List of Contents

LESSON: 1

BASIC CONCEPTS OF HTML

STRUCTURE

1.0 OBJECTIVE

Internet has changed the way of doing the business today. And HTML is the basic language with the help of which web pages and websites can be developed for internet.

After going through this lesson, you will be able to:

- Describe the concept of HTML
- Identify the Benefits of HTML

- Define the limitations of HTML
- Illustrate the phases of website development
- Define the structure of HTML Tags

1.1 INTRODUCTION

Internet has changed the way in which business is being done today. It has changed the world into large virtual market place where customer can place order and buy things from his computer terminal. It means he has access to entire world shopping market it doesn't matter where he is.

For a manufacturer to spread his business online he has to:

- Develop a website
- And publish it to be open to the public

Web Site

Website is collection of web pages, which are interlinked with each other. The Web pages are designed independently and then these are linked in required sequence to provide logical structure to the Website. The first page of a web site known as Home Page is designed to welcome the reader who has logged on the site. Home page development is necessary for all Web sites. Home Page is like a menu card at restaurant, which lists all the items available at the restaurant. Same purpose is served by Home Page, which provides labeled links to the main web pages.

Web Page

A Web Page is a document which serves as basic unit of information. It contains text information, sound, image, animation and even video. Web page can also contain links to other pages stored anywhere on the web. There is no restriction on the size of Web site. If the size of Web site is large, then it can be placed over number of servers. One Web site can have cross-links to the files of another Web site means sharing of files.

Accessing Web

Page access to a Web Page is made through a special type of software known as Browser. There are several software which makes access to Web site available in the market. The most popular ones are Netscape Navigator and Internet Explorer.

1.2 WEB SITE DEVELOPMENT

The layout of the Web site should be such that information can be quickly and easily accessed. “Beauty lies in the eyes of beholder” This means that every person has his own opinion about beauty. Same is the case with Web pages. While there is no prefect and concrete layout that will appeal to all readers but still there are some guidelines that must be followed while designing web pages. Developer of Web site has to play three roles:

- **Visualiser**

  Its job is to maintain the attractiveness of Web page

- **Architect**

  To provide simple and structured navigational model so that any one can access any section

- **Librarian**

  To ensure that contents are arranged in such a way that reader does not face any difficulty to locate any piece of information on Web Site.

Phases of Development of Web Site

1.21 Design Phase

a) Goals and objectives

Developer must identify the objective of web site, why it is needed and what purpose is served by Web site.

b) Target Audience

Who are going to catch your Web site? What are their interests, their tastes and why they are visiting to your Web site? For e.g., corporate sites and Entertainment sites will have different looks depending on their target audience.

c) Storyboarding

Storyboard is a description of layout, contents, sequence of web pages, which defines the conceptual design of Web Site. Never expect the reader to read Website as novel. Reader must be able to jump to the point of his own interest.

d) Organize Information

Navigation structure is decided by organizing information. Four basic steps of organizing information are:

- Divide it into logical units
- Create a hierarchy of important and general topics.
- Use the hierarchy to build relationship among logical units of information
- Analyze the usefulness and aesthetics of the Website

1.22 Development Phase

Two important factors are considered while designing a Web page and then Website:

- The layout of Web page and Website
- Navigation around the Website

1.22.1 Layout of Web page

a) Unified face

Everything in Web page and Website must fit together as a recognizable whole. There must be consistency among appearance of all Web pages of Website. Fonts, Colours, Margin, Headings and other elements should be consistent throughout every section of Website.

b) Conform to Objective

Objective of Website is to give information to the readers so messages should be properly conveyed and avoid too many fonts and sharp colours which overshadow message

c) Transition Effects

Effects used in the Website depend upon the message to be conveyed for e.g.: Site related to children is colourful with lots of animation but corporate site is always given a sophisticated look

d) Using Colour

Colours should be chosen according to the purpose of site for eg: use of bright colours is always avoided in the corporate sites but in entertainments sites use of bright primary and secondary colour is always preferable.

e) Background

Background of the page should not be overbearing, it will distract the reader from the content on the page. A well-chosen colour background can give a unique identity to the site.

f) Text

- Text information should be well laid out.
- Use commonly available fonts and provide a smooth look, which does not raise any confusion or fatigue in the mind of readers.
- Avoid capitalize words. Use Bold, Italic and Underline features wherever it is necessary to highlight some information.
- Bulleted list can be used to provide logical structure to information

1.22.2 Navigation around the web site

If web site is attractive but does not provide simple navigation structure then it has to loose its readers. There are several ways to organize web site:

a) Hyperlinks

The word which is underlined and when mouse pointer is placed over this word then its shape is changed to hand pointer known as hyperlink which jumps to another Web page when it is clicked.

b) Navigation or Slide Bar

This bar should be placed on all Web pages other than Home page, which has links to the main pages of Web site means user, can jump to any document from any sub page.

c) Table of Contents

It is displayed on the bottom of the Web page, which has the listing of all-important topics. User can click on any topic to visit to related Web Page.

d) Hyper Tree

It is hierarchical organization scheme, which looks like left pane of Windows Explorer.

e) Image Map

Visual representations are used as a means of link to another Web pages. By clicking on the hot spot of image area, Reader can visit to related Web page.

1.3 BASIC OF HTML

Earlier, the documents were recorded with pen and paper. These documents were formatted by following some pre-defined mark up instructions. For example, to provide extra information or highlight information different colours or quotations marks were used. Highlighting text using Bold or Italic feature, adding quotations marks or colour to document, to add special meaning to it, is referred as markup.

A markup language defines some set of rules and helps to add some meaning to content and structure of documents. Markup can be classifieds as follows:

- Stylistic Markup This markup defines presentation of document. Word processor applying stylistic markup means making the text bold, italicized or changing the font.
- Structural Markup The structure of a document is determined by this markup. It determines the heading or paragraph in a document.

- Semantic Markup The content of document is determined by this mark up. Title of Web page defines the content to be described in Web page.

In the late 1960's, three researchers at IBM began working on the problem of dealing with documents created on computers of different hardware and operating systems. During the research, three primary requirements were considered essential in order to have an interoperable system:

- There should be common document format supported by document processing program.
- The common format should be specific to particular domain, for Example, a domain for legal documents, a domain for medical documents.
- There should be specific rules for the format of document.

The system of formatting documents was named as Generalized Markup Language (GML). Over the Years, GML was fine-tuned and came to be known as Standard Generalized Markup Language (SGML).HTML or Hyper Text Markup Language has evolved from SGML.HTML is used to develop formatted pages of the web known as Web pages. It is specialized language, designed to display and access Web pages.

1.4 What are HTML Tags Like?

When browser displays a Web page, it reads the page text and looks for special symbols indicating tags. These tags specify how the text should be displayed. For example, a tag may specify that text should be displayed in bold or italics or underlined with a certain font. Besides the appearance of text, a tag can indicate that certain text is really the address of another Web page, which will be accessed when that text is clicked.

Tags are normally specified in pairs, delimiting the text that will have some type of formatting. But there are several individual tags also.

Tags are identified by the < > or </> signs. In the case of tags, which need to enclose the text, the closing tag should include the slash (/) to indicate the end of tag. Tag names are not case sensitive.

Syntax of a tag is < Tag name> text</Tag name>

Need to Be Connected to the Web to Use the Browser

No. The browser work independently. The best way of developing a web page is to work offline .All the pages are created and stored on hard disk and when logical structure of Website is over then place all the Web pages on a server to be open to public to have access.

1.5 STRUCTURE OF AN HTML PAGE

The skeleton of HTML page is

<HTML>
<HEAD>
<TITLE>
Page Title
</TITLE>
</HEAD>
<BODY>
…….
……
</BODY>
</HTML>

These commands are mandatory. The <HTML> command is used in tandem with </HTML> command. It delimits the area of the HTML language commands. The <HEAD> </HEAD>pair of commands is used to specify few of language commands such as <TITLE> and <META> commands.

The <BODY></BODY> pair of commands should be used to enclose all of page formatting commands.

The Basic Structure of an HTML Document

Like other languages, HTML has a basic structure for its programs. In order for a browser to correctly interpret the program, it needs to have some basic commands that should always be present. Some browsers dispense with their use; however, it is better to assume such commands are a fundamental part of the program.

An HTML program has three basic parts:

- Main structure (<HTML></HTML>)
- Heading (<HEAD></HEAD>)
- Body (<BODY></BODY>)

Every HTML program must start with the <HTML> command and end with the </HTML> command. This pair of commands is essential. The heading area is optional and is delimited by the <HEAD></HEAD> pair of commands. Most of the HTML commands will be placed in the program body area delimited by the <BODY> and </BODY> commands.

1.6 HTMLTAGS

- **Headline Tags**

The <H> Command

Headings are document lines that have a letter size different from the rest of the text, with the purpose of identifying the start of a section or topic. There are six sizes, or levels, of headings. The <Hn> command has the following syntax:

<Hn>Heading text</Hn>

Where a number from 1 to 6 should replace “n”. The largest heading is specified with the H1 command, and the smallest with H6.

For example: <H1>this is largest heading</H1> To use the rest of the Headline Tags, the respective Headline tags like H2 and so on will replace H1

Attribute for Heading Tag :

“ALIGN” attribute of <Hn> tag is used to align the heading on the web page. The Syntax is:

<H1 ALIGN=”…”>the text to be placed here</H1>

The possible values for this attribute are:

- Center
- Left
- Right

➢ **Paragraph formatting tags**

The <P> Command

The main difference between the HTML page and the traditional editor is that HTML does not recognize the Enter key. You need to force the end of the paragraph and the line break by using special commands. The command responsible for the paragraph break is the <P> command. Its syntax is:

<P>

The paragraph text should be added here within the <P> tag.

</P>

The paragraph tag by default will display a blank line both on the top and the bottom of the paragraph.

Attribute for Paragraph Tag :

“ALIGN” attribute of <P> tag is used to align the paragraph on the web page.

The syntax is:

<P ALIGN=”VALUE”>

The possible values are:

- Center
- Left
- Right

➢ **The
 command**

The
 command inserts a blank line immediately after it is specified. The syntax is:

➢ **Font Settings Tag**

One of the most interesting text treatment features of HTML language is the ability to change the size, colour and type of text font used. This purpose is served by <FONT> tag.

The syntax is:

<FONT> …</FONT>

Attributes are as follows :

- FACE attribute of the <FONT> tag is used to set the font of the text The syntax is: <FONTFACE= “VALUE”> text </FONT>
- The SIZE attribute of the <FONT> is used to specify the size of the text. The syntax is: <FONT SIZE= “VALUE”> text </FONT>
- COLOR attribute of the <FONT> is used to specify the color of the text.

The syntax is:

<FONT COLOR= “VALUE”>
Text
</FONT>

HTML code to incorporate Break Tag, Font Tag, Paragraph Tag and its attributes.

<HTML>
<TITLE> Use of tags</TITLE>
</HEAD>
<BODY>

This two break tags will display two blank lines at the beginning of document.
<P>
<FONT FACE= “TIMES NEW ROMAN” SIZE=6 COLOR= “GREEN”>
This text will be displayed in Times New Roman font of size 6 with color green</FONT>
<P>
<FONT FACE= “ARIAL” SIZE=4 COLOR= “RED”>
This paragraph tag will display blank line above and below paragraph and font tag causes the paragraph matter to display in red color of size 5 with “Arial” font.
</FONT>
</P>
</BODY>
</HTML>

- **Text Styles**

As in the text editor, you can create a series of effects in the text with HTML by changing the form and type of font. All the commands that change the text style are of the container, or on-off, type.

The main text style commands are:

- <B>command: It applies the bold style on text.
  Syntax is: <B>text</B>
- <I>command: It applies the italic style on text.
  Syntax is: <I>text</I>
- <U>command: It underlines the text.
  Syntax is: <U>text</U>
- <Strong>command: It applies the bold style on text.
  Syntax is: <Strong>text</Strong>
- <TT>command: It puts text in a typewriter-style font with fixed spacing.
  Syntax is: <TT>text</TT>
- <BIG>command: It increases the font and applies bold on text.
  Syntax is: <BIG>text</BIG>
- <Small>command: It reduces the font.
  Syntax is: <Small>text</Small>
- <SUP>command: Superscript command raises and reduces the text.
  Syntax is: ^{text}
- <SUB>command: Subscript command lowers and reduces the text.
  Syntax is: _{text}
- <Blink>command: it makes the text intermittent.
  Syntax is: <Blink>text</Blink>
- <Strike> command: it can be used to display the text in the browser, which has been strike and is normally used to highlight the text.

HTML code to incorporate all Text Styles described above.

```
<HTML>
<HEAD>
<TITLE>Text Style Commands</TITLE>
</HEAD>
<BODY> This is in <B>Bold.</B>
<BR>
 This is in <I>Italics.</I>
<BR>
This is in <U>Underlined.</U>
<BR>
 This is in <B>><I><U>Bold, Italics and Underlined.</B></I></U>
<BR> This text is in <STRONG>Strong style which is similar to Bold.</STRONG>
<BR>
This text received with <TT>Typewriter style.</TT>
<BR>
 This text will be displayed with <BIG> Big font format </BIG>
<BR>
 This text will be displayed with <SMALL>Small font format.</SMALL>
<BR>
<BR>
This text is called <SUP>Superscript.</SUP>
This text is called <SUB>Subscript.</SUB>
Here is the <BLINK>Blinking Text</BLINK>
<BR>
This text will be displayed with <STRIKE>features</STRIKE>
<BODY>
<HTML>
```

- **Preformatted Text Display**

With the <PRE> command, you can include text, which was edited by a text editor, and preserve the original text formatting, such as tabulation marks, end-of-line generated by the Enter key, and other formats.
The syntax is:

<PRE>text
text
text
</PRE>

HTML code to incorporate PRE tag.

<HTML>
<HEAD>
<TITLE>Preformatted Text display</TITLE>
</HEAD>
<BODY>
<PRE>
This tag is used to display user defined formatting, as he wants

Enter key works with Pre tag and it preserves the original text formatting

* *
* *
****
* *
* *
End of preformatted area
</PRE>
</BODY>

</HTML>

- **<DIV> Tag**

The DIV tag encloses a section, which can receive specific alignment parameters. The Syntax is: <DIVALLIGN=”VALUE”> Text </DIV> The possible values for the DIV tag are:

CENTER
LEFT
RIGHT

The following example uses the <DIV> command to align a group of paragraphs on right without having to use a <P ALIGN> command for each one of them:

<HTML>
<HEAD>
<TITLE> Use of DIV command</TITLE>
</HEAD>
<DIV ALIGN=”RIGHT”>

This command will align all the enclosing paragraphs on right side. There is no need to give paragraph tag with align feature for each paragraph. It is used when all the paragraph of page have same alignment features.

This command will align all the enclosing paragraphs on right side. There is no need to give paragraph tag with align feature for each paragraph. It is used when all the paragraph of page have same alignment features.

</DIV>

This command will align all the enclosing paragraphs on right side. There is no need to give paragraph tag with align feature for each paragraph. It is used when all the paragraph of page have same alignment features.

</BODY>
</HTML>

- **Multicolumn text**

The MULTICOL tag places the text of the document into multiple, equal-width columns. The syntax is:

<MULTICOL
COLS=”VALUE”
GUTTER=”VALUE”
WIDTH=”VALUE”
>
text
</MULTICOL>

The various attributes of the <MULTICOL> tag are:
COLS
Specifies the number of text columns for the text.

GUTTER
Specifies the distance between the text columns, by default it is 10 pixels.

WIDTH
Specifies the width of each column and the width of each column should be the same. For Example:

<MULTICOLCOLS=3,GUTTER=5,WIDTH=30>

Text in the document

</MULTICOL>

This source code divides the screen into three columns separated at the distance of 5 pixels

- **Horizontal line**

Horizontal lines can be added to the HTML document using the <HR> tag. This tag does not have any closing tag.

The syntax is:
<HR>
Paragraph specification
<HR>
The code above inserts a line above and below the paragraph. Attributes of <HR> tag:

- Width

The width of the line can be specified in terms of percentage of the window width or in terms of no of pixels.

<HR WIDTH = “VALUE”>

For Example:
<HR WIDTH = “50%”>

Here width of line is expressed in a percentage relative to the width of window. In this case, its size varies as a function of window width.

<HR WIDTH= “200”>

Here width of line will be fixed of 200 pixels and it won’t change as we resize window.

- SIZE

  The thickness of the line is expressed in terms of pixels.

  <HR SIZE = “VALUE”>

  For Example: <HR SIZE = “20”>

➢ **Body Attributes**

BGCOLOR: This attribute specifies the background color.

For example: <BODY BGCOLOR = “GREEN”>

Background of Window will be green in colour.

TEXT : This attribute specifies the colour of text.

For example: <BODY TEXT = “YELLOW”>

The entire text in Window will be yellow in colour.

BACKGROUND: This attribute places an image as a background of page.

For example: <BODY BACKGROUND= “Hlpcd.GIF”>

Entire background will be filled with this image

1.7 BENEFITS OF HTML

The benefits of HTML are:

- HTML is a simple but powerful formatting language to use.
- The Web pages can be linked together using links. Hence controlled navigation is possible
- HTML documents are platform independent.

1.8 LIMITATIONS OF HTML

- Data interchange is not possible. HTML does not have any programming capabilities and cannot provide anything more than formatted text, pictures and sound, which demands for other programming and scripting languages, to be along with it such as Perl, Java Script and Java.
- HTML's Presentation technology does not provide any information about the content and it has fixed tags. An HTML tag does not provide information about the content with in it. For example, consider the following command

  <LI> SOCT Cruise

  This tag does not give information about the data 'SOCT Cruise'. From this we cannot say whether 'SOCT cruise' is the name of a ship or person or Hotel. It only specifies how the data should be displayed, namely as List Item. It also does not provide the facility of creating customized tags that can be understood by others.
- HTML is flat

  In HTML the only formatting supported are paragraphs, sections and simple structure. The hierarchy of the data cannot be determined since the importance of tags cannot be specified.
- Clogging

  Applications in HTML clog up the network with high volume of traffic. For example, we have to send a large volume of record sets across the net, when only a small amount of information is required. The limitation of HTML of not providing information about the content makes generalized processing and retrieval of the file, difficult.
- Robust linking mechanism is not there

  The links in HTML takes the user from one resource to another but they do not provide the user any options to select them. If location of any link is changed then link has to be updated manually otherwise the Web suffers from broken links.

1.9 SUMMARY

The emergence of the Internet through out the world has been contributing such a variety medium in doing business as well as people lifestyle. In fact, Internet is the essential prerequisite for the existence of E- commerce. The explosion of Internet has created new phenomena in our lifestyle especially in shopping activities. Consumers can easily buy products or services like magazines and airlines tickets through website via Internet.

For a manufacturer to spread his business online he has to:

- Develop a website
- And publish it to be open to the public

The layout of the Web site should be such that information can be quickly and easily accessed. “Beauty lies in the eyes of beholder” This means that every person has his own opinion about beauty.

HTML language is used to develop formatted pages of the web known as web pages. It is specialized language, designed to display and access web pages.

1.10 KEYWORDS

HTML: Hyper Text Markup Language

SGML: Standard Generalized Markup Language

WEB SITE: Website is collection of web pages, which are interlinked with each other.

WEB BROWSER: It is a software that is required to surf the internet.

GUTTER: Specifies the distance between the text columns,

BGCOLOR: Background color

1.11 SELF ASSESSMENT QUESTIONS

1. Define HTML. Why HTML is required
2. Discuss the benefits and limitations of HTML.
3. What do you mean by Tag.? Discuss the syntax and meaning of the following tag:
    a) <H1>

b) <font>

c) <body>

<

4. Differentiate between
 and <p> tag with example.
5. Define web page and web site. Explain the steps of website development.

1.12 SUGGESTED READINGS

Ivan Bay Ross, HTML DHTML JAVASCRIPT PERL and CGI, BPB Publication.

Jose A.Ramalho, Advanced HTML 4.0 with DHTML, BPB Publication.

****

LESSON: 2

WORKING WITH LINKS, IMAGES, AUDIO AND VIDEO

STRUCTURE

2.0 OBJECTIVE

After going through this lesson, you will be able to:

- Describe how to link documents
- Lean how to place image, audio and video files in web pages
- Define the syntax of tags related with hyperlink, graphics, audio and video files,

2.1 INTRODUCTION

Having studied the basic HTML tags from creating and formatting paragraphs like alignment, font settings, text styles and horizontal lines, now we will deal with some interesting tags of HTML related to linking and multimedia.

The Internet's main attraction is the ability to create documents with the Hypertext concept, that is, a document which links to other documents through special connections

called hyperlinks. The use of Hypertext also makes the creation of extensive documents easier, where sections can be quickly accessed through links.

The inclusion of images in Web pages adds a new dimension to the distribution of information. The inclusion of an image makes the information much more attractive, land as the saying goes a picture is worth a thousand words.

The main attraction of the Internet is the hyperlink, which allows creation of documents with connections to other documents contained in any computer connected to the Internet. Most of the users who access the Internet do so from a graphic environment like Windows. For users of this or other graphic operational systems who are accustomed to programs with multimedia features, the inclusion of images, audio and video in a home page is natural.

The main multimedia elements of any application are:

- Images
- Audio
- Video

2.2 HYPERLINKS AND ANCHORS

An anchor in a Web page that enables objects to be used as hyperlinks. A Hyperlink in a page is an area that opens a new site or page. The objects used to create hyperlinks can be either text or image. Anchors and Hyperlinks can be defined by the tag <A-----</A>

- **Hyperlink**

The key strength of HTML lies in its expertise to link one part of information to another, on the same page or on a different page. When the mouse pointer is moved over any text and if it turns into a pointing hand, it means that a link has been established. Clicking on the link will either:

- Open a new section within the same web page
- Create a link to another page within same website.
- Create a link to another web page from a different website.

➢ **Anchor**

Anchor is a container element that is used to create a Hyperlink. The object to which the link has to be made is defined by the HREF attribute. The anchor graphics generally have a border around them. There are two types of links:

- Internal Links
  The links, which has reference to Web pages of same Web site.
- External Links
  The links, which has reference to Web pages of other Web sites.

When links are created then it is essential to

- Specify the name of the file to be linked
- Specify the text or image on which user has to click for activating the link

<A> is the tag used to create an anchor. Anchor can be used for two purposes:

1. Create Hyperlinks
2. Name a target location in a document

The syntax is: <A HREF= “URL”> TEXT- TO- CLICK </A>
<A NAME=VALUE>…..</A>

HREF
HREF stands for Hypertext Reference that contains the URL of the link.

NAME

This attribute of the Anchor tag is used to identify a location within the same HTML document.

- Creating links within the same document

Bookmark the location with the NAME attribute of the <A> tag and then create a link with the HREF attribute of <A>.

Syntax to create a bookmark<A NAME= “VALUE”>
Syntax to create a link is <A HREF=#VALUE”>

The value in the hyperlink must be the same as the value in the name attribute of the anchor tag. When clicked on the link, the named section of the page will be displayed on top.

For example:

<HTML>
<HEAD>
<TITLE> Link with in the same page</TITLE>
</HEAD>
<A NAME= “intro”><H1> introduction</H1>
<P>
This is introduction about links, images, audio and video files. There are different format for audio and video files.
…….
……
</P>
</BODY>
</HTML>

Here an anchor by name intro is specified. This is the target location that is to be displayed when link is clicked.

Now it is required to create a link to this anchor as follows

<A HREF= “#intro”>Go to introduction</A>
A Hyperlink “Go to introduction” will be created, and when the user clicks on this hyperlink then control will jump to the paragraph named “intro”.

- **Creating links to an email address**

On clicking this link the e-mail editor will be automatically opened where the user can straight away type his message.

For this purpose **MAILTO**: tag is used.
The syntax is:
<A href= “mailto:Value”>text-to-click</A>
MAILTO: the value in this tag will specify the email id of the person to whom this is to be sent.

For example:

<A href= “mailto:mymail@whereever.com”>E-mail</A>
E-mail is the hyperlink when it is clicked e-mail editor will be opened and will have the mymail@whereever.com as an email address in the sent textbox.

- **Creating links to an email address with subject**

The subject of the mail can be set so that it can be added by default as soon as the link is clicked on by using the subject attribute. ?SUBJECT is used at the end of the email address to set the subject by default.

The syntax is:

<A HREF=MAILTO:Email Id?SUBJECT=Subject Matter> Text to click</A>

To create a hyperlink using an image

Just as text can be specified as link, you can use the <IMG> command as an attribute of the <A HREF> command to create a hyperlink. To create a clickable image

Syntax is: <A HREF= “targetfile.html”><img src= “image.jpg/gif”></A>

For Example:

<HTML>
<HEAD>
<TITLE>Image as link</TITLE>
</HEAD>
<BODY>
<H1>Examples of clickable image</H1>
<A href = “a1.html”><IMG ALIGN = middle SRC = “hlpcd.gif”></A>description about links

<A href = “a2.html”><IMG ALIGN = middle SRC = “hlpbell.gif”></A>description about video
</BODY>
</HTML>

2.3 INSERTING IMAGES

GIF and JPG are the standard graphic file formats accepted by the Web. GIF format (Graphics Interchange Format) can be used on different hardware platforms, such as the PC and the Macintosh. Compared to the standard Windows format, it generates much smaller files.

A second format considered a standard is the JPG format from the Joint Photographic Experts Group. It manages to reduce the size of an image by up to ten times. Version 2 and above of Netscape Navigator and Internet Explorer work with this type of file

- **IMG Command**

<IMG> command is used to insert an image in the current position where it is specified as:

<IMS SRC= “filename.gif/jpg”>

If a URL is not specified, the browser looks for the image in the current directory.

Attributes for IMG command

If you place an image close to text, you may specify the alignment of the image in relation to the test around it.

As standard, the image is aligned with its bottom in the same line as the text. Through the ALIGN attribute, you can align the text at the bottom, which is the standard, at the middle or at the top.

- ALIGN = TOP, MIDDLE or BOTTOM

  <IMG ALIGN = “top” SRC = “image.gif/jpg”>

  <IMG ALIGN = “middle” SRC = “image.gif/jpg”>

  <IMG ALIGN = “top” SRC = “image.gif/jpg”>

For Example:

<HTML>
<HEAD>
<TITLE>Images with align attribute</TITLE>
</HEAD>
<BODY>
<H1>Examples of image with different align attribute</H1>
<IMG ALIGN=TOP SRC= “hlpcd.gif”>Image with Top alignment attribute
<IMG ALIGN=MIDDLE SRC= “hlpcd.gif”>Image with Middle alignment attribute
<IMG ALIGN=BOTTOM SRC= “hlpcd.gif”>Image with Bottom alignment attribute
</BODY>
</HTML>

- **ALIGN=LEFT OR RIGHT**

Purpose: the image or video is aligned to the left or to the right of the page.

Example: <IMG SRC = “abc.gif” ALIGN= LEFT> this text appears on the left side of the image.

- **ALT = “text”**

Purpose: specifies an alternate text to be displayed in place of the image
Example<IMG SRC = “abc.gif” ALT = ”hello”>

- **BORDER = n**

Purpose: specifies the width of the image border in pixels.
Example: <IMG SRC = “abc.gif” BORDER =5> this image has a border of 5 pixels in width.

- **HEIGHT = n**

Purpose: specifies the height of the image. If the image has another size, it will be adjusted to the size specified.

Example: <IMG SRC = “abc.gif” WIDTH=150 HEIGHT=200>

- **WIDTH = n**

Purpose: specifies the width of the image. If the image has another size, it will be adjusted to the size specified.

Example: <IMG SRC = “abc.gif” WIDTH = 150 HEIGHT = 200>

- **HSPACE** = n

Purpose: specifies a horizontal margin around the image to space it.

Example: <IMG SRC = “abc.gif” HSPACE = 10 VSPACE = 10>

- **HSPACE = n**

Purpose: specifies a vertical margin around the image to space it.

Example: <IMG SRC = “abc.gif” HSPACE = 10 VSPACE = 10>

- **SRC**

Purpose: specifies the address of the image

Example : <IMG SRC = “http:// www.abc.com/abc.gif”>

- **Foreign Images**

Placing images in a page (inline) can be very interesting. However, this results in as low loading time, especially when there are several images that are very large. A very useful technique is to make a link to the page using the <A HREF> command. In this way, the reader who wishes to see a certain photograph can click on the reference and is not burdened with the automatic loading time of the images as when you simply use the <IMG> command.

For example:

```
<html>
<head>
<title>foreign images ></title>
</head>
<body>
<h1> images are loaded with <A HREF > command </h1>
<br>
this is an image describing the position of different countries with good resolution
<a href= “globe.gif”>Globe </a>
</body>
</html>
```

Upon clicking on the word Globe, the associated image is displayed in a browser window. To return to the previous windows, click on the back button. Note that this is not an HTML page, but just the figure itself.

2.4 WORKING WITH VIDEO

➢ Video File Formats

There are several video file formats available on the market, the three most common types of files are:

- **AVI** – The Microsoft standard video format
- **MOV**-The Apple standard video format, which is also supported by Windows.
- **MPEG / MPG** – a compacted video format that has become an industry standard.

Not all browsers support all the video formats. Depending on the browser, installation of a plug in (an external program with which the video type is associated) may be necessary. In this way, upon identifying the video, the browser automatically triggers the corresponding plug-in.

- **Inserting Video**

There are several ways to insert video in your page. The video can appear in the form of a link, which, upon being clicked, runs the program responsible for displaying it, or it can appear directly in the page.

After clicking on the link, the image will be loaded and executed by player program then use the run button to execute the video and close button to go back to the browser.

- **Using the <IMG> command to insert video**

The simplest way to add a video to your page is to use the options of the <IMG> command. The main parameter used to insert video is DYNSRC, the abbreviation of Dynamic Source. This parameter works in Internet Explorer 3 and 4, but is not recognized by Netscape Navigator 3 and 4

For example
<IMG DYNSRC= “abc.avi” SRC= “abc.gif” WIDTH=50 HEIGHT=50 LOOP=infinite ALIGN=RIGHT>

This command tells the browser to display the abc.avi file. In case the browser does not support displaying of videos then abc.gif image will be displayed. The LOOP option causes the video to be continuously displayed when the value infinite is specified or the number of times determined by a number.

Attributes

- **ALIGN = TOP, MIDDLE or BOTTOM**

Purpose: the text bordering the image is aligned at the top, middle or bottom of the video

Example: <IMG SRC = “abc.gif” ALIGN= MIDDLE> this text is aligned at the middle of the abc.gif figure

- **ALIGN = LEFT OR RIGHT**

Purpose: the image or video is aligned to the left or to the right of the page.

Example: <IMG SRC = “abc.gif” ALIGN= LEFT> this text appears on the left side of the image.

- **ALT = ”text”**

  Example: <IMG SRC = “abc.gif” ALIGN= LEFT> this text appears on the left side of the image.

- **ALT = “text”**

  Purpose: specifies an alternate text to be displayed in place of the figure or video

  Example : <IMG SRC = “abc.gif” ALT = “hello”>

- **BORDER = n**

  Purpose: specifies the width of the image or video border in pixels.

  Example: <IMG SRC = “abc.gif” BORDER =5> this image has a border of 5 pixels in width.

- **CONTROLS**

  Purpose: displays VCR type control buttons under the video image

  Example <IMG DYNSRC = “abc.avi” CONTROLS >

- **DYNSRC = URL**

  Purpose: specifies the URL of the video clip to be displayed. The most common video file formats are MPG, AVI and MOV

  Example: <IMG SRC = “abc.gif” DYNSRC = “abc.avi”>

- **HEIGHT = n**

  Purpose: specifies the height of the image or video. If the image has another size, it will be adjusted to the size specified.

  Example: <IMG SRC = “abc.gif” WIDTH = 150 HEIGHT = 200>

- **WIDTH = n**

  Purpose: specifies the width of the image or video. If the image has another size, it

  will be adjusted to the size specified.

  Example: <IMG SRC = “abc.gif” WIDTH = 150 HEIGHT = 200>

- **HSPACE = n**

Purpose: specifies a horizontal margin around the image to space it.

Example: <IMG SRC = “abc.gif” HSPACE = 10 VSPACE = 10>

- **HSPACE = n**

  Purpose: specifies a vertical margin around the image to space it.

  Example: <IMG SRC = “abc.gif” HSPACE = 10 VSPACE = 10>

- **Loop = n or infinite**

  Purpose: specifies how many times the video will be executed. If n is equal to –1 or infinite, the video will be executed continuously. Otherwise, it is executed the number

  of times specified.

  Example: <IMG SRC = “abc.gif” DYNSRC = “abc.avi” LOOP = 3>

  The video is executed three times.

- **SRC**

  Purpose: specifies the address of the image or video

  Example : <IMG SRC = “http:// www.abc.com/abc.gif”>

- **START = FILEOPEN / MOUSEOVER**

  Purpose: for video clips, specifies when the video should be executed. As standard the video assumes

- The value FILEOPEN and is executed immediately after being loaded.
- The value mouseover causes the video to be executed when the mouse cursorpasses over the video image area. These two options can be combined also.

  Example : <IMG SRC = “abc.gif” DYNSRC = “abc.avi” START = FILEOPEN>The video is executed upon being loaded.<IMG SRC = “abc.gif” DYNSRC = “abc.avi” START = MOUSEOVER>

  The videos executed when the cursor passes over it.

Netscape Navigator does not support the DYNSRC attribute

- **Using the <EMBED> to display Video**

  The embed command allows the display not only of video but also of files of different types which are executed by a previously installed plug-ins. It works in both Internet Explorer and in Netscape Navigator.

  Syntax:

  <EMBED

  ALIGN = LEFT | RIGHT| TOP| BOITTOM

  BORDER = PIXELS

  FRAMEBORDER = NO

  HEIGHT = pixels

  WIDTH = pixels

  HIDDEN = TRUE|FALSE

  HSPACE = pixels

  VSPACE = pixels

  NAME = appletname

  PALETTE = FOREGROUND|BACKGROUND

  PLUGINSPACE = instrURL SRC = address TYPE=MIMETYPE

  >

  …..

  </EMBED>

  - ALIGN-specifies the applet alignment
  - LEFT-Aligns the text to the Left
  - RIGHT-Aligns the text to the right
  - TOP-Aligns the text with the top part
  - BOTTOM-Aligns the text with the bottom part.

- BORDER = pixels-specifies the size, in pixels, of the border around the appelt.

- FRAMEBORDER = no-specifies that the frame has no border.
- HEIGHT = pixels-specifies the height, in pixels, required by the applet.
- HIDDEN = TRUE|FALSE-specifies if the plug-in is visible on the page
- HSPACE = pixels-specifies a margin, in pixels, between the left and right borders of the applet and the text and the surrounding images.
- NAME = applet name-specifies the name of the applet.
- PALETE = FOREGROND|BACKGROUND- is only relevant for the windows platform. The foreground and background value causes the palette used by the plug-in to use the foreground palette or background palette.
- PLUGINSPACE = instrURL-specifies the URL containing the plug-in installation instructions. This URL is used by the installation program if the plug-in registered for the MIME type of this <EMBED> tag is not found
- SRC = address-specifies the name of the data source to the object.
- Type = MIMEtype-specifies the mime type of the <EMBED> tag, which in turn determines which plug in should be loaded.
- The <NOEMBED> command

  To maintain compatibility with older browsers, use the<NOEMBED> command to display a message indicating that the browser is not capable of displaying the object.

2.5 WORKING WITH AUDIO

The addition of sound to a Homepage or Internet application can be a very interesting feature to captivate the user. There are several types of formats for audio files:

- AU- this format is dominant in the Unix platform and is supported by almost all the other platforms. It is adequate for instruments and voice.
- WAV- standard sound format in the windows platform, it offers good sound quality, but generates very large size files. It is adequate for instruments and voice.

- MIDI- this format is a standard for representation of musical instruments produced by an electronic instrument such as synthesizer. Unlike the above formats, it does not support voice.
- MP2- it is a compacted audio file equivalent to the MPEG video files. Its size is significantly smaller than WAV files.
- RAM (or RA)- the real audio format is becoming very popular as it allows execution on demand, that is, live transmissions.
- Plug-ins

Third party programs that are automatically activated to execute any type of file that the browser cannot execute. Many times when attempting to execute an unknown file, the browser itself will direct you to a page of the NETSCAPE site where the main plug-ins on the market are available for downloading.

- **Inserting audio Through hyperlinks**

  The audio can appear in the form of a link, which upon being clicked runs the program responsible for displaying it. After clicking on the link, the file will be executed by player program then use the VCR control buttons to interrupt or continue the sound execution.

- **Inserting background music**

  The <BGSOUND>Command (Internet Explorer)

  This new command introduced by Microsoft's browser is very appealing. It loads and executes a sound file as soon as the page is accessed. Once the sound execution begins, pressing the ESC key or changing pages can interrupt it.

  Syntax is:

  <BGSOUND SRC = URL LOOP = N | INFINITE>

Attributes

- **SRC = URL**

  Purpose: specifies the address of the audio file which will be executed

  Example: <BGSOUND SRC = “RING.AU”> It executes the ring.au file

- **Loop = n or infinite**

  Purpose: specifies how many times the file will be reproduced.

  Example: <BGSOUND SRC = “RING.AU” LOOP = 3> It executes the ring.au file three times.

  Example: <BGSOUND SRC = “RING.AU” LOOP = INFINITE> The value infinite reproduces the file continuously. The<BGSOUND> command presents two disadvantages:

- The sound file is always executed when the page is accessed. If you place this command in a page serving as a menu to access other pages, execution of the music can irritate the user after a few accesses.
- Netscape Navigator does not execute it.
- **<META> command**

Using the<META> command you can execute a sound file immediately upon loadingthe page, or specify a time for it to be played.

For example:<HEAD><META HTTP-EQUIV= “REFRESH” CONTENT= “10;

URL=abc.mid”></HEAD>This is an example of a midi file being executed ten seconds after access to the page

2.6 SUMMARY

The Internet’s main attraction is the ability to create documents with the Hypertext concept, that is, a document which links to other documents through special connections called hyperlinks. The use of Hypertext also makes the creation of extensive documents easier, where sections can be quickly accessed through links.

The main attraction of the Internet is the hyperlink, which allows creation of documents with connections to other documents contained in any computer connected to the Internet.

Hyperlinks and Anchors

An anchor in a Web page that enables objects to be used as hyperlinks. A Hyperlink in a page is an area that opens a new site or page. The objects used to create hyperlinks can be either text or image. Anchors and Hyperlinks can be defined by the tag <A-----</A>

Inserting Images

GIF and JPG are the standard graphic file formats accepted by the Web. GIF format (Graphics Interchange Format) can be used on different hardware platforms, such as the PC and the Macintosh. Compared to the standard Windows format, it generates much smaller files.

A second format considered a standard is the JPG format from the Joint Photographic Experts Group. It manages to reduce the size of an image by up to ten times.

IMG Command

<IMG> command is used to insert an image in the current position where it is specified.

Inserting Video File

There are several video file formats available on the market, the three most common types of files are:

AVI – The Microsoft standard video format

MOV-The Apple standard video format, which is also supported by Windows.

MPEG / MPG – a compacted video format that has become an industry standard.

Using the <IMG> command to insert video

The simplest way to add a video to your page is to use the options of the <IMG> command. The main parameter used to insert video is DYNSRC, the abbreviation of Dynamic Source.

For example

<IMG DYNSRC= “abc.avi” SRC= “abc.gif” WIDTH=50 HEIGHT=50 LOOP=infinite ALIGN=RIGHT>

Inserting Audio Files

The addition of sound to a Homepage or Internet application can be a very interesting feature to captivate the user. There are several types of formats for audio files:

AU- this format is dominant in the Unix platform and is supported by almost all the other platforms. It is adequate for instruments and voice.

WAV- standard sound format in the windows platform, it offers good sound quality, but generates very large size files. It is adequate for instruments and voice.

MIDI- this format is a standard for representation of musical instruments produced by an electronic instrument such as synthesizer. Unlike the above formats, it does not support voice.

MP2- it is a compacted audio file equivalent to the MPEG video files. Its size is significantly smaller than WAV files.

RAM (or RA)- the real audio format is becoming very popular as it allows execution on demand, that is, live transmissions.

Inserting background music

This new command introduced by Microsoft’s browser is very appealing. It loads and executes a sound file as soon as the page is accessed. Once the sound execution begins, pressing the ESC key or changing pages can interrupt it. Syntax is:

<BGSOUND SRC = URL LOOP = N | INFINITE>

2.7 KEYWORDS

Hyperlink: A hyperlink in a web page is an area that opens a new site or web page

Anchor: Tag used for linking other web pages through hyperlink

IMG: Tag use for inserting graphics, audio and video files in a web page

AVI: Microsoft standard video format

MOV: Apple standard video format

MPEG: Motion Picture Experts Group

DYNSRC: Attribute used for dynamic source

BGSOUND: Tag used for inserting background music

2.8 SUGGESTED READINGS

Ivan Bay Ross, HTML DHTML JAVASCRIPT PERL and CGI, BPB Publication

Jose A.Ramalho, Advanced HTML 4.0 with DHTML, BPB Publication

****

LESSON: 3

WORKING WITH JAVA SCRIPT AND VB SCRIPT

STRUCTURE

3.0 OBJECTIVES

After going through this lesson, you will be able to:

- Describe the role of scripting languages
- Defines the working of JavaScript and VBScript Language.

3.1 INTRODUCTION

The HTML language is a specialized language for formatting pages and creating hyperlinks. It has no resources for validating contents and has no structural commands to allow repetitive execution of parts of the program. If you create a form to receive data from the user, HTML, simply shows the field and accepts the entry. That's it, you cannot test for invalid entries and take action. In this case, one solution is to use a CGI program, written in another language and residing at the server, to evaluate the contents of all the fields typed and returns an error or warning message. The other solution is to use a scripting language like JavaScript or VBScript, which complement the HTML language.

JavaScript was developed by Netscape and was first supported in version 2 of Netscape navigation. The VBScript language was Microsoft's answer to JavaScript. It is a subset of the visual basic language. While VBScript is supported only by version 3 of Internet Explorer, java script is supported both by Internet Explorer and Netscape Navigator.

3.2 JAVASCRIPT

The first thing to understand is that JavaScript should not be confused with the java programming language. Java allows creation of an independent application and has all the resources of a language that can create commercial applications, such as 'c' or 'Delphi'. More specifically, Java is specialized in creating small programs called applets, which are made to be distributed on the Internet. JavaScript is a scripting language that fits into an HTML program. You cannot create a program in JavaScript and execute it without having a browser. Netscape initially created it with the name LiveScript. For marketing reasons and due to its association with sun, it was decided to change its name to JavaScript, as the two languages share many characteristics, since both uses the c language as a model.

JavaScript is an interpreted language, that is, the source code is always translated to a machine language that the computer understands when being executed. It is based on objects, which means that the programmer can use predefined objects or else create new objects to satisfy his needs.

Object orientation

JavaScript is an object-based language. This means, in a very simple way, that it treats each of the elements of a web page as an object. The objects are normally grouped according to their type or purpose. Learning JavaScript is half way to learning Dynamic HTML, which uses the concept of object orientation.

The JavaScript language has objects that are automatically created by the language and allow you to create new objects. An object normally stores a series of information

(properties), which can be accessed and used for processing or changed by the programmer. For instance, a Window displaying a home page has several intrinsic objects. Let's look at some of them:

- **Document:** this object contains information about the page or HTML document as whole, including data about the forms, links, and anchor elements, along with a series function that allow changing the page characteristics.
- **Form:** This keeps specific information about the current page's forms such as its method, URL, and data or its elements or fields.
- **History:** This object maintains a list of all the sites visited in the current browser session.
- **Location:** This object has information about the page location and related information, such as the protocol used and its domain.

Properties

An object has its own characteristics that make it unique within a group. For example, a car has characteristics such as the type, number of doors, engine power, passenger capacity, exterior colour, weight, interior color, etc., while a refrigerator has characteristics of type, capacity (in cubic feet), color, height, weight, etc.

An object's properties are accessed through a very simple syntax consisting of the object name, a period, and the property name. For example, the document object (which addresses the HTML page) has the background color property. To refer to this property, use the following syntax:

document.bgcolor

Methods

Besides properties, most objects have special functions called methods that execute some operation related to the object. The syntax is:

Object.method_ name (“argument”)
Object – The name of the object undergoing the method’s action.
Method name --- An expression or optional value which will be used to change or act upon the object.

The document object has a method called write, the purpose of which is to insert lines in the HTML page during its execution:

document. write (“<h1>JavaScript demonstration < /h1>”).

The window object has a method called Alert which displays a warning message in the dialog box:

window.alert(“invalid operation”)

Events

An event is an action occurring on the page due to the interaction of the user. For example, clicking on a button, or entering the content of a field are events. Below are some examples:

- OnClick-Execute some code when an object is clicked.
- OnMouseOver-Executes some code when the cursor passes over the object.
- OnBlur-Executes when a field is changed.

Functions

Functions are independent routines that execute a specific task. They are different from methods, as they are not associated with an object. JavaScript has generic functions that perform mathematical operations, with strings, date, etc. The language also allows the user to create this own functions.

```
Function Calculate(a,b,c)
d = a*b*c
return d
```

Variables

A variable can receive its content through the simple attribution of the content or by the result of the action of an expression or function.

```
Name = ”John”
Date1= date()
Product = 9*8
```

The name of a variable may start with a letter or underline (_) followed by numbers or letters without spaces. JavaScript differentiates between upper and lowercase letters.

Literals

Literals are the representation of a number or string. They are fixed information, which cannot be changed.

Here are some examples:

67	integer number
5.678	floating point number
“john”	text

Literals may be of several types:

- ***Integer Literal***

Represent a number that may be a positive, negative, or functional.

e.g., A = 690 B = 0.89 C = -89

- ***Floating Point Literal***

This type of literal is also known as scientific notation, in which the numbers expressed in the following form: 5.67e5

- ***Boolean Literal***

Boolean literals may be True or False.

- ***String Literal***

These consist of text enclosed by quotation marks or accent marks.

e.g., “John Martin” “ ”

- ***Special Characters***

Within a string, special characters may be specified as shown:

- \n Inserts a line break
- \t Inserts a tabulation character
- \r Inserts a carriage return
- \a Inserts a beep
- \f Inserts a page advance

Expressions

An expression is a combination of variables, literals, methods, functions and operators, which return some result. An expression is used to attribute a value to a variable or to be tested, and an action is taken as a function of its result.

To create a numeric variable, the expression to be used would be:

A = 8

B = 9*9

Operators

There are several operators that can be used in expressions. They can be grouped by the type of operation they perform.

- ***Assignment Operator (“=”)***

It attributes the content of the expression on its right to the variable on its left. X=20 Other attribution operators are:

- x + = y (x = x + y)
- x - = y (x = x - y)
- x * = y (x = x * y)
- x / = y (x = x / y)
- x % = y (x = x % y)

➢ ***Incremental and Decremental Operators***

The incremental operator is composed of two signs, "++". It increases the value of a numeric operand every time it is executed. The syntax of this operator is:

x++ or ++x.

If the ++ sign is placed before the operand (++x), It increments the operand and returns the new value.

If the ++ sign is placed after the operand (x++), It returns the operand and value and then increments it.

For example

x = 5

a = x++

It attributes the value 5 to a and increases the value of x to 6.

x = 5

a = ++ x

It increases the value of x to 6 and attributes this value to a.

The decremental operator is composed of two signs, "--".It decreases the value of a numeric operand every time it is executed.

The syntax of this operator is:

x- or –x

If the -- sign is placed before the operand (--), It decrements the operand and returns the new value.

If the -- sign is placed after the operand (x--), It returns the operand and value and then decrements it.

For example
x = 5
a = x

--It attributes the value 5 to a and decreases the value of x to 4.
x = 5
a = -- x

It decreases the value of x to 4 and attributes this value to a.

- ***Logical Operators***

These operators require Boolean values as operands and return a logical value.

- AND(&&)

Syntax : expr1 && expr2
This operator returns True if the two expressions are true.

- OR(||)

Syntax : expr1 || expr2
This operator returns True if one of the expression is true.

- NOT(!)

Syntax : ! expr This NOT operator reverses the expression. If expr is true then it return the value False If expr is False then it returns the value True.

- ***Comparison Operators***

The operators compare the operands, which can be numeric or string values and return a True or False logical value based on the comparison result.

Operators are:

- Equal (= =) Returns true if the operands are equal.
- Not equal (! =) Returns true if the operands are different.
- Greater than (>) Returns true if the operand on the left is greater than that on the right.
- Greater than or equal to(>=) Returns true if the operand on the left is greater than or equal to that on the right.
- Less than (<) Returns true if the operand on the left is less than that on the right.
- Less than or Equal to(<=) Returns true if the operand on the left is less than or equal to that on the right.

- **String Operators**

The plus sign "+" can be used to concatenate or join two strings into a single value.
A = "first"
Z = x+y

Main Objects of JavaScript

- Document Object

Contains information about the current HTML document.

Properties

Title	Contains the title of the document or "Untitled" in the event it has not been defined.
Location	Contains the complete document URL.
Lastmodified	Contains the date of the last document modification

BgColor	Contains the RGB value of the background color expressed in hexadecimal.
FgColor	Contains the RGB value of the foreground color expressed in hexadecimal.
LinkColor	Contains the RGB value of the links color expressed in hexadecimal.
VlinkColor	Contains the RGB value of the visited links expressed in hexadecimal.
AlinkColor	Contains the RGB value of the activated links.
forms[index]	Matrix with the form objects. The objects appear in the order they were defined.
forms.length	The number of form objects of the current document.
links[index]	Array objects corresponding to all HREF links in source order.
links.length	The number of links generated by HREF contained in the current document.
anchors[index]	Matrix containing objects referring to anchors (A NAME) in the current document.

Methods

write( )	Records a line in the current document.
writeln( )	Records a line in the current document adding a line advance. This advance has an effect only within the <PRE> or <XMP> commands.
clear( )	Clears the window.
close( )	Closes the window.

- **Forms Object**

Every form within a document corresponds to a distinct object. These objects are maintained in a matrix called forms, which is a property of the Document object. The

forms are composed of elements of various types that have their own properties and methods.

Properties

name	String with the content of the NAME attribute.
method	value of the METHOD attribute; “get” returns 0 and “post” returns 1.
action	String with the content of the ACTION attribute.
target	Target window used for response after sending the form.

Event Handlers

onSubmit( )	Activates code executed when the form is sent.

Methods

submit ( )	Sends the form.

- **Text Type Elements**

The elements of Text and Text area type forms have the same properties, methods, and event handlers.

Properties

name	String with the content of the NAME attribute.
value	String with the content of the field.
defaultValue	String with the initial content of the field.

Methods

blur( )	Removes the object
focus.focus( )	Moves the focus to the object
select( )	Selects the input data field of the object

Event Handlers

onFocus	Executes the code associated when the focus falls on the field, either by using Tab or by clicking without selecting a field.
onBlur	Executes the code when the focus is removed from the field.

onSelect	Executes the code when the field receives focus due to the selection of a part of its text.
onChange	Executes the code when the field loses the focus and the user changes its content.

- **Checkbox Type Elements**

A checkbox type form field is an on/off type switch.

Properties

name	String containing the content of the NAME attribute
value	String; has the value “on” if the field is selected or “off” otherwise.
status	Boolean value; False if the item is not marked, or True otherwise.
DefaultStatus	Boolean value indicating if the element is selected as standard by the attribute checked.

Event Handlers

onClick	Code executed when the user clicks on the item.

Methods

click( )	Selects the checkbox, activating it (“on”).

- **Radio Button Type Elements (Option Buttons)**

If the form has more than one, all the radio buttons receive the same NAME attribute. When more than one radio button is specified with the same value attributed to the Name field, a group of buttons is created.

Properties

name	String with the content of the NAME attribute.
index	Number corresponding to the field in the list of items. Starts with zero.
value	String with the content of the VALUE attribute.
status	Boolean value; False if the item is not pressed, or True otherwise.

DefaultStatus	Boolean value indicating if the element is selected as standard by the attribute checked.

Event Handlers

onClick	Code executed when the button is pressed.

Methods

click( )	Selects the radio button.

- **Selection List Type Elements**

A selection list allows selection of one or more values for a same field.

Properties

index	Number identifying the position of the option in the selection list. Starts with zero.
text	Content of the Option tag.
value	content of the VALUE attribute.
defaultSelected	Boolean value indicating if the option is selected as standard by the SELECTED attribute of the Option tag.
Selected	previously selected value.

Event Handlers

OnFocus	Executed when the field receives focus.
OnBlur	Executed when the focus leaves the field.
onChange	Executed when the focus leaves the field and it is changed.

Methods

click ( ) selects a radio button.

- **Button Type Elements**

Button type elements can be classified as Submit type buttons, which send a form to the server, Reset type buttons, which erase or return the initial values of each field, and customized button that execute a specific task determined by the user.

Properties

value String with the content of the VALUE attribute.

name String with the content of the NAME attribute.

Event Handlers

onClick Code executed when the button clicked.

Methods

click( ) Selects the button.

Declarations

A list of the main commands or declarations of the JavaScript language as follows:

- **break**

This command interrupts the while or for command being executed and transfers control of the program to the command following the command ending the loop.

Syntax: break;

Example

```
Function display()
{
        var i=0;
        while(i<=10)
        {
        if( i == 5)
                break;
        document.write(“ ” + i);
        i++;
         }
 }
```

- **comment**

As with most languages, comment line may be included in the JavaScript code by using a special sign to start the line before the comment text. For comments with a single line, use two bars(//). For multiple line comments, start the comment with /* and end if with*/.

Syntax: //text /* text */

Example

```
Function display()
{
        var i=0;
        while(i<=10)
        {
        if( i == 5)
                break;
                //when i reaches value 5 then break statement terminates the while  loop.
                /*when i reaches value 5then break statement terminates the while loop.*/
        document.write(“ ” + i);
        i++;
         }
}
```

- **continue**

The continue command ends the execution of a group of commands with in a loop generated by the while or for command, restarting the process at the next iteration means it returns the control to the line containing the condition validated by the loop.

Syntax: continue;

Example

```
Function display()
```

```
{
        var i = 0;
        while(i< = 10)
        {
        if( i == 5)
                continue;
                //when i reaches value 5 then remaining statement of this iteration will not
                be executed.
        document.write(“ ” + i);
        i++;
        }
}
```

- **for command**

The for command is a control structure allowing repetitive execution of commands enclosed with in the limits of for command.

Syntax:

```
for(expr1;cond1;expr2)
{
commands;
}
```

expr1 is a variable that receives an initial value which will be used to counter the number of times the for command will be executed. Cond1 is the expression which is evaluated every time the loop is executed, if condition is true, the commands with in the loop are executed otherwise for loop will be terminated and control will pass to the line following the for command.

Exp2 is the expression used to increment the counter.

Example

```
for(var x = 0; x< = 10;x++)
{
document.write(“ “ + x);
}
```

- **for .. in**

This command interacts the var variable with the properties of the obj object. For each property of the object, the command executes the commands specified by commands. Syntax: for(var in obj) { commands }

- **Function**

This command creates a function defined by the user. It receives a name determined by the text specified in name and number of parameters indicated in param. In case of the function returning a value, the return command should be specified with return expression.

Syntax:

```
Function name ( [param] [ , param ] [ . . . , param ] )
{
commands
[return expr]
}
```

All the parameters are passed by value not by reference.

Example

```
Function square(x)
{
return x*x;
```

}

- **if…else**

This commands evaluates an expression and executes the commands within if clause. If the result of expression is true otherwise commands within else clause will be executed.

Syntax:

```
if(condition)
{
commands
}
[ else {
commands
}]
```

Example

```
If( x<=10)
{
y=x*5;
}
else
{
y=x*10;
}
```

- **return**

This command returns the result of the expression.

Example

```
Function cube(x)
```

```
{
return x*x;
}
```

- **this**

This command replaces the current object name when a method or property is applied.

Syntax: this.property

- **var**

This command declares the name of the variable and optionally initializes the variable with a value.

Syntax: var varname [=value] [….,var varname [=value]]

Example

```
var name,salary; var salary = ”1000”;
```

- **while command**

This command is a control structure that evaluates the expression. If it is true then group of commands with in while statement will be executed otherwise control will be transferred to the command following the while command.

Syntax: while(condition) { commands }

Example

```
var x = 0;
while(x<=10)
{
document.write(“ “ + x);
```

```
x++;
}
```

JavaScript extensions for the HTML language

The container defined by the <SCRIPT> </SCRIPT> command is used to store commands that should be executed immediately after the loading of HTML document and functions defined by the user which are executed because of the occurrence of a certain event with an object.

Example 1 (on Blur Event)

```
<html>
<head>
<title>OnBlur</title>
<script language = “JavaScript”>

function manadatory(x)
{
if (x.value=””)
{
alert(“field” + x.name + “may not be left blank”)
}
</script.
</head>
<body>
<form>
<br>
Name:<input type= “text” name= “name1”value = “” size=5 onBlur= manadatory(this)”>
Age: <input type= “text” name= “name1”value = “” size=5 onBlur= “manadatory(this)”>
Marks: <input type=“text” name=“name1”value =“” size=5 onBlur=“manadatory(this)”>
```

```
</form>
</body>
</html>
```

On the execution of this program if we leave any of the required field blank and jump to another field then the message will be immediately displayed.

Example 2 (onChange Event)

```
<html>
<head>
<title>OnChange</title>
<script language = “JavaScript”>
function manadatory(x)
{
if (x.value= “”)
{
alert(“field” + x.name + “may not be left blank”)
}
}
function test(x)
{
If (x.value.length<5)
{
alert(“field” + x.name + “may not have less than 5 characters”)
}
}
</script>
</head>
<body>
<form>
```


Name:<input type= “text” name= “name1”value = “” size=5 onBlur= “manadatory(this)” onChange= “test(this)”>
Age: <input type= “text” name= “name1”value = “” size=5 onBlur= “manadatory(this)”>
Marks: <input type=“text” name=“name1”value =“” size=5 onBlur=“manadatory(this)”>
</form>
</body>
</html>

The Change event occurs when a select, text or text area type field loses focus and its content is changed. It occurs whenever the field loses focus and having its content changed. The name field is specified to activate a function which checks the number of characters entered, if this is less than 5, it will display an error message.

Example 3 (onClick Event)

```
<html>
<head>
<title> Use of events and functions</title>
<script language = “javascript”>
document.write(“Line generated by JavaScript during its loading” + “<br>”)
</script>
<body>
<hr>
<form>
<input type=button name= “Red” onClick = “document.bgcolor=’red’”>
<input type=button name= “Green” onClick = “document.bgcolor=’Green’”>
</form>
<hr>
</body>
```

</html>

Example 4 (OnFocus Event)

This event occurs when a field receives the input or typing focus by pressing tab in the previous field or by clicking mouse. It acts on select, text or text area type fields.

```
<html>
<head>
<title>onFocus</title>
<script language= “JavaScript”>
function sum()
{
d=eval(document.form1.field1.value)+eval(document.form1.field2.value)+eval(documen
t.form1.field3.value) document.form1.field4.value=d }
</script>
</head>
<body>
<form name= “form1”>
field1:<input type=text name= “field1” value= “” onBlur= ‘if(this.value= “”)
{this.value=0}’> <br>
field2:<input type=text name= “field2” value= “onBlur= ‘if(this.value=””)
{this.value=0}’>
field3:<input type=text name=”field3” value=”” onBlur= ‘if(this.value= “”)
 {this.value=0}’>
Sum:<input type=text name= “field4” value= “” onFocus= “sum()”> </form> </body>
 </html>
```

3.3 VBSCRIPT

The Visual Basic language comes in several “flavors”. Each flavor has a specific use. One of the flavors of Visual Basic is Visual Basic Scripting Edition or VBScript. VBScript is a subset of the Visual Basic language and supports most of its syntax and

structure. VBScript code is embedded in a web page and translated by the web browser when the page is viewed.

VBScript can be used in:

- **Web pages** – client-side scripting in Internet explorer. VBScript can be embedded in HTML documents to add interactive functionality to web pages. Data validation and user interaction can be provided at the client-end.
- **Active Server pages** – server-side scripting in Internet Information Server is used in active server pages. Web pages can be generated and altered at the server-end before being sent to the client (through the browser).

The applications with which you can use VBScript are called the host applications. The common host applications are:

- Internet Explorer
- Internet Information Server
- Outlook uses VBScript as its macro language
- Windows Scripting Host

While the VBScript language works the same on all the host applications the objects that are available with each application may differ. That is, the objects that are available with Internet Explorer may not be available with Internet Information Server. Or, the objects available with Internet Information Server may not be supported by Internet explorer.

Structure of VBScript

The <SCRIPT > … </SCRIPT> tags are used to mark the beginning and end of VBScript syntax within a web page.

Example

```
<html>
<head>
<script language= “VBScript”>
MsgBox(“This is fun”)
</script>
</head>
<body>
<h2>
<font color= “red” size=4>Using VBScript</font>
</body>
</html>
```

Example

```
<html>
<head>
<script language= “VBScript”>
Sub display()
document.write(“This is fun”)
End sub
display
</script>
</head>
<body>
<h2>
<font color= “red” size=4>Using VBScript</font>
</body>
</html>
```

To call a procedure, you have to include the name of the procedure within a command. You can also use the call keyword to call a procedure. Any arguments that are passed to

the procedure are enclosed within parentheses. If you do not use the call keyword, then the arguments are not enclosed in parentheses.

Example

```
<html>
<head>
<script language= “VBScript”>
Sub sum (a, b)
x=a+b
document.write(“sum of ”+ a + “ and ”+b+“ is ”+x)
End sub

call sum(8,9)
</script>
</head>
<body>
<h2>
<font color= “red” size=4>Using VBScript</font>
</body>
</html>
```

Variables

A variable is a container that refers to a memory location, it is used to hold values that may change while the script is executing. A variable does not have to be declared before it is used in the script. However, it is good programming practice to declare a variable before using it. The Dim, Public or Private statements are used to declare a variable.

```
<%Dim/Public/Private%>
```

The Public or Private keyword specifies the scope of variable. If the variable is declared as public, then the variable is within page scope and If it is private then it is within script where the variable is declared. The Option Explicit is used to ensure that all variables are declared before they are used. The Option Explicit statement is included after before any HTML text or script commands.

Example

```
<html>
<head>
<script language= “VBScript”>
Option Explicit
Sub sum (a, b)
x= a + b
document.write(“sum of ”+ a + “ and ”+b+“ is ”+x)
End sub

call sum(8,9)
</script>
</head>
<body>
<h2>
<font color= “red” size=4>Using VBScript</font>
</body>
</html>
```

The code specified above will generate an error when if will be executed because Option explicit statement specifies that all variables should be declared before they are used.

To rectify the error the code is changed to:

```
<html>
<head>
<script language= “VBScript”>
Option Explicit
Sub sum (a, b)
Dim x
x=a+b
document.write(“sum of ”+ a + “ and ”+b+“ is ”+x)
End sub
Call sum(8,9)
</script>
</head>
<body>
<h2>
<font color= “red” size=4>Using VBScript</font>
</body>
</html>
```

Data types

VBScript supports only one data type, variant. The data type can hold any type of data that is supported by Visual Basic. For example integers, strings, Boolean and so on.

Example

```
<html>
<head>
<script language= “VBScript”>

Dim x
X=678
document.write(x+”<br>”)
```

```
x=”hello”
document.write(x+”<br>”)
</script>
</head>
<body>
<h2>
<font color=”red” size=4>Using VBScript</font>
</body>
```

the data type of the variable is determined when the script is executed.

Operators

VBScript provides a lot of operators that help in manipulating the variables in the page. These operators are logically grouped into:

- **Arithmetic operators**

Symbol	Operator
+	Addition
	Subtraction
*	Multiplication
/	Division
\	Integer Division
Mod	Modulus Operator
^	Exponentiation

- **Comparison operators**

Symbol	Operator
=	Equality
<>	Inequality

>	Greater than
>=	Greater than or equal to
<	Lesser than
<=	Less than or equal to
IS	Compare two objects

- **Logical Operators**

Symbol	Operator
AND	Conjunction
OR	Disjunction
NOT	Logical Negation
XOR	Exclusive Or

- **Concatenation Operators**

Symbol	Operator
&	String Concatenation
+	String and Numeric Operation

Comments

A single quote (‘) is used to tell the browser to ignore statements that allow the symbol while processing the code.

Example

```
<html>
<head>
<script language=”VBScript”>
Dim x ‘ this is declaration statement
document.write(x) ‘this is used to display the value on page
```

```
</script>
</head>
<body>
<h2>
<font color= “red” size=4>Using VBScript</font>
</body>
</html>
```

Control statements

The statements that control decisions and loops in your scripts are called control structures. That is because they are used to control execution of the program and make it perform the way you want it to perform at run time.

Decision Structures

- **If…Then…Else statement**

The syntax is

```
If Condition Then satatements [Else elsestatements ]
End If
```

Example

```
<html>
<head>
<script language= “VBScript”>
x=InputBox(“enter a number between 1 and 10”)
If x>10 then
document.write(“The number would be less than 10”)
Else
document.write(“Entered valid input”)
End If
```

</script>
</head>
<body>
<h2>
Using VBScript
</body>
</html>

- **Select Case statement**

The Syntax is
select case x
Case expressionlist1 Statements
Case expressionlistn Statements
Case else
Statements
end select

Decision structures are used to select between two or more alternative blocks of code during run time. The if statements make their decision by evaluating logical conditions as true or false the select case structure chooses a block of code by comparing a single test value with lists of case values. When a match is found the corresponding block of code is performed.

In the following example, the a select case statement is used to evaluate the input. A message is displayed depending upon the value input by user.

Example

<html>
<head>
<script language= “VBScript”>
x=InputBox(“enter any of following choice, John, James, Julie”)

```
select case x
case “John”
document.write(“Welcome John”)
case “James”
document.write(“Welcome James”)
case “Julie”
document.write(“Welcome Julie”)
case else
document.write(“Welcome stranger”)
end select
</script>
</head>
<body>
<h2>
<font color= “red” size=4>
Using VBScript
</font>
</body>
</html>
```

- **Loops**

Structures that control repetition in a program are known as loops. There are several kinds of loops -Loops that are used to repeat statements until a condition is false -Loops that are used to repeat statements until a condition is true -Loops that repeat statements a specific number of times.

VBScript has the following looping statements.

a. Do…loop

The Do… loop statement is used to run a block of statements for an indefinite number of times. The statements repeat a block of statements while a condition is true or until a condition becomes true.

Example

```
<html>
<head>
<script language= “VBScript”>
j=1
Do while j <=10
document.write(j + “<br>”)
j = j+1
loop
</script>
</head>
<body>
<h2>
<font color= “red” size=4>Using VBScript</font>
</body>
</html>
```

b. while….wend

The while… wend loop executes a block of statements as long as a given condition is true. Control is then passed to the while statement and condition is again checked. If condition is still true, the process is repeated. If condition is not true then execution resume with the statement following the wend statement, if the condition is null, then it is treated as false.

Example

```
<html>
```

```
<head>
<script language= “VBScript”>
j=1
while j <=10
document.write(j + “<br>”)
j = j+1
wend
</script>
</head>
<body>
<h2>
<font color= “red” size=4>Using VBScript</font>
</body> </html>
```

c. for … next

The do…loop works well if you do not know the number of times the statement in the loop is to be executed. The for loop on the other hand, repeats a group of instructions a specified number of times. The for loop uses a counter variable that increases or decreases in value during each repetition of the loop.

Example

```
<html>
<head>
<script language= “VBScript”>
For j =1 to 10
document.write(j + “<br>”)
Next
</script>
</head>
<body>
```

<h2>
Using VBScript
</body>
</html>

d. Exit statement

To change the navigational flow from the normal sequence, an exit statement can be used. For e.g. to interrupt and exit out of for loop, we have to use:

Example

<html>
<head>
<script language= “VBScript”>
For j =1 to 10
document.write(j + “
”)
If j=5 Then
Exit For
End If
Next
</script>
</head>
<body>
<h2>
Using VBScript
</body>
</html>

In this code when value of j reaches to 5 then exit statements terminates the for Loop and control will be transferred to the statement following For..Next loop.

Responding to Events

Much of the popularity of a scripting language stems from the ability to support event handling. Events may be generated by the user or by the system. For example, when the user clicks the mouse button on the web page, the onClick event is generated. An example of a system-generated event is page resizing. A web page can also include HTML controls and ActiveX controls. These controls support events. For example, a button element supports the onClick event. The text elements support the onChange event. Event handling routine can be created to perform a certain activity in response to the event that is generated.

Example

```
<html>
<body>
<input type=button value= “click me” name= “but1”>
<script language= “VBScript”
Sub but1_onclick()
MsgBox “You clicked the button control”
End Sub
</script> </body> </html>
```

Accessing object properties and methods

In addition to events, an object also exposes properties and methods. To access the property on method you have to specify the object name and the property and assign a value to it. The syntax is:

Objectname.property = value/expression

Example

```
<html>
<body>
```

```
<input type=button value= “click me” name= “but1”>
<script language= “VBScript”
Sub but1_onclick()
but1.Value= “You clicked the button control”
End Sub
</script>
</body>
</html>
```

VBScript is not case sensitive. So you can use Document. Write or document.write when sending output to the browser.

3.4 SUMMARY

The HTML language is a specialized language for formatting pages and creating hyperlinks. It has no resources for validating contents and has no structural commands to allow repetitive execution of parts of the program. The solution is to use a scripting language like JavaScript or VBScript, which complement the HTML language.

JavaScript

JavaScript is an interpreted language, that is, the source code is always translated to a machine language that the computer understands when being executed. It is based on objects, which means that the programmer can use predefined objects or else create new objects to satisfy his needs.

The JavaScript language has objects that are automatically created by the language and allow to create new objects. An object normally stores a series of information (properties), which can be accessed and used for processing or changed by the programmer.

VBScript

VBScript is a subset of the Visual Basic language and supports most of its syntax and structure. VBScript code is embedded in a web page and translated by the web browser when the page is viewed.

VBScript can be used in:

Web pages – client-side scripting in Internet explorer. VBScript can be embedded in HTML documents to add interactive functionality to web pages. Data validation and user interaction can be provided at the client-end.

Active Server pages – server-side scripting in Internet Information Server is used in active server pages. Web pages can be generated and altered at the server-end before being sent to the client (through the browser).

The applications with which you can use VBScript are called the host applications. The common host applications are:

- Internet Explorer
- Internet Information Server
- Outlook uses VBScript as its macro language
- Windows Scripting Host

While the VBScript language works the same on all the host applications the objects that are available with each application may differ. That is, the objects that are available with Internet Explorer may not be available with Internet Information Server. Or, the objects available with Internet Information Server may not be supported by Internet explorer.

3.5 KEYWORDS

APPLET: Small programs in Java that are executed in web browser

EVENTS: An event is an action occurring on the page due to the interaction of the user

FUNCTIONS: Functions are independent routines that execute a specific task

ASP: Active Server Pages

MOD: Modulus Operator, the arithmetical operator to find out the remainder when one integer is divided by another integer.

VARIABLES: A variable can receive its content through the simple attribution of the content or by the result of the action of an expression or function

LITERALS: Literals are the representation of a number or string. They are fixed information, which cannot be changed.

EXPRESSIONS: An expression is a combination of variables, literals, methods, functions and operators, which return some result

VARIABLES: A variable is a container that refers to a memory location, it is used to hold values that may change while the script is executing

COMMENTS: A single quote (‘) is used to tell the browser to ignore statements that allow the symbol while processing the code.

CONTROL STATEMENT: The statements that control decisions and loops in your scripts are called control structures

LOOPS: Structures that control repetition in a program are known as loops

EXIT: This statement is used to interrupt and exit out of for loop

3.6 SUGGESTED READINGS

Ivan Bay Ross, HTML DHTML JAVASCRIPT PERL and CGI, BPB Publication.

Jose A.Ramalho, Advanced HTML 4.0 with DHTML, BPB Publication.

****

LESSON: 4

INTRODUCING LINUX

STRUCTURE

4.0 OBJECTIVES

After going though this lesson, you will be able to:

- Define the basic concepts of Operating System
- Describe different types of operating systems
- Learn Linux, its commands, vi editor
- Learn shell programming elements of Linux.

4.1 INTRODUCTION

An operating system is the program that contains a set of core functionality for other programs, providing both the interface between other programs and the hardware, and the interface between other programs and the user sitting at the computer.

User ⟶ Application ⟶ Operating System ⟶ Hardware

(Relationship between the user, applications, operating systems and hardware)

Operating System Functions

All applications rely on the operating system for basic services; the operating system in turn controls the hardware. An operating system will usually do the following:

- Initialize the computer hardware so that the operating system and other programs can function correctly.
- Allocate system resources, such as memory and processing time, to the programs that are using the operating system.
- Keep track of multiple programs running at the same time.
- Provide an organized method for all programs to use system devices. The effectiveness with which an operating system handles these basic tasks determines its power. The core of an operating system, called the kernel, controls their tasks.

Major parts of an operating system are:

- **Kernel:** The core of this operating system, which schedules when programs can use computer resources and interfaces directly with core components of the computer hardware, such as memory and hard disks.
- **Device drivers:** special software that provides access to additional hardware beyond core device support provided by the kernel.
- **Utility Programs:** special software that helps manage the hardware and operating system features
- **Graphical Interface:** the program that provides mouse driven applications with menu bars, buttons and so forth.

4.2 INTRODUCING LINUX

Linux is free Unix-type operating system. It enables multitasking, simultaneous multiple users, the sharing of a system libraries for efficiency, TCP/IP networking, virtual memory and swap spaces, and other Unix OS features. Users can use a GUI or the command line.

Linux enables you to set up Internet or intranet services and many use it for setting up Internet firewalls. Because Linux does not require steep licensing fees and can be used on relatively inexpensive equipment, it is becoming a favorite of Internet service providers (ISPs). Linux can accommodate existing Microsoft Windows applications and can be dual booted with windows operating systems. It can also be integrated into existing multi-vendor networks –especially Unix based ones-because of their similarities.

Features of Linux

Stability: Linux has proven its stability in many organizations. Many businesses have run a Linux server continuously for more than a year at a time without any problems and without the need to reboot the system. This stability is in part due to the fact that Linux can end a program without affecting other programs or the operating systems as a whole. Another reason for its stability is that the core functionality of Linux, such as how system memory is used, how the hard disk is accessed, and how programs share system resources, has been thoroughly tested by the thousand of people involved in each version of Linux.

Security: The same development process that yielded a highly stable operating system also yielded a very secure operating system. You might be tempted to conclude that an operating system with freely available source code could not possibly be secure. On the contrary, the fact that source code is available to all, and released in a controlled manner by well-known, respected professionals, means that all interested developers can help identify and fix security problems.

Speed: Linux was designed to use limited hardware resources efficiently. As a result, Linux makes better use of hardware resources than almost any other operating system.

Linux operating system could run on a system with only 4 MB of system memory. The efficiency of Linux when operating with such limited resources translates into speed when more extensive resources are available.

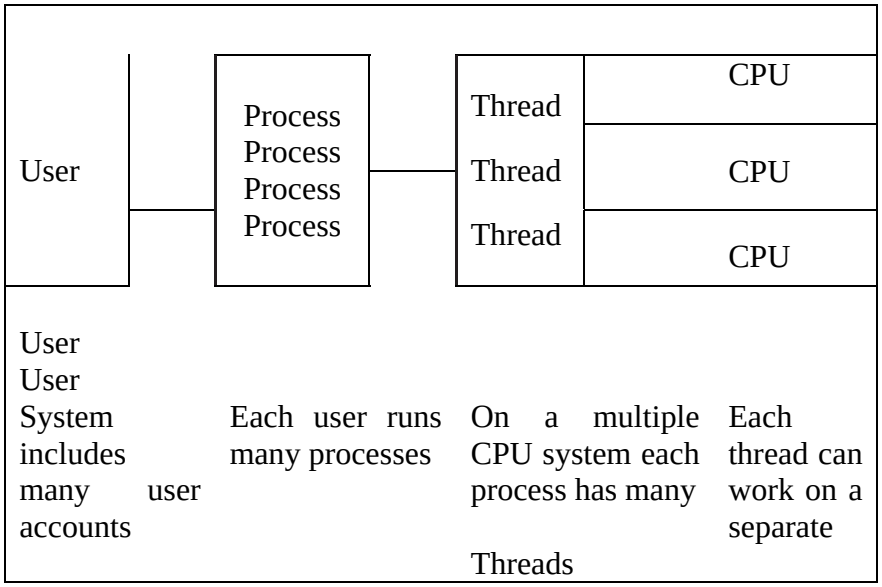
Multitasking, Multi-user, Multiprocessing system: Linux is a true multitasking system, which means that it can run many programs at the same time. A typical Linux system will have 20 to 50 programs running at the same time. Although many operating systems can run multiple programs simultaneously, Linux does this very efficiently. A program can crash without affecting the other programs running on the system. This helps to create a stable system.

Linux manages multiple programs through a technology called preemptive multitasking, in which the Linux kernel controls which program runs at any given moment. Once a program has had a small time to work, the kernel intervenes and gives the control to another program for a time. By contrast, some operating system use cooperative multitasking, in which the kernel is forced to wait for a program to yield the control. Cooperative multitasking can be problematic because it is possible for a poorly written program to crash before the kernel can regain control of the system, thus disabling all other programs running on it.

Linux was also designed as multi-user system, which means that multiple users can log in to the same Linux system over a network connection and run programs. The programs run by one user do not affect the work of other users. A administrative account can configure and control all user accounts.

Linux also supports multiple CPUs on the same computer means support symmetrical multiprocessing. Systems with multiple processors perform faster than single CPU systems because the processors can combine forces to work on one task at the same time. Linux divide the components of a task between multiprocessors via a technique called

multithreading in which program is divided into parts, known as threads. The various threads of one program are then run simultaneously on multiple processors.



(A multi-user, multitasking, multiprocessing operating system)

Flexibility: Linux distributions are extremely flexible because they always include the source code to the operating system, allowing technically oriented system administrators and software developers to modify a system any way they want.

By comparison, operating systems that do not include source code offer little in the way of flexibility. With these systems, administrators are limited to asking the operating system manufacturer for an update that meets a specific need.

Linux is flexible enough to allow you to use old, stable technology that fulfills the job at hand or to experiment with latest trends or features. Flexibility of Linux allows running newer Linux applications on older versions of kernel with little modifications. But in case of other operating systems, there is need to install a new version of operating system to run the latest programs.

Applications: Years ago, Linux was used almost exclusively for developing Unix Software or for specialized Internet severs. But today Linux is used for various applications.

Application	Description
WordPerfect for Linux	Complete, powerful word processor; from corel
APPIXWare	Complete office suite
StarOffice	Office suite and integrated tools; attempt to imitate Microsoft Office
DB2	Powerful database package from IBM
Oracle	The most widely used client/server database
Sybase	A popular client/server database package
Informix	A popular client/server database package
UniCenter TNG	A graphical management console for very large net works; from Computer Associates

4.3 VI EDITOR

Linux provides several text editor programs, from the simplistic vi to the more elegant ones. The vi editor is on almost every system because it requires comparatively little space and still does the job adequately.

Vi is the standard editor in all Unix-related systems. The original vi editor was distributed with the Berkeley Software Distribution; the various Linux distributions use a form of Visual Editor Improved (vim), which claims to be an improvement over the classic vi. When you enter vi filename to invoke the editor, you are actually using a symbolic link to vim. vim then emulates the classic vi editor (an instance of upward compatibility).

The following features are fairly standard across vi versions and types:

- Full-screen editor
- Two modes of operation: Command and Insert
- Use of one-letter commands
- Unformatted text
- Flexible search-and-replace facility with pattern matching
- User-defined editing features using macros

We will describe and use two vi modes: Command and Insert. Some Linux/Unix gurus claim that vi has three modes: Command, Insert, and Last-line. Users are in Last-line mode when they have used Esc to leave Insert mode and then typed a colon (:) so that they can enter specific single-letter commands to quit, save, and so on. In this lesson, we fold the Last-line mode in with Command mode.

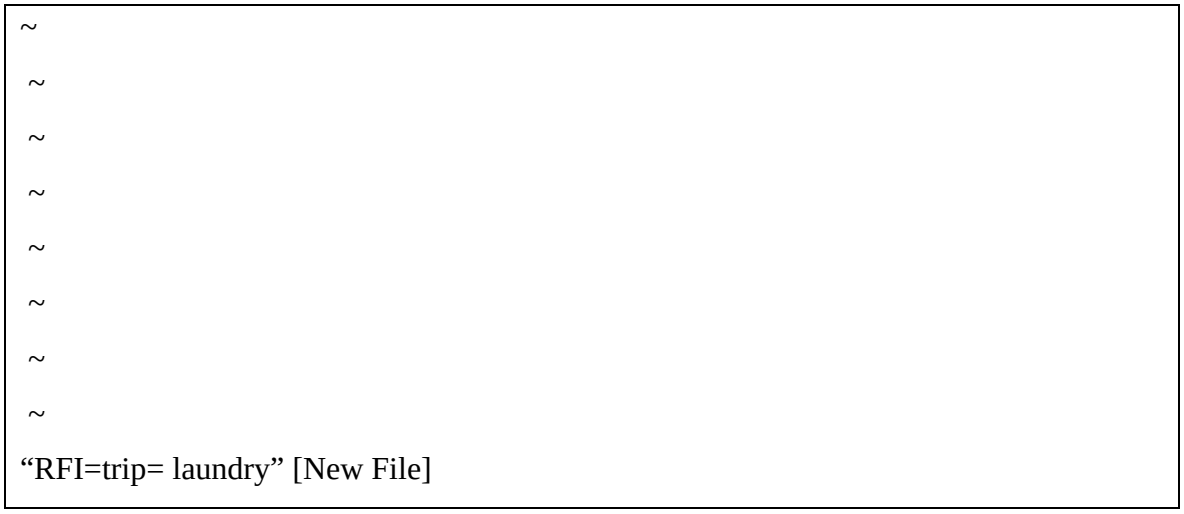
Starting vi

Assuming that you have logged into Linux and are now facing the command-line prompt, enter the following to invoke vi :

$ vi filename<Enter>

If the specified file already exists, vi creates a copy of it and puts the copy into a buffer in the /tmp directory for you to work on. If the file does not exist, vi opens an empty buffer in the same directory and gives the file the name specified in your vi command.

When invoked, vi checks for a file called **.exrc** and incorporates any specifications found there. Then, vi starts in command mode and waits for directions from you. What you see on the screen is a flashing cursor and the filename at the bottom of the screen. If the file is new, the editor tells you so, on the last line. For instance, if you invoke vi to create a new file called RFI_trip_laundry, you would see an empty screen.



(Starting vi)

Dashes represent blank lines, and like we said, the filename appears on the last line. If the file already exists, Linux gives you the filename in double quotes as well as the number of lines and number of characters in the file.

Exiting vi

To exit from the vi editor, you must be in Command mode. To ensure that you are in Command mode before inputting commands at any time, press Esc. You can exit vi in one of the following ways (all end with pressing Enter):

- :q quits vi without saving your file.
- :wq writes changes and quits.
- <Shift>-zz writes changes and quits.

The : q option works only if you have not made any changes. If you have made changes and you try to quit this way, Linux gives you the following message:

No write since last change (use ! to override)

If you want to exit at this point, type one of the following:

- :wq writes changes and quits(as we mentioned earlier).
- :q ! quits without writing changes.
- :x also writes changes and quits.

If you choose :wq or :x, vi will display the updated status of the file (in other words, “filename”, Line count, Character count) then quit and return you to a shell prompt. If, on the other hand, you enter :q!, then vi will simply quit and return you to a shell prompt.

Adding Text in Insert Mode

Let’s say that you open a file in vi and want to enter text. Then, you must be in Insert mode (also known as input mode or text mode). Take the following steps to get into Insert mode and then insert text:

1. Using the up and down arrows, position the cursor at the point in a new or existing file where you want to begin inserting text.
2. Use one of the following single-letter commands:
    - a adds text immediately after the cursor.
    - A (that is, Shift-A) adds text beginning at the end of the line on which the cursor is sitting.
    - i inserts text beginning at the same position presently underlined by the cursor.
    - I inserts text at the beginning of the line on which cursor is sitting.
    - <Insert> also inserts text beginning at the same position presently underlined by the cursor.
3. Add your text.

The difference between a and i will become apparent with practice. Note that while you are adding text, the filename, which was at the bottom of the terminal screen, has been replaced by – INSERT. This message is a reminder that you are in Insert mode. If you

want to save your text as you go, press Esc to re-enter Command mode and then type the following:

: w<Enter>

This action saves the text you have entered thus far, updates the file's status for you ("filename" Line count, Character count), and keeps the file available for input. To resume entering text, use one of the single-letter commands just listed. When you finish adding text, you need to leave Insert mode by pressing Esc. You are then returned to Command mode. Then, you will see that your file is still on the screen, but the filename is no longer displayed at the bottom.

Manipulating Text in Command Mode

The first method you need to know in order to manipulate text is how to move around in the vi editor.

Cursor Movements in vi

Movement with in a line	**Cursor moves**
<left arrow> or h	One character to the left
<right arrow> or l	One character to the right
0(zero)	To the beginning of the line
$	To the end of the line
Movement among words	**Cursor moves**
w	To the next word
b	To the previous word
e	To the end of existing word; if the cursor is already at the end of a word, then it will move to the end of next word
Movement with in the screen	**Cursor moves**

<up arrow> or k	One line up
<Down arrow> or l	One line down
H	To the beginning of the top line on the screen
M	To the beginning of the middle line on the screen
L	To the beginning of the last line on the screen
Movement among screens in same file	**Cursor moves**
<ctrl>-f	Forward to the top of the next screen
<ctrl>-b	Backward to the bottom of the previous screen
Movement with in a file	**Cursor moves**
1G	To the first line of the file
xxG	To the line number xx of the file.
G	To the last line of the file

Deleting text

There are many commands available for deleting text in Command mode. We list several ways of doing so in the following table:

Key sequences for deleting text in vi

Key sequence	**Delete action applied**
x	To a single character
dw	To the end of the current word.
d$	To the end of the line
d0	To the start of the line
dd	The entire line
dG	To the end of the line
u	Undo the last change

U	Restore the entire line (only if the cursor has not left the line)

Searching for Text Strings

When you are in Command mode, pressing the forward slash (/) button automatically puts you in text search forward mode and takes you to the last line on the screen, where vi has placed a forward slash prompt. Similarly, pressing the question mark (?) automatically puts you in text search backward mode and takes you to the last line on the screen, where vi has placed a question mark prompt.

After the prompt, enter the string of text you want to search for and press Enter. The search begins in the chosen direction from the position of the cursor. The cursor stops underneath the first character of the first found text string. If you want to continue searching in the same direction, press n. if you want to search in the opposite direction, press N. eventually, you will reach the bottom or top of the file, and vi will notify you when you do.

To exit from text search, simply enter any other command. You do not press Esc first.

Searching for and Replacing Text

Manually cutting and pasting can be tiresome, inefficient, and occasionally inaccurate. The following is a command for automating it when you are in Command mode in vi:

: g/searchstring / s // replacementstring /g<Enter>

The command works as follows:

- The first g (global) tells the editor to perform a search for the first occurrence of the text string in every line of the file.
- The two forward slashes bracket the search string. Please note that in this case, with / searchstring /, the characters are followed by a space. If the search string is found ending a line or immediately followed by punctuation, it is not replaced.

- The s means substitute.
- The forward slash following the s tells the editor to use the text string preceding the s as the target for the substitution (that is, the preceding text string will be the one replaced).
- The next two forward slashes bracket the text string to be substituted for the replaced text string (in this case, / replacementstring /; again, the characters are followed by a space).
- The last g tells the editor to make the editor to make the substitution at every occurrence in each line found by the first g.

An extra feature we could have added to the syntax example is a c before the last g (that is, / cg). The c tells the editor to ask for confirmation before making each change.

Another of many ways you could modify the command is to make changes to only one line or to the first occurrence on every relevant line.

Moving and Copying Text by Characters and Words

As we mentioned previously, vi utilizes 36 buffers into which you can cut or copy (or, as vi says, yank) file data. Unless you specify a buffer number or letter, the data is cut or copied into buffer 0 by default.

There are also many commands available for copying or moving text in Command mode. We have several ways of doing so as follows:

Key sequence	**Action applied**
cw	Remove all the character to the end of the current word
c$	Remove all the characters to the end of the line
c0	Remove all the characters to the start of the line except the character the cursor is under.

cG	Remove all the characters to the end of the line
u	Undo the last change
U	Restore the entire line (only if the cursor has not left the line)

Moving or Cutting Text Line(s) by Line(s)

The vi editor utilizes 36 buffers (numbered 0 through 9 and lettered a through z) into which you can cut or copy file data. Here is the general line-by-line text moving procedure:

- In command mode, move the cursor to the line you want to move.
- Press dd. The specified line disappears into buffer0 (the default buffer), and the cursor moves to the next line.
- Move the cursor to the line after which you want to place the specified line.
- Press p. the original line appears after the line on which you placed the cursor. If you press P instead of p, the original line appears above the line on which you had placed the cursor. Copying Text Line by Line in vi Here is the procedure for copying text line by line:
- In command mode, move the cursor to the line you want to copy.
- Press yy. The line is copied into buffer 0 (the default buffer), but the original line remains on the screen. The cursor, meanwhile, stays where it was.
- Move the cursor to the line below, which you want to place the yanked line.
- Press p. the yanked copy of the original line appears below the line where you placed the cursor. Now, the cursor moves to the newly placed (in other words, the copied) line.

Executing Linux/Unix Commands in vi

Suppose that while you are working in vi, you realize that you need to exit vi to run a command. But you do not really want to exit vi, run the command, and then re-enter vi. What is your solution? Using the exclamation point (!) command within vi creates an

appropriate shell to execute the chosen command, prompts you for the command, executes it, and displays the results.

4.4 SHELL PROGRAMMING SHELL SCRIPTS

A shell script is:

- A readable text file that can be edited with a text editor like vi.
- A program that contains system commands, variable assignments, flow control syntax, and shell commands
- A program that contains comments to let you or some other reader or developer in the future or in some unknown place know what the objectives, subroutines, and expected outputs were of the script itself.

Thus, a shell script is a collection of system commands stored in a text file that the shell reads and executes in sequence. A script can enable you to do anything that you could normally do from the shell prompt. It will contain at least one Linux/Unix command. When the shell processes a shell script, it reads the script file on command at a time, parses the commands, and sends them to the separating system for execution. The commands are executed in turn, just as if you had typed them at the terminal command line.

Using the cat Command to Create a Simple Shell Script

John wants to execute the date command, he does not want to enter it on the command line every time he wants to invoke it. So, he creates a shell script called script1 using the cat command as he text editor:

$ cat > script1<Enter>
$ date<Enter>
<Ctrl>-d

Executing Shell Scripts

Before executing a shell script, you have to determine two things:

- Can the users access the script file?
- How will the script be executed? Are the permissions on the shell script file set so the script can be executed in the manner you intended?

Can Users Access the Script File?

In order for users to be able to execute the script file, they have to be able to access it. One way to make it accessible is to put the script le in a directory to which there is a path. So, try to make sure its directory is list in the PATH environment variable: yours, your group's or everyone's. If there is no path to the directory, then whoever tries to invoke it will get a message from Linux similar to the following:

bash : script1: No such file or directory

Executing the Shell Script

There are two basic ways to execute a shell script:

- Use the bash command
- Make the script file an executable file all its own

Using the bash command to execute the Script File

At the command prompt type:

$ bash script1

in this script, the first line of the script tells the shell to execute the script using the bash shell. You can also use other shells to execute shells scripts like sh, tcsh, ksh. In case you get the error "No such file or directory" in the first line of the script, there could be an error in the first line. Use the command "whereis bash" to locate the path of bash program and use it in the shell script.

Making Shell Script file executable file

The alternative for script execution relies on the owner of the shell script file making that an “executable” by using the chmod command. Users of the shell script file must have both read and execute permissions for the shell script file. Why? The answer is, because the shell needs to open the script file to read the commands with in it. If you give “group” the execute permission, however, you need not give it to “others” and vice-versa. You can be selective.

The owner can use either of the following two syntax formats:

.$ chmod u+x script1<Enter> (change FAP)

in order to execute a shell script directly at the $ prompt, you can change the File Access Permission (FAP) by granting the execute permission. Once the execute permission is granted, the shell script can be executed directly by typing its name at the $ prompt as follows.

$ script1<Enter>

The echo command

The echo command is used to display messages on the screen. The syntax is$ echo “Message to be displayed”

Example

$ echo “This is an example of the echo command” The echo command displays text enclosed between “ ” on the screen. By default, the echo command displays the text and then puts a new line character at the end of it. The new line character causes the cursor to move to the next line after the text is displayed. You can keep the cursor on the same line using the –n option.

Example

Create a file called hello with the following contents:

Echo “Hello”

Echo “World”

Assign permissions to the file using the command:

$ chmod u+x hello

Inserting Comments

Comments entries can be included in a shell script by prefixing statements with the # symbol. The shell, on encountering #, ignores what follows in that line.

Example

#!/bin/bash

echo “Hello”

#This is a comment line.

Echo “World”

In this example the third line is an example of comment. It would be ignored by shell and would not produce any message.

Variables

In the bash shell, variables do not have to be explicitly declared. They can be created at any point of time by a simple assignment of value.

Syntax

<variable name=<value>

When declaring a variable, there must be no space on either side of the assignment operator(=).

Example

name= “john”

or

name =john

The quotes are optional.

If the value being assigned contains any an embedded space, it should be enclosed within either single or double quotes.

name= “john martin”

num=10

Here num is not a numeric variable. It is a character string. Therefore, the variable num contains the character “10” and not the number 10.

Variables can be created

- either in shell scripts Any variable created within a shell script is lost when the script stops executing
- or at the shell prompt A variable created at the prompt will, however, remain in existence until the shell is terminated by logging out of the system. Logging out from a Linux session will close all the applications that are running, including the login shell.

Referencing Variables

The $ symbol is used to refer to the contents of a variable. The syntax is

Variable=${variable2}

For example we want to assign the value of num variable, declared above, to x variable then the command will be

$ x=$num

If you want another variable x to contain this value concatenated with string “th”(means 10th)
Then the command will be
$ x={$num}th
In case of concatenation, the braces are essential.

Reading a value into a Variable

Read command is used to read the value of variable. On execution, this command waits for the user to enter a value for the variable. When the user presses <Enter> after entering the value, the remaining part of the shell script, if any, is executed. Note that the read command does not prompt the user to enter data. To do so, the echo command must be used.

Example
#!/bin/bash echo “Enter your name.” Read name Echo “Enter your marks.” Read marks
Save this detail into “personaldata” file.

This code accepts the name and marks and stores these values in the “personaldata” file”
Execute the shell script:

$ bash personaldata

Local and Global Shell Variables

When a variable is referenced, only the shell that created it is aware of the variable. When a new shell is created, it is unaware of the variables of the parent shell. Now, the same variable name can be given a different value without the parent shell knowing about it. Such a variable is called a local variable. The following example illustrates the above concepts:

$ name=John

$ echo “$name”	
John	
$ sh	creates a new shell
$ echo “$name”	
	there is no response
$ name=James	Gives a new value, James to name
$ echo “$name”	
James	
Press <Ctrl>d	
$ exit	displays exit and returns to the parent shell
$echo “$name”	
John	Parent is unaware of James
$ sh	Creates a child shell
$ echo “$name”	name does not have any value
Press<Ctrl>d	
$ exit	Displays exit and returns to the parent shell
$ _	

The variables created in a shell are local to the shell that created it, unless specifically made global by the use of the export command. It makes possible for all the child shells to know about the parent shell variables. This is illustrated in the following commands:

$ name=John	
$ export name	
$ echo “$name”	
John	
$ sh	creates a new shell
$ echo “$name”	
John	Child shell has the variable name
$	

$ name=James	Gives a new value, James to name
$ echo “$name”	
James	
Press <Ctrl>d	
$ exit	displays exit and returns to the parent shell
$echo “$name”	
John	Parent shell continues to have the value John
$ _	

The last two commands show that variables can be exported or passed on to child shells, but the reverse is not possible. This is because the export command causes a copy of the variable name and values to be passed on to a child shell. The child shell can change the value of the copy but when it terminates, so does the copy. The original variable remains untouched.

Conditional Execution

The test and [] Command

The test command evaluates and expression and either returns a true (0) or a false (1). The test keyword can also be replaced with [] (square brackets). The syntax for test and [] is as follows:

- Test expression
- [Expression]

The test command can be used to check the values of variables. Some of the operators and options used with string tests are given below:

“$var1” = “$var2”	True, if the strings var1 and var2 are identical
“$var1”! = “$var2”	false, if the strings var1 and var2 are not identical

Alternatively you can use the [ ] instead of the test command to check the values of variables.

[$user_name = “Roger”]

There must be a space on either side of =. There must be a space after [and before]. Multiple conditions can also be tested in one test command. The options are ¾-a (Similar to AND logical operator)

Example
test $var1=$var2 –a $num= “10”This will checks if variables var1 and var2 are equal and variable num has the value 10.

- -o (Similar to OR logical operator)

  test $var1=$var2 –o $num= “10”

  This will checks if variables var1 and var2 are equal or variable num has the value 10.

The if Construct

The if construct is conditional and offers decision-making. It is usually used in conjunction with the test command along with its arithmetic and string operators. The general syntax of this construct is:

if<condition>
then<command(s)>
[else<command(s)>]fi
fi is used to indicate the end of the construct.
Example
Consider the following shell script called Compare.
#!/bin/bash
echo “Enter a number”
read no

echo “Enter another number”

read no1

if [$no –le $no1]

then echo “no is less than or equal to no1.”

else echo “no number is greater than no1.”

fi

Arithmetic test operators

- -eq ---Equal to
- -ne ---Not equal to
- -gt ---Greater than
- -ge ---Greater than or equal to
- -lt ---Less than
- -le ---Less than or equal to

String test operators

Option	***Value***	***Meaning***
string	true	the string is not null.
-z string	true	the length of the string is zero.
-n string	true	the length of the string is non-zero.
string1 = string2	true	the strings are equal.
string1 != string2	true	the strings are not equal.

Example

Consider the following shell script called **CompareStr**.

#!/bin/bash

echo “Enter name”

read name

echo “Enter another name”

```
read name1
if [ $name = $name1]
then echo “both strings are equal.”
else echo “ strings are not equal.”
fi
```

The Exit Command

This is used to stop the execution of a shell script and return to the $ prompt.

Example

```
#!/bin/bash
echo “Enter name”
read name
echo “Enter another name”
read name1
if [ $name = $name1]
then echo “both strings are equal.”
exit
fi
echo “enter one more string.”
```

In the above example, when you both strings having the same value, the script stops execution otherwise the script displays another message to enter one more string.

The case….esac Construct

The case….esac construct is used in shell scripts to perform a specific set of instructions depending on the value of variable and is often used in place of the if construct.

It evaluates a value of the variable and compares it with each value specified. When the value of the variable matches on of the values specified, the set of command(s) written under that value is executed.

The last command to be executed for any value of the variable must be followed by a pair of semicolons to delimit it from the set of commands for the next value.

Example

```
echo “Enter a string”
read strcase
${str} in
John)
echo “hello”
;;
James)
echo “bye”
;;
*)
echo “this means no condition is satisfied”
esac
```

In the above example, the value of str is first compared with value John. If the variable has this value then command for this value will be executed otherwise it compares with the value James. If str1 does not match with John or james then last message will be displayed.

The while Construct

One of the constructs used to support iteration (looping) in shell scripts is the while construct.

The Syntax of this construct is:

```
While <condition>
do
```

```
        <command(s)>
done
```

Example

To display 1 to 10 numbers

```
#!/bin/bash
x=1
while [ $x –le 10 ]
do
        echo $x ( (x=$x+1) )
done
```

The until Construct

The until loop will stop execution when condition becomes true. Its execution is opposite to that of while loop.

Example

To display 1 to 10 numbers

```
#!/bin/bash
x=1
until [ $x –gt 10 ]
do
        echo $x ( (x=$x+1) )
done
```

The for Construct

The for loop takes a list of values as input and executes the loop for every value in the loop. In the for construct, the commands to be executed are specified between the words do ….done.

Syntax is:

```
For((expression1; expression2;expression3))
do
        ….
        ….
done
```

Example

```
#!/bin/bash
for ( (x =1;x<=10;x=x+1) )
do
        echo $x
done
```

The break Command

The break command is used with the while loop. The break command causes the termination of a loop.

Example

```
#!/bin/bash
x=1
while [ $x –le 10 ]
do
        if [ $x –gt 5]
                break
        fi
echo $x ( (x=$x+1) )
done
```

This command displays number from 1 to 5 because when value of x will be incremented to 6 then break statement terminates the while loop.

The continue command

The continue command is used to force a new iteration. When a loop is getting executed, to skip the remaining portion of the loop, the continue command is used.

Example

```
#!/bin/bash x=0
while [ $x –le 10 ]
do
        ( (x=$x+1) )
        if [ $x –eq 5]
                continue
        fi
echo $x
done
```

This command displays number from 1 to 10 except 5 because when value of x will be incremented to 5 then continue statement is executed that takes the control back to the while loop and the echo statement is not executed.

4.5 LINUX COMMANDS

Linux commands generally follow the syntax and format of Unix command. The order and correct separation of elements are important. The name of command always comes first. The command name can then be followed by one or more options that can, in turn, be followed by one or more arguments. You must separate options from the command name and from other options by single spaces. Also, options must be preceded by a hyphen (-). In the following –f and –l are options to their respective commands

[username@hostname “home dir”] $ mail-f newmail<Enter>
[username@hostname “home dir”] $ wc -l filename<Enter>

The first command line reads, “Bring me the contents of my mailbox for processing. Then return the undeleted messages to an alternate mailbox called newmail. Notice that filename newmail is the argument to command mail and its option –f.

The second command line reads. “Count the number of lines in the file called filename.” You can group multiple options together and precede them by a single hyphen. For example,

$ ls –lf<Enter>
This command says,” List the files found in the directory I’m in now, but only the files(not the directories). Also, provide detailed information about those files.” The fmeans “files only,” and the –l means “detailed description.”

If you do not precede an option with a hyphen, the system might try to treat it as an argument instead, which could result in an error message. An argument is a further refinement of the command, usually indicating an object to be retrieved and worked on or an object to be created as a result of the requested process. If you use more than one argument, then each argument must be separated from the option(s) and from other argument(s) by a single space. Unlike options, however, arguments cannot be bunched together.

Directory Commands in LINUX

Directory Contents

Although it is customary to refer to a directory as a type of envelope that contains entire subdirectories, files, and their contents, in truth a directory is a unique type of file that is used to organize other files into a hierarchical structure. Thus, it contains only the information that is needed to access the files or other directories that are affiliated with it

according to some sort of logical order. As a result, a directory occupies less space than other types of files. A directory resembles a table of contents. It lists the names of files and subdirectories and their corresponding inode numbers. When users execute a command to access a file, they use the filename. The system consults the directory to match the filename with its corresponding inode number and then accesses the inode table, which holds information about the file's characteristics (including its location). Then, once the system knows the location of the file, the data can be located.

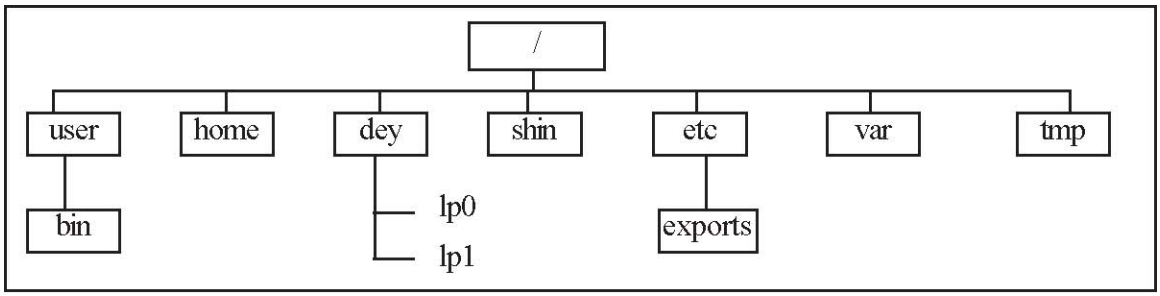
The quixoted Direcory		#	type	mode	link	The inode table			size	loc
name	inode	4	dir	755	1	user	group	Date		
admin	4					quixoted	knightsal	July11 13:17	512	
manuals	10	10	file	644	1	panazasan	knightsa	July 12 10:10	96	

(Directory Contents)

The following categories of information are stored in the inode table:

- The file type(directory or file) and The mode (directory or file permissions)
- Links(which enable you to refer to a file by more than one name)
- The userid of the file's owner
- Group permissions
- The date the file was last accessed and modified
- The file size and its location

Hierarchical Structure A directory tree represents only part of a typical Linux file system. In this depiction, directory names appear in boxes and filenames are unboxed words. The top of the structure is the root(/) directory. The root contains many directories that are critical in system operations.



Path Names

The purpose of a path name is to tell you the location of a file. You write a path name as a string of names separated by forward slashes (/). The rightmost name is the filename and can represent any type of file and other names must be directories. A full path name, which is also referred to as an absolute path name, always begins with a forward slash (/) to indicate that it begins at the root directory. Path name that do not begin with a forward slash are termed relative.

Navigating the Direcory Structure

- **Locate the Working Directory: The pwd command**

The pwd (print working directory) command is used for finding out which directory you are in, where you are in the directory tree. This command always returns the full or absolute path name of the current directory.

Example

[username@userhost James]

$ pwd /home/James

[username@userhost James]

$ _

- **Navigating Directories: The cd command**

The cd command enables you to navigate the directory structure. It changes the current directory to the directory specified.

Example

[username@userhost James]
$ pwd /home/James
[username@userhost James]
$ cd /usr/bin
[username@userhost bin]
$ pwd /usr/bin
[username@userhost bin]
$ _

Note that the complete path name has been specified with the cd command. Linux also allows the use of relative path names with commands.

Example
[username@userhost /usr]
$ pwd /usr
[username@userhost /usr]
$ cd bin
[username@userhost bin]
$ pwd /usr/bin
[username@userhost bin]
$ _

In the above example, the user James has changed the working directory from /usr to the directory /usr/bin.

You can also use the .. (double dot) option with the cd command to move to the parent directory of your current directory. For example, James can change to the parent directory of his HOME directory by entering the following command after logging on.

Example

[username@userhost James]
$ pwd
/home/James
[username@userhost James]
$ cd ..
[username@userhost /home] $ pwd
/home
[username@userhost /home]
$ cd ..
[username@userhost /]
$ pwd/[username@userhost /]
$ _

The two dots refer to the parent directory of the current directory. The cd command without any path name always takes a user back to the HOME directory.

Example
[username@userhost bin]
$ pwd /usr/bin
[username@userhost bin]
$ cd
[username@userhost James]
$ pwd /home/James
[username@userhost James]
$ _

recollect that the tilde(~) sign is used to denote the full path for a HOME directory. Assume there are two directories sub1 and sub2 under James's HOME directory.

Example

[username@userhost etc]

$ pwd /etc

[username@userhost etc]

$ ~/sub1

[username@userhost sub1]

$ pwd /home/James/sub1

[username@userhost sub1]

$ cd ~

[username@userhost James]

$ pwd /home/James

[username@userhost James]

$ _

Managing the Directory Structure

- **Creating a Directory: The mkdir command**

The mkdir (make directory) command is used to create directories.

Example

[username@userhost James]

$ mkdir James1

[username@userhost James]

$ _

The sub-directory James1 is created under the current directory.

- **Deleting a Directory: rmdir Command**

The rmdir (remove directory) command removes the directory specified.

Example

[username@userhost James]

$ rmdir James1

[username@userhost James]

$ _

Here, James1 directory is deleted.

- **Listing Directory Contents: ls Command**

The ls command is used to display the names of the files and sub-directories in a directory.

Example

[username@userhost James]

$ ls/home/James

In above example, all files and directories under the James directory are listed. If the files and directories under the current directory are to be listed, it is optional to specify the directory name with ls. To get the types of files along with file names, use –l option. The –l option when used with ls displays a detailed list of files and directories.

File command in Linux

A file is a collection of data or a stream of characters. A typical file can contain either text (which can be displayed or printed) or code data (more commonly called a binary file, which can also be executable). In Linux everything is represented as file - even system devices. Linux does not impose any internal structure on the contents of a file, nor does it require the file to have any specific attributes. Only the application or tool is concerned with a file's structure and contents.

Linux can recognize three major file types: ordinary files, directories, and special files. All information about a file, except its contents, is stored in the file's inode.

- **Cat command:**

The cat command displays the contents of the specified file. The cat command can be used to vertically concatenate the contents of more than one file.

Example
[username@userhost James] $ cat filel
A sample file
[username@userhost James]
$ _

The command assumes that the file filel is in the current directory. Complete path names can also be specified to display a file in another directory. The cat command can also display more than one file, as shown in the following command:

Example
[username@userhost James] $ cat filel file2 A sample file Another sample file [username@userhost James] $ _

- **copying files: cp command**

The cp command duplicates the contents of the source file into a target file.

Syntax: cp[options]<source file/s> <destination directory/file>

Example
[username@userhost James]
$ cp filel file2

in the above example, the contents of filel are copied to a new file file2. if file2 already exists, its contents will be overwritten by the contents of file1. Complete path names can be specified with the cp command to copy files across directories.

You can also copy a directory recursively using the cp command with the –r option.

Example
[username@userhost James]
$ cp -r dir1 dir2

the above command copies the dir1 directory and all its files and sub-directories to the dir2 directory. If the directory, dir2, exists, all the contents are added to that directory, otherwise dir2 is created in the current working directory. The other common options and their functions are given below:

Option	Function
-i	prompts before overwriting
-l	links a file instead of copying it
-s	creates a symbolic link
-v	Verbose-explains what is being done, in detail

- **removing files: rm command**

The rm command is used to delete files or directories.

Syntax: rm [options] file/s

Example
[username@userhost James] $ cp -r file1 file2
the above command will remove the files data1 and data 2 from your current directory.

If the file to be deleted is not located in the current directory, the complete path name has to be given.
Example

[username@userhost James] $ rm/home/James/filel The –r option is used with the rm command to remove a directory along with its subdirectories.

Example
[username@userhost James]
$ rm –r dir1

the above command removes the dir1 directory along with all its sub-directories. The other commonly-used options with the rm command are shown in the table below.

Option	Function
-i	Prompts before removing.
-f	Removes a file by force. It ignores the non-existence of a file, that is, if the file does not exist, the command does not flag an error.
-r or –R	Deletes recursively, that is, deletes a directory along with its sub-directories.
-v	Verbose- explains what is being done, in detail.

- **Moving and Renaming files: mv command**

The mv command is used to move a file or directory from on location to another or to change its name. Note that moving a file to another location is different from copying it. While moving a file, no new file is created.

Syntax:
mv[option] source destination

Example
[username@userhost James]
$ mv dir1 directory1

in the above example, the dir1 Directory is renamed to directory1. A file can be moved to another directory also.

Example
[username@userhost James]
$ mv file1 /home/James/dir1

in the above example, the file data3 is moved from the current directory to the /home/James/dir1 directory.

Example
[username@userhost James]
$ mv dir1 dir2

In the above example, the directory dir2 exists in the current directory; therefore, the dir1 directory is moved from the current directory to the dir2 directory. Some of the options available with the mv command are given below.

Option	Function
-f	if the file exists at destination, it overwrites it without prompting
-i	interactive, prompts before overwriting at the destination location
-v	Verbose—explains what is being done, in detail

- **Displaying the Contents Page-wise:**

The cat command is used to display the contents of a file on the screen. However, if the file being displayed is large, then the entire contents will scroll up the screen. To view the file one screen-full at a time, you can use the more or less command.

The more command is used to display data one screen-full at a time. While viewing a file using the more command, once you have scrolled down, you cannot move up.

Syntax:
more [options] <filename>

Example
[username@userhost James]
$ more file1

the above command will display a page-wise listing of the contents of the file, file1.The less command is similar to the more command except that you can scroll upwards also while viewing the contents of a file. The less command is also a little faster than the more command.

Syntax:Less [options] <filename>

Example
[username@userhost James]
$ less file1

The above command will display a page-wise listing of the contents of the file file1. To move and down the screen, you can use the arrow keys. You can also specify a number to move down the screen by that number of lines. To quit the display, you have to type ‘q’.

- **Wildcard Characters**

The shell offers the facility to perform an operation on a set of files without having to specify all the names of the files on which the operation is to be performed. This is made possible by the use of certain special characters in the command in place of the actual file name. The shell interprets these special characters as a specific pattern of characters. It then compares all the file names under the directory specified in the command to find the

file names that match the pattern. The command is executed on the files with names that match the pattern.

The following table lists the wildcards available, with a description of each.

Character	Purpose
*	Matches none or one character or a string of characters
?	Matches exactly one character
[]	Matches exactly one of a specified set of character

The * Wildcard

The * wildcard is interpreted as a string of none, one , or more characters.

Example

[username@userhost James] $ ls c*

the above command displays all files whose names start with ‘c’. the * wildcard can also be repeated in the command line.

Example

[username@userhost James]

$ ls c*.*

The above command displays all files whose names start with ‘c’ and containing any sequence of characters, followed by dot, and then followed by any sequence of characters.

The ? Wildcard

The ? wildcard matches exactly one occurrence of any character.

Example

[username@userhost James]

$ ls *.?

The above command displays all files having any character(s) before a dot, followed by a single character after the dot.

The [] Wildcard

The [] wildcard can be used to restrict the characters to be matched.

Example

[username@userhost James]

$ cat a[123]

This displays the contents of the files with two character file names starting with a and with the next character as 1,2,3 for example,a1,a2 and a3.

4.6 SUMMARY

An operating system is the program that contains a set of core functionality for other programs, providing both the interface between other programs and the hardware, and the interface between other programs and the user sitting at the computer.

Operating system performs following functions:

- Memory management
- File and software management
- Input/Output and peripheral management
- CPU management

Major parts of an operating system are:

Kernel: The core of this operating system, which schedules when programs can use computer resources and interfaces directly with core components of the computer hardware, such as memory and hard disks.

Device drivers: special software that provides access to additional hardware beyond core device support provided by the kernel.

Utility Programs: special software that helps manage the hardware and operating system features

Graphical Interface: the program that provides mouse driven applications with menu bars, buttons and so forth.

Linux

Linux is free Unix type operating system. It enables multitasking, simultaneous multiple users, the sharing of a system libraries for efficiency, TCP/IP networking, virtual memory and swap spaces, and other Unix OS features. Users can use a GUI or the command line.

VI Editor

Linux provides several text editor programs, from the simplistic vi to the more elegant ones. The vi editor is on almost every system because it requires comparatively little space and still does the job adequately.

Vi editor has two modes: Command and Insert. Some Linux/Unix gurus claim that vi has three modes: Command, Insert, and Last-line. Users are in Last-line mode when they have used Esc to leave Insert mode and then typed a colon (:) so that they can enter specific single-letter commands to quit, save, and so on.

Shell scripts

Shell script is a collection of system commands stored in a text file that the shell reads and executes in sequence. A script can enable you to do anything that you could normally do from the shell prompt.

4.7 KEYWORDS

ISP: Internet Service Provider, the companies which provides internet services to the users

MULTITASKING: Multitasking means that running many programs at the same time

MULTIPROCESSING SYSTEM: That has the ability o support more than one process at the same time i.e., allowing more than one program to run concurrently on multiple CPUs in a single computer system

KERNEL: The core of this operating system, which schedules when programs can use computer resources and interfaces directly with core components of the computer hardware

GUI: Graphical User Interface

TCP/IP: Transmission Control Protocol/Internet Protocol, the basic protocol used for accessing internet

VI: a famous edition in Unix/Linux for writing shell scripts or creating text files

SHELL SCRIPT: Shell script is a program that contains system commands, variable assignments, flow control syntax, and shell commands

4.8 SUGGESTED READINGS

Nicholas Wells, Linux Installation and Administration, Vikas Publishing House

Matt Welsh, Matthias Kalle, Running Linux, Shroff Publishers and Distribution

LESSON: 5

CRYPTOGRAPTHY, DIGITAL SIGNATURE AND CYBER LAW

5.0 OBJECTIVE

After going through this lesson, you will be able to:

- Define the concept of cryptography
- Define the Data Encryption Standards
- Explain the Private and public key mechanism of cryptography

- Describe the significance and working of digital signatures
- Know the major characteristics of IT Act 2000

5.1 INTRODUCTION

When you make a purchase in a shop or superstore, usually you pay by credit/debit card, in cash or sometimes by cheque. Since you are making payment yourself, you are usually sure that nothing is going wrong. However, there are always chances that your credit card (when lost) or just the credit card number (even when not lost) can be misused. Someone may attempt to make the payee believe that the credit card belongs not to you, but to him.

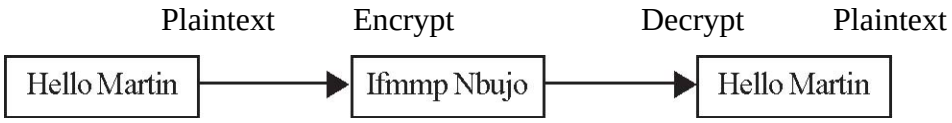
The dangers of making payments can become much more serious when you are dealing with a merchant on the Internet. The dangers are actually not restricted to payments. They can take other forms. In this chapter we will explore the mechanism that have been in use to tackle the dangers involved in the e-commerce transaction such as Cryptography and Digital signature and The Information Technology Act that aims to provide the legal framework under which legal sanctity is accorded to all electronic records and other activities carried out by electronic means.

5.2 CRYPTOGRAPHY

Cryptography is a technique of encoding and decoding messages, so that cannot be understood by anybody except the sender and intended recipient. For instance, you can have a convention wherein Ifmmp Nbsujo actually means saying hello to a friend Martin (that is, Hello Martin!). Here each alphabet of the original message (i.e. H,e,l,l,o) is changed to next immediate alphabet(I,f,m,m,p).Thus Hello becomes Ifmpp and Martin becomes Nbsujo.

Cryptography uses the same basic principle. The sender and recipient of the message decide on the encoding and decoding scheme and use it for communication. In technical terms, the process of encoding messages is known as encryption. As we know the original message text called as plain text. When it is encrypted, it is known as cipher text.

The recipient understands the meaning and decodes the message to extract the correct meaning out of it. This process is known as decryption.



(Encryption decryption process)

Encryption is the mutation of information in any form (text, video, graphics) into a representation unreadable by anyone without a decryption key. Suppose Martin wants to send you a message but does not want anyone but you to read it. Martin can encrypt the message, which means that martin can scramble it in a complicated way, rendering it unreadable to anyone except you, the intended recipient. Martin then supplies a cryptographic "key" to encrypt the message, and you have to use the same key to decrypt it. These are the basics of single key cryptography. For example A wishes to send a called the plaintext, with an encryption key and sends the encrypted purchase order, called the cipher text, to B.B decrypts the cipher text with the decryption key and reads the purchase order but without the decryption key it is impossible to recover the cipher text into plaintext by any hacker.

5.21 Types of Cipher text

The generation of cipher text from plaintext itself can be done in two basic ways, as follows:

(1) Stream ciphers: In this approach, the plain text is encrypted one bit at a time. Suppose the original message (plaintext) is Pay 100 in ASCII. When we convert these ASCII characters to their binary value, let us assume that it translates to 01011100. Actually it will be 56 bits!) Suppose the key to be applied is 10010101 in binary. Let us also assume that we apply the XOR logic as the encryption algorithm.

In normal format	In binary format	
Pay 100	01011100 10010101	Plaintext XOR operation with the key
ZTU 91 ^%	11001001	

(Stream ciphers)

The logic of XOR can be summarized as:

0+0=0

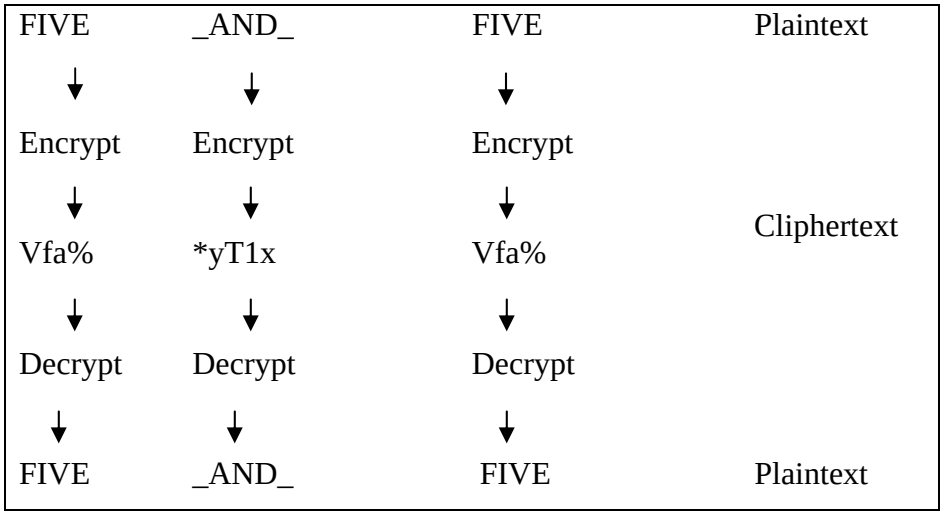
0+1=1

1+0=1

1+1=0

When we apply this XOR logic column by column on two bits (1 from plaintext and 1 from key), we get the cipher text. (Actually it will be 56 bits!) When we convert it back to ASCII, we may get ZTU91 ^%. Thus, what is transmitted is 11001001 in binary, which even when translated back to ASCII would mean ZTU91 ^%. This makes no sense and thus protects the information.

(2) Block Ciphers: In this scheme, rather than encrypting one bit at a time, a block of bits is encrypted together. Suppose we have a plaintext FIVE_AND_FIVE that needs to be encrypted. Using block cipher, FIVE could be encrypted first, followed by _AND_, then finally FIVE. Thus, one block of characters gets encrypted at a time. During decryption, each block would be translated back to the original form. In actual practice, the communication takes place only in bits. Therefore, FIVE actually means the binary equivalent of the ASCII characters FIVE. After any algorithm encrypts these, the resultant bits are converted back into their ASCII equivalents. Therefore we get funny symbols such as Vfa% etc. that protects information. In actual practice, their binary equivalents are sent and received, which are decrypted back into their binary equivalent of ASCII FIVE.



Block Ciphers

There are two types of Cryptography

- Secret key cryptography (Symmetric key cryptography)
- Public key cryptography (Asymmetric key cryptography)

Secret Key Cryptography

Secret key cryptography involves the use of a shared key for both encryption by the transmitter and decryption by the receiver. Shared key techniques suffer from the problem of key distribution since shared keys must be securely distributed to each pare of communicated parties. Secure key distribution becomes cumbersome in large networks. Suppose A encrypts a message with a secret key and e-mails to the encrypted message to B. On receiving the message, B checks the header to identify the sender, then unlocks his electronic key storage area and takes out the duplicate key of secret key. B then uses a secret key to decrypt message.

The main part of secret key cryptography is getting the sender and receiver to agree on the secret key without a third party finding out. This is difficult because if A and B are in separate sites, they must trust not being overhead during face-to-face meetings or over a

public messaging system when the secret key is being exchanged. Anyone who intercepts the key in transit can later read all the encrypted messages using that key.

The generation, transmission and storage of keys is called key management. All cryptosystems must deal with key management issues. Although the secret key is feasible and practical for one-to-one document interchange but it does not scale. In a business environment where a company deals with thousands of online customers, it is impractical to assume that key management will be flawless. Hence, we can safely assume that secret key cryptography will be a dominant player in e-commerce given its difficulty providing secure key management.

5.3 DATA ENCRYPTION STANDARD (DES)

Data Encryption Standard is a secret-key, symmetric cryptosystem. When used for communication both sender and receiver must know the same secret key which is used both to encrypt and decrypt the message. DES can also use for single user encryption, for example, to store files on a hard disk in encrypted form. But in a multi-user environment secure key cryptography becomes difficult.

DES operates on 64-bit blocks with a 56-bit secret key. Its operation is relatively fast and works well for large bulk documents or encryption. Instead of defining just one encryption algorithm, DES defines a whole family of them. A different algorithm is generated for each secret key this means that everybody can be told about the algorithm and your message will still be secure. You just need to tell others your secret key, a number less than 256. The number 256 is also large enough to make it difficult to break the code using a brute force attack. A new technique for improving the security of DES is triple encryption (Triple DES), which encrypts each message using three different keys in succession. Triple DES, thought to be equivalent to doubling the key size of DES, to 112 bits, should prevent decryption by third party capable of single key exhaustive search.

Public key Cryptography

This cryptography system involves the use of public keys. Public key technique involves a pair of keys: a private key and a public key associated with each user. Information encrypted by private key can be decrypted only using the corresponding public key. The private key is used to encrypt the transmitted information by the user, is kept secret. The public key is used to decrypt the information at the receiver and is not kept secret. Since only the bona fide author of an encrypted message has knowledge of the private key, a successful decryption using the corresponding public key verifies the identity of the author and ensures message integrity.

Each party to public key pairing receives a pair of keys, the public key and private key. When A wishes to send a message to B then A looks up B's public key in the directory, A then uses the public key to encrypt the message and mail it to B.B uses the secret private key to decrypt the message and read it. Anyone can send an encrypted message to be but B can read it. Unless, a third party say C has access to B's private key, it is impossible to decrypt the message sent by A. This ensures confidentiality.

Advantage of public key cryptography is that no one can find out the private key from the corresponding public key. Hence, the key management problem is mostly confined to the management of private keys. The need for sender and receiver to share secret information over public channels is completely eliminated. All transactions involve only public keys and no private key is ever transmitted or shared. The secret key never leaves the user's PC. Thus a sender can send a confidential message merely by using public information and that message can be decrypted only with a private key in the sole possession of the intended recipient.

5.4 RSA AND PUBLIC KEY CRYPTOGRAPHY

RSA is a public key cryptosystem for both encryption and authentication developed in 1977 by Ron Rivest, Adi Shamir and Leonard Adleman. This system uses a matched pair of encryption and decryption keys, each performing a one-way transformation of the data.

The security of RSA is based on the fact that it is extremely difficult even for fastest computers to factor large numbers that are the products of two prime numbers (keys), each greater than 2512. RSA is important which enables digital signature, which can be used to authenticate electronic documents the same way handwritten signatures are used to authenticate paper documents. Suppose sender X runs a program that uses a hash algorithm to generate a digital fingerprint – a pattern of bits that uniquely identifies a much larger pattern of bits for the document and encrypts the fingerprint with his private key.

This is X's digital signature, which is transmitted along with the data .Y receiver decrypts the signature with X's public key and runs the same hash program on the document. If the digital fingerprint by the hash program does not match the fingerprint sent by X then the signature is invalid. If the fingerprints do match then Y can be quite sure that the digital signature is authentic. If the document were altered on the route, the fingerprints will not match and the receiver will know that data tampering occurred. If the sender's signature has been forged, the fingerprints won't match either. Therefore the digital signature verifies both the identity of the sender and the authenticity of the data in the document.

5.5 MIXING RSA AND DES

RSA allows two important functions not provided by DES:

- Secure key exchange without prior exchange of keys
- Digital signatures

For encrypting messages RSA and DES are usually combined as follows First the message is encrypted with a random DES key then before being sent over an insecure communication channel, the DES key is encrypted with RSA. Together, DES encrypted message and RSA encrypted DES key are sent. This protocol is known as RSA digital envelope.

Charactertistic	Symmetric key cryptography	Asymmetric key cryptography
Key used for encryption/ decryption	Same key is used for encryption and decryption	Different keys are used forencryption and decryption
Speed of encryption/ decryption	Very fast	Very slow
key exchange	A big problem	No problem at all
Number of keys required as compared to the number of participants in the message exchange	Equals about the square of the number of participants, so scalability is an issue	Same as the number of participants so scalesup quite well
Usage	Mainly for encryption and decryption (confidentially), can not be used for digital signatures (integrity and non-repudiation)	Can be used for encryption and decryption (confedent-iality) as well as for digital signature (integrity and non-repudiation.

(Symmetric Versus Asymmetric Key Cryptography)

5.6 DIGITAL SIGNATURES

In the case of business transaction authentication refers to the use of digital signature, which play a function for digital document similar to that played by handwritten signature for printed documents. The signature is an unforgeable piece of data asserting that a named person wrote or otherwise agreed to the document to which the signature is attached.

The recipient as well as third party can verify that the document indeed originate from the person whose signature is attached and that document has not been altered since it was signed. A secure digital signature system consists of two parts:

- A method of signing a document such that forgery is not possible
- A method verifying that a signature was actually generated by whomever it represents.

Furthermore, Secure digital signatures cannot be repudiated, that is, signer of a document cannot later disown it by claiming it was forged. Suppose A uses public key cryptography to digitally signed a document and then put his private key and the document together and performs a computation on the composite (document + key) to generate a unique number called the digital signature. For instance, when an electronic document such as an order form with a credit card number, is run through the method, the output is unique "fingerprint" of the document.

This "fingerprint" is attached to the original message and further encrypted with signer A's private key. The result of the second encryption is then sent to b, who then first decrypts the document using A's public key. B checks whether the message has been tampered with or is coming from a third party C, posing as A. to verify the signature, B does some further computation involving the original document, the purported signature, and A's public key. If the result of computation generates a matching "fingerprint" of the document, the digital signature is verified as genuine otherwise the signature may be the fraudulent or the message altered and they are discarded.

Step by step procedure of Digital signature

1. A encrypts the original plaintext message (PT1) into ciphertext (CT1) using B's public key.

2. Rather than sending the ciphertext to B, A runs an algorithm on the original plaintext to calculate a Message Digest (MD), also known as hash, which is MD1 here. This algorithm simply takes the original plaintext in the binary format, performs the hashing algorithm (i.e. generating a fingerprint) producing a string of binary digits, which can be treated as a small text in an unreadable format. This hashing algorithm is public, which means that anyone can use it. The most popular message digest algorithms are MD5 and SHA-1. Next, A encrypts the message digest with its own private key. The output of step 2 is A's digital signature (DS1).
3. A now concatenates the ciphertext CT1 and A's digital signature DS1. This composite is sent over the network to B. It like signing a document before faxing it.
4. B receives the ciphertext (CT1) and A's digital signature DS1 , as CT2 and DS2 respectively. B has to decrypt both of these. B first decrypts the ciphertext back to plaintext PT2 with its own private key.
5. B now wants to ensure that the message indeed came from A and not from someone who has trying to impersonate A. For this purpose B takes A's original signature and decrypts it with A's public key. This gives B the message digest as was generated by A (MD2).
6. Next, B calculates its own message Digest MD3.by using the hash algorithm on the decrypted message of step 4.
7. From steps 5 and 6, B has two message digests. If the two matches, B can be sure that message came indeed from A otherwise message is coming from someone who is posing as A.

5.7 CYBER LAW

Cyber law governs the legal issues of cyberspace. The term cyberspace is not restricted to the Internet. It is very wide term that includes:

- Computers

- Computer networks
- The Internet
- Data
- Software etc.

Cyber law encompasses laws relating to:

5.72 Electronic and digital signatures

5.73 Computer crime

5.74 Intellectual property

5.75 Data protection and privacy

5.76 Telecommunication Laws

5.71 Electronic and Digital signatures:

Electronic signatures (especially digital Signatures) are fast becoming the defacto standard for authentication of electronic records, Electronic records, Electronic Data Interchange, Emails etc.

Comprehensive laws are required so that uniform standards and procedures can be established. These laws relating to Electronic signatures e.g.: the Electronic Signatures in Global and National Commerce Act of the USA, are part of cyber law.

5.72 Computer Crime Law:

Outgrowing dependence on computers and the Internet has made us all potential victims of Internet threats. Some countries have enacted legislations that specifically deal with computer crime and yet others have adapted their existing laws to make computer crime an offence existing statutes.

These laws are under the gambit of cyber law.

5.73 Intellectual Property Law:

Cyber law covers the intellectual property laws that relate to cyber space and its constituents. This includes:

- Copyright law in relation to computer software, computer source code etc.
- Trademark law in relation to domain names
- Semiconductor law, which relates to the protection of Semiconductor design and Layouts
- Patent law in relation to computer hardware and software

5.74 Data Protection and Privacy laws:

Many nations have enacted legislation relating to data protection and privacy within their jurisdictions. It is pertinent to note that due to the nature of the Internet and the amount of information that may be accessed through it, such legislation is critical to protect the fundamental rights of privacy of an individual. These laws would probably play a vital role, as the dependence on insecure networks such as the Internet grows further.

5.75 Telecommunication Laws:

Telecommunication systems also fall within the ambit of cyberspace and therefore would form an integral part of cyber laws.

5.8 CYBER LAWS IN INDIA

In May 2000, both the houses of the Indian Parliament passed the Information Technology Bill. The Bill received the assent of the President in August 2000 and came to be known as the Information Technology Act, 2000.Cyber laws are contained in the IT Act, 2000.

This Act aims to provide the legal infrastructure for e-commerce in India. and the cyber laws have a major impact for e-businesses and the new economy in India. The information Technology Act, 2000 also aims to provide for the legal framework so that

legal sanctity is accorded to all electronic records and other activities carried out by electronic means.

Objectives of the Act:

The objectives of the act are:

- To grant legal recognition for transactions carried out by means of electronic data interchange and other means of electronic communication commonly referred to as "Electronic Commerce" in place of paper based methods communication.
- To give legal recognition of Digital signature for authentication of any information or matter which requires authentication under any law.
- To facilitate electronic filling of documents with Government departments.
- To facilitate electronic storage of data.
- To facilitate and give legal sanction to electronic fund transfers between banks and financial institutions
- To give legal recognition for keeping books of account by Bankers in electronic form.
- To amend the Indian Panel code, the Indian Evidence Act, 1872, the Banker's Book evidence Act, 1891 and the Reserve Bank of India act, 1934. The detailed act is discussed below.

5.9 INFORMATION TECHNOLOGY ACT, 2000

SCOPE OF THE ACT AND DEFINITIONS [CHAPTER 1]

This Act is called the Information technology act, 2000. The act shall not apply to the following:

- A negotiable instrument as defined in section 13 of the Negotiable Instruments Act, 1881
- A power-of-attorney as defined in section 1A of the Powers-of-Attorney act, 1882
- A trust as defined in section 3 of the Indian Trusts act, 1882

- A will as defined in Section (h) of section 2 of the Indian Succession Act, 1925 including any other testamentary disposition by whatever name called.
- Any contract for the sale or conveyance of immovable property or any interest in such property.
- Any such class of documents or transactions as may be notified by the Central Government in the Official Gazette.

Arrangement of sections:

The act consists of 94 sections spread over thirteen chapters, and four schedules to the Act. The Schedules of the Act contain related amendments made in other acts as outlined in the objectives of the Act, namely, the Indian Panel Code, the Indian Evidence act, 1972, the Banker's book Evidence act, 1891 and The Reserve bank of India, 1934.

AUTHENTICATION OF ELECTRONIC RECORDS USING DIGITAL SIGNATURES [CHAPTER-II]

This chapter gives the legal recognition to electronic records and digital signatures. It contains only section 3.

This section provides the conditions subject to which an electronic record may be authenticated by means of affixing digital signature. The digital signature is created in two distinct steps.

- First the electronic record is converted into a message digest by using a mathematical function known as hash function, which digitally freezes the electronic record thus ensuring the integrity of the content of the intended communication contained in the electronic record. Any tampering with the contents of electronic document will immediately invalidate the digital signature.
- Secondly, the identity of the person affixing the digital signature is authenticated through the use of private key which attaches itself to the message digest and which can be verified by anybody who has the public key corresponding to such

private key. This will enable anybody to verify whether the electronic record is retained intact or has been tampered with since it was so fixed with digital signature. It will also enable a person who has a public key to identify the originator of the message.

ELECTRONIC GOVERNANCE [CHAPTER III]

This chapter specifies the procedures to be followed for sending and receiving of electronic records and the time and the place of the dispatch and receipt.

This chapter contains sections 4 to 10.

Section 4 -this section provides for "legal recognition of electronic records". It provides that where any law requires that any information or matter should be in the typewritten or printed form then such requirement shall be deemed to be satisfied if it is an electronic form.

Section 5 -this section provides for "legal recognition of Digital Signatures". Where any law requires that any information or matter should be authenticated by affixing the signature of any person, then such requirement shall be satisfied if it is authenticated by means of digital signatures affixed in such manner as may be prescribed by the Central Government.

For the purposes of this section, "signed", with its grammatical variations and cognate expressions, shall, with reference to a person, mean affixing of his hand written signature or any mark on any document and the expression "signature" shall be construed accordingly.

Section 6 lays down the foundation of Electron Governance. It provides that the filing of any form, application or other documents, creation, retention or preservation of records, issue or grant of any license or permit or receipt or payment in Government Offices and

its agencies may be done through the means of electronic form. The appropriate government has the power to prescribe the manner and format of electronic records and the method of payment of fee in that connection.

Section7 -this section provides that the documents, records or information, which has to be retained for any specified period, shall be deemed to have been retained if the same is retained in the electronic form provided the following conditions are satisfied:

- The information therein remains accessible so as to be usable subsequently
- The electronic record is retained in its original format or in a format, which accurately represents the information, contained
- The details, which facilitate the identification of the origin, destination, dates and time of dispatch or receipt of such electronic record, are available therein.

This section does not apply to any information, which is automatically generated solely for the purpose of enabling an electronic record to be dispatched or received.

Section 8 provides for the publication of rules, regulations and notifications in the Electronic Gazette. It provides that where any law requires publication of any rule, regulation, order, bye-law, notification or any other matter in the Official gazette, the such requirement shall be deemed to be satisfied if the same is published in electronic form. It also provides where the Official gazette is published both in the printed as well as in the electronic form, the date of publication shall be the date of publication of the official gazette, which was first published in any form.

Section 9 of the act provides that the conditions stipulated in sections 6, 7 and 8 shall not confer any right to insist that document should be accepted in an electronic form by ministry or department of central government or the state government.

Power to Central Government to make rules [section 10]

This section provides that the central government, in respect of digital signature may prescribe by rules the following:

- The type of digital signature
- The manner and format in which the digital signature shall be affixed.
- The manner or procedure, which facilitates identification of the person affixing digital signature
- Control processes and procedures to ensure integrity, security and confidentiality of electronic records or payments.
- Any other matter which is necessary to give legal effect to digital signatures.

ATTRIBUTION, RECEIPT AND DISPATCH OF ELECTRONIC RECORDS [CHAPTER IV]

This chapter deals with attribution, receipt and dispatch of electronic records. Attribution means to consider it to be written or made by someone. Hence, this section lays down how an electronic record is to be attributed to the person who originated it. This is given in section 11.

Section 12 provides for the manner in which acknowledgement of receipt of an electronic record by various modes shall be made.

Section 13 provides for the manner in which the time and place of dispatch and receipt of electronic record is deemed to be dispatched at the place where originator has his place of business and received where the addressee has his place of business. For the purpose of this section:

- If the originator or the addressee has more than one place of business, the principal place of business shall be the place of business.
- If the originator or the addressee does not have a place of business, his usual place of residence shall be deemed to be the place of business.

- “usual place of residence”, in relation to a body corporate, means the place where it is registered.

SECURE ELECTRONIC RECORDS AND SECURE DIGITAL SIGNATURES [CHAPTER V]

This chapter sets out the conditions that would apply to qualify electronic records and digital signatures as being secure. It contains sections 14 to 16.

Section 15 provides for the security procedure to be applied to the Digital Signatures for being treated as secure digital signature.

Section 16 provides for the power of the Central government to prescribe the security procedure in respect of secure electronic records and secure digital signatures.

REGULATION CERTIFYINGAUTHORITIES [CHAPTER VI]

This chapter contains detailed provisions relating to the appointment and powers of the Controller and Certifying Authorities. It contains sections 17 to 34. Section 17 provides for the appointment of Controller and other officers to regulate the certifying authorities.

Section 18 lays down the functions which Controller may perform in respect of activities of Certifying Authorities.

Section 19 provides for the power of the Controller with the previous approval of Central government to grant recognition to foreign Certifying Authorities subject to such conditions and restrictions as may be imposed by regulations.

Section 20 provides that the Controller shall be acting as repository of all Digital Signature Certificates issued under the Act. He shall also adhere to certain security procedure to ensure secrecy and privacy of the digital signatures and also to satisfy such other standards a may be prescribed by the Central Government. He shall maintain a

computerized database of all public keys in such a manner that they are available to the general public.

Section 21 provides that a license to be issued to a Certifying Authority to issue Digital Signature Certificates by the Controller shall be in such form and shall be accompanied with such fees and other documents as may be prescribed by the Central Government.

Further Controller after considering the application may either grant the license or reject the application after giving reasonable opportunity of being heard.

Section 22 provides that the application of license shall be accompanied by a certification practice statement and statement including the procedure with respect to the identification of the applicant.

Section 23 provides that application for renewal of license shall be in such form and accompanied by renewal fees as prescribed by the government.

Section 24 deals with procedure for grant or rejection of license by the controller on certain grounds.

Section 25 provides that the Controller may revoke a license on grounds such as incorrect or false material particulars being mentioned in the application and also on the ground of contravention of any provision of the Act, rule, regulation or order made there under.

Controller's power to delegate

Under **section 27** the Controller may in writing authorize the Deputy Controller, Assistant Controller or any officer to exercise any of his power under the Act. The Controller shall have the power to investigate contravention of the provisions of the Act or the rules or regulations made there under either by himself or through any officer authorized in this behalf.

Section 30 describes the duties of Certifying Authorities.

DIGITAL SIGNATURE CERTIFICATION [CHAPTWR VII]

It contains sections 35 to 40.Section 35 lays down the procedure for issuance of a Digital Signature Certificate. It provides that an application for such certificate shall be made in the prescribed form and shall be accompanied by a fee not exceeding Rs. 25,000. The fee shall be prescribed by the Central government and different fees may be prescribed for different classes of applications.

This section also provides that no Digital Signature certificate shall be granted unless the certifying authority is satisfied that

- The applicant holds the private key corresponding to the public key to be listed in Digital Signature Certificate.
- The applicant holds a private key, which is capable of creating a digital signature.
- The public key to be listed in the certificate can be used to verify a digital signature affixed by the private key held by the applicant.

No application shall be rejected unless the applicant has been given a reasonable opportunity of showing cause against the proposed rejection.

Suspension of Digital signature certificate

The certifying authority may suspend such certificate if it is of the opinion that such a step needs to taken in public interest. Section 38 provides for the revocation of Digital Signature Certificates under certain circumstances. Such revocation shall not be done unless the subscriber has been given an opportunity of being heard in the matter. Upon revocation or suspension, the Certifying Authority shall publish the notice of suspension or revocation of a Digital Signature Certificate.

DUTIES OF SUBSCRIBERS [CHAPTER VIII]

This chapter contains section 40 to 42. It specifies duties of subscribers.

- On acceptance of the Digital Signature Certificate the subscriber shall generate a key pair using a secure system.

  A subscriber shall be deemed to have accepted a Digital Signature certificate if he publishes or authorizes the publication of such signature to one or more persons or otherwise demonstrates his approval of the Digital Signature Certificate. By so accepting the certificate, the subscriber certifies to the public following

- that he holds the private key corresponding to the public key listed in the digital signature Certificate.
- that all the information contained in the certificate as well as material relevant to them are true.

- The subscriber shall exercise all reasonable care to retain control of his private key corresponding to the public key. If such private key has been compromised, the subscriber must immediately communicate the fact to the Certifying authority.

PENALTIES AND ADJUDICATION [CHAPTER IX]

This chapter contains sections 43 to 47. It provides for awarding compensation or damages for certain types of computer frauds. It also provides for the appointment of Adjudicating Officer for holding an enquiry in relation to certain crimes and for awarding compensation.

Types of Penalties

Sections 43 to 45 deal with different nature of penalties.

Penalty of damage to computer, computer system or network

Section 43 deals with penalty for damage to computer, computer system etc. by any of the following methods:

- Downloading or extracting any data, computer database or information from such computer system or those stored in any removable storage medium.
- Damaging any computer system, or network or any computer data ,database or programme
- Denying access to any person to access any computer, computer system or network
- Introducing any computer virus into any computer, computer system or network
- Providing assistance to any person to access any computer, computer system or network in contravention of any provisions of this Act or its rules.

Section 46 confers the power to adjudicate contravention under the act to an officer not below than the rank of Director to the Government of India or an equivalent officer of State Government. Such appointment shall be made by the Central Government. In order to be eligible for appointment as an adjudicating office, a person must possess adequate experience in the field of Information Technology and such legal or judicial experience as may be prescribed by Central Government. He shall be responsible for holding an enquiry in the prescribed manner after giving reasonable opportunity of being heard and thereafter, imposing penalty where required.

Section 47 provides that while deciding upon the quantum of compensation, the adjudicating officer shall have due to regard to the amount of gain of unfair advantage and the amount of loss caused to any person as well as the respective nature of the default.

CYBER REGULATIONS [CHAPTER X]

The “Cyber regulations Appellate Tribunal” has appellate powers in respect of orders passed by any adjudicating officer.

Section 48 provides for establishment of one or more appellate Tribunal to be known as Cyber Regulations Appellate Tribunals. It consists of one person, Presiding Officer, only who shall be appointed by notification by Central Government.

Section 52 provides for the salary and allowances and other terms and conditions of service of the Presiding officer.

Section 53 provides that in the situation of any vacancy occurring in the office of the Presiding Officer of Cyber Regulations Tribunal. The Central Government will appoint another person in accordance with the provisions of the act.

Section 54 deals with the Resignation and removal of the Presiding officer. The Presiding Officer shall, unless he is permitted by the Central Government to relinquish his officer sooner, continue to hold office until the expiry of three months from the date of receipt of such notice.

Section 58 provides for the procedures and powers of the Cyber Appellate Tribunal. Some of the powers are:

- Summoning and enforcing the attendance of any person and examining him on oath
- Receiving evidence on affidavits
- Reviewing its decisions
- Issuing commission for examination of witness etc.

Section 60 provides for the period of limitation for admission of appeals from the aggrieved persons to the Cyber appellate tribunal.

Section 61 provides that no court shall have jurisdiction to entertain any suit or proceeding in respect of any matter which an Adjudicating Officer has jurisdiction to determine.

Section 62 provides for an appeal to the High Court by an aggrieved person from the decision of Cyber Appellate Tribunal. Section 63 provides any contravention under the Act may be compounded by the Controller or Adjudication Officer, either before or after the institution of the adjudication proceedings subject to such conditions as he may impose.

OFFENCES [CHAPTER XI]

This section provides for the punishment imprisonment up to three years or with a fine, which may extend to Rs. 2 lakhs or with both whoever knowingly or intentionally tamper with computer code source documents.

NETWORK SERVICE PROVIDERS NOTTO BE LIABLE IN CERTAIN CASES [CHAPTER XII]

This chapter contains **section 79**, which provides that the Network service providers shall be liable for any third party information or data made available by him if he proves that the offence was committed without his knowledge or consent.

MISCELLANEOUS [CHAPTER XIII]

Section 87 of the Act confers on the Central Government the power to make rules by notifying in the Official Gazette and the Electronic Gazette, in respect of certain, matters, some of which following are:

- The manner in which any matter may be authenticated by a digital signature
- The manner and format in which electronic records shall be filled or issued
- The type of digital signature manner and format in which it may be affixed
- The security procedure for the purpose of creating same electronic record and secure digital signature
- Period of validity of license
- The qualification and experience and terms and conditions of service of Controller, Deputy Controllers and Assistant Controllers

- The qualification and experience and of an Adjudicating Officer and other officers.

Every notification made by the Central Government shall be paid, as soon as possible after it is made, before each House of Parliament, while it is in session, for a total period of thirty days. This period may be comprised in one session or in two or more successive sessions.

Power of State Government to make rules

The State Government may by notification in the Official Gazette, make rules to carry out the provisions of the Act, such rules may provide for all or any of the following matters:

- The electronic form in which filing, issue, grant receipt or payment shall be effected in respect of use of electronic records and digital signatures in Government and its agencies
- The manner and format in which such electronic records shall be filed or issued and the fee or charges in connection of the same
- Any other matter required to be provided by rules by the State Government.

Every such rule shall be laid before each House of the State Legislature. Section 80 provides that notwithstanding anything contained in the Code of Criminal Procedure, 1973, any police officer, not below the rank of Deputy Superintendent of Police or any other officer of the Central or State government, if so authorized by the Central Government, may enter any public place and search and arrest without warrant any person found therein who is reasonably suspected of having committed or of committing or is about to commit any offence under this Act.

For this purpose, ’public place’ would include a public conveyance, any hotel, any shop or any other place accessible to the public.

Liability of Companies [section 85]

Where a company commits any offence under this Act or any rule there under, every person who, at the time of the contravention, was in change of and was responsible for the conduct of the business of the company shall be guilty of the contravention.

The Information Technology Act will go a long way in facilitating and regulating electronic commerce. It has provided a legal framework for smooth conduct of e-commerce. It has been admitted in evidence in a court of law.

However, the Act has not addressed the following grey areas:

- Protection for domain names
- Infringement of copyright laws
- Jurisdiction aspect of electronic contracts
- Taxation of goods and services traded through e-commerce
- Stamp duty aspect of electronic contracts

5.10 SUMMARY

Cryptography is a technique of encoding and decoding messages, so that cannot be understood by anybody except the sender and intended recipient. Cryptography uses the same basic principle. The sender and recipient of the message decide on the encoding and decoding scheme and use it for communication. In technical terms, the process of encoding messages is known as encryption. As we know the original message text called as plain text. When it is encrypted, it is known as cipher text. The recipient understands the meaning and decodes the message to extract the correct meaning out of it. This process is known as decryption. Encryption is the mutation of information in any form (text, video, graphics) into a representation unreadable by anyone without a decryption key. Encryption is the mutation of information in any form (text, video, graphics) into a representation unreadable by anyone without a decryption key. The generation of cipher text from plaintext itself can be done in two basic ways:

- Stream ciphers

- Block Ciphers

Cryptography is also of two types:

- Secret key cryptography (Symmetric key cryptography)
- Public key cryptography (Asymmetric key cryptography)

Data Encryption Standard is a secret-key, symmetric cryptosystem. DES operates on 64-bit blocks with a 56-bit secret key. Its operation is relatively fast and works well for large bulk documents or encryption. Instead of defining just one encryption algorithm, DES defines a whole family of them.

Public key Cryptography: This cryptography system involves the use of public keys. Public key technique involves a pair of keys: a private key and a public key associated with each user. Information encrypted by private key can be decrypted only using the corresponding public key. The private key is used to encrypt the transmitted information by the user, is kept secret. The public key is used to decrypt the information at the receiver and is not kept secret. Since only the bona fide author of an encrypted message has knowledge of the private key, a successful decryption using the corresponding public key verifies the identity of the author and ensures message integrity.

Digital Signatures : In the case of business transaction authentication refers to the use of digital signature, which play a function for digital document similar to that played by handwritten signature for printed documents. The signature is an unforgeable piece of data asserting that a named person wrote or otherwise agreed to the document to which the signature is attached.

Cyber Law: Cyber law governs the legal issues of cyberspace. The term cyberspace is not restricted to the Internet. Cyber law encompasses laws relating to:

- Electronic and digital signatures
- Computer crime

- Intellectual property
- Data protection and privacy
- Telecommunication Laws

In May 2000, both the houses of the Indian Parliament passed the Information Technology Bill. The Bill received the assent of the President in August 2000 and came to be known as the Information Technology Act, 2000.Cyber laws are contained in the IT Act, 2000.

This Act aims to provide the legal infrastructure for e-commerce in India. and the cyber laws have a major impact for e-businesses and the new economy in India. The information Technology Act, 2000 also aims to provide for the legal framework so that legal sanctity is accorded to all electronic records and other activities carried out by electronic means.

5.11 KEYWORDS

CRYPTOGRAPHY: Cryptography is a technique of encoding and decoding messages, so that cannot be understood by anybody except the sender and intended recipient.

CIPHERTEXT: When original message is encrypted, it is known as cipher text.

DES: Data Encryption Standards

RSA: RSA is a public key cryptosystem for both encryption and authentication developed in 1977 by Ron Rivest, Adi Shamir and Leonard Adleman.

CYBER LAW: Cyber law governs the legal issues of cyberspace. The term cyberspace is not restricted to the Internet.

5.12 SELF ASSESSMENT QUESTIONS

1. Define the concept of cryptography. What is its importance in today's scenario?
2. Discuss different types of Digital Encryption Standard.
3. Illustrate the concept and working of Digital Signatures.
4. Define Cyber Law. What is their importance in It industry.

5. Define major provisions of IT Act 2000.

5.13 SUGGESTED READINGS

Tanenbaum, Computer Networks, Prentice Hall of India

Ravi Kalakota, Andrew B.Whinston, Frontiers of Electronic Commerce, Pearson's Publication

LESSON: 6

INTRODUCTION TO INTERNET INFORMATION SERVER (IIS)

Subject: Internet Programme for E-Commerce Paper Code: ITM-322

Author: Dr. Anil Khurana Vetter: Prof. Dharminder Kumar

6.0 OBJECTIVES

After going through this lesson, you will be able to:

- Understand the concept of IIS
- Learn about matrices for estimating an IIS system configuration
- Know the installation, testing and administration of IIS
- Understand various security issues involved with IIS

6.1 INTRODUCTION

Internet web server configuration and management used to be considered a difficult task. With all the aspects of installation, security, performance monitoring, and service partitioning, prospective internet and intranet site managers were usually so intimidated that they often gave up any hope of setting up a site on their own. Microsoft Internet Information Server (IIS) has changed all of that.

Internet Information Server is quickly becoming a de facto standard in the burgeoning Internet server market. It provides an easy way to create an Internet or intranet site. It installs and runs all services on an existing Windows NT Server in just minutes. It is downloadable for free from Microsoft's World Wide Web page.

Special Edition Using Microsoft Internet Information Server fulfills two functions. First, it steps the novice web site manager through the entire installation and configuration process while providing a complete orientation along the way. Second, it is a one-stop resource for more advanced site managers and provides useful information on advanced topics such as security, performance, HTTP/ODBC, CGI, ISAPI, NSAPI, and Java.

IIS (Internet Information Server) is a group of Internet servers (including a Web or Hypertext Transfer Protocol server and a File Transfer Protocol server) with additional capabilities for Microsoft's Windows NT and Windows 2000 Server operating systems. IIS is Microsoft's entry to compete in the internet server market that is also addressed by Apache, Sun Microsystems, O'Reilly, and others. With IIS, Microsoft includes a set of programs for building and administering Web sites, a search engine, and support for writing Web-based applications that access databases. Microsoft points out that IIS is tightly integrated with the Windows NT and 2000 Servers in a number of ways, resulting in faster Web page serving.

A typical company that buys IIS can create pages for Web sites using Microsoft's Front Page product (with its WYSIWYG user interface). Web developers can use Microsoft's Active Server Page (ASP) technology, which means that applications - including ActiveX controls - can be imbedded in Web pages that modify the content sent back to users. Developers can also write programs that filter requests and get the correct web pages for different users by using Microsoft's Internet Server Application Program Interface (ISAPI) interface. ASPs and ISAPI programs run more efficiently than common gateway interface (CGI) and server-side include (SSI) programs, two current technologies.

Microsoft includes special capabilities for server administrators designed to appeal to Internet service providers (ISPs). It includes a single window (or "console") from which all services and users can be administered. It's designed to be easy to add components as snap-ins that you didn't initially install. The administrative windows can be customized for access by individual customers.

Microsoft's Internet Information Server (IIS) is an internet/ intranet file and application server. It includes many services that are common or soon to be standard on the Internet. Basic Web and File Transfer Protocol (FTP) services have been central to IIS since the earliest versions. However, many new services have been added to higher versions. This module provides a basic overview of the entire product.

IIS is a file and application server that can be used on a local area network (LAN), a wide area network (WAN), or the Internet. IIS can be installed on Windows NT Server 4.0 or higher version; scaled down versions are available for Windows NT Workstation 4.0 and Windows 95/98.

6.2 COMPONENTS OF IIS

Six major components on the NT 4.0 Option Pack CD-ROM form the basis of IIS. These components are:

- Microsoft Internet Information Server
- Microsoft Management Console
- Microsoft Certificate Server
- Microsoft Index Server
- Microsoft Site Server Express
- Microsoft Transaction Server

Microsoft Internet Information Server

IIS provides the foundation for most of the components on the NT 4.0 Option Pack CD. IIS includes the Web Hypertext Transfer Protocol (HTTP) service and the FTP Service,

which together allow setting up an Internet or intranet server in the organization. Almost any web browser, including Internet Explorer (IE) and Netscape Navigator, can access the web and FTP Services provided by IIS. The FTP server in IIS is compatible with standard FTP client software, including the Microsoft command-line client and WS_FTP.

Microsoft Management Console

The Microsoft Management Console (MMC) is a new feature of IIS that lets user to administer all services from a single interface. This new interface includes a "snap-in" model so that control applets can be created and customized for the MMC. For example, instead of controlling only the Internet Services through the console, user can add in applets that let user to control Server Manager and User Manager for Domains. One can save the MMC configuration and send it to other administrators, which let users provide a common customized interface for all administrators and Microsoft computers.

Microsoft Certificate Server

One important feature that was not available in earlier version of IIS was a Certificate Server. Before this addition, user could not enable Secure Sockets Layer (SSL) protection without obtaining the services of a third-party Certificate Authority (CA). With newer version of IIS, this is no longer an issue. One can install Microsoft Certificate Server during IIS installation. The Certificate Server provides secure access to Web sites and enhanced e-mail security via digital signatures.

Microsoft Index Server

The Microsoft Index Server has been revised for IIS. The Index Server allows cataloging and indexing the content provided on user internet or intranet site. Then, user can provide an interface for clients to search user site so they can find the information they need without manually paging through entire web site. The search engine returns links to the locations on your site that match the search words or criteria entered by the clients so they can click a link and pull up the information they want.

Microsoft Site Server Express

There are actually three pieces to Site Server Express (SSE): Content Analyzer, Usage Import and Report Writer, and Posting Acceptor.

The Content Analyzer provides several services manage the content on the site. For example, it can display user site links graphically, much like the Microsoft FrontPage tool. In addition, Content Analyzer can report statistics about the size of files on user site, report broken links, and subdivide content into categories.

The Usage Import and Report Writer takes information collected from IIS log files and converts it into easy-to-read reports. This tool allows to generate hit statistics, page popularity, and other useful information on a regular basis.

The Posting Acceptor is an add-on tool that allows for HTTP postings, defined in Request for Comment (RFC) 1867. This component provides a hosting service for people who would like to post content to server over a web connection.

Microsoft Transaction Server

Microsoft Transaction Server (MTS) is a new component for IIS and is included on the NT 4.0 Option Pack. MTS is a transaction-based processing system that makes transaction-oriented processes possible. For example, financial transactions over a network are more reliable when MTS is used because the service ensures that the banking transactions are either fully completed or aborted. A failed online money transfer could be a disaster if money is removed from one account and not credited to another. MTS prevents this from happening by making the entire process a single transaction, so that if a debit is issued to one account but the other account never receives the money, the entire transaction is rolled back. A transaction is not rolled back when every component involved in the transaction agrees that the entire process has been completed successfully.

6.3 BENEFITS AND FEATURES OF IIS

IIS includes new features designed to help businesses, IT professionals, and web administrators achieve their goals of performance, reliability, scalability, and security for potentially thousands of web sites either on a single IIS server or on multiple servers.

Reliability

IIS 6.0 uses a new request-processing architecture and application isolation environment that enables individual web applications to function within a self-contained worker process. This environment prevents one application or web site from stopping another, and it reduces the amount of time that administrators spend restarting services to correct application-related problems. The new environment also includes proactive health monitoring for application pools.

Scalability

IIS 6.0 introduces a new kernel-mode driver for Hypertext Transfer Protocol (HTTP) parsing and caching that is specifically tuned to increase web server throughput and scalability of multiprocessor computers. The result is an increase in the following:

- The number of Web sites that a single IIS 6.0 server can host
- The number of concurrently active worker processes
- The performance for startup and shutdown times for the web server and for individual Web sites
- The number of simultaneous requests that a web server can service.

Also, by configuring the startup and shutdown time limits for worker processes, IIS allocates resources to active web sites instead of keeping resources on idle requests.

Security

IIS new versions provide significantly improved security. For example, to reduce the attack surface of systems, IIS 6.0 is not installed by default on Windows Server 2003, Standard Edition; Windows Server 2003, Enterprise Edition; and Windows Server 2003, Datacenter Edition. After installing these products, administrators must manually install

IIS 6.0. When IIS 6.0 is installed, it is locked down by default so that it can serve only static content. By using the Web Service Extensions node in IIS Manager, Web site administrators can enable or disable IIS functionality based on the individual needs of their organization.

IIS 6.0 includes a variety of security features and technologies to help ensure the integrity of the web and FTP site content, as well as the data that is transmitted through your sites. These security features and technologies include Advanced Digest authentication, improved access control, Secure Sockets Layer (SSL) encryption, centralized certificate storage, and detailed auditing capabilities.

Manageability

To meet the needs of a diverse set of organizations, IIS 6.0 provides a variety of manageability and administration tools. Administrators can configure an IIS 6.0 server by using IIS Manager, by running administration scripts, or by directly editing the IIS metabase. Administrators can also remotely administer IIS servers and Web sites.

Enhanced Development

Compared to Windows 2000 Server, Windows Server 2003 offers an improved developer experience with ASP.NET and IIS integration. ASP.NET runs most Active Server Pages (ASP) code while providing greater functionality for building enterprise-class web applications that can work as a part of the .NET Framework. Use ASP.NET to fully utilize the features of the common language runtime, such as type safety, inheritance, language interoperability, and versioning. IIS 6.0 also offers support for the latest web standards including XML, SOAP, and Internet Protocol version 6 (IPv6).

Application Compatibility

According to feedback from thousands of customers and independent software vendors (ISVs), IIS 6.0 is compatible with most of their existing web applications.

6.4 IIS HARDWARE/SOFTWARE REQUIREMENTS

IIS runs on the top of Windows NT 4.0 server. So the total IIS Intranet server requirements are Windows NT 4.0 requirements plus IIS requirements.

CPU and Memory

Users need a Pentium 120 or 133 MHz machine with 32 MB RAM for Windows NT Server and IIS. Add a minimum of 32 MB more memory for SQL server. For more than 25 users, it is recommend about 80 MB RAM. Windows NT performance falls sharply when it exhausts the memory; it is not a gradual decay. This performance cutoff point can be extended by adding more memory. Servers with 128 MB memory are the norm today, so plan ahead.

Hard Disk

The recommended disk space is 4 to 8 GB. Again, this number will depend on the amount of data to be published. If one is also going to add SQL Server data, that should be taken into account.

Microsoft recommends that all IIS disks be formatted by using the NTFS format and enable auditing. This is recommended for security reasons as well as redundancy reasons. If the Intranet server has mission-critical information, one or more of the following fault tolerance mechanisms should be considered. This strategy will affect the disk capacity required.

Disk Mirroring

In this case, two drives are connected to the same controller and all data on the first drive is duplicated on the second drive. Even though mirroring essentially duplicates one disk on another disk, NT does not require identical hard disks for mirroring. This is RAID Level 1. RAID (Redundant Arrays of Inexpensive Disks) is a scheme to increase performance and reliability of disk storage using normal hard disks. The RAID levels range from 0 to 5. Windows NT server supports RAID 0, 1 and 5. The different RAID levels have different performance and reliability characteristics. The RAID level for a

system depends on the requirement (for example, Mission critical systems need maximum reliability), type (for example, publishing systems with a lot of read only data can use a Level 5 RAID), and so on.

Disk Duplexing

Disk Duplexing is mirroring with two controllers, where the two drives are connected to two disk controllers. Duplexing improves performance (as parallel reads/writes result in faster I/O) and fault tolerance (as it protects against controller failures also).

Disk Striping with Parity

In this case, multiple partitions from different drives are combined to form logical drives. This is RAID level 5. The disk striping gives maximum performance, and the parity information gives the redundancy. This strategy is recommended over mirroring for applications that require redundancy and that are primarily read-oriented.

A CRT and Network Cards

Any normal VGA CRT is sufficient. Usually, when there are many servers, it is better to connect many computers to a master console and master mouse share device. This way, one CRT and mouse can be shared by many servers.

Now there are faster Ethernet cards (100 Mbps throughput, for example) that will give good performance in terms of raw throughput. To use one of these cards, the router or hub also should support the data rates and the cards.

Internet Connection

An Internet connection is needed to publish on the web. For small businesses, this will translate to getting the services of an Internet Service Provider (ISP).

TCP/IP

TCP/IP should be installed and configured as one of the network protocols and can be accessed from the network applet in the control panel. The typical Intranet server will be on a shared 10-100 MB network.

6.5 IIS ARCHITECTURE

IIS architecture has changed over time. Understanding the architectural features of a specific version of IIS can help to decide how to design an application for IIS.

Comparison of IIS Versions

The following table summarizes the important differences between versions of IIS.

Feature	IIS 4.0	IIS 5.0	IIS 5.1	IIS 6.0
Platform	NT4	Windows 2000	Windows XP Professional	Windows Server 2003
Architecture	32-bit processor	32-bit processor	32-bit and 64-bit processors	32-bit and 64-bit processors
Application process model	TCP/IP kernel Inetinfo.exe runs in-process applications (low isolation) MTX.exe runs out-of-process applications (high isolation)	TCP/IP kernel Inetinfo.exe runs in-process applications (low isolation) Multiple DLLhost.exe processes run pooled-process or out-of-process applications (medium or high isolation)	TCP/IP kernel Inetinfo.exe runs in-process applications (low isolation) Multiple DLLhost.exe processes run pooled-process or out-of-process applications (medium or high isolation)	HTTP.sys kernel When IIS is running in IIS 5.0 isolation mode: Same process model as for IIS 5.0 When IIS is running in worker process isolation mode: W3Wp.exe (multiple worker processes) For more information about the IIS 6.0 process model, see Worker Processes in IIS 6.0

Metabase configuration file type	Binary	Binary	Binary	XML-formatted
Security	Windows authentication SSL	Windows authentication SSL Kerberos	Windows authentication SSL Kerberos Web Server Certificate Wizard	Windows authentication SSL Kerberos Web Server Certificate Wizard Microsoft. NET Passport support
Remote administration	HTMLA	HTMLA	No HTMLA Terminal Services	Remote Administration Tool (HTML) Web Server Appliance Kit (SAK) Terminal Services
Administration technologies	ADSI and ABO	ADSI and ABO	ADSI and ABO	ADSI, WMI, and ABO
Cluster support	Windows support	IIS clustering	Windows support	Windows support
WWW Services	IIS on NT 4.0 Personal Web Manager on Windows 95 and Windows 98	IIS on Windows 2000 Server and Windows 2000 Professional (IIS is not installed by default on Windows 2000 Professional)	IIS on Windows XP Professional (IIS is not installed by default)	IIS on Windows Server 2003 (IIS is not installed by default)

Source: www.microsoft.com

IIS 6.0 provides a redesigned World Wide Web Publishing Service (WWW service) architecture that can help to achieve better performance, reliability, scalability, and security for web sites, whether they run on a single server running IIS or on multiple servers.

IIS 6.0 runs a server in one of two distinct request processing models, called application isolation modes. Application isolation is the separation of applications by process boundaries that prevents one application or web site from affecting another and reduces the time that you spend restarting services to correct problems related to applications.

In IIS 6.0, application isolation is configured differently for each of the two IIS application isolation modes. Both modes rely on the HTTP protocol stack to receive Hypertext Transfer Protocol (HTTP) requests from the internet and return responses. HTTP.sys resides in kernel mode, where operating system code, such as device drivers, runs. HTTP.sys listens for, and queues, HTTP requests.

The new request-processing architecture and application isolation environment enables individual Web applications, which always run in user mode, to function within a self-contained worker process. A worker process is user-mode code whose role is to process requests, such as returning a static page or invoking an Internet Server API (ISAPI) extension or filter. Worker processes use HTTP.sys to receive requests and send responses over HTTP.

IIS 6.0 Request Processing Models

Worker process isolation mode is the new IIS request processing model. In this application isolation mode, one can group web applications into application pools, through which one can apply configuration settings to the worker processes that service those applications. An application pool corresponds to one request routing queue within HTTP.sys and one or more worker processes.

Worker process isolation mode enables to completely separate an application in its own process, with no dependence on a central process such as Inetinfo.exe to load and execute the application. All requests are handled by worker processes that are isolated from the web server itself. Process boundaries separate each application pool so that when an application is routed to one application pool, applications in other application pools do

not affect that application. By using application pools, user can run all application code in an isolated environment without incurring a performance penalty. The visual representation of worker process isolation mode architecture is presented in figure 6.1

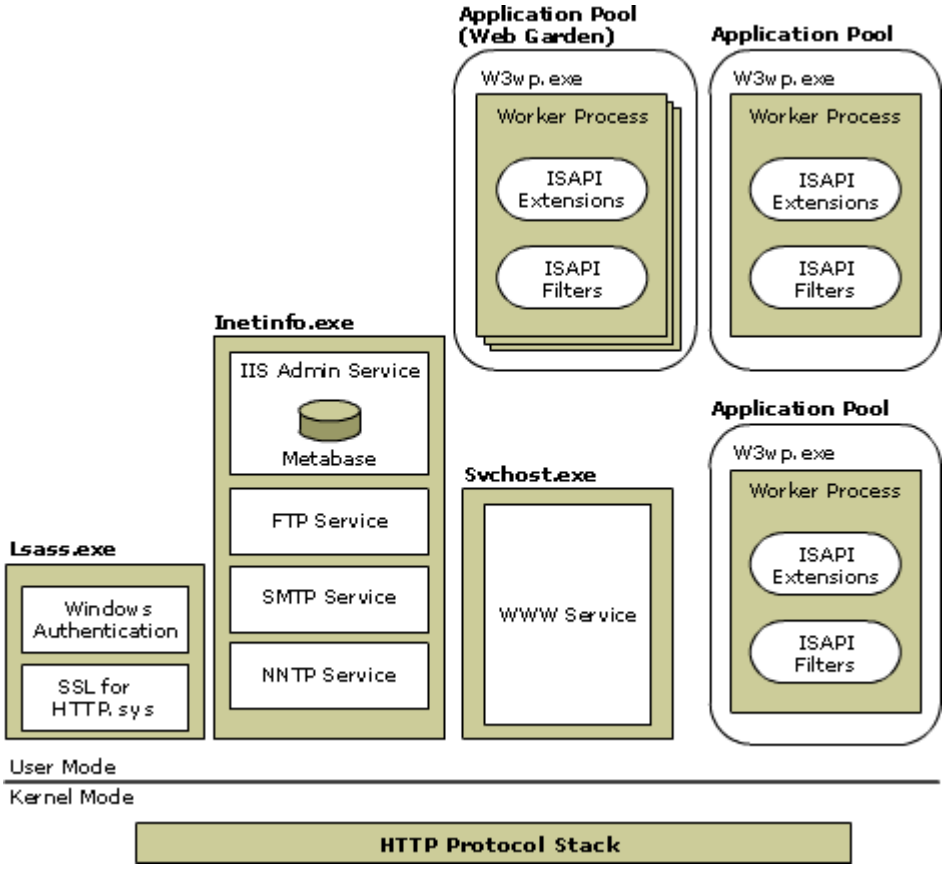


Figure 6.1 Architecture of Worker Process Isolation Mode

Worker process isolation mode delivers all the benefits of the new IIS 6.0 architecture, including multiple application pools, health monitoring and recycling, increased security and performance, improved scalability, and processor affinity. For example, the new health monitoring features can help you discover and prevent application failures, and can also help protect your Web server from imperfect applications.

IIS 5.0 isolation mode provides compatibility for applications that were designed to run in earlier versions of IIS. When IIS 6.0 is running in IIS 5.0 isolation mode, request processing is almost identical to the request processing in IIS 5.0. When a server is working in IIS 5.0 isolation mode, application pools, recycling, and health monitoring

features are unavailable. The visual representation of IIS 5.0 isolation mode architecture is shown in figure 6.2. The dashed line in Figure 6.2 indicates the dependency of the worker process on the WWW service, which manages the worker process.

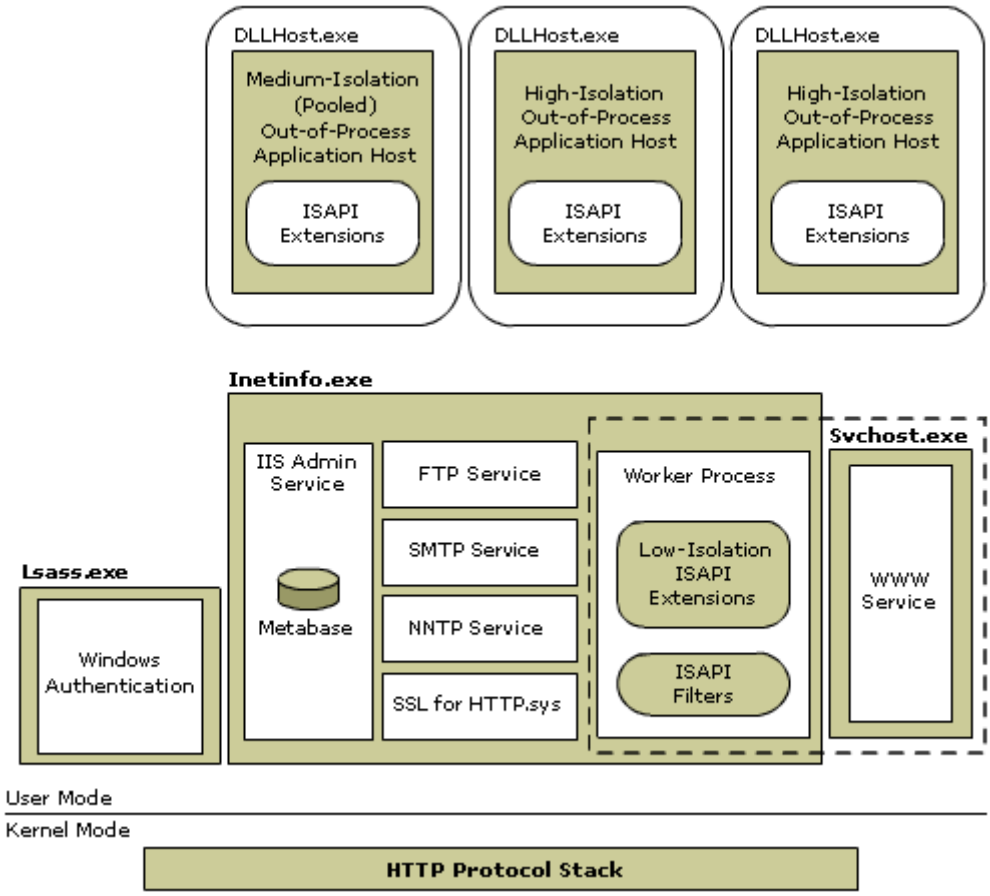


Figure 6.2 Architecture of IIS 5.0 Isolation Mode

Use IIS 5.0 isolation mode only if components or applications do not function in worker process isolation mode. The latter mode is designed to provide an environment in which most existing applications or sites function correctly.

IIS 6.0 Services

IIS 6.0 provides four Internet services: the World Wide Web Publishing Service (WWW service) for hosting internet and intranet content; the File Transfer Protocol (FTP) service for hosting sites where users can upload and download files; the Network News Transfer

Protocol (NNTP) service for hosting discussion groups; and the Simple Mail Transfer Protocol (SMTP) service for sending and receiving e-mail messages.

After installing these services, one can create sites or virtual servers, configure properties and security settings, and set up components to further customize your system.

WWW Service Administration and Monitoring, which is a new set of features that were added to the WWW service in IIS 6.0, manages worker processes, supports the new request processing model, and is responsible for health management and maintenance, including application pool health monitoring, recycling worker processes, and rapid-fail protection.

IIS Admin service is a service of the Microsoft® Windows® Server 2003, Standard Edition; Microsoft® Windows® Server 2003, Enterprise Edition; Microsoft® Windows® Server 2003, Web Edition; and Microsoft® Windows® Server 2003, Datacenter Edition operating systems. The IIS Admin service manages the IIS metabase, which stores IIS configuration data. The IIS Admin service makes metabase data available to applications and the core components of IIS.

6.5 IIS SECURITY

Microsoft designed IIS and Windows NT Server to provide administrators with a powerful framework for deploying Web servers. Above all, IIS and Windows NT Server provide administrators with a single integrated security model. In other words, IIS security is fully integrated with Windows NT security. This gives it a number of advantages, including the ability to:

- Take full advantage of the strong, secure underpinnings of the U.S. Government C2 and ITSEC FC2-rated Windows NT security.
- Eliminate possibilities for security weaknesses and holes by not adding redundant security layers. This sets IIS apart from other operating systems and web servers

with multiple security layers that increase their complexity and possibility for security holes.

- Take advantage of existing Window NT knowledge, making it easy to learn and configure.
- Provide better performance by eliminating unnecessary overheads of additional security and access control layers.
- The framework allows the administrator to determine everything from what type of end user authentication will be used on the web server, to how the web server itself will be physically locked down.

Access Control

One of the most important areas of focus for IIS is providing powerful access control functionality for web access to files and applications on the server. IIS was designed to make it easy to use a wide range of access control mechanisms to critical business data, depending on the needs of the organization. These include the following:

- Support for the Windows NT Challenge/Response (NTLM) authentication
- IP address grant/deny restrictions
- Ability to implement restrictions on virtual servers and directories
- Support for the Windows NT File System (NTFS)
- Impersonation of users when running applications
- Client and server digital certificates
- Advanced security filters

User Authentication and Authorization

IIS security is integrated with the Windows NT Directory Service, and every resource must be accessed by a user with a valid Windows NT user account. This allows administrators to use the full power of the Windows NT Directory Service account management, including the ability to audit and log all activity, set time of day restrictions, expire passwords, and force secure password policies.

Anonymous access

At setup, IIS creates an anonymous account for unauthenticated web connections. When file security is not required, the request is processed by the server in the security context of this anonymous user account. The anonymous user account can access only files and applications for which permission has been granted.

User name and password

Files and applications can be restricted to access only by specific users or groups. This requires obtaining and verifying the user name. IIS can be configured to require basic HTTP authentication. Users are prompted for a name and password, which are compared to accounts in the Windows NT Server directory. However, the name and password in basic authentication are passed as clear text over the network, and can potentially be intercepted by a network packet sniffer.

Secure Windows challenge/response

IIS also provides support for Windows NT Challenge/Response (NTLM) authentication, which uses a cryptographic technique to authenticate the password; the actual password is never sent across the network. Because every connection is mapped directly to a Windows NT user account, Internet users also get the benefit of a single logon to all servers and services in the Windows NT domain, just as they do on an intranet.

Currently, NTLM is supported by Microsoft Internet Explorer. The IIS Software Developer's Kit includes documentation and sample source code so that other software vendors may include NTLM support in their browsers and applications.

Digital certificates

Additionally, IIS supports using X.509 certificates for access control. A certificate verifies a user's identity in much the same way as a driver's license or corporate identification card does. They are issued by a trusted certificate authority, either within an organization or a public company. How rigorously IIS checks the user's identity or credentials when issuing a certificate depends upon the level of security—or trust—

required for the information or application being accessed. Users enter a password when signing their certificate, and this password is required every time the certificate is activated for use. As with a driver's license, mere possession of a certificate does not constitute proof of ownership. Because only the owner of the certificate should know it, the password is the key to verifying access.

Certificate-based client authentication requires a protocol able to handle certificates at both the client and server end, as well as the appropriate requests and replies. A server certificate is presented to a client, so that the client may authenticate the identity of the server. When running Secure Sockets Layer (SSL), a server is required to have a server certificate. Optionally, a server can ask for the client's certificate. The server certificate contains the Web site name, and the browser will verify that the Web site is the name that was entered.

Access control using custom authentication filters

IIS provides a set of open APIs that developers can use to create filters that authenticate users based on custom rules. This gives administrators the flexibility to control access using any authentication scheme or external directories.

Access Controls

Once users are authenticated, IIS checks to see if they have permission to access the requested file or application.

IP addresses

On the Internet, each server and client (or proxy for a group of clients) has a specific Internet address called the "IP address". IIS can be configured to grant or deny access to specified IP addresses. This gives the administrator the ability to exclude users by denying access from a particular IP address, or prevent entire networks from accessing the server. Conversely, administrators can choose to allow only specific IP addresses to have access to the service.

Windows NT File System permissions

The Windows NT File System (NTFS) was designed to provide security features required for high-end Web servers in both intranet and Internet scenarios. The NTFS file system supports discretionary access control and ownership privileges that are important for the integrity of critical business data. NTFS allows administrators to assign permission to individual files, not just to folders and directories. By using the NTFS file system for the content made available by IIS, administrators can help ensure only the right individuals have access to individual files on the web server.

Once the user's IP address restrictions are satisfied, the user name or password is validated, and the service's virtual directory permissions are completed, IIS will then attempt to access the specified resource (based on the URL) using the security context of the authenticated user. This allows Windows NT Server to enforce access control based on NTFS permissions on the resources, offering administrators extremely granular control over sensitive resources and data.

Windows NT identifies each user by globally unique security identification (SID), not by user name. This SID is mapped in the background to the user's account name, so file permissions and group accounts are managed using a friendly name but applied using the SID. When an account is deleted, all ACLs and group assignments for the account are also removed. SIDs and synchronization ensures that an account later created with the same user name cannot inherit permission to the old account.

Impersonation

IIS accesses all files and runs all applications in the security context of the user requesting the file, restricting what can be accessed. This is either the anonymous user account specified in the server administration, or an authenticated user account. This means that a CGI application or component in a user directory cannot access data or services restricted to other users or the server administrator. Moreover, application developers have much more flexibility in developing applications than they would if all codes were required to run in the security context of the server itself. Impersonation

allows Web-based applications to be used securely for applications or administrator-like functions that must limit both who accesses the application and what they are allowed to do.

6.7 SUMMARY

In the beginning of the Web, Web applications were nothing more than distributed static documents. With the advent of CGI (Command Gateway Interface) programming, Web pages became a little more sophisticated, offering features like hit counters, guest books, maybe some flat-file database interaction, but still lacked the ability to provide anything more than a glorified online catalog.

The old Web application model has changed dramatically over the past couple years, thanks to technologies like Internet Information Services (IIS) that allow for tight integration of the Web services into the operating system. Web applications can now provide interaction with various databases, file systems, and other services that were strictly the realm of client-server desktop applications.

The Web is becoming the standard for distributed applications, not only for the Internet, but also for internal business communications and mission-critical business-to-business operations. The Web plays a very critical role in our current economy and day-to-day operations than ever before, and the current trend is for the Web to grow beyond what is realized today. Microsoft having recognized this growing use of Web technologies incorporated IIS into their latest operating system. IIS is now part of the default installation of Windows 2000 Professional, Server, and Advanced Server and higher versions.

6.8 KEYWORDS

IIS: IIS (Internet Information Server) is a group of Internet servers (including a Web or Hypertext Transfer Protocol server and a File Transfer Protocol server) with additional capabilities for Microsoft's Windows NT and Windows 2000 Server operating systems.

Microsoft Internet Information Server: It allows setting up an Internet or intranet server in the organization.

Microsoft Management Console: This option of IIS allows the user to administer all services from a single interface.

Microsoft Certificate Server: This server provides secure access to Web sites and enhanced e-mail security via digital signatures.

Microsoft Index Server: This server lets users to catalog and index the content provided on Internet or intranet site.

Microsoft Site Server Express: There are actually three pieces to Site Server Express (SSE): Content Analyzer, Usage Import and Report Writer, and Posting Acceptor.

Content Analyzer: It provides several services that allows the user to manage the content on your site.

Usage Import and Report Writer: The Usage Import and Report Writer takes information collected from user IIS log files and converts it into easy-to-read reports.

Posting Acceptor: This component allows the user to provide a hosting service for people who would like to post content to user server over a web connection.

Microsoft Transaction Server: It is a transaction-based processing system that makes transaction-oriented processes possible.

CGI: Common Gateway Interface, a programming interface between a web server and the systems backend functions - such as processing systems and databases. It allows web servers to perform data functions and interact with users.

6.9 SELF ASSESSMENT QUESTIONS

1 Define IIS. Discuss its various components.

2 Highlight various benefits and features provided by the latest version of IIS.

3 Describe the hardware and software requirement of IIS.

4 Illustrate the concept of IIS. Also describe its architecture.

5 Write a detailed note on IIS security.

6 Write short notes on:

(a) Microsoft Internet Information Server

(b) Microsoft Management Console

(c) Microsoft Certificate Server

(d) Microsoft Index Server

(e) Microsoft Site Server Express

(f) Microsoft Transaction Server

1.5 SUGGESTED READINGS / REFERENCES

1 Hethe Henrickson, Scott Hofmann, IIS 6: the complete reference, McGraw Hill, New York

2 Martin C. Brown, Don Jones, Microsoft IIS 6 Delta Guide, Sams Publishing, USA

3 Rod Trent, IIS 5.0: A beginner's guide, Osborne, USA

LESSON: 6

INTRODUCTION TO APACHE SERVER

Subject: Internet Programme for E-Commerce **Paper Code: ITM - 322**

Author: Dr. Anil Khurana **Vetter: Prof. Dharminder Kumar**

7.0 OBJECTIVES

After going through this lesson, you will be able to:

- Understand the basics of Apache
- Know the features and limitation of Apache
- Know about various modules in Apache
- Get familiar with security considerations in Apache
- Under the method of installation & configuration of Apache in different operating systems
- To learn about different most commonly used command in Apache

7.1 INTRODUCTION

Apache is a web server. In fact Apache is the worlds most popular and dominating web server with over 61% of all Internet web servers running Apache. A far second is Microsoft IIS, with a measly 19% of market share. This is a true testament to Apache's popularity.

Apache actually stands for "A patchy server". Since the original Apache was built from "patching" the original NCSA HTTP daemon in early 1995. Apache is not owned by a single commercial entity (like IIS is owned by Microsoft, iPlanet is owned by Sun/Netscape Alliance) but rather, is developed by a loose knit team of voluntary programmers scattered across the globe, collaborating through the Internet. Today, development of Apache is coordinated by a non-profit organization called the Apache Foundation.

Apache has been written in C, using a dynamic, modular architecture (much like the kernel module architecture) in which pieces of functionality can be inserted into the web server by loading pieces of code known as modules. The pieces of code are built as shared libraries/objects on Unix systems. These pieces of code can also be statically compiled into Apache. This makes Apache highly extensible and configurable.

Apache is available for download on the Internet, free of charge and is bundled with all Linux distributions. In fact, most (or maybe all) Linux distributions install and configure Apache by default. In other words when while installing Linux, Apache is also installed.

7.2 FEATURES OF APACHE

All versions of Apache have the following features:

- **Powerful:** Apache's performance and reliability is legendary.
- **Feature-Rich:** The Apache server sports a host of features, including: XML support, server-side includes, powerful URL-rewriting, and virtual hosting, to name but a few.

- **Modular**: Apache server provides the modularity concept that adds the functionality required by the users.
- **Extensible:** As Apache is open source, users can write themselves. In fact, users can even make changes to the inner workings of Apache. All the information users need is right there in the source code and numerous online resources. Users can share their patches or modules with the community by making them open source as well.
- **Popular**: Apache holds a smidge under 60 percent of the web server market. And, yes, popularity does count; help abounds and is only a mailing list or newsgroup posting away.
- **Free:** Apache is available for download on the Internet, free of charge and is bundled with all Unix / Linux distributions.

Additional features of Apache server which are available in higher / newer versions are:

- **Unix Threading:** On Unix systems with POSIX threads support, Apache can now run in a hybrid multiprocessing, multithreaded mode. This improves scalability for many, but not all configurations.
- **New Build System:** The build system has been rewritten from scratch to be based on autoconf and libtool. This makes Apache's configuration system more similar to that of other packages.
- **Multi-protocol Support:** Apache now has some of the infrastructure in place to support serving multiple protocols. mod_echo has been written as an example.
- **Better support for non-Unix platforms:** Apache 2.0 is faster and more stable on non-Unix platforms such as BeOS, OS/2, and Windows. With the introduction of platform-specific multi-processing modules (MPMs) and the Apache Portable Runtime (APR), these platforms are now implemented in their native API, avoiding the often buggy and poorly performing POSIX-emulation layers.
- **New Apache API:** The API for modules has changed significantly for 2.0. Many of the module-ordering/-priority problems from 1.3 should be gone. 2.0 does much of this automatically, and module ordering is now done per-hook to allow

more flexibility. Also, new calls have been added that provide additional module capabilities without patching the core Apache server.

- **IPv6 Support:** On systems where IPv6 is supported by the underlying Apache Portable Runtime library, Apache gets IPv6 listening sockets by default.
- **Filtering:** Apache modules may now be written as filters which act on the stream of content as it is delivered to or from the server. This allows, for example, the output of CGI scripts to be parsed for Server Side Include directives using the INCLUDES filter in mod_include.
- **Multilanguage Error Responses:** Error response messages to the browser are now provided in several languages, using SSI documents. They may be customized by the administrator to achieve a consistent look and feel.
- **Simplified configuration:** Many confusing directives have been simplified. The often confusing Port and BindAddress directives are gone; only the Listen directive is used for IP address binding; the ServerName directive specifies the server name and port number only for redirection and vhost recognition.
- **Native Windows NT Unicode Support:** Apache 2.0 on Windows NT now uses utf-8 for all filename encodings. These directly translate to the underlying Unicode file system, providing multi-language support for all Windows NT-based installations, including Windows 2000 and Windows XP. This support does not extend to Windows 95, 98 or ME, which continue to use the machine's local codepage for filesystem access

The only limiting factor to Apache's feature set is our imagination. Due to the modular nature of Apache, new functionality can be added to the server on demand by plugging in the module to support the functionality. Among the standard modules bundled with Apache include URL rewriting, access control and authentication, setting of non standard HTTP headers, server-side includes, CGI support, server side image map support, proxy and caching support etc.

7.3 LIMITATIONS OF APACHE

Even though Apache is an extremely powerful server with a wide assortment of features, it does have certain, arguably minor, caveats:

- Apache does not provide the GUI-based convenience administration of configuration files.
- The Macintosh version of Apache (unlike its UNIX or Windows counterparts) is not freely available.
- Apache (except the Windows NT version) is not multithreaded.
- The Windows versions do not have all the bells and whistles found in the Unix versions. For example, it has not yet been optimized for performance. As the Windows version is being actively pursued, it is expected that these differences will vanish in future.

7.4 APACHE'S MODULES

The choice of modules is one of the most important steps of securing Apache. One should go by the rule: ***the less the better***. To fulfill the functionality and security assumptions, the following modules must remain enabled:

Module's name	**Description**
httpd_core	The core Apache features, required in every Apache installation.
mod_access	Provides access control based on client hostname, IP address, or other characteristics of the client request. Because this module is needed to use "order", "allow" and "deny" directives, it should remain enabled.
mod_auth	Required in order to implement user authentication using text files (HTTP Basic Authentication), which was specified in functionality assumptions.
mod_dir	Required to search and serve directory index files: "index.html", "default.htm", etc.
mod_log_config	Required to implement logging of the requests made to the server.
mod_mime	Required to set the character set, content- encoding, handler, content-language, and MIME types of documents.

All other Apache's modules must be disabled. Users can safely turn them off, mainly because normally that are not required. By disabling unneeded modules, users can avoid potential break-ins when new security vulnerabilities are found in one of them.

It is also worth to note that two of Apache's modules can be more dangerous than others: mod_autoindex and mod_info. The first module provides for automatic directory indexing, and is enabled by default. It is very easy to use this module in order to check if Apache runs on a server (e.g. http://server_name/icons/) and to get the content of the Web server's directories, when no index files are found in them. The second module, mod_info, should never be accessible from the Internet, mainly because it reveals the Apache server's configuration.

7.5 SECURING APACHE

One of the most important elements of every computer project is the specification of security assumptions. This must be fulfilled before the project is implemented. The security assumptions for Apache web server are as follows:

- The operating system must be hardened as much as possible, both against local and remote attacks;
- The server must not offer any network services except HTTP: (80/TCP);
- Remote access to the server must be controlled by a firewall, which should block all outbound connections, and allow inbound connections only to the 80/TCP port of the web server;
- The Apache web server must be the only service available on the system;
- Only absolutely necessary Apache modules should be enabled;
- Any diagnostic web pages and automatic directory indexing service must be turned off;
- The server should disclose the least amount of information about itself (security by obscurity);

- The Apache server must run under a unique UID/GID, not used by any other system process;
- Apache's processes must have limited access to the file systems (chrooting); and,
- No shell programs can be present in the Apache's chrooted environment (/bin/sh, /bin/csh etc.).

7.6 APACHE DISTRIBUTION

Apache distributions come in various forms. These are:

1. **Source distribution:** This consists of the source code and no pre-built binaries. Once one download a source distribution, one must compile it and run the scripts bundled with the distribution to install it. This gives the user maximum flexibility to custom configure and install Apache. The user even has the liberty of extending or modifying the source code and installing other functional modules as well. The install procedure is built directly into the Makefile to build the source and simply issuing the command 'make install' will install Apache with it's defaults on your system.
2. **Binary distribution:** This consists of pre-built binaries for the various supported operating environments and platforms. These distributions come with installation scripts that allow users to install Apache on their system.
3. **Packaged distributions:** These distributions come in packaged formats like RPM (for RedHat and derivative systems). These distributions are installed via the standard installation management program called rpm. This allows the software installation of Apache to be tracked by the operating System.

7.7 INSTALLATION & CONFIGURING OF APACHE IN WINDOWS NT

Installing Apache

Go to the Apache Win32 download directory. One will need to download two pieces of software in a temporary directory. The first is the new Microsoft MSI package installer (MSIEXEC) instmsi.exe, standard in Windows Me and 2000, recently made available by

Microsoft for Windows NT and 95/98. The second is the file apache 1.3.x.x-win32-xxxx.msi file with the apache server packaged to be installed with MSIEXEC. Run the instmsi.exe file to install MSIEXEC. Once MSIEXEC is installed then double-click on the 1.3.x.x-win32-xxxx.msi file. Install apache in C:\Apache instead of the default in Program Files. During the installation you will be asked for a server name (one can enter localhost, or an IP number, or yet a FQDN), and a domain name -- use the domain of the service provider. Make sure to have Perl downloaded and installed before installation of Apache, if one is intended to use Perl to write CGI scripts.

The package will automatically create all the directories for user besides installing the software. The document root will be in c:/Apache/Apache/htdocs. The server root will be located in c:/Apache/Apache. The path to the apache program is C:/Apache/Apache/Apache.exe, but in NT it runs as a service, which is also automatically installed.

In addition a variety of directories are created: cgi-bin, htdocs, icons, include, lib, libexec, etc. In the htdocs directory you will store the web pages for your server. In the cgi-bin directory user will store his CGI programs.

In the server root three directories are created: conf, logs and modules. In the conf directory you will find the three basic Apache configuration files: httpd.conf, access.conf and srm.conf. In the logs directory user will find the access and error logs. The modules directory is where the Apache modules are resident.

Configuring Apache in NT

User needs to configure (change) the C:/Apache/Apache/conf/httpd.conf file. The other two configuration files are considered obsolete and should not be changed.

httpd.conf: this is the overall configuration file. Open the file httpd.conf in Notepad or any other text editor.

Find first the line #BindAddress * and delete the # (uncomment it) to make it active. Find the line ServerAdmin and enter own e-mail address, and look for the line ServerName and enter the FQDN, or IP number of your machine, or yet localhost. The server comes configured to run in standalone mode, to listen in port 80, and user doesn't need to change these options. If user installed Apache in C:/Apache the document root directory is written by default in the line

DocumentRoot "c:/Apache/Apache/htdocs"

Look for the section that starts with <Directory "c:/Apache/Apache/htdocs"> and look for the Options line and change it to allow Server Side Includes, but disabling scripts to be run from a web page, as follows: Options Indexes Includes FollowSymLinks IncludesNOEXEC. This will allow the dynamic dating of user changes, the dynamic display of time and date in your pages, but will prevent scripts to be run outside of user cgi-bin directory.

For example, the HTML code <!--#config timefmt="%A %B %d, %Y -- %I:%M %p " --><!--#echo var="DATE_LOCAL"-->

will be displayed as Day of the week Moth Date, Year -- Time

User may want to change the order of his index files, making index.shtml (the SSI version) to be the first to be displayed by the browsers. In order to do so, user will look for the line DirectoryIndex and will change it to DirectoryIndex index.shtml index.html index.htm

User will look for the line #ScriptInterpreterSource registry and will to remove the # (uncomment it). This will allow scripts written in different languages to use their extention associations in Windows. For example, hello.pl would be associated with Perl. In case you wish to keep it commented -- do not use associations, then the first line of a script would be like in Linux/Unix -- a shebang line as follows:

#!c:/Perl/bin/Perl.exe

User will check that his cgi-bin directory is properly identified as follows:

ScriptAlias /cgi-bin/ ”C:/Apache/Apache/cgi-bin/” .

If user moved his document, root path change this line accordingly.

User will look for the line starting with # AddHandler. If user wants his CGI script files to be identified with .cgi be sure that the line AddHandler cgi-script .cgi is not commented out (marked with in # in front). To complete the installation of server side includes be sure that the following two lines are present and uncommented (without the # in front):

AddType text/html .shtml
AddHandler server-parsed .shtml

Finally, if user wants to make his Web server support image maps be sure that the following line exists:

AddHandler imap-file map

Starting Apache in Windows NT

Apache runs as a service in Windows NT, therefore user can start, stop and restart the Apache Web server using the Services applet in the Windows NT Control Panel.

7.8 INSTALLATION AND CONFIGURING OF THE APACHE IN UNIX

By UNIX means here UNIX and its variants, including Linux are included in defining the installation and configuration of Apache Server.

Installing Apache

Mount the CD-ROM and install the package for apache. The package will automatically create all the directories besides installing the software. The document root will be placed in /home/httpd in 6.0 and in /var/www in 7.0. The server root will be located in /etc/httpd. The path to the apache program will be /usr/sbin/httpd.

In the document root three directories are created: cgi-bin, html and icons. In the html directory user store the web pages for his server. In the cgi-bin directory user will store you're his programs.

In the server root three directories are created: conf, logs and modules. In the /etc/httpd/conf directory you will find the three basic Apache configuration files: httpd.conf, access.conf and srm.conf. In the /etc/httpd/logs directory user will find the access and error logs. The /etc/httpd/modules directory is where the Apache modules are resident.

Configuring Apache in Linux

Apache comes with the same three configuration files, but only httpd.conf needs to be configured, changed, by user. The other two are considered obsolete and should not be changed.

httpd.conf: This is the overall configuration file. User will open the file httpd.conf in Kedit or any other text editor. User will find first the line #BindAddress * and delete the # (uncomment it) to make it active. User will again find the line ServerAdmin and enter his e-mail address, and look for the line ServerName and enter the FQDN, or IP number of your machine, or yet localhost. The server comes configured to run in standalone mode, to listen in port 80, with user and group apache (with low level of permissions for security reasons), and there is no need to change these options. The document root directory is by default in the line DocumentRoot "/var/www/html"

User look for the section that starts with *<Directory "/var/www/html">* and look for the *Options* line and change it to allow Server Side Includes, but disabling scripts to be run from a Web page, as follows: *Options Indexes Includes FollowSymLinks IncludesNOEXEC*. This will allow the dynamic dating of user changes, the dynamic display of time and date in your pages, but will prevent scripts to be run outside of your cgi-bin directory.

For example, the HTML code *<!--#config timefmt="%A %B %d, %Y -- %I:%M %p " --><!--#echo var="DATE_LOCAL"-->*

will be displayed as Day of week Month Date, Year – Time

User may want to change the order of his index files, making index.shtml (the SSI version) to be the first to be displayed by the browsers. In order to do so, look for the line DirectoryIndex and change it to DirectoryIndex index.shtml index.html index.htm

Check that cgi-bin directory is properly identified as follows: ScriptAlias /cgi-bin/ /var/www/cgi-bin/ . If one moved his document root path change this line accordingly. As an additional security measure one may also change the section starting with <Directory /var/www/cgi-bin> so that the Options line be commented out as follows: # Options ExecCGI. This will not prevent you to run CGI programs in cgi-bin, but rather it will prevent the command exec to be used even in the cgi-bin directory.

User look for the line starting with *# AddHandler*. If one wants his CGI script files to be identified with .cgi be sure that the line *AddHandler cgi-script .cgi* is not commented out (market with in * in front). To complete the installation of server side includes be sure that the following two lines are present:

AddType text/html .shtml
AddHandler server-parsed .shtml

Finally, to make web server support image maps be sure that the following line exists:

AddHandler imap-file map

Starting Apache in Linux

One can start, stop and restart the Apache web server by using scripts created for this purpose in Red Hat Linux. Type in a shell prompt as root: /etc/rc.d/init.d/httpd start to start the server. To stop or restart use the same script but replacing start with stop or restart. One must start the server as root, in order to run the server in port 80, as defined in httpd.conf.

7.9 APACHE COMMAND LINE OPTIONS

These options are applicable to all Apache ports, including UNIX and Windows 9x/NT.

OPTION	FUNCTION
apache -d serverroot	Set the initial value for the ServerRoot variable to server root. This can be overridden by the ServerRoot command in the configuration file. The default is /usr/local/apache on UNIX and /apache on Windows.
apache -D name	Define a name for use in in IfDefine directives. This option can be used to optionally enable certain functionality in the configuration file, or to use a common configuration for several independent hosts, where host specific information is enclosed in <IfDefine> sections.
apache -f config	Execute the commands in the file config on startup. If config does not begin with a /, then it is taken to be a path relative to the ServerRoot. The default is httpd.conf.
apache -C "directive"	Process the given apache "directive" (just as if it had been part of a configuration file) before actually reading the regular configuration files.
apache -c "directive"	Process the given apache "directive" after reading all the regular configuration files.
apache -X	Run a single-process mode for internal debugging purposes only. (The daemon does not detach from the terminal or fork any children.) It should not be used to provide ordinary Web service.
apache -v	Print the version of httpd and its build date, and then exit.

apache -V	Print the base version of httpd, its build date, and a list of compile time settings which influence the behaviour and performance of Apache, then exit.
apache -L	Give a list of directives together with expected arguments and places where the directive is valid, then exit.
apache -l	Give a list of all modules compiled into the server, then exit.
apache -h	Print a list of the httpd options, then exit.
apache -S	Show the settings as parsed from the config file (currently only shows a breakdown of the vhost settings) but do not start the server.
apache -t	Test the configuration file syntax (i.e., read all configuration files and interpret them) but do not start the server. If the configuration contains errors, display an error message and exit with a non-zero exit status, otherwise display "Syntax OK" and terminate with a zero exit status.

Table 7.1. Apache General Command Line Options.

Windows-Specific

OPTION	FUNCTION
apache	Start running Apache.
apache -f "\path_to\conf\file_name.conf"	Specify a configuration file. If a configuration file name with -f is not specified, Apache will use the file name compiled into the server, usually httpd.conf.
apache -k shutdown (or CTRL-C)	Signal Apache to stop. User will need to open another console window.
apache -k restart	Signal Apache to restart. This makes it re-read the configuration files. Any transactions in progress are allowed to complete without interruption.

Table 7.2. Apache as a Console Application: Command Line Options.

OPTION	FUNCTION
apache -i -n "service name"	Install Apache as a service. The default "service

	name", if one is not specified, is "Apache".
apache -u -n "service name"	Remove an Apache service.
apache -i -n "service name" -f "\path_to\conf\file_name.conf"	Install a service with a specific configuration file. If a configuration file name with -n is not specified, Apache will use the file name compiled into the server, usually httpd.conf.
apache -n "service name" -k start	Start running Apache.
apache -n "service name" -k shutdown	Signal Apache to stop.
apache -n "service name" -k restart	Signal Apache to restart. This makes it re-read the configuration files. Any transactions in progress are allowed to complete without interruption.

Table 7.3. Apache as a Service: Command Line Options.

7.10 SUMMARY

The decision of choosing the “right” Web server is crucial for anybody who intends to have a presence and provide a service on the Web. Apache, with its numerous features, makes an excellent choice. Apache is acknowledged as the leading web server on the Internet. Its modular architecture makes it inherently flexible to change. It enjoys wide scale industry support and all popular web development environments work with Apache. It comes with all the advantages of open source allowing system administrators to highly optimize the web platform.

7.11 KEYWORDS

Apache: It is famous web server used for uploading the files on internet.

Unix / Linux: Famous multi-user operating system

Windows NT: Multi-user operating system from Microsoft which is user friendly

7.12 SELF ASSESSMENT QUESTIONS

1. What is Apache? Why is it named so? Discuss its features and limitations.
2. What are the various modules available in Apache?
3. Discuss the security considerations while installing Apache web server.

4. Illustrate the method of installation & configuring of Apache in Unix / Linux and Windows NT.
5. What are various command lines available in Apache? Discuss these command lines with their function.

7.13 SUGGESTED READINGS

1. Rich Bowen, Allan Liska, Daniel Lopez Ridruejo, Daniel Lopez, Apache Administrator's Handbook, SAMS Publishing, USA
2. Berislav Kucan, Apache Server 2.0: The Complete Reference, McGraw-Hill Professional, USA
3. Ben Laurie & Peter Laurie, Apache: The Definitive Guide, Second Edition, O'Reilly & Associates, 1999.