At the bottom right of the window, there are two buttons: "Back" and "Next".

fedora.

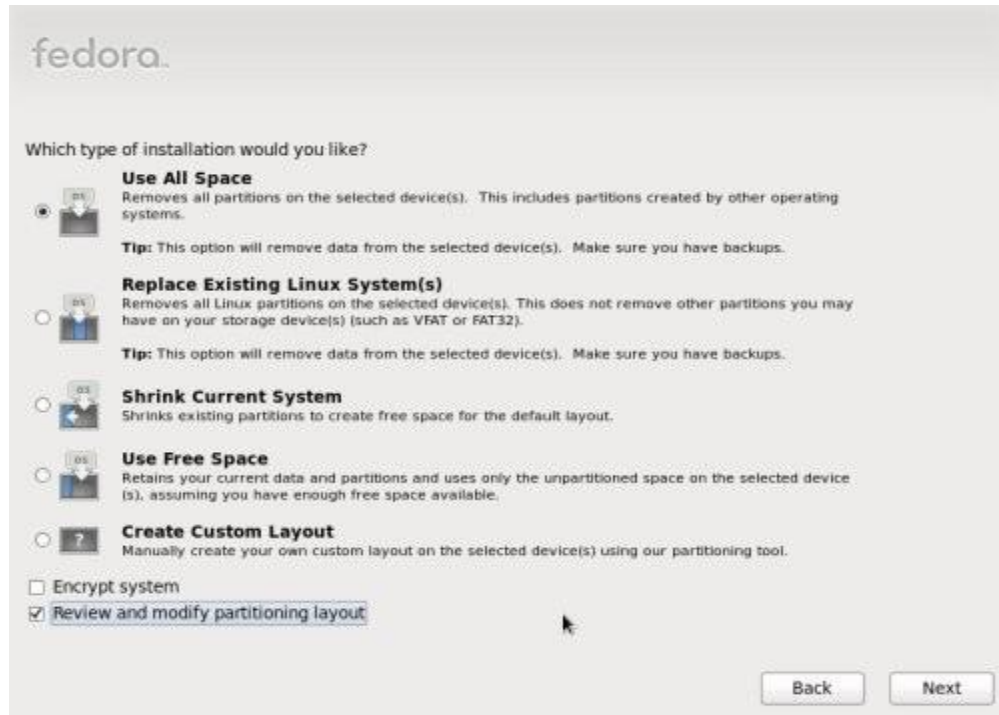
The root account is used for administering the system. Enter a password for the root user.

Root Password:

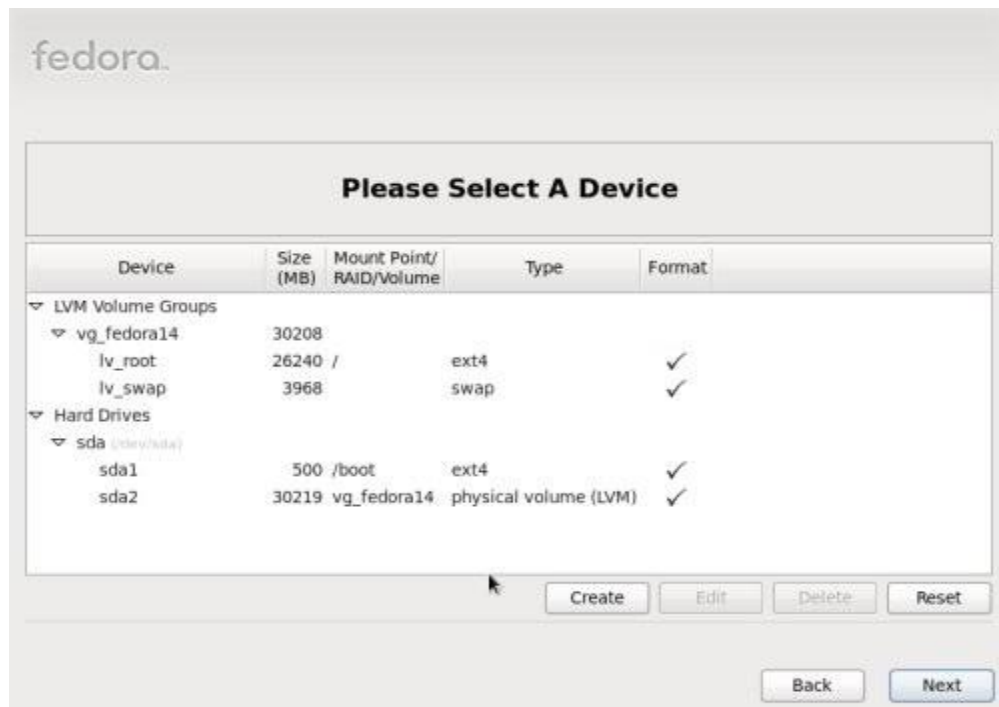
Confirm:

Back Next

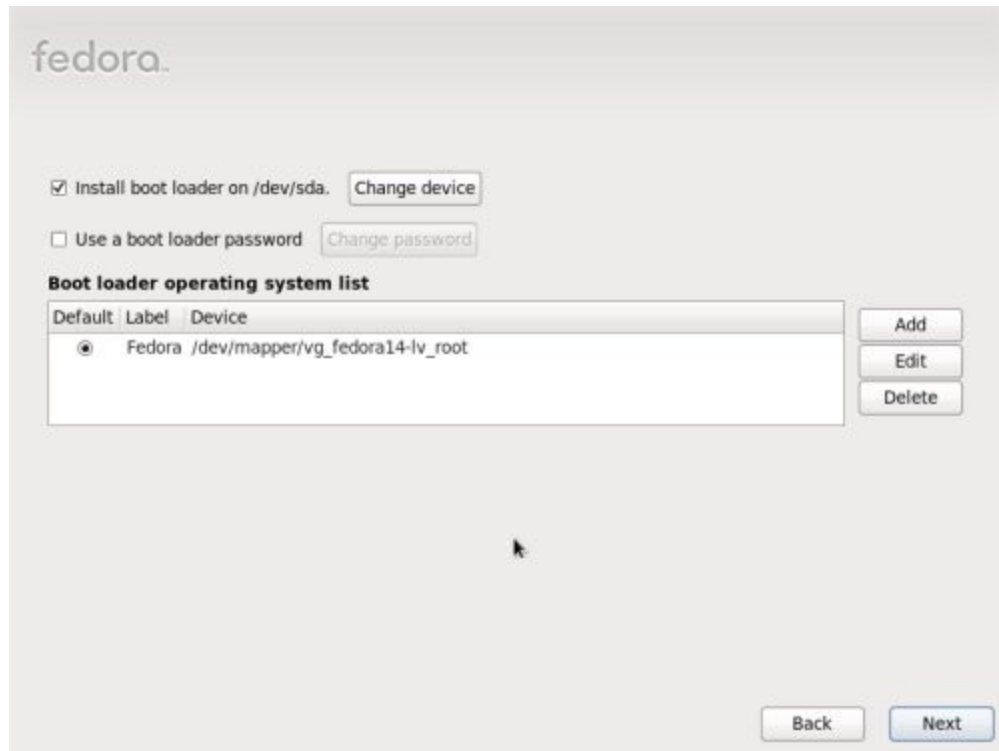
11. Select the "Use All Space" option, uncheck the "Encrypt system" option and check the "Review and modify partitioning layout" option, then click the "Next" button.



12. The current partitioning layout is presented. If the OS is to be used for an Oracle installation, make sure the swap partition is at least 2G (2048M) in size. Once you are happy with the partition structure, click the "Next" button followed by the "Format" and "Write changes to disk" button.



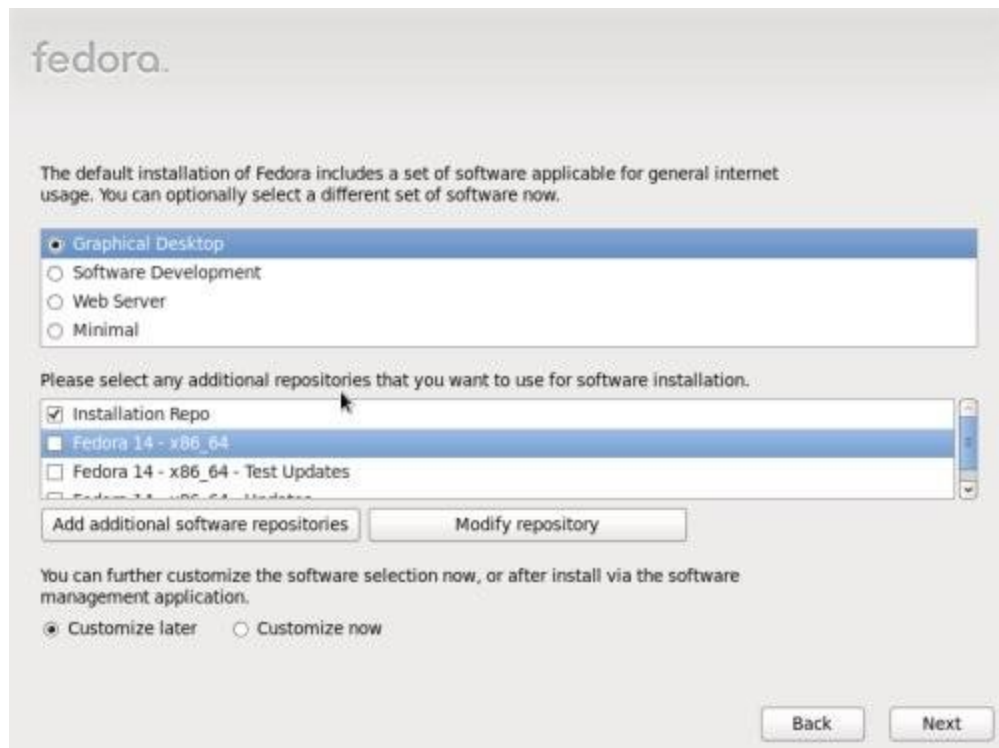
13. Accept the default boot loader settings by clicking the "Next" button.



The screenshot shows the Fedora installer's boot loader configuration screen. At the top, the 'fedora.' logo is visible. Below it, there are two options: 'Install boot loader on /dev/sda.' which is checked, and 'Use a boot loader password' which is unchecked. Each option has a 'Change' button next to it. A section titled 'Boot loader operating system list' contains a table with columns 'Default', 'Label', and 'Device'. The table has one entry: 'Fedora' as the label and '/dev/mapper/vg_fedora14-lv_root' as the device, with a radio button in the 'Default' column. To the right of the table are 'Add', 'Edit', and 'Delete' buttons. At the bottom right, there are 'Back' and 'Next' buttons.

Default	Label	Device
<input checked="" type="radio"/>	Fedora	/dev/mapper/vg_fedora14-lv_root

14. Select the appropriate packages to install and check all three additional repositories.

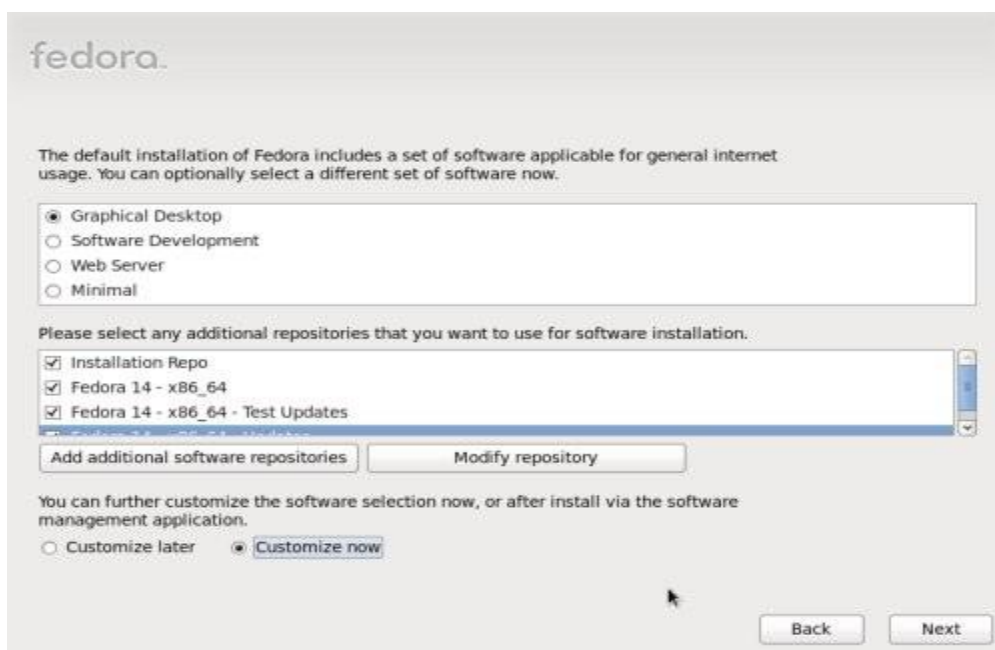


The screenshot shows the Fedora installer's software selection screen. It starts with the 'fedora.' logo and a paragraph explaining that the default installation includes software for general internet usage, but users can select a different set. Below this is a list of software environments: 'Graphical Desktop' (selected with a radio button), 'Software Development', 'Web Server', and 'Minimal'. A section titled 'Please select any additional repositories that you want to use for software installation.' follows, with a list of repositories: 'Installation Repo' (checked), 'Fedora 14 - x86_64' (checked), 'Fedora 14 - x86_64 - Test Updates' (unchecked), and 'Fedora 14 - x86_64 - Updates' (unchecked). Below the list are 'Add additional software repositories' and 'Modify repository' buttons. At the bottom, there is a paragraph about customizing software selection, with 'Customize later' selected and 'Customize now' as an option. 'Back' and 'Next' buttons are at the bottom right.

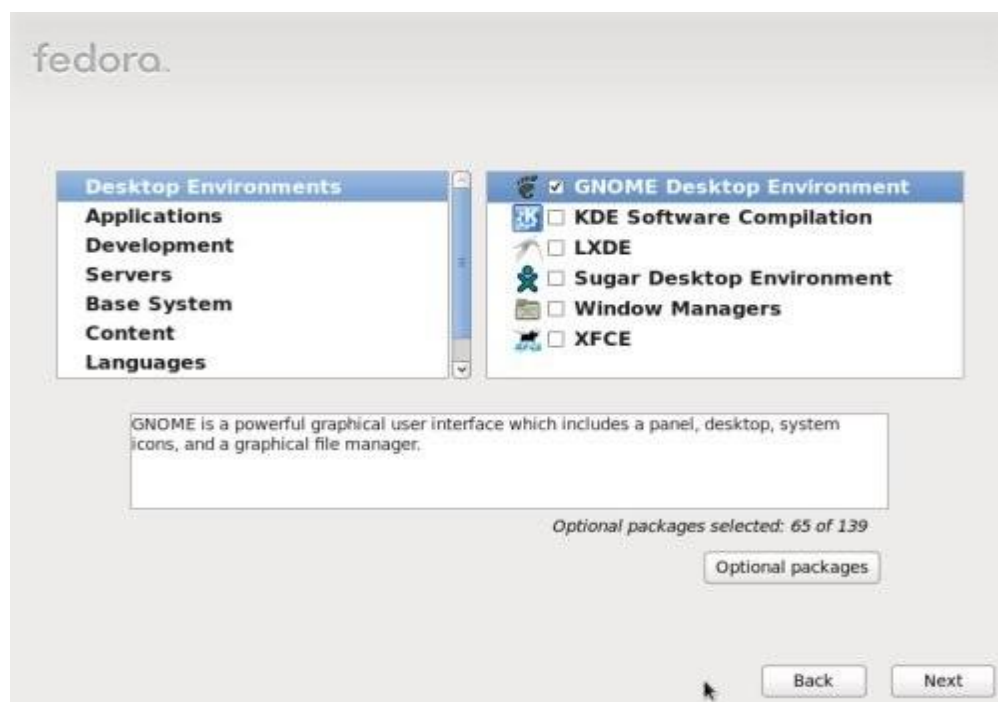
15. Selecting the repositories will require you to enter your network configuration. Select the appropriate network interface, highlight the "System eth0" adapter and click the "Edit" button. Click on the the "IPv4 Settings" tab, select the appropriate method and enter the required information. When complete, click the "Apply" and "Close" buttons.



16. Once the repositories are selected, select the "Customize Now" option and click the "Next" button.



17. Select the appropriate package groups and optional packages for your installation and click the "Next" button.



18. Wait while the installation completes.



19. When the installation is complete, restart the machine as instructed. When the machine has restarted, click the "Forward" button on the welcome screen.



20. After you've logged in, you are presented with the default Fedora desktop.



EX.NO:02

DATE:

MYSQL INSTALLATION

Aim:

To install the **mysql** package in the fedora operating system.

```
[root@server root]# cd /root
[root@server root]# mkdir /root/files /data /data/mysql
[root@server root]# cd files
[root@server root]# wget http://www.d9x.net/downloads/unix/mysql-4.0.12.tar.gz
[root@server root]# tar xzvf mysql-4.0.12.tar.gz
[root@server root]# cd mysql-4.0.12
if your using FreeBSD use "adduser" instead of "useradd".
[root@server root]# groupadd mysql
[root@server root]# useradd -g mysql mysql
[root@server root]# ./configure --prefix=/usr/local/mysql--datadir=/data/mysql
```

DataDir is where your mysql databases will be stored including the database which contains all mysql user accounts; you can easily backup all databases/users by saving the contents of this folder.
/data/mysql

NOTE: If you see ERROR: No curses/termcap library found" find where libncurses.so.5 is installed on your system and add the following flag to your

configure statement : --with-named-curses-libs=/usr/lib/libncurses.so.5 (or where ever it may be)

if you get a "mod_auth_dbm" Error Copy the ndbm.h file from /usr/include to the /usr/include/db1 directory (or install db1-devel)if you get a "no acceptable C compiler found in \$PATH" Error

Install a c compiler, example: gcc

if you get "libmysql.c:1349: warning: passing arg 5 of `gethostbyname_r'

from incompatible pointer type" Error

you need to install a c++ compiler, install: libstdc++-devel, gcc-c++ (same version as gcc)

```
[root@server root]# make
[root@server root]# make install
[root@server root]# ./scripts/mysql_install_db
```

```
[root@server root]# ln -s /usr/local/mysql/share/mysql/mysql.server
/sbin/mysqld
[root@server root]# chown -R mysql:mysql /data/mysql
[root@server root]# chown -R mysql:mysql /usr/local/mysql
```

Start, Stop & Restart MySQL

```
[root@server root]# mysqld start
[root@server root]# mysqld stop
[root@server root]# mysqld restart
```

Using MySQL

Export Database:

```
[root@server root]# /usr/local/mysql/bin/mysqldump -u root -p
database_name > database_name.sql
```

Import Database:

```
[root@server root]# /usr/local/mysql/bin/mysql -u root -p database_name <
database_name.sql
```

Set root password:

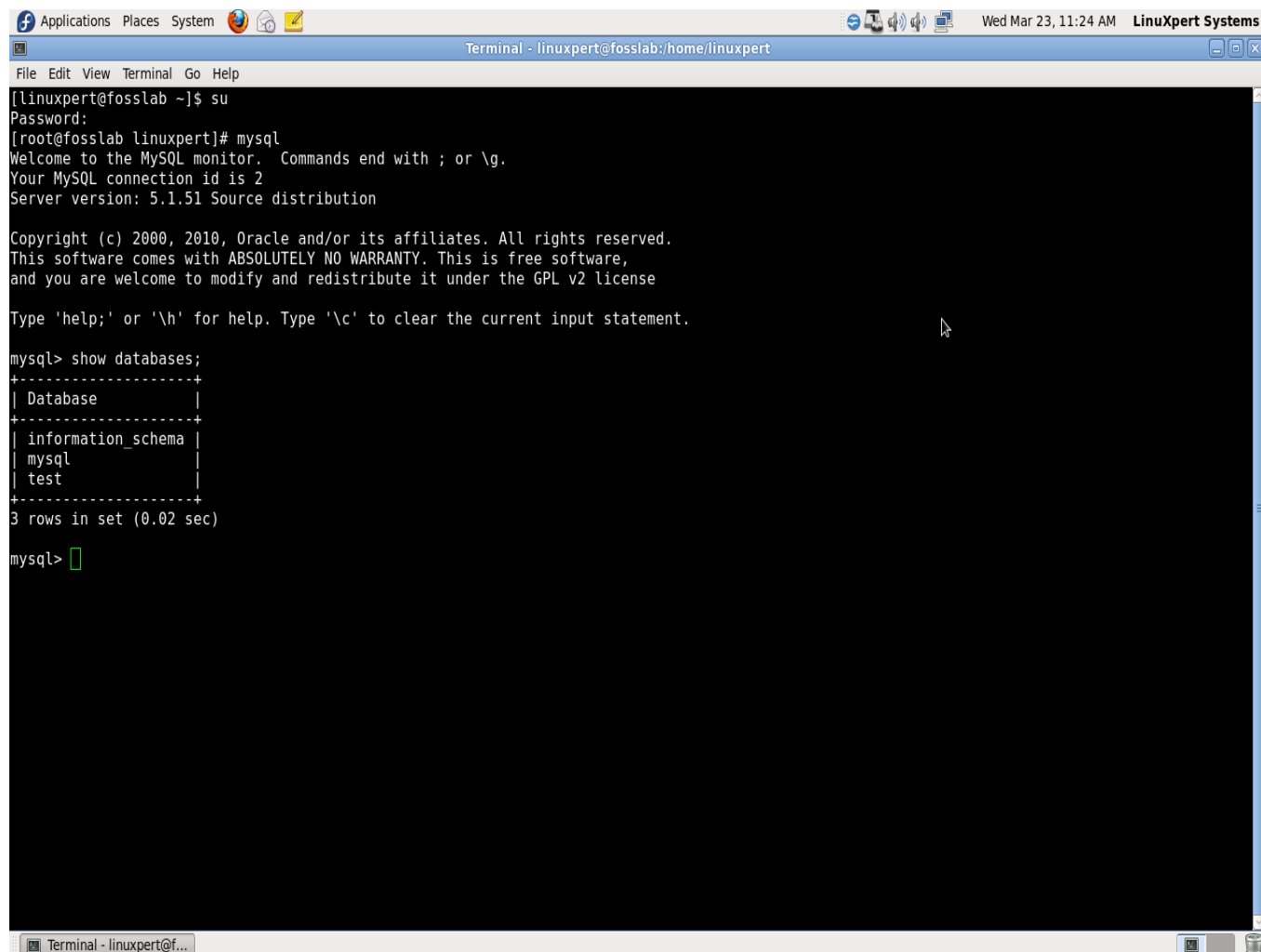
```
[root@server root]# /usr/local/mysql/bin/mysqladmin -u root -p password
NEW_PASS
```

NOTE: When it prompts you to enter the password, just hit Enter
if you don't see an error, the password was successfully changed.

Uninstalling MySQL

```
[root@server root]# rm -rf /usr/local/mysql
[root@server root]# rm -rf /data/mysql
[root@server root]# rm -rf /sbin/mysqld
```


OUTPUT:



The screenshot shows a Linux terminal window titled "Terminal - linuxpert@fossilab:/home/linuxpert". The user is logged in as "linuxpert@fossilab" and has run the command "su" to become the root user. The root user then runs the "mysql" command to start the MySQL monitor. The MySQL monitor displays the following information:

```
[linuxpert@fossilab ~]$ su
Password:
[root@fossilab linuxpert]# mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.1.51 Source distribution

Copyright (c) 2000, 2010, Oracle and/or its affiliates. All rights reserved.
This software comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to modify and redistribute it under the GPL v2 license

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql      |
| test       |
+-----+
3 rows in set (0.02 sec)

mysql>
```

The terminal window also shows the system menu (Applications, Places, System) and the system clock (Wed Mar 23, 11:24 AM) in the top bar.

EX.NO:03

DATE:

APACHE INSTALLATION

Aim:

To install the **apache server** package in the fedora operating system.

Introduction:

The Apache Web server is arguably the most popular Web server in use on the Internet today. Here are some of the reasons why Apache is so popular; you don't have to be running Windows to run Apache. It was developed on various Unix/Linux/BSD platforms, and then recently ported to Win32. Apache has been shown to be substantially faster, more stable, and more feature-full than many other web servers.

Installation Options:

The three methods of installing Apache under Linux. They are

- Binary installation
- Using an RPM (Red Hat Package Manager) – recommended for people running Red Hat Linux
- Building from source

Apache Installation Guide

We recommend using the RPM installation method. This is the easiest way to install Apache for people that are new to Linux.

Download the Software

You can download Apache from the Apache Software Foundation web site located at <http://www.apache.org>, in source and binary forms. While your downloading Apache, you may want to browse over the documentation.

To install Apache, you will need the following things:

1. A computer running Linux
2. Root access on this computer
3. For binary and source installations, the tar and gunzip Unix utilities

Binary Installation

A binary is pre-configured, which means someone else has gone to the trouble of configuring and building the software for you. There are, however, a few things you should keep in mind:

1. Binaries are compiled for a particular operating system. In other words, you must use a binary built specifically for FreeBSD on your FreeBSD machine and a Linux binary on your Linux machine. You need to be sure to grab the correct binary; if you don't see a binary for your particular operatingSystem, you must choose a different method of installation.

2. Apache binaries are usually a version or two behind the current source distribution. This means you don't reap the benefits of the latest bug fixes and feature enhancements.

3. Because binaries are pre-configured, you don't have much opportunity to alter the way the software works. If you're a newcomer, you may not care about this loss of flexibility. Fortunately most Apache binaries include a full source distribution, providing you with the best of both worlds -- play now,

Learn later.

Now let's install a binary. Point your browser at

<http://www.apache.org/dist/httpd/binaries/>

Download the binary for your operating system (in our case, Linux). You'll most likely be presented with a directory containing multiple versions of Apache in various compressed forms. For the purposes of this guide, I'll assume you've downloaded the gzip'd form of the latest 2.0.x Apache binary (currently that's `httpd-2.0.35-i686-pc-linux-rh72.tar.gz`). If there is a README associated with the file you're downloading, you may want to review it for any interesting installation tidbits or possible bugs.

If you can't find a binary for your operating system, choose either the RPM Installation (if you are running Red Hat Linux) or Build from Source method. Now let's uncompress the archive using the handy combination of gunzip and tar. You should replace the "`httpd-2.0.35-i686-pc-linux-rh72.tar.gz`" text below with the name of the gzip'd file you downloaded.

```
gunzip < httpd-2.0.35-i686-pc-linux-rh72.tar.gz | tar xvf -
```

Some of you may be lucky enough to have a version of tar that is capable of taking care of both tasks.

```
tar xvzf httpd-2.0.35-i686-pc-linux-rh72.tar.gz
```

Either way, you should end up with an `httpd-2.0.x` directory, with x being the particular subversion of Apache you downloaded. Move into the newly created directory.

```
cd httpd-2.0.x
```

As of Apache 1.3.11, binary distributions contain an install script called `install.bindist.sh`. If your binary does not seem to contain such an install script, take a look at the `README.bindist` and/or

INSTALL.bindist

Documents for further information; if these documents don't outline a simple installation method, you'll probably want to use one of the other methods. If you've not already done so, you'll need to become root. You can become root by typing su, then the root password. Then go ahead and run the install

script.

```
/install.bindist.sh
```

This command should install the various bits of the Apache distribution into the appropriate locations; the default is usually to install everything under /usr/local/apache (confirm this by consulting your README.bindist and/or INSTALL.bindist documents). That's all there is to it on installing binaries.

2. RPM Installation

Those of you running Red Hat Linux may want to take advantage of Red Hat's RPM ("RedHat Package Manager") system. Almost identical to a binary, an RPM is further customized to play nicely with other RPMs and provide a consistent interface to installing, updating, and removing binaries.

For Linux newcomers or when installing a small standard component, RPMs are simple and reliable. Bear in mind that an Apache RPM may already be installed on your system depending on how Linux was originally installed on your computer. To find out, at the shell prompt, type:

```
rpm -qa | grep apache
```

 If you see something like apache-1.3.9xxx, an Apache RPM has already been installed. You can also type that command typing httpd instead of apache to see if it's installed. If you don't have an Apache RPM, you must obtain one. Red Hat 7.3 ships apache-1.3.23-11.src.rpm in the RedHat/RPMS directory on the installation CD. Or, point your browser at

<ftp://ftp.redhat.com/pub/redhat/redhat-7.3-en/os/i386/RedHat/RPMS> and download it.

If you've not already done so, you'll need to become root. Navigate to the same directory as the .rpm file you obtained, and then type the following command, substituting the name of the .rpm you're using for

Example: apache-1.3.23-11.src.rpm.

```
rpm -ivh apache-1.3.23-11.src.rpm
```

RPM should grind away, displaying its progress with a primitive ##### progress bar. Barring any errors, you're done.

Build from Source

Building Apache from source may seem like a daunting task to newcomers, but the Apache developers have done a wonderful job of making the task about as simple as could be. Just three more commands than a binary installation and you skip the arduous task of figuring out which binary is the right one for your particular operating system. Point your browser at

<http://www.apache.org/dist/httpd/> and download the gzip'd form of the current version of Apache (2.0.36 at the time of this writing). Now let's uncompress that archive using `gunzip` and `tar`. You should replace the `httpd-2.0.36.tar.gz` below with the name of the gzip'd file you downloaded.

```
gunzip < httpd-2.0.36.tar.gz | tar xvf -
```

You should end up with an `httpd-2.0.x` directory, `x` being the particular sub-version of Apache you downloaded. Move into the newly created directory.

```
cd httpd-2.0.x
```

Now we'll use the `configure` and `make` commands to configure, make, and install Apache. If you've not already done so, now would be the time to become root.

```
./configure
```

Your screen should look something like:

```
# ./configure
checking for chosen layout... Apache
checking for working mkdir -p... yes
checking build system type... i686-pc-linux-gnu
checking host system type... i686-pc-linux-gnu
checking target system type... i686-pc-linux-gnu
Configuring Apache Portable Runtime library ...
...
config.status: executing default commands
```

Unless errors were reported (not warnings), your Apache installation is now configured and we can move on. This is where things get a bit ugly.

Make'ing Apache produces screenfulls of output.

```
Make
```

Your screen should look something like:

```
# make
```

Making all in srclib

```
make[1]: Entering directory `/home/ryan/dl/apache_guide/httpd-2.0.36/srclib'
```

Making all in apr

```
make[2]: Entering directory `/home/ryan/dl/apache_guide/httpd-2.0.36/srclib/apr'
```

...

```
make[1]: Leaving directory `/home/ryan/dl/apache_guide/httpd-2.0.36'#
```

Finally, you're ready to install your Apache build.

```
# make install
```

Now Apache is installed.

Starting Apache

Let's take your new Apache installation out for a spin. If you installed Apache using a binary or from scratch, as root, type:

```
/usr/local/apache/bin/apachectl start
```

If you used an RPM, as root, type:

```
/sbin/service httpd start
```

Point your browser at your brand new Web server, <http://localhost/>. If everything worked you should see the default home page.

Customize

Apache uses some rather easy to understand text files for configuration. On a Red Hat system, you'll find them in `/etc/httpd/conf`. Quite a few Linux distributions place them in this same place, but if you can't find such a directory, do a search for "httpd.conf". Once you find these, you've found the main config files. If you're new to Linux, and need help finding this file, here's how you can find it.

1. Login as root

2. Type: `cd /`

3. Type: `find -name httpd.conf`

Now you should see where the file is located. When you move into the directory containing

httpd.conf, you should see these three files:

- httpd.conf – This has the settings for the overall configuration for the server.
- access.conf – This file contains all the security settings for Apache.
- srm.conf – This file contains the MIME definitions and default document names for files on the server.

Restarting Apache

Whenever you make changes to the server configuration files, such as httpd.conf, they won't take effect until the server is restarted. In Linux, Apache can be restarted depending on how you installed it. If you installed Apache using a binary or from scratch, as root, type:

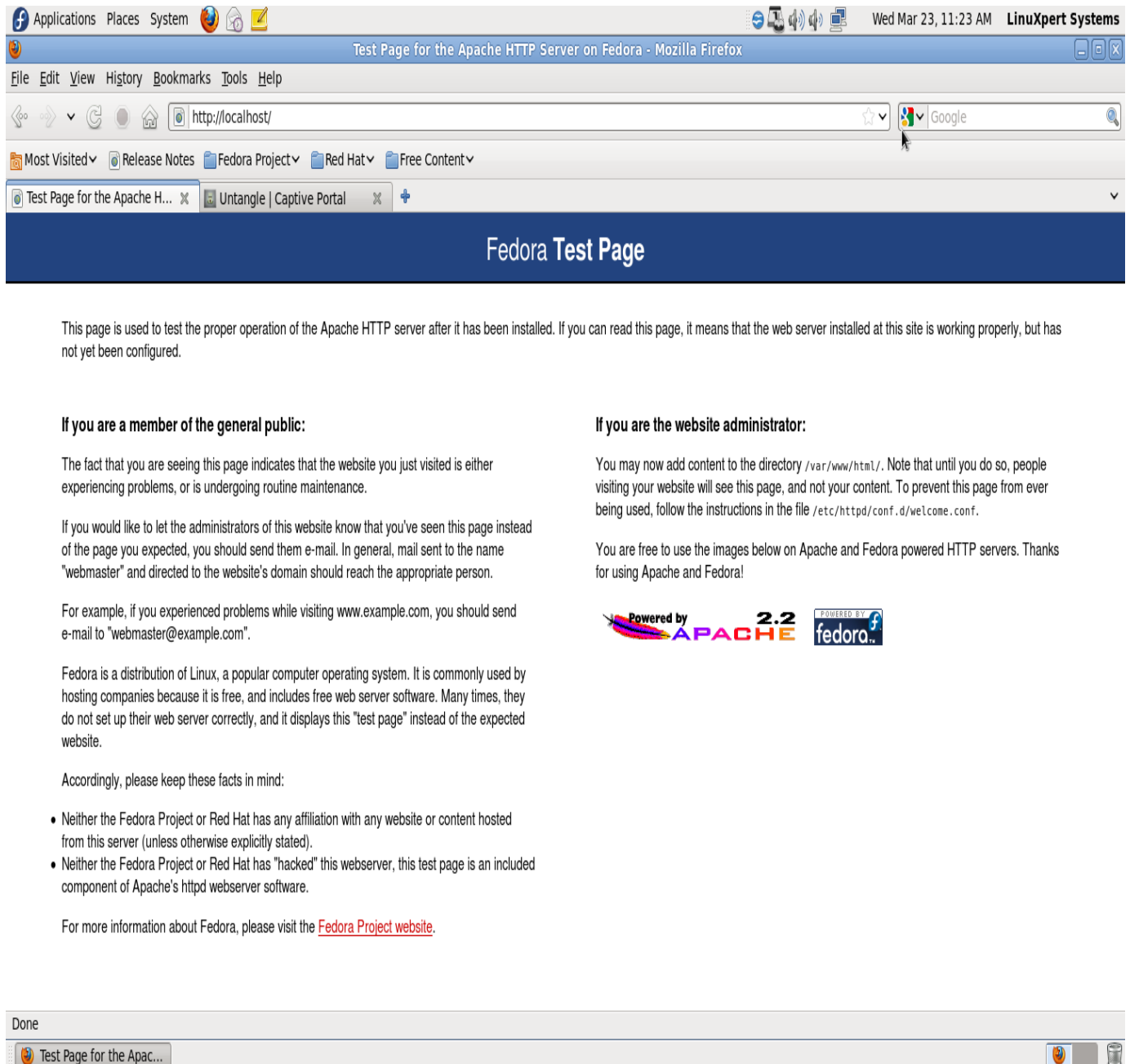
```
/usr/local/apache/bin/apachectl start
```

If you used an RPM, as root, type:

```
/sbin/service httpd start
```

After being restarted the changes will have taken effect.

OUTPUT:



EX.NO:04

DATE:

PHP PROGRAMS

Aim:

To write a

- i. simple programs using php
- ii. mysql database connection program

iii. Creating a web page using php.

i) Date & time display

```
<HTML><HEAD>
<TITLE> PHP PROGRAM</TITLE>
</HEAD>
<BODY><?PHP
$date = Date("l - F d, Y");
print("Hello World<BR />\n");
print("Today's date is: $date\n");
?></BODY>
</HTML>
```

Output:

Run in Mozilla Firefox:

HelloWorld

Today's date is: Monday - March 21, 2011

ii) Conditional statements: (statements)

if-else

```
<?php
if ($a > $b) {
    echo "a is bigger than b";
} elseif ($a == $b) {
    echo "a is equal to b";
} else {
    echo "a is smaller than b";
}?>
```

Output:

a is equal to b

iii) Swapping in php (Variables)

```
<html>
<head><title>test</title></head>
```

```

<body>
<?php
$a=15;
$b=10;
echo"before the values are $a and $b<br>";
$c=$a;
$a=$b;
$b=$c;
echo"after swapping the vaues are  value of a is $a and b is $b";
?>
</body>
</html>

```

Output:

before the values are 15 and 10

after swapping the vaues are value of a is 10 and b is 15

iv)case sensitive:

```

<?php
$phrase = "i love my india";
echo "<br>";strtoupper($phrase);
echo strtoupper($phrase);"<br>";
?>

```

output:

I LOVE MY INDIA

v)reverse.php (Array)

```

<?PHP
$list=array(
    0=>"j",
    1=>"u",

```

```

2=>"m",
3=>"a");
echo "<pre>";
var_dump($list);
echo implode("", $list) . "<br>";
$array1 = array_reverse($list);
echo implode("", $array1);
echo "</pre>";
?>

```

Output:

```

array(4) {
  [0]=>
  string(1) "j"
  [1]=>
  string(1) "u"
  [2]=>
  string(1) "m"
  [3]=>
  string(1) "a"
}
juma
amuj

```

Functionn.php

```

<!DOCTYPE html>
<html>
<body>
<?php
function familyName($fname) {
    echo "$fname Refsnes.<br>";
}
familyName("Jani");
familyName("kumar");
familyName("thiru");

```

```
familyName("bond");
familyName("Borge");
?>
</body>
</html>
```

Operatorss.php

```
<?php
$t = date("H");

if ($t < "10") {
    echo "Have a good morning!";
    $t++;
} elseif ($t < "20") {
    echo "Have a good day!";
    $t--;
} else {
    echo "Have a good night!";
}
?>
```

Xampp code to connect PHP with Mysql

```
<?php
$con =new mysqli("localhost","root","","student");
if (!$con)
{
    die('Could not connect: ' . mysql_error());
}

// Create table

$sql = $con->query( "CREATE TABLE Persons1(FirstName varchar(15),LastName varchar(15),Age
int)");

// Execute query
```

```
echo "Table selected database<br>";
$res=$con->query("INSERT INTO Persons (FirstName, LastName, Age)VALUES ('ss', 'sss', '10')");
echo $res;
echo "Record created";
mysqli_close($con);
?>
```

CONNECTING PHP WITH MYSQL in Linux

```
[linuxpert@localhost ~]$ su
Password: admin123
[root@localhost linuxpert]# cd /var/www/html
[root@localhost html]# gedit
type the following in gedit and save it as form.html
```

program:

```
<?php
$con = mysql_connect("localhost","root","");
if (!$con)
{
    die('Could not connect: ' . mysql_error());
}

// Create database

if (mysql_query("CREATE DATABASE my_db1",$con))
```

```
{  
    echo "Database created<br>";  
}  
else  
{  
    echo "Error creating database: " . mysql_error();  
}
```

// Create table

```
mysql_select_db("my_db1", $con);  
$sql = "CREATE TABLE Persons(FirstName varchar(15),LastName varchar(15),Age int)";
```

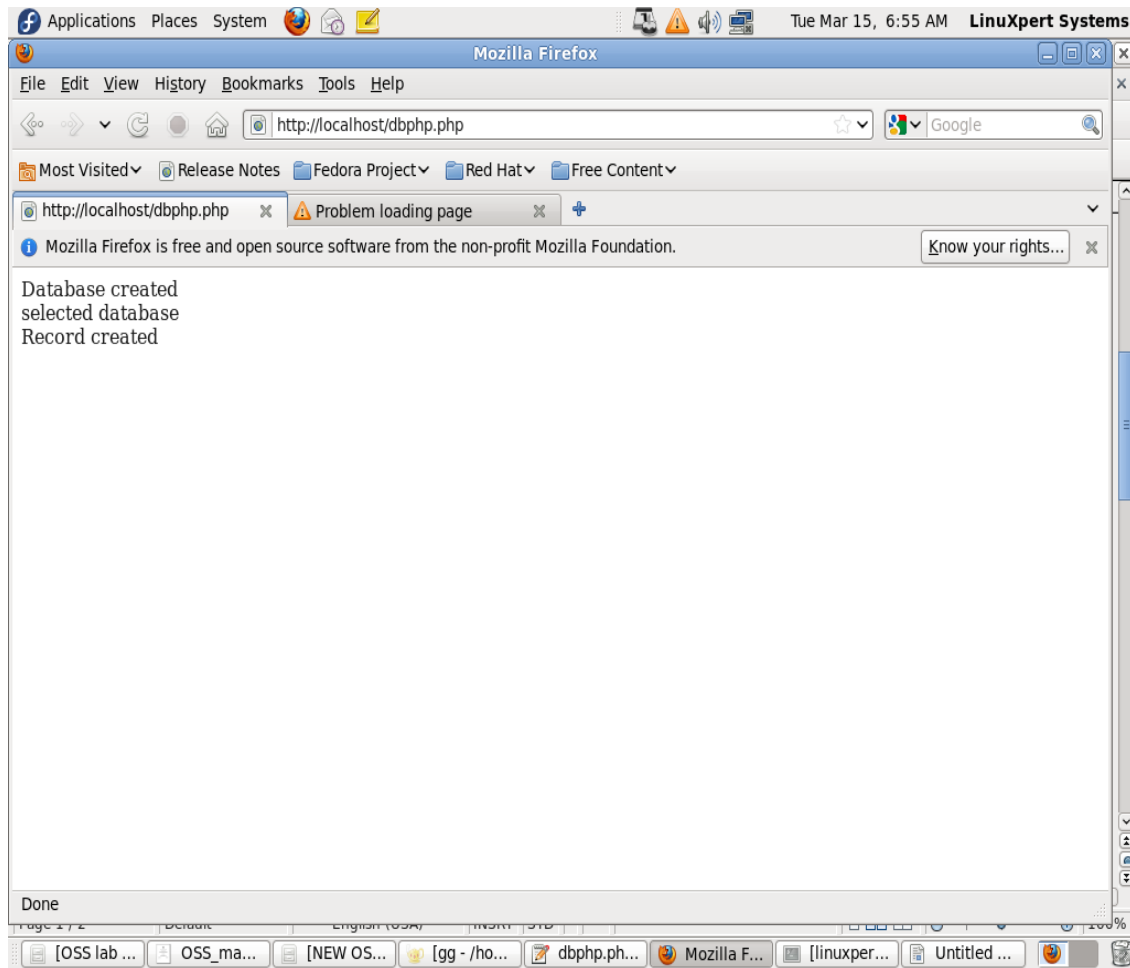
// Execute query

```
mysql_query($sql,$con);  
$db=mysql_select_db("my_db1",$con)or die(mysql_error());  
echo "selected database<br>";  
mysql_query("INSERT INTO Persons (FirstName, LastName, Age)VALUES ('ss', 'sss', '10')");  
echo "Record created";  
mysql_close($con);  
?>
```

TYPE THE FOLLOWING IN THE BROWSER

<http://localhost/dbphp.php>

Output:



```
mysql> show databases;
```

```
+-----+
| Database |
+-----+
| information_schema |
| my_db1 |
| mysql |
| test |
+-----+
```

```
4 rows in set (0.03 sec)
```

```
mysql> use my_db1;
```

```
Reading table information for completion of table and column names
```

You can turn off this feature to get a quicker startup with -A

Database changed

```
mysql> show tables;
```

```
+-----+
| Tables_in_my_db1 |
+-----+
| Persons          |
+-----+
1 row in set (0.00 sec)
```

```
mysql> select * from Persons;
```

```
+-----+-----+-----+
| FirstName | LastName | Age |
+-----+-----+-----+
| ss       | sss     | 10 |
+-----+-----+-----+
1 row in set (0.00 sec)
```

WEBPAGE CREATION

Create Login Page

Create Login Page (HomePage.php)

```
<table width="300" border="0" align="center" cellpadding="0" cellspacing="1" bgcolor="#CCCCCC">
<tr>
<form name="form1" method="post" action="checklogin.php">
<td>
<table width="100%" border="0" cellpadding="3" cellspacing="1" bgcolor="#FFFFFF">
<tr>
<td colspan="3"><strong>Member Login </strong></td>
</tr>
```



```

<tr>
<td width="78">Username</td>
<td width="6">:</td>
<td width="294"><input name="myusername" type="text" id="myusername"></td>
</tr>

<tr>
<td>Password</td>
<td>:</td>
<td><input name="mypassword" type="text" id="mypassword"></td>
</tr>

<tr>
<td>&nbsp;</td>
<td>&nbsp;</td>
<td><input type="submit" name="Submit" value="Login"></td>
</tr>
</table>

</td>
</form>
</tr>
</table>

```

Create Validation Page (checklogin.php)

```

<?php
    $host="localhost"; // Host name
    $username="root"; // Mysql default username
    //$password=""; // Mysql No password
    $db_name="my_db2"; // Database name
    $tbl_name="members"; // Table name
    // Connect to server and select databse.
    mysql_connect("$host", "$username")or die("cannot connect");
    mysql_select_db("$db_name")or die("cannot select DB");

```

```

// username and password sent from form
$username=$_POST['myusername'];
$password=$_POST['mypassword'];
$sql="SELECT * FROM $tbl_name WHERE
username='$myusername' and password='$mypassword'";
$result=mysql_query($sql);
// Mysql_num_row is counting table row
$count=mysql_num_rows($result);
// If result matched $myusername and $mypassword, table row must be 1 row
if($count==1)
    echo "Welcome To Our Web Page";
else
    echo "Wrong Username or Password";
?>

```

database:

```
mysql> show databases;
```

```

+-----+
| Database          |
+-----+
| information_schema |
| my_db1            |
| my_db2            |
| mysql             |
| test              |

```

+-----+

5 rows in set (0.00 sec)

mysql> use my_db2;

Database changed

mysql> create table members(username varchar(10),password varchar(10));

Query OK, 0 rows affected (0.07 sec)

mysql> insert into members values("gokul","gocool");

Query OK, 1 row affected (0.00 sec)

mysql> select * from members;

+-----+

| username | password |

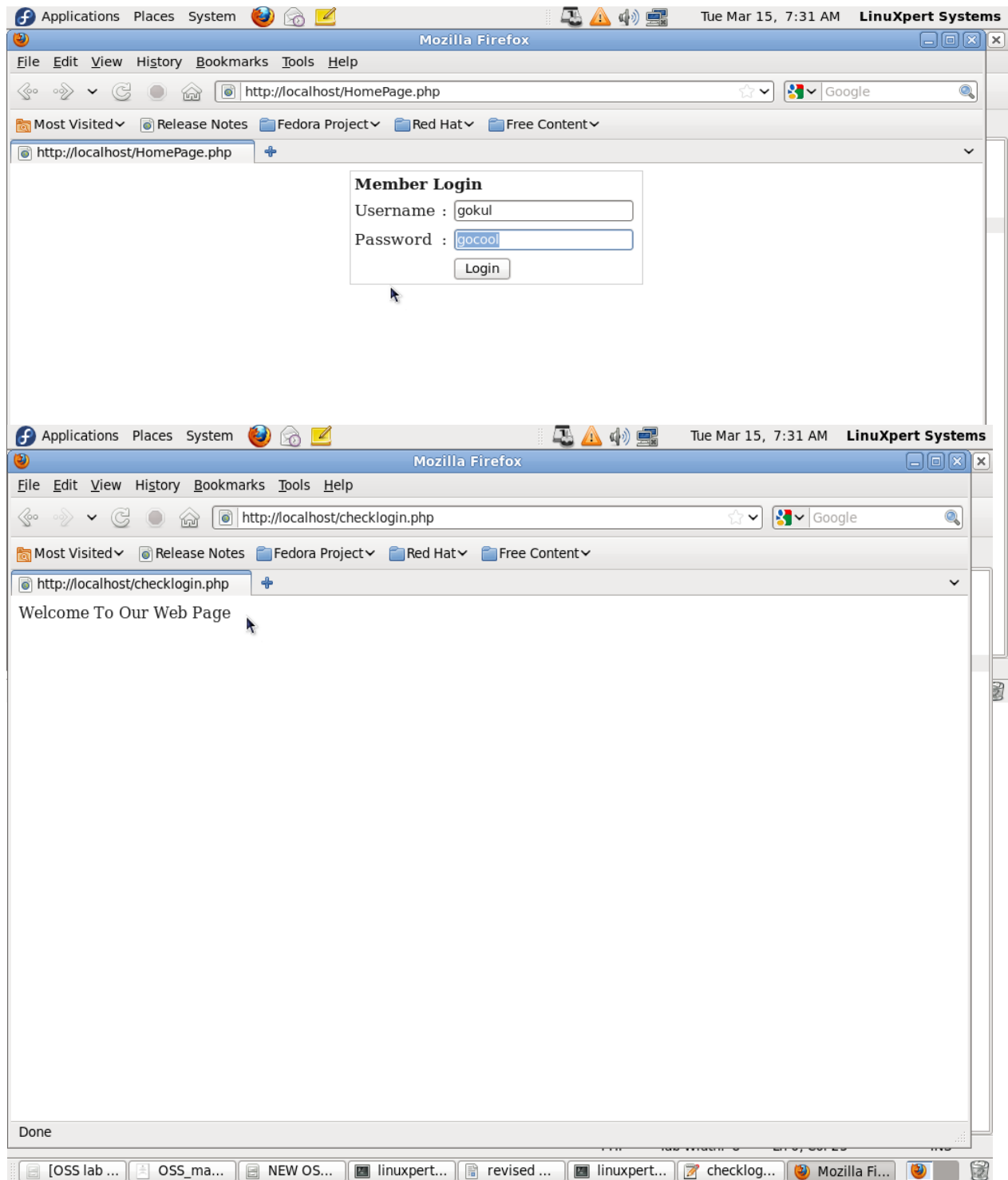
+-----+

| gokul | gocool |

+-----+

1 row in set (0.00 sec)

Output:



EX.NO:05

DATE:

PYTHON PROGRAMS

Aim:

To write a

- i. Simple programs using python
- ii. CGI program using python

SIMPLE PROGRAMS

1.A first example "Hello World"

```
#!/usr/bin/env python
#
# This program says "hello" to the world
print "Hello World!"
```

Output:

```
[root@localhost linuxpert]# python hello.py
```

Hello World!

2.Operators for Numbers

```
#!/usr/bin/env python
#
# A program for numbers
#
a = 3 - 4 + 10
b = 5 * 6
c = 7.0/8.0
print "These are the values:", a, b, c
print "Increment", a, "by one: "
a = a +1
print a
print "The sum of", a, "and", b, "is"
d = a + b
print d
number = input("Input a number ")
print number
```

output:

```
[root@localhost linuxpert]# python no.py
```

These are the values: 9 30 0.875

Increment 9 by one:

10

The sum of 10 and 30 is

40

Input a number 25

25

3.#String concatenation

```
worda='computer';
```

```
wordb='science';
```

```
print("worda is ",worda);
```

```
print("wordb is",wordb);
```

```
wordc=worda+" " +wordb;
```

```
print("wordc is",wordc);
```

```
wordd=worda*3;
```

```
print("wordd is ",wordd);
```

```
str = 'HelloWorld!'
```

```
length=len(str);
```

```
print ("str :",str);
```

```
print("length:",length);
```

```
print ("first character is",str[0]);
```

```
print ("print character from 2nd to 6th :", str[2:7] );
```

```
print ("Prints string starting from 3rd character:",str[2:]);  
print ("Prints string two times",str * 2);  
print ("Prints concatenated string :",str + "TEST" );  
print(str[-1]); #print last character  
print(str[-6]);#print character from last 6th position  
print(str[:-2]);# Everything except the last two characters
```

Output:

```
[root@fossilab linuxpert]# chmod u+x string.py  
[root@fossilab linuxpert]# python string.py  
(worda is ', 'computer')  
(wordb is', 'science')  
(wordc is', 'computer science')  
(wordd is ', 'computercomputercomputer')  
(str :, 'HelloWorld!')  
(length:', 11)  
(first character is', 'H')  
(print character from 2rd to 6th :, 'lloWo')  
(Prints string starting from 3rd character:', 'lloWorld!')  
(Prints string two times', 'HelloWorld!HelloWorld!')  
(Prints concatenated string :, 'HelloWorld!TEST')  
!  
W  
HelloWorld
```

4. Write a python Program to select odd number from the lists

```
#!/usr/bin/python

#program to select odd number from the list

a=[11,12,13,14,15,16,17,18,19,20,21,31,44,45,10];

print("List is:",a);

n=len(a);

print("length:",n);

i=0;

print("Odd number");

for i in range(len(a)):

    if(a[i]%2==1):

        print(a[i]);
```

output:

```
[root@fossilab linuxpert]# vi nos.py
```

```
[root@fossilab linuxpert]# python nos.py
```

```
('List is:', [11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 31, 44, 45, 10])
```

```
('length:', 15)
```

```
Odd number
```

```
11
13
15
17
19
```


21
31
45

5.Prime Number using Python.

```
for n in range(2, 10):
```

```
    for x in range(2, n):
```

```
        if n % x == 0:
```

```
            print n, 'equals', x, '*', n/x
```

```
            break
```

```
        else:
```

```
            print n, 'is a prime number'
```

```
senthilkumar@senthilkumar-System-Product-Name:~$ python ./prince.py
```

OUTPUT:

2 is a prime number

3 is a prime number

4 equals 2 * 2

5 is a prime number

6 equals 2 * 3

7 is a prime number

8 equals 2 * 4

9 equals 3 * 3

6. Fibonacci Series using Python

```
def fib(n): # write Fibonacci series up to n
```

"Print a Fibonacci series up to n"

a, b = 0, 1

while b < n:

print b,

a, b = b, a+b

Now call the function we just defined:

fib(2000)

senthilkumar@senthilkumar-System-Product-Name:~\$ python ./fib.py

1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597

CGI - Common Gateway Interface

1. Save some simple cgi code in your cgi directory

```
#!/usr/bin/env python
```

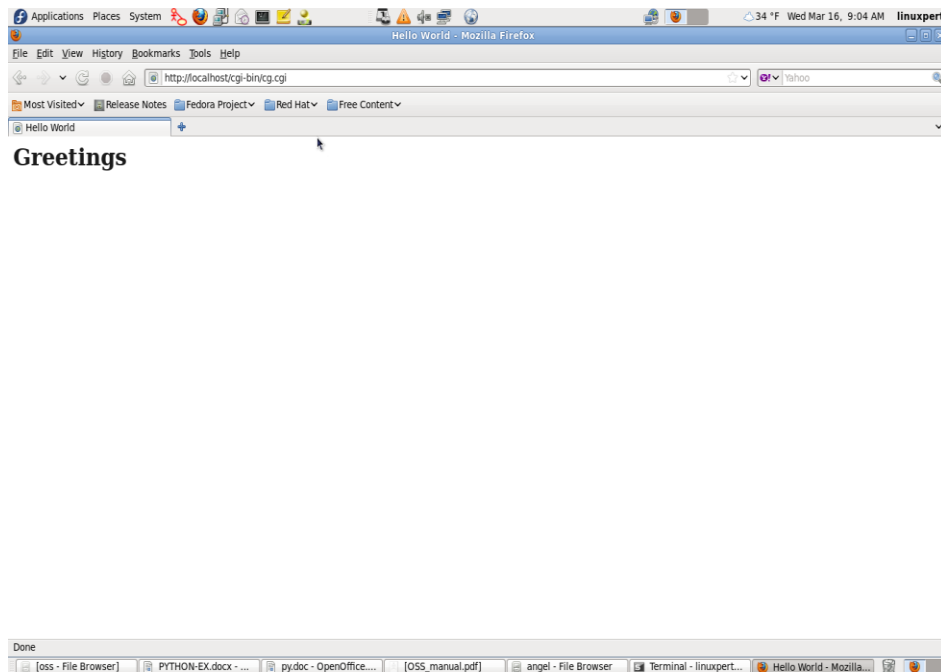
```
import cgi
print "Content-Type: text/html\n"
```

```
print """
<HTML>
<HEAD>
<TITLE>Hello World</TITLE>
</HEAD>
<BODY>
<H1>Greetings</H1>
</BODY>
</HTML>
```

''''''

The file permissions must be set to executable (chmod u+x filename).

Output:



EX.NO:06

DATE:

PERL PROGRAMS

Aim:

To write a

- i. simple programs using PERL
- ii. mysql database connection program using PERL
- iii. CGI program using PERL.

1. Sample PERL

`#!/usr/bin/perl` # the above line is shebang directive

```
$name=<STDIN>;
```

```
chomp($name);
```

```
print "$name\n";
```

output:

```
[linuxpert@localhost ~]$ perl first.pl
gokul
gokul
```

2. Array in PERL

```
#!/usr/bin/perl
my @animal=("cow","Buffalo","Camel");
print "@animal\n"; # list all elements in array
print "$#animal\n"; # list last element position
print "$animal[0]\n"; #list 0th position element
$count=@animal;
print "$count"; # count no of elements in array
```

output:

```
[linuxpert@localhost ~]$ perl array.pl
cow Buffalo Camel
2cow
3
```

3.If loop in perl

```
#!/usr/bin/perl
my $a=10;
$condition=1;
if($condition)
{
my $y=100;
print "$a\n";
print "$y\n";
}
print "$a\n";
print "$y\n";
```

output:

```
[linuxpert@localhost ~]$ perl ifloop.pl
```

```
10
```

```
100
```

```
10
```

4. While loop (until) in perl

```
#!/usr/bin/perl
```

```
$a=0;
```

```
until($a>10) #is equal to while
```

```
{
```

```
print "$a\n";
```

```
$a++;
```

```
}
```

out put:

```
[linuxpert@localhost ~]$ perl unless.pl
```

```
a less than 10[linuxpert@localhost ~]$ perl until.pl
```

```
0 1 2 3 4 5 6 7 8 9 10
```

5. for each loop (upper limit is not fixed)in perl

```
#!/usr/bin/perl
```

```
my @animals=("cow","buffalo","camel",123,100,243,300);
```

```
foreach $key(@animals)
```

```
{
```

```
print "$key\n";
```

```
}
```

output:

```
[linuxpert@localhost ~]$ perl foreach.pl
```

```
cow
```

```
buffalo
```

camel

123

100

243

300

6. String operation:

```
#!/usr/bin/perl
```

```
$a="hello";
```

```
$b="world";
```

```
print $a.$b,"\n";
```

```
$str="-";
```

```
print $str x 80,"\n";
```

```
@a=(10..25);
```

```
print "@a\n";
```

output

```
[linuxpert@localhost ~]$ perl string.pl
```

```
helloworld
```

```
-----/n10 11
```

```
12 13 14 15 16 17 18 19 20 21 22 23 24 25
```

7.Function

```
#!/usr/bin/perl
```

```
sub sayHello()
```

```
{
```

```
print "Hello\n";
```

```
}
```

```
&sayHello();
```

output:

```
[linuxpert@localhost ~]$ perl function.pl
```

Hello

8.program to check greater among 3 number

```
#Greatest among 3 number
```

```
print "Enter A value : ";
```

```
$a=<>;
```

```
print "Enter b value : ";
```

```
$b=<>;
```

```
print "Enter c value : ";
```

```
$c=<>;
```

```
if(($a > $b)&&($a > $c))
```

```
{
```

```
print "A is greater";
```

```
}
```

```
elsif(($b > $c)&&($b > $a))
```

```
{
```

```
print "B is greater";
```

```
}
```

```
else
```

```
{
```

```
print "C is greater";
```

```
}
```

output:

```
[root@localhost ~]# vi big.pl
```

```
[root@localhost ~]# perl big.pl
```

```
Enter A value : 56
```

```
Enter b value : 66
```

```
Enter c value : 85
```

```
C is greater
```

9. DATABASE CONNECTION USING PERL

1. Write the PERL script to connect with mysql as follows

```
#!/usr/bin/perl
use DBI; #to use the build in package we use "Use", DBI is the build in package in perl
my $dbh=DBI->connect("dbi:mysql:student","root",""); #connect to database
if(!$dbh)
{
die("error:$!");
}
$sth=$dbh->prepare("create table students(rollno int,sname varchar(50))");
# create the table
$sth->execute();
$dbh->disconnect;
```

Run the Perl script

```
[linuxpert@localhost ~]$ perl connect.pl
```

now see the tables in database ("student")

```
mysql> show tables;
```

```
+-----+
| Tables_in_student |
+-----+
| students          |
+-----+
```

```
1 row in set (0.00 sec)
```


2. insert the values in perl that will be automatically updated in database using mysql as follows

```
#!/usr/bin/perl

use DBI; #to use the build in package we use "Use", DBI is the build in
package in perl

my $dbh=DBI->connect("dbi:mysql:student","root",""); #connect to
database
if(!$dbh)
{
die("error:$!");
}
$sth=$dbh->prepare("insert into students values(100,'thamarai')"); # create the table
$sth->execute();
$dbh->disconnect;
```

compile the perl

```
[linuxpert@localhost ~]$ perl dbinsert.pl
```

now the output is

```
mysql> select * from students;
```

```
+-----+-----+
| rollno | sname  |
+-----+-----+
| 100    | thamarai |
+-----+-----+
```

```
1 row in set (0.00 sec)
```

10.CGI PROGRAMMING

Program -1

type the following in terminal

```
[linuxpert@localhost ~]$ su
```

Password:

```
[root@localhost linuxpert]# cd /var/www/cgi-bin
```

```
[root@localhost cgi-bin]# gedit
```

type the following in gedit

```
#!/usr/bin/perl
```

```
use CGI;
```

```
$cgi=new CGI;
```

```
print $cgi->header,
```

```
$cgi->start_html,
```

```
$cgi->h1("A simple Example"),
```

```
$cgi->end_html;
```

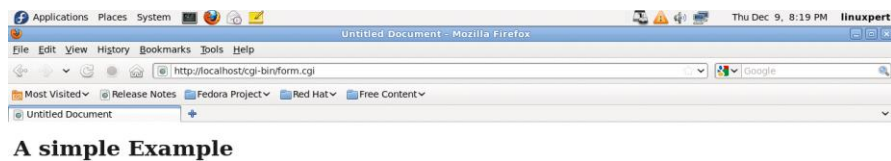
type the following in the same terminal

```
[root@localhost cgi-bin]# chmod +x form.cgi
```

go to browser and type the following URL

<http://localhost/cgi-bin/form.cgi>

OUTPUT:



Program -2

in terminal

```
[linuxpert@localhost ~]$ su
```

Password:

```
[root@localhost linuxpert]# cd /var/www/html
```

```
[root@localhost html]# gedit
```

type the following in gedit editor

```
<html>
<head>
<title>LOGIN</title></head>
<body>
<form action="/cgi-bin/form2.cgi" method="post">
<p>

"Enter student roll no"<input type="text" name="rollno"></p>
<p>"enter the student name"<input type="text" name="sname"></p>
<p>"click here to submit"<input type="submit" name="submit"></p>
</form>
</body>
```

save the page as form.html & close it

type the following URL in browser

<http://localhost/form.html>

then type the following in the terminal

```
[root@localhost html]# cd /var/www/cgi-bin
```

```
[root@localhost cgi-bin]# gedit
```

type the following in gedit

```
#!/usr/bin/perl
use CGI;
$cgi=new CGI;
use DBI;
$rollno=$cgi->param('rollno');
```

```
$name=$cgi->param('sname');  
my $dbh=DBI->connect("dbi:mysql:student","root","");  
my $sth=$dbh->prepare("insert into students values(?,?)");  
$res=$sth->execute($rollno,$name);  
$dbh->disconnect;  
  
if($res)  
{  
print $cgi->header,  
$cgi->start_html,  
$cgi->h1("Record created"),  
  
$cgi->end_html;  
}
```

save that file as form2.cgi

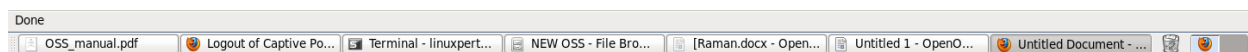
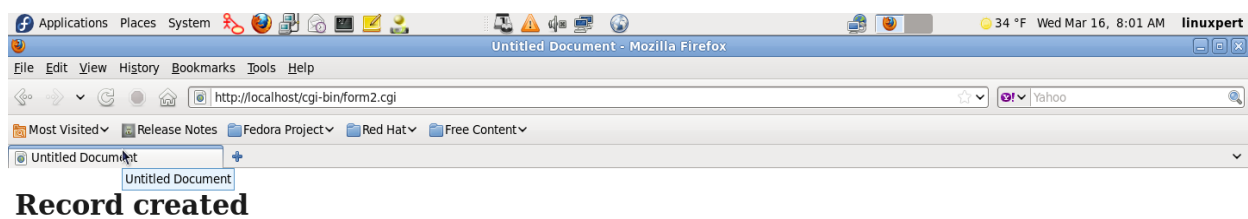
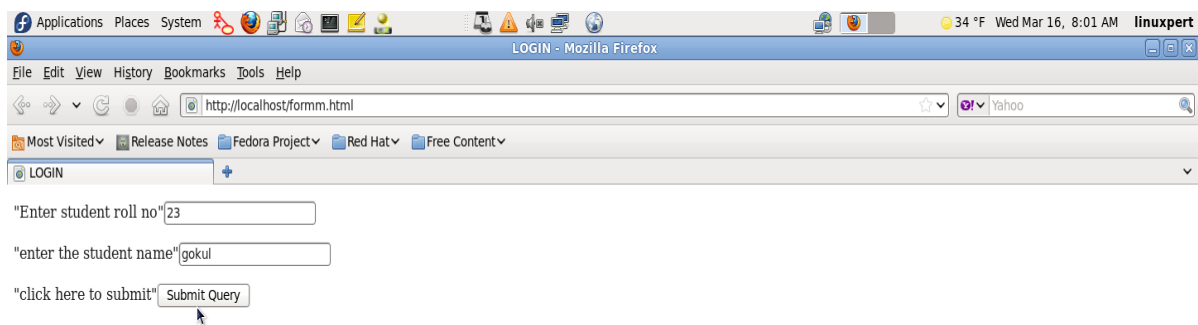
close that file & open a same terminal type as

```
[root@localhost cgi-bin]# chmod +x form2.cgi
```

In new browser type the following

<http://localhost/form.html>

OUTPUT:



mysql> use student

Reading table information for completion of table and column names

You can turn off this feature to get a quicker startup with -A

Database changed

```
mysql> show tables;
```

```
+-----+
| Tables_in_student |
+-----+
| student          |
| students          |
+-----+
```

2 rows in set (0.00 sec)

```
mysql> select * from student;
```

```
+-----+-----+
| rollno | name  |
+-----+-----+
| 200    | selvi |
+-----+-----+
```

1 row in set (0.01 sec)

EX.NO:07

DATE:

NETWORK SIMULATOR -2

Aim:

To install the network simulator in the fedora operating system .

Steps:

NETWORK SIMULATOR-2 is designed to run from on most UNIX based operating systems.Ns is a discrete event simulator targeted at networking research. Ns provides substantial support for simulation of TCP, routing, and multicast protocols over wired and wireless (local and satellite)networks.

Download a copy of ns-allinone-2.34.tar.gz. Then from the command prompt there, execute the following:

```
tar -xzf ns-allinone-2.34.tar.gz
```

```
cd ns-allinone-2.34
```

```
./install
```

(If this fails, try the Distribution Specific instructions)

After a long wait and a whole lot of text, you should see the installation finish up with text like the following:

Nam has been installed successfully.

Ns-allinone package has been installed successfully.

Here are the installation places:

```
tcl8.4.11:    /home/pcraven/ns-allinone-2.29/{bin,include,lib}
```

```
tk8.4.11:     /home/pcraven/ns-allinone-2.29/{bin,include,lib}
```

```
otcl:        /home/pcraven/ns-allinone-2.29/otcl-1.11
```

```
tclcl:       /home/pcraven/ns-allinone-2.29/tclcl-1.17
```

```
ns:          /home/pcraven/ns-allinone-2.29/ns-2.29/ns
```

nam: /home/pcraven/ns-allinone-2.29/nam-1.11/nam

xgraph: /home/pcraven/ns-allinone-2.29/xgraph-12.1

gt-itm: /home/pcraven/ns-allinone-2.29/itm, edriver, sgb2alt, sgb2ns,

sgb2comns, sgb2hierns

Please put /home/myusername/ns-allinone-

2.29/bin:/home/myusername/ns-allinone-

2.29/tcl8.4.11/unix:/home/myusername/ns-allinone-2.29/tk8.4.11/unix

into your PATH environment; so that you'll be able to run

itm/tclsh/wish/xgraph.

IMPORTANT NOTICES:

(1) You MUST put /home/myusername/ns-allinone-2.29/otcl-1.11,
/home/myusername/ns-allinone-2.29/lib, into your LD_LIBRARY_PATH environment variable.

If it complains about X libraries, add path to your X libraries into LD_LIBRARY_PATH.

If you are using csh, you can set it like:

setenv LD_LIBRARY_PATH <paths>

If you are using sh, you can set it like:

export LD_LIBRARY_PATH=<paths>

(2) You MUST put /home/myusername/ns-allinone-2.29/tcl8.4.11/library into your TCL_LIBRARY environmental variable. Otherwise ns/nam will complain during startup.

(3) [OPTIONAL] To save disk space, you can now delete directories

tc18.4.11

and tk8.4.11. They are now installed under /home/myusername/ns-allinone-2.29/{bin,include,lib}

After these steps, you can now run the ns validation suite with `cd ns-2.29; ./validate`

For trouble shooting, please first read ns problems page

<http://www.isi.edu/nsnam/ns/ns-problems.html>. Also search the ns mailing list archive for related posts.

At this point, you should follow the advice here and update your environment variables. You should also add ns-allinone-2.29/bin to you path. This has links to all the executables created by NS-2. Since the Tcl scripts may call these executables (like nam or xgraph), it is a good idea to have them in the path.

You can test the installation by doing the following:

`cd ns-2.29`

`./validate`

Note that this validation takes a really long time. If it starts out ok, you probably have a good installation.

At this point, you can see Getting Started with NS-2 .

OUTPUT:

