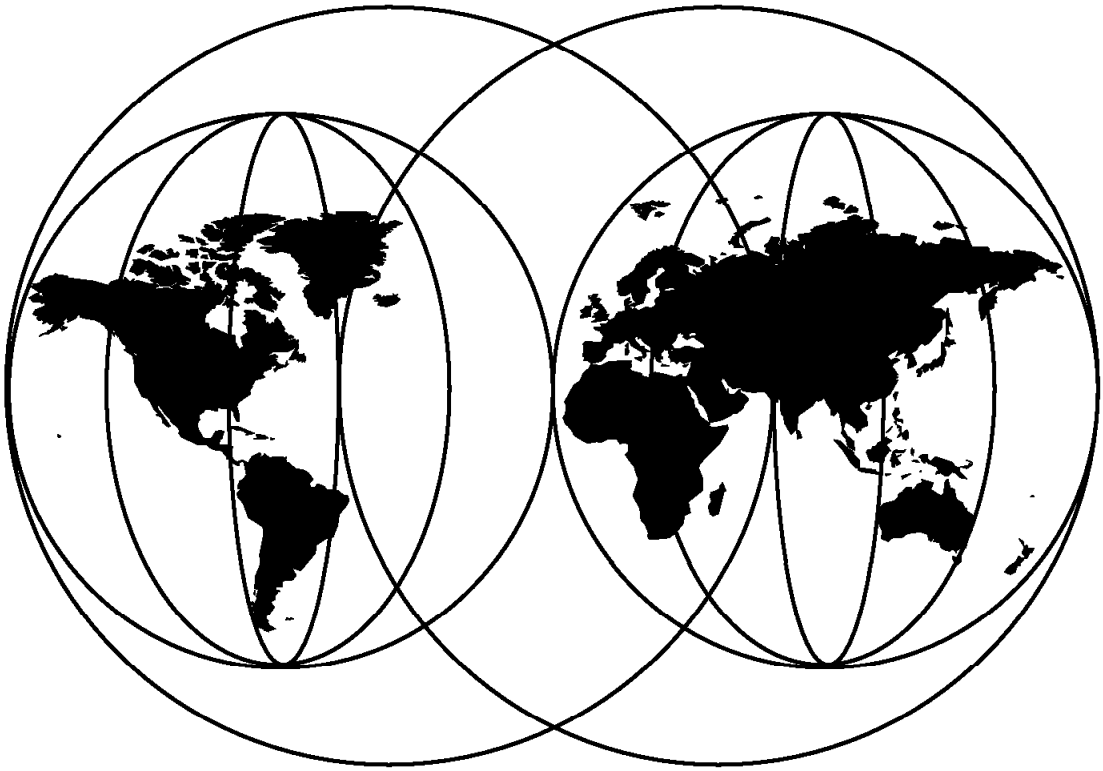
Web Server Solutions for VM/ESA

Erich Amrehn, Nicholas J. Gimbrone, Bruce J. Hayden, Stephen Record



International Technical Support Organization

http://www.redbooks.ibm.com

SG24-4874-01

International Technical Support Organization

SG24-4874-01

Web Server Solutions for VM/ESA

December 1998

Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix D, “Special Notices” on page 227.

Second Edition (December 1998)

This edition reflects Virtual Machine/Enterprise Systems Architecture Version 2 Release 2.0, Program Number 5654-030, and subsequent releases, EnterpriseWeb/VM, VM:Webgateway Release 2.2, VM:Webserver OfficeVision Interface and Webshare.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. HYJ Mail Station P099
522 South Road
Poughkeepsie, New York 12601-5400

Contents

Figures

Tables

Preface

The World Wide Web is the hottest client/server technology to emerge in the last ten years. Behind the bold new graphics offered by the Web browser, there exists a basic Hypertext Transfer Protocol (HTTP) server. This redbook describes every available HTTP server for the VM/ESA platform.

The information presented in this publication is intended to assist IT architects, VM/ESA system programmers, and marketing specialists in deciding which server solution is best for their respective enterprises.

The VM/ESA platform has been the benchmark of low-cost office and mail serving in the large scale computing environment. Only with a Web server can VM reach a progressive new workstation client, the Web browser.

In this publication you will find a collection of common Internet terms and their definitions, TCP/IP information, a Web server implementation strategy, and a discussion of the following products aided by a comparison chart:

- Webshare, shareware Web server
- EnterpriseWeb/VM, from Beyond Software Inc.
- VM:Webgateway, from Sterling Software, Inc.

A basic understanding of VM/ESA is assumed.

The Team That Wrote This Redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization Poughkeepsie Center.

The authors of the second edition of this redbook are:

Nicholas J. Gimbrone	Sterling Software Inc.
Bruce J. Hayden	IBM Endicott

The project that produced the second edition of this publication was coordinated by:

Erich Amrehn	IBM ITSO Poughkeepsie
Stephen Record	IBM Böblingen/Endicott

The authors of the first edition of this redbook were:

Jeffrey L. Forte	ISSC Technology Solutions South - Lexington
Steve Lee	IBM United Kingdom
Heinz Werner Mattheis	IBM Germany
Gerhard Widmayer	IBM Germany
Holger Woller	IBM Germany

The project that produced the first edition of this publication was coordinated by:

Scott Vetter	IBM ITSO Poughkeepsie

We would also like to acknowledge the professionals who took time to review this document, and provided invaluable advice and guidance during its development.

For the second edition:

Mike Ludé	Beyond Software Inc.
Romney White	IBM Endicott

For the first edition:

Christine Cowan	Beyond Software Inc.
Jerry Rascano	Beyond Software Inc.
James Weissman	Beyond Software Inc.
Pam Bradford	Sterling Software, Inc.
Steve Harriman	Sterling Software, Inc.
Steve Revell	Sterling Software, Inc.
Melinda Varian	Princeton University
Brian Wade	IBM Endicott
Fred Borchers	ITSO PS and S
Roy Costa	ITSO PS and S
Mike Schwartz	ITSO PS and S

Comments Welcome

Your comments are important to us!

We want our redbooks to be as helpful as possible. Please send us your comments about this or other redbooks in one of the following ways:

- Fax the evaluation form found in “ITSO Redbook Evaluation” on page 241 to the fax number shown on the form.
- Use the online evaluation form found at `http://www.redbooks.ibm.com/`
- Send your comments in an Internet note to `redbook@us.ibm.com`

Chapter 1. Internet Concepts

In this chapter we introduce some basic Internet concepts. Understanding these concepts will help you to better appreciate the features and functions of the VM/ESA Web servers.

1.1 The World Wide Web

The World Wide Web, or WWW, is the latest information service to arrive on the Internet. It is based on a technology called *hypertext*. Hypertext is a method of presenting information where selected words or images have links to other information. Therefore, the WWW organizes all the information on the Internet into hypertext documents to form an enormous database.

The WWW is designed around the client/server concept.

Typically, a user will use a client program to interface with a server somewhere on a network. This client program is generally called a browser, as it is used to browse through the information available on the Web. The server can either be internal to an organization (see section 1.2, “An Intranet”), or it can be external to the organization’s network and attached to the Internet.

The World Wide Web is not owned or run by any one organization, although the World Wide Web Consortium (W³C) does provide some monitoring and guidance. More information about W³C can be found at `http://www.w3.org/`

1.2 An Intranet

The term *Internet* is widely used and most people are aware that it is the name given to the large, wide-area network connecting many smaller disparate networks. Each of these networks is generally owned by an organization (either industry, education, or government) that allows external Internet traffic to flow over part of its network.

The term *intranet* defines a private network over which the Internet or similar traffic flows. In a large organization with more than one site, this traffic may flow between sites carried on a network owned by an external network provider. This is still considered an intranet, as all the data flowing is still only accessible from within the organization.

Often, the intranet can be considered more important to the running of the particular business than the Internet. The methods used and developed for the Internet are now used by organizations to distribute data around their departments over intranets. For example, Web servers such as VM Webshare are being widely used to serve data internally over an intranet and externally over the Internet.

1.3 The URL

URL stands for Universal Resource Locator and is the unique address of each page on the Web. The following are some examples of URLs:

- http://www.vm.ibm.com/
- http://www.austin.ibm.com/pspinfo/server3.html
- ftp://ftp.hursley.ibm.com/
- gopher://gopher.ibmlink.ibm.com/

A basic consists of four elements, shown in Figure 1.

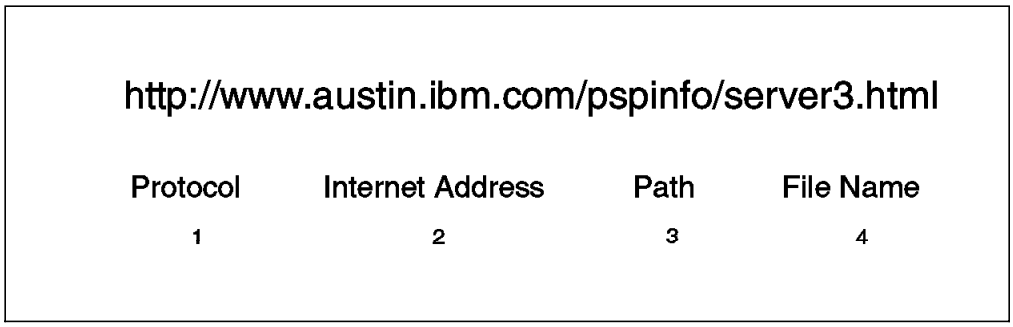


Figure 1. URL Format

1. This is the *protocol* of the server you are talking to. There are several common protocols in use on the Internet. These protocol tokens are normally written in lower case. While any case should work, some applications will only accept them in lower case.

   http Hypertext Transfer Protocol is the protocol used by Web servers and browsers.

   https Hypertext Transfer Protocol with SSL. See section 1.10.1, “Secure Sockets Layer (SSL)” on page 43 for more information.

   ftp File Transfer Protocol is used for transferring files to and from computers across a network.

   gopher The Gopher protocol provides a menu-driven interface to data.

   news The protocol used to access newsgroups, which are forums on particular subjects where people can openly discuss ideas.

   mailto The protocol used to send mail to an Internet style mail address.

2. The *Internet address* of the host. This is commonly known as the IP address of the host and can be specified either in its numeric form or by its full domain name. When a domain name is used, it is case insensitive.

3. *Path* information of the paths or directories on the host where the data resides. This path may or may not be case sensitive, depending upon the characteristics of the Web server and its file system.

4. The *file name* and extension of the file on the host that contains the data you are interested in. As with the path, this component may or may not be case sensitive.

For a more complete definition of URL formats, see the following URLs and RFCs:

- The Internet Engineering Task Force (IETF) home page at http://www.ietf.org/. There you will find links to all of the Internet standards, experimental and informational RFCs and draft documents of proposed Internet standards.
- *RFC 1738 - Uniform Resource Locators (URL)*
- *RFC 1630 - Universal Resource Identifiers (URI)*
- *RFC 1808 - Relative Uniform Resource Locators*
- *RFC 2368 - The mailto URL scheme*
- http://dir.yahoo.com/Computers_and_Internet/...
  ...Information_and_Documentation/Metadata/...
  ...URIs___Universal_Resource_Identifiers/

1.4 Home Pages

In online information, the term *page* is used to mean a discrete block of information. This block may be more than a single window's worth of information. It may also contain links to (and be linked to by) other pages.

The term *home page* is generally used to define the main or root page that is displayed for a particular organization, company, department or user ID. It is, therefore, considered the starting or entry point for people who visit a Web site. Because of this, it is important that the page gives a good first impression to Web visitors. If it takes too long to load, does not have visual appeal, or does not convey what the Web site contains, then visitors will not bother to browse further pages and you will have lost potential customers.

Web servers, such as VM:Webgateway from Sterling Software, Inc., can host pages for multiple companies or organizations either under one host name, or under multiple host names for a single physical computer (for instance, by using *virtual hosting*). In such an environment there will typically be multiple home pages, one for each organization. If your server is likely to be providing this function, for example hosting pages for various departments within your organization, it is imperative that you develop some sort of naming convention for the structure of your directories. Chapter 7, "VM/ESA Web Server Implementation" on page 149 contains some ideas and guidelines on this topic.

1.5 HTTP

HTTP is Hypertext Transfer Protocol. In simple terms, HTTP determines how a browser interacts with a server.

The version of HTTP deployed by most existing Web applications is HTTP/1.0, whose specification is available as Informational RFC 1945. A newer version, HTTP/1.1, defined by Proposed Standard RFC 2068, is evolving toward a true Internet standard. These RFCs are created under the authority of the Internet Engineering Task Force (IETF) whose home page can be found at http://www.ietf.org/. There you will find links to all of the Internet standards, experimental and informational RFCs and draft documents of proposed Internet standards, including those related to HTTP. Specifications may be obtained at the following site:

```
http://www.w3.org/pub/WWW/Protocols
```

HTTP allows distributed systems to communicate with each other. HTTP is an application-level protocol and has been in use by the WWW global information initiative since 1990.

HTTP is based on request-response activity. A client establishes a connection with a server and sends a request to the server as a request method. The server responds with a status line, including the message's protocol version and a success or error code, followed by a message containing server information, entity information, and body content. This communication is shown in Figure 2.

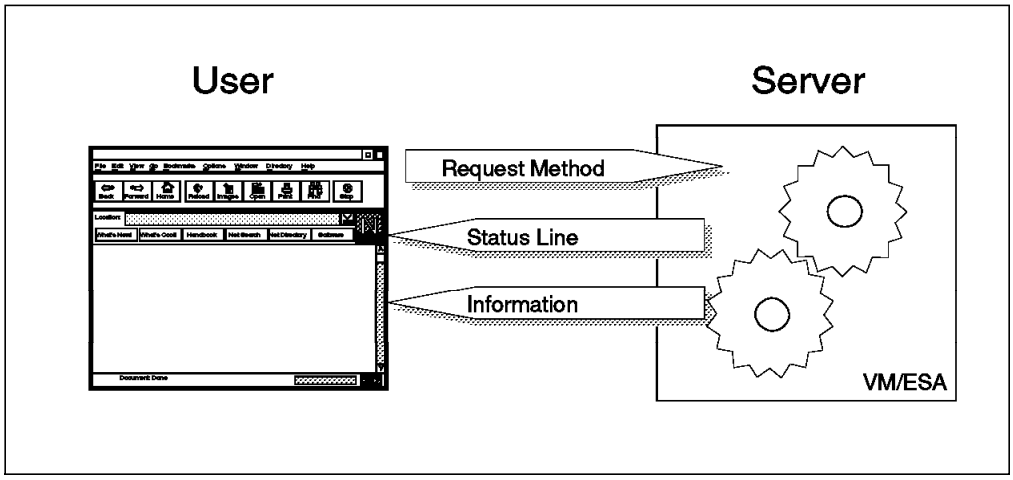


Figure 2. HTTP Client/Server Communication

An HTTP transaction is divided into four steps:

1. The browser opens a connection.
2. The browser sends a request to the server.
3. The server sends a response to the browser.
4. The connection is closed.

On the Internet, HTTP communication generally takes place over TCP/IP connections. The default port is TCP 80, but other ports can be used. This does not preclude HTTP from being implemented on top of any other protocol on the Internet, or on other networks. HTTP only presumes a reliable transport; any protocol that provides such guarantees can be used.

Except for experimental applications, current practice requires that the connection be established by the client before each request and closed by the server after sending the response. Both clients and servers should be aware that either party may close the connection prematurely, because of user action, automated timeout, or program failure, and should handle such closing in a predictable fashion. In any case, the closing of the connection by either or both parties always terminates the current request, regardless of its status.

What we have just described means that, in simple terms, HTTP is a connectionless protocol. For example, to load a page including two graphics, a graphic-enabled browser will open three TCP connections: one for the page, and two for the graphics. Most browsers, however, are able to handle several of these connections simultaneously.

HTTP is stateless, because it does not keep a record of the connections. If a request depends on the information exchanged during a previous connection, then this information must be kept outside the protocol.

1.6 HTML

Hypertext Markup Language (HTML) is the language used to write hypermedia documents for the World Wide Web (WWW). It is a subset of the Standard Generalized Markup Language (SGML); SGML is an international standard for document markup conforming to ISO 8879.

If you used a markup language previously (such as SCRIPT or GML) to create documents, you will find HTML easy to use and adapt to.

Because the World Wide Web and HTML are still in the process of evolving, the specifications of the various tags are constantly changing and being added to. The currently defined standard is HTML 2.0, as defined by the Internet Engineering Task Force (IETF), whose home page can be found at http://www.ietf.org/. There you will find links to all of the Internet standards, experimental and informational RFCs and draft documents of proposed Internet standards. Documents of interest to the HTML writer include:

- *RFC 1866 - Hypertext Markup Language - 2.0*
- *RFC 1867 - Form-based File Upload in HTML*
- *RFC 1942 - HTML Tables*
- *RFC 1980 - A Proposed Extension to HTML: Client-Side Image Maps*
- *RFC 2110 - MIME E-mail Encapsulation of Aggregate Documents, such as HTML (MHTML)*

In practice, many new features have since been added to the de facto standard language currently in use on the &www.. The new set of functions is known variously as HTML 3.0, HTML 3.2 and HTML 4.0. You can find a summary of these recommended (but non-standard) extentions to HTML by visiting http://www.w3.org/TR/. This new proposed level of HTML adds tags that can for instance be used to create tables and alter backgrounds and colors. These specifications have not been agreed on yet, and it is expected that more functions will be added over time.

HTML focuses on the content of the document and not on page layout. An HTML document consists of text and as such can be updated using any text editor. The documents created are small and can be sent over the network faster than larger documents.

HTML can be used on any platform that has a browser that understands HTML. Therefore the creator of an HTML document does not need to be concerned with the platform on which a user will be viewing the document.

There are also extensions to HTML found in browsers such as Netscape or Internet Explorer. If you use these extensions, only an updated browser can understand them; other browsers will not.

No one company or organization owns the World Wide Web and the definitions of HTML, so the way it evolves is unpredictable (although W3C does provide some guidance).

Generally, one of the companies that markets a Web browser will add support for a new tag (it is the browser that converts the tags into a viewable form based on its own rules). People will start using these tags in their Web pages, and this new tag will become a standard that other browsers must support to remain competitive.

1.6.1 Creating an HTML Document

HTML documents are created by using a text editor, word processor, or specific tools such as HTML-Wizard, HoTMetaL or HTML Assistant. HTML documents have an extension of HTM or HTML so browsers can find them.

What Is an HTML Tag?

The HTML language uses markup tags to identify the various elements of a document. The tags have predefined syntax rules, which are understood by the client browser that is interpreting and displaying (also known as *rendering*) the page.

On the home page for the World Wide Web Consortium, you can also find standards for HTML (http://www.w3.org/).

Tag Syntax

Generally, tags come in pairs. That means there is usually an opening tag and a closing tag for each function. For example:

```
<B>This is bold</B>
```

Notice how the closing tag is the same as the opening tag except it is preceded by a forward slash. Also, note how each tag starts with a less-than sign (<) and ends with a greater-than sign (>).

HTML tags are not case sensitive, so all of the following are valid:

```
<Strong>
<STRONG>
<strong>
<stROnG>
```

Text, which will be displayed by a browser, is entered between the starting and ending tags. Some tags do not have a starting and ending point.

The following example shows how an HTML document is structured:

```
<HTML>
     <HEAD>
<TITLE>
     </HEAD>
<BODY>
 ....
 ....
     </BODY>
     </HTML>
```

Three tags are used to describe a document's structure:

1. <HTML> starts and ends an HTML document. The document is created between the <HTML> and the </HTML> tags.
2. <HEAD> describes the prologue to the rest of the document. This section contains a <TITLE> tag to allow the browser to display it.
3. <BODY> delimits the rest of the HTML document. The main portion of the document is bounded by the <BODY> and </BODY> tags.

These structure tags are optional, but recommended. More tools and browsers will be able to understand your document if you use them.

It is up to the writers of the HTML to decide how they want to format their tags, although most people seem to use capitals to make their HTML easier to read.

Tip

Do not forget to write clean HTML with comments. Even though the client will format your pages before displaying them, most client browsers can easily display and locally save the HTML source files. This feature is often exploited by people who are building Web pages to get new ideas for pages and to learn new ways of using HTML.

Making your HTML pages consistent and easy to read can give your Web site a professional appearance that others can learn from. After all, the main reason for making your Web server available is to provide a service to your users.

1.6.2 Major HTML Tags

<HTML> </HTML> The HTML tag identifies the file as an HTML document.

Only one pair of HTML tags is allowed per document. Make them the first and last tags in the document.

<! ... > The ! tag allows you to create comments in your HTML document. Information written between comment tags is not displayed by browsers.

Use comments to include information in a document that you do not want to display, such as the date the document was created. Although browsers do not display the comment contents, the comment is part of the source and is accessible to the reader. Comments should only be one line long. A comment block can be formed using multiple comment lines.

<HEAD> </HEAD> The HEAD tag identifies the section of the document that contains general document information, such as the title and root URL of the document.

Only one pair of HEAD tags is allowed per HTML document.

<BASE> The BASE tag identifies the root URL of the document, which is used with relative linking. This information is necessary in cases where the document has been copied to another URL. The mandatory HREF attribute identifies the original URL of the document. In cases where the document has been moved from its original URL, this string is prepended to any relative links. The string does not consist of a pair of starting and ending tags. When used, it must be included within the HEAD tags.

<TITLE> </TITLE> The TITLE tag must occur within the HEAD tags. It cannot contain any links, highlighting or paragraph tags. There can be only one title per document. Titles should be as specific as possible. However, because of the small amount of space that most browsers allow for a title, we recommend that you not use titles of more than 50 characters. In most browsers, the title is displayed in the window title bar. Also, many browsers display the title in their history list or hot list.

<BODY> </BODY> The BODY tag identifies the section of the document that contains the text and graphics.

<ISINDEX> Use the ISINDEX tag to provide a Web search interface.

The searchable index tag indicates to the Web browser that this page provides an interface for keyword searches. The searchable index tag is a single tag; it does not have a paired starting and ending tag. The tag, if used, should occur within the HEAD tags of the HTML document. The searchable index tag automatically generates an entry field and instructions for using it.

An example follows:

```
<HTML>
<HEAD>
<TITLE>Example of Searchable Index</TITLE>
<ISINDEX>
</HEAD>
<BODY>
<H1>Example of a Searchable Index</H1>
</BODY>
</HTML>
```

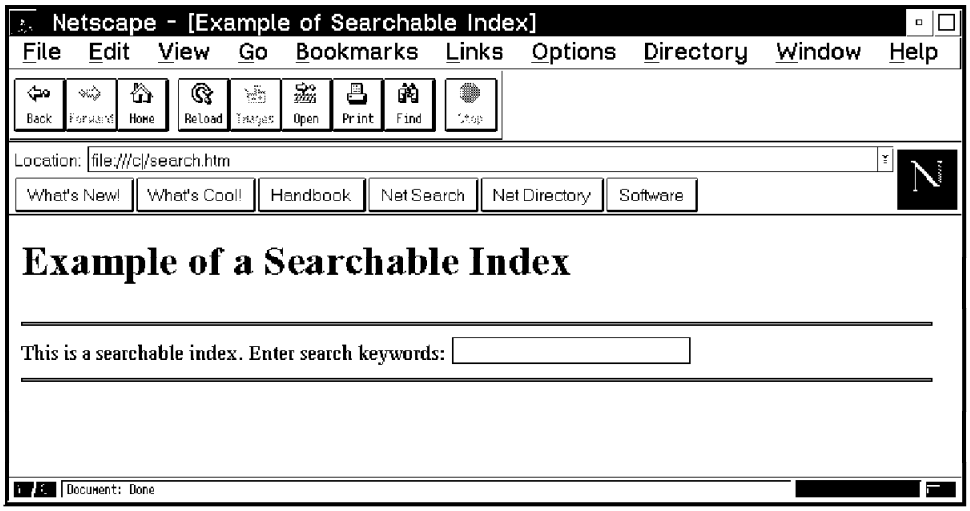


Figure 3. Searchable Index

<H1> </H1> Heading tag levels range from H1 (largest) to H6 (smallest).

You should begin each document with a primary heading <H1>. You should not skip head levels within a document. For example, if the first heading is an H1, subordinate information should begin with an H2, not an H3 tag.

An example follows:

```
<HTML>
<HEAD>
<Title>Example of Headings</TITLE>
</HEAD>
<BODY>
<H1>This is a level-one heading (H1)</H1>
<H2>This is a level-two heading (H2)</H2>
<H3>This is a level-three heading (H3)</H3>
<H4>This is a level-four heading (H4)</H4>
<H5>This is a level-five heading (H5)</H5>
<H6>This is a level-six heading (H6)</H6>
```

```
</BODY>
</HTML>
```

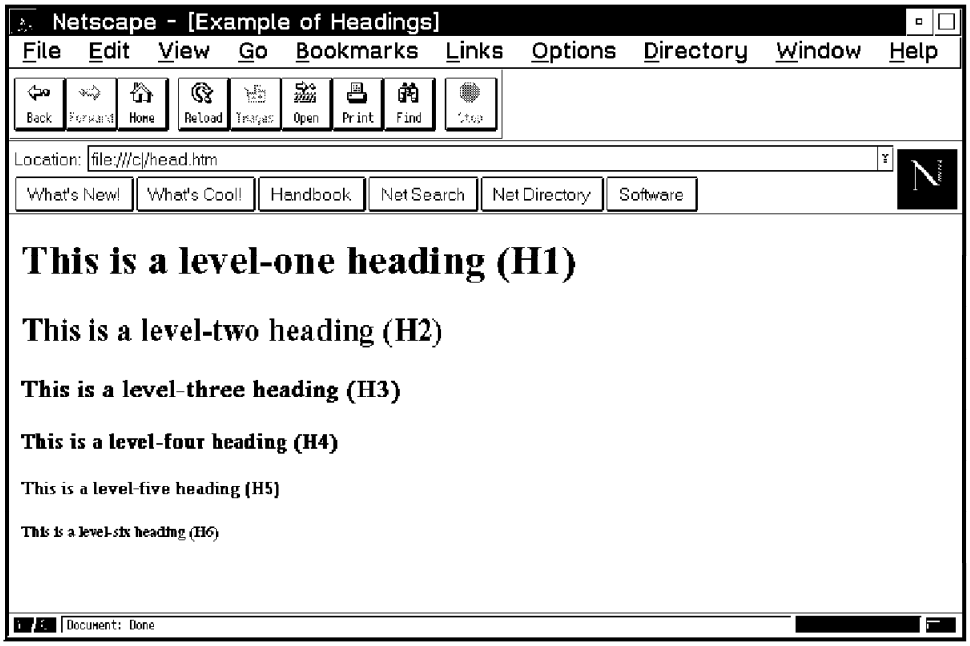


Figure 4. Different Headings

< P > < / P > You can also use the paragraph tag as a single tag <P>. If you use the single tag, place the paragraph tag at the end of a paragraph or the beginning of one, but not both. Do not use a paragraph tag with tags that also generate a blank line. Usually, a paragraph tag generates a blank line or half a blank line of space before the next body of text. With some browsers, a paragraph tag will also cause the first line of the paragraph to be slightly indented.

< A > < / A > The A (or anchor) tag is used to assign an ID to text to which you want to link.

Attributes:

- ***NAME***

  Optional. Used to assign an ID to text to which you want to link (a destination).

  For example:

  ```
  <H3><A Name="ingred">Ingredients</A></H3>
  ```

- ***HREF***

  Optional. Used to identify the URL or ID of the information to which you are linking. If the destination is outside the current page, the identifier is the URL of the destination surrounded by quotation marks. This URL can be absolute or relative. Absolute URLs specify the protocol, the name of the server and location (path and file name) of the linked file. For example:

  ```
  It is worth seeing the <A HREF="http://mistral.enst.fr"> site with
  links to various museums</A>.
  ```

  Relative URLs specify only the location (path and file name) of the linked file and must begin with a slash (/), or a file ID or directory

path, which begins relative to the current document's path. For example, to insert a link from an arbitrary page to the site's home page:

```
Click <A HREF="/">here</A> to visit our home page.
```

If the destination is inside the current page, the identifier is the ID assigned to the destination text (the value of the NAME attribute) preceded by a pound sign (#) and delimited by quotation marks. For example:

```
<H3><A NAME="ingred">Ingredients</A></H3>
...
Before you begin, make sure you have all the necessary
<A HREF="#ingred">ingredients</A>.
```

Refer to 7.4.1, "Relative URL Addressing Inside an Application" on page 153 for more information.

- ***TITLE***

  Optional. When used with the HREF attribute, it indicates the title of the destination. If the destination document or page has a title, this attribute is informational only. If the destination document or page does not have a title, as with some Gopher servers, many browsers will display the value of this attribute as the title.

1.6.3 Text Formatting Tags

Tags for highlighting text distinguish certain text from the rest of the document by displaying the text in a different manner. There are two types of highlighting:

- Explicit
- Interpreted

Explicit highlighting instructs the browser to display the text in a specific manner. Interpreted highlighting instructs the browser to display the text in whatever manner it determines is appropriate for the given emphasis.

If a browser does not know how to interpret a highlighting tag, it will not highlight the surrounded text.

HTML provides many ways of formatting text. When HTML 1.0 came out, many highlighting tags were added, such as <STRONG> and <I>.

For example, if you wanted some text to be bold, you could code either of the following:

```
<B> This is bold </B>
<STRONG> This is also bold </STRONG>
```

However, be aware that the second choice is only displayed bold if the browser has decided to interpret the STRONG tag as bold. There is no reason a later version of the browser would not decide that STRONG should be displayed in flashing pink.

So in summary, if you really want the page to look a certain way, use the physical text formatting tags.

An example follows:

```
<HTML>
<HEAD>
<TITLE>Example of Highlighting</TITLE>
</HEAD>
<BODY>
<H1>Examples of Highlighting</H1>
<TT>This is monospaced text.</TT><P>
<B>This is bold text.</B><P>
<I>This is italic text.</I><P>
<U>This is underscored text.</U><P>
<EM>This is emphasized text.</EM><P>
<STRONG>This is text with stronger emphasis.</STRONG><P>
<CODE>This represents program text.</CODE><P>
<SAMP>This represents sample text. </SAMP><P>
<CITE>This represents a citation.</CITE><P>
</BODY>
</HTML>
```

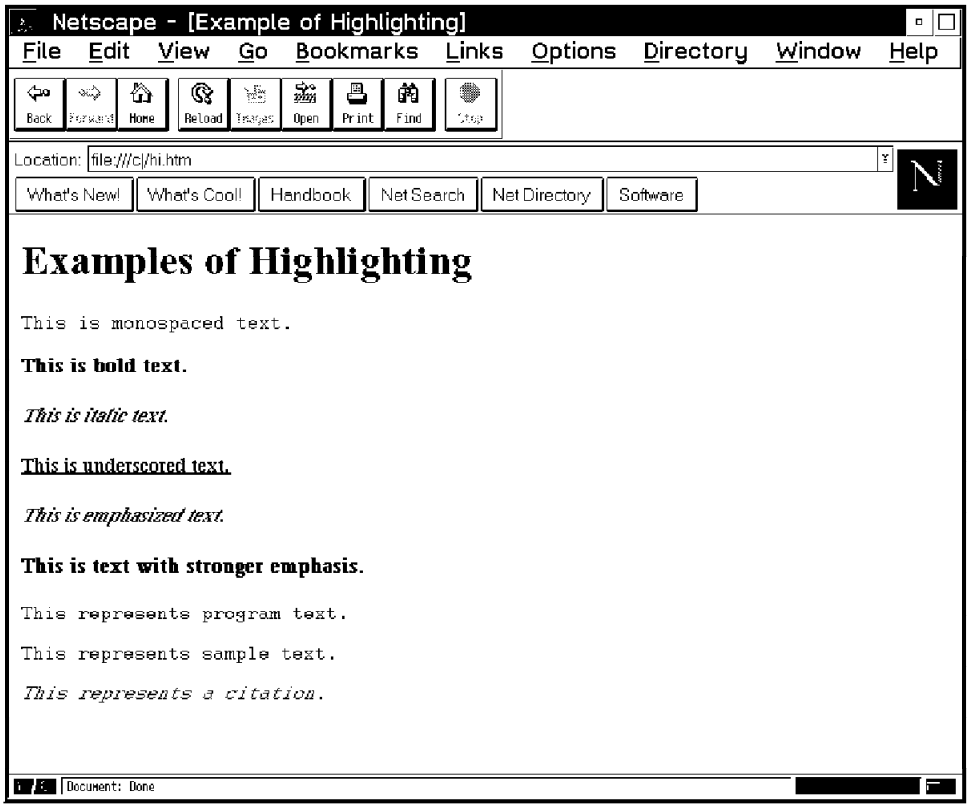


Figure 5. Different Highlighting

<TT> </TT> Monospaced text is an explicit highlighting that causes the surrounded text to be displayed in a font resembling a typewriter where every character (whether it is an i or a w) occupies the same amount of horizontal space.

<B> </B> Bold text is a thicker version of the default font. Bold is an explicit highlighting that causes the surrounded text to be displayed in a bold version of the default font.

<I> </I> Italic text is a slanted version of the default font. Italic is an explicit highlighting that causes the surrounded text to be displayed in an italic (slanted) version of the default font.

<U> </U> Underscored text is the default font with an underline. Underscored is an explicit highlighting that causes the surrounded text to be displayed in the default font with a line drawn beneath.

<EM> </EM> Emphasized text appears in a different font, usually italics.

<STRONG> </STRONG> Strongly emphasized text appears in a different font, usually bold. Strong is an interpreted highlighting that causes the surrounded text to be displayed with stronger emphasis than <EM>. Most browsers will display the text in bold font.

<CODE> </CODE> Code is an interpreted highlighting that causes the surrounded text to be displayed as simulated code. Most browsers will display the text in monospaced font.

<SAMP> </SAMP> Sample is an interpreted highlighting that causes the surrounded text to be displayed as a simulated sample. Most browsers will display the text in a monospaced font.

<KBD> </KBD> Keyboard is a simulated text entry, usually displayed in a monospaced font.

<VAR> </VAR> Variable is an interpreted highlighting that causes the surrounded text to be displayed with emphasis, indicating it is a variable (that is, it is information that can be changed, such as the value of a parameter). Most browsers will display the text in italics.

<DFN> </DFN> Definition is an interpreted highlighting that causes the surrounded text to be displayed with emphasis, indicating it is a term or phrase defined in text. Most browsers will display the text in bold or in bold italic font.

<CITE> </CITE> Citation is an interpreted highlighting that causes the surrounded text to be displayed with emphasis indicating that it cites a reference, such as a book title. Most browsers will display the text in italic.

<OL> </OL> The OL or ordered list tags generate a sequential list of individual items. Use an ordered list when the items must be addressed in a specific sequence, such as the steps of a procedure.

Place the beginning and ending tags on separate lines. The first line after the starting tag (<OL>) must be a list item and begin with <LI>. Each item in the list must be preceded by <LI>.

Each item begins on a separate line and is preceded by a number and a period.

An example follows:

```
<HTML>
<HEAD>
<TITLE>Example of an Ordered List</TITLE>
</HEAD>
<BODY>
<H1>Example of an Ordered List</H1>
<OL>
<LI>Select the desired transaction.
<LI>Enter the desired amount.
<LI>Press Enter.
</OL>
</BODY>
</HTML>
```

Figure 6. Ordered List

<UL> </UL> The UL or unordered list tag is used to generate a bulleted list of individual items. Use an unordered list when the sequence of the items is unimportant, such as a list of benefits.

Place the beginning and ending tags on separate lines. The first line after the starting tag (<UL>) must be a list item and begin with <LI>. Each item in the list must be preceded by <LI>.

<MENU> </MENU> The MENU tag generates a list of individual items without numbers or bullets. Use a menu list when the items are brief (usually no more than one line) and a sequence of the items is unimportant, such as a shopping list.

Place the beginning and ending tags on separate lines. The first line after the starting tag <MENU> must be a list item and begin with <LI>. Each item in the list must be preceded by <LI>.

<DIR> </DIR> The DIR or directory tag generates a tabular list of individual items. Use a directory list when items need to be displayed in columns such as a price list.

Place the beginning and ending tags on separate lines. The first line after the starting tag (<DIR>) must contain the items of the first row. Each item in the list must be preceded by <LI>.

Each row of items begins on a separate line. In most cases, a column is limited to about 25 characters. This is not a widely supported list type.

<DL> </DL> The DL or definition list tag generates a list of paired items. Use a definition list when you want to list and describe items, such as a list of glossary terms and definitions.

The attribute compact removes the vertical spacing between the term and its description. Place the beginning and ending tags on separate lines. Each term must be preceded by <DT>. Each description must be preceded by <DD>. For each <DT> you must have a corresponding <DD>.

**
** The BR or line break tag is a single tag; it does not have paired starting and ending tags.

Text following the break is placed on the next line. No additional vertical space is generated.

<HR> The HR or horizontal rule tag generates a line across the page (no ending tag). Use horizontal rule tags to create a visual break in the page.

<ADDRESS> </ADDRESS> The ADDRESS tag combines interpreted highlighting and paragraph formatting. Use address tags to set off document identification information and addresses.

Paragraph tags used within address tags act as line break tags. They do not generate additional vertical space. To generate a stacked address format, similar to the format of an address on an envelope, use line break tags inside the address tags.

As with interpreted highlighting tags, such as CITE, the browser controls the emphasis that is placed on the surrounded text. In general, most browsers will display the text in italic. Also, the ending address tag acts as a paragraph tag, causing a line break and generating additional vertical space before any following information.

An example follows:

```
<HTML>
<HEAD>
<Title>Example of an Address</TITLE>
</HEAD>
<BODY>
<H1>Example of an Address</H1>
<ADDRESS>
Pat Smith Software<BR>
3039 Cornwallis Road<BR>
RTP, NC 27709
</ADDRESS>
</BODY>
</HTML>
```

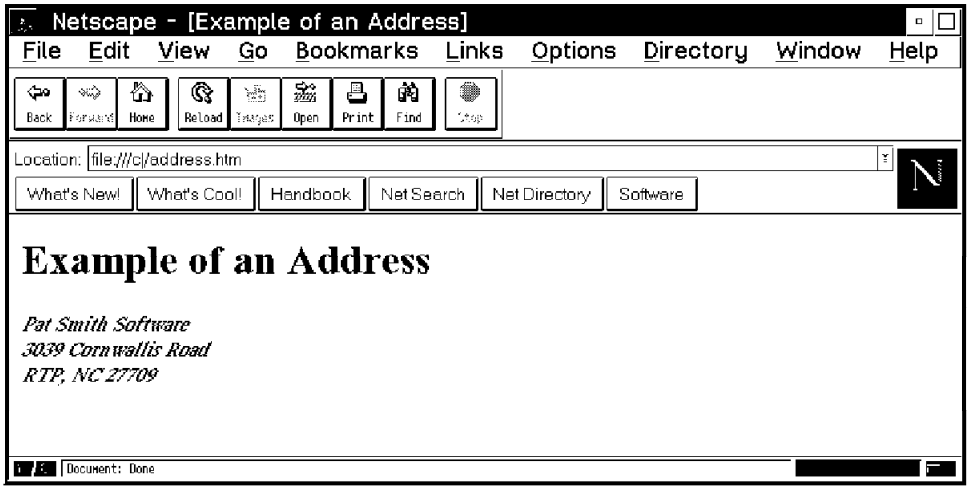


Figure 7. Address

<BLOCKQUOTE> </BLOCKQUOTE> The BLOCKQUOTE tag highlights text intended as a quote, usually in italic font.

As with interpreted highlighting tags, such as CITE, the browser controls the emphasis that is placed on the surrounded text. In general, most browsers will display the text in italic. Some also indent the margins for the surrounded text. Also the ending block quote tag acts as a paragraph tag, causing a line break and generating additional vertical space before any following information.

<PRE> </PRE> The PRE or preformatted text tag combines interpreted highlighting and line breaks. Use preformatted text tags to set off information that needs to be displayed as it is in the HTML file, for example, lines from a program or sample file.

HTML tags included within preformatted text tags are interpreted as tags. If you want to include an example that shows HTML tagging, you must use the HTML symbols (*<* and *>*) to create the tag delimiters (< and >). See 1.6.8, “Special Characters” on page 22 for more information. Highlighting and linking can be used within preformatted text. However, headings and formatting tags, such as paragraph and address tags, should not be used within preformatted text.

<IMG> The IMG or image tag allows you to include an inline image in your document.

Image tags can take the following attributes:

- ***SRC*** - Required. Specifies the source file of the inline image. It is followed by a URL that identifies the image file and its location.
- ***ALIGN*** - Optional. Specifies how the image should be aligned vertically. Possible values are TOP, MIDDLE, and BOTTOM.
- ***ALT*** - Optional, but highly recommended. Specifies the label for the image that is to be displayed if the browser does not support inline graphical images.
- ***BORDER*** - Optional. Specifies the thickness, in pixels, of the border surrounding the image. The default is 1. Specifying a value of 0 produces a borderless image.
- ***HEIGHT*** - Optional. Specifies the suggested height for the image. By default, this is given in pixels.
- ***WIDTH*** - Optional. Specifies the suggested width for the image. By default, this is given in pixels.
- ***UNIT*** - Optional. Specifies the units for the width and heights attributes. It is one of: “unit=pixels” (the default), or “unit=en” (one half the point size).
- ***ISMAP*** - Optional. Specifies that the image contains defined areas that, when selected, link to other URLs.

The image tag is a single tag. It is not a paired set of starting and ending tags. Some browsers cannot display inline images, but can display linked images. If an image is crucial to a document, you may want to link to it rather than include it inline.

For browsers that can display graphical images inline, the image is left justified. If a series of images is specified, the images are

displayed on the same line, if possible. For browsers that cannot display images inline, the label, if any, is displayed.

An example follows:

```
<HTML>
<TITLE>Example of Inline Graphics</TITLE>
<BODY>
<H1>Example of Inline Graphics</H1>
This site contains information about the following Acme products:
<P>
<IMG SRC='ball_pur.gif'><A HREF='st200.html'>Spring Trap Model 220</A><P>
<IMG SRC='ball_pur.gif'><A HREF='cat40.html'>Spring Large Model 40</A><P>
<IMG SRC='ball_pur.gif'><A HREF='tun50.html'>Spring Tuned Model 50</A><P>
<IMG SRC="/av/pix/altavista.gif" BORDER=0 ALIGN=middle HEIGHT=54 WIDTH=9>
<IMG SRC=line_owl.gif><P>
<IMG SRC='mailbox.gif'>Feel free to drop a card in our<A HREF='mail.html'
>suggestion box</A>.
</BODY>
</HTML>
```

Images are mostly GIF files. They are included on pages using the IMG tag. You specify where the image is located using the SRC attribute. The SRC attribute uses the same file conventions as the HREF attribute. Just put the IMG tag where you want the image to appear. Instead of a GIF image file, you can also use the following formats:

- JPEG compressed image format (.jpg)
- Wave format sound files (.wav)
- MPEG audio (.mp2)
- MPEG video (.mpg)
- AVI video (.avi)

1.6.4 Creating Forms

The WWW can be considered a large database of interlinked documents that users navigate around using hypertext links. This basic scenario provides a powerful medium for users to access data, but without extra function, users will not consider it to be fully interactive. This interaction is currently provided by the use of forms and Common Gateway Interface (CGI) scripts. New technologies, such as JAVA, which is discussed in section 1.7, “Java” on page 31, are being developed to add to this function.

Forms are the basic way the Web server retrieves information from clients. This information is then processed by a CGI script (for more information on CGI see 1.8.1, “Common Gateway Interface (CGI)” on page 33). The form consists of a normal HTML page that contains special HTML tags that provide entry fields, push buttons, radio buttons, and so forth, and details of the CGI script to be invoked.

The FORM tag allows you to create input forms. Data from these forms can be submitted to a Web server for processing and analysis.

Example of creating a form:

```
<HTML>
<HEAD>
<TITLE>Example of a Form</TITLE>
</HEAD>
<BODY>
<H1>Example of a Form</H1>
<FORM METHOD=POST ACTION="mailto:survey@vnet.ibm.com">
Please complete this survey so that we may better improve our service.<p>
Name: <INPUT TYPE="text" NAME="name" SIZE=45 MAXLENGTH=50
VALUE="Enter your name here"><P>
Address: <TEXTAREA NAME='address' COLS=40 ROWS=3>
Enter your address here.
</TEXTAREA><P>
Account ID: <INPUT TYPE="text" NAME="acctID" SIZE=30><P>
Password: <INPUT TYPE="password" NAME="password" SIZE=10><P>
Which application do you use?<P>
<INPUT TYPE="checkbox" NAME="apps" VALUE="Gopher"> Gopher
<INPUT TYPE="checkbox" NAME="apps" VALUE="Newsreader">
Newsreader
<INPUT TYPE="checkbox" NAME="apps" VALUE="WebExplorer">
WebExplorer
<INPUT TYPE="checkbox" NAME="apps" VALUE="Archie">
Archie
<INPUT TYPE="checkbox" NAME="apps" VALUE="FTP">
FTP
<INPUT TYPE="checkbox" NAME="apps" VALUE="Telnet">
Telnet<P>
How would you rate your overall satisfaction with our applications?<P>
<INPUT TYPE="radio" NAME="rating" VALUE="1"> Very satisfied
<INPUT TYPE="radio" NAME="rating" VALUE="2"> Satisfied
<INPUT TYPE="radio" NAME="rating" VALUE="3"> Neutral
<INPUT TYPE="radio" NAME="rating" VALUE="4"> Dissatisfied
<INPUT TYPE="radio" NAME="rating" VALUE="5"> Very dissatisfied<P>
Reset fields: <INPUT TYPE="reset">
Select to submit your responses: <INPUT TYPE="submit"
Value="SEND"
</FORM>
</BODY>
</HTML>
```

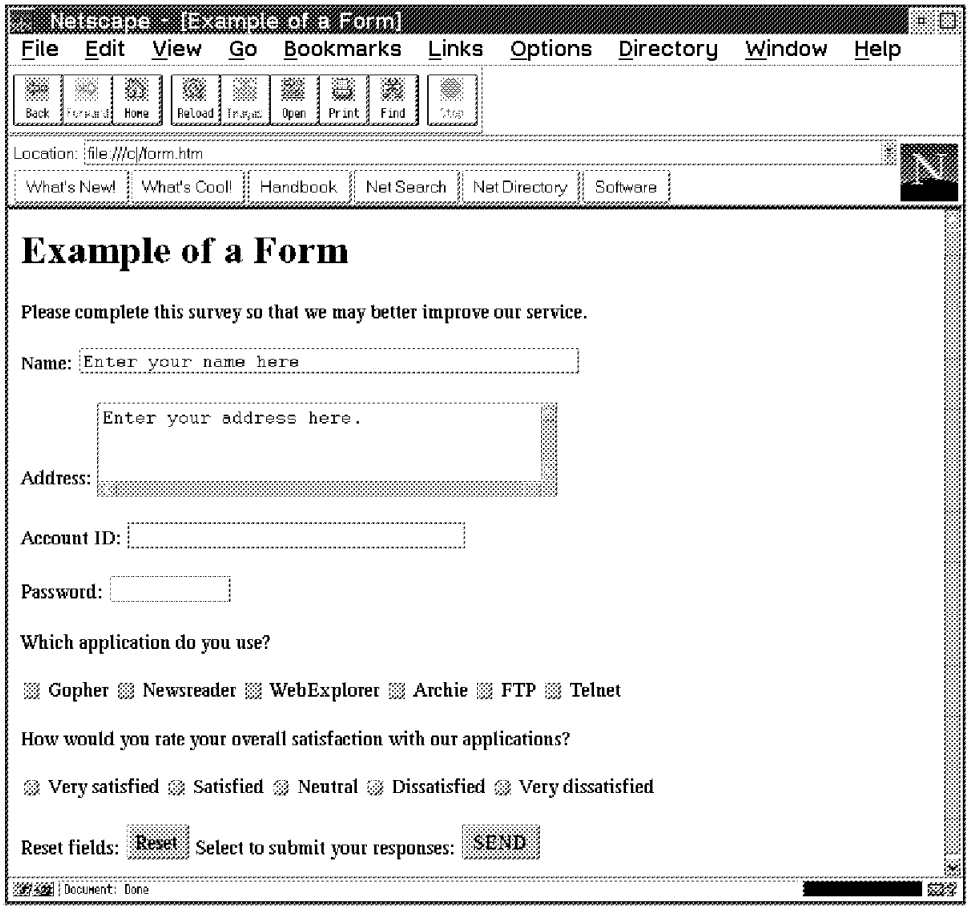


Figure 8. An Example of a Form

<FORM> </FORM> The FORM tag indicates that the surrounded information is part of a data entry form.

Form tags can take the following attributes:

- ***METHOD*** - Required option. Indicates the HTTP method used to send the data to the server. Possible values are GET and POST. It is recommended you use POST.
- ***ACTION*** - Optional. Specifies the URL of the processing script. The URL must be enclosed in quotation marks. If the ACTION attribute is not specified, the default is the URL of the document.
- ***ENCTYPE*** - Optional. Specifies how the input data is encoded.

<INPUT> </INPUT> In an HTML document, the INPUT tag can be used to create elements that accept user input. These elements may be selections or entry fields.

The attributes that an input tag can take depend on the input type.

- ***TYPE*** - Required.

  Specifies the type of fields to be displayed. Possible values for field type are:

  – CHECKBOX

    Displays a box that can be selected. Use check boxes for Boolean selections (on or off, yes or no). Multiple check

boxes can be grouped together. One or more check boxes in a group may be selected.

Required attributes are NAME and VALUE.

An optional attribute is CHECKED.

– HIDDEN

Does not accept or display any information to the user. Hidden form fields are used to send status information to the server.

A required attribute is NAME.

An optional attribute is VALUE.

– IMAGE

Displays a graphic, which, when selected, submits the data to the specified URL. Because IMAGE may be obsolete in the future, it is recommended that you use the SUBMIT form type.

Required attributes are NAME and SRC.

An optional attribute is ALIGN.

– PASSWORD

Displays a single line entry field. Password fields are similar to text fields except information entered in this field is not displayed.

A required attribute is NAME.

Optional attributes are MAXLENGTH, SIZE, and VALUE.

– RADIO

Displays a radio button (a circle that can be selected). Use radio buttons for multiple choice selections where only one in a series can be selected. All radio buttons in a group should be assigned the same NAME.

Required attributes are NAME and VALUE.

An optional attribute is CHECKED.

– RESET

Displays a push button that when selected returns all the form′s fields to their original values. Use the VALUE attribute to define the label for the push button. The default label is RESET.

An optional attribute is VALUE.

– SUBMIT

Displays a push button that when selected, submits the data. Use the VALUE attribute to define the label for the push button. The default label is SUBMIT. You can also use the SRC attribute to include an image on the push button.

Optional attributes are NAME, SRC, and VALUE.

– TEXT

Displays a single-line entry field. If you require a multiple-line entry field, use the TEXTAREA tag.

A required attribute is NAME.

Optional attributes are MAXLENGTH, SIZE, and VALUE.

- ***SRC*** - Required with type="image".

  Specifies the URL of the graphic to be displayed.

- ***ALIGN*** - Optional.

  Specifies how the graphic should be aligned vertically. Possible values are TOP, MIDDLE, and BOTTOM.

- ***CHECKED*** - Optional.

  Indicates the default section.

- ***MAXLENGTH*** - Optional.

  Indicates the maximum number of characters that can be entered into a field. If MAXLENGTH is greater than SIZE, the entry field will allow the field to scroll as information is entered.

- ***NAME*** - Required.

  The identifier assigned to a field. When a form is submitted, the name of a field is paired with its value.

- ***SIZE*** - Optional.

  Specifies the width in characters of a field area.

- ***VALUE*** - Required with TYPE="checkbox" and TYPE="radio".

  For entry fields, VALUE is used to specify the initial setting. For check boxes and radio buttons, VALUE is used to specify the value assigned to a selection. For SUBMIT and RESET, VALUE is the label to appear on the push button.

<TEXTAREA> </TEXTAREA> The TEXTAREA tag can be used to create multiple-line entry fields.

TEXTAREA tags can take the following attributes:

- ***ROWS*** - Required.

  Indicates the number of rows of input allowed.

- ***COLS*** - Required.

  Indicates the number of characters allowed for each row.

- ***NAME*** - Required.

  The identifier assigned to a field. When a form is submitted, the name of a field is paired with its value.

<SELECT> </SELECT> The SELECT tag is used to create a list box of choices, similar to the radio or check box input types.

Select tags can take the following attributes:

- ***MULTIPLE*** - Optional.

  Indicates that multiple selections are allowed.

- ***SIZE*** - Optional.

Indicates the number of items displayed in the selection list at one time. If SIZE is less than the number of items listed, a scroll bar is displayed to the right of the field, allowing users to scroll through the other items in the list.

- ***NAME*** - Required.

  The identifier assigned to a field. When a form is submitted, the name of a field is paired with its value.

<OPTION> </OPTION> The OPTION tag is used with SELECT tags to create a list of choices, similar to the radio or check box input types.

Option tags can take the following attributes:

- ***SELECTED*** - Optional.

  Indicates that this option is the default.

- ***VALUE*** - Optional.

  The value to be paired with the name if this option is chosen. If no VALUE is specified, the text associated with the option is sent as the value.

1.6.5 Sample Document

The following example puts together some of the tags discussed and shows a sample template on which to base future HTML documents.

```
<html>
<head>
<TITLE> my little sample HTML example     </TITLE>
                                          </HEAD>
<BODY>
<H1> HTML is fun and easy to learn        </H1>

<P>  WELCOME
     This is the first paragraph          </P>

<P>  This is the second paragraph         </P>

                                          </BODY>
                                          </HTML>
```

The required elements are the <HTML>, <HEAD> and <BODY> tags with their end tags. Because you should include these tags in each file, you want to create a template file with them for further use.

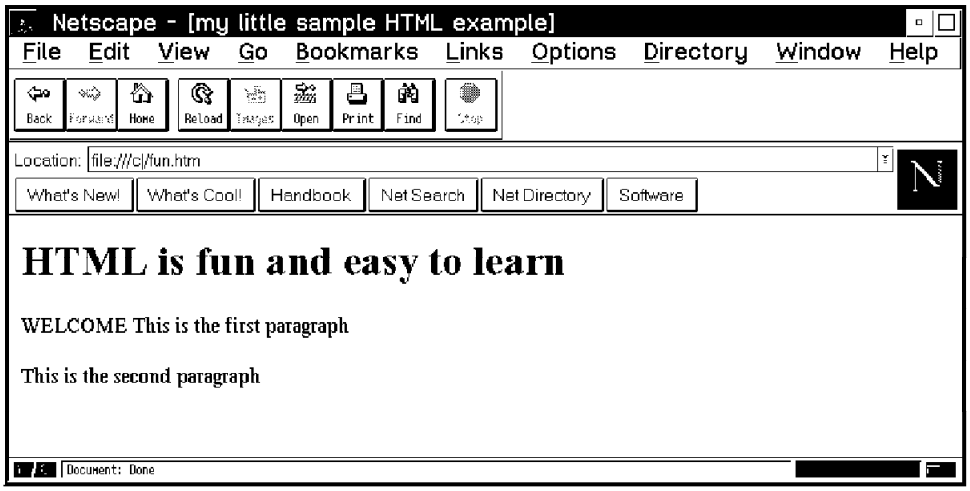


Figure 9. Web Browser View of Sample Document

If you require a complete definition of the HTML commands, you can choose from a wide range of books. Also, there are numerous WWW links that will assist you in the proper use of HTML. For example:

`http://www.ncsa.uiuc.edu/General/Internet/WWW/HTMLPrimer.html`
`http://www.ucc.ie/info/net/htmldoc.html`

1.6.6 Netscape-Specific Extensions

Here are some extensions to HTML that have been provided by Netscape:

NOBR No line break.

WBR Word break.

BASEFONT Specifies the default font size for the document.

BLINK Marks the enclosed text as blinking text.

APPLET Used to include an inline applet.

PARAM Defines a parameter for an applet.

FRAMESET Defines the layout of window frames within the browser window.

SCRIPT Used to include program scripts within an HTML document.

1.6.7 Microsoft Internet-Specific Extensions

Here are some extensions to HTML that have been provided by Microsoft:

MARQUEE Denotes a text string to be scrolled horizontally on the display; the content of the element is the text to be scrolled.

BGSOUND Inserts an inline audio snippet. By default, the sound is loaded and played once.

1.6.8 Special Characters

In this context, the term *special characters* includes National Language Support (NLS) characters and characters such as trademark (™) and copyright (©). They are supported in two different ways in HTML:

1. By using their ASCII equivalent number

   For example, the copyright symbol would be displayed as &#169

2. By using a special abbreviation

   For example, the copyright symbol can also be displayed as &copy.

Tip

If you are going to write pages in a language other than American English, it is preferable to use these methods of displaying the special characters of your language, rather than just typing them on your keyboard. This will ensure that they are displayed correctly, regardless of the code page that is being used by the client. Also remember that although you may be initially targeting users that are in your own country and speak your own language, your pages could (if your VM Web server was attached to the Internet) be viewed by people worldwide. Therefore, it may be worth while constructing an English version of your home page as well as the one in your national language.

1.6.9 Frames

Frames are a new method of designing your Web pages and are not accepted (at the time of this publication) by the WWW consortium as a standard part of HTML.

Frames are an extension of HTML made by Netscape Communications to divide a browser window into multiple, independently scrollable areas.

Hyperlinks in one frame can be used to update the contents of other frames. This gives the Web programmer the possibility of designing pleasant and sophisticated Web pages that provide information in a contextual way.

If you are using a browser that is not able to build frames, the Web page can be divided into two sections:

- One section for browsers that support frames
- One section for browsers that cannot build pages containing frames and those information consumers who do not like frames (as some people do not)

This technique allows everyone to access the Web page without loss of information.

You can find more information about frames on the Internet by using the following search arguments on the WWW search engines:

- Netscape
- Frames
- Tutorial

1.6.10 Images

Images are widely used on the Web, as they make documents easier to read and pages more visually appealing. Along with Hypertext links, the ability to show images is one reason the World Wide Web has become such a popular medium.

It is the Web browser that interprets the HTML document sent from the Web server. Once the browser has requested and received a page, it scans the HTML source, trying to interpret all of the tags that it finds. When it comes

across an <IMG> tag, it knows it has to request an image file from the Web server. The image file is then sent by the server to the client in binary form without the server doing any translation or checking, so the image can be in any format if the browser has the capability of displaying images in that format. The most common image types supported by browsers are Graphic Interchange Format (GIF) and Joint Photographic Expert Group (JPEG).

Most browsers have the capability of caching images and documents; their numbers (or rather the size of the cache) are generally determined by the user. When the browser loads an image, it saves a copy locally on the client, normally in the `/tmp/` directory. Then, when the same file name is used a second time for an image, the browser displays the image that is available locally from its cache instead of requesting another copy to be transferred from the server. This considerably improves performance by reducing the number of URLs resolved and the amount of data transferred over the network.

Tip

For caching to work successfully, each reference to the image on the various pages of HTML must use exactly the same file name, including the same path, with the same case, to enable the browser to determine that the image requested is the same as what is already in the cache. For example, if one reference to an image uses a fully qualified file name and path, and the other just uses the file name (because the HTML is in the same path as the image), the caching will not work. The browser cannot guarantee that both references are for the same image. It has to request a new image from the server.

Browsers also have the capability of suppressing graphics, which means that when the browser has received the HTML document, it bypasses the references to any <IMG> files it finds. This option is often set by remote users who are accessing the Web server with a dial-up link of some description, especially if the link is not very fast, as the transfer time for a large image can be prohibitive. The Charlotte VM Web browser does not support graphics. You should therefore always code your <IMG> tags with an alternative text description that will be displayed if image transfer is disabled by the browser. You should also make sure that any links or important information relayed as images is also available in textual form.

You can manipulate an image in many different ways to make it display differently. Some of the most common techniques used are transparency and interlacing.

Transparency This involves making one color of a GIF (usually the background) transparent. The following are the two major types of GIF files:

GIF87 These cannot be made transparent.

GIF89a This format can have one of the colors changed to make the image transparent.

A normal GIF is rectangular. If you make the background color transparent and the image is of a circle, the image will appear circular instead of being a circle in a rectangular box. This also lets the Web page background

color or image to show through the image that is placed on the page.

Interlacing This involves saving the image in a special format so when the image is displayed by the browser, the detail of the image is built up by three or four passes instead of slowly being painted in full detail from the top down.

Although the time taken to display the complete image is the same, it gives the appearance of being faster to load because the user can determine the purpose of the image before all of the detail is added.

Tip

If you want to get several pictures on your Web page in a hurry and you do not have access to a scanner, think about taking a roll of film and getting it developed onto a Photo CD. Or if you want some charts from a spreadsheet on your Web pages, display them on your PC and use a screen capture program to save them as a GIF.

1.6.11 Meta Information

Meta tags contain information about information. That is, they contain information about the file that contains the information that the Web server is making available to clients on the Internet or intranet.

Currently, Web browsers do not use the information available on Meta tags, but this will probably change in the future.

1.6.12 Multimedia

Multimedia is the logical extension beyond providing static images to a Web browser.

The term *multimedia* includes standard images, moving images (either computer generated graphics or captured video), and sound. Simple, static images are often not classed as multimedia objects.

Multimedia is supported in many different ways on different platforms. For example, sound files can be in many different and incompatible formats (.WAV .AU .MIDI). Each format requires a separate application to process it, so it may not be supported by every Web browser that your users are using.

The latest multimedia enhancement are 3D images. Users can control their perspective views of the image served.

Remember that the limiting factor to what multimedia you can include in your Web pages is not the Web server, but the browser that the user is using.

1.6.13 Creating Effective Online Information

Although there are many printable documents available on the Web, the primary purpose of the Web is to provide information in an online format.

Regardless of whether you are producing printed or online information, there are three aspects to consider:

1. Style
2. Content
3. Mechanics

This section provides some style guidelines and tips to help you produce Web information.

Understanding Form

Information is often displayed in sizable windows. You do not have control over the size of the window as you would with a printed book. Therefore, you should steer away from complex formats, such as multiple columns and large amounts of information in tabular format.

Often, the form can reinforce the content. If the information describes a procedure, place the steps in an ordered (numbered) list. If the information contains a set of choices, benefits, or considerations, use an unordered (bulleted) list. Lists can enhance the appearance of information and aid in user comprehension.

Deciding When to Link

Sometimes, despite attempts to keep it simple and concise, a topic cannot be covered in a few pages of information. This situation calls for linking. Consistency is one of the most important guidelines of linking. It is important that you adopt well-defined criteria for deciding when to include information inline and when to separate it out as a entity to which you link. Users become accustomed to how information is presented. For example, we are all accustomed to finding the index in the back of the book. Because online information is a young medium, the rules of consistency are still being developed.

In general, you should design the initial page of information for the high-level user, the user who needs the least amount of information. Then, provide links to additional information, such as definitions of terms that a novice user might not understand, technical information that an experienced user might want, or information about related topics. This will help reduce the amount of information on the initial page.

General Guidelines for Web Information

In this section some general guidelines for providing information in HTML documents on the Web are presented.

Be Brief: The acceptable length of a Web page document varies. First, try to limit the number of times a reader has to scroll through a document. Three times is a good limit. However, remember that the acceptable length of a document should also depend on the topic. Everything considered, high-level documents, such as overviews, should be kept short. The reader expects this type of information to be brief. In-depth documents are expected to be longer. Remember, you can separate out and link to related information. But do not try to break up a document unnaturally just to make it shorter.

Another element to consider is time. The information a reader is accessing is likely to reside on a computer hundreds or thousands of miles away. The amount of time it takes a browser to access and display a document will depend on the speed of the reader's connection. For example, using a computer

attached to a company network that is connected to the Internet with a dedicated line, it might take 12 seconds to display the entire document. Displaying the same document using a computer with a 14,400 baud dial-in connection to the Internet might take 23 seconds. Individually, these access times may seem acceptable. But when you consider that readers may be waiting several seconds for each document, they may not be willing to wait for your document to be displayed if it takes a few seconds longer than expected.

Provide Navigation Aids: It is not unusual for a reader to get lost perusing the thousands of interconnected documents on the Web. In addition to any links that you may have in your document, it will be helpful to your readers if you provide push buttons or icons at the bottom of your document that link back to the parent document or forward to another related topic. If yours is an exceptionally long document, you might consider providing a link at the bottom to return the reader to the top.

Clearly Identify the Document: When you send out a memo, write a letter, or provide a report for others to read, it is important that you include the date and your name. This helps those who receive the document to know where it came from and when. Considering the increased distribution that your document may experience on the Web, this identification is even more important.

Although HTML allows you to easily link to documents on other servers, sometimes people prefer to copy a document to their own server. Therefore it is a good idea to always clearly identify your document at the beginning of it. You should include the date, the status (such as Draft or First Revision), and your name (including your e-mail address). Including your e-mail address will allow others to contact you if they have comments or notify you if they plan to link to your document.

Construct Links Appropriately: The intent of hypertext coding is to allow you to highlight phrases that, when selected, lead the reader to additional information. However, the content of a document should read well without the links. Consider, for example, if someone wanted to print a document. The links would no longer be functional, but the sentences would nonetheless read coherently. An example is found in Figure 10.

```
If you want more information about any of these products,
contact your IBM Marketing Representative.
```

Figure 10. A good link form

Here, the words **IBM Marketing Representative** provide a link to information about how to contact IBM using a phone or e-mail. However, if you remove the link, the sentence still makes sense.Figure 11 demonstrates a poor use of linking.

```
If you want more information about any of these products, click here.
```

Figure 11. A bad link form

If the link is not operational, the sentence no longer contains useful information. Restructuring it as in Figure 12 on page 28 makes this into an even more useful link.

```
If you want more information about any of these products,
contact your IBM Marketing Representative at ibmrep@US.IBM.Com.
```

Figure 12. The best link form

Note that this link is even more useful than the first example because even if the link is not functional, not only does it read correctly, but it also tells you who or what to contact in detail, not just with a title.

Avoid Copying Documents: When you find information that you want to include in your document, do not copy it; simply link to it. Many of the documents on the Web are living documents, which means that they are likely to evolve and change. If you copy a document to your server, the author may update the original document, making your copy obsolete. If you link to the document, your readers will always be linking to the latest information.

Once you begin to maintain your own Web server, you will be surprised at how quickly you can run out of space if you do not plan and manage your information well. In addition to saving time (with change management), linking to a document rather than copying one will also save space.

Use Active Links: Make references to other Web documents and e-mail addresses as active links. If the information is worth referring to, it is worth making it easy for the reader to use it with a simple click.

Using Graphics: Hypertext documents on the World Wide Web represent a significant achievement in documentation. More information is available to more people and the retrievability of the information is easy. With the introduction of graphics and multimedia, these documents can be more effective than their printed predecessors.

Graphics can enhance a document. An illustration of a concept is sometimes more significant than words. In most cases, a graphic can convey a message without words. The graphic is therefore language-independent, making your document more useful to those who may not speak your language. Graphics can also add flair to a document, making it more interesting to your reader.

However, using graphics in a Web document is not without its problems. Some browsers do not support inline graphics. Therefore, if your graphic contains important information, such as links to other information, you will need to supply a textual equivalent for the graphic.

Graphics are transferred separately from the text of a Web document; they require additional transfer time. For a reader with a dial-in connection, the additional time can cause the reader to lose interest and cancel the request for the document. This is particularly true for readers who have a browser that does not display Web documents until all associated graphics have been transferred.

Using Multimedia: Multimedia (video and audio clips) can also enhance the appeal and usefulness of a document. Video clips bring a concept to life. For example, without multimedia, an overview of heart functions would take paragraphs of text and include numerous medical terms that require further explanation. Many readers would probably abandon the document after they hit the second or third paragraph. By including a video clip, the action of the heart comes to life. The video clip captures the attention of the reader while providing the information in a format that can be understood by readers of various knowledge levels. Add to that an audio clip that narrates the action portrayed in the video, and it is like having a biology class in your computer.

It is quite easy to include video and audio clips in your Web document. Once you have created your video or audio file, it is simply a matter of linking to that life. The server and browser take care of the rest.

However, beware of relying too heavily on multimedia to convey your message. Not all browsers are capable of audio and video playback. Also remember that for most browsers, the playback of these files is provided through a separate program. Therefore, the entire audio or video file must be transferred to the reader′s computer before it can be played back. This will require additional wait time, additional disk space, and additional memory. This may prevent some of your readers with more modest equipment from viewing or hearing your message. In addition, some of your information consumers will undoubtedly be connected via slow TCP/IP connections (such as dial-up lines) and run on slow, low-function rendering engines (such as 100MHz PCs with black and white monitors and no sound). These information consumers should not feel disfranchised by the form your message takes on the Web.

Summary: Write your HTML to be both device and browser independent. A document should include:

- Your sign - Include your name, electronic address and phone number in the document, so that the reader can identify the author of a document. Include also the date when the document was last reviewed.
- Navigational icons - Use icons to make it easier to navigate through a long document. Icons may be used to go back to the top of the document, back to the chapter beginning.
- Title - The title element should occur in the head of the document and identify the content of the document. The title should ideally be less than 64 character in length. Many applications will display documents in window titles, where there is only limited place to display.

1.6.14 HTML Editing Tools

HTML documents can be written and maintained in many different ways.

These range from using a simple text editor, through a word processor that may have built-in support for HTML, right up to some of the specialist tools currently available, which handle all the tags for you and can display the page dynamically as you create it using a WYSIWYG approach.

Although most of the WYSIWYG editors or word processors may help you create the layout and most of the pages, you will find some cases where they do not give you the flexibility that editing the text does. Pages edited on the workstation may be uploaded and stored on the VM Web server. An HTML

editor assists you in creating Web pages. Depending on the power of the editor, a full set of HTML tags may be supported.

Many of these packages are shareware that can be found on the Internet. The easiest way to find them is to use one of the search facilities on the Internet to track down what is currently around.

In the following, several search engines of the World Wide Web are listed:

- http://www.yahoo.com
- http://www.altavista.com
- http://webcrawler.com
- http://search.com
- http://dogpile.com

The following is a list of some of the editing tools available on the Internet:

- Boxer
- HTML Assistant
- HTML Hyperedit
- HTML Generator
- HTML Wizard

  Figure 13 shows a sample screen from HTML-Wizard.

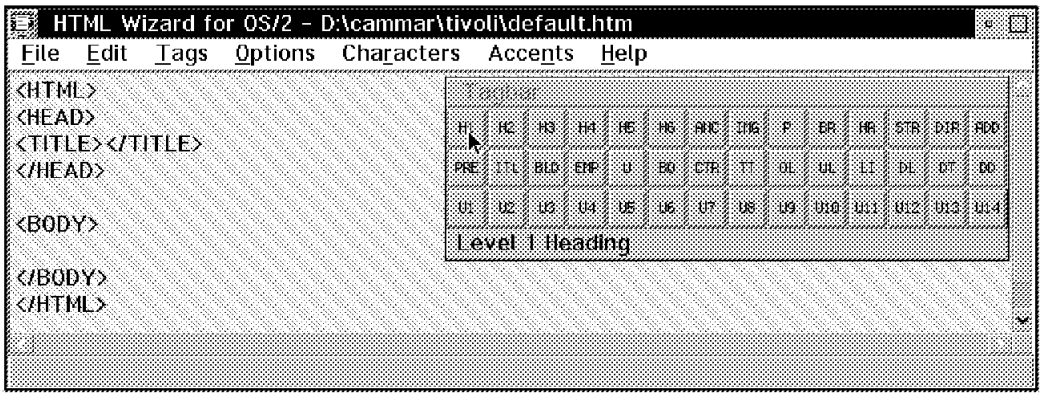


Figure 13. HTML-Wizard

- HotDog
- HoTMetal
- SpHydir
- WebWriter/2
- EPM with HTML extensions

The following are GIF/JPEG editing programs:

- PMJpeg
- WebGif

HTML Validation Service

This service uses HTML forms and a CGI script that runs an HTML validation program.

Complete information may be found at the following URL:

`http://validator.w3.org/`

1.6.15 Icons and Clip Art

To produce HTML documents containing icons and clip art, you may find the resources available from the following URL useful:

`http://www.cli.di.unipi.it/iconbrowser/icons.html`

1.6.16 Best of the Web

A good way to learn what works is to see examples of good Web sites. A link with references to other URLs that show the best pages is at:

`http://www.yahoo.com/Computers_and_Internet/Internet/World_Wide_Web/Best_of_the_Web/index.html`

PC Magazine Top 100 Web sites may be found at:

`http://www.pcmag.com/special/web100`

1.7 Java

Java is an important new technology in the world of the Internet. In summary, it is a simple, robust, object-oriented, platform-independent, multithreaded, dynamic general-purpose programming environment for creating applications for the Internet and intranet. Java was developed by Sun Microsystems Inc. It includes the following components:

- Java

  Although Java is a general purpose programming language, it is most closely associated with Web applications. The most common type of Java program, called an *applet*, is designed to be transmitted over the Web to run on any client platform. Other types of Java programs are used to extend the capabilities of Web servers. Like all Java programs, an applet is a byte-code program; that is, the program is compiled into instructions that are independent of the platform the client is running on. Instead, the client runs a program (normally part of the browser) that can interpret and run the byte-code of the applet.

  Java applets can be used to make Web pages more visually appealing, such as with animation. The user can view and interact with an applet; for example, making an image rotate. As the applet is running on the local system, it can provide high content graphics without the network problems normally associated with client/server graphics applications.

- JavaScript

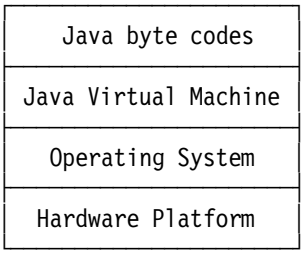
  JavaScript is an HTML extension and programming language, developed by Netscape, which is a simple object-based language compatible with Java. JavaScript programs are embedded as source directly in an HTML document. They can control the behavior of forms, buttons, and text

elements. It is used to create dynamic behavior in elements of the Web page. Also, it can be used to create forms whose fields have built-in error-checking routines.

- Java Virtual Machine

  The Java Virtual Machine (JVM) is an abstract computer that runs compiled Java programs. JVM is virtual because it is generally implemented in software on top of a real hardware platform and operating system. In this way, it is architecture neutral and platform independent. All Java programs must be compiled to run in a JVM.

  The following diagram describes in simple terms how Java is implemented:



- Just In Time Compliers

  A Just In Time (JIT) compiler compiles a Java byte code program into the native operating system and hardware instructions. This improves the performance of the program at runtime. This compilation is done at runtime to preserve the portable nature of the byte code instruction set.

- HotJava

  HotJava is a Java-enabled Web browser, developed by Sun Microsystems, which lets you view Java applets.

- JavaOS

  JavaOS is a highly compact operating system, developed by JavaSoft, which is designed to run Java applications directly on microprocessors in anything from personal computers to pagers. JavaOS will run equally well on a network computer, a PDA, a printer, a game machine, or countless other devices that require a very compact OS and the ability to run Java applications.

- Java Beans

  An initiative called Java Beans is brewing a similar set of APIs that will make it easy to create Java applications from reusable components. Java Beans will be used in a wide range of applications, from simple widgets to full-scale, mission-critical applications. Many software vendors, including IBM, have announced plans to support it.

For a detailed discussion of Java programming for VM/ESA, see the ITSO redbook *VM/ESA Network Computing with Java and NetRexx*, SG24-5148. For further information about Java, visit the following Web sites:

```
http://java.sun.com/
http://www.ibm.com/java/
http://ncc.hursley.ibm.com/javainfo/
```

1.8 Application Programming Interfaces

The Web is an interactive medium that potentially allows users to give feedback to a company about its products, use search utilities to locate information on a topic, use conversion programs to convert one value to another, and more. They are performed by external programs using information passed to them by the server.

There are two application programming interfaces available today for extending Web servers with external programs:

- Common Gateway Interface (CGI)
- Server-Side Include (SSI)

1.8.1 Common Gateway Interface (CGI)

The CGI allows the server and an external program to communicate. It is a standard interface, supported by almost all Web servers, that defines how information is exchanged between a Web server and an external program (CGI program). CGI programs can be written in any language supported by the operating system on which the server is run. The language can be a programming language, like C++, or it can be a scripting language, such as Perl (an interpretive language that started out on UNIX) or REXX. Most large system users will be more familiar with REXX and CMS Pipelines.

Programs written in programming languages need to be compiled, and typically run faster than uncompiled programs. On the other hand, those written in scripting languages tend to be easier to write, maintain, and debug.

A detailed description of CGI programming techniques for the VM/ESA Web serving environments may be found in the ITSO redbook *Web-Enabling VM Resources*, SG24-5347 (which will be available at a later date). For more general information about CGI, see `http://hoohoo.ncsa.uiuc.edu/cgi/`

1.8.2 Server-Side Include

The concept of Server-Side Include (SSI) is still relatively new and still in the process of evolving. The basic idea of SSI is to enable the server to do some processing of the Web page before the page is sent to the client. The function provided is specific to the server that is running and not to the client, as happens with normal HTML tags.

The following are some of the common functions provided by SSI:

- Displaying information about the current HTML document, such as the date last updated.
- Providing the current local date and time.
- Including text from another document, which can also contain HTML or pure text.

Different servers support SSI in different ways, but the most common approach is to use a special tag in the HTML script. HTML pages that have these special tags have to be processed in a different way by the server, and so they are often given a different file extension, usually .htmls or .shtml.

Otherwise, the server has to assume all pages could possibly contain SSI tags, requiring it to parse every file looking for these special tags and process them accordingly before sending the page to the client. This could cause unnecessary server overhead for those files that do not contain SSI tags.

For security reasons, you must be very careful which directories you enable for Server Side Includes.

For more information about SSI, see:

- http://hoohoo.ncsa.uiuc.edu/docs/tutorials/includes.html
- http://www.Apache.Org/docs/mods/mod_includes.html
- http://sw.cse.bris.ac.uk/WebTools/fakessi.html
- http://carleton.ca/~ dmcfet/html/ssi3.html
- standard reference HTML reference books

1.8.3 How to Choose a Programming Interface

Choosing a programming interface depends on your priorities. Different programming interfaces have different characteristics. For example:

- Flexibility

  CGI has great flexibility to develop applications. On the other hand, SSI has the lowest flexibility because it is a substitution mechanism; that is, SSI script is inserted into a document at commented tag locations. For programming with CGI, you can use many languages that are supported by your operating system.

- Development Cost

  SSI has the least development cost, but it does not have much flexibility. CGI is widely known and many tools exist.

- Operational Risk

  If your CGI/SSI script program ABENDs, the Web server must not be brought down by the error.

- Running Cost

  CGI and SSI are low performance because the CPU and memory overheads are high. As a result, hardware costs are likely to be higher. If transactions using CGI/SSI increase or CGI/SSI programs become larger, you may have to upgrade your hardware or distribute your function.

1.9 TCP/IP

In this section we introduce TCP/IP. For installation notes for the VM/ESA platform, see Appendix A, “TCP/IP Configuration Notes” on page 187. For more information, refer to the redbook *TCP/IP Tutorial and Technical Overview*, GG24-3376, published by Prentice Hall as ISBN 0-13-460858-5.

TCP/IP is the acronym for Transmission Control Protocol and Internet Protocol. TCP and IP are the two major protocols in a family of related protocols that is generally called the TCP/IP protocol suite. Because TCP/IP protocols are widely used in the Internet, an alternative name is the Internet protocol suite.

The Internet protocol suite allows Internet-connected hosts to engage in such activities as sharing files with other hosts, logging in to other hosts and many other forms of cooperative processing.

TCP/IP was developed as the result of a project funded by the Defense Advanced Research Projects Agency (DARPA) in the USA. The requirement was for a set of open networking protocols which would foster communications between affiliated defense, research, and academic organizations. During the 1970s, the set of protocols, originally known as Network Control Program (NCP), evolved so that by the early 1980s, the TCP/IP-based Internet as we know it today was in existence.

What has changed, particularly in the 1990s, is that the Internet is no longer restricted to the research and defense industries but has been increasingly used for general commercial purposes.

TCP/IP is a communication protocol, which, like SNA, is based on layers. It permits heterogeneous computers to communicate by performing network-related processing such as message routing, network control, error detection, and correction.

1.9.1 TCP/IP Operation

TCP/IP is built on connectionless technology. Information is transferred as a sequence of datagrams. A datagram is a collection of data that is sent as a single message. Each of these datagrams is sent through the network individually. The datagrams are sent in transit; the network does not know that there is any connection between them. It is possible that datagram 10 will arrive before datagram 5. It is also possible that somewhere in the network, an error will occur and a datagram will not get through at all. In that case, that datagram must be sent again.

An IP datagram can cross different physical networks. Physical networks have a maximum frame size, called the Maximum Transmission Unit (MTU). MTU limits the length of a datagram that can be placed in one physical frame. IP requires that each link has an MTU of at least 68 bytes, so if any network provides a lower value than this, fragmentation and reassembly must be implemented in the network interface layer in a way that is transparent to IP.

The terms *datagram* and *packet* are often used interchangeably. To be precise, a datagram is the unit of data transfer in a TCP/IP context. A packet is the unit of data transfer in a network layer context.

TCP is responsible for breaking up the message into datagrams, reassembling them at the other end, resending anything that gets lost, and putting things back in the right order.

IP is responsible for routing individual datagrams. TCP passes IP a datagram with a destination. IP identifies in its routing table the first host in the path to the destination host and sends the datagram. This process is then repeated by IP in the first host and all subsequent hosts until the datagram reaches its destination.

1.9.2 Request for Comments (RFC)

Internet standards are called RFCs or Requests for Comments. A proposed standard is initially issued as a proposal and has an RFC number. When it is finally accepted, it is added to official Internet protocols but is still referred to by its RFC number. We found the database that contains the RFCs at:

```
http://www.rfc-editor.org/
```

1.9.3 TCP/IP Layers

In the following sections, the four TCP/IP layers are introduced, and then each is discussed in its own section.

- **Application Layer**

  The application layer is provided by the program that uses TCP/IP for communication. Examples of applications are FTP, Telnet, e-mail, Gopher, and SMTP.

- **Transport Layer**

  The transport layer provides communication between application programs. The applications may be on the same host or on different hosts. Multiple applications can be supported simultaneously. The transport layer is responsible for providing a reliable exchange of information. The main transport layer protocol is TCP. Another is User Datagram Protocol (UDP), which provides a connectionless service in comparison to TCP, which provides a connection-oriented service.

- **Internet Layer**

  The Internet layer provides communication between computers. Part of communicating messages between computers is a routing function that ensures that messages will be correctly delivered to their destination. The Internet Protocol (IP) provides this routing function. Examples of Internet layers are IP, ICMP, IGMP, ARP, and RARP.

- **Network Layer**

  The network layer is implemented by the physical network that connects the computers. Examples are X.25, ISDN, Ethernet, token ring, and ATM.

1.9.4 Application Layer

In this section we describe some of the most commonly used TCP/IP applications.

File Transfer Protocol (FTP)

Files may be transferred between hosts using the file transfer protocol (FTP). FTP is designed under the client/server model: if both client and server components of FTP are provided on a host, then file transfer in both directions is possible. Security is provided by prompting for a user ID and password.

FTP is built on the services of TCP in the transport layer. FTP transfers files as either ASCII characters or binary data. ASCII characters are used to transfer data that contains only text characters. FTP provides functions, such as listing remote directories, changing the current remote directory, creating and removing remote directories, and transferring one or more files in a single request. For more information about FTP, see RFC 959.

Telnet

Telnet allows a user to log into other computers on the network. By specifying a computer on the network, a connection will be established. Once the connection is done, anything you type is sent to the computer. The connection to the remote computer behaves much like a dial-up connection. For example, the remote computer asks for a user ID and password as would be done with direct access to the computer. There are Telnet implementations for 3270 and 5250 emulation available in the IBM TCP/IP products.

Telnet is built on the services of TCP in the transport layer. Telnet provides duplex communication and sends data either as ASCII characters or binary data. For more information about the Telnet protocol, see RFCs 854, 856, 857, 885, and 1091.

Network File System (NFS)

The network file system (NFS) allows a system to access files on another computer as if they were on your own computer. NFS uses the Remote Procedure Call (RPC) protocol to communicate between the client and the server. The files to be accessed reside on the server host and are made available to the user on the client host. NFS supports a hierarchical file structure. The directory and subdirectory structure can be different for individual client systems. For more information about NFS, see RFC 1094.

Remote Printing (LPR and LPD)

The line printer requester (LPR) allows access to printers on other computers running the line printer daemon (LPD) as though they were on your computer. The clients provided (LPR, LPQ, LPRM, or LPRMON) allow the user to send files or redirect printer output to a remote host running a remote print server (LPD). These clients can also be used to query the status of a job, and to delegate a job. For more information about remote printing, see RFC 1179.

Remote Execution (REXEC and RSH)

Remote shell (RSH) and remote execution (REXEC) are similar protocols that allow you to run programs and commands on different computers. The results are received and displayed on the local host. This can be useful for small computers to harness the power of large computers.

Simple Mail Transfer Protocol (SMTP)

The simple mail transfer protocol (SMTP) is an electronic mail protocol with both client (sender) and server (receive) functions. For more information about SMTP, see RFCs 821, 822, and 974.

Multipurpose Internet Mail Extensions (MIME)

To overcome the shortcomings of SMTP, a new architecture has been defined that allows for a much greater variety of what can be contained in an electronic message, such as:

- 8-bit text and lines longer than 1000 characters
- International code pages and character sets
- Binary and multimedia objects, such as
  - Fonts
  - Images, audio and video objects

MIME is defined in RFCs 2045 to 2049 and currently has state of draft standard. MIME does not solely apply to electronic mail, it rather defines a way to

incorporate different objects in any electronic message. For instance, it is used widely throughout the Internet today by means of browsing the World Wide Web.

Post Office Protocol (POP)

The post office protocol (POP) is an electronic mail protocol with both client (sender/receiver) and server (storage) functions. POP allows mail for multiple users to be stored in a central location until a request for delivery is made by the electronic mail program. For more information about POP, see RFC 1725.

Internet Message Access Protocol Version 4 (IMAP4)

IMAP4 is an electronic messaging protocol with both client and server functions. Similar to POP, IMAP4 servers store messages for multiple users to be retrieved upon client request, but IMAP4 clients have more capabilities in doing so than POP clients. IMAP4 allows clients to have multiple remote mailboxes to retrieve messages from and to choose any of those any time. IMAP4 clients can specify criteria for downloading messages, such as not to transfer large messages over slow links. Also, IMAP4 always keeps messages on the server and replicates copies to the clients. Transactions performed by disconnected clients are effected on server mailboxes by periodic re-synchronization of client and server. For more information on IMAP4 and its underlying electronic mail models, see RFC 2060 and RFC 1733.

Gopher

Gopher is a client/server protocol designed for information location and retrieval. The client function provides a menu-driven interface to access the files stored on a Gopher server. The server function allows descriptive names to be assigned to the files, making it easier to identify the content of each file. Gopher was designed at the University of Minnesota. For more information about Gopher, see RCF 1436.

Domain Name System (DNS)

The domain name system uses a hierarchical system for naming hosts. Each host name is composed of domain labels separated by periods. Each label represents an increasingly higher domain level within an Internet. The fully qualified domain name of a host connected to a large Internet generally has one or more subdomains.

For example:

```
host.subdomain.subdomain.rootdomain
```

or

```
host.subdomain.rootdomain
```

Domain names often reflect the hierarchy level used by network administrators to assign domain names. For example, the domain name eng.mit.edu is a subdomain of edu. Local network administrators have the authority to name local domains within an Internet.

You can refer to hosts in your domain by host name only; however, a name server requires a fully qualified domain name. The local resolver combines the host name with the domain name before sending the address resolution request to the domain name server.

Virtual Hosting A process by which a single real computer on the TCP/IP network is given multiple IP addresses and DNS names and responds to each as if it were a computer with only that DNS name. Thus, in the same way that VM/ESA supports running multiple user IDs as if they each owned their own separate S/390 system, virtual hosting allows a TCP/IP server to represent itself as if there are multiple S/390 systems.

Simple Network Management Protocol (SNMP)

The simple network management protocol (SNMP) provides a means for managing an Internet environment. SNMP allows network management by elements, such as gateways, routers, and hosts. Network elements act as servers and contain management agents, which perform the management functions requested. Network management stations act as clients; they run the management applications, which monitor and control the network. SNMP provides a means of communicating between these elements and stations so they can send and receive information about network resources. For more information about network management using SNMP, see RFCs 1155, 1157, 1187, and 1213.

Talk

Talk allows you to send interactive messages, as opposed to the batch mail capabilities of SMTP. When a local host sends a Talk request to a remote host, the user on the remote host is notified that there is a connection request. The user on the remote host must respond with a Talk message to the local host. Message exchange can thus occur between the local and remote hosts.

Finger

The Finger protocol provides an interface for querying the current status of a remote host or a user ID on a remote host. Finger uses TCP as the underlying protocol. For more information about Finger, see RFC 1196.

Routing Information Protocol (RIP)

The Routing Information Protocol creates and dynamically maintains network routing tables. RIP arranges to have gateways and routers periodically broadcast their routing tables to neighbors. Using this information, a RouteD server can update a host′s routing tables. For example, RouteD determines if a new route has been created, if a route is temporarily unavailable, or if a more efficient route exists. For more information about routing using RIP, see RFC 1058.

X Window System

The X Window System protocol supports network-transparent windowing and graphics. For more information about X Window System protocol, see RFC 1013.

Sockets Application Programming Interface

The sockets API allows you to write your own applications to supplement those supplied by TCP/IP. Most of these additional applications communicate using either TCP or UDP.

1.9.5 Transport Layer

This section describes the transport protocols in TCP/IP that allow communication between application programs.

Transmission Control Protocol (TCP)

The Transmission Control Protocol (TCP) provides a reliable vehicle for delivering packets between hosts on an Internet. TCP takes a stream of data, breaks it into datagrams, sends each one individually using IP and reassembles the datagrams at the destination node. If any datagrams are lost or damaged during transmission, TCP detects this and resends the missing datagrams. The received data stream is a reliable copy of the transmitted data stream. For more information, see RFC 793.

User Datagram Protocol (UDP)

The user datagram protocol (UDP) provides a less reliable mode of communication than does TCP and is an alternative to TCP as a transport protocol.

Like IP, UDP does not guarantee datagram delivery or duplication protection. UDP does provide checksums for both the header and data portions of a datagram. Therefore, applications that require reliable delivery of streams of data should use TCP. For more information about UDP, see RFC 768.

1.9.6 Internet Layer

This section describes the Internet protocols in TCP/IP. Protocols in the Internet layer provide connection services for TCP/IP. These protocols connect physical networks and transport protocols. RFCs 1118, 1180, 1206 and 1208 provide more information.

Internet Protocol (IP)

The Internet Protocol (IP) provides the interface from the transport layer (host-to-host) protocols to the physical level protocols. IP is the basic transport mechanism for routing IP packets to the next gateway, router, or destination host.

IP provides the means to transmit blocks of data (or packets of bits) from sources to destinations. Sources and destinations are hosts identified by fixed-length addresses. Outgoing packets automatically have an IP header sent to the higher-level protocols. This protocol provides the universal addressing of hosts in an Internet network.

IP does not ensure a reliable communication because it does not require acknowledgments from the sending host, the receiving host, or intermediate hosts. IP does not provide error control for data; it provides only a header checksum. IP does not perform retransmissions or flow control. A higher-level protocol that uses IP must implement its own reliability procedures.

For more information about IP, see RFC 791.

Internet Control Message Protocol (ICMP)

The Internet Control Message Protocol (ICMP) passes control messages between hosts, gateways, and routers. For example, ICMP messages can be sent in any of the following situations:

- When a host checks to see if another host is available (PING)
- When a packet cannot reach its destination
- When a gateway or router can direct a host to send traffic on a shorter route
- When a host requests a netmask or a time stamp
- When a gateway or router does not have the buffering capacity to forward a packet

ICMP provides feedback about problems in the communication environment but it does not make IP reliable. The use of ICMP does not guarantee that an IP packet will be delivered or that an ICMP message will be returned to the source when an IP packet is not delivered or is incorrectly delivered.

For more information about ICMP, see RFC 792.

Address Resolution Protocol (ARP)

The Address Resolution Protocol (ARP) maps Internet addresses to hardware addresses. ARP is not directly available to users or applications. When an application sends an Internet packet, IP requests the appropriate address mapping. If the mapping is not in the mapping table, an ARP broadcast packet is sent to all the hosts on the network requesting the physical hardware address for the host.

For more information about ARP, see RFC 826.

1.9.7 Network Layer

This section describes the major protocols that implement the network layer in TCP/IP. Network protocols define how data is transported over a physical network. Typically a single host interface will be defined to use one of these protocols.

Network Driver Interface Specification

Network driver interface specification (NDIS) is a medium access control (MAC) interface for local area network (LAN) adapter and protocol drivers. NDIS has become an industry standard, providing a common, open interface that enables network adapters and LAN software from different manufacturers to communicate with each other.

A network adapter driver provides the communication between a network adapter and a protocol, using NDIS as the interface. Network adapter drivers handle the basic transmission and reception of packets on the network.

A protocol driver provides the communication between an application and a network adapter driver, using NDIS as the interface. Protocol drivers provide a high level of communication between the data link layer and the application layer.

Serial Line Internet Protocol (SLIP)

The Serial Line Internet Protocol (SLIP) allows you to set up a point-to-point connection between two TCP/IP hosts over a serial line, for example, a serial cable or a RS-232 connection into a modem and over a telephone line. You can use SLIP to access a remote TCP/IP network (such as a service provider's network) from your local host or to route datagrams between two TCP/IP networks.

For more information about SLIP, see RFC 1055.

Point-to-Point Protocol (PPP)

The point-to-point protocol (PPP) allows you to set up a point-to-point connection between two TCP/IP hosts over a serial line, for example, a serial cable or a RS-232 connection into a modem and over a telephone line. You can use PPP to access a remote TCP/IP network (such as a service provider's network) from your local host to route datagrams between two TCP/IP networks.

For more information about PPP, see RFCs 1717 and 1661.

X.25

You can use an X.25 network to establish a TCP/IP connection between two hosts. X.25, recommended as a communication interface standard by the International Telegraph and Telephone Consultative Committee (CCITT), defines the interface between data terminal equipment (DTE) and data circuit-terminating equipment (DCE). A DTE is a computer or workstation connected to a network. A DCE is the equipment at the point of the connection to the network, such as a modem.

For more information about TCP/IP over X.25, see RFC 877.

1.10 Security

One of the major concerns when providing commercial services on the Internet is providing for transaction security and communications security.

Information exchanges are secure if all the following are true:

- Messages are confidential. Your message content is private and not available to others as your messages flow through the Internet. Encryption is used to ensure that the message content is confidential and no one eavesdrops on your communications.
- The information exchange has integrity. With integrity, your messages are not altered as they flow from router to router. You can be assured that the message received is the same as the message you sent. For example, financial transactions are unchanged. Encryption and digital signatures ensure integrity.
- Both sender and receiver are accountable. They both agree they took part in the exchange. For example, the receiver knows that the sender signed the contract. Digital signatures assure accountability.
- You can authenticate both parties in the exchange. Not only was the contract signed, but it was signed by the proper person. The sender knows the receiver is authentic and not someone masquerading as the receiver. Authentication ensures that someone is who they say they are.

The following security implementations are discussed:

- Secure Sockets Layer (SSL)
- Secure HyperText Transport Protocol (SHTTP)
- Secure Electronic Transactions (SET)

It is your responsibility to review the protection offered by these methods, the methods offered by the Web server you are using, and your configuration of your Web server, and to restrict the information provided by your VM/ESA Web server accordingly.

This topic is also covered extensively in

- *Web-Enabling VM Resources*, SG24-5347 (which will be available at a later date).

1.10.1 Secure Sockets Layer (SSL)

SSL is a security protocol that was developed by Netscape Communications Corporation, along with RSA Data Security, Inc. The primary goal of the SSL protocol is to provide a private channel between communicating applications that ensures privacy of data, authentication of the partners, and integrity. SSL appears at this time to be the leading security protocol on the World Wide Web. All commercial grade Web servers and graphical browsers currently offer SSL protection.

When a URL wishes to make use of the SSL protocol it replaces the “http” protocol indication with “https.” The default TCP port for https is port 443, as opposed to http′s default port of 80.

SSL provides an alternative to the standard TCP/IP socket API that has security implemented within it. Hence, in theory it is possible to run any TCP/IP application in a secure way without changing it. In practice, SSL is so far only widely implemented for HTTP connections.

The protocol is composed of two layers:

- At the lowest layer is the SSL Record protocol. It is used for data encapsulation.
- On the top of this layer is the SSL Handshake protocol, used for initial authentication and transfer of encryption keys.

The SSL protocol addresses the following security issues:

- Privacy. After the symmetric key is established in the initial handshake, the messages are encrypted using this key.
- Integrity. Messages contain a message authentication code ensuring the message integrity.
- Authentication. During the handshake, the client authenticates the server using an asymmetric or public key.

SSL requires each message to be encrypted and decrypted and therefore has a high performance and resource impact. Also, since only the server is authenticated, SSL is not suitable for applications, such as electronic banking, that require that the server authenticate their clients.

An advantage of SSL is that it is easy to implement in a WWW server environment. When you have a key pair and a signed certificate (for authentication purpose), you can begin serving SSL-protected documents to SSL browsers. You have to set up SSL security in your WWW server only once. Your HTML pages and your server configuration file require no additional statements. For more information, see the following documents:

- *Introduction to Public-Key Cryptography* at http://developer.netscape.com/docs/manuals/security/pkin/index.htm
- *Introduction to SSL* at http://developer.netscape.com/docs/manuals/security/sslin/index.htm
- *The SSL Protocol, Version 3* at http://home.netscape.com/eng/ssl3/ssl-toc.html
- http://www.vm.sterling.com/conferences/teleconf.html#ssl is the transcript of the Sterling Software teleconference on *Addressing Web Security Issues* from October 21, 1997.

1.10.2 Secure HyperText Transport Protocol (S-HTTP)

S-HTTP is a security-enhanced version of HTTP developed by Enterprise Integration Technologies (EIT). It uses a modified version of HTTP clients and servers to allow negotiation of privacy, authentication, and integrity characteristics. Secure HTTP provides transaction security at a document or even a field level. Fields, for example, with an account code or electronic signature, or entire documents may be marked as private and signed by the sender. S-HTTP has not become a widely accepted security protocol on the World Wide Web. Few commercial grade Web servers and browsers currently offer S-HTTP protection.

To use S-HTTP, the document code or server administrator specifies the security options for each document anchor or hypertext link that they want to secure. These cryptographic options, called CRYPTOPTS, allow them to specify parameters for communication such as:

- Should the document be encrypted?
- Should the request be signed, encrypted, or both?
- What about the reply?
- What cryptographic algorithm should be used?
- What certificate should be used?

CRYPTOPTS can also be specified for directories so that all the files in the directory inherit the parameters for the directory.

When a user requests a secured document, the client sends an HTTP request with its own CRYPTOPT enclosed. The negotiation is initiated. The server knows that it is to be handled by S-HTTP because the URL starts with *shttp:* instead of *http:*. The server responds with its side of the negotiation.

In this negotiation phase, the client and the server exchange messages detailing what CRYPOPT features they accept. If the server is finally satisfied that the client is authorized, it meets the request; otherwise the request fails. The negotiation is provided so that integrity, privacy, and authentication of both parties are or can be carried out in almost any combination depending on the cryptographic options.

Unlike SSL, S-HTTP requires some HTML changes to invoke it. This implies application involvement. Therefore, S-HTTP implementation is not as straightforward as SSL implementation. However, the fact that the granularity of security is at the documentation level increases flexibility and improves performance.

One major advantage of S-HTTP is its ability to perform client authentication. However, the fact that this requires the client to have a public-key certificate limits the degree to which it may be applied.

1.10.3 Secure Electronic Transactions (SET)

SET is designed for securing credit card authorizations and other forms of electronic payments and e-commerce. It is a joint venture of several companies, including VISA, MasterCard, Microsoft, Netscape, SAIC, GTE, Terisa Systems, Verisign, and IBM. The protocol is fairly complex, and involves multiple third parties to perform its validations. This protocol does not yet have wide-spread use, and has limited applicability to the environments found in most VM/ESA-based shops.

1.11 Firewalls

When you connect your network to the Internet, you give your internal users the possibility to access this tremendous world, but you also offer the opportunity to external users to reach your private network. Potentially, anyone can connect to anyone, that is, any client can communicate with any server. In this way, you can expose your network to *attacks*.

This any-to-any connectivity can give you many security problems. You will need to protect your own private data and also protect access to the machines inside your private network against abusive use from outside.

The first step to achieving this is to limit the number of points at which your network is connected to the Internet. The best configuration has a single point of connection, as shown in Figure 14 on page 46. If you have a unique path, you gain control over which traffic to allow into and out of the Internet. The gateway that gives you this control is called a *firewall*.

Firewalls tend to be seen as a protection between the Internet and a private network. But generally speaking, a firewall should be considered a means to divide the world into two or more networks: one or more secure networks and one or more non-secure networks.

Imagine a company where all the departments are connected to the internal network, including sales, accounts, development, and human resources departments. The administrator wants to be able to restrict access between the development department machines to the human resources department machines and from the sales department to the development department.

In the following discussion, for ease of comprehension, we equate a non-secure network to the Internet or public network and a secure network to a private IP network.

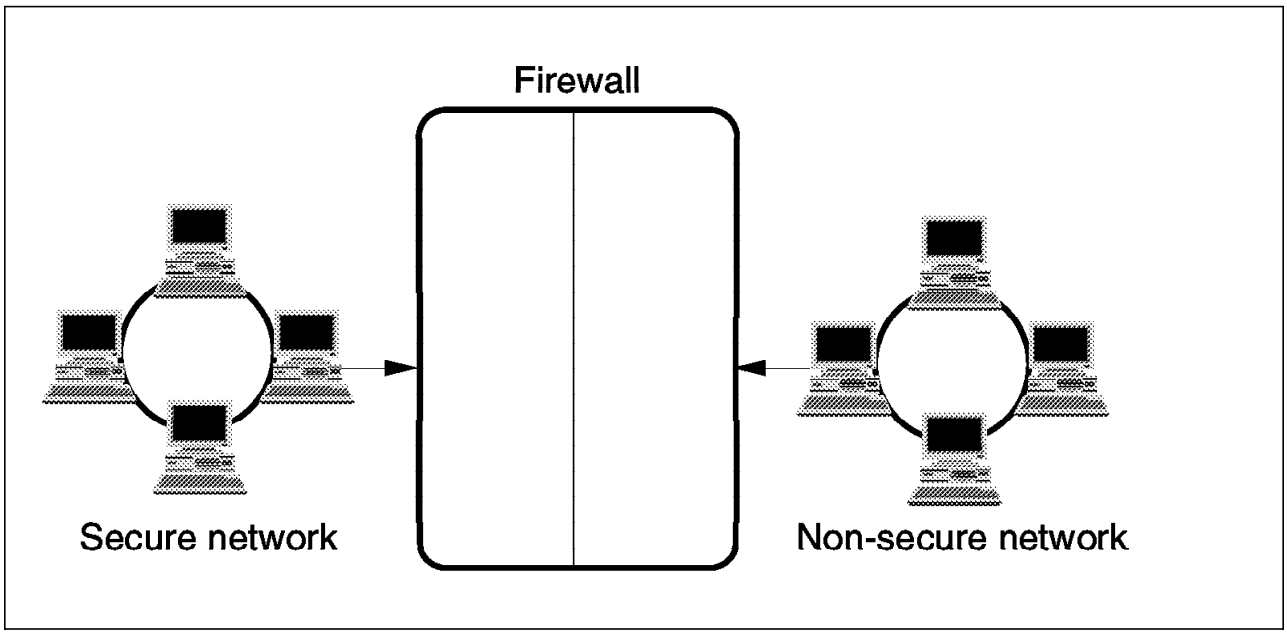


Figure 14. A Single Internet Connection

Firewalls differ in their implementation and in the degree of protection that they offer. The most common types of firewalls are:

- Screening Filter: This uses a router to connect the private network to the Internet. The screening filter looks at each IP packet flowing through it, controlling access to machines and ports in the private network and possibly limiting access from the private network to the Internet. Screening filters operate at the Internet Protocol (IP) layer and cannot control access at the application layer.
- Bastion: This is a machine placed between the private network and the Internet, which breaks the connection between the two. The bastion relays messages to or from authorized users and denies access to all others. Bastions can control access at the user or application layer, but can be costly if many users are supported.
- Dual-Homed Gateway: This combines a screening filter and a bastion into either a single machine or a series of machines. Combining a bastion and a screening filter in a single machine protects the private network from general access while making some bastion applications accessible.

The IBM Internet Connection Secured Network Gateway (SNG) supports all of these implementations. It also includes many other technologies to help in implementing security in an Internet environment.

1.11.1 Proxy Servers

Proxy servers are implementations of bastions. They control access to or from the private network, relaying only acceptable communications from known users.

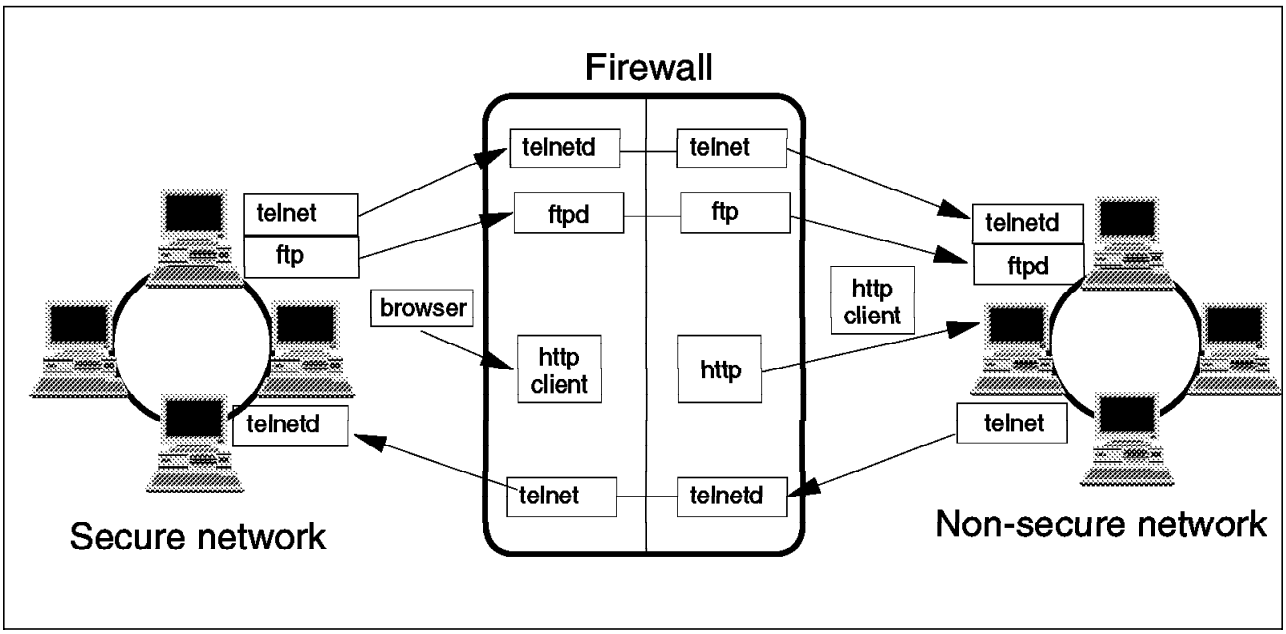


Figure 15. Firewall with Proxy Servers

Users in the private network can access an application, such as FTP, in the proxy server using their usual utilities (clients). Users authenticate themselves to the proxy server and can then access the application on the desired machine in the public network. Proxy servers can also be used from the public network to access applications in the private network, but this exposes login names and passwords to attackers in the public network. The proxy services supplied with SNG include Telnet and FTP.

The Internet Connection servers can act as proxy servers. The internal users ask the server running on the firewall to retrieve documents from external servers.

1.11.2 SOCKS Servers

SOCKS servers are like proxy servers without the requirement for double connections. With SOCKS, users can benefit from secure communications without needing to be aware that it is happening.

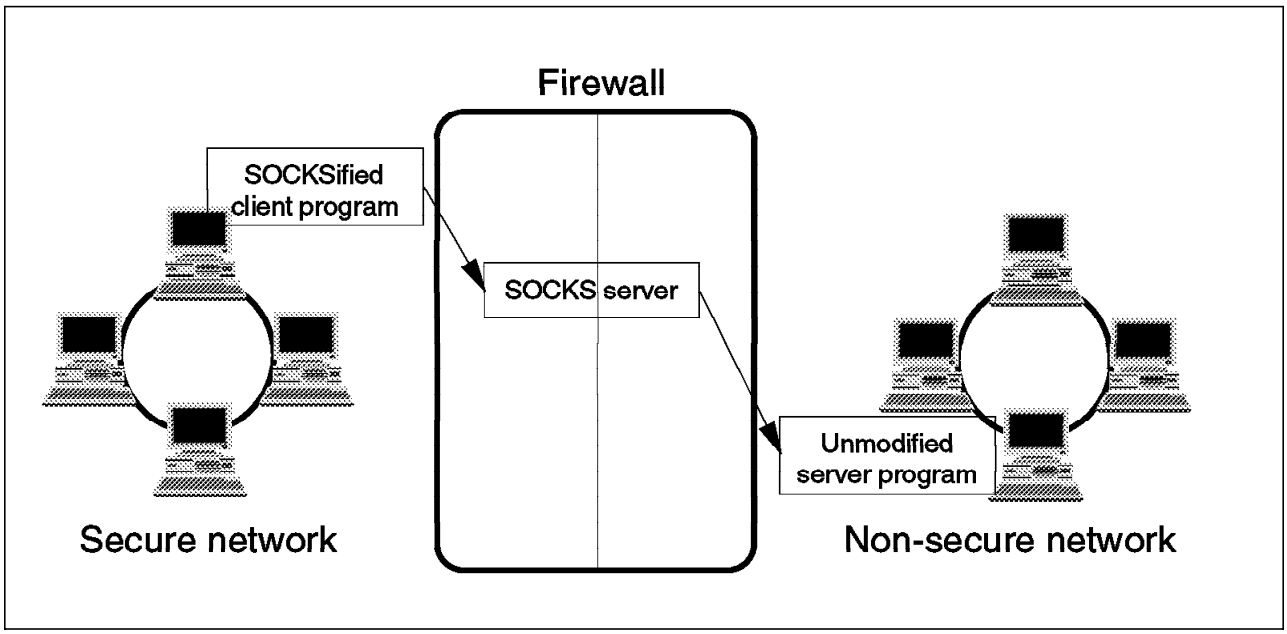


Figure 16. Firewall with SOCKS Server

Users have to use new versions of the client called SOCKSified clients. The SOCKSified client code directs its requests to the SOCKS port on the firewall. Sessions are broken at the firewall, as they are with proxy servers. With SOCKS, however, the connection to the destination application is created automatically once the user is validated.

Both the client and the SOCKS server need to have SOCKS code. The SOCKS server acts as an application-level router between the client and the real application server. SOCKS is for outbound sessions only. It is simpler for the private network user, but does not have secure password delivery, so it is not intended for sessions between public network users and private network applications.

Most browsers are SOCKsified and you can get SOCKSified TCP/IP stacks for most platforms. For additional information, refer to:

- `http://www.raleigh.ibm.com/sng/sng-socks.html`
- `http://www.socks.nec.com`

Chapter 2. VM/ESA and the Web

The technology that provides the single highest return on your IS investment today is end-user access to VM/ESA data through a Web server.

This chapter outlines this concept, and the following chapters provide the supporting information that will help you evaluate and choose the best VM Web server solution for your enterprise.

2.1 The Universal Client

If all of today's computing technology were distilled into a couple of words, they would be "access to data."

Perhaps one of the most ambitious efforts in the last decade has been to establish a *true* client/server infrastructure that has the promise of allowing global access of data to a universal client. While true client/server technologies are emerging, it is often a long and expensive transition to convert legacy applications to best use a three-tiered computing model.

Large and small industries rely on VM/ESA to provide their large scale computing needs. New technologies, such as DCE and VisualAge Generator, have been introduced to provide a migration path for customers wishing to implement a client/server topology.

As customers become comfortable with using workstation applications to access enterprise data, data acquisition becomes fragmented over a variety of services, CPUs, and workstations.

The World Wide Web was born out of the desire to provide data in a configurable, dynamic way. The browser has become the universal client. A Web browser is available for almost every hardware and software platform. An industry battle exists as every major vendor tries to make the better browser.

2.2 VM/ESA As a Web Server

The technology required for a Web server is rather basic compared to that of a browser. Data is stored as EBCDIC or binary files, and served as requests are received. Special programs, using *Common Gateway Interfaces*, access databases and perform computational tasks for the browser. The most logical place for a CGI to run is on the same system on which the data resides.

Recently, organizations such as Princeton University, the U.S. Food and Drug Administration, Pacific Bell, 3M, and Nationwide Insurance, to name a few, have realized the cost savings of placing a Web server on VM/ESA.

For many, the advantage of using their enterprise server to perform Web services goes beyond cost. The enterprise server has better security, stable connections, and an existing infrastructure for backups and a help desk. In addition, the raw I/O performance of an ES/9000 has caused a re-evaluation of where a Web server should go when serving large amounts of interactive dynamic data.

With a native Web server, VM data becomes available to a huge audience of data hungry Web surfers. Instantly gone is the drab 3270 interface, and through a Web server and simple Hypertext Markup Language (HTML), VM is given a bright new look that integrates into any desktop.

Maintenance of HTML is minimal. The data resides in flat files stored on conventional CMS disks. Browsers are able to capture HTML from other servers to use as prototypes. Interesting new presentation concepts can be quickly applied to give an old application a bold new look.

For almost no investment, a Web server can be installed on VM, and access to panel-driven, keyboard-navigated applications can be replaced with a colorful point and click, animated Web page. You can add graphics, sound, or even video. Or, you can simply allow the server to present an organized hierarchical view of your data.

But keep in mind that a flashy front end to an application is not the key point. Providing data in a reliable, low cost, secure way is the most important aspect of IS. Your end users are already accessing volumes of data from the Web. Integrate existing VM/ESA services into the Web, then build new services that leverage the advanced function of VM/ESA. Go beyond the 3270 interface and connect VM to the universal client, the Web browser.

2.3 Web Servers for VM/ESA

The number of Web servers available for VM/ESA has grown from one to three. The following list briefly describes each server offering. For a summary of function, there is a cross-reference table located in Chapter 3, “VM/ESA Web Server Feature Summary” on page 53.

Webshare

Webshare is a no-cost Web server written in CMS Pipelines and REXX. It gives VM the ability to serve text, graphics, sound, and video on the Web. No additional security is included in this product. The data you provide should be restricted to information you wish to serve to all of your computing community. Corporate announcements, users′ home pages, help desk news, access to forums, and public domain documentation are some of the items suited for this server. See Chapter 4, “Webshare” on page 57 for a detailed discussion.

EnterpriseWeb

EnterpriseWeb from Beyond Software Inc. is built on a Webshare base. The original code was modified to use CMS Pipelines dispatcher to interleave requests with I/O operations. This Web server is best suited for traditional VM/ESA programmers who can see how things work or how a feature is implemented through the optional source code, or one-lined EXECs.

Security is provided through authorization records in control files. Users can be prompted for their user ID and password for authentication. The exploitation of the SFS is also improved. See Chapter 5, “EnterpriseWeb/VM” on page 79 for a detailed discussion.

VM:Webgateway

VM:Webgateway from Sterling Software, Inc. is built upon an integrated assembler nucleus using technology ported from existing

Sterling Software, Inc. applications. Webshare control files can be ported to VM:Webgateway control files, but the two are not interchangeable.

Sterling Software, Inc.′s Automated Installation and Maintenance (AIM) process will appeal to customers that already own Sterling Software, Inc. products. The documentation for this server, however, is the most robust. Sterling Software, Inc. has a long and outstanding reputation as a commercial vendor of VM products. See Chapter 6, “VM:Webgateway” on page 109 for a detailed discussion.

When choosing a Web server for VM, the best place to start is with Webshare. This no-cost shareware program is easily downloadable (see section 4.2, “Obtaining Webshare” on page 59) and gives you a running Web machine in one afternoon. You can evaluate VM/ESA as a server using this product. Many companies are providing production services using Webshare.

Each of the vendors offers an evaluation program. Both commercial vendors offer competitive pricing and have an aggressive schedule to introduce new function. Both offer increased security, as well as improved support of the CMS Shared File System.

Long-running CGIs impact throughput in all the products. They can block the server from running new requests since they typically do not cooperate well in a multitasking environment. One response to this is to create additional service machines that share the same TCP/IP port, and have common access to data. In this scenario, the VM:Webgateway from Sterling Software, Inc. (prior to Release 1.2) requires additional administration since it records configuration data in a R/W disk. Each VM:Webgateway user ID would need to be updated to provide the same configuration. See section 6.8.8, “VM:Webgateway Dynamic Worker Machines” on page 140 for a look at how Sterling Software, Inc. has solved this challenge.

VM:Webgateway has the most complete interface for administration over the Net. You can administer most major functions from a Web browser. Most actions that require a recycle of Webshare-based servers do not require a reboot in VM:Webgateway.

Taking into consideration all of the planned new function and various existing differences of each of the servers, it is too early to pick a favorite. All three products have their advantages. The following chapters provide additional information that will assist you in leveraging your VM/ESA investment using a Web server, and introduce you to each of the servers currently available.

Chapter 3. VM/ESA Web Server Feature Summary

Table 1 contains a summary of the features supported by the current VM/ESA Web servers. It is offered as a guideline. The vendors that offer these products may change their implementation at any time, and thus obsolete this table.

Table 1 (Page 1 of 4). Key Server Features

Feature	Webshare	EnterpriseWeb	VM:Webgateway
Architectural Foundation			
Designed for high-volume transaction processing		√	√
Nucleus	REXX & CMS Pipelines	REXX & CMS Pipelines	Assembler & compiled REXX
Exploits multitasking to support concurrent users on a single server		Through multistream pipes	√
Provides multiprocess TCP/IP sockets to maximize information delivery			√
Supports multiple Web servers per TCP/IP port	√	√	√
Provides dynamic worker/slave virtual machines			√
Supports Web enabling linemode and 3270 applications running on end user userids			√
Serves different directory roots for different server IP addresses	√	√	√
Full, integrated support for “virtual hosting”		Through use of URLEXIT	√
Server-Side Information and Storage			
Stores and serves home pages written in HTML	√	√	√
Stores and serves multimedia files such as graphics, image, audio, and video	√	√	√
Stores and serves JAVA applets	√	√	√
Stores and serves user-defined data (MIME) types	√	√	√
Acts as a repository for downloadable files	√	√	√
Serves user home directories	√	√	√
Creates automatic directory tree when URL specifies a directory without a specific file	√	√	√
Serves default directory root file when URL specifies a directory without a specific file	√	√	√

Table 1 (Page 2 of 4). Key Server Features			
Feature	**Webshare**	**EnterpriseWeb**	**VM:Webgateway**
Supports hierarchical data structures on minidisk and SFS	√	√	√
Full support for BFS resident data and programs			√
Logging			
Records and audits all connections and information transactions using CERN/NCSA common log format		√	√
Allows customization of normal (hit) log entries and adds administrator defined log entry fields		√	Planned
Can write to multiple logs (such as for certain kinds of hits, or different logs with different record formats)		√	Planned
Server can generate non-hit log entries (such as time events and threshold-reached information)		√	Planned
Enables CGI programmers to create their own log entries	√	√	√
Automatically archives or cycles log files		√	Indirectly
Protocol Support, Actions, and Includes			
Shows server product name in response header	√	√	√
Automatically responds to *If Modified Since* requests		√	√
Automatically includes HTTP headers for HTML documents		√	√
Automatically includes HTTP headers for non-HTML documents		√	√
Script or action based on output file type		√	√
Script or action based on output MIME type		√	√
Custom icons based on MIME type		√	√
Input filters (content modification)		√	
Output filters (content modification)		√	√
Pre- and post-processing exits (no content modification)		√	Upon Request
Recognizes migrated SFS files and carries out special actions		Proposed	√
Supports server PUT method to store new and updated information on the server		√	Planned

Table 1 (Page 3 of 4). Key Server Features

Feature	Webshare	EnterpriseWeb	VM:Webgateway
Supports server PUSH method to enable transmission of dynamic (animated) documents or images		√	√
Recognizes non-supported methods and invokes alternate site defined script		√	Planned
Allows administrator to select documents based on User-Agent header		√	Using a CGI or FILTER
Allows administrator to change server action based on User-Agent header		√	Using a CGI or FILTER
Allows HTML authors to specify server-side include references for other documents to be served along with the source document		√	√
Supports clickable imagemaps		√	√
Supports imagemaps that use rectangles, circles, polygons, and points	Rectangles Only	√	√
CGI Scripts			
Supports site written Common Gateway Interfaces (CGIs) scripts to create dynamic HTML documents and to interface to existing applications	√	√	√
Supports CGI scripts written in REXX and CMS Pipelines	√	√	√
Supports calls to other languages from REXX stub	√	√	√
Enables CGI programmers to access server state variables	√	√	√
Provides padded cell CGI environment			Using worker environment
Security and Access Control			
Can require password to authorize a user		√	√
Access to data hierarchies based on host user ID/password verification		√	√
Supports secure sockets interface (SSL)		√	√
Supports optionally fetching SSL client certificates		configuration of never or required	√
Prohibits by domain name		√	√
Prohibits by IP address		√	√
Prohibits by user's status as a system administrator or operator			√
Prohibits by HTTP Referer header information			√

Table 1 (Page 4 of 4). Key Server Features

Feature	Webshare	EnterpriseWeb	VM:Webgateway
Access control controlled by SSL client certificate fields		Using a CGI	√
Hierarchical permissions for directory-based documents		√	√
Unique access control on individual files in a directory			√
Security rules can be based on URLs		√	√
Default security model for file-based documents		√	√
Interfaces to VM/ESA security packages (ACF/2, RACF, VM:Secure)		√	√
Security control exits		√	√
Configurable user groups (not just a single user list)		√	√
Dynamic change to user access control list		√	√
ACI group rules support		√	√
Provides padded cell CGI environment under data owner's authorities			Using worker environment
Provides padded cell application environment under client's authorities			Using VM:Webserver CGI Extension
Documentation			
Online documentation in HTML format		Limited	√
Setup and Administration			
Provides upward compatibility for Webshare configuration and CGI scripts	Not Applicable	√	√
Provides browser-based (GUI) product setup, configuration, and maintenance		Limited	√
Provides browser-based (GUI) Application/MIME configuration		Limited	√
Provides browser-based (GUI) site content management		Limited	√
Supports standard PC based HTML content publishing tools		Limited	√
Remote maintenance		√	√
Server is dynamically reconfigurable		Limited	√
Customer Support			
World-wide service hotline	Not Applicable	√	√

Chapter 4. Webshare

This chapter discusses Webshare, originally written by Rick Troth, now owned and distributed by Beyond Software Inc.. Webshare is a VM/CMS Hyper-Text Transmission Protocol (HTTP) server that is downloadable from the Internet. You can use it in its unmodified form on your existing VM system to provide Web services for the Internet and your intranet. It is now being used by over forty VM/ESA servers on the Internet, and many past users of this software have moved up to a commercial offering. See the following URL for a list of Webshare systems:

`http://www.beyond-software.com/Related/WebshareServers.html`

4.1 Introduction

Until 1996, Webshare was the only Internet server for the VM platform. It is completely independent from the HTTP implementations on other platforms because it is not written in C or C++. The entire server is written in REXX, REXX/Sockets, and CMS Pipelines, all native languages to VM/ESA. (The EnterpriseWeb/VM from Beyond Software is the commercial version of this server and is also based on REXX and CMS Pipelines.) The fact that you receive this server at no cost does not mean that you cannot support a full production or business critical workload. It provides many useful features, which are discussed in the following sections.

4.1.1 Why Use Webshare?

Many arguments support the use of Webshare on your existing VM/ESA platform, such as:

- You want to offer VM/ESA data as HTML pages to your intranet clients.
- You want to have a presence on the Internet and understand the security issues involved with this.
- You have no budget to contract a commercial program, and are willing to accept the legal terms outlined in the files CMSHTTPD COPYRIGHT and WEBSHARE LICENSE, which are distributed by Beyond Software Inc. with the package.
- You have an investment in VM, and want to leverage that investment. This investment contains, to name just a few:
  - Integrated backup
  - Existing change control
  - World class security and auditing with RACF or VM:Secure, if installed
- You want to evaluate VM/ESA's overall performance as a Web server.
- You want to gain experience before making a capital investment in one of the commercially available VM/ESA servers.

4.1.2 Webshare Features

As you see in Figure 17, there are many features provided in Webshare. These features are briefly described in the list that follows.

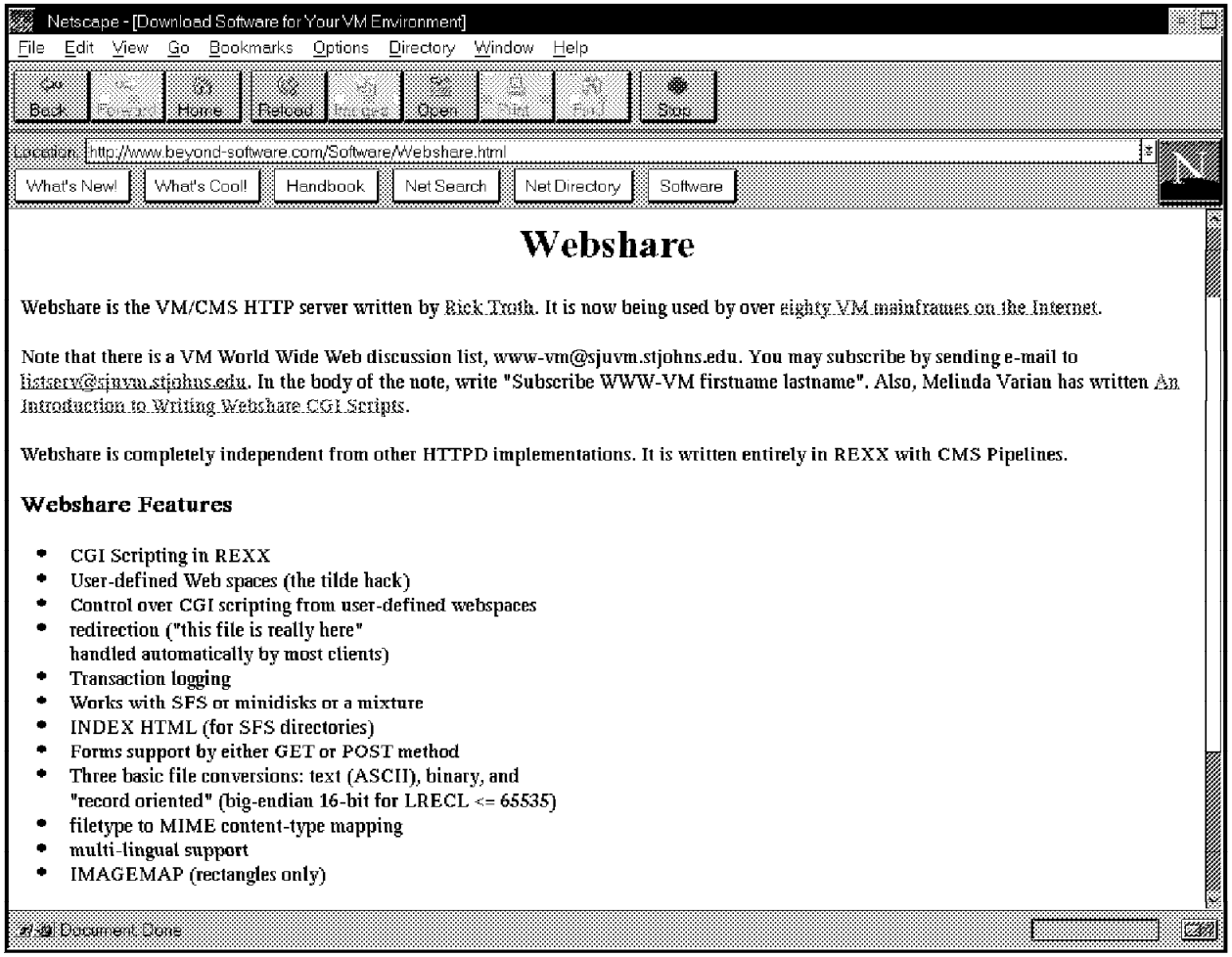


Figure 17. Internet Page Describing the Webshare Features

- CGI Scripting in REXX

  It allows you to write your own programs to connect the Web clients with VM/ESA data. The language is REXX and CMS Pipelines. Both are provided by the VM/ESA system.

- User-defined Web spaces

  Webshare supports user-defined Web spaces. Users can share their own CGI programs and HTML files. They are stored on users' data space and provided by the server to the browsers. You can access this space by the using the URL form: `http://hostname(:port)/~user-id/path(/file-name)`

- Control over CGI scripting from user-defined Web spaces

  In your configuration file, you can select which users can provide CGI scripts.

- Redirection

  If the client requests a file and the file is not at the location, it is rerouted to the right location. This mechanism works only in a very simple form on this

server. You will be redirected to the next higher directory automatically, or an administrator must modify control files.

- Transaction logging

  All transactions can be logged. This provides an audit trail of which IP addresses accessed your objects.

- Works with SFS, or minidisk, or a mixture of them

  You have the choice to provide your user data on SFS, minidisks, or on any combination.

- INDEX HTML

  The HTML content in INDEX HTML is presented as the default homepage for the server.

- HTML forms support either the GET or POST method

  The two methods to transfer data to a CGI.

- Three basic conversions

  There are text-, binary-, and record-oriented conversions.

- File type to MIME content type mapping

  Webshare knows how to handle files by their file types. It determines if it should handle files as ASCII, EBCDIC, or CMS files.

- Multi-lingual support

- IMAGEMAP

  Clicking hot spots of an image can cause an action. Webshare only supports rectangular areas within an image to be selectable.

4.1.3 About the Author

The author, Rick Troth, is working for BMC Software Inc. in Houston, Texas, USA. Before that he was at Rice University Houston, Texas, USA. He is very skilled in C, REXX, TCP/IP, UNIX, and VM/CMS. For a better look at Rick Troth's background see `http://ualvm.ua.edu/~troth/trothsig.html`.

4.2 Obtaining Webshare

You can obtain Webshare from the Internet. Use a Web browser (for example Web Explorer or Netscape Navigator) to enter the *INTERNET SOFTWARE for VM* page, provided by Beyond Software Inc., using the following URL:

```
http://www.beyond-software.com/Software/Software.html
```

There are also freeware products you can obtain on this page, namely:

REXX Sockets An interface from REXX to TCP/IP for VM, from Arthur Ecock of City University of New York. It is required for Webshare.

GOPHER A VM-based version of the popular Internet server, from Rick Troth.

Charlotte A VM/ESA text Web browser, from Carl Forde.

Albert A VM/ESA text Web browser, from David Nessl.

4.3 Getting Started

Use the following directions to download these files from the Web and then load them to CMS:

- VMARC MODULE
- CMSHTTPD VMARC

You first must load the VMARC MODULE to be able to unpack the archived server file. With your Web browser, choose the *Internet Software for VM* page. It is provided by the Beyond Software Corporation at URL:

```
http://www.beyond-software.com/Software/Software.html
```

First load the VMARC MODULE to your PC or workstation by selecting the highlighted word **VMARC** on your browser. The name of the module you receive is VMARC MODULE. It will need 17 KB space on your hard disk. If your PC or workstation does not support long named files, you may receive it under any other name fitting the naming convention of your system. When you load it up in binary form to your VM/ESA CMS minidisk, the name must be VMARC MODULE. Table 2 shows the workstation/host naming relationship.

Table 2. Example for Renaming Files

PC/Workstation File name	VM/ESA CMS File Name
VMARC.MOD	VMARC MODULE A

After you restore the file to CMS minidisk, you must use the following command to restore it as a usable file:

```
PIPE < VMARC MODULE A | DEBLOCK CMS | > VMARC MODULE A
```

Receive the Webshare code in the same way you received the VMARC MODULE. The size of the CMSHTTPD VMARC file is 99 KB. Restore it in binary form to your CMS minidisk and use following commands to deblock the file:

```
PIPE < CMSHTTPD VMARC A | FBLOCK 80 00 | > CMSHTTPD VMARC A F 80
```

If you need help on how to use VMARC MODULE, just type VMARC without any parameters, as shown in Figure 18 on page 61.

```
Ready; T=0.04/0.05 10:51:02
vmarc
No function specified.
VMARC Version 1.2.18
Usage:
   VMARC  function  fileid(s)  (  options...
Functions:
   PACK   infileid  <outfileid>
   UNPK  <infileid> <outfileid>
   LIST   infileid
Options:
   Lzw | S2 | STore    TRace | NOTrace
   REPlace | NOReplac  OLDDate | NEWDate
   APPend | NOAppend   BLock nnnn
   TAPn                PUnch | PRint
   REader              FIfo | LIfo
Ready(00008); T=0.01/0.02 13:13:43
```

Figure 18. Getting VMARC Online Help

4.3.1 Obtaining Documentation

At this step, you do not have any documentation. Even though there are no installation instructions provided, you should extract two files from your CMSHTTPD VMARC archive to give you an idea of how to install the server and how it works. Use the command:

```
VMARC UNPK CMSHTTPD VMARC A CMSHTTPD README A
```

to get the CMSHTTPD README file and

```
VMARC UNPK CMSHTTPD VMARC A HTTPD DIRECT A
```

to get the HTTPD DIRECT file.

The CMSHTTPD README file contains a (very) short description how the server works, and about the security provided by the server. It will help you to decide if you want to use a CMS minidisk as database for the server or the Shared File System (SFS). The HTTPD DIRECT file contains an example of a CP directory entry for Webshare. Once you have unpacked all of your files, examine all of the files with file type HELP*. For example, use XEDIT and look at the files:

```
WEBSHARE HELPHTTP A1
HTTPD HELPTASK A1
```

You will find valuable information about Webshare contained in these files.

You can find additional documentation on the Internet:

- *An Introduction Writing Webshare CGI Scripts*, by Melinda Varian

  `http://www.beyond-software.com/Related/WebshareCGIs/WebshareCGIs.html`

- *A Beginners Guide to HTML*

  `http://www.ncsa.uiuc.edu/General/Internet/WWW/HTMLPrimer.html`

- *A Guide to URLs*

  `http://www.netspace.org/users/dwb/url-guide.html`

- *The WWW Common Gateway Interface Version 1.1*

  `http://www.ics.uci.edu/pub/ietf/http/related/draft-robinson-www-interface-01.txt`

Take a look on Melinda Varian's homepage at the following location. There you will find many links and information concerning REXX and CMS Pipelines.

`http://pucc.princeton.edu/~Melinda/`

4.3.2 Installation Requirements

To install Webshare, you must have at least:

- A Web browser installed on a PC or workstation. To test text only pages, the Charlotte Web browser will work well.
- The browser must have a gateway connection to your VM/ESA system.
- A function on your PC or workstation to load the received files up to your host in binary format (for example, OS/2 Communication Manager, Almcopy, or any other 3270 emulator with a feature to load files to a CMS user in binary form).

Software Requirements

For this Web server, there is no published guideline for the software and hardware requirements. It was tested on an VM/ESA System 1.2.2, but it should run on any prior release of VM/ESA. Table 3 shows the software requirements and Table 4 contains the hardware requirements.

Table 3. System Software Required

Requirement	**Minimum Acceptable Level**
VM/ESA	1.1.0
Note: VM/SP5 may used if either the CMS Pipelines PRPQ (RPQ P81059 5799-DKE) or the no-cost Princeton CMS/TSO Pipelines runtime is installed (5785-RAC/SB5409).	
TCP/IP	VM TCP/IP Version 2
REXX	REXX/Sockets Version 2

Table 4. System Resources Required

Requirement	**Minimum Acceptable Level**
Free disk space	>= 1 Cylinder 3380, depending on the files you want to provide
Storage	12 MB virtual storage
Privilege class	CP privilege class G

Programmer Skills

The skills that are required to install, configure, and administrate Webshare are as follows:

- System programmer skills for VM

System programmer skills are required to provide data space for Webshare. Skills with SFS are required to decide whether to use the minidisk file system, the SFS, or a mixture of both.

- Networking skills

  You must have TCP/IP skills. You must be able to update the existing TCP/IP configuration. If you do not have an installed TCP/IP network, you must be able to install and configure it.

- System administration skills

  You must know the security structure of your company, and implement Webshare within this structure.

- Programming skills

  You need skills in HTML, if you want to provide your own pages. Skills in REXX and CMS Pipelines are required to write CGI scripts.

- PC or workstation skills

  General skills include operating a Web browser and uploading files to the host.

4.3.3 Installation

Start the installation by creating a directory entry for the Webshare, as shown in Figure 19.

```
00000 * * * Top of File * * *
00001 USER HTTPDn   ********  12M  12M G
00002 * where n is 0, 1, 2, 3, etc,  or even just nothing
00003 * That is, you can have just one "HTTPD" or you can
00004 * have several.
00005 INCLUDE  CMS
00006 ACCOUNT  BOH101Z  SYS
00007 LINK     TCPMAINT  591  591 RR
00008 LINK     TCPMAINT  592  592 RR
00009 *
```

Figure 19. Example of a Directory Entry

MDISK statements for this directory entry will be required if you implement the server on CMS minidisks. When using SFS, use the IPL statement to specify the default file pool, for example:

IPL CMS PARM FILEPOOL SFSLSY2:

Of course you can supply both MDISK and SFS statements.

Extract the files to the minidisk, or to the subdirectories using the CMSHTTPD VMARC file you obtained from the Web.

If you use the SFS, it runs as shown in Figure 20 on page 64.

```
Ready; T=0.03/0.03 13:59:36
VMARC UNPK CMSHTTPD VMARC A = = G
CMSHTTPD FILELIST G1. Bytes in=    1520, bytes out=    2140 (   140%).
CMSHTTPD README   G1. Bytes in=    2640, bytes out=    3586 (   135%).
CMSHTTPD COPYRIGH G1. Bytes in=     960, bytes out=    1082 (   112%).
CMSHTTPD HTML     G1. Bytes in=    4000, bytes out=    5801 (   145%).
HTTPD    EXEC     G1. Bytes in=    4160, bytes out=    6607 (   158%).
HTTPD    CONFIG   G1. Bytes in=    4480, bytes out=   13434 (   299%).
HTTPD    DIRECT   G1. Bytes in=     720, bytes out=    1040 (   144%).
HTTPD    REXX     G1. Bytes in=    3760, bytes out=    6061 (   161%).
WEBSRVSP REXX     G1. Bytes in=    1680, bytes out=    2179 (   129%).
WEBSRVLS REXX     G1. Bytes in=    2000, bytes out=    2932 (   146%).
WEBSRVHT REXX     G1. Bytes in=    2880, bytes out=    4951 (   171%).
WEBSRVHM REXX     G1. Bytes in=    3440, bytes out=    5533 (   160%).
WEBSRVHL REXX     G1. Bytes in=     880, bytes out=     835 (    94%).
WEBSRVRT REXX     G1. Bytes in=    2400, bytes out=    3979 (   165%).
WEBSRVDC REXX     G1. Bytes in=    1120, bytes out=    1755 (   156%).
MAKETEXT REXX     G1. Bytes in=    4320, bytes out=    6922 (   160%).
WEBUME   TEXT     G1. Bytes in=    8400, bytes out=   15680 (   186%).
cont.
```

Figure 20. Example Extracting Files from A-Disk to G-Disk

Once the files are loaded, access the TCP/IP 591 and 592 disks, then enter HTTPD to start the server. QUIT will terminate the server process. Figure 21 shows a view of the default home page.

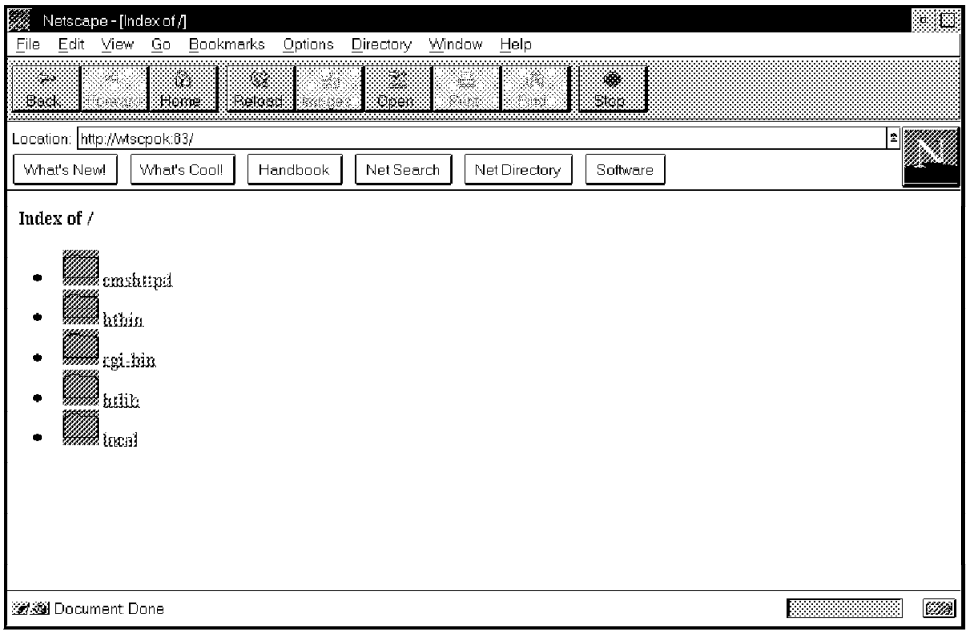


Figure 21. First Connection to Webshare after Installation

4.4 Configuration

To configure Webshare, you have to decide which file system you want to use.

The SFS is comparable with the file system used in UNIX systems. To use it, create a subdirectory under your existing file pool with the name WEBSHARE. This WEBSHARE directory works the same as it does in UNIX file systems. All

contents of the WEBSHARE directory are shared with the Web. That means all objects provided here are available for a Web browser. Subdirectories under the WEBSHARE directory are shown as an index by your Web browser. Selecting one of the subdirectories will show its content. See Figure 22 for a comparison of the UNIX file structure and the SFS file structure.

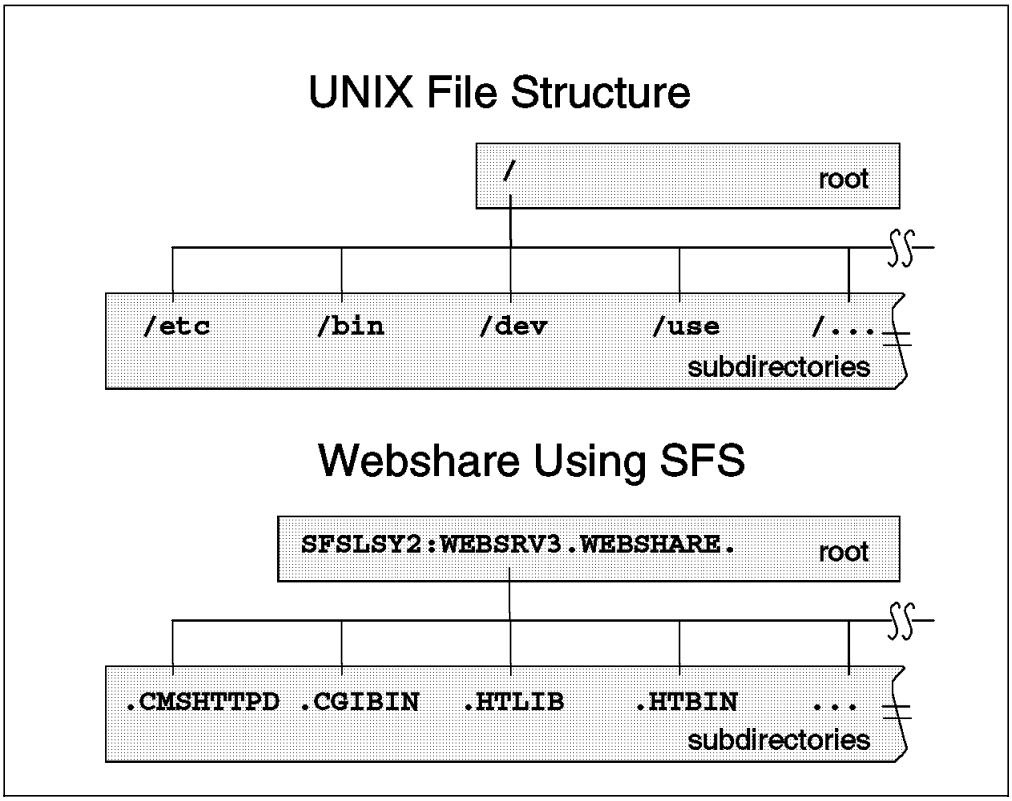


Figure 22. UNIX File System and Webshare in Shared File System

In a minidisk file system, you have to use file lists instead of subdirectories. File lists are simply text files, with the file type FILELIST. The file list with the name WEBSHARE FILELIST works like an WEBSHARE SFS directory. File lists that are contained in the WEBSHARE FILELIST work similar to subdirectories. Use the file lists to order your files and the programs you want to serve on the Web.

Refer to Figure 23 on page 66 to see the relationship between file lists and the files. Files can only be accessed by your server if they are recorded in a file list. As shown, the accessible files on your minidisk may be only a subset of its content.

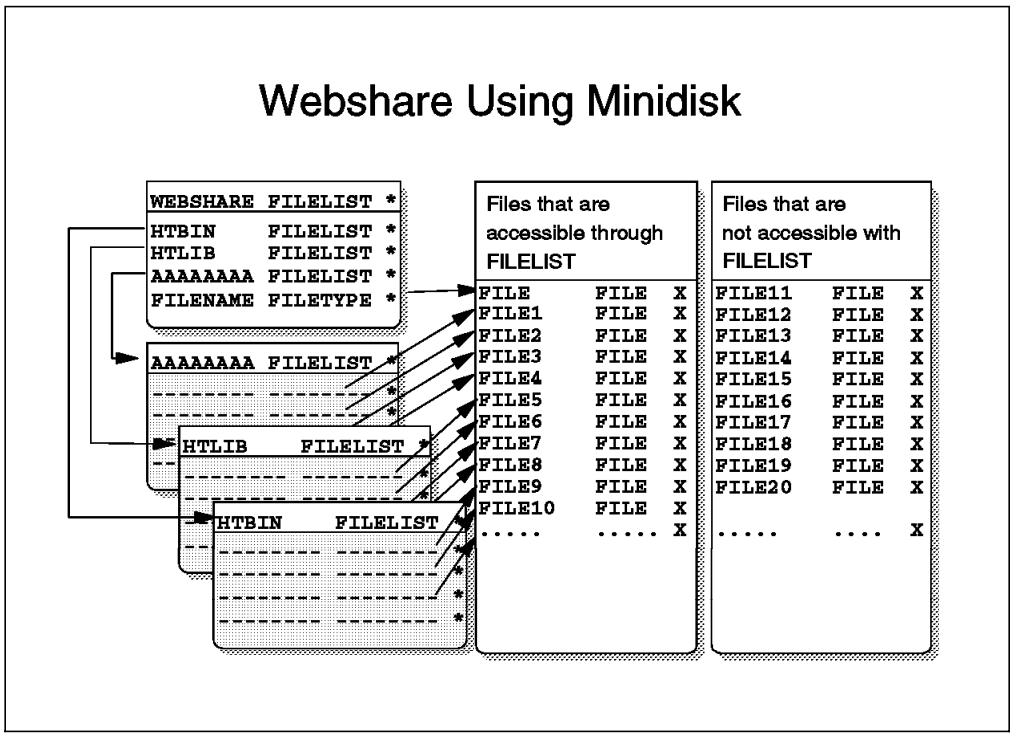


Figure 23. Webshare Using Minidisk File System

If you do not have a security system such as RACF or VM:Secure you should implement the SFS to use the authorizations it provides. If you do so, remember to use a file list for your CGIs. File lists for CGIs must reside on CMS minidisks, not SFS directories. CGIs must be on a disk or directory already accessed by Webshare. That means Webshare does not look at the contents of a subdirectory and then provide the CGIs. You have to provide them by name and location.

Another SFS restriction from Webshare is that there is no support for multiple SFS file pools. The user data the server accesses must reside in the server's file pool.

See Figure 24 for the structure of a FILELIST file. A comment is indicated by an asterisk. All other lines must start with a blank.

```
00000 * * * Top of File * * *
00001 * - Copyright 1995, Richard M. Troth
00002 *
00003 *         Name: HTBIN FILELIST
00010       POSTTEST *CGI     *
00011       CMSHELP  *CGI     *
00012       CPQ      *CGI     *
00013       FINGER   *CGI     *
00014       IMAGEMAP *CGI     *
00015       NAMEFIND *CGI     *
00016       YOURCGI  *CGI     *
00017 *
```

Figure 24. Example of a File List Containing CGI Entries

4.4.1 Customization

Once you have installed the server, start it by invoking the HTTPD EXEC. Do not forget to have the TCP/IP disks 591 and 592 accessed. (You can do it by adding an ACCESS statement to the PROFILE EXEC of your server machine.) Stop the server with the command QUIT. If the TCP/IP port you want to use is port 80, then the server will run with the default port provided by the HTTPD EXEC. If you have other applications using port 80, you have to change the default value.

We recommend that you run with the default port 80 for your Web server. There are versions of old Web browsers that hard code port 80 at the end of every URL. But, if you must change it, override it by editing the HTTPD CONFIG file. A sample configuration file is shown in Figure 25 and the changed port is in bold.

```
00000 * * * Top of File * * *
00001 * - Copyright 1994, Richard M. Troth, all rights reserved.
00002 *
00003 *         Name: HTTPD CONFIG
00004 *               CMS HTTP Server configuration file
00005 *       Author: Rick Troth, Houston, Texas, USA
00006 *               (with lotso help from several Internet sources)
00007 *         Date: 1994-Oct-22
00008 *
00009 *         Note: Some things you can set are:
00010 *
00011 *                     PORT=
00012 *                     LOGPIPE=
00013 *                     CGIUSERS=
00014 *
00015 *
00016 * send logging information to the console (default)
00017 * LOGPIPE=CONSOLE
00018 PORT=83
00019 *
```

Figure 25. Extract of HTTPD CONFIG File

Update the PORT statement to the HTTPD CONFIG as it is shown in Figure 25. Figure 26 shows the server starting with your specified port number.

```
Ready; T=0.04/0.05 11:44:58
httpd
WEBSRV8399  HTTPD Version 1 Release 2.3
TCPSHELL: PORT 83
TCPSHELL: Ready;
```

Figure 26. Webshare Running

You may experience poor performance between your browser and Webshare. This can result from the client being unknown to any name server. To solve this problem, update the TCPIP DATA file by changing the parameter for RESOLVERTIMEOUT to a value of 1.

In the HTTPD CONFIG file, there is an argument that also has an influence on performance. The argument IDENT is set to ON by default. That means the server looks for an identification server at the requesting client. This may be needed if the client is a multi-user system, providing many user IDs (such as an

AIX workstation). Most VM installations can switch IDENT to OFF for increased performance.

4.4.2 Administration

The administration of Webshare server depends on the file system used.

- Using minidisk file system

  All files you want to share with the Web must be contained in the WEBSHARE FILELIST file. The contained file lists and their contents are provided to the browsers. Use various file lists to categorize the available files. See Figure 27 as an example for a WEBSHARE FILELIST.

```
00000 * * * Top of File * * *
00001 * - Copyright 1994, Richard M. Troth, all rights reserved.
00002 *
00003 *         Name: WEBSHARE FILELIST
00004 *               Specifies those files which you wish
00005 *               to share with "the web", matching their local
00006 *               CMS FILEIDs (fn/ft/fm) to their external (web) name
00007 *         Date: 1994-Jan-29, Oct-30
00008 *
00009 *         Note: comments must start with an asterisk
00010 *               otherwise, lines must start with a blank
00011 *
00012 *         Note: this is your "root" menu definition.
00013 *               If you want something to be available to the web,
00014 *               either list it here,  or list it in one of the
00015 *               sub-lists listed here  (SFS dirs or FILELISTs).
00016 *               LOCAL is specifically intended for your own use.
00017 *               I will not put anything in "/local" in the package
00018 *
00019 *     fn       ft       fm path     name
00020 *     xxxxxxxx xxxxxxxx xx xxxxxxxx "xxxxxxxx"
00021 *     INDEX    HTML     *
00022       HTBIN    *        *
00023       HTBIN    *        *  cgi-bin
00024       HTLIB    *        *
00025       LOCAL    *        *
```

Figure 27. WEBSHARE FILELIST Provided by Webshare

  The file list with the name CMSHTTPD FILELIST represents the server code itself. It must be on a disk that is accessed by the server machine.

- Using SFS

  Create a subdirectory under your existing file pool with the name WEBSHARE. The subdirectories that you create under the WEBSHARE subdirectory are provided as an index. That means if you browse the subdirectory WEBSHARE, then you see the names of the other subdirectories there. Selecting one of the subdirectories with the browser will show their contents.

  Do not forget to grant the authority for the CMSHTTPD subdirectory to the server machine.

- Using a mixture of SFS and minidisks

  You have to use a mixture of SFS and minidisks if you want to provide CGI scripts to the users. It is not supported by the server to look in the index of a subdirectory and provide the CGI scripts. You have to provide them by a

file list. The server first looks in the subdirectories and then to the file lists on the accessed minidisks.

- MIME type mapping

  In the HTTPD CONFIG file there is a table of how to transmit files based on their file type. It controls whether the server uses 7bit, 8bit, binary, CMS, or code page translation required by the browser. If you have to add your own file types, add them to the configuration file. Use the form:

  TYPE *filetype gopher_type MIME_type transport*

4.5 Security

Webshare does not provide any additional security mechanisms beyond those already existing within VM/ESA. There are, however, ways to discriminate the access to files.

4.5.1 INDEX HTML

You can use the INDEX HTML file to provide a controlled access to your server.

If a file named INDEX HTML exists, the Webshare server sends its contents instead of an automatic index of the SFS directory or FILELIST. You can use this HTML file as the server's root menu. But this will not prevent the use of your files and programs. It will only hide these objects. Though they are hidden, they can be accessed. You only have to write their complete URLs at the line entry of your Web browser to access them. Therefore, it is highly recommended that you use the security provided by your security system; for example, RACF or VM:Secure. At a minimum, you should use the authorities provided by the SFS.

4.5.2 User Web Spaces

The server provides user-defined Web spaces. You must decide very carefully to whom you will give the permission to provide CGI scripts to the server. Authorize users to write CGI scripts for the server in the HTTPD CONFIG file. See Figure 28 how to allow users to write CGIs. Their user IDs must be listed besides the CGIUSERS statement.

```
00027 * List those users who are permitted to write CGI scripts.
00028 * For any user in this list, CGI scripts will run from their
00029 * personal web space w/o having to move to the server's file space.
00030 *
00031 CGIUSERS=maint d1mattw leess jf
```

Figure 28. User IDs Providing CGI Scripts

It is very dangerous when people write CGI scripts for the server in their own data space; these CGIs do not run in the machine of the user who provides them. They run in your server machine.

Figure 29 on page 70 shows how a request to a Web space works. If a shutdown CGI is run on a server with CP class A privilege, you could have more than the desired Web server shutdown.

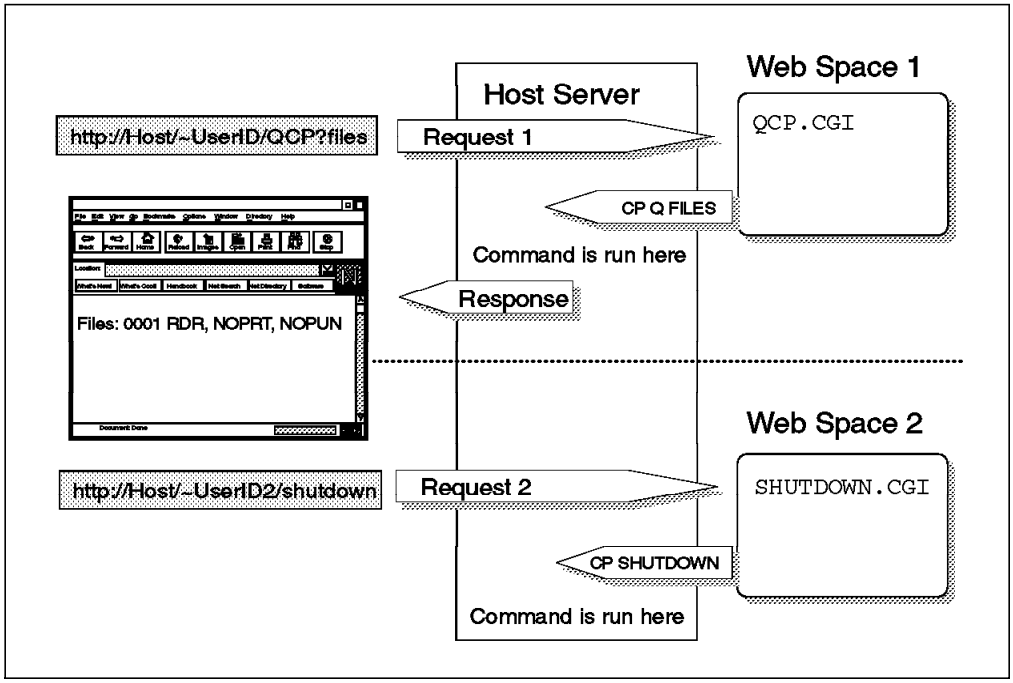


Figure 29. Requesting CGIs on Web Spaces

4.5.3 Logging

When you log on the Webshare server the first time, the logging is set to CONSOLE by default. It is also set to TERSE, filtering the information to the console. See Figure 30 for an example of TERSE. Every access is shown at the console as a one line entry.

```
Ready; T=0.01/0.01 09:17:36
httpd
WEBSRV8399  HTTPD Version 1 Release 2.3
TCPSHELL: PORT 83
TCPSHELL: Ready;
@9.12.14.69 YThu Nov 07 09:17:55 1996" GET / HTTP/1.0
@9.12.14.69 YThu Nov 07 09:18:03 1996" GET /cgi-bin/ HTTP/1.0
@9.12.14.69 YThu Nov 07 09:18:27 1996" GET /cgi-bin/cpq?files HTTP/1.0
```

Figure 30. Default Logging

You can change to VERBOSE by including a LOGPIPE statement in the HTTPD CONFIG file. In the servers, code LOGPIPE=CONSOLE is the default value. You get more logging information if you explicitly state VERBOSE in the HTTPD CONFIG file. Figure 31 on page 71 is an example of VERBOSE logging.

```
Ready; T=0.01/0.01 09:20:13
httpd
WEBSRV8399  HTTPD Version 1 Release 2.3
DMSHTT8066  Logging console
DMSHTT8274  Loading ...
GETFMADR HTTPD MAKETEXT PIPESOCK TCPSHELL WEBSRVDC WEBSRVHE WEBSRVHL
BSRVHT WEBSRVLS WEBSRVOF WEBSRVPO WEBSRVRT WEBSRVSP
DMSTCP2323I Restarting HTTPD at 1996110T 09:20:15.
TCPSHELL: MAXDESC 40
TCPSHELL: SVM TCPIP
TCPSHELL: LOCALHOST wtscpok.itso.ibm.com
TCPSHELL: SOCKET 1
TCPSHELL: PORT 83
DMSTCP820I  HTTPD has been started
DMSTCP740I Execution begins...
TCPSHELL: Ready;

TCPSHELL: EL: READ 1 WRITE EXCEPTION
TCPSHELL: ES:  1
TCPSHELL: Accepted 2 at 09:22:59 client @9.12.14.69
@9.12.14.69 YThu Nov 07 09:22:59 1996" GET / HTTP/1.0
WEBSRVSP:
WEBSRVRT:
WEBSRVSP:
WEBSRVHT:       WEBSHARE *FL      *
WEBSRVHM:       WEBSHARE *FL      *
WEBSRVLS:       WEBSHARE *FL      *
WEBSRVHM: * - Copyright 1994, Richard M. Troth, all rights reserved.
WEBSRVOF: Content-Type: text/html
WEBSRVHM: done.
WEBSRVHT: done.
WEBSRVOF: done.
TCPSHELL: Closed 2 at 09:22:59
```

Figure 31. LOGPIPE Statement Switched to VERBOSE

Of course you can record all information to the console log of your Webshare server machine. But you can also provide a file name besides the LOGPIPE statement in the HTTPD CONFIG file. In this file, your logging information is stored. Use two greater-than signs (>>) to append the logging information to your logging file, as shown here:

```
00024 LOGPIPE >> HTTPD LOGFIL A
```

If you only use one greater-than sign, only the last access is logged.

Figure 32 shows the contents of a logging file.

```
00000 * * * Top of File * * *
00001 @9.12.14.69 -Thu Nov 07 09:24:54 1996- GET / HTTP/1.0
00002 @9.12.14.69 -Thu Nov 07 09:24:56 1996- GET /cgi-bin/ HTTP/1.0
00003 @9.12.14.69 -Thu Nov 07 09:25:06 1996- GET /cgi-bin/cpq?files HTTP/
00004 @9.12.14.69 -Thu Nov 07 09:25:18 1996- GET / HTTP/1.0
00005 * * * End of File * * *
```

Figure 32. Example of a Logging File

Error Messages, including Webshare's, can be silenced by setting the CP SET EMSG OFF in the server's PROFILE EXEC.

4.6 Examples

There are six CGI examples and one HTML example provided by Webshare:

CMSHELP CGI	Pipeline stage for interpreting the CMS Help database.
CPQ CGI	Query some things from CP, returning results as HTML. This is a good example for providing VM functions to HTML browsers.
FINGER CGI	Gateway finger that looks up IDs in the Web.
IMAGEMAP CGI	Processes events on imagemap images from the client.
POSTTEST CGI	Verifies the correct operation of HTML forms.
YOUTRYIT CGI	A skeleton to create your own CGI.
CMSHTTPD HTML	The HOMEPAGE HTML is the only provided html file on the server.

Figure 33 shows the results of the CPQ CGI, provided as a sample CGI. The parameter for CPQ is separated by a question mark; here it is the CPLEVEL parameter.

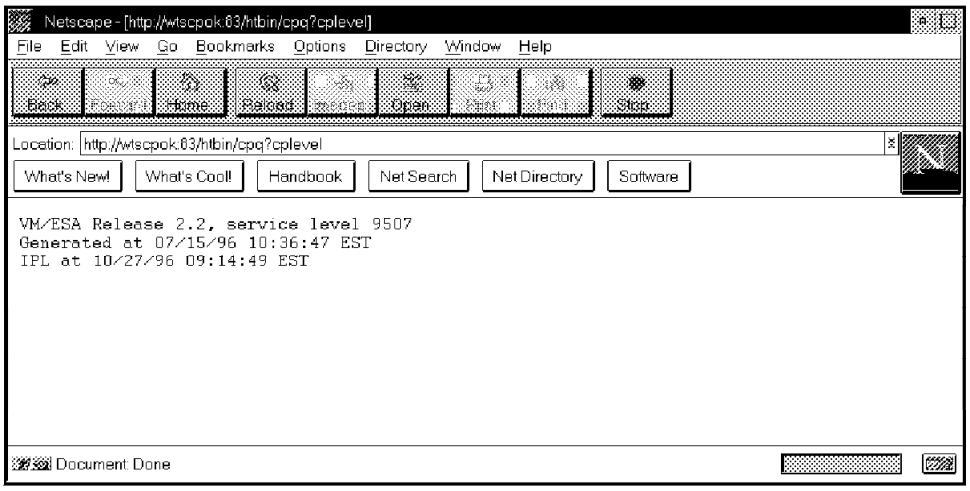


Figure 33. Using the QPQ CPI for QUERY CPLEVEL

Have a short look at two of the provided examples. They will help you to understand and write your own CGI and HTML scripts.

4.6.1 Sample CPQ CGI

A Common Gateway Interface (CGI) script is just a program whose name is a URL. When your browser asks the server for that URL, the server runs the program in its own virtual machine. On a VM system, a CGI script is just a CMS Pipeline filter written in REXX. Anything after a question mark (?) in the URL is passed to the CGI script and is retrieved with the REXX statements parse arg.

```
/* Copyright 1994, Richard M. Troth
 *
 *        Name: CPQ HTTPPROC
 *              query some things from CP returning results as HTML
 *              same arguments as RSCS CPQ except raw "CPUID"
 *      Author: Rick Troth, Houston, Texas, USA
 *        Date: 1994-Mar-01
 *
 *              Thanks to Tim Drais for pointing out the '15'x hole.
 */

'OUTPUT' "<html>"                                          1
Address "COMMAND" 'GLOBALV SELECT HTTPD GET VRM'           2
'OUTPUT' "<!CMS HTTPD" vrm "CPQ CGI>"                      3

/*  eliminate piggy-backed CP command trojans  */
'ADDPIPE *.INPUT: | SPLIT AT' '001500'x '|  *.INPUT:'
If rc [= 0 Then Exit rc

'PEEKTO RECORD'                                            4
If rc [= 0 Then record = ""
Parse Upper Var record arg1 arg2 .

Select  /*  arg1  */                                       5

    When Abbrev("CPLEVEL",arg1,2) Then  args = "QUERY CPLEVEL"
    When Abbrev("CPUID",arg1,3) Then    args = "QUERY CPUID"
    When Abbrev("FILES",arg1,1) Then    args = "QUERY FILES"
    When Abbrev("INDICATE",arg1,1) Then args = "INDICATE LOAD"
    When Abbrev("LOGMSG",arg1,1) Then   args = "QUERY LOGMSG"
    When Abbrev("NAMES",arg1,1) Then    args = "QUERY NAMES"
    When Abbrev("TIME",arg1,1) Then     args = "QUERY TIME"
    When Abbrev("USER",arg1,1) Then     args = "QUERY USER" arg2
    When Abbrev("USERS",arg1,5) Then    args = "QUERY USERS"

    Otherwise                           args = "QUERY"

    End  /*  Select  arg1  */

'OUTPUT' "<pre>"                                           6
'CALLPIPE VAR ARGS | CP | *:'                              7
'OUTPUT' "</pre>"                                          6

'OUTPUT' "</html>"                                         8

Exit
```

Figure 34. Source Code of CPQ CGI Script

The notes refer to Figure 34.

1 Output of <html> provides the start of an HTML page for the browser.

2 The server gets a variable.

3 It creates an output of a comment, including the previous variable.

The next three lines parse the input for a CP line-end X'15'. Make sure this value matches the CP line end value used by your server.

4 PEEKTO RECORD command. There the server obtains the arguments following the question mark in the URL.

5 The SELECT block verifies the input and creates a VM CP command, for example, QUERY CPLEVEL.

6 The next output is <pre>, creating a tag for the HTML page. This tag tells the HTML browser that the text is already preformatted (by VM). There is no need to reformat it. Tag </pre> shows the end of the preformatted output.

7 The CALLPIPE command gives the previously created arguments to CP and passes along all the output from there.

8 The output of </html> finishes the HTML page. When you look at Figure 35, you will see that the CGI script created an HTML page for the browser, including the output you requested in the URL after the question mark.

```
<html>
<!CMS HTTPD 1.2.3 CPQ CGI>
<pre>
VM/ESA Release 2.2, service level 9507
Generated at 07/15/96 10:36:47 EST
IPL at 10/27/96 09:14:49 EST
</pre>
</html>
```

Figure 35. Output of the QCP CGI Script As Source Code

4.6.2 Sample CMSHTTPD HTML Script

The HTML as shown in Figure 37 on page 75 was used to create the screen shown in Figure 36.

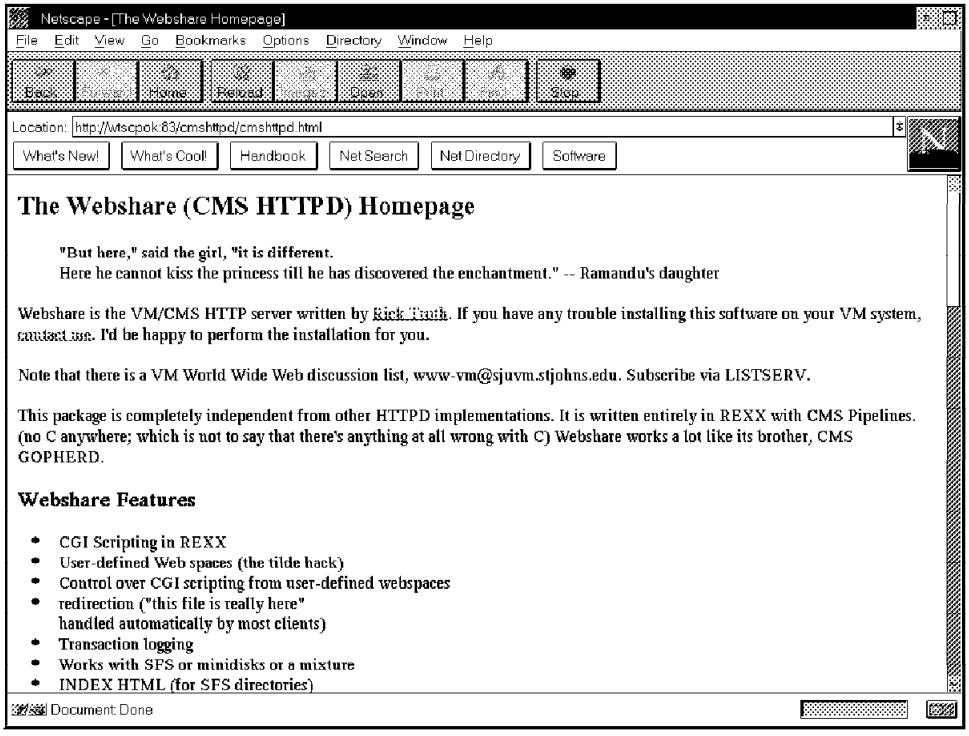


Figure 36. The CMSHTTPD HTML Is the Only Provided HTML File

Look at the extract of the source code of the only provided HTML file CMSHTTPD. Compare it to obtain the effect of the control tags in an HTML script.

An HTML file is only a document. It is marked up using the Hypertext Markup Language (HTML).

```
00001 <html>                                                              1
00002 <head>                                                              2
00003 <title>The Webshare Homepage</title>                                3
00004 </head>                                                             2
00005 <body>                                                              4
00006 <h2>The Webshare (CMS HTTPD) Homepage</h2>"                         5
00007 <p>                                                                 7
00008 <blockquote>                                                        6
00009 "But here,"  said the girl,  "it is different. <br> Here he
00010 cannot kiss the princess till he has discovered the enchantment."
00011 -- Ramandu's daughter"
00012 </blockquote>"                                                      6
00013 <p>                                                                 7
00014 Webshare is the VM/CMS HTTP server written by                       10
00015 <a href="http://ua1vm.ua.edu/-troth/trothsig.html">Rick Troth</a>.
00016 If you have any trouble installing this software on your VM system,
00017 <a href="http://ua1vm.ua.edu/-troth/trothsig.html">contact me.</a>
00018 I'd be happy to perform the installation for you.
00019 <p>                                                                 7
00020 Note that there is a VM World Wide Web discussion list,
00021 www-vm@sjuvm.stjohns.edu.   Subscribe via LISTSERV.
00022 <p>                                                                 7
00023 This package is completely independent from other HTTPD
00024 implementations. It is written entirely in REXX with CMS Pipelines.
00025 (no C anywhere; which is not to say that there's anything wrong
00026 with C)
00027 Webshare works a lot like its brother, CMS GOPHERD.
00028 <p>                                                                 7
00029 <h3>Webshare Features</h3>                                          5
00030 <ul>                                                               9
00031 <li>    CGI Scripting in REXX
00032 <li>    User-defined Web spaces (the tilde hack)
00033 <li>    Control over CGI scripting from user-defined web spaces
00034 <li>    redirection ("this file is really here"
00035 <br>    handled automatically by most clients)
00036 <li>    Transaction logging
00037 <li>    Works with SFS or minidisks or a mixture
00038 <li>    INDEX HTML (for SFS directories)
00039 .
00206 .
00207 .
00208 </body>                                                             4
00209 </html>                                                             1
```

Figure 37. Extract of CMSHTTPD HTML Source

1 An HTML document always starts with the <html> tag and ends with </html>. You can take it as a frame.

2 The <head> and </head> tags mark a prologue. In this prologue, a title included.

3 The title is marked by the <title> and </title> tags.

4 The <body> and </body> tags include the main part of the document. The body of a document contains all the displayed information for a document. Unlike the head, the body of a document can contain format information. Some of the key components of the body of the document are the text of the document itself, section titles, or headings. It can also contain address or author-contact information.

5 HTML supports six levels of headings. It is important to note that headings are actually logical formatting directives. That means that each browser implements the headings as they are provided for that platform. It may be that the same document shows a 14-point Courier Bold on one platform and a 20-point Helvetica Bold on another platform. Headings are indicated by <h#> and </h#>, where # is the level of the heading. The levels shown in the CMSHTTPD HTML file are <h2>, and <h3>.

6 The <blockquote> and </blockquote> tags indent the text as though it were a quotation.

7 The <p> tag marks a paragraph in the text. It starts at a new line and leaves a blank line between the text before the tag and the text after the tag.

8 The
 tag also starts at a new line, but it does not place a blank line between the two sections of the text.

9 The <ul> flag indicates an unnumbered list. Each item in this list starts with the <li> tag. The tag ends the unnumbered list.

10 A link is created when the anchor tags <a> and are placed around the text. There is a hypertext reference tag, href, added to it. The server creates a link to the URL in double quotes and loads http://ualvm.ua.edu/~troth/trothsig.html when you click either the highlighted >*Rick Troth*<, or the *contact me*.

See 1.6.1, “Creating an HTML Document” on page 6 for more information on writing HTML. You will soon be able to write your own Web pages.

4.6.3 Imagemap Support

Webshare supports imagemaps. Imagemaps contain images with special hot spots that you can click with your browser. The location you click determines what predetermined URL is loaded. To create an imagemap, you must provide an HTML page. An example of this page is shown in Figure 38 on page 77. In this HTML page, you must refer to the program IMAGEMAP, which is provided by the server. The program needs the name and location of an imagemap as an argument. Use the HREF statement to refer to the IMAGEMAP program and to the imagemap. Use the IMG SRC statements to define the areas. The image must be a GIF file.

```
00000 * * * Top of File * * *
00001 <html>
00002 <head>
00003 <Title>WEBSHARE IMAGEMAP SUPPORT </Title>
00004 </head>
00005 <body>
00006 <h1> Sample Home Page Testing Imagemaps  </h1>
00007 <A HREF="/htbin/imagemap;/imglib/img.map">
00008 <IMG SRC="http://WTSCPOK:83/imglib/tworect.gif" ISMAP></A>
00009 <P>
00010 </body>
00011 </html>
00012 * * * End of File * * *
```

Figure 38. Sample of an Image Html File

The imagemap file contains the locations of the hot spots and connects them to the URLs that you want to load. You must use a default URL, if no location was selected. Use the keyword RECT as a keyword in the image file, followed by the path or URL you want to load. The upper left corner of the hot spots is defined by two numbers, reflecting to the location on the X-axis and Y-axis in the picture. They are separated by a comma. The second pair of numbers reflect to the lower right corner of the rectangle. Only rectangles are supported by Webshare. An imagemap file is shown in Figure 39.

```
00000 * * * Top of File * * *
00001 # loads rectone.html if clicking on the first rectangle
00002 RECT  /imglib/rectone.html   001,020 238,089
00003 # loads recttwo.html if clicking on the second rectangle
00004 RECT  /imglib/recttwo.html   001,100 050,150
00005 # loads homepage of WTSPOK:83 if clicking another spot
00006 DEFAULT http://WTSCPOK:83
00007 * * * End of File * * *
```

Figure 39. Sample of an Imagemap File

Chapter 5. EnterpriseWeb/VM

This chapter introduces the Beyond Software Inc. EnterpriseWeb/VM product. EnterpriseWeb/VM is a cost effective Web server for the VM/ESA platform that will provide Web services for your company on your existing VM/ESA installation. The Web Server is used for both Internet and intranet services. Figure 40 shows the default image shown after EnterpriseWeb/VM is installed.

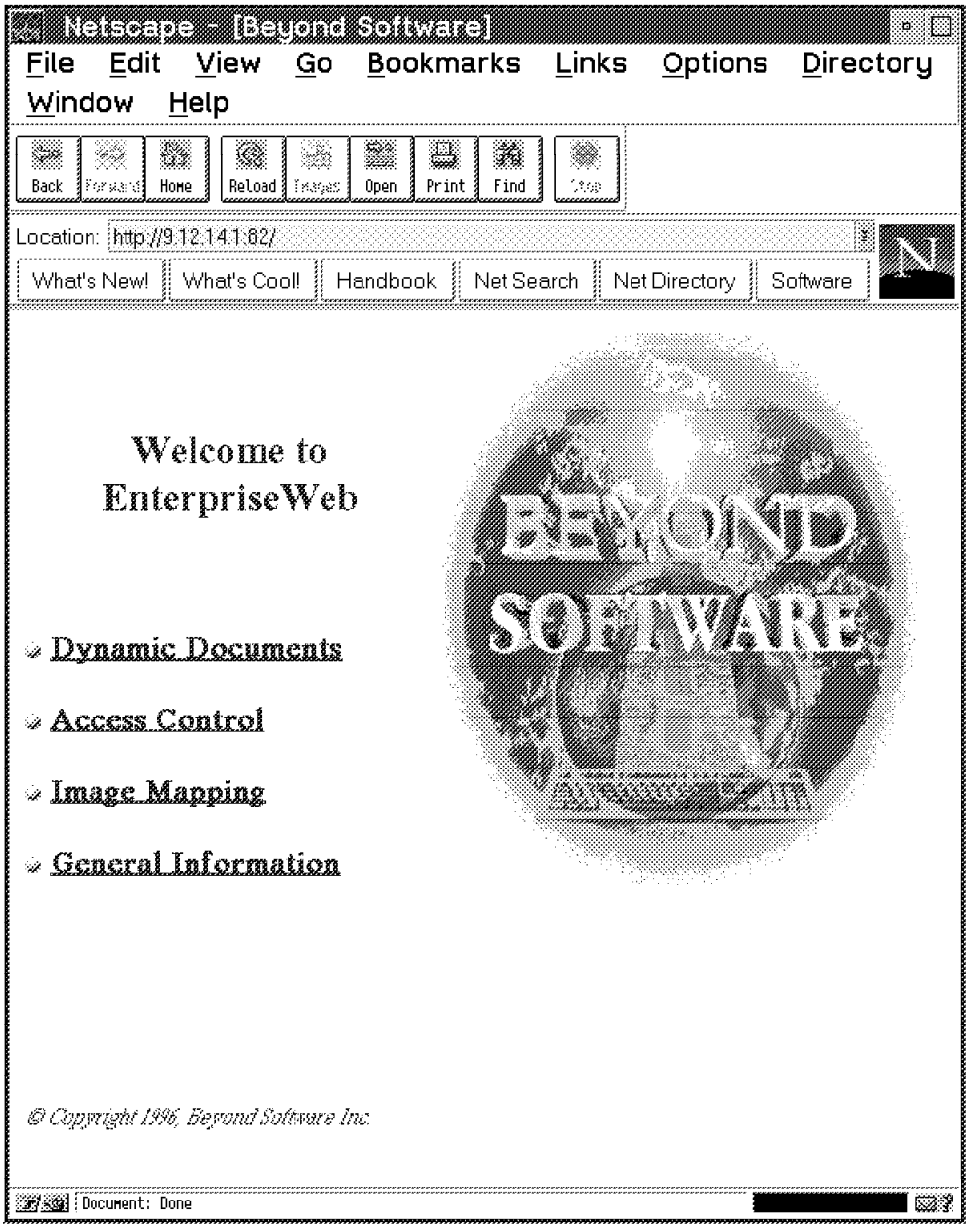


Figure 40. Beyond Software Inc.'s EnterpriseWeb/VM

5.1 Introduction

Today, new technologies are being developed all the time, and companies must use these technologies if they wish to survive and grow in today′s competitive society. Generally, when new technology is introduced, the old equipment is discarded; however, Web servers for the Internet and intranet are now available on your existing VM/ESA platform.

Beyond Software Inc.′s EnterpriseWeb/VM is a World Wide Web (WWW) server based on the popular Webshare server, which has been in use for some time now. EnterpriseWeb/VM uses Hypertext Transfer Protocol (HTTP) and provides similar functions to NCSA http or Netscape′s Netsite. It runs on your existing VM/ESA system, allowing access to existing applications and data. EnterpriseWeb/VM gives you the capability to unlock your system′s information in a secure way.

This chapter describes the company, Beyond Software Inc., the product, the functions it contains, and some of the experiences met while installing and testing this product.

5.2 About Beyond Software Inc.

Beyond Software Inc. is a Silicon Valley (CA) based company founded on July 1995 with the mission of bringing Internet technology to the enterprise server environments.

Beyond′s initial product, EnterpriseWeb/VM has been shipping since April 1996. It is based on a highly reengineered version of Webshare (the commercial rights for which where acquired by Beyond Software). In May 1996, Beyond Software became an IBM Business Partner and in August 1996 Beyond Software established a partnership with Velocity Software which provides performance monitoring capability and marketing channels.

Further information about Beyond Software Inc., EnterpriseWeb/VM and other products from Beyond can be found on the Web at the following address:

```
http://www.beyond-software.com
```

When you access Beyond Software Inc.′s WebPage, you will find:

- A demonstration of EnterpriseWeb/VM
- How to order the product
- A 30 day free trial version, which is unlocked through a software key you receive within 24 hours from the download process
- Information about the company
- Reference sites for EnterpriseWeb/VM
- Additional interesting company information

5.3 Obtaining EnterpriseWeb/VM

You can obtain a copy of EnterpriseWeb/VM as either a 30 day trial period or the full package.

- 30 day trial package

  Having down loaded EnterpriseWeb/VM over the Internet, you must contact Beyond Software Inc. through e-mail to obtain a key. This version may not be the latest version; however, it does contain most of the key features. With the trial version, you receive a copy of the code and enough documentation for you to install and run the Web server.

- Full Version

  To obtain the full version, send an order either through e-mail or to the address shown on the Web page or on Beyond direct-marketing materials. Shipped to you with your paid order are the following:

  – A magnetic tape that contains all the machine readable product codes and examples

  – Documentation

5.4 Documentation

The documentation has three elements:

1. *EnterpriseWeb/VM Installation Guide and Reference Manual*
2. Various reference materials that relate to the World Wide Web (WWW)
3. An HTML external publication, *The HTML Sourcebook; A Complete Guide to HTML 3.0*, ISBN 0-471-14242-5

The manual and reference material, together with the product and feedback form, come in one standard white binder. The dividers are already in place between the different sections, making the material easier to access.

Inexperienced Web administrators and system programmers may feel they require more than the provided documentation. This chapter helps to augment the provided documentation.

5.4.1 The Manual

The following discussion introduces the sections in the manual.

Introduction: This chapter contains the usual information about how the manual is organized and how to interpret it. It also contains information about the Urgent Care Support and how to obtain support for EnterpriseWeb/VM.

EnterpriseWeb/VM Installation: This chapter contains the basic standard instructions on how to install the product with some planning information. It contains examples of the directory entries for the virtual machines that are required. Their minidisk size requirements are in Appendix A of the manual, or in Table 10 on page 87 of this publication. There are no sample directory definitions on the installation tape. You must type them yourself.

No mention is made in this section of the manual of the SFS default directory names. You can look in Appendix A of the manual, or go through the installation program to obtain what they are.

There a bias toward the use of minidisks rather than the SFS in the manual. Beyond is aware that most VM/ESA systems use minidisks for product installation. However, using the SFS gives you more control over data that is allocated to your Web server, and this is the installation method we recommend.

Configuring Your Installation: This chapter explains the files and parameters that are needed to customize the EnterpriseWeb/VM server to your requirements. It does not cover how to set up and control access to the data. Included in this chapter is a section on how to run the installation verification programs (IVPs). This chapter contains topics on TCP/IP, external security managers (ESMs), changing default messages, installation verification, configuration directives (parameters), to name a few.

Security: This chapter explains how security is implemented using EnterpriseWeb/VM, the different types of files, and how directives are used.

Commands and Utilities: This chapter lists the commands and utilities that are used by EnterpriseWeb/VM. The commands include those to start and stop the server, and utilities to compile various files.

Using Your Web Site: This chapter explains FILELISTs, access to user data, and many more facilities of EnterpriseWeb/VM.

Logging: This chapter explains how to set up and use the Log Facility within EnterpriseWeb/VM.

Frequently Asked Questions: This chapter is under development. It will contain a list of common problems and tips on how to solve them.

Glossary and Appendixes: The back matter of the documentation.

Note

The commands that are contained in the control files are referred to as directives in this manual.

5.4.2 Reference Material

If you want to read the reference material before receiving the full package, you can download a copy from the World Wide Web.

Table 5 contains a list of the materials and their locations on the Web.

Table 5 (Page 1 of 2). Documents and Where to Locate Them

Document	Location on the WWW (URL)
Plunging into Pipes by Melinda Varian	http://pucc.princeton.edu/~pipeline/#MWV
Pipe Dreams - What's New in CMS Pipelines by Melinda Varian	http://pucc.princeton.edu/~pipeline/#MWV

Table 5 (Page 2 of 2). Documents and Where to Locate Them	
Document	**Location on the WWW (URL)**
An Introduction to Writing WEBSHARE CGI Scripts by Melinda Varian	http://pucc.princeton.edu/~ pipeline/#MWV
Hypertext Transfer Protocol - HTTP/1.0	http://www.ics.uci.edu/pub/ietf/http/
The WWW Common Gateway Interface Version 1.1	http://www.ics.uci.edu/pub/ietf/http/related/
Identification Protocol	gopher://ds.internic.net/00/rfc/rfc1413.txt
Uniform Resource Locator	gopher://ds.internic.net/00/rfc/rfc1738.txt
Uniform Resource Locator Reference	gopher://ds.internic.net/00/rfc/rfc1808.txt

5.4.3 HTML Sourcebook

HTML Sourcebook - A Complete Guide to HTML 3.0, ISBN 0-471-14242-5 is an external publication that is a very good introduction to HTML. This publication contains many examples of how HTML scripts are coded.

5.4.4 Sample Code

The sample code is delivered on the same tape as the product tape. It contains copies of the FILELIST, HTML, CGI, and many more types of files that are used with the IVP scripts.

By using the IVP and sample code, you can see how the different parts of the EnterpriseWeb/VM combine to make a useful Web server. It is a very good base to start from.

The sample code is installed as part of the product installation, but it can be loaded later using the CMS TAPE commands. The sample code is the fourth file on the tape and consists of the files types shown in Table 6.

Table 6. Included Samples	
Component	**Number**
Sample GIF Files	21
Sample CGI Scripts	15
HTML Pages	15
FILELISTs	7

HTML and FILELISTs will be discussed in detail in sections 5.7, “Functional Overview” on page 89 and shown in Figure 48 on page 100.

5.5 Pre-Installation Requirements

To run EnterpriseWeb/VM, certain resources must be allocated before you start.

5.5.1 Software Requirements

Table 7 contains the minimum software levels required to run EnterpriseWeb/VM.

Table 7. Minimum Software Levels

Resource	**Minimum Acceptable Level**
VM/ESA	1.1.0
Note: VM/SP5, VM/SP6 and VM/ESA 370 Feature may used if either the CMS Pipelines PRPQ (RPQ P81059 5799-DKE) or the no-cost Princeton CMS/TSO Pipelines runtime is installed (5785-RAC/SB5409).	
TCP/IP	2.0
REXX Sockets	2.01

5.5.2 Programmer Skills

The following skills are required to install, test, and configure EnterpriseWeb/VM.

- System Programming Skills

  The system programming skills are required to set up the hardware requirements and to install the code.

- System Administrator

  This person must know the security structure of the company and set this up within the Web server.

- Networking Skills

  The networking skills are required to set up and tailor the TCP/IP requirements.

- Design Skills

  A Web page must be interesting to a user. Therefore, it must be well-designed and easy to change.

- Programming Skills

  The programmers are required to write HTML and CGIs to leverage VM/ESA's resources.

- PC Skills

  IS infrastructures should be put in place to handle questions about the Web browsers.

It is probable that one individual will have more than one of these skills and sometimes all of them. At times, it might be a team which is constituted to get the product into the final requirements.

5.5.3 Other Requirements

To ensure that the installation is completed correctly, you will need the following resources:

- All the Web servers should be defined in your TCP/IP profile with a PORT number that is unique to the group of Web servers.
- Any HTML compliant Web browser with TCP/IP access to the server.

5.6 Installation and Operation

The installation of EnterpriseWeb/VM is quick and easy. To load all the code requires less than 10 minutes. To complete the installation should take less than 30 minutes.

The installation EXEC is easy to use, but it is rather restricted in some of the functions that it provides. You should be aware that the EXEC does not format minidisks or create SFS subdirectories. The minidisks should be formatted or the SFS subdirectories created before the installation program is executed.

Figure 41 shows a typical configuration for a VM Web Server.

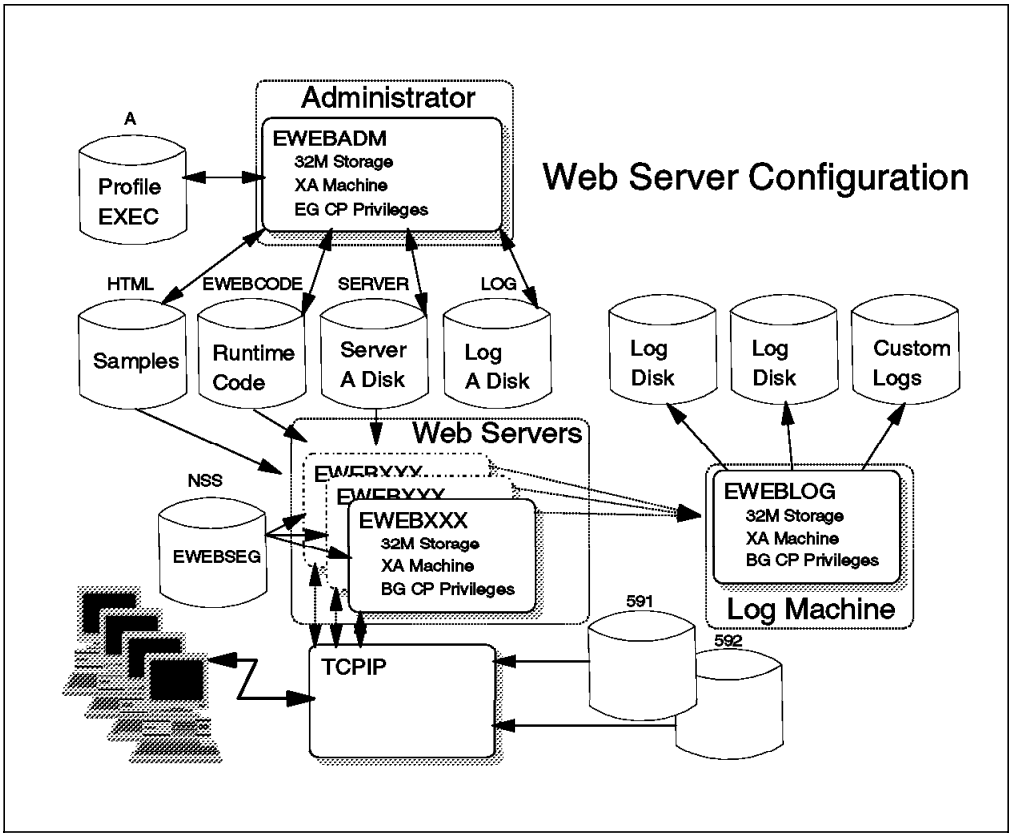


Figure 41. Web Server Configuration Overview

The diagram in Figure 41, is only a simple representation of what might be required in your installation.

The system can be run from either minidisks or the SFS, but it is recommended that you use the SFS to allow individual authorizations to files, a native directory structure, and a reduction in storage costs.

The recommendation for the number of Web servers you have on your system is discussed in the section entitled 5.6.4, “Number of Server Machines” on page 88.

5.6.1 User ID Preparation

The product is designed to have a minimum of two user IDs (without logging) or three (with logging). Depending on your requirements, you may define additional several servers.

EWEBADM Administration

EWEBXXX The server

EWEBLOG The logging system

Table 8 contains the user IDs and explains their purposes during the installation.

Table 8. User IDs and Their Function

User ID	Usage	Special Requirements
EWEBADM	Administer the System	This user ID owns all the source code and base files. This includes the A-disk for all the EWEBxxx server user IDs.
EWEBxxx	The Web server	This user ID (or multiple user IDs) share a common A-disk or SFS directory the owner of this disk or directory is EWEBADM. It needs CP privilege class B for security checking (diagnose 84 or A0). It should share EWEBADMs code and data because a user can access the system from any server and should have consistency. It also makes administering the system easier, as you only have to maintain one copy.
EWEBLOG, an optional user ID	Records logging information	This user ID records who has accessed the system.
Note: The files are loaded onto either the SFS or minidisks.		

Table 9 contains system resources required by EnterpriseWeb/VM.

Table 9. System User IDs

Resource	Name	Size	Further Information
VM user IDs	EWEBADM	32M	Installation Guide Chapter 1
	EWEBSRV	32M	Installation Guide Chapter 1
	EWEBDLOG	32M	Installation Guide Chapter 1
Saved Segment	EWEBSEG	1M	Installation Guide Chapter 1

The amount of disk space required to install and run the base system as supplied is contained in Table 10 on page 87.

Table 10. Hardware Requirements Table for VM/ESA							
User ID	**Addr**	**SFS Directory**	**Usage**	**3390 Size**	**3380 Size**	**FBA Size**	**SFS Size**
EWEBADM	191	EWEBADM	A-Disk	1	1	800	100
EWEBADM	193	RXSOCKET	REXX Sockets	3	4	4000	500
EWEBADM	194	HTML	HTML Code	10	9	12000	1500
EWEBADM	195	EWEBCODE	EWEB Code	14	12	5000	500
EWEBADM	291	SERVER	Server's A-disk	1	1	800	100
EWEBLOG	191	EWEBLOG	A-disk	1	1	800	100
EWEBLOG	191	LOGDATA	Log files	Varies	Varies	Varies	Varies

The disks that contain the executable code for EnterpriseWeb/VM and the sample code are listed in Table 11. All the screens that are used in the examples are generated from these disks.

Table 11. CMS Disk Access Order				
Mode	**Stat**	**Files**	**Vdev**	**Label/Directory**
A	R/W	14	191	WEB191
B	R/W	1	DIR	SFSLSY2:WEBADM2.SERVER
C	R/W	155	DIR	SFSLSY2:WEBADM2.EWEBCODE
D	R/W	92	DIR	SFSLSY2:WEBADM2.RXSOCKET
E	R/W	99	DIR	SFSLSY2:WEBADM2.HTML
G	R/O	588	592	TCP592
S	R/O	494	190	S-DISK
Y/S	R/O	2939	19E	Y-DISK

5.6.2 Installation Overview

The installation is simple and the following is a brief overview. Always refer to the instructions provided by Beyond Software Inc. The instructions in this section do not tailor the product, they only install the code.

1. Create the VM user IDs and format the minidisks or create the subdirectories in the SFS.
2. Log on to EWEBADM and load onto your A-disk the first file from the tape. This is the installation program named *EWSETUP*.
3. Run the installation program and follow the prompts.
4. Define and load the shared segment.
5. Update your TCP/IP configuration.

The installation is now complete, and you are ready to test.

5.6.3 Testing the Installed Product

Follow these steps to test that the product has loaded successfully:

1. LOGON EWEBSRV
2. At the CMS Ready prompt enter:

   ```
   EWEB PortNo CONFIG IVP CONFIG *
   ```

3. Access the Web server from a suitable Web browser on your PC.
4. Try the various functions. Some of the EnterpriseWeb/VM facilities are explained in the following sections.

5.6.4 Number of Server Machines

Because, using shared disks, the system cost of additional Web servers that remain idle is minimal, we suggest that you start with five servers and add or subtract depending on the load.

Tip

You have the right number of Web server if there is one server idle during your peak load.

Another point to consider is that you may have some very long running tasks or CGIs. The CGI would monopolize the server, thus preventing any other requests from being processed. Additional servers should be allowed for these.

To use more than one server, you need to set up read-only links to the following disks or SFS directories:

- Default A-disk, containing:
  - PROFILE EXEC
  - Default configuration file
- Default or base SFS disk containing security files and FILELISTs
- Data files
- Product code
- Normal CMS disks

TCP/IP has to have defined to it all of the servers' user IDs configured with the same port number. Giving the Web servers all the same port number within TCP/IP and also the same profiles and configuration files will ensure that they have the same consistency of access all the time.

5.6.5 Server Operation

The commands listed in Table 12 on page 89 are the EnterpriseWeb/VM commands. Some are issued to the Web server and some are issued from any user ID that has access to them.

Table 12. Configuration File Directives		
Command	**From**	**Usage**
EWEB	Server	Start the server code running. You can specify a specific port number and a different CONFIG file name as well as other options.
EWCOMP	Any	Compile HTACCESS or FILELIST files for encoding and performance benefits.
EWGET	Called from a CGI	To access variables from an HTML form.
EWSET	Called from a CGI	To place variables into an HTML form.

The product is designed so that EWEBADM will own all the EnterpriseWeb/VM's product disks and the servers will have access to these disks. A directive in the configuration file allows you to specify that you can refresh the minidisks that are accessed by the server in several different ways:

- Only when starting the server
- Every time that the server wants access to the disks
- After regular periods of time

5.7 Functional Overview

Figure 42 shows the EWEB application starting on port 82 and executing a EWEB CONFIG. Inside the config, the root FILELIST is identified, then referenced to satisfy the Web browser's request.

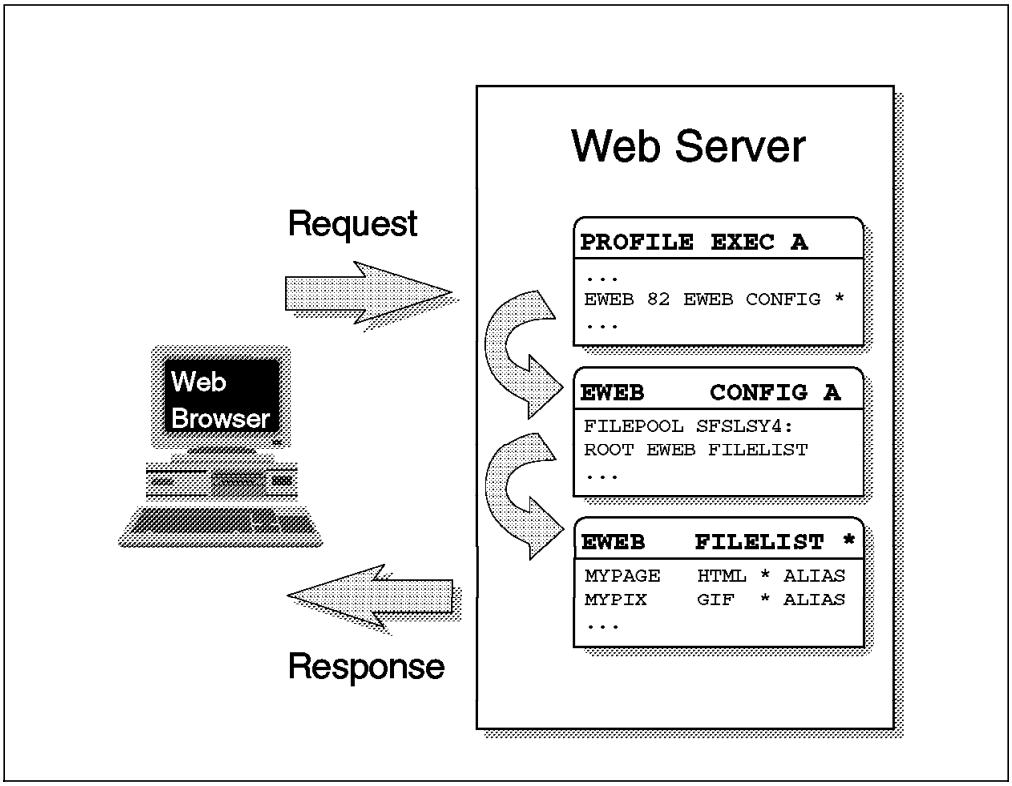


Figure 42. The Server File Usage Flow

When a browser connects to a Web server, it is presented with the screen that you select (the default) unless a different page is explicitly assigned and the client is given access.

To explain how the product works, some of the samples that are supplied with this product are discussed. A brief explanation is given before showing you the screen, which is then followed by the code that lies behind the screen and makes the screen work.

5.7.1 HTML File Walkthrough

An HTML file contains the code that sends the information that is required by your browser to generate the screen. For a more complete discussion on HTML, see 1.6.1, “Creating an HTML Document” on page 6.

Figure 43 and the HTML that created it follow:

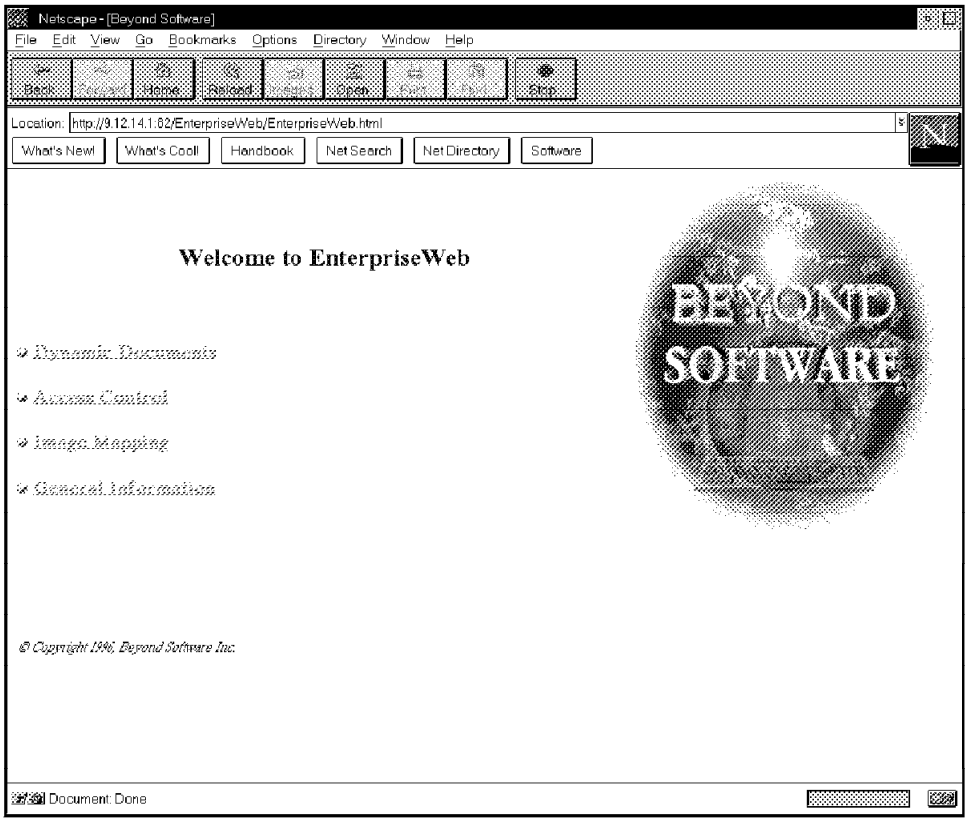


Figure 43. Entering Beyond Software Inc.'s EnterpriseWeb/VM

```
<!--  Copyright 1996 Beyond Software Inc. All rights reserved  -->

<html>                                                          1
<head><title>Beyond Software</title></head>

<body bgcolor="#FFFFFF" text="#001577" link="#1a00bb" vlink="#aaaaee">

<!-- BGSOUND src=/EnterpriseWeb/ewebsounds/beyond1.wav" loop=3 -->

<img ALIGN=right
     src="/EnterpriseWeb/ewebgifs/beyond.gif" width=306 height=365 ALT="">
                                                                2
```

```
<font size="+2"><b>
<br><br>
<center>
Welcome to EnterpriseWeb<br>                                    3
</center>
</font>

<font size="+1"><b>
<br><br><br>

<img src="/EnterpriseWeb/ewebgifs/wht_rnd.gif" ALT="-">         4
<a HREF="/EnterpriseWeb/demos/dynamic.html">
Dynamic Documents</a><br><br>                                   5

<img src="/EnterpriseWeb/ewebgifs/wht_rnd.gif" ALT="-">
<a HREF="/EnterpriseWeb/demos/ewaccess.html">
Access Control</a><br><br>

<img src="/EnterpriseWeb/ewebgifs/wht_rnd.gif" ALT="-">         6
<a HREF="/EnterpriseWeb/demos/imagemaps.html">
Image Mapping</a><br><br>

<img src="/EnterpriseWeb/ewebgifs/wht_rnd.gif" ALT="-">
<a HREF="/EnterpriseWeb/demos/">
General Information</a><br><br>

</b></font><p>
<br><br><br><br>
<font size="-1">
<cite>&#169; Copyright 1996, Beyond Software Inc.</cite>
</font>
</body>
</html>
```

1 The tag <html> must be the first element of the script.

2 The logo ID positioned and the GIF to be used is specified here.

3 The words “Welcome to EnterpriseWeb” are printed next, using this statement.

4 This statement places the small round ball (a dingbat) on the left side. When your terminal is unable to print the figure, then the ALT character is printed

5 <a> and define a hot spot where you can point and click. It contains the address to go to and also any static text to print on the screen.

6 Additional lines are defined to compose a selection list.

The next action taken is to move the cursor to the text **Image Mapping** and press the left mouse button. You are then presented with the next screen.

The browser interprets the request together with the HTML and tells the Web server it needs the information from the HREF containing /EnterpriseWeb/demos/imagemaps.html. This relates to the text **Image Mapping** displayed on your browser.

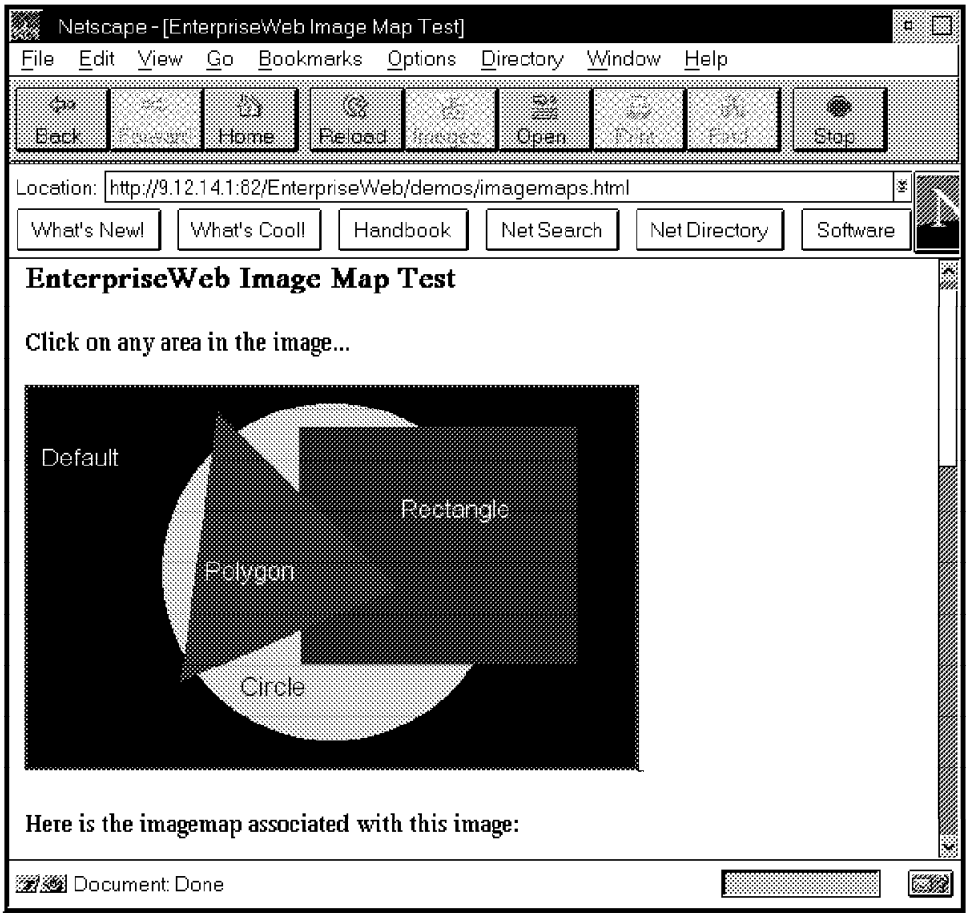


Figure 44. Imagemap Test Screen

Following is the code that created the page shown in Figure 44.

```
<html><head>
<title>EnterpriseWeb Image Map Test</title></head>
<body>
<h3>EnterpriseWeb Image Map Test</h3>
Click on any area in the image...
<p>
<a href="ewimdemo.map">
<img src="ewebgifs/ewimdemo.gif" ismap>

<!--For compatibility with other servers, you may
    choose to code your maps such as:

<a href="/htbin/imagemap;/EnterpriseWeb/demos/ewimdemo.map">
<img src="ewebgifs/ewimdemo.gif" ismap>
-->

</a>
<br>
<p>Here is the imagemap associated with this image:
<h3>EWIMDEMO MAP</h3>
<pre>
# Default -- if nothing else is selected
DEFAULT /ewebmain/ewebdemo/ewdeflt.html
#
# Polygon
```

```
POLY /ewebmain/ewebdemo/ewpoly.html 241,134 101,194 125,15
#
# Rectangle
RECT /ewebmain/ewebdemo/ewrect.html 179,26 361,181
#
# Circle
CIRCLE /ewebmain/ewebdemo/ewcirc.html 201,122 200,9
#
#  Decomment the next statements to test the POINT feature
#
## Point at lower right corner
#POINT /ewebmain/ewebdemo/ewlright.html 354,219
#
## Point at upper left corner
#POINT /ewebmain/ewebdemo/ewuleft.html 60,65
</pre>
<br>
<h3>Notes on Directives & Processing</h3>
<p>
The order which directives appear is significant. The input to the
imagemap routine is the x and y coordinates of the user click point.
This point is then tested against each shape (not including POINTs) in
the map in the order which they appear. The first shape that matches
causes the respective URL to be invoked.
<p>
If no shapes match then POINTs are checked. If there is a single POINT
entry, then it automatically matches. If there are multiple points, the
closest one matches. If no shapes match, then DEFAULT is checked. If it
is absent, an error is raised and the user will get a browser error
screen. If DEFAULT is present, then its URL is invoked.
<p>
DEFAULT is coded without a defined URL. Here, if a user
selects outside all defined regions, the browser will be told to "stay
put." This is useful if you are using a transparent GIF and want to
make the invisible background "none-selectable."
<p>
Consider a click at the point (239,133), close to the rightmost vertex
of the polygon. This particular points lies within all three shapes, but
because the polygon is defined first in the imagemap file, EWPOLY.HTML
gets called. If RECT came before POLY in the file, then the exact same
point would invoke EWRECT.HTML.
</body>
</html>
```

Note: The following code is called by the HTML code just described.

```
# Default -- if nothing else is selected
DEFAULT /ewebmain/ewebdemo/ewdeflt.html
#
# Polygon
POLY /ewebmain/ewebdemo/ewpoly.html 241,134 101,194 125,15
#
# Rectangle
RECT /ewebmain/ewebdemo/ewrect.html 179,26 361,181
#
# Circle
CIRCLE /ewebmain/ewebdemo/ewcirc.html 201,122 200,9
#
#  Decomment the next statements to test the POINT feature
#
```

```
# Point at lower right corner
#POINT /ewebmain/ewebdemo/ewlright.html 354,219
#
# Point at upper left corner
#POINT /ewebmain/ewebdemo/ewuleft.html 60,65
```

5.7.2 Common Gateway Interface

A common gateway interface (CGI) is simply a program that is invoked through a URL supplied by a Web browser and run on a Web server. CGIs can call other programs (written in any available language), interpret parameters, and work as any other program would.

You may name a CGI anything you wish, providing it follows standard CMS naming conventions and a few Internet rules.

You can invoke a CGI from several places; however, the most common are:

- A URL in the browser on your workstation
- An HREF attribute in an anchor tag in an HTML
- From another CGI or program running from the server

When you invoke a CGI from a URL, you can pass parameters. You must enter the full URL, including the CGI you wish to call, followed by a question mark (?) (this suggests that parameters follow), then the parameters separated by an ampersand (&).

The following example shows a CGI call with parameters. The parts of this URL are defined in the list that follows.

```
http://servername.co.uk/CGIname.CGI?Parm1&Parm2&Parm3
```

Parameter	Explanation
servername.co.UK	Server address
CGIname.CGI	Name of CGI to run
Parm1&Parm2&Parm3	Parameters to be passed to the CGI

5.8 Configuration

Before the initial setup of the system, you have to plan what is going to be available and who should have access to it. When you are planning your Web server, you should consider the following:

- What is the data you want people to see?
- Who do you want to see it?
- How many do you want to see it?
- Do you want them to see only part of it?
- Do you want people to be able to set up their own web pages?

When you have installed and tested the Web server, the next thing to do is to configure everything to your requirements.

There are several files that will require tailoring for your specific installation. The CONFIG FILE and the FILELISTs are two examples discussed in this section.

5.8.1 The CONFIG FILE

In the CONFIG file, you can tailor your system the way that you want to do it. You can balance your system resources to give the level of service you want to give to your customers. Table 13 lists some of the parameters which you can set in the CONFIG file, their default values, and their usage.

Table 13. Configuration Values and What They Do

Directive	Default	Usage
ACCESSINT	60	The interval between reaccessing minidisks.
ACCESSMODES	*	List of minidisks that will be reaccessed at ACCESSINT.
AUTOINDEX	OFF	Create an index automatically if no root or FILELIST is found for an accessed minidisk.
CGIUSERS		List of users who can write and execute CGIs from their own disks.
FASTDISKS		Lists file modes that are searched when a fastpath URL is specified.
FILEPOOL		Defines the servers default SFS file pool.
IDENT	OFF	Provides access to more remote identification information. It can have a detrimental effect on performance.
LOGPIPE		Controls whether logging is enabled or not.
PORT	80	Defines to which TCP/IP port EnterpriseWeb/VM listens. It may be overridden at startup time.
ROOT		Defines which is the first minidisk or SFS to use as the server root - where to start from.
SSI	OFF	Allows EnterpriseWeb/VM to process files that contain server-side include tags
USERWEBDISKS	191	Specifies the search order of minidisk addresses to be searched for EWEB or WEBSHARE FILELIST.
USERWEBLINKP	ALL	Controls the user that should be prompted for a minidisk password.
USERWEBS	OFF	Allows for access to other users' SFS directories.
VERBOSE	OFF	Determines if output debugging information is produced on the console.

5.8.2 FILELISTs

FILELISTs are most useful when operating off of minidisk, although they can be used when running from SFS. FILELESTs exist simply to overlay a virtual web-like, hierarchical file system, on top of CMS minidisk file system. FILELISTs perform functions like security and access to your system. They also act as an index to the files on your Web server that your users can access. The first file in the list is EWEBMAIN and this is the default file to access.

EWEBMAIN is a FILELIST. The first file in the FILELIST is EWEB HTML, which is picked up as the default.

Because FILELISTs are a core part of the Web server, they are discussed in detail in sections 5.9, “Security” and 5.10, “Accessing Other User ID′s Data (″userwebs″)” on page 99.

5.9 Security

Security within Beyond Software Inc.′s EnterpriseWeb/VM is accomplished in several different ways:

- No security
- VM security
- External security manager
- Internal EnterpriseWeb/VM security
- User defined by use of a supplied exit point

All these security measures are used individually or with each other, except for the no security option.

5.9.1 Overview of Security and Its Structure

EnterpriseWeb/VM security is based on the use of the FILELIST (refer to section 5.8.2, “FILELISTs” on page 95 for a description) file except when SFS or Fastpath is used. This section is split into

See Beyond Software′s *Installation Guide and Reference Manual* for further information. In this file you can also specify the ESM and groups of users, and restrict groups of users to certain areas.

General Security

Security is based around the FILELIST file. A FILELIST file is augmented with two additional files, namely HTACCESS and EWACCESS that has the same file name as the FILELIST.

The scope of a *FILELIST* is a particular minidisk or SFS subdirectory level and all lower level minidisk and SFSs, until another *FILELIST* with authorization is encountered.

The EWACCESS file is always used if it is present. It is a compiled file and therefore more secure. HTACCESS is the source file. These files contain information such as the name of the password file or the external security manager, the name of a file containing a group of files that is accessed, and IP addresses.

If the FILELIST file contains an AUTHORIZATION directive as the first command, this will override the HTACCESS or EWACCESS file. It can either point to a different ESM, a different access file, or a different group of files.

Special Considerations for SFS Directories

If an SFS is used, then EnterpriseWeb/VM only searches for a file called $EWEB (EWACCESS or HTACCESS) in a particular directory. This restriction does not apply if an AUTHORIZATION directive exists in the FILELIST file used.

FASTPATH Security Access

FASTPATH is a directive within the EnterpriseWeb/VM CONFIG file. It allows anyone to access a default SFS directory or minidisk without the checking of the FILELIST or *xx*ACCESS files.

Access through FASTPATH requires that security files reside on the same minidisk or SFS directory as the files which are to be accessed. The security file must be called $EWEB (HTACCESS or EWACCESS).

5.9.2 Examples of the Security Access Paths

This section contains a few examples with explanations to make the security within EnterpriseWeb/VM clear. There are a few general rules that you should be aware of, namely:

- SFS and minidisks are treated differently.
- The FILELIST always takes precedence for security.
- If a file name exists for both EWACCESS and HTACCESS, EWACCESS takes priority.

Figure 45 shows different scenarios that will be documented in Table 14 on page 98. The files contained in the scenarios are files that are accessible by the server.

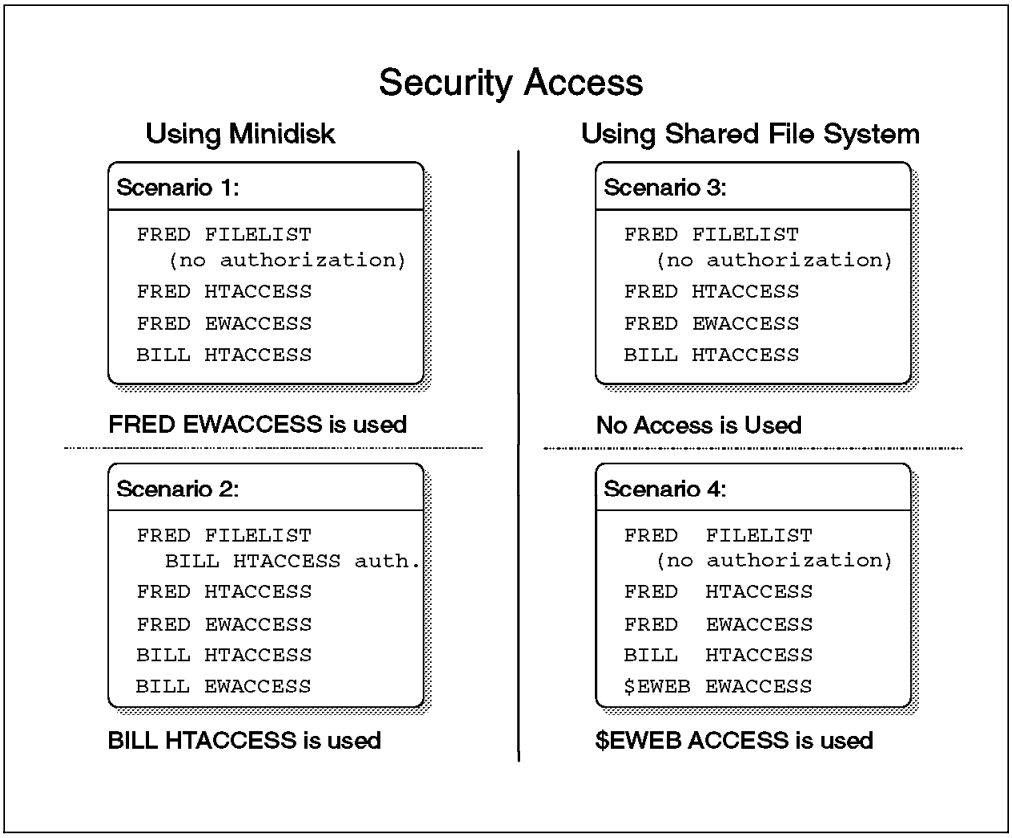


Figure 45. Security Within EnterpriseWeb/VM

Table 14. Internal Security Priorities	
Reference	**Description**
Scenario 1	In this example, security is based on the contents of the file FRED EWACCESS. The file FRED FILELIST has no authorization directive but a security file FRED EWACCESS does; this is therefore used for security.
Scenario 2	In this example, security is based on the contents of the file BILL HTACCESS. The file FRED FILELIST has an authorization directive for BILL HTACCESS, and even though a FRED EWACCESS (to match the FILELIST name) and a BILL EWACCESS (EWACCESS takes precedence over HTACCESS) exist, the FILELIST authorization has precedence.
Scenario 3	No security access is used because the SFS is being used. FRED FILELIST has no authorization included and no $EWEB *xx*ACCESS file exists. On the SFS, no notice is taken of any files with the same file name as the FILELIST file.
Scenario 4	In this example, security is based on the contents of $EWEB EWACCESS. FRED FILELIST has no authorization directive. However, for the SFS, security is taken from this file, because a $EWEB XXACCESS file exists.

FASTPATH Security Access

FASTPATH security access requires additional scenarios, as shown in Figure 46 and discussed in Table 15.

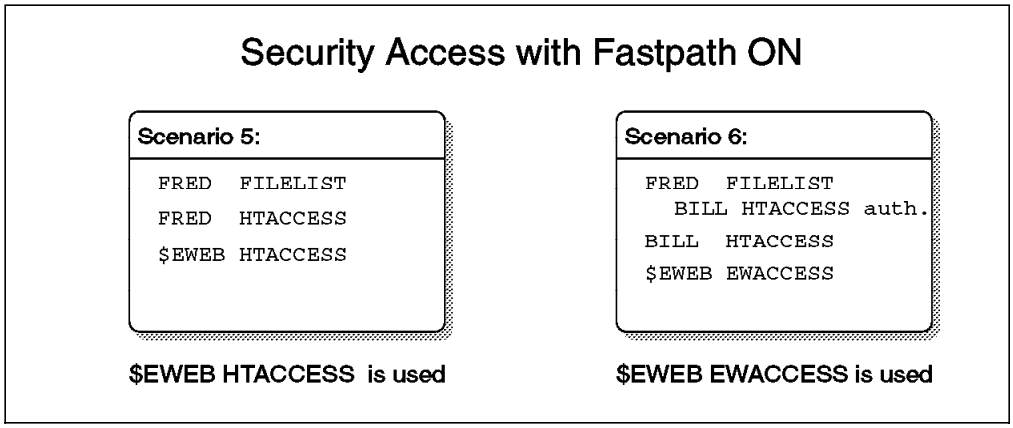


Figure 46. More Security Within EnterpriseWeb/VM

Table 15. Internal Security with Fastpath	
Reference	**Description**
Scenario 5	There is a FILELIST, but it has no authorization included. The $EWEB HTACCESS is used as it would be in any other case.
Scenario 6	Even though there is an authorization in the FILELIST, $EWEB EWACCESS is used because fastpath is being used.

5.9.3 VM′s Security

It is possible to use the CP directory to provide user ID and password access to data. This kind of authentication should be limited to intranet Web services. Providing a password over a public unencrypted line could allow someone to gain access to your user ID through Telnet or FTP.

The Web server′s user ID should be well protected.

5.9.4 External Security Managers

You can use a proprietary security manager such as RACF or VM:Secure to handle the security for you. EnterpriseWeb/VM makes provisions to use external security managers.

5.9.5 Special Security Implications for CGIs

CGIs are very powerful and can lead to security exposures if they are not carefully planned and executed.

CGIs run in the Web server, and as a result, have access to all the files and privileges that the server itself has. An example of this might be that a general user is not allowed to see confidential data, but privileged users also use the same Web server, and the data is therefore available to the server. If the user is allowed to write and execute a CGI on that server, that user will also have access to the confidential data.

Another example of users having access to privileges that they should not have access to is when the server can issue privileged CP and CMS commands. This would allow CGIs written by unknowing or malicious users to do damage to not only the Web server but to the complete system.

5.10 Accessing Other User ID′s Data (″userwebs″)

It is possible to access any VM user ID′s data if the security allows the data to be accessed. The system should be set up to protect against unauthorized access; however, it is ultimately up to the owner of the data to provide adequate protection.

To access a VM user ID′s data, the user must have either a subdirectory named EWEB or have an EWEB FILELIST in the base directory. If AUTOINDEX is ON, the user′s files will be presented even if neither of the previous conditions are true.

It is the Web server, not the user ID, who owns, accesses, and reads the user data. The Web server must be authorized to the data.

Minidisks can be accessed in a similar way; however, there are some differences. One of the main differences is that when a specific minidisk is accessed and there is no automatic linkage to any other minidisks, they must be specifically accessed in the server.

An SFS subdirectory, if the security allows it, is accessed when required, not before.

Figure 47 on page 100 lists the files that are contained in the SFSLSY4:LEESS.′s SFS root directory. You will notice that a file named EWEB FILELIST exists. This is the file that controls access to the files that can be looked at from a browser

(indirectly through the server). Be aware that not all files in the user's directory structure can be accessed, only those that are specified in the FILELIST file.

It is not recommended to mix FILELISTs with SFS directories, though it is possible. There is another method of file access particular to the SFS.

As well as being able to control access from a FILELIST on a 191 (or another minidisk if specified by USERWEBDISKS) or root SFS directory, the system looks for an SFS directory called EWEB. It looks for this directory automatically. If this directory is found, then all files in this directory will be available to the browser. This causes a search for a file in the ROOT SFS of the user whose data you are accessing, named EWEB FILELIST (refer to Figure 48 for the contents of this file). By the types of files listed, you can see that LEESS uses this directory accessed as an A-disk. With incorrect authorizations or planning, private or confidential data may be exposed to unauthorized users.

```
EWEB     FILELIST A1 V         70         10          1 11/14/96 11:04:10
FILELIST FILE     A0 V        107         23          1 11/13/96 11:19:43
SPECIAL  FILE     A1 F         80          5          1 11/13/96 10:31:26
LISTFILE LIST     A0 V        107         25          1 11/13/96  9:05:40
SUBDIR            A  DIR       -          -          - 11/12/96 10:21:58
FRED              A  DIR       -          -          - 11/12/96  9:32:13
COMMANDS FILE     A1 F         80         32          1 11/06/96 11:07:25
TABLE    SCRIPT   A1 V         43         12          1 11/06/96  9:27:01
DDISK    LIST     A0 V        107        155          5 11/05/96  9:12:31
ACCESS   LIST     A1 F        132         13          1 11/05/96  8:08:48
CDISK    LIST     A0 V        107        155          5 11/05/96  8:07:59
EWEBCODE FILE     A0 V        107        155          5 11/05/96  8:07:59
LEESS    SYNONYM  A1 F         80          1          1 11/04/96  8:45:47
PROFILE  EXEC     A1 V         78        125          2 11/01/96 11:19:18
LEESS    NOTEBOOK A0 V         73         30          1 10/31/96 10:22:04
WEB5HARE          A  DIR       -          -          - 10/30/96 11:56:01
```

Figure 47. List of the Files in the SFSLSY4:LEESS. SFS Directory

If a file named EWEB FILELIST cannot be found in the SFS default file pool then it will look for an EWEB FILELIST on the DEFAULT minidisks, as specified in the server's configuration file USERWEBDISKS directive. A tilde (~) is always required when accessing a user's file space.

A file will not be found if it is not in an accessed directory or path, even though it exists on the SFS or on a minidisk on that user.

The format of the FILELIST is shown in Figure 48 and how it is used can be seen using Figure 49 on page 101 and Figure 50 on page 102.

```
00001 ************************************************************************
00002 *Filename Filetype  Filemode   Alias       Displayed Name
00003 ************************************************************************
00004
00005  MY       Trace      *        Traces      "All Trace Files"
00006  LISTFILE LIST       *        ListFiles   "Listing Files on Leess"
00007  SUBDIR   *          *        SUBDIRECT   "A File in a Sub Directory"
00008  Inform   File       *        Inform_File "File INFORM File in above"
00009  Filelist File       *        Flist       "Filelist File in above"
```

Figure 48. EWEB FILELIST File

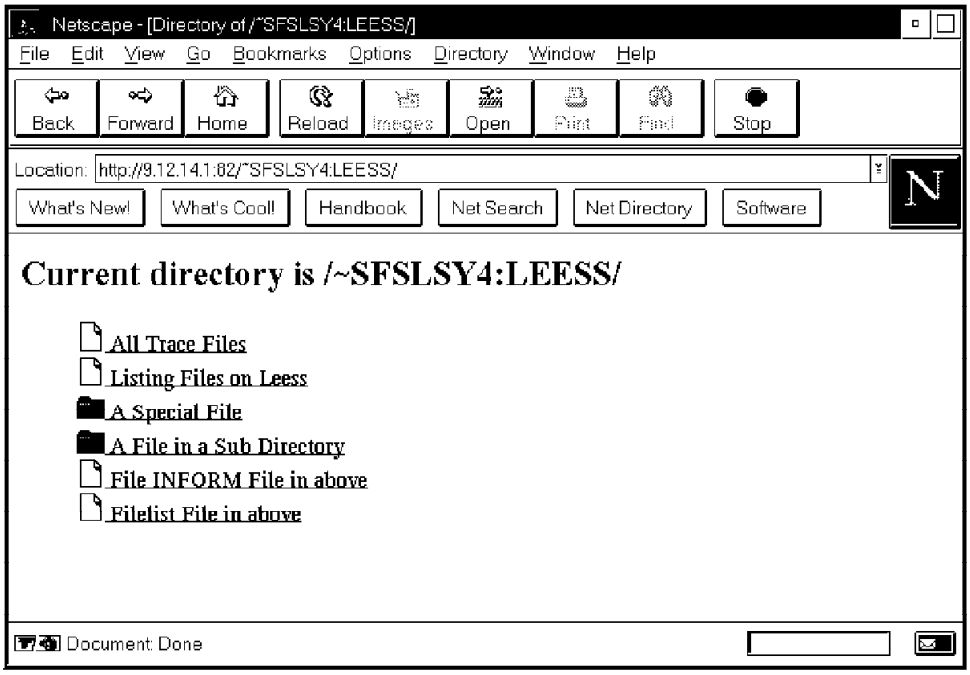


Figure 49. Display of Contents of the First Screen

The structure of a FILELIST can be explained as follows:

- You must specify a file name.
- You cannot use an asterisk to complete a partial file type.
- The file description (Displayed Name) must be surrounded by double quotes.
- If only the file name is specified, then a search is made to determine if the accessed modes for any file or directory match the parameters.
- If the fully qualified name is present, then the file is searched for.
- The alias is used by the browser in the location field and is sent to the server as part of the address. The server translates this to the search argument.
- The displayed name is the name that is sent back to the browser and should have a significant description to the user.

The panel presented in Figure 49 shows the six choices that are available. For this example, **A File in a Sub Directory** is selected. Figure 50 on page 102 is the panel that results from this action.

Figure 50. Display of SUBDIRECT Screen

The screen that is sent back to the browser contains a list of files that can be viewed.

Some points that should be noted from the previous figures are:

- The EWEB file in this list is not the base EWEB FILELIST that was shown in Figure 48 on page 100. This EWEB FILELIST exists to allow searches further down the subdirectory chain.
- The name displayed on the browser is taken from the alias name within the EWEB FILELIST file.
- Using FILELISTs, you have several paths that you can use to get to the file. The file structure is virtual.

The **inform.file** is selected and is displayed on the screen. The output of this can be seen in Figure 51 on page 103. If the full location address is typed in on the initial web browser screen, then the result would be to come straight here.

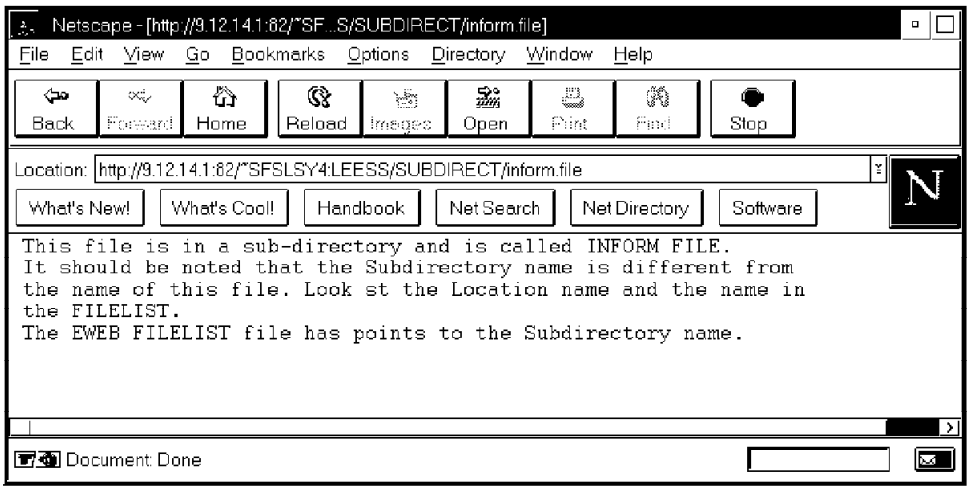


Figure 51. The Contents of the Inform File

5.11 Logging

If you want to record information about who is accessing the Web site and what they are doing, EnterpriseWeb/VM provides the facilities to do this. The EnterpriseWeb/VM provides you with the options to record basic NCSA format records, to record some additional information, or even to customize information and record it in the log.

The Web server sends messages to the log server and they are written to a file on a minidisk or in an SFS directory to which the log server has write access.

5.11.1 Setup

The log server name can be any valid user ID name you choose.

The Web server configuration file accepts parameters that dictate whether to send the standard NCSA or an Extended form of the NCSA record. The PIPE command that sends the data to the log server is also included in the configuration file and should be changed to the logserver's user ID. This PIPE command can also be used to change the format of the record.

The log server also has a configuration file that can be assigned any valid CMS file name.

Following are a few of the options that can be set up in the CONFIG file:

CONSUSER The user that the log server will send any console messages to.

LOGFORMAT Indicates the type of log record being used.

LOGPOOL Points to the file containing the list of minidisks to be used for logs.

Note: There are many more options available. It is possible for you to do other tasks as well, including set up and customize log servers, set up lists of authorized users, define the time to run cleanup procedures, and many other options.

5.11.2 Controlling the Logserver

Commands to control the logserver are divided into two sets:

Immediate These commands are executed directly on the server.

Server These are issued to the logserver using the immediate command *COMMAND*.

Note: These commands are issued directly from the log server console or through an authorized user (from the AUTHUSER directive in the configuration file).

The raw data is available from the logging process, but no report programs are provided with this package. There are freeware, shareware and commercial webserver report programs available from a multitude of vendors to help you with the report.

5.11.3 Controlling the Log Files

The log files are held on the SFS, minidisks, or both. If you have the log files on the SFS, then you should have a logfile setup on a minidisk as well. Minidisks have a fixed size and when a minidisk becomes full, you can swap to another minidisk; however, in the SFS, you cannot specify a maximum size for a subdirectory, because it uses all available space until the directory becomes full. To allow you to swap logs in a file full situation, you must therefore have a minidisk to be used.

5.12 Problem Determination

EnterpriseWeb/VM is based on the traditional functions and strengths of VM, so if you have a problem you can use the normal diagnostic facilities of CMS, REXX, TRACE, PER TRACE, dumps, and spooled console output, to name a few. There are a few console options that will further assist your debugging.

The command DEBUG, which is also a configuration directive, can turn on or off various tracing options including low-level SOCKET traces. The command VERBOSE or ECHO is input from the console to get diagnostic information on the console. The logged data appears on the console only if you have a CONSOLE stage in the LOGPIPE setting. With VERBOSE set ON, you will see more detailed information. Figure 52 on page 105 shows a verbose console.

```
E-WEB: SOCKET READ: READ 1 WRITE EXCEPTION
EWBEGIN: Begin 2 at 19:04:38 client 9.12.14.5
EWFLIST:   EWEB *FL F
EWFLIST: Looking in directory: EWEB *FL F
EWLOAD: SUBDIR   *           *        SUBDIRECT   "A File in a Sub Directory"
EWLOAD: Path translated: SUBDIR * F
EWFLIST: SUBDIR * F
EWFLIST: Looking in directory: SUBDIR * F
EWOUTPUT: Begins with: Content-Type: text/html - -
HTTP/1.0 200 OK
Content-Type: text/html
Server: EnterpriseWeb/1.1.1 VM_ESA/2.2.9507 CMS/11.507 REXX/4.00 CMS_Pipelines/
 .0202 REXX_SOCKETS/2.01
MIME-Version: 1.0

EWOUTPUT: Content-Type: text/html - -
EWMENU: Complete.
EWOUTPUT: complete. 0
EWBEGIN: Closed 2 at 19:04:38
@EXT 9.12.14.5 - - Y04/Jan/1997:19:04:38 -5.00" "GET /sfslsy4:leess/subdirect/
HTTP/1.0" 200 - "" "Mozilla/2.02E (OS/2; I)" @#E-WEB#@ 19970104 68678 0.040806
E-WEB Ready;
```

Figure 52. Example of a Trace with VERBOSE

5.13 New Features

This product is continuously receiving enhancements. During the project that created this publication, a GUI facility to manage a Web site from a Web browser was made available; however, because of time limitations, it was not possible to fully document the facility. The results of our limited access to this enhancement are discussed in section 5.13.2, "Remote Configuration" on page 107.

5.13.1 OfficeVision Connection

At the time of this publication, a new Office Vision connection was being developed. This will allow access to Office Vision functions, such as calendar, from a Web browser. Figure 53 on page 106 shows a sample screen from this facility.

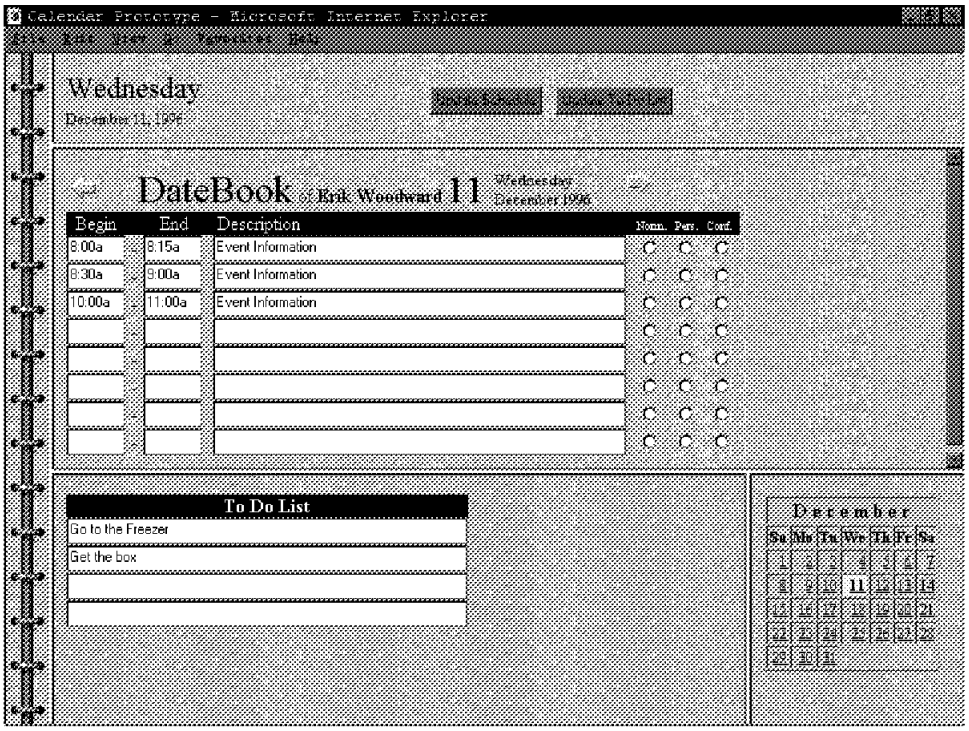


Figure 53. EnterpriseWeb/VM Calendar Feature

The EnterpiseWeb OfficeVision Calendar add-on provides end-users with complete access to OV calendars from any HTML compliant browser. All calendar functions are performed centrally by one or more EWEB servers using the published calendar APIs. In order for a user's calendar to be available on the Web, the user must explicitly grant all Eweb servers the authority to change the access to the user's calendar. Once granted, full access control can then be further administered via the interface.

All users are authenticated against the CP directory before performing calendar activities. Once validated, the calendar temporarily assumes any privileges granted to the requester by the target calendar owner. Consequently, the calendar server itself never "sees" anything more than the requesting user would have had the request been made by the user at a real 3270.

Calendar events may be added, modified, or deleted. Calendar notes are also supported. A month-at-a-glance view is provided that essentially mimics the display provided by OV. Daily views include a mini calendar that provides quick point and click navigation to a new day.

Other features include full nickname support and cross-system calendar access. The starting time on the month view is configurable at run-time by the user. In support of cross-system access, a time adjustment field allows for dynamic changes in all displayed times.

Note: The EnterpriseWeb OfficeVision/VM Calendar feature was not available at the time this Redbook was generated. The above paragraphs were added as it was assumed that at availability time of this book the feature is released and available.

5.13.2 Remote Configuration

This new feature allows EnterpriseWeb/VM to be maintained over the WWW. Figure 54 shows a sample of this feature.

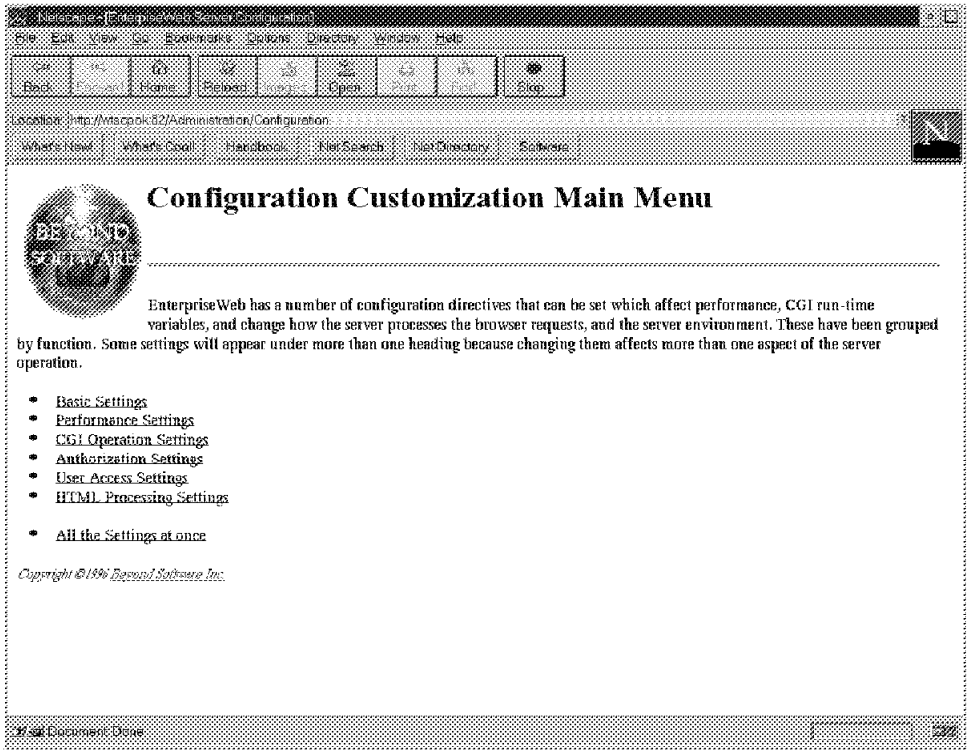


Figure 54. The Remote Configuration Feature

The administration feature was not explored beyond displaying some of the screens; however, the following information should allow you to at least display the screens.

1. Remove the asterisk (*) from the EWEB FILELIST ADMIN entry.
2. Reload the Installation Verification Home Page.
3. Invoke the configuration utility by appending the directory name `/Administration/` to the root URL.
4. The USER is **eweb** in lower case.
5. The Password is **administrator**.
6. Explore the functions.

Other enhancements are being developed for this product. If you require any further information, either look at Beyond Software Inc.'s WWW home page (`http://www.beyond-software.com/`) or contact the company directly.

5.14 Summary of Features

Beyond Software Inc.'s EnterpriseWeb/VM has evolved substantially from its base, WEBSHARE. This section contains a summary of some of the features and observations that were obtained during the installation and testing of this product.

The EnterpriseWeb/VM is built on the existing and proven strengths for which VM has become noted over many years. It is built using REXX and CMS Pipelines together with TCP/IP and HTML.

Compiled REXX and CMS Pipelines, together with some re-engineering of WEBSHARE, now provide the speed that the Web servers need. These enhancements together with TCP/IP′s multiple server per port capability make EnterpriseWeb/VM a substantial product in the Web server market.

EnterpriseWeb/VM delivers any MIME type including JAVA.

One of the nice facilities of EnterpriseWeb/VM is in allowing users (if they are authorized to do so) to create their own home pages and allow access to them. This is possible though the use of EnterpriseWeb/VM′s own hierarchical data structure. The file structure allows accesses to data and applications to be controlled by the user.

EnterpriseWeb/VM provides the facility for accesses to the server to be logged in the CERN/NCSA log format. As an extension to this facility, it also allows you to write log records in a format that you want. These records can be recorded in different log files.

Multiple logs are provided so that logs can be cycled easily and then cleared automatically at a time you specify. You can then use the logs to produce your own reports.

EnterpriseWeb/VM allows many different types of data to be stored and retrieved, and HTTP headers to be automatically included for both HTML and non-HTML documents.

Imagemaps, both externally and internally generated, are supported as is server-side include. You can also invoke alternate routines when specific formats are not recognized.

CGI scripts are supported and interfaces are provided to read input parameters from your browser and call other programs. CGI scripts are written in REXX and CMS Pipelines and can be compiled.

Security is extensive. EnterpriseWeb/VM relies on both security within the product and security provided by ESMs and home written routines.

As already mentioned in section 5.13.2, “Remote Configuration” on page 107, GUI support is in the product but is not documented at the time of publication.

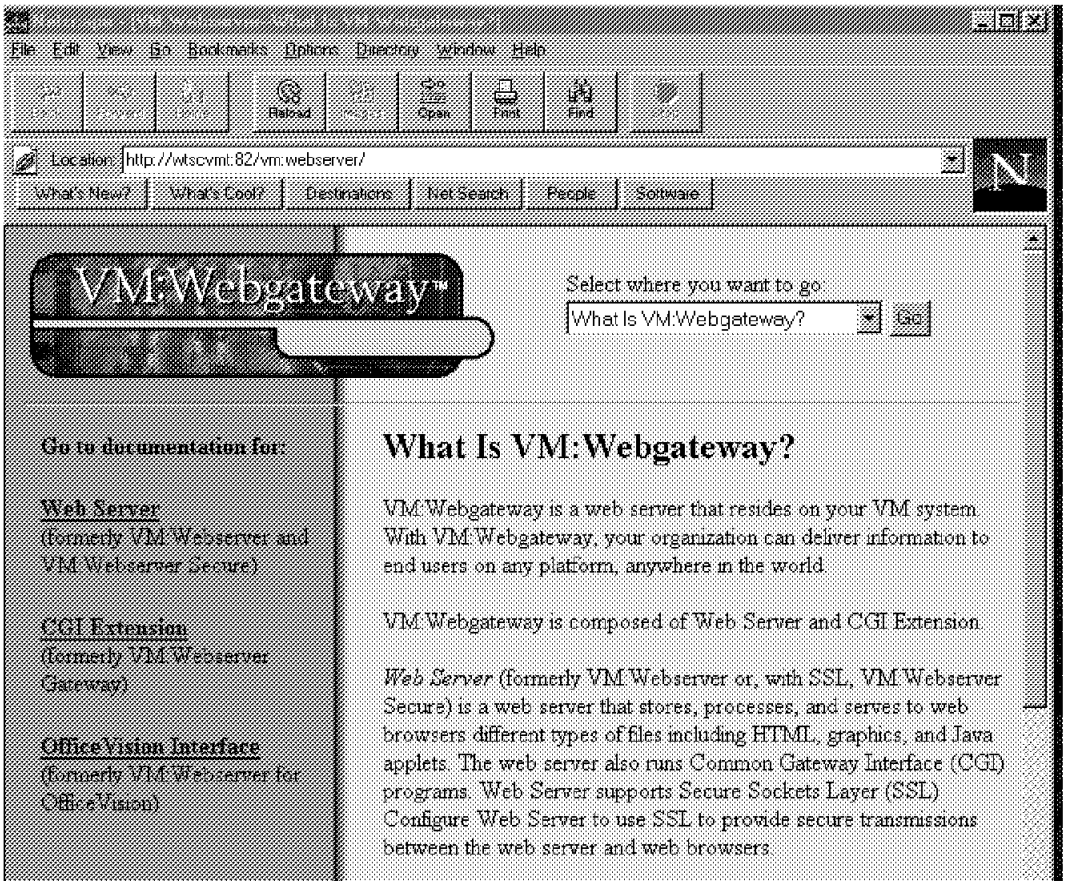
Chapter 6. VM:Webgateway

This chapter discusses the VM:Webgateway Release 2.2 and VM:Webserver OfficeVision Interface products. VM:Webgateway is a cost-effective, robust, high-performance, second generation World Wide Web Server for your VM/ESA operating system. VM:Webgateway and VM:Webserver OfficeVision Interface are products of Sterling Software, Inc.′s VM Software Division.

6.1 Introduction

VM:Webgateway is a VM implementation of an Hypertext Transfer Protocol (HTTP) server. With VM:Webgateway running on an existing VM/ESA system, a company can use the World Wide Web to modernize and disseminate data and applications while maintaining centralized control. There is no need to purchase new technology, thereby, allowing a company to be more competitive. Figure 55 shows the VM:Webgateway home page.

Figure 55. VM:Webgateway Home Page. This page, http://hostname:port/VM:Webserver/, is the root of the VM:Webgateway online documentation URL tree.



VM:Webgateway supports the SFS and BFS file systems, CMS minidisks, and data in the server′s CMS search order (a combination of ACCESSed SFS directories and CMS minidisks) for storing the data it is serving to the Web. The use of SFS or BFS can further cut costs in the storage media required to

maintain the system. In addition to serving data from within the server's own domain, you can configure VM:Webgateway to serve user pages from an individual's CMS user ID.

This chapter discusses various aspects of the product, from installation and configuration to user experiences while evaluating the product.

6.2 About Sterling Software, Inc.

Sterling Software is a leading provider of software and services for the applications development, information management, systems management and federal systems markets. The company is ranked among Business Week's 1998 "Info Tech 100" as one of the world's best performing information technology companies. Headquartered in Dallas, Sterling Software has a worldwide installed base of more than 20,000 customer sites and 3,600 employees in 90 offices worldwide.

Sterling Software's VM Software Division provides comprehensive solutions for the VM operating environment. The VM:Webgateway family of Internet and intranet products enables users to provide Web access to legacy data and applications. The VM:DB family of products offers extensive solutions for managing DB2 databases and developing database applications. The VM:Manager family is an integrated VM systems management solution for storage management, automated operations, service level management, security, recovery, and Shared File System management. VM Services provide a full complement of offerings to assist organizations throughout their VM lifecycle. Services include Year 2000 Assessment and Testing, VM Consulting, Software Implementation and Deployment, and Education and Training. The VM Software Divisions superlative customer support is available 24 hours-a-day, seven days-a-week.

Sterling Software, Inc.'s corporate offices are located at:

Sterling Software, Inc. Corporate Headquarters
300 Crescent Court
Suite 1200
Dallas, TX 75201, USA

(214) 981-1000
(214) 981-1255 (fax)

For VM product information, contact the following:

Sterling Software, Inc.
VM Software Division
PRODUCT INFORMATION
1800 Alexander Bell Drive
Reston, VA 20191, USA

(800) 533-5128 North America
(703) 264-8000 North America
61-2-99-75-4777 Pacific Rim
972-527-5255 Latin America
33.1.53.93.2200 Europe and all other areas
(703) 264-0840 (fax)

Sterling Software, Inc.'s VM World Wide Web site is powered by VM:Webgateway under VM/ESA on an IBM S/390 processor to provide its Internet and intranet services. Sterling's VM WWW site can be found at:

```
http://www.vm.sterling.com/
```

Sterling Software, Inc. on the World Wide Web is found at:

```
http://www.sterling.com/
```

From this site, you can gather information on the company, including financial information, company news, product information, contact information, and employment opportunities.

Following is a list of additional products developed by Sterling Software, Inc. for the VM/ESA platform:

- Integrated VM systems management products for storage management, automated operations, service level management, security, recovery, and Shared File System management:

  VM:Account Accounting, costing, and reporting tool.

  VM:Archiver Allows users to archive CMS files to less expensive storage media.

  VM:Backup A full-function product for backing up and restoring VM systems.

  VM:Batch Allows interactive tasks to be run while running resource-intensive processes as batch jobs.

  VM:Director Administration of the VM directory and SFS, plus automated DASD management.

  VM:Manager A suite of systems management solutions that address automated operations, service level management, security, recovery, and storage management.

  VM:Migrate Implements system-managed storage for VM.

  VM:Monitor Monitors system performance.

  VM:Notekeeper Automatically archives OfficeVision/VM and PROFS notes.

  VM:Operator Automates VM and CSE operations and application testing.

  VM:Schedule Schedule work during off-hours on a date and time or event basis.

  VM:Secure Integrates resource access control with directory and SFS management to provide a comprehensive security management system for the VM environment.

  VM:Sort Sorts and merges data in CMS files.

  VM:Spool Monitors spool-space utilization.

  VM:Tape A complete tape management systems.

- Products for managing DB2 databases and developing database applications:

DB/SQLMAP	Analyzes systematic and regular Explains of applications and queries critical to any DB2 for VM tuning strategy.
VM:DBA	An integrated database administration solution that unites all important aspects of DB2/VM management under your control.
VM:DB/Admin	Comprehensive central administrative platform for DB2 for VM.
VM:DB/Monitor	Comprehensive facility for monitoring and reporting on the performance of DB2 for VM.
VM:DB/Reorganizer	Powerful utility for maintaining objects and data stored in DB2 for VM databases.
VM:DB/Restore	Archive management utility that allows database administrators (DBAs) to quickly and easily restore individual tables from DB2 database archives, and VM:Backup user database archives on VM.
VM:DB/Developer	Suite of tools that provides complete application development and management in DB2 for VM.
VM:DB/Editor	Sophisticated report writer for DB2 for VM.
VM:DB/Reporter	Full-screen utility that allows you to easily update and validate data stored in DB2 tables and views on VM.
VM:DB/REXX	Provides applications programmers a quick, efficient, and cost effective interface to DB2 for VM.

Following is a list of products marketed by Sterling Software, Inc. for the VM/ESA platform:

VM:Cadback Backs up and restores computer-aided design (CAD) models from CADAM and CATIA.

Vital Signs for VM/ESA
Provides comprehensive performance monitoring through real-time displays and historical data reporting.

MasterView for Vital Signs for VM/ESA
Windows-based tool that retrieves information from the Vital Signs performance database and provides graphical displays of the information.

6.3 Obtaining VM:Webgateway

To obtain the VM:Webgateway product, contact:

Sterling Software, Inc.
VM Software Division
1800 Alexander Bell Drive
Reston, VA 20191, USA
(703) 264-8000

A complete list of world-wide offices is located on the Sterling Software, Inc. web site.

6.4 What Comes in the Box for VM:Webgateway

The package contains the following:

- Installation tapes (2)
- Maintenance tape
- Documentation for VM:Webgateway's install process
- *VM:Webgateway Getting Started* manual
- *VM:Webgateway Tutorial* manual
- *VM:Webgateway Quick Reference* manual
- *VM:Manager Installation Guide* manual
- *VM:Manager Reference Manual*
- *VM Software Documentation Library* CD

6.4.1 VM:Webgateway Documentation

VM:Webgateway documentation consists of a comprehensive *Getting Started* guide, an introductory *VM:Webgateway Tutorial* manual introducing the VM:Webserver CGI Extension programming environment, a *VM:Webgateway Quick Reference* manual, and full online documentation. The online documentation contains pages suitable for online use and pages suitable for printing all information on a topic area. The online documentation is discussed in "VM:Webgateway Online Documentation."

VM:Webgateway Getting Started

The *VM:Webgateway Getting Started* guide provides the following:

- Summary of changes in the product family since the last release
- Quick overview of the VM:Webgateway product
- Step-by-step procedure on how to install or upgrade the product family using the Automated Installation and Maintenance (AIM) process
- Required updates for the Transmission Control Protocol/Internet Protocol (TCP/IP) service virtual machine (SVM)
- Installation verification
- Conversion from WEBSHARE (if required)
- Possible updates needed on the VM:Webgateway SVM user ID
- The proper way to start and stop the server
- Summary of product static configuration file records, including suggested formula for setting of internal database cache sizes
- Information on CPU identification code (CPUID) software licensing verification values
- Summary of process to apply source and zap-based corrective maintenance
- Sample files

VM:Webgateway Online Documentation

Once the product is installed and working properly, you are able to access documentation online. There are CMS help files for the VM system operator commands in the product. In addition, all end user, VM:Webgateway system administrator, and VM:Webgateway system operator commands and general product information is available as online documentation, which is accessible using your browser. The online documentation contains pages suitable for online use and pages suitable for printing all information on a topic area. To access the online documentation, use browser and the URL listed in Figure 56 on page 114.

```
http://hostname:port/vm:webserver/
```

Figure 56. Accessing VM:Webgateway Online Documentation

The value *hostname* is the name of your host system. The value *port* is the TCP port that VM:Webgateway uses to listen for incoming communications. For instance, for the host WWW.VM.Sterling.Com and the default of port 80 for the http protocol, you would use the link: `http://WWW.VM.Sterling.Com/vm:webserver/`

The online documentation is also accessible from any of the error panels that may be displayed. There is a hypertext link at the top that will take you to the menu screen for the VM:Webgateway product.

The online documentation describes all the features of the product and was our primary source of information for this product. By selecting the URL shown in Figure 56, you are presented with the page shown in Figure 55 on page 109. From there you may select from the various items listed. Each of these pages leads to other pages with more help. For instance, the Web Server link leads to a page including selections for:

- Using Online Documentation
- What's New in VM:Webserver?
- Product Overview
- System Administrator Tasks
- Operator Tasks
- User Tasks (actually tasks for Information Providers)
- Reference
- Glossary
- Index

In addition, the documentation pages have pop-up selections for information such as:

- Commands
- Configuring VM:Webserver
- Glossary
- Index
- Messages & Abends
- Operator Tasks
- Printing Online Documentation
- Reference
- SSL Tasks
- System Administrator Tasks
- Table of Contents
- User Tasks (actually tasks for Information Providers)
- Using Online Documentation
- What's New in VM:Webserver
- What's VM:Webgateway

Similar links exist on the VM:Webserver CGI Extension and VM:Webserver OfficeVision Interface pages, served by links on this initial page.

The online index allows you to search for the item and quickly go to that page and get the information you need.

Although the online documentation covers all the major topics you might have questions on, you may need more documentation on items that are less likely to be implemented. One such item is the setup for multiple servers, which is described further in topic 6.8.2, “Multiple VM:Webgateway Servers” on page 131 in this document.

6.4.2 VM:Webgateway Sample Code

Part of the package shipped are two VM:Webgateway demonstration applications.

Basic Introductory VM:Webgateway Sample Code

The first demonstrates elementary HTML and CGI programming constructs. It is contained in the “Software Corrections” section of the distribution tape. The procedures for loading these materials are described in the *VM:Webgateway Getting Started* guide under the corrective service application section. The application consists of sample Hypertext Markup Language (HTML) pages, Graphics Interchange Format (GIF) files (some are animated), Joint Photographic Experts Group (JPEG) files, and sound files. These make up typical Web pages that could be developed for any company that decides to publish on the World Wide Web. Figure 57 shows the beginning of the demonstration.

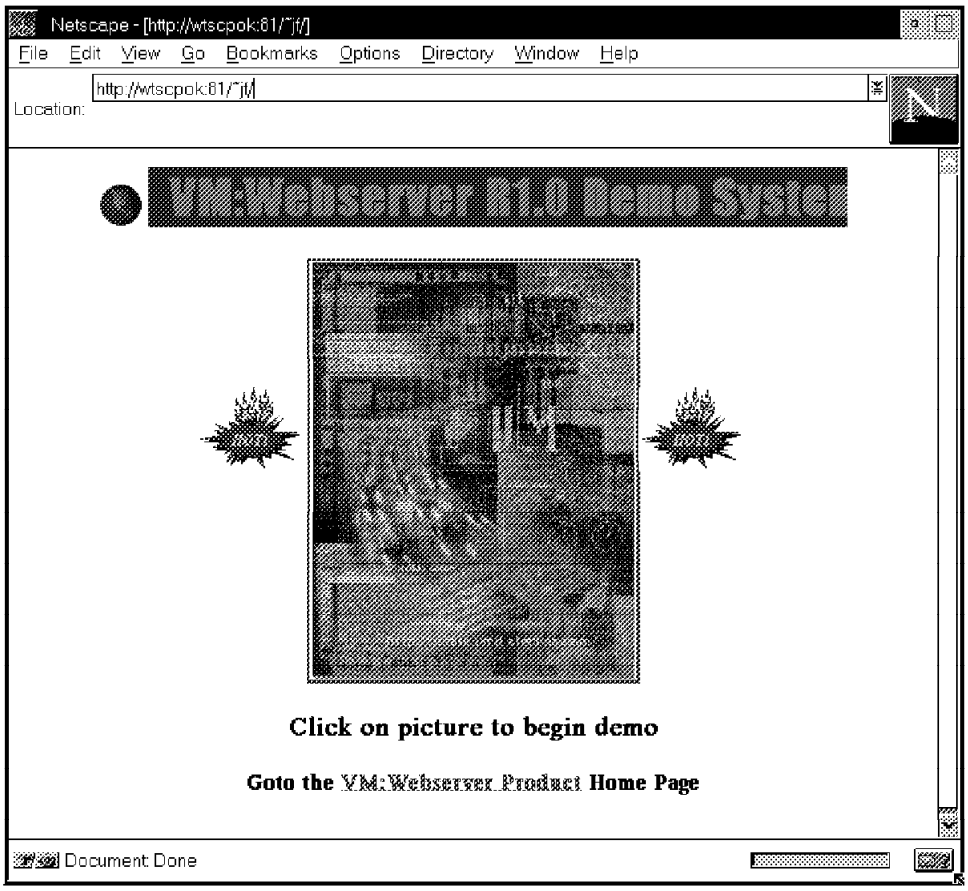


Figure 57. VM:Webgateway Demonstration Web Page. This is the first of many pages in this demonstration.

There are many examples of HTML pages within this demonstration. Some are the typical documentation that a company might have. Others set up the

imagemaps, and use links to get to other parts of the Web page. A summary of the provided material is shown in Table 16 on page 116.

Table 16. Included Samples in VM:Webgateway

Component	**Number**
Sample CGI Scripts	1
HTML pages	36
GIF and JPEG files	14
Sound, Imagemap, Multimedia demonstrations	3
DIRMAP Files	4

Part of the demonstration is a form that could be filled out for reporting a problem to the company help desk. The form prompts for information on the employee and on the problem being reported.

REPORT CGIEXEC is an excellent example of a typical form and CGI program. Figure 58 on page 117 shows the sample of the CGIEXEC. A discussion of the sample follows.

```
/* Read a document using the CGI READ program       */
/* command with the STEM option.                    */
/*                                                  */
trace 'o'
Say 'Hello'
/* Read the form data                               */
'CGI READ 1 ( VAR FIELDS TRANSLATE ASCII EBCDIC'            1
/* Decode the FIELDS string into individual fields  */
Say fields
'CGI URLDECODE (INTO FIELD. VAR FIELDS'                     2
/* Say what we got back from decode                 */
   Do j = 1 to words(field.0)
   x = word(field.0,j)
   Say x'='field.x
   End
   /*  Place code here to handle fields from       */       3
   /*  the document.                                */
Address COMMAND 'GLOBALV SELECT $RPTDMO$ GET PRBNUM'
If datatype(prbnum,'W') Then prbnum = prbnum + 1
    Else prbnum = 1
Address COMMAND 'GLOBALV SELECT $RPTDMO$ PUTP PRBNUM'
/* Write a document using the CGI WRITE     */
/* program command.                         */
/*                                          */

/* Write the CGI Status: header.            */
'CGI WRITE HEADER (STRING Status: 200 Ok'                   4

/* Write an HTTP Content-type: header.      */
header = 'Content-type: text/html'                          5
'CGI WRITE HEADER ( VAR HEADER'

/* Write the document:                      */
Lines.1 = '<HTML>'                                          6
Lines.2 = '<TITLE> HTML created by a CGI REPORT CGIEXEC</TITLE>'
Lines.3 = '<BODY>'
Lines.4 = '<H2>Created by CGI Program REPORT CGIEXEC</H2>'
Lines.5 = '<P>'
Lines.6 = 'Thank you for your problem report' field.name '.'
Lines.7 = '<P>Your problem has been recorded on' date() 'and assigned'
Lines.8 = 'problem #' prbnum '.'
Lines.9 = '</BODY>'
Lines.10 = '</HTML>'
Lines.0 = 10
'CGI WRITE DOCUMENT (STEM LINES.' ,                         7
   'TRANSLATE EBCDIC ASCII CRLF'

Exit
```

Figure 58. Sample VM:Webgateway CGI Script

Notes:

1. This reads the fields passed by the HTML Form. It is all captured from the form as a single record.
2. This breaks down the record back into the individual fields. It creates a stem with each fields included.
3. This is where you would work with the fields, perform some processing, and send the data somewhere.
4. This writes out a header record saying all is OK.
5. This is more header information.
6. This builds the HTML that is displayed.
7. This sends the text to the browser.

VM:Webserver CGI Extension Sample Code

The second sample application is loaded during the installation of VM:Webserver CGI Extension to the server′s run time disks (normally its 194 disk). This sample application is fully described in the *VM:Webgateway Tutorial* manual. If you are interested in Web-enabling 3270 full screen applications, we strongly encourage you to read this manual and work through its examples. While the concepts presented may seem daunting at first, your perseverance is rewarded as you discover the power of the API Sterling Software, Inc. has created in the interfaces illustrated by the *VM:Webgateway Tutorial* manual.

6.5 VM:Webgateway Installation

Because the VM:Webgateway was the first product installed from Sterling Software, Inc. by the redbook team, the VMRMAINT user ID had to be set up.

In AIM′s Installation Phase One, the VMRMAINT user ID attempts to link to every user ID on the VM system to collect necessary system information used to quickly install any combination of the Sterling Software, Inc. products. Because we had limited the RACF authority for VMRMAINT, Phase One failed. We contacted Sterling and obtained alternate installation instructions. The alternate installation bypasses the AIM Phase One process and allowed us to install VM:Webgateway in a manual format without the operations RACF privilege. In most sites this installation process would be handled by normal VM systems administration staff and the VMRMAINT user ID would have these RACF privileges. Thus, these alternate instructions would normally not be needed.

If you already have AIM installed, or plan on installing other VM software products from Sterling, you should use AIM.

In general we find the usage of AIM to be a painless method for software installation. However, we also find that it does not deal gracefully with unexpected errors during the installation process. In particular, we find that due to incorrectly documented minidisk sizes, our installation process failed repeatedly. These failures caused us to have to repeat the installation process several times before it worked. Despite AIM′s efforts to gracefully recover from installation errors, it did not always do so. In particular, we found that it is critically important that the Phase Two installation of the VM:Webgateway components must all occur exactly in the order AIM attempts to perform them in. Any failure in any step seems to lead to later problems with either the installation process itself, or operational problems with the installed products. In general, when errors occurred we found it was necessary to contact Sterling Software, Inc. Customer Support for instructions on how to cause AIM to start the process from scratch rather than relying upon its check-pointed state information. We strongly suggest that you report all errors found during the installation process to Sterling Software, Inc. rather than attempting to solve or bypass the problems yourself. To Sterling Software, Inc.′s credit, all problems were quickly resolved once we called their Customer Support staff with complete problem descriptions.

6.5.1 VM:Webgateway Software Requirements

Table 17 lists the minimum software requirements for VM:Webgateway. The exact current requirements are also fully outlined in the *VM:Webgateway Getting Started* manual.

Table 17. Required Code Levels for VM:Webgateway

Requirement	Minimum Acceptable Level
VM/ESA	VM/ESA 1.2.2 or higher
TCP/IP	TCP/IP release 1.2.2 or higher

6.5.2 VM:Webgateway Hardware Requirements

Besides the items listed in Table 18, an existing TCP/IP network must already be installed. The VM:Webgateway product does not install into SFS file pools. Minidisks must be used for the install. However, data that will be served by the VM:Webgateway SVM may be placed on minidisk, SFS, or BFS DASD areas. No other hardware items are required.

We found that the Sterling Software, Inc. installation documentation was both inconsistent and incorrect in reporting disk sizes needed by the software installation process. We suggest that you take a conservative approach to the allocation of disks by creating them to be too large initially, and then resizing them downwards later if you wish to conserve DASD space. This will potentially save you much grief in the installation process. We are providing here a table of disk sizes that worked for our installation of VM:Webgateway Release 2.2, including VM:Webserver OfficeVision Interface.

Table 18 (Page 1 of 2). VM:Webgateway Release 2.2 User IDs and Disks

User ID/Disk	Size in 3390 cylinders	Purpose
VMRMAINT/191	4 or more	VMRMAINT work files
VMRMAINT/192	15	AIM and AIC program and control files
VMRMAINT/193	5	Public files for all Sterling Software, Inc. products
VMRMAINT/196	4	VM Software News minidisk
VMRMAINT/160	10	VM:Webgateway system administrator minidisk
VMRMAINT/161	10	VM:Webserver OfficeVision Interface system administrator minidisk
VMRMAINT/162	2	VM:Webserver CGI Extension system administrator minidisk
WEBSRV1/191	2 or more	Contains materials that you modify - PROFILE EXEC and VMWEBSRV CONFIG
WEBSRV1/192	5	VM:Webgateway code disk - includes all its online documentation

Table 18 (Page 2 of 2). VM:Webgateway Release 2.2 User IDs and Disks		
User ID/Disk	**Size in 3390 cylinders**	**Purpose**
WEBSRV1/193	6	VM:Webserver OfficeVision Interface code disk - includes all its online documentation
WEBSRV1/194	10	VM:Webserver CGI Extension code disk - includes all its online documentation and its sample application
WEBSRV1/1B0	2	Database disk that contains all the configuration data for VM:Webgateway
Note: WEBSRV1 is the user ID that was chosen when the product was installed locally. The default user ID is VMWEBSRV.		

6.5.3 Programmer Skills for VM:Webgateway

If you are planning on installing VM:Webgateway using AIM, be sure to completely familiarize yourself with the installation process. Other skills that might be required for the installation of VM:Webgateway are listed here:

- Knowledge of HTML. If you are not familiar with HTML, a knowledge of a script language such as DCF or Waterloo Script would be helpful.
- Knowledge of REXX. This is helpful when running or creating CGI scripts.
- Knowledge of Webshare or a current Webshare user.
- Knowledge of PIPELINES. This can be used when writing CGI programs.
- Updating TCP/IP configuration. You may need to specify a different port number. Port 80 is the default and could be in use by another application. If so, a new port number must be supplied. If running with multiple servers, the servers need to be listed along with the port they will share.
- Some general knowledge of SFS and BFS would be helpful. This is only required if the data will be stored in SFS or BFS directories. You need to be able to GRANT AUTHORITY for the server to access the data in the directories. You also need to be able to ACCESS the directories to place the data. You must ENROLL the VM:Webgateway SVM user ID in a file pool and MODIFY USER to add blocks for the SVM.

6.5.4 VM:Webgateway Alternate Installation Instructions

Through Sterling Software, Inc. Customer Support, obtain a copy of INSDBASE VMSI, which must be modified for your system. This file is used to record information during the install. Use the following steps to customize this file for your installation:

1. Put the INSDBASE VMSI file on the VMRMAINT 192 disk. Make it logical record length (LRECL) 132 and record format fixed (RECFM F).
2. Within INSDBASE VMSI, update the DIRECT record to reflect the VOLSER on which the online directory resides. Use the real address information along with the real VOLSER. Figure 59 on page 121 shows the updated record with indications of which fields need updates.

```
DIRECT     192 POKE22
           1   2
```

Figure 59. Updating the DIRECT Record of INSDBASE VMSI

1 Update this to reflect the real address of the volume that contains the online directory.

2 Update this to reflect the VOLSER of the volume that contains the online directory.

3. Update the COMPID record within INSDBASE VMSI in three places. Figure 60 shows the updated record with mark ups to point to the fields needing updates.

```
COMPID     VM:WebBASE     VMWEBSRV  WEBSRV1    LYS20000  WEBSRV1    INSTALL
                                     1          2         3
```

Figure 60. Updating the COMPID Record of INSDBASE VMSI

1 Update this to reflect the name of the SVM.

2 Update this to reflect the account code to be used for this SVM.

3 Update this to reflect the distribution to be used for this SVM.

4. Create the server user ID in the CP directory. Figure 61 on page 122 shows the sample directory entry that we received, as modified to insert links and MDISK records. Minidisk sizes were adjusted according to Table 18 on page 119.

```
00000 * * * Top of File * * *
00001 USER VMWEBSRV VMWEBSRV 56M 48M BEG 64
00002 *-------------------------------------------------------------*
00003 * VM:Webserver Service Virtual Machine                        *
00004 *-------------------------------------------------------------*
00005  ACCOUNT VMWEBSRV
00006 *AC= VMWEBSRV
00007  MACHINE ESA
00008  OPTION MAXCONN 3000 LNKNOPAS
00009  IUCV ALLOW
00010  IUCV ANY
00011  IPL CMS
00012  CONSOLE 0009 3215
00013  SPOOL 00C 2540 READER *
00014  SPOOL 00D 2540 PUNCH A
00015  SPOOL 00E 1403 A
00016  LINK MAINT 190 190 RR
00017  LINK MAINT 19E 19E RR
00018  LINK VMRMAINT 192 1FF RR
00019  LINK VMRMAINT 193 1FE RR
00020 * A link to the DASD area containing the CP directory.
00021  LINK $DASD$ 0192 1A0 RR
00022 *-------------------------------------------------------------
00023 **NOTE** Minidisk link passwords must remain VMRPASS
00024 *        until after Installation Phase II has completed.
00025 *-------------------------------------------------------------
00026 * -Minidisk VMWEBSRV 191 requires 2 cylinders
00027 MDISK  191  3390  48  2  VST020  MR  VMRPASS  VMRPASS
00028 * -Minidisk VMWEBSRV 192 requires 5 cylinders
00029 MDISK  192  3390  141 5  VST020  RR  VMRPASS  VMRPASS
00030 * -Minidisk VMWEBSRV 193 requires 6 cylinders
00031 MDISK  193  3390  161 6  VST020  RR  VMRPASS  VMRPASS
00032 * -Minidisk VMWEBSRV 194 requires 10 cylinders
00033 MDISK  194  3390  181 10  VST020  RR  VMRPASS  VMRPASS
00034 * -Minidisk VMWEBSRV 1B0 requires 1 cylinders
00035 MDISK  1B0  3390  49  1  VST020  MR  VMRPASS  VMRPASS
00036 * * * End of File * * *
```

Figure 61. VM:Webgateway Release 2.2 SVM Sample Directory Entry

5. Give VMRMAINT write access to the minidisk defined to the VM:Webgateway SVM.
6. Mount the installation tape for VMRMAINT as 181. From VMRMAINT, issue VMIMAINT and select Installation Phase Two. This will format the minidisk, issue the TAPE LOAD commands, and complete the installation process.
7. At the end, you will receive messages about Automated Implementation and Control (AIC) and loading help files. While AIC facilities do not apply for this product, and there is complete help for VM:Webgateway online as Web pages, there are CMS HELP files for the VM system operator commands in the product. Thus, we recommend loading the CMS HELP files for all product components for the benefit of your VM operations staff.

6.6 VM:Webgateway Configuration

There are two SVM user IDs required to run the VM:Webgateway code. One ID is the actual server, and the other is the maintenance ID. We chose WEBSRV1 for the server (see 8.2.2, “Server Naming Convention” on page 165 for more information). VMRMAINT is the maintenance user ID. With VM:Webgateway, it is possible to run with multiple servers and have them all listen for incoming

communications on the same TCP/IP port. See topic 6.8.2, “Multiple VM:Webgateway Servers” on page 131 for more details.

Figure 62 shows the typical configuration for the VM:Webgateway product.

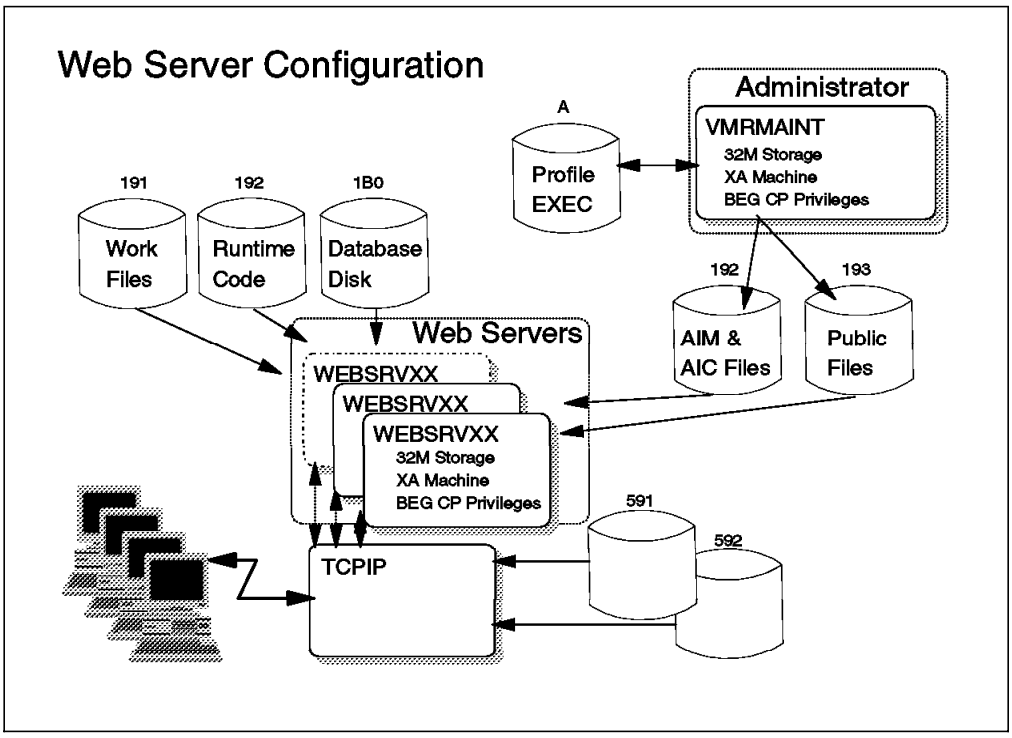


Figure 62. VM:Webgateway Configuration

One of the first things that must be configured is the port number and the root domain that the VM:Webgateway SVM will use when serving data. This is entered with the CONFIG SOCKET command.

```
CONFIG SOCKET ADD * portnumber ROOT SFS filepoolid:svmid.dirname
```

This command sets the port number specified for all IP addresses serviced by the TCP/IP server listed in TCPIP DATA. *Portnumber* is the TCP/IP port that the VM:Webgateway SVM user ID will use to listen for incoming communications. It also sets the root domain for the server to an SFS directory. *Filepoolid* is the SFS file pool that the VM:Webgateway SVM user ID is enrolled in. *Svmid* is the name of the VM:Webgateway SVM. *Dirname* is the directory that is used for storing data.

Also, MDISK, BFS, or CMS may be specified as the root domain. See the online documentation for details on this command.

When the option ROOT is specified, it defines the default root domain for the host system. When users select this host name in a URL, they are directed to this root domain.

Most of the configuration data is stored in a database located on the server’s 1B0 disk. However, there is one file that does contain some configuration information. This file is named VMWEBSRV CONFIG. It is located on the server’s 191 disk.

This file contains information that cannot otherwise be saved in the configuration database. The format and meaning of the records that may be placed in this file are fully documented in the *VM:Webgateway Getting Started* manual. Records that are found in this file include:

CPUID Information supplied by Sterling Software, Inc. as part of its asset protection.

DATABASE Defines the name and location of the database file and the size of the cache that should be used. A formula for determining the size of the cache is included in the *VM:Webgateway Getting Started* manual. The default, which is currently 1000, was used for our testing. This cache is for database records only. It is not a cache for HTML files or any other data that will be served.

DIRECT Defines the location of the CP online directory.

KEYPASS Defines the phrase that SSL uses to encrypt all keys it stores in its database.

OPTIONS Enables VM:Webgateway components and features to run.

PRODUCT Enables VMRMAINT to perform maintenance.

RESVADDR Identifies additional virtual addresses that you do not want VM:Webgateway to detach during initialization.

The changing of the VMWEBSRV CONFIG file will require that the server be recycled to pick up the change.

6.6.1 VM:Webgateway Server Configuration Commands

The CONFIG SOCKET command is one of the commands used to make configuration changes. These configuration changes to the product may be done the following four ways:

- Line-mode interface
- SMSG interface
- CMS interface
- Through the World Wide Web

VM:Webgateway Line-Mode Interface

All of the configuration commands may be entered through the line-mode interface when you are logged onto the VM:Webgateway SVM user ID. Figure 63 shows a command and the output received.

```
q userpages
14:24:59 WEBSRV1  OC VIWCMD1110I Enter:  q userpages
14:24:59 WEBSRV1  OC VIWCMD1111I Beginning command: q userpages
14:24:59 WEBSRV1  OC User pages are ALLOWED. File INITIAL is providing initial
access control
14:24:59 WEBSRV1  OC VIWCMD1112I Ending command q with completion code 0.
```

Figure 63. VM:Webgateway Command Entered Using the Line-Mode Interface

VM:Webgateway SMSG Interface

The SMSG interface allows communication with the server through CP. Any of the configuration commands may be entered with this interface. Figure 64 shows a command entered through the SMSG interface.

```
sm websrv1 q userpages
Ready;
14:26:21 User pages are ALLOWED. File INITIAL is providing initial access
control
```

Figure 64. VM:Webgateway Command Entered Using the SMSG Interface

VM:Webgateway CMS Interface

The CMS interface uses the WEBSRV1 and VIWCOM MODULEs. These are placed on a PUBLIC minidisk for all users to access. Any of the configuration commands may be entered with this interface. Using this method, CP will honor your EMSG setting.

```
websrv1 q userpages
User pages are ALLOWED. File INITIAL is providing initial access control
Ready;
```

Figure 65. VM:Webgateway Command Entered Using the CMS Interface

VM:Webgateway Configuration Through the World Wide Web

Any of the configuration commands may be executed through your Web browser. See Figure 66 on page 126 for the list of actions that may be executed. The general user may not communicate with the VM:Webgateway SVM in this manner. Information providers (for instance, CGI writers) will have a VM user ID, and thus can use one of the other command interfaces to issue commands such as QUERY FILETYPE.

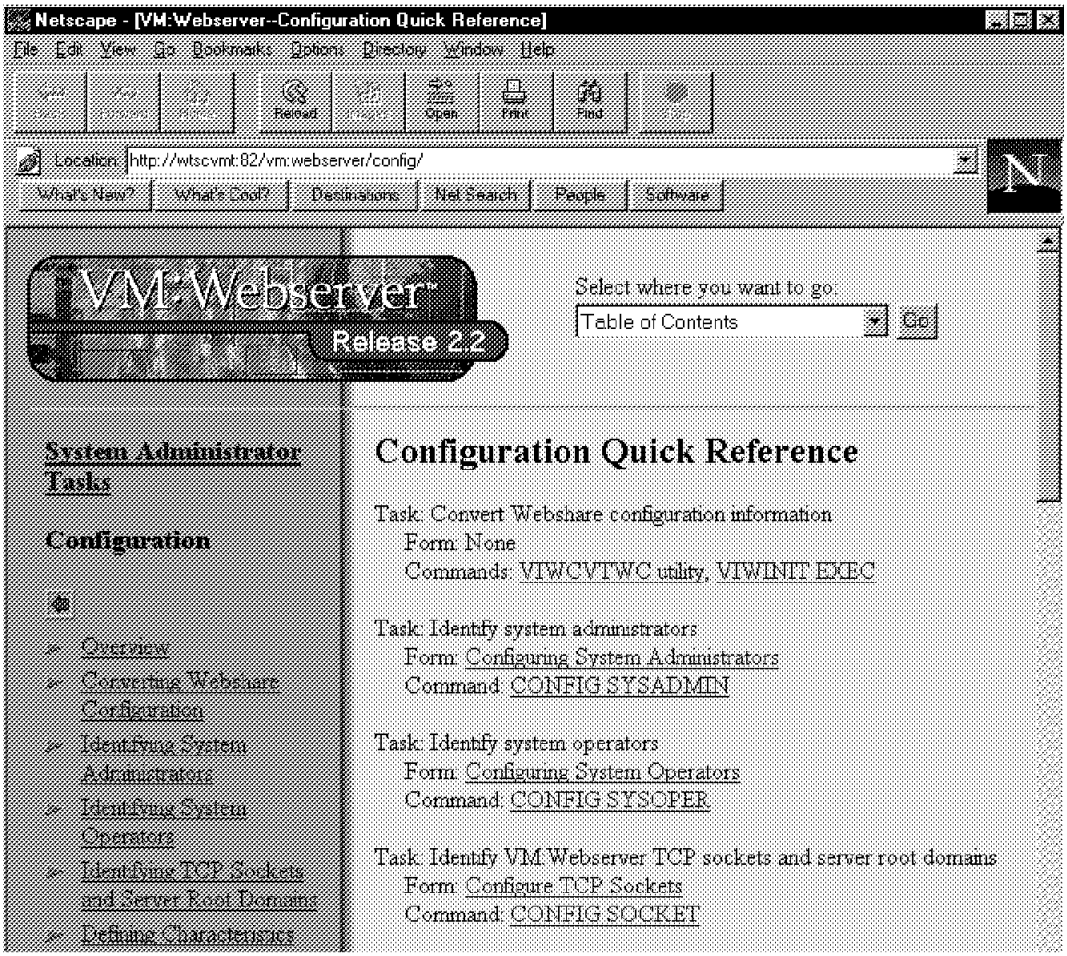


Figure 66. VM:Webgateway Configuration Commands on the World Wide Web. From this page, http://hostname:port/VM:Webserver/Config/, any of the configuration changes may be made.

Anyone knowing the correct user ID and password of a configured VM:Webgateway system administrator may access the configuration pages through the World Wide Web. VMRMAINT is the default system administrator for the VM:Webgateway product. Many of the changes may be made by selecting and clicking.

Figure 67 on page 127 shows the choices that are made regarding messages, the user ID that will receive dumps in case of a product abend, and the configuration of the generation of VM accounting records.

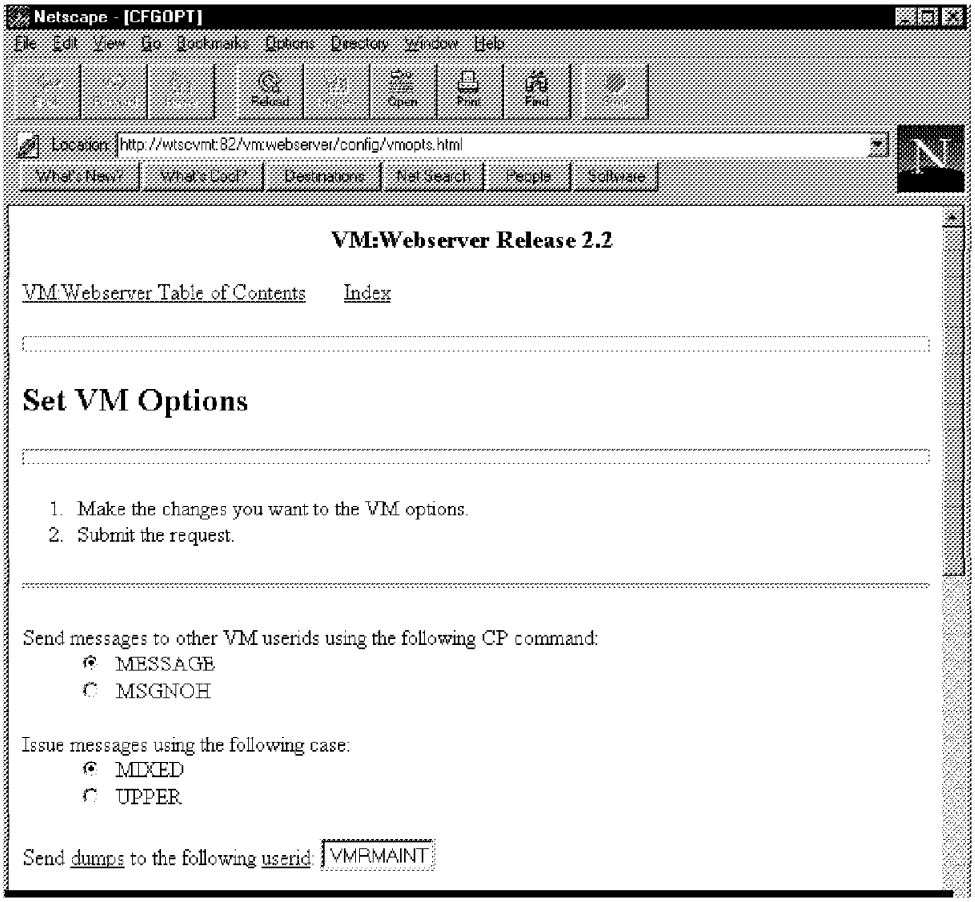


Figure 67. Configuring VM:Webgateway through the World Wide Web. From this page, http://hostname:port/VM:Webserver/ConfigVMOpts.html/, you can change the VM options product settings.

Configuration changes take effect immediately, no matter which interface is used. There is no need for the server to be recycled to pick up the changes. This allows the server to be available for use 24 hours per day, 7 days per week.

6.7 VM:Webgateway Administration

The two types of administrators for VM:Webgateway are system administrators and system operators.

6.7.1 VM:Webgateway System Administrator

The system administrator is responsible for configuring and maintaining the VM:Webgateway SVM user ID. The SVM user ID has all of the administration privileges. It is allowed to enter any command. It may enter the commands that require SYSADMIN, SYSOPER, and no authority. System administrator user IDs will also receive VM:Webgateway messages destined for a system administrator.

VM:Webgateway System Administrator Commands

The CONFIG SOCKET command is discussed in 6.6, “VM:Webgateway Configuration” on page 122. There are other commands that a system administrator may enter that will alter the configuration and otherwise control the operation of the server. The following list shows some of them:

CERTREQ Creates a request for a server certificate that can be sent to a certificate authority.

CMS Lets you run a CMS subset or CP command on the VM:Webgateway SVM.

CONFIG ACCOUNT Specifies if and how VM accounting records should be generated.

CONFIG CERTIFICATE Adds or deletes an SSL certificate.

CONFIG CGIUSER Specifies who can serve CGI programs.

CONFIG DUMP Specifies the user ID that will receive dumps.

CONFIG FILETYPE Specifies how VM:Webgateway handles files based on the file type. An example would be how VM:Webgateway handles character set translation for the file type in question.

CONFIG KEY Adds or deletes an SSL key.

CONFIG MSGCASE Specifies whether VM:Webgateway will display messages in upper or mixed case.

CONFIG MSGCMD Specifies the command VM:Webgateway will use to send messages to the user.

CONFIG QUIESCE Adds or deletes user IDs that may quiesce the database. VMBACKUP is the default.

CONFIG SYSADMIN Adds or deletes system administrator user IDs. VMRMAINT is the default.

CONFIG SYSOPER Adds or deletes system operator user IDs. VMRMAINT is the default.

CONFIG USERPAGES Specifies whether user pages are allowed to serve data. If so, you may also specify the file name of the initial ACCESS file. This file would be located in the server’s ROOT domain.

CONFIG USERROOT Allows a system administrator user ID to set a system default for the user page location. This can be overridden by the user.

CONFIG WORKER Creates or removes worker machines.

EXPORT Allows SSL keys and certificates to be saved in a CMS file for sharing with other VM:Webgateway servers.

SET USERROOT Allows a system administrator user ID to set the root for a user page. This overrides the location specified in the CONFIG USERROOT command. This command is also used to define which ACCESS file will be used for security purposes. See topic 6.8.4, “Security in VM:Webgateway” on page 134 for more details.

All of the commands listed have a corresponding QUERY command. This allows the system administrator to determine the current settings for the VM:Webgateway SVM. There is also a QUERY CONFIG command that returns the values for all settings at one time.

The system administrator form of the SET USERROOT command can be used to create an alias. You may specify a non-existing user ID and have VM:Webgateway serve the data from the specified location. You are limited to 8 characters.

Figure 68 shows the system default for user pages. Also, it shows that DEPT123 is an alias for JIM's 123 minidisk. A tilde (~) is used to tell VM:Webgateway to use DEPT123 as a user page. It would then serve the data from the 123 minidisk belonging to JIM.

```
sm websrv1 q userroot *
Ready;
 14:13:47 Userid    USERROOT
 14:13:47 --------  ---------------------------------------------
 14:13:47 SYSTEM    SFS SFSLSY2:*.VMWEBSERVER
 14:13:47 DEPT123   MDISK JIM 123
```

Figure 68. Creating Alias USERROOT in VM:Webgateway

See the online documentation for the complete syntax of the commands mentioned in this section.

6.7.2 VM:Webgateway System Operator

The system operator is responsible for normal operational matters on the VM:Webgateway SVM, and it may enter the commands that require SYSOPER, and no authority. The following list shows some of them:

ACCOUNT Close the current accounting stream to the accounting data collector.

END Allow the system operator to perform an orderly shutdown of the VM:Webgateway SVM.

WORKER Control the operational availability of worker service machines.

System operator user IDs will also receive VM:Webgateway messages destined for a system operator.

See the online documentation for the complete syntax of the commands mentioned in this section.

6.7.3 VM:Webgateway General User

The user may also communicate with the VM:Webgateway SVM with the following commands:

CAN Checks the syntax of records in DIRMAP and ACCESS files for a specified URL.

QUERY USERROOT Determines the root location for that user ID.

QUERY CGIUSER Displays which user IDs may serve CGI program. It will also display which filetypes of CGI programs that user may serve.

QUERY FILETYPE Displays how the VM:Webgateway SVM will handle files based on the file type.

QUERY USERPAGES Displays if user pages are allowed to serve data.

SET USERROOT Allows the user to specify the location of the user page for that user ID. This overrides the system default.

See the online documentation for complete syntax of the commands mentioned in this section.

6.8 VM:Webgateway Features and Experiences

This section discusses some of the various features and experiences during our use of the VM:Webgateway product.

6.8.1 General Features of VM:Webgateway

Listed in this section is a cross section of features of VM:Webgateway Release 2.2:

- Support for Web enabling VM applications that operate in either a line mode interface or via 3270 screens (via a sophisticated *screen scraping* technology). This includes a Web browser-based 3270 emulator that is specially tailored to simplify the process of Web enabling 3270-based applications.
- Extensive and complete online (HTML) documentation, including separate print versions that cover entire functional areas. See “VM:Webgateway Online Documentation” on page 113 for more information.
- Support of the SSL security protocol. See 1.10.1, “Secure Sockets Layer (SSL)” on page 43 for more information.
- National Language Support (NLS) for character sets other than U.S. English.
- A rich access control facility. See 6.8.4, “Security in VM:Webgateway” on page 134 for more information.
- Support for serving data that is resident on CMS minidisks, the SFS and BFS file systems, and the standard CMS search order. This includes files created by standard PC-based Web authoring tools.
- A rich and varied set of CGI programming environments, including both emulation of Webshare′s CMS Pipelines-based CGI environment and a command oriented CGI environment. See 6.8.7, “VM:Webgateway CGI Scripts” on page 137 for more information.
- Support for dynamic reformatting of static documents under control of a CMS Pipelines customer-written filter stage.
- Support for a dynamic-worker machine facility. See 6.8.8, “VM:Webgateway Dynamic Worker Machines” on page 140 for more information.
- Support for server-side includes following the de facto Internet standards in this area. See 7.7, “Server Side Includes” on page 160 for more information.
- Full virtual hosting support.
- Support for CMS multitasking exploitative applications.

- An extensive, real-time configuration control interface. See 6.6, "VM:Webgateway Configuration" on page 122 and 6.7, "VM:Webgateway Administration" on page 127 for more information.
- Flexible, efficient support for generation of VM accounting records.
- Support to recognize expired VM passwords.
- Support to change VM passwords without requiring a VM logon.
- A rich URL decoding facility that handles both form encoded data and simple URL encoded query strings.
- Ability to serve data in a format controlled by either its native CMS file system file type or a configured *content file type*.
- Transaction logging in a CERN/NCSA-common log format suitable for processing by standard log file analysis programs available freely on the World Wide Web. See 6.8.5, "VM:Webgateway Logging" on page 135 for more information.
- Support for server side image map resolution via a provided CGI. See 6.8.6, "VM:Webgateway Imagemaps" on page 137 for more information.
- When data is being served from SFS, it is possible that the data will be migrated to tape. If VM:Webgateway encounters this situation, it will report the file is not found and suggest contacting the administrator.
- VM:Webgateway will automatically create a directory tree when the URL specifies a directory without a specific file. Alternately, the owner of the directory can provide the name of a CMS file to serve rather than a dynamic listing of the contents of the directory.
- If a URL points to a directory or file that does not exist, the browser is presented with a page that states the file is not found. From here, you can access the online documentation for VM:Webgateway. An index to all the VM:Webgateway topics is available from this and all other error pages.
- Conversion utilities for Webshare configuration and FILELIST files. See 6.8.9, "Converting from Webshare to VM:Webgateway" on page 140 for more information.
- Optional VM:Webserver OfficeVision Interface product. See 6.9, "VM:Webserver OfficeVision Interface" on page 142 for more information.

See the online documentation for VM:Webgateway for more information on all of these facilities.

6.8.2 Multiple VM:Webgateway Servers

With VM:Webgateway, the goal is to run as few servers as possible. Features, such as multitasking and dynamic workers (see 6.8.8, "VM:Webgateway Dynamic Worker Machines" on page 140), maximize the throughput of a single server. If the performance of a single server is inadequate for your specific needs, you can set up multiple servers, but you should consider the following items before making this decision.

- With multiple servers, there are multiple VM:Webgateway databases. These are stored on the VM:Webgateway SVM 1B0 disk. Each of the servers needs write access to this data. Because of this, each individual server must be updated with any configuration changes that the administrator or the user might make. For example, if there are four servers defined and you want to update the location of your user page, you must issue the SET USERROOT

command to each server. Such a setup precludes the usage of the online configuration forms, as they will only update a single server, which will be selected at random by the TCPIP server for each configuration request made.

- Ensure that all servers have access to the data that will be served; that is, that they have access to the minidisk or to the SFS directories where the data is being maintained.

If you have decided to create multiple servers, the following will help with the setup of the additional user IDs:

1. Add the additional user IDs to the CP directory. Each must be a mirror of the first SVM with one exception: there is no need for product code disks (see Table 18 on page 119 for more information) on each of the user IDs. The additional servers should be given CP directory links to these disks.
2. On VMRMAINT's 192 disk, update the file VMRMAINT CONFIG. Duplicate the line that contains VM:WEBSERVER, VM:WEBBASE, VM:WEBOV, or VM:WEBCGIEXT. Update each line to contain the name of the additional servers. Figure 69 shows WEBSRV1 and WEBSRV4 as VM:Webgateway servers.

```
PRODUCT    VM:WEBSERVER     WEBSRV1     1.1Y
PRODUCT    VM:WEBSERVER     WEBSRV4     1.1Y
```

Figure 69. Updating VMRMAINT CONFIG for VM:Webgateway

3. On VMRMAINT's 192 disk, there is a file named *servername* MDISKS. *Servername* is the name of the first server. Make a copy of this file for each of the additional servers being defined. The file name should be the user ID of the new server. This file requires a couple of changes. See Figure 70 for details.

```
 WEBSRV4  MDISKS   D1  F 80  Trunc=80 Size=14 Line=0 Col=1 Alt=0
====>
00000 * * * Top of File * * *
00001 *-------------------------------------------------------
00002 * This file identifies the minidisks associated with
00003 * the VM:Webserver Service Virtual Machine named WEBSRV1.
00004 *-------------------------------------------------------
00005 * The record format is:
00006 *
00007 * KEYWORD     OWNERID       VADDR  RPASS    WPASS    MPASS
00008
00009   LOCAL       WEBSRV4  1    191    VMRPASS  VMRPASS
00010   VMSI        WEBSRV1  2    192    VMRPASS  VMRPASS
00011   DBASE       WEBSRV4  1    1B0    VMRPASS  VMRPASS
00012   ALOCAL      VMRMAINT      191    VMRPASS  VMRPASS
00013   AWEBSERV    VMRMAINT      160    VMRPASS  VMRPASS
00014   AWEBOV      VMRMAINT      161    VMRPASS  VMRPASS
00015   VMSIOV      WEBSRV4  2    193    VMRPASS  VMRPASS
00016   VMSIGW      WEBSRV4  2    194    VMRPASS  VMRPASS
00017   AWEBGW      VMRMAINT      162    VMRPASS  VMRPASS
00018 * * * End of File * * *
```

Figure 70. Updating WEBSRV4 MDISKS for VM:Webgateway

1 Update to reflect the name of the new server.

2 There is nothing to update; this is how the new server will gain access to the VM:Webgateway code.

4. On the TCP/IP SVM, update PROFILE TCPIP to include the new servers. They will share the same port as the original. Figure 71 shows that WEBSRV1 and WEBSRV4 are sharing port 81.

```
81 TCP WEBSRV1             ; Web     Server
81 TCP WEBSRV4             ; Web     Server
```

Figure 71. Updating PROFILE TCPIP for VM:Webgateway

5. Check that the new servers have the same access to data, both now and in the future, as the original server.
6. Bring up the additional servers, then issue the CONFIG SOCKET command to point the new servers to the port they are using and update the root domain for the servers. Point all servers to the same root domain.
7. Make any other configuration changes that may be required. Any configuration changes made to the original server should be made to each of the additional servers. The configurations should match or unpredictable results will occur.
8. On VMRMAINT's 193 disk, there is a file named *servername* MODULE. This is used for the CMS interface. Make a copy of this file for each of the additional servers being defined. The file name should be the user ID of the new server. This file should also be placed on a public disk.

6.8.3 VM:Webgateway's DIRMAP Files

DIRMAP files are CMS files that help VM:Webgateway locate the files you want to serve. They have a file type of DIRMAP. They may be found in the VM:Webgateway root domain or on user pages. The following indicates whether DIRMAP files are required, based on how the root domain or user pages are defined:

Defined As:	**DIRMAP file:**
Minidisk	Required
SFS Directory	Optional
BFS Directory	Optional
CMS Search Order	Required

Records within the DIRMAP files fall into one of two categories:

Directory Control Records These records control tasks pertaining to files.

Access Control Records These records enforce security over the data.

DIRMAP files may be used to create a subdirectory hierarchy for root domain or user pages defined as minidisk or CMS search order. When the root domain or user page is defined using BFS or SFS, VM:Webgateway will serve any of the files for which it has authorization, unless an access record control specifies that the data should not be served.

DIRMAP files can be used for the following purposes:

- Identify files you want to serve.

- Logically group files you want to serve.
- Allow users to identify a file to serve by specifying only the file name in the URL.
- Allow users to identify a file by a character string of more than eight characters.
- Specify a file you want VM:Webgateway to serve when a user does not identify a file in a URL (for instance, as a directory index).
- Enforce security over the data you want to serve.
- Cause a file to be served with characteristics of a file type other than the one it has in the file system (a *content file type*).

The DIRMAP files are thoroughly explained as part of the online documentation.

6.8.4 Security in VM:Webgateway

With VM:Webgateway basic security is established. By default, there are no restrictions placed on what data may be accessed or who may access the data. Any data that the server has access to is available to the users on the World Wide Web.

With the use of access records within the DIRMAP files, the system administrator or a user can control who can gain access to data. Access records may also be placed in files with the file type of ACCESS. The file name is determined by the CONFIG USERPAGES command. This file would be located in the VM:Webgateway root domain. Access records allow a security check based on any of the following:

- IP address, or the TCP/IP DNS name (see the topic “Domain Name System (DNS)” on page 38 for more information) associated with it
- HTTP request method
- Requesting user, or an ACI group associated with it, or its status as a VM:Webgateway system administrator or operator
- File ID of the requested file
- The referring URL for this request′s URL

IP Address and DNS Name

This is used to allow or deny access to your data based on the IP address that TCP/IP supplies or the DNS name associated with the IP address. When using access records, you may use specific IP addresses or DNS names or pattern matching to authorize access. With pattern matching, you can use an asterisk (*) for multiple characters and a percent sign (%) to be used for a single character. With IP addresses, bit masking may also be used. When setting up the access, a mask may be added to the access control record. This mask is logically ANDed with the specified IP address, and the resulting value determines who is allowed or denied access.

HTTP Request Method

This is used to allow or deny access to your data based on the specific HTTP request method. This control has three forms:

- Method Get
- Method Post
- Method Get Post

All HEAD requests will, by definition, be treated as a GET request.

Requesting User

Requesting user security allows or denies access to your data based on the specific user requesting the access, or attributes associated with that user, such as its ACI group or its status as a privileged user in VM:Webgateway. Using access records, you may use specific user IDs or pattern matching to authorize access. Pattern matching is the same as that discussed in "IP Address and DNS Name" on page 134.

When controlling data access in this manner, the user ID and password combination may be verified in one of three ways. Within the DIRMAP file, the PASSWORD record may specify VMDIR, FILE, or USEREXIT. If VMDIR is specified, the user ID and password would be verified by use of an external security manager (ESM). If an ESM is not used on the system, the CP directory password is used to verify the user ID and password.

When FILE is specified, the specified file is searched to verify that the given user ID/password pair is valid. This search parallels the search of the CP directory when VMDIR is used.

When USEREXIT is specified, a site-written user exit is used for authorization. This user exit is an EXEC that is located within the search order of the VM:Webgateway SVM. Arguments passed to this EXEC are the user ID and password as entered by the user. The name of the exit is specified within the DIRMAP file on the PASSWORD access record.

VM:Webgateway recognizes the following return codes and will allow or deny access accordingly:

RC=0 User ID and password entered are valid.

RC=4 User ID and password entered are not valid.

RC=8 User ID and password entered could not be evaluated because the security manager is not available.

A complete listing of ACCESS control records and their syntax is included in the online documentation.

6.8.5 VM:Webgateway Logging

Logging for the VM:Webgateway server is maintained in the console file that is produced. By default, this goes to the VMRMAINT user ID. This could be changed to another user ID by updating the profile EXEC for the server.

All logging is done using CERN/National Center for Super-Computer Research (NCSA) common log format. See Figure 72 on page 136 for a sample of the logging that takes place. It shows some typical GET requests, along with a user issuing a command.

```
10:33:33 VMWEBSRV F6 VIWHTTO8OOI 9.12.2.141:1134::9.12.3.4:82 GET /vm:webserver/help/ HTTP/1.0.
10:33:35 VMWEBSRV F7 VIWHTTO8OOI 9.12.2.141:1135::9.12.3.4:82 GET /VM:Webserver/images/backgrnd.gif HTTP/1.0.
10:33:35 VMWEBSRV F7 VIWLOGO771I 173 .9.12.2.141 - - -13/Oct/1998:14:33:35“ ’GET /VM:Webserver/images/backgrnd.gif HTTP/1.0’ 304 93.
10:33:35 VMWEBSRV F7 VIWLOGO773I 173 . ’http://wtscvmt:82/vm:webserver/help/’ ’Mozilla/4.04 -en“ (Win95; U ;Nav)’.
10:33:35 VMWEBSRV F8 VIWHTTO8OOI 9.12.2.141:1136::9.12.3.4:82 GET /VM:Webserver/images/viwhdr.gif HTTP/1.0.
10:33:35 VMWEBSRV F8 VIWLOGO771I 174 .9.12.2.141 - - -13/Oct/1998:14:33:35“ ’GET /VM:Webserver/images/viwhdr.gif HTTP/1.0’ 304 93 ’.
10:33:35 VMWEBSRV F8 VIWLOGO773I 174 .http://wtscvmt:82/vm:webserver/help/’ ’Mozilla/4.04 -en“ (Win95; U ;Nav)’.
10:33:48 VMWEBSRV F6 VIWLOGO771I 172 .9.12.2.141 - - -13/Oct/1998:14:33:48“ ’GET /vm:webserver/help/ HTTP/1.0’ 200 199781 ’’ ’Mozil.
10:33:48 VMWEBSRV F6 VIWLOGO773I 172 .la/4.04 -en“ (Win95; U ;Nav)’.
10:34:14 VMWEBSRV F9 VIWHTTO8OOI 9.12.2.141:1137::9.12.3.4:82 GET /VM:Webserver/images/topbutn.gif HTTP/1.0.
10:34:14 VMWEBSRV F9 VIWLOGO771I 175 .9.12.2.141 - - -13/Oct/1998:14:34:14“ ’GET /VM:Webserver/images/topbutn.gif HTTP/1.0’ 304 93 .
10:34:14 VMWEBSRV F9 VIWLOGO773I 175 .’http://wtscvmt:82/vm:webserver/help/’ ’Mozilla/4.04 -en“ (Win95; U ;Nav)’.
10:38:17 LS8105NG FB VIWCMD1111I Beginning command: query userroot *
10:38:17 LS8105NG FB VIWQUEO059E You are not authorized to issue the QUERY USERROOT command for another userid.
10:38:17 LS8105NG FB VIWCMD1112I Ending command query with completion code 59.
10:38:33 LS8105NG O4 VIWCMD1111I Beginning command: query userroot ls8105ng
10:38:33 LS8105NG O4 VIWCMD1112I Ending command query with completion code 56.
10:39:12 LS8105NG OD VIWCMD1111I Beginning command: set userroot ls8105ng mdisk ls8105ng 191
10:39:13 LS8105NG OD VIWCMD1112I Ending command set with completion code 0.
10:39:40 LS8105NG 11 VIWCMD1111I Beginning command: query cgiuser ls8105ng
10:39:40 LS8105NG 11 VIWCMD1112I Ending command query with completion code 0.
10:40:15 VMWEBSRV 12 VIWHTTO8OOI 9.12.2.141:1139::9.12.3.4:82 GET /vm:webserver/config/ HTTP/1.0.
10:40:17 VMWEBSRV 12 VIWLOGO771I 176 .9.12.2.141 - - -13/Oct/1998:14:40:17“ ’GET /vm:webserver/config/ HTTP/1.0’ 200 24792 ’’ ’Mozi.
10:40:17 VMWEBSRV 12 VIWLOGO773I 176 .lla/4.04 -en“ (Win95; U ;Nav)’.
10:40:17 VMWEBSRV 13 VIWHTTO8OOI 9.12.2.141:1140::9.12.3.4:82 GET /VM:Webserver/images/prevbutn.gif HTTP/1.0.
10:40:18 VMWEBSRV 13 VIWLOGO771I 177 .9.12.2.141 - - -13/Oct/1998:14:40:18“ ’GET /VM:Webserver/images/prevbutn.gif HTTP/1.0’ 304 93.
10:40:18 VMWEBSRV 13 VIWLOGO773I 177 . ’http://wtscvmt:82/vm:webserver/config/’ ’Mozilla/4.04 -en“ (Win95; U ;Nav)’.
10:40:18 VMWEBSRV 14 VIWHTTO8OOI 9.12.2.141:1141::9.12.3.4:82 GET /VM:Webserver/images/ptrother.gif HTTP/1.0.
10:40:18 VMWEBSRV 14 VIWLOGO771I 178 .9.12.2.141 - - -13/Oct/1998:14:40:18“ ’GET /VM:Webserver/images/ptrother.gif HTTP/1.0’ 304 93.
10:40:18 VMWEBSRV 14 VIWLOGO773I 178 . ’http://wtscvmt:82/vm:webserver/config/’ ’Mozilla/4.04 -en“ (Win95; U ;Nav)’.
10:40:19 VMWEBSRV 15 VIWHTTO8OOI 9.12.2.141:1142::9.12.3.4:82 GET /VM:Webserver/images/ptrcurr.gif HTTP/1.0.
10:40:19 VMWEBSRV 15 VIWLOGO771I 179 .9.12.2.141 - - -13/Oct/1998:14:40:19“ ’GET /VM:Webserver/images/ptrcurr.gif HTTP/1.0’ 304 93 .
10:40:19 VMWEBSRV 15 VIWLOGO773I 179 .’http://wtscvmt:82/vm:webserver/config/’ ’Mozilla/4.04 -en“ (Win95; U ;Nav)’.
10:40:25 VMWEBSRV 16 VIWHTTO8OOI 9.12.2.141:1143::9.12.3.4:82 GET /vm:webserver/config/filetype.html HTTP/1.0.
10:40:26 VMWEBSRV 16 VIWHTTO76OI Error document UserNotAuth delivered to client, status: 401.
10:40:26 VMWEBSRV 16 VIWHTTO761I , , , , , , , , , , , , , , , , , ,.
10:40:26 VMWEBSRV 16 VIWLOGO771I 180 .9.12.2.141 - - -13/Oct/1998:14:40:26“ ’GET /vm:webserver/config/filetype.html HTTP/1.0’ 401 -.
10:40:26 VMWEBSRV 16 VIWLOGO773I 180 . ’http://wtscvmt:82/vm:webserver/config/’ ’Mozilla/4.04 -en“ (Win95; U ;Nav)’.
10:40:37 VMWEBSRV 17 VIWHTTO8OOI 9.12.2.141:1144::9.12.3.4:82 GET /vm:webserver/config/filetype.html HTTP/1.0.
10:40:37 VMWEBSRV 17 VIWLOGO771I 181 .9.12.2.141 - vmrmaint -13/Oct/1998:14:40:37“ ’GET /vm:webserver/config/filetype.html HTTP/1.0.
10:40:37 VMWEBSRV 17 VIWLOGO773I 181 .’ 200 4395 ’http://wtscvmt:82/vm:webserver/config/’ ’Mozilla/4.04 -en“ (Win95; U ;Nav)’.
10:40:48 VMWEBSRV 18 VIWHTTO8OOI 9.12.2.141:1145::9.12.3.4:82 GET /vm:webserver/config/FileType.Choice?action=change&allft=*&mimetyp
e=*&translation=an.
10:40:49 VMWEBSRV 18 VIWLOGO771I 182 .9.12.2.141 - vmrmaint -13/Oct/1998:14:40:49“ ’GET /vm:webserver/config/FileType.Choice?action.
10:40:49 VMWEBSRV 18 VIWLOGO772I 182 .=change&allft=*&mimetype=*&translation=any&ssi=any&filter=any&mappedto=cgi&environment=any&au.
10:40:49 VMWEBSRV 18 VIWLOGO772I 182 .thheaderpassed=any HTTP/1.0’ 200 14361 ’http://wtscvmt:82/vm:webserver/config/filetype.html’ .
10:40:49 VMWEBSRV 18 VIWLOGO773I 182 .’Mozilla/4.04 -en“ (Win95; U ;Nav)’.
```

Figure 72. Sample VM:Webgateway Log Entries

The logging function records and audits connections, commands, configuration changes, and information transactions. CGI scripts can use the logging function to put any information into the log that might be needed. See topic 6.8.7, “VM:Webgateway CGI Scripts” on page 137 for more details on CGI scripts. The user-written scripts could place details about the execution of the CGI script into the log. This information may be used for debugging or audit purposes. It is up to the CGI script writer to decide what is logged. Figure 73 on page 137 shows a sample CGI script. It pulls some information and logs it. See Figure 74 on page 137 for the entries that were cut in the console file when the sample CGI script is executed.

```
 OTHER    CGIEXEC F1  V 80  Trunc=80 Size=14 Line=0 Col=1 Alt=0

00000 * * * Top of File * * *
00001 /**/
00002
00003 'CGI GETVAR REMOTE_ADDR     (VAR IP'
00004 'CGI GETVAR SERVER_NAME     (VAR SERVER'
00005 'CGI GETVAR SERVER_PORT     (VAR PORT'
00006 'CGI GETVAR SCRIPT_NAME     (VAR SCRNAME'
00007
00008
00009 msg1 = 'Running 'scrname' from Port 'port' on host 'server' for 'ip
00010 'CGI LOG (VAR MSG1 '
00011
00012 'CGI WRITE HEADER (STRING Status: 204'
00013
00014 Exit
00015 * * * End of File * * *
```

Figure 73. Sample VM:Webgateway CGI Script to Log an Entry

```
10:43:44 WEBSRV1 55 VIWLOG077OI 77 .Running /jf/execute/other.cgiexec from Port 81 on host WTSCPOK for 9.12.14.157.
10:43:44 WEBSRV1 55 VIWLOG0771I 78 .9.12.14.157 - - ·07/Nov/1996:15:43:44“ 'POST /jf/execute/other.cgiexec HTTP/1.0'.
10:43:44 WEBSRV1 55 VIWLOG0773I 78 .-  83.
```

Figure 74. Sample VM:Webgateway Log Entries from a CGI Script

6.8.6 VM:Webgateway Imagemaps

The VM:Webgateway server supports imagemaps that use rectangles, polygons, circles, points, and a default. VM:Webgateway supports records in both the CERN and NCSA formats.

The online documentation contains the descriptions of each of the shapes that may be used.

The default name for the imagemap CGI program is *imagemap*. This can be found in the /VM:Webserver directory. This program is used by a user who has an imagemap in the HTML that is being written. The CGI will use the information passed to it and determine which location will be displayed next on the browser.

6.8.7 VM:Webgateway CGI Scripts

CGI scripts are supported by VM:Webgateway. A CGI script is an EXEC that has been written to handle a certain situation. The EXECs are written in REXX and may implement either a VM:Webgateway CGI command-based program, or a CMS Pipeline stage-based program. Other languages may be used if they are called from within a REXX stub.

Because a CGI program is an EXEC running in the VM:Webgateway SVM, you should avoid long-running tasks. Even though VM:Webgateway provides multitasking support, CGIs may make process calls to single tasking operations, such as a CMS or CP service. These calls will block the execution of other tasks. In addition, the VM:Webgateway SVM must have access to any data that the CGI program may try to obtain. You should take caution when allowing

users to serve CGI programs. By doing so, the user has access to all system privileges the server owns.

VM:Webgateway Dynamic Worker Machines address these issues and are discussed in 6.8.8, “VM:Webgateway Dynamic Worker Machines” on page 140.

Figure Figure 75 on page 139 illustrates how VM:Webgateway’s multiple CGI environments interrelate. The product code running in the server, VMWEBSRV, coordinates the execution of multiple CGIs simultaneously. These CGIs may be any of the following:

- A CGI controlling a full screen 3270-based application running in a CMS end user’s user ID that is logged on by VM:Operator under the CGI’s control
- A CGI controlling a line mode application running in a CMS end user’s user ID that is logged on and under the CGI’s control
- A CGI and application running isolated from other CGIs and applications on a worker machine
- A FILTER and optional application running isolated from other FILTERs and applications on a worker machine
- A CGI and application running completely within the VM:Webgateway server

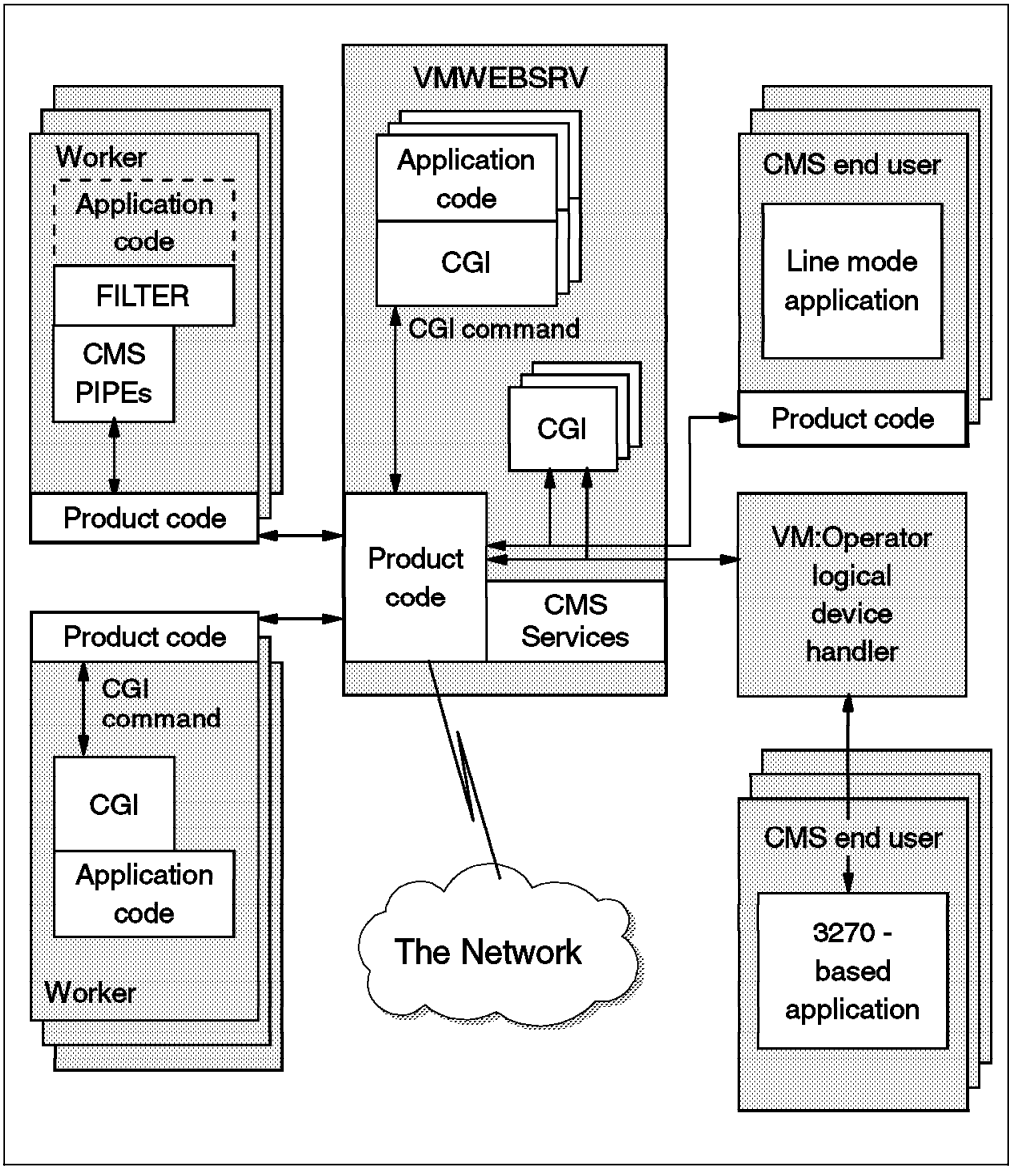


Figure 75. VM:Webgateway CGI Environments Interrelationships

Before a user can serve CGI programs, the system administrator must authorize the user to do so. See topic “VM:Webgateway System Administrator Commands” on page 128. In addition, the system administrator will authorize which file types that user may serve.

The following list shows the commands that are supported within VM:Webgateway. See the online documentation for additional information.

CGI EMSG This allows the CGI script to issue Error Messages. Messages 600-699 are reserved for site-written CGI programs.

CGI GETVAR This allows the CGI program to access variables, such as script name, and path.

CGI LOG This allows the CGI program to cut a log entry.

CGI READ This allows the CGI program to read the data posted from a document.

CGI URLDECODE This decodes the information passed from a form (a set of HTML variables) or a URL encoded string (for instance, the query string from a URL). It creates a stem containing the decoded results.

CGI WRITE This allows the CGI program to write a document that is displayed by a browser.

6.8.8 VM:Webgateway Dynamic Worker Machines

VM:Webgateway Dynamic Worker Machines, or *slaves* allow you to:

- Eliminate the need for multiple Web servers on a single port
- Reduce demand and CGI and FILTER blocking on your primary server
- Exploit multiple processor hardware
- Simplify TCP/IP configuration and network changes
- Eliminate the risk of running *untrusted* user CGIs on the primary server
- Assign surrogate privileges based on the authenticated VM user ID

VM:Webgateway autologs worker machines to process CGIs as required. When a CGI is invoked, VM:Webgateway determines if a *hot* worker is available to execute the CGI. If not, it autologs a new worker machine and dispatches the CGI. Once the CGI has completed, the worker remains autologged and is added to the *hot* worker list ready to process additional CGIs. See the online documentation for VM:Webgateway for more information. This topic is also covered extensively in

- *Web-Enabling VM Resources*, SG24-5347 (which will be available at a later date).

6.8.9 Converting from Webshare to VM:Webgateway

If you already have Webshare running, and are going to convert to the VM:Webgateway product, a utility is provided that can convert your current Webshare configuration file to VM:Webgateway format. On the VM:Webgateway SVM 192 disk, there is a utility named VIWCVTWC EXEC. This creates a file named VIWINIT EXEC. Running this EXEC will update the VM:Webgateway configuration with the WEBSHARE configuration information.

If you are running multiple servers, the VIWINIT EXEC must be run multiple times. Each time, the EXEC should be modified to include the server that you are configuring.

An additional utility that is provided is the VIWCVTFL EXEC. This EXEC will take the WEBSHARE FILELIST file and create the appropriate VM:Webgateway DIRMAP files. This utility must be executed through the CMS interface. Using the line-mode or CP interface will result in a “FILE NOT FOUND” message. The resulting DIRMAP file can then be placed in the appropriate location and used by the VM:Webgateway server.

Figure 76 on page 141 shows the process to use when converting from Webshare to VM:Webgateway.

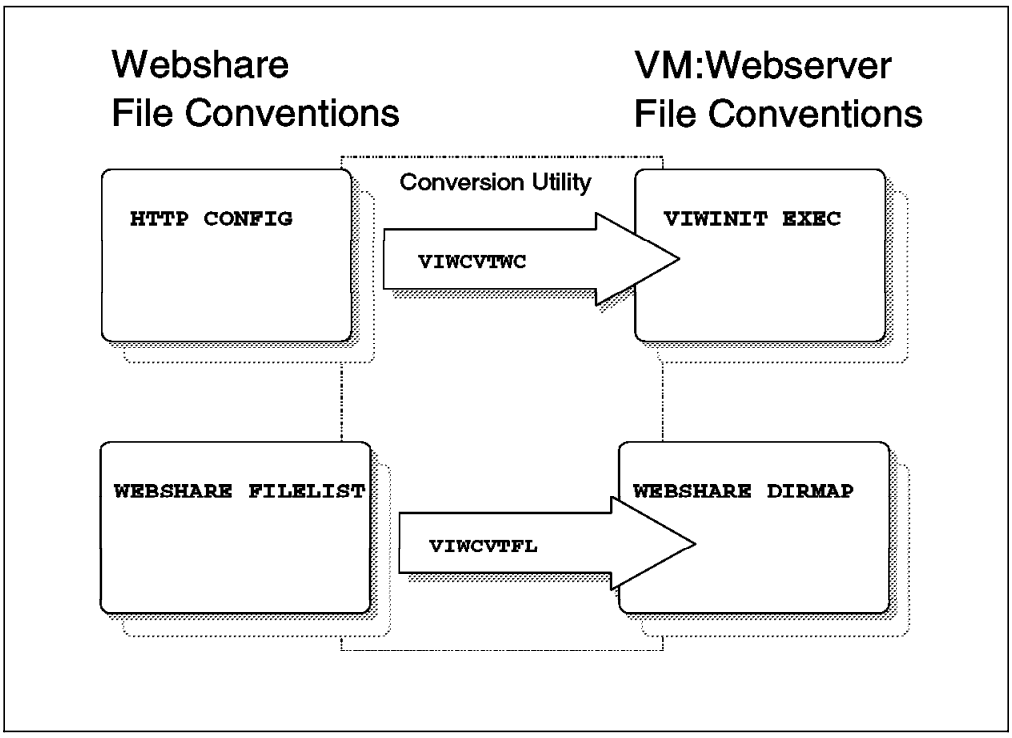


Figure 76. VM:Webgateway to WEBSHARE Conversion Utilities

6.8.10 Client Pull and Server Push with VM:Webgateway

The following concepts are not HTML standards, but extensions that must be supported by either your server, client, or both.

Client Pull with VM:Webgateway

For client-pull, consider the following statement in an HTML file:

```
<META HTTP-EQUIV='Refresh" CONTENT=3> <h1>This is a client-pull demo</h1>
```

If your browser supports this, it will *pull* the document from the server every three seconds, forever. To stop it, you must request another page. No special support is required on the Web server. It is also possible to serve different documents rather than the same document over and over, but that is outside the scope of this introduction.

Server Push with VM:Webgateway

Server push is more complex. The easiest way is to do it from a CGI; this gives you more control and requires no configuration changes in you server (other than the ones you would need to run CGIs to begin with). Consider the following CGI that does server-push:

```
/* SRVPUSH CGIEXEC (environment SVMEXEC) */ 'CGI WRITE HEADER (STRING' ,
'Content-type: multipart/x-mixed-replace;boundary=GOOBER'

cmd_doc = 'CGI WRITE DOCUMENT (TRANSLATE EBCDIC ASCII CRLF STRING'

cmd_doc '--GOOBER'
cmd_doc 'Content-type: text/plain'
cmd_doc ''
cmd_doc 'Found 1 peanut'
cmd_doc '--GOOBER'
```

```
cmd_doc 'Content-type: text/html'
cmd_doc ''
cmd_doc '<h1>Found 2 peanuts</h1>'
cmd_doc '--GOOBER--'

exit
```

Notice that there is only one space in the Content-type line, and that there must be a null line between the Content-type line and each part of the document. Also notice that each boundary must be preceded by exactly two hyphens, and the last boundary is followed by two hyphens.

The second way to do server push does not involve CGIs, but you do have to configure VM:Webgateway and write some tricky HTML. First, pick a unique file type for your server-push HTML and add it to VM:Webgateway:

```
CONFIG FILETYPE ADD HTMLPUSH FILE TRANSLATE EBCDIC ASCII
   CONTENT-TYPE multipart/x-mixed-replace;boundary=GOOBER
```

Next, create your tricky HTML with file type HTMLPUSH:

```
--GOOBER
Content-type: text/plain#$
Here's my first cracker
--GOOBER
Content-type: text/html#$
<h1>Here's my second cracker<h1>
--GOOBER--
```

Notice the same features as before - only one space on the content line, and the boundary hyphens. However, notice that this time there is no blank line between the Content-type and the document line. That is because XEDIT does not support empty lines (length 0) - there is always a space - and the spec requires a bare CR/LF between the Content-type and the content. So to compensate, put XEDIT into VERIFY HEX mode and add an extra X'0D25' (EBCDIC CR/LF) onto the end of each Content-type line (as symbolized by the "#$" above).

For a complete discussion of dynamic documents, look at this web site:

```
http://home.netscape.com/assist/net_sites/pushpull.html
```

6.9 VM:Webserver OfficeVision Interface

A new product by Sterling Software, Inc. is VM:Webserver OfficeVision Interface. This is an extension to VM:Webgateway and provides a Web browser interface for the OfficeVision/VM (OV) calendar system. Now, besides conventional access from your VM user ID, you can access OV through your preferred Web browser. By exploiting popular Web browsers, VM:Webserver OfficeVision Interface provides a user-friendly graphical interface for OV, replacing the traditional 3270 character-based interface.

6.9.1 General Features of VM:Webserver OfficeVision Interface

The following are a cross section of the features of VM:Webserver OfficeVision Interface Release 1.4:

- Access many of the VM/OV calendar interfaces. See 6.9.2, "VM/OV Calendar Support in VM:Webserver OfficeVision Interface 1.0" on page 143 for more information.

- Schedule meetings, including:
  - Automatically find available times
  - Schedule recurring meetings
  - Send meeting notices
- Send and receive mail, including:
  - Send notes, with or without attachments, to OfficeVision/VM users, VM users not registered in OfficeVision/VM, and users accessible through the Internet
  - Receive, display, save, and delete mail from your inbox
  - Reply and forward notes and meeting notices, with or without attachments
  - Create mail folders
  - Display and delete mail you have stored in mail folders
  - Add a meeting to your schedule or the conference room schedule
  - Accept or defer mail management actions set up for you by a delegated OfficeVision/VM user
- Toleration of following OfficeVision/VM PRPQs:
  - OfficeVision/VM Enhanced Mail Addressing for VM/ESA (5799-FPL)
  - OfficeVision/VM Enhanced Calendar/VM (5799-FFL)
  - AWAY Facility/VM (5799-FLP)
- Make use of Internet style addresses
- Customize the main menu
- Refresh your view of an inbox or folder
- Support dates in the range of January 1, 1970 through December 31, 2069 in OfficeVision/VM 1.4 (January 1, 1951 through December 31, 2050 for earlier releases)
- Maintain the OfficeVision/VM address book
- Display header fields for notes
- Access your OfficeVision/VM session through your browser (to work with facilities of OV/VM that are not addressed by VM:Webserver OfficeVision Interface)

6.9.2 VM/OV Calendar Support in VM:Webserver OfficeVision Interface 1.0

VM:Webserver OfficeVision Interface Release 1.0 was evaluated for this documentation. Although the e-mail function was not available in Release 1.0, it is available at the current level, Release 1.4. To see how VM:Webserver OfficeVision Interface appears, see Figure 77 on page 144.

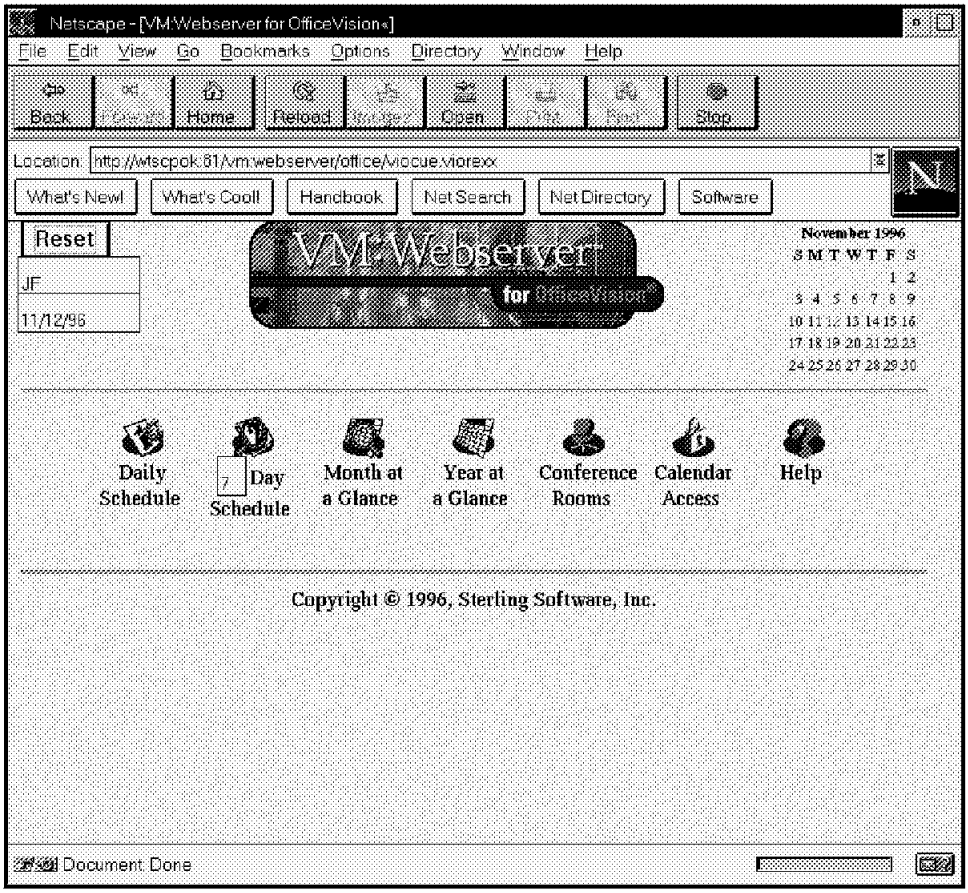


Figure 77. VM:Webserver OfficeVision Interface

With VM:Webserver OfficeVision Interface, you are able to use your browser to access many of the calendar options on the host OV system through the World Wide Web. All of the default calendar menu items are there except for schedule a meeting and add company holidays. You are able to perform the following:

- Check your Daily Schedule
- Check your Daily Schedule for the next 7 (default value) days
- Check your Schedule for the Month
- Check your Schedule for the Year
- Check Conference Room Schedule
- Check Calendar Access
- Access the online documentation

Printing of the weekly calendar is available by using your browser's print function. When displaying the seven day schedule (or whatever number of days you specify), select the print option within the browser.

Because you are using the browser on your workstation, you are able to use a select and click method to work with your calendar. All of the maneuvering that is accomplished with PF-keys on the host is accomplished with the mouse on the workstation. Figure 78 on page 145 shows how you can add a user to the access list for your calendar. This also allows you to select what type of access the user will have.

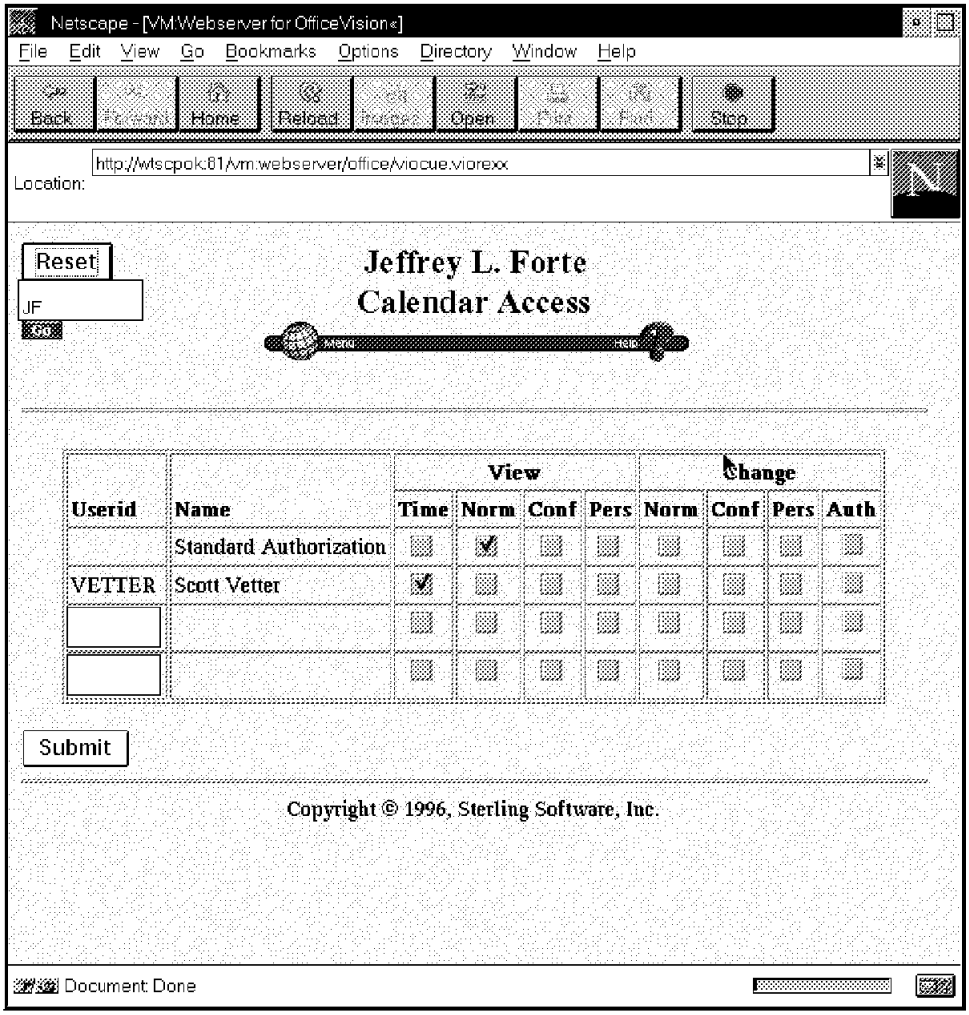


Figure 78. VM:Webserver OfficeVision Interface Calendar Access

As with the host level of OV, recurring events can be scheduled with VM:Webserver OfficeVision Interface. When checking the daily events, check the item that you wish to work with. Select **Schedule recurring events** and click hp2.Submit. This will bring up the page displayed in Figure 79 on page 146. Do not forget to select **Save all changes** and **Submit**.

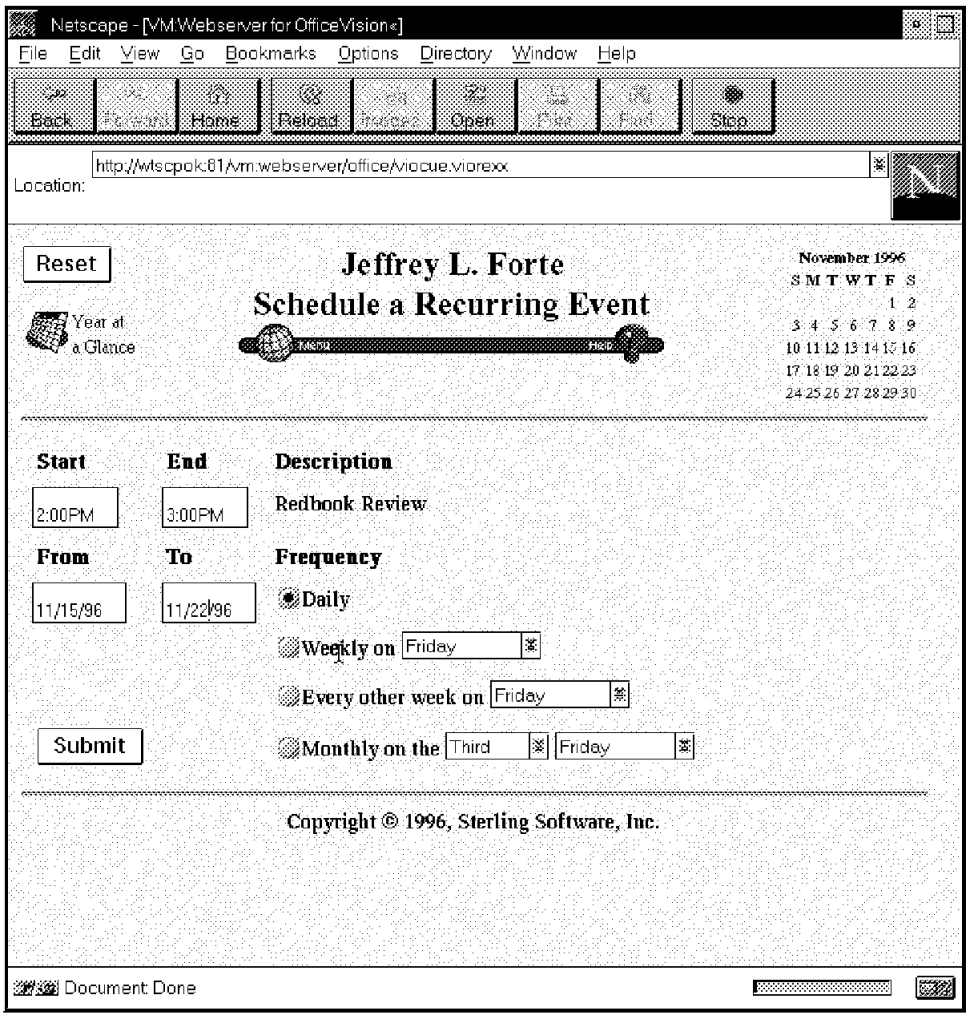


Figure 79. Scheduling a Recurring Event with VM:Webserver OfficeVision Interface

Along with scheduling a recurring event, you may also delete, delete recurring, copy, and move events. All of these functions can be done from your browser.

6.10 Summary of VM:Webgateway Features

VM:Webgateway is based on an assembler kernel ported from existing Sterling Software, Inc. products. This section contains a summary of some of the features and observations that were obtained during the installation and testing of this product.

The kernel used in VM:Webgateway makes use of multithreading and a multiprocess sockets interface to provide a multitasking environment. The goal of Sterling's VM/ESA Web server is to eliminate the need for multiple servers. This is fully realized in the product when the dynamic worker facility is made use of for all long-running and user ID synchronous work. Dynamic worker machines eliminate most of the need for additional Web server machines. These machines will handle long-running CGIs and free the Web server from blocking during single-threaded processes.

VM:Webgateway has a well developed Web browser interface (GUI) that provides remote administration capabilities. The GUI requires little knowledge of VM, and

contains information to get the job done without a manual. When administration is performed directly on VM/ESA, the command interface is flexible. This flexibility should accommodate automated operations, homemade utilities, and personal preferences.

When help is required, the documentation is complete and easy to understand.

Through the use of ACCESS records in the DIRMAP configuration files, VM passwords and user IDs can be used to control the access to data. You can also disable user pages, if you wish to control all of the site content of your server. If you store your data on BFS or SFS, you can use file system settings to further control what data is allowed for Web access.

Migrating your Webshare control files through provided conversion utilities is direct, though one-way. Webshare is an excellent way to gain experience before making a capital investment in a commercial Web server.

Sterling′s VM:Webgateway allows you to make almost all configuration changes without a re-IPL of the server virtual machine. During this project we found that in a multiple-server environment, an orderly shutdown of a single server did not interrupt service. TCP/IP would route new requests to the next available server.

An optional product, VM:Webserver OfficeVision Interface, integrates VM:Webgateway with OfficeVision calendaring. A company′s OfficeVision component can now interoperate with any end-user workstation.

Note that the information contained here is only a brief summary of some of the features. See the VM:Webgateway online documentation for more information.

Chapter 7. VM/ESA Web Server Implementation

Before establishing a VM Web service site, you need to understand how the VM Web servers are implemented on VM. This may be completely different from implementations on other operating systems. The knowledge about the VM specifics is essential to the setup of your Web site and its follow-on administration and in assisting application providers in obtaining the full benefits of a VM/ESA-based Web service.

The information in this chapter is based on the most basic VM/ESA Web implementation, using Webshare. The basic techniques shown can easily be transferred to one of the commercial servers. A fictitious scenario is used to help illustrate the examples.

7.1 A Standard Web Service Scenario

Assume you have to provide an intranet Web service to all of the clients belonging to a fictitious company. Through the Web site, three applications are provided, of which one is restricted to a defined group of people.

The overall goal of all implementation recommendations is to achieve a good performing Web service for the least possible administration cost, while permitting a mix of protected and nonprotected applications. Figure 80 shows a file structure for the scenario.

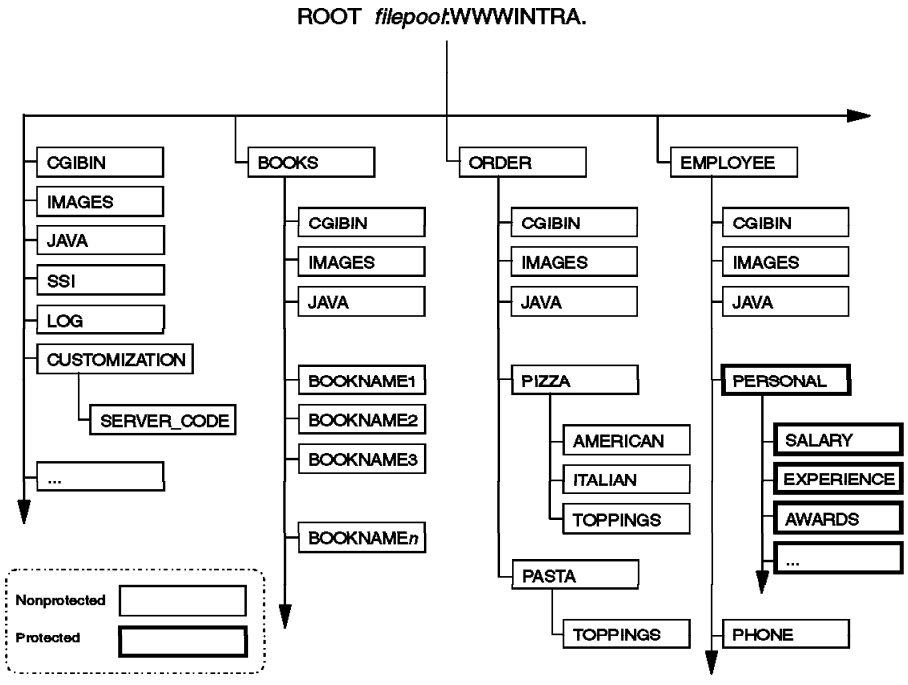


Figure 80. The Web Server Directory Topology

7.2 URL Hierarchical Addressing Scheme

The basic structure of an HTTP request coming from a WEB browser is a URL address. Any URL address has a hierarchical directory structure.

Using the VM Web servers, you can implement the URL hierarchy with the grouping mechanism of FILELIST or DIRMAP files or by exploiting the SFS.

Experience has shown that maintaining a FILELIST or DIRMAP based grouping structure over time requires devoted attention. Web applications tend to have rapidly changing requirements and content.

Using the SFS to implement URL hierarchical addressing allows you to exploit the following features at a very low cost:

- Multiple homepages with no administration overhead
- Ease of debugging
- Ease of navigation
- Delegation of security (*trust*)
- Data space exploitation for high-speed serving
- Local multiple-write access from multiple WEB servers
- Remote data sharing through remote SFS capabilities
- Remote multiple-write access through coordinated-resource recovery
- Maintenance of WEB served data through remote SFS from any other connected VM/ESA system. For a Web data provider, this is the most transparent, easy-to-use, and low-cost maintenance currently possible.

For the administrator and application owners, a URL to SFS directory structure mapping means optimal built-in transparency and avoidance of misinterpretations. Navigation becomes easy.

Figure 81 shows URL to SFS directory mapping.

```
URL: http://server/application/subdirs/fn.ft
SFS: filepool:.serverroot.application.subdirs. fn ft

URL: http://www.company.com/order/pizza/select.html
SFS: MYWEBSFS:.WEBSRV.ORDER.PIZZA. SELECT HTML
```

Figure 81. Hierarchical File Structure Mapping

TIP

Exploit the SFS or BFS file system to map the URL hierarchical file addressing scheme. Minimize use of FILELIST or DIRMAP navigation as much as possible.

You may want to use FILELIST navigation instead. A FILELIST or DIRMAP file defines a structure similar to a subdirectory. The files listed in the FILELIST are members of that subdirectory. Note that all of the files reside on the same minidisk or SFS or BFS file system directory, with other data. The FILELIST files are mainly used as a way to implement the mapping of a flat file system, such as the CMS minidisk file system, to the hierarchical structure of a URL address. DIRMAPs are used in a similar manner by VM:Webgateway.

Figure 82 on page 151 shows how a WEBSHARE FILELIST is used to locate a CPQ CGI. This figure is the basis for further discussion.

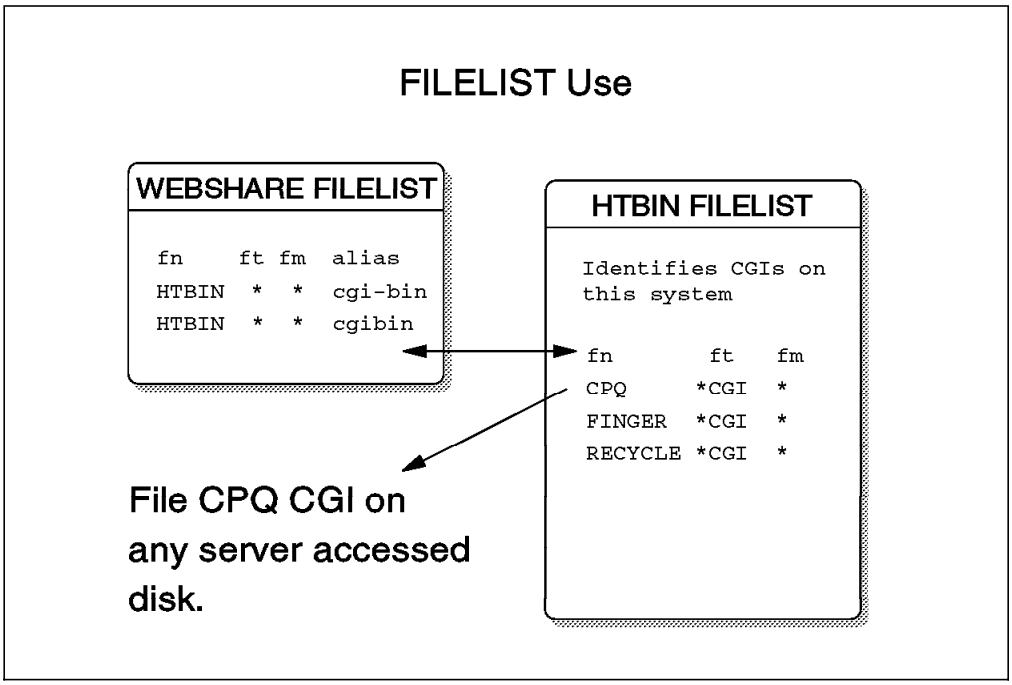


Figure 82. Webshare FILELIST Use

In WEBSHARE FILELIST, a required file, you define the URL hierarchy *cgi-bin* or *cgibin* to be mapped to the Web server hierarchy HTBIN. When specifying a URL, such as:

```
http://server/cgibin/cpq?users
```

the Web server will first translate cgibin into HTBIN. Then the file HTBIN FILELIST is searched for the CPQ entry, using the pattern *CPQ *CGI **. A LISTFILE is issued to find an existing file for this file name, file type, and file mode pattern. Here, the CPQ CGI is located on Webshare's root address file mode. Because a match was found, the CGI called CPQ is executed with any parameters that were passed to it.

Webshare has a requirement that CGIs reside on CMS accessed disks. The commercial servers allow you to create a subdirectory named HTBIN, in which you place the CGIs of your server, and will access it automatically.
VM:Webgateway also allows CGIs to be resident anywhere in a URL tree to be served. This is the same work as for an initial minidisk-based setup, but the grouping is performed automatically. There is no maintenance cost later when adding a new CGI.

7.3 Naming Conventions

Good navigation makes it easy for a visitor to locate information. For this, you might want to pass along significant hierarchical naming for the URL address.

7.3.1 Directory and File Naming

With the SFS, this is easy to achieve, since SFS subdirectories can be up to 16 characters long. Note, however, that there are naming conventions for SFS directories. As in any operating system, only some characters can be used to name a hierarchy.

SFS Directory Naming Conventions

- The complete directory name (also called *dirname*) is made up of its parent directory's name and any subdirectory names, with each name separated by a period.
- A directory name can be from one-to-sixteen characters.
- The valid characters are A-Z, a-z, 0-9, $, #, @, and _ (underscore).
- Two or more subdirectories may have the same name if each belongs to a different parent directory.

Also, the CMS file system describes a flat file by a file name and a file type. When you create a CMS file, you can give it any file name or file type you wish, provided you follow some naming rules.

CMS File Name and File Type Conventions

- The file name and file type can each be from one to eight characters.
- The valid characters are: A-Z, a-z, 0-9, $, #, §, +, - (hyphen), : (colon), and _ (underscore).

Note: Lowercase letters are valid in a file ID for use within the CMS file system. However, most CMS commands do not support file IDs that contain lowercase letters.

7.3.2 URL Alias or Nickname

If you like to use other characters, such as the commonly used minus sign (-) in a UNIX environment, you can create an *alias* (or nickname) by adding it to a FILELIST or DIRMAP file. This file must be placed one directory level higher in the hierarchy to be recognized by the Web server. It is advisable not to use the FILELIST method; you should restrict your naming to CMS conventions.

Note: Some Web servers restrict the length of an alias to eight bytes as well, while others allow for arbitrarily long alias names.

Tip

To avoid the need for a URL alias or nickname, use the CMS naming convention offered by the SFS.

7.4 Locating Files Within a File System

There are several techniques available in today's Web servers that are used to categorize and reference data. The following sections introduce some of the techniques.

7.4.1 Relative URL Addressing Inside an Application

A relative URL is a way of navigation using only part of a URL. Using relative URL addressing is part of the design of a Web application. But to achieve the goal of a low-cost Web service is also part of the Web administrator considerations.

To stay portable and independent from Web server changes within an application, use relative URL addressing. Also, when referencing server functions, address these relative to the root directory of the server. Doing so minimizes the administration of both the application and the Web server. The following sections discuss this concept in greater detail.

Given the following URL:

```
http://www.company.com/order/pizza/select.html
```

the following applies:

http://	This identifies the HTTP protocol to be used.
www.company.com	This identifies the server address to be resolved using a name server.
order/pizza/	This identifies subdirectories under the defined root of the addressed server.
select.html	This identifies the file to be addressed.

Using this example, you can use relative addressing by leaving out the protocol and the server address in all references made inside the application. To do so, the following addressing may be used in front of a file:

	No character means inside the currently active path (subdirectory).
../	This addresses a subdirectory one level up from the currently active path.
../../	This addresses a subdirectory two levels up from the currently active path. Any additional ../ moves the address up another level.
./subdir/	This addresses a subdirectory one level deeper than the currently active path.
subdir/	This also addresses a subdirectory one level deeper than the currently active path. This illustrates that the specification of ./ to define the current active path is optional.
../beside/	This addresses a subdirectory beside the currently active path. It replaces the lowest level path definition with the one specified. In this example, the subdirectory is named *beside*.
/	This addresses the servers root directory. Any definition that starts with a / will do so, so alternatives like /../ and /../.. all have the same effect.
/SSI/header.html	This addresses the file header.html in the SSI subdirectory of the WEB servers root.

Figure 83 on page 154 and Figure 84 on page 154 show a directory structure and a Web page that uses relative URLs to traverse the file structure.

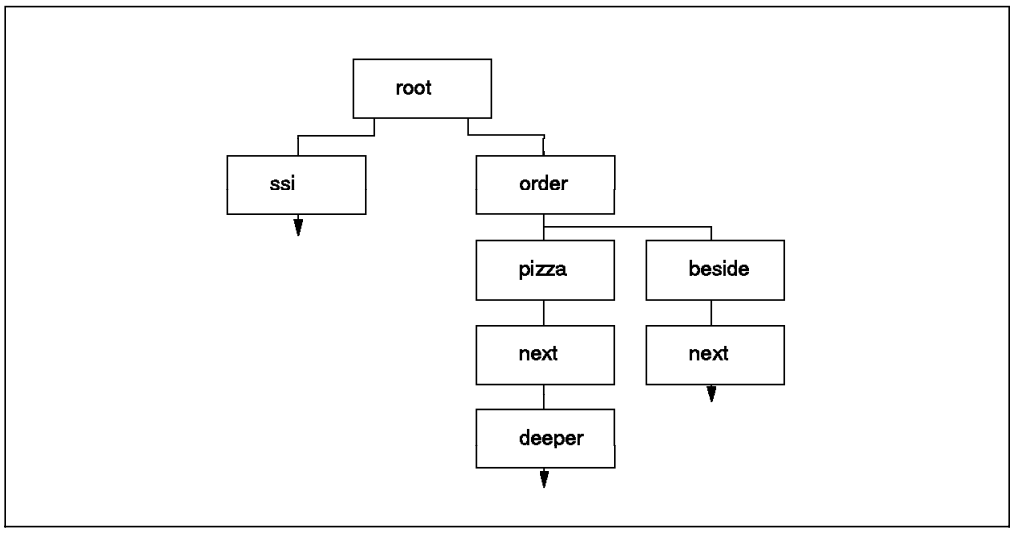


Figure 83. Sample Server Directory Structure

```
00000 <html>
00001 <!-- Useful sample HTML for relative URL tests  -->
00002 <head>
00003 <title>Sample HTML to verify relative URLs</title>
00004 <hr>
00005 </head>
00006 <p><font size=+1> Relative upward URLs </font>
00007 <p>The following should go to the same directory:
00008 <a href="testt.html">"testt.html"</a>
00009 <br>The following should go to one level up:
00010 <a href="../testt.html">"../testt.html"</a>
00011 <br>The following should go to two levels up:
00012 <a href="../../testt.html">"../../testt.html"</a>
00013 <br>The following should go to three levels up:
00014 <a href="../../../testt.html">"../../../testt.html"</a>
00015 <br> The following should go to four levels up:
00016 <a href="../../../../testt.html">"../../../../testt.html"</a>
00017 <hr>
00018 <p><font size=+1> Relative downward URLs </font>
00019 <p>The following should go one level down:
00020 <a href="./next/testt.html">"./next/testt.html"</a>
00021 <a href="next/testt.html">"next/testt.html"</a>
00022 <br>The following should go two levels down:
00023 <a href="./next/deeper/testt.html">"./next/deeper/testt.html"</a>
00024 <a href="next/deeper/testt.html">"next/deeper/testt.html"</a>
00025 <hr>
00026 <p><font size=+1> Relative beside URLs </font>
00027 <p>The following should go one level sideways:
00028 <a href="../beside/testt.html">"../beside/testt.html"</a>
00029 <br>The following should go one level beside, one levels down:
00030 <a href="../beside/next/testt.html">"../beside/next/testt.html"</a>
00031 <hr>
00032 <p><font size=+1> Root addressing samples </font>
00033 <p><a href="/testt.html">"/testt.html"</a>
00034 <br><a href="/../testt.html">"/../testt.html"</a>
00035 <br><a href="/../../testt.html">"/../../testt.html"</a>
00036 <p>Go to root/SSI for file header.html
00037 <br><a href="/ssi/header.html">"/ssi/header.html"</a>
00038 </body>
00039 </html>
```

Figure 84. Sample HTML Page to Show Relative URL Usage

7.4.2 FILELIST Interpretation

All Web servers allow alias settings. This is handy when a URL subpath definition contains characters not allowed for a VM directory. Aliases are also used to translate disallowed characters or file names and file types that exceed CMS's eight character limit. If you cannot avoid using aliases (for example, after moving an application from a non-VM operating system into a VM file system other than BFS), you must use FILELIST (or, in the VM:Webgateway case, DIRMAP) files. This section discusses details of how FILELISTs work, which is not well documented by Webshare-based product documentation. While VM:Webgateway's DIRMAP files are similar, they do differ in the details. Please refer to VM:Webgateway's online documentation for a complete description of how DIRMAPs function.

Figure 85 shows a sample FILELIST. It aliases (replaces) the specification cgi-bin in a URL address to HTBIN.

```
00001 *     fn       ft       fm path     name
00002 *     xxxxxxxx xxxxxxxx xx xxxxxxxx "xxxxxxxx"
00003 *
00004       HTBIN    *        *  cgi-bin
00005 *
```

Figure 85. Sample Fragment in WEBSHARE FILELIST for an ALIAS

Note: The first column must always contain a blank.

FILELISTs have other uses. Assume you find in the subdirectory MYPOOL:ORDER.WEBSHARE. the file WEBSHARE FILELIST with the contents shown in Figure 86.

```
00001 *     fn       ft       fm path     name
00002 *     xxxxxxxx xxxxxxxx xx xxxxxxxx "xxxxxxxx"
00003 *
00004       PIZZA    *        *  pizzaorder   "Pizza Ordering application"
00005       EMPL     *        *  employee     "Company organization"
00006       MOVED    *URL     *  oldappl
00007       *        HTML     *
00008       ORDER    *CGI     *  getmeone
00009 *
```

Figure 86. Sample Fragment of WEBSHARE FILELIST in User Root

Note: The first column must always contain a blank.

The statements are explained in the following:

- If a request is received by the server for:

  `http://www.company.server/~order/pizza/select.html`

  the statement `* HTML *` is read, and file SELECT HTML located in the `MYPOOL:ORDER.PIZZA.` directory (or in PIZZA FILELIST if that exists) is sent back to the browser.

- If a request is received for:

  `http://www.company.server/~order/pizzaorder`

the statement PIZZA * * pizzaorder is used. The defined path is an alias, pointing to the file PIZZA FILELIST. Its contents or a list of the files in subdirectory MYPOOL:ORDER.PIZZA. will be sent back to the browser.

- If a request is received for:

  http://www.company.server/~order/employee

  the statement EMPL * * employee is used. The defined path is an alias, pointing to the file EMPL FILELIST. Its contents, or a list of what is in the subdirectory MYPOOL:ORDER.EMPL. will be sent back to the browser.

- If a request is received for:

  http://www.company.server/~order/getmeone?A+B+C

  the statement ORDER *CGI * getmeone is used. Its interpretation is to execute the CGI named ORDER CGI if it exists and if the user ID ORDER is authorized to execute CGIs. In Webshare, this is done by adding the user ID to the CGIUSERS variable in the configuration file. The ORDER CGI will get passed the parameters A B C.

- If the following request is sent to the server:

  http://www.company.server/~order/oldappl

  the statement MOVED *URL * oldappl is used. Its interpretation is to assume the application has been relocated. To assist the user, the following files will be interpreted:

  – MOVED WEBLINK
  – MOVED HOTLIST
  – MOVED URL

  They all should contain URL title. All lines found will be interpreted as such. These files must be accessed by the Web server. Figure 87 and Figure 88 show what a sample of a MOVED URL and a MOVED HOTLIST could look like.

```
00001 "http://www.newserver.com/~order/APPL2/index.html"  "Moved application: Pubs ordering"
00002 "http://www.myserver.com/employee/"  "Moved application: employee org of subcompany"
```

Figure 87. Sample File MOVED URL

```
 "http://server/pubs/catalog/index.html"  "List of available books"
 "http://www.ibm.com/"                "IBM's home page"
```

Figure 88. Sample File MOVED HOTLIST

All four entries will be displayed as HOTLINKs. They warn the user to renew existing bookmarks, or provide guidance to other places. If a request is received for:

http://www.company.server/~order

no statement will be matched. The browser will be sent a list of all files matching the generic statement * HTML along with two subdirectory icons pointing to PIZZA FILELIST and EMPL FILELIST. Both icons will have the description taken from the name field defined in the WEBSHARE FILELIST file.

If an INDEX HTML file exists, the list of the files contained in INDEX HTML will be sent back to the Web Browser instead.

7.4.3 FILELIST Structures

Creating hierarchical structures using FILELIST files on a single minidisk can become complicated. Transparent mapping of URL addresses to FILELIST content requires repeated maintenance. Even if you think the setup is not expensive, maintaining such a navigation can become laborious because data is constantly changing. If there is no need for aliasing, stay away from the hierarchical approach using FILELISTs, and exploit the Shared File System. Staying with minidisks and using FILELISTs allows for the same functions, but at a much higher administration cost.

The easiest way to navigate is with a one-to-one relationship in SFS directory structures using only INDEX HTML files to guide a client to the data they are searching for.

Note: The topics discussed in this section apply to EnterpriseWeb/VM and VM:Webgateway, with small implementation differences.

7.5 Security and Performance Issues

There are several security issues that will be discussed. These issues center around the CGI and the applications that use a CGI.

These topics are also covered extensively in

- *Web-Enabling VM Resources*, SG24-5347 (which will be available at a later date).

7.5.1 How to Restrict Running CGIs

The default setup in VM Web servers is the use of FILELIST files. The user IDs that own the file space where the CGI resides (as data) must be explicitly named so that CGIs are executable.

In Webshare, this is done by the definition of the CGIUSERS parameter:

```
CGIUSERS = WWWINTRA USER4 USER8
```

In VM:Webgateway, this is done with the CONFIG CGIUSER command:

```
CONFIG CGIUSER ADD WWWINTRA filetype1
CONFIG CGIUSER ADD WWWINTRA filetype2
CONFIG CGIUSER ADD USER4    filetype1
CONFIG CGIUSER ADD USER4    filetype2
CONFIG CGIUSER ADD USER8    filetype1
CONFIG CGIUSER ADD USER8    filetype2
```

7.5.2 CGIs Effect on a Web Server

The reason CGIs should be looked at closely is the way they run under VM Web servers:

- CGIs can block the server.
- CGIs can change the server′s environment.
- CGIs run with the privilege of the Web server.

Because of these restrictions, the next logical evolution for a VM Web server is to move the execution of CGIs off the Web server virtual machine, for example, to Web server slave virtual machines. This has been realized by VM:Webgateway. See section 6.8.8, “VM:Webgateway Dynamic Worker Machines” on page 140 for a look at how Sterling Software, Inc. has bypassed this limitation.

The reason CGIs can block a server are CMS- and CP-related. For example, many CMS services are often not multitasking enabled. This means that the server may be put into a virtual machine synchronous wait state because a CGI is waiting for an event. There is no way that the PIPELINE scheduler or the CMS multitasking can gain back control by themselves. The wait is completed when the event occurs. The same is true for CP diagnose calls, which can put the virtual machine in wait state.

The reason CGIs can change the server′s environment is also CMS- and CP-related. Any virtual machine has common CMS and CP settings. Sometimes these need to be changed to be able to program a behavior required by a CGI program. Care must be taken that all changed parameters are restored at all possible CGI loss of control and exit points. Poorly coded CGIs may also affect the PIPELINE environment and thus bring the server down.

The reason CGIs run with the privilege of the Web server is CP-related. By default, any virtual machine receives the privileges it can use through the CP directory settings and the ESM. Even the Web server can issue, for example, a DIAGNOSE X′D4′ (alternate user ID support) to switch some CP privileges to those of an identified user ID, and other authorizations given to the Web service virtual machine still apply. Because of this, you should not give any authorization to the Web service machine other than the ones you would give to any CGI programmer with access to the server.

You may want to inspect any user-provided CGI carefully to avoid privilege misuse or service impact.

7.5.3 Protect an Application

The way to protect access to an application or part of the data is in the way the VM Web servers implement their security.

In the example shown in Figure 80 on page 149, anything in the EMPLOYEE application which is in and below the subdirectory PERSONAL should be protected. Using this, all accesses, including data and CGIs, should be protected. Since access restriction rules by IP address or domain name are easily foiled, authenticate the users by their VM user IDs and current passwords, checking these with the built-in VM security (through DIAGNOSE X′84′, or if an ESM is installed, DIAGNOSE X′A0′).

Webshare does not provide these security features. You must purchase a commercial server for this function. Locally developed security on Webshare is possible, for example checking the REALM field, but is beyond the scope of this publication.

In EnterpriseWeb/VM, user ID and password validation is performed by statements in HTACCESS files. Since we recommend SFS exploitation, and FASTPATH is possible in EnterpriseWeb/VM, you have to place a file named

$EWEB HTACCESS in the directory *filepool*:WWWINTRA.EMPLOYEE.PERSONAL. This file may have the contents shown in Figure 89 on page 159.

```
00001 AuthUserFile *VALIDATE VMDIR
00002 AuthGroupFile PERSONAL HTGROUP =
00003 AuthName Personal
00004 AuthType Basic
00005 AuthLimits 1
00006 <limit>
00007    require valid-user
00008 </limit>
```

Figure 89. Sample $EWEB HTACCESS File

The PERSONAL HTGROUP file defines the users with access, as shown in Figure 90.

```
00001 * The following user IDs are allowed to access the PERSONAL
00002 * application
00003 JF
00004 WIDMAYER
```

Figure 90. Sample PERSONAL HTGROUP File

The Web server will prompt the client for a user ID and password combination related to a REALM of choice. Subsequent requests of the browser will contain this REALM information, so no further prompt will happen on the client.

The protection method validates against the CP directory. If the users are correctly identified, and they belong to the group of users defined, then access is allowed. The user must have a valid user ID and password combination.

In VM:Webgateway, this kind of access is performed by statements in DIRMAP files. Since we recommend SFS exploitation, you must place a file called VMWEBSRV DIRMAP in the directory *filepool*:WWWINTRA.EMPLOYEE.PERSONAL. Figure 91 is a sample of a DIRMAP file.

```
00001 SELECT
00002   REALM VM Userid and password
00003   PASSWORD VMDIR
00004   WHEN USER JF WIDMAYER
00005     ALLOW
00006   OTHERWISE
00007     DENY
00008 END SELECT
```

Figure 91. Sample DIRMAP File

The Web server will prompt the client for a user ID and Password combination related to a REALM of choice. Subsequent requests of the browser will contain this REALM information, so no further prompt will happen on the client. This protection method validates against the CP directory. If the users are correctly identified, and they belong to the group of users defined, then access is allowed. Clients must have a valid user ID and password combination for access.

7.6 User Root Resolution

The following section describes the way a Webshare-based server interprets a URL address when it refers to a user root. User root resolution is more efficient in VM:Webgateway, which uses an indexed database lookup to resolve the user root of a user id to a minidisk, SFS directory, or BFS directory. A user root is indicated when the URL contains a tilde character (~).

Assume a client passed the following URL address to a server:

```
http://www.company.server/~order/pizza/select.html
```

The root, MYPOOL:WWWINTRA.HOME., is defined for the server.

A Webshare-based server performs the following steps:

- Parse out the ~order as the user root.
- Verify if user ID ORDER is in the same file pool as used for Webshare's user root (search for MYPOOL:ORDER).
  - If yes, verify if there is a subdirectory named WEBSHARE (search for MYPOOL:ORDER.WEBSHARE). Display either all the files on there as a generated menu or display the content of a WEBSHARE FILELIST. Any INDEX HTML file will take precedence over the WEBSHARE FILELIST file or auto indexing.
  - If no, step back to the users main directory (access MYPOOL:ORDER).
- Verify if there is a WEBSHARE FILELIST.
  - If yes, display the content of it at the browser.
  - If no, verify if ORDER 191 minidisk is accessible (LINK and ACCESS ORDER 191).
- Verify if there is a WEBSHARE FILELIST.
  - If yes, display the contents of it on the browser.
  - If no, verify if ORDER 192 minidisk is accessible (LINK and ACCESS ORDER 192).
- Verify if there is a WEBSHARE FILELIST.
  - If yes, display the content of it on the browser.
  - If no, ~user ID is not allowed to serve data. This error will be reflected back to the browser.

Tip

You may want to disallow user roots to improve Web service administration and control capabilities.

7.7 Server Side Includes

Server side include (SSI) allows an HTML file to imbed other data such as HTML files. This is a way to reduce data in applications that contain many repeated data pieces. Its main use is achieving a consistent view by imbedding header and trailer parts. The implementation of SSIs is specific to each Web server.

For EnterpriseWeb/VM, the data to be served must be accessed by the Web server all times. Also, no period (.) is allowed between the file name and file type. As a result, any application should refer to SSIs by the file name and file type. Other specifications such as a period between file name and file type or relative URL is not allowed in the release of server tested for this publication. Figure 92 shows the supported and not supported SSI references.

```
src="comphdr html"          supported
src="comphdr.html"          not supported
src="/ssi/comphdr.html"     not supported
```

Figure 92. Sample Specifications of a EnterpriseWeb/VM Server Side Include File

For VM:Webgateway, the data to be served may be defined by any valid CGI variable or URL, within the same security profile, that the server can resolve locally to a static, non-FILTERed file. The syntax of these records for VM:Webgateway was arrived at by analysis of the defacto Internet standards in this area. Figure 93 shows some of the supported SSI formats.

```
<!--#CONFIG ERRMSG="Error including file example.html."-->
<!--#CONFIG SIZEFMT="BYTES"-->
<!--#CONFIG TIMEFMT="%x"-->
<!--#ECHO VAR="cgi_variable_name"-->
<!--#FLASTMOD VIRTUAL="url/relative/to/current/url"-->
<!--#FSIZE VIRTUAL="/local/absolute/url"-->
<!--#INCLUDE VIRTUAL="~userid/user/rooted/url"-->
```

Figure 93. Sample Specifications of VM:Webgateway Server Side Include Statements

Only virtual references are supported due to the security issues of attempting to support absolute local file system and program references on these directives.

See the individual product documentation and 1.8.2, “Server-Side Include” on page 33 for more information.

7.8 Common Gateway Interfaces

The following sections do not cover in detail how to write a CGI, but they do mention some points that are specific to VM.

7.8.1 REXX and CMS Pipelines

VM allows rapid programming and the fast realization of ideas. Programming languages such as REXX and CMS pipelines are well-suited for rapid program development.

The combination of REXX and CMS Pipelines to create CGIs allows for quick development and low-cost maintenance. The performance is also much better than expected. The decision of which language to write CGIs on VM is easy. See Appendix C, “Sample Universal CGI for Use with All VM Web Servers” on page 211 for a CGI executable on several servers.

To get a feeling about the ease of programming CGIs in REXX and PIPELINE, look at Melinda Varian′s papers *An Introduction to Writing Webshare CGI Scripts* and *Plumbing The Internet*, available at http://pucc.princeton.edu/%7Epipeline/.

7.8.2 CGIs on VM Are Stateless

In VM, there is currently no capability to isolate multiple tasks with multiple CP and CMS authorities. Also, as multitasking is sometimes implemented by simply starting multiple Web server virtual machines, parameters or back-end connections kept from a previous client transaction may be needed by another virtual machine because the follow-on transaction is given to the next free server (for example, when handling a multi-part form).

As a result, you should try to design forms in a way to be executed atomic items; they should not be dependent on information from a previous transaction, nor should they need to pass along parameters or data to any future transaction.

This makes the design of a CGI a bit more complicated than writing an EXEC running in a virtual machine.

Note that there are ways to store information across multiple transactions (such as GLOBALV for a single server or through DASD files or IUCV connections between multiple servers), but these may still interfere with other clients′ requests and in general are too problematic to be practical.

See *Web-Enabling VM Resources*, SG24-5347 (which will be available at a later date) for more information.

7.8.3 Write Portable CGIs

If you start out with Webshare as your Web server, try to write portable CGI programs that enable easy migration. You may want to use the common CGI documented in Appendix C, “Sample Universal CGI for Use with All VM Web Servers” on page 211. Alternately, you may wish to simply take advantage of the capability to run Webshare compatibility mode CGIs in the commercial Web servers (VM:Webgateway and EnterpriseWeb/VM).

Chapter 8. How to Set Up a Web Site

The initial setup of a VM/ESA Web server should not take you more than an afternoon, assuming that your process allows you to obtain user ID definitions, privileges, and DASD space easily.

Following is a generic guide to performing an initial installation. It does not attempt to repeat the server-specific installation aspects contained in other sections of this book, or to replace vendor documentation concerning this process.

Although it is very easy to install a VM/ESA Web server, some considerations should be applied before providing a Web service. The real cost comes later, when users request changes to the navigation within the Web site or when application providers want to add new tools. To achieve *low cost* Web site administration, consider the following sections.

8.1 Using the Shared File System

The use of VM/ESA Shared File System is strongly recommended to achieve low-cost Web site administration. SFS provides easy cross-system access to data and centralized data management, which are essential to establishing practical and cost-effective Web service. So, if you do not have SFS installed, or only have the default file pools running, this is a good time to set up a file pool. Refer to Appendix B, “Quick Start to SFS” on page 197 for a brief overview on how to set up a starter SFS file pool.

8.1.1 Server Root Directory

With Webshare and EnterpriseWeb/VM, when placing Web server data into the Shared File System, be careful *not* to use a file space owned by the Web server itself. These Web servers may try to re-access an SFS directory that you want to be accessed permanently (for example, one that contains server side include data). The default for SFS access is to obtain a directory in WRITE mode if the file space name is identical to the user ID issuing the ACCESS command. In such a case, the directory already in access will be released and data on it may no longer be accessible to the Web server. VM:Webgateway does not have this restriction.

SFS allows only eight characters as the file space name. All other subdirectories can be named by using up to 16 characters. Longer subdirectory names are common in other operating systems and BFS, so you may want to consider placing applications directly under the root of the Web server or in BFS. When the root is being defined to match a file space, the SFS will allow depth of up to eight subdirectories.

Tips

- With Webshare and EnterpriseWeb/VM, do not use a file space name equal to the Web server user ID for any data to be served.

```
Web Server User Id:  WEBSRV01
SFS file space    :  WWWINTRA or WWWINTER
```

- Place the root in a subdirectory. (Do not place it in a file space.)

```
WWWINTRA.
WWWINTRA.CGIBIN
WWWINTRA.IMAGES
WWWINTRA.SSI
WWWINTRA.JAVA
```

- With VM:Webgateway, consider using BFS in cases where SFS does not meet your needs.

8.1.2 User Root Directory

A user root directory is identified in the URL addressed by a tilde (~) character. The VM Web servers permit you to create aliases for these user roots. All alias definitions still require the tilde character to identify a user root.

In VM, the default for file pool for a tilde hack is the same file pool as the Web server root directory resides in. To specify a different file pool name, use the file space name as specified behind the ~ character. In Webshare, the user file space must have either a subdirectory named WEBSHARE or a WEBSHARE FILELIST file in the main file space directory.

If no SFS file space meets this criteria, the default search switches to the user's 191 or 192 minidisk and looks for a WEBSHARE FILELIST there. For more details on the interpretation logic see 7.6, "User Root Resolution" on page 160.

User root directories are not recommended for applications because they may:

- Provide too much data to the Web if minidisks are used.
- Add to administration costs when the definition is made in FILELIST files.
- Appear separate from normal production applications. The tilde (~) character typically means user areas, not system areas.

On the other hand, it may be desirable to use a user root directory for a single application, as it will allow for some isolation of the data and programs of that application. A reasonable alternative is to use either virtual hosting or alternate ports on one host name to serve the application as a server root on the alternate host name and port pair.

8.1.3 Application Root Directories

Applications and their Web server related data should be placed one subdirectory deeper than the Web server's root directory. Doing so allows movement to other servers by taking a subdirectory tree and moving it. No data needs to be changed inside the application. Note that users pointing to a moved application need to update URLs stored as bookmarks.

The advantage of having application root directories below the Web server's root are:

1. Transparency, since URL and SFS directory structure maps one to one.

2. Delegation of administration tasks to the Web application owner. Note that the provided data belongs to a single owner, which is the file space user ID.
3. Pass-along application ownership using SFS′s built-in GRANT authorization.
4. Optional inclusion into INDEX HTML.
5. The tilde hack (~) is not required in the URL.

The disadvantage is that when using SFS, all data belonging to a file space must belong to the same storage group. For huge Web sites, this may become a management and a performance bottleneck.

A reasonable alternative is to use either virtual hosting or alternate ports on one host name to serve the application as a server root on the alternate host name and port pair.

8.2 Server Topology Design

When designing your Web site, you need to think about what kind of service it will provide, and what the maximum load may be. When you start providing data to the Web through a VM/ESA server, the tendency is to underestimate the popularity of the service. Because of this, try to design the topology in a way that allows for virtually unlimited growth in the future.

8.2.1 Multiple Web Servers

When you set up your first Web server, have multiple Web servers in mind from the start.

The Web server virtual machines should be able to share all their data. This includes the setup phase (PROFILE EXEC), definitions (single administration tasks), and the data served. SFS is ideal for this.

You may want to start your servers directly from the file pool by adding the correct CP directory statements. Figure 94 shows the FILEPOOL option on the IPL statement to set the Web server′s default file pool name.

```
00001 USER WEBSVI01 XXXXXXXX 64M 64M G
00002  INCLUDE DEFAULT
00003  IPL CMS PARM AUTOCR FILEPOOL WWWINTRA:
```

Figure 94. Sample Server Directory

If you are concerned about changing port numbers for a server, this is a simple change. Servers can share the same port number, or have different port numbers. The default port number is port 80.

8.2.2 Server Naming Convention

Even when you start with one Web server, it is a good idea to establish a user ID naming convention.

If you provide Internet and intranet service, a sample may be:

User ID	Usage
WEBSVI01	WEBSV for VM:Webgateway, I for Intranet or internal, and 01 as the placeholder for multiple Web server service machines belonging to a single service group.
WEBSRV01	WEBSRV for VM:Webgateway, nothing for Internet or external, and 01 as the placeholder for multiple Web server service machines belonging to a single service group.

8.3 Creating Directories

After you have designed your server topology, you should create the required SFS directories or minidisks. Since SFS is recommended, the following example shows only the definition of the SFS directories. Assume that you are either logged on in the WWWINTRA user ID, or that the user ID you use has SFS ADMIN authority. Figure 95 shows sample directories for a Web server. The data was entered in a file called ORDER DIRIDS A.

```
00001 WWWINTRA.CGIBIN
00002 WWWINTRA.IMAGES
00003 WWWINTRA.JAVA
00004 WWWINTRA.SSI
00005
00006 WWWINTRA.CUSTOMIZATION
00007 WWWINTRA.CUSTOMIZATION.SOURCE
00008 WWWINTRA.LOG
00009
00010 WWWINTRA.MYAPPLICATION
00011 WWWINTRA.MYAPPLICATION.CGIBIN
00012 WWWINTRA.MYAPPLICATION.IMAGES
00013 WWWINTRA.MYAPPLICATION.JAVA
```

Figure 95. Create ORDER DIRIDS A

Figure 100 on page 176 shows an EXEC that will read the ORDER DIRIDS file and create the required directories. The EXEC is named APPLENR.

8.4 Load Web Server Code into SFS

Now load the server code into the directories established.

For this step you must follow the installation instructions of the Web server you are installing.

Note: VM:Webgateway installs to minidisks initially. You can move it to SFS later and update the MDISK records in the SVM ID′s MDISK file to PDIRECT records that point to your SFS directories.

8.5 TCP/IP Set Up

Refer to Appendix A, “TCP/IP Configuration Notes” on page 187 for details about TCP/IP setup. The customization of TCP/IP can be performed before or after the Web server installation.

For now, just verify that TCP/IP port 88 is not in use at your site and use port 88 for the installation verification. Later you will need to change this port to port 80.

Tip

Use port 80 as the production port for your main non-SSL Web service. This is because some older browsers always assume port 80, and your users may wish to access your data with them.

Use port 443 for SSL services, as this is the default port for the https protocol.

8.6 Product Installation Verification

After you start the Web server, verify the installation by connecting to it using a Web browser of your choice.

You may want to use the Charlotte Web Browser on VM to do so if no workstation with a graphical Web browser exists.

Enter:

```
CHARLOTT http://myserver.company.com:88/
```

and your homepage (defined in INDEX HTML on your defined user root) will be displayed.

Chapter 9. How to Administer a Web Site

Since setting up a Web server on VM/ESA is simple, it can be easy to ignore the administration considerations involved. Your goal is to provide a Web site rich in function without administration overhead. A good design will make provision for these considerations:

- Future expandability
- 24x7 availability
- Change control
- Application changes
- Authorization management
- Security implications

9.1 Administrator′s Overview

As outlined before in the recommendations (refer to 7.2, “URL Hierarchical Addressing Scheme” on page 150), the VM/ESA Shared File System provides an easy-to-use, transparent structure for administering a Web site. If you want to delegate this administration inside your company, then you should enforce some basic rules.

The WEB server itself should place data into a file space whose name easily identifies its use. A sample group separation would be the use of the file space WWWINTRA for intranet users, and WWWINTER for Internet users. Both may use SFS file aliases to avoid data replication. Some recommended subdirectories under this root directory are:

CGIBIN To store CGIs
IMAGES To store pictures, audio, and video
JAVA To store JAVA objects
SSI To store server side includes

If you are not running VM:Webgateway, make sure that all four directories are permanently accessed in your PROFILE EXEC. Also, your configuration file should point to the chosen file modes. The file modes should not allow dynamically accessed user data to overlay the data contained in the server′s directories. Figure 96 on page 170 shows a sample startup EXEC for a Web server.

```
/* Sample Web server PROFSTRT EXEC, called by PROFILE EXEC        */

Parse Upper Arg intra . 1 'PORT' port .
If intra='' | intra='PORT' ,
  Then intra='WWWINTRA'          /* Assume INTRANET server        */
If DATATYPE(port,'W')=0 ,
  Then port='80'                 /* If not Webshare, pass port    */

filepool='MYPOOL'                /* Ensure filepool is set        */
'QUERY FILEPOOL PRIMARY ( LIFO'
If rc=0 Then Parse Pull . ipl_fp ':' .
        Else 'SET FILEPOOL 'filepool

'SET FILESPACE 'intra            /* Point to our Web file space   */

'ACCESS .         D'             /* HOME or ROOT directory        */
If rc<>0 Then Do
  Say 'Unable to access Web Server root directory. RC was "'rc'".'
  Exit rc
  End
'ACCESS .CGIBIN   E'             /* Server CGIs                   */
'ACCESS .IMAGES   F'             /* Server common images          */
'ACCESS .SSI      G'             /* Server Side Include           */
'ACCESS .JAVA     H'             /* Server JAVA objects           */

'ACCESS .CUSTOMIZATION. B'       /* Server local customization    */
'ACCESS .CUSTOMIZATION.SERVER_CODE C'  /* Server code unchanged   */

'EXEC TSPACE 5000 Blocks FM I'   /* Obtain temporary workspace fm=I*/
                                 /* for use by CGIs               */

/* For Webshare, port assignment should be done in CONFIG file    */
/* There is no dynamic port change without modifying the code     */
Queue 'EXEC HTTPD'               /* Start WEB Server on port xx   */
```

Figure 96. Sample PROFILE EXEC Fragment

9.1.1 Delegation of Authority

Because the Web server administrator owns the file space and user applications can be referenced without any additional definitions, you can enforce the same structure on the applications. Special attention must be given to any executable code that is run with the privileges of the Web server user ID.

All VM/ESA Web servers have methods to allow user CGIs to be executed with some controls.

For the Webshare-based server's CGIs, a CGIBIN FILELIST can be created to contain just the trusted CGIs. No other file will be executed as a CGI, but it will be served as plain text.

For user CGIs (that is, CGIs in user file spaces, accessed with a tilde), execution of CGIs can be protected easily by either of these methods:

- Trusting the data provider

  The user ID is added to the CGIUSERS definition.

- Disallowing CGI execution

  The user ID will not be added to the CGIUSERS definition.

If CGIs belong to applications placed in a subdirectory of the root directory, they can be executed without being explicitly defined for execution. This makes these applications the same as other trusted data providers.

A trusted user can further control execution of their CGIs by:

- Creating FILELIST files in a directory level above where the CGI is located
- Creating access limitations with HTACCESS or DIRMAP directives in the applying directory tree

For more information on how to protect CGIs, see 7.5.3, “Protect an Application” on page 158.

9.1.2 Administration of User Roots

Since the goal is to reduce administrative overhead to only the most necessary functions, placing applications below the Web server′s root is a good idea. Data placement and updates can be delegated to the data supplier. This reduces the introduction of a new user-controlled Web service to the following tasks:

- Creation of SFS directories with the CMS command CREATE DIRECTORY. Since an administrator needs to be contacted to execute this command, this request allows for some growth control and enforcement of service policies.
- Optional inclusion into the server′s INDEX file that is presented to all visitors, which just specifies the servers address. The additional service can be cataloged for everyone to see.

The same structure can be established on minidisks using FILELIST or DIRMAP files. This, generally, has two basic disadvantages:

- The files will require updating.

  FILELIST or DIRMAP files have to be set up to include the data to be served. It is difficult to verify that all the needed data is included, and no overlap exists. Administration and security responsibility stays with the administrator, while the data comes from the Web service provider.

- There is no R/W data sharing across multiple Web Servers.

  Since your site will grow and the number of visitors will grow as well, you may reach the limit of what a single Web server virtual machine can serve. TCP/IP is able to balance the connection requests across multiple Web server virtual machines.

  With minidisks, multiple write access is very dangerous. Therefore, minidisk-based services are restricted to read-only data, and updates require a temporary switch to write mode through CGIs. Other Web server virtual machines must be informed that data has changed.

  With the SFS, data can easily be shared in write mode. You may even share the data between multiple real systems, exploiting the remote capabilities of the SFS.

9.2 Reconfiguring the Web Server

As with other VM services, the need to provide service on a 24-hour a day, 7-day a week basis is a requirement. The VM Web servers address this availability issue as follows:

- Webshare is a standard VM virtual machine; any user ID can become a Web server.

  Webshare provides good stability, and since no new functions were developed for this shareware, maintenance should be zero.

- EnterpriseWeb only requires a server restart when the configuration file requires an update.

  Using the supplied WEB browser interface, automatic recycling is provided.

  You can write an EXEC to recycle EnterpriseWeb if necessary. With an external scheduler such as Host Management Facilities/VM (HMF) or a WAKEUP-based service machine, the recycle can be scheduled during any maintenance window.

- VM:Webgateway includes server reconfiguration with commands. Updates are possible through the WEB browser administration interface.

Because the HTTP protocol ends after something is served, any fast server restart needed should not affect current users of the VM/ESA Web server. In the future, when VM applications receive a front end through a WEB browser (requiring the WEB server to keep sessions to a back-end service like DB2/VM access), all reconfiguration will have to preserve these sessions, as is done in VM:Webgateway. A simple virtual machine restart will no longer be a first-choice option.

9.3 Change and Problem Management

To achieve minimal administration, only the Web server itself and the infrastructure provided through it, should be controlled by a change management process. Examples would be managing software changes in the Web server itself, and managing the effects of these changes.

Adding a new service under user control is done dynamically. The content is managed by the users, and they control the process. The Web server administrator can monitor any changes easily through simple CMS commands.

9.3.1 Changes That Require Attention

Changes in the server that require attention include those that either jeopardize the service commitments, or change external end-user references.

Applying PTFs is something which should be planned for. If corrections to your errors exist, a problem record should be kept to obtain fixes for the change.

For major changes, follow these steps:

1. Create a backup.
2. Create a copy of the server data to be changed. If the server data is on the SFS, create another directory and alias all files first to the new target.
3. Make your changes.

4. Start a spare Web server virtual machine and connect to any TCP/IP port not in use (for example, port 88).
5. Verify that the change works.
6. Plan the change to the production Web servers according to your local change management rules.

9.3.2 Moving Applications to Another Server

Moving applications to a different URL address might require a change to external references. You should therefore avoid this as much as possible. Complete URLs, pointing to the applications to be moved, may exist somewhere in your users' bookmarks. If the need arises to move an application, add redirection to the VM/ESA Web server through an alias in the main FILELIST or DIRMAP.

As this only works for the entry point of an application, you may want to add an HTML page instead in the old directories, to accomplish two things:

- Explain to the user that this application is moved to a new place. (The page may also contain the reasons for doing so.)
- Add a fully-qualified URL address of the new location to ease the task of updating possible bookmark entries in the client's browser.

Figure 97 shows an HTML file that points a user to a new location.

```
00001 <html>
00002 <head>
00003 <title>Application has been moved</title>
00004 </head>
00005 <body>
00006 <p> This service have been moved to a new location.
00007 <p> Please use the URL below to go there and to ease updating
00008 your stored references.
00009 <p><p>
00010 The reason to move was an overwhelming number of visitors which
00011 required a server of our own.
00012 <p>
00013 <a href="http://new.server.loc/application/index.html" New home is now
00014 "http://new.server.loc/application/index.html"</a>
00015 </body>
00016 </html>
```

Figure 97. Sample Rerouting HTML File

9.4 How to Manage an Application for the WEB

As an application owner, or as a data provider for a Web server, you might want to consider automating tasks such as:

- Backup
- Selective recovery
- Grouping of files
- VM FORUM files for news readers
- Mirroring to other servers
- Peer-to-peer communication

All of this can be handled by TOOLSRUN. TOOLSRUN is easy to control, easy to administer, and provides all the functions listed above.

Your existing TOOLSRUN service can be front-ended by the Web server. You will extend your current information to a new user base.

You may want to have a look at the TOOLSRUN package, which is heavily used inside IBM for VM-based information exchange FORUMs. You find it in the VM DOWNLOAD LIBRARY as PACKAGE TOOLSRUN, and you can access it from this address: `http://www.vm.ibm.com/download/packages`.

9.5 Web Server Administration through a Web Browser

A Web browser is the ideal interface for administration of a Web server. Even though you may prefer to administer your Web server with native VM methods, using a Web browser will allow you access to your server functions when there is no 3270 connection available.

If you have installed VM:Webgateway, refer to 6.6, “VM:Webgateway Configuration” on page 122.

If you have installed EnterpriseWeb/VM, refer to 5.8, “Configuration” on page 94.

If you have installed Webshare, this function is not currently implemented.

9.6 Web Server Administration from VM

The following sections describe how to simplify server administration using VM/ESA.

9.6.1 Minidisks

If you decided to implement your server using minidisks, keep in mind the following points:

- If maintenance is required on minidisks that are accessed as write in the Web server virtual machine, you must log on to the server machine, which may potentially disrupt the service.
- If maintenance is required on minidisks that do not require write access in the Web server virtual machine, you can perform the following steps:

```
LINK server vaddr 999 MR
ACC 999 Z
     (perform the required maintenance)
     (make sure you first rename the file you want to change)
RELEASE Z (DET

CP SET SECUSER webserver *
CP SEND webserver ACCESS vaddr vfm
CP SET SECUSER webserver RESET
```

9.6.2 Shared File System

If you decided to implement your Web server using the shared file system, you have more flexibility regarding from where you can perform your maintenance. It is possible to do it from another system, provided your shared file system is a shared global resource.

If the directory must be in write access by the Web server virtual machine, ensure that it is a FILECONTROL directory. The SFS allows for shared read/write access. This eliminates the need to re-access a disk from the server side. CMS does all the required SFS locking for you, to prevent two updates from happening at the same time.

To update files on the SFS, you can do the following:

```
ACCESS fp:WWWINTRA.dirid Z (FORCERW
      (perform the required maintenance
       for backout reasons you may want to COPY the files you intend
       to change first to an inactive name)
RELEASE Z
```

9.6.3 Reload after Configuration Changes

This step is completely automated in VM:Webgateway. For a Webshare-based server, there are two ways to activate changes to the Web server configuration:

1. Recycle the server virtual machine.

   This is the easiest way and is the only way supported by Webshare without writing your own RELOAD CGI.

   You should recycle all active WEB servers connected to a single port number. The TCP/IP command, NETSTAT ALLCONN, is useful if available. A sample that could work in many installations (shown configured for port 80) is shown in Figure 98.

```
'PIPE command NETSTAT ALLCONN',      /* Get all userids connected */
  '| locate / *..80 /',              /* to port 80                */
  '| spec /NETSTAT DROP /1 w2 1',    /* Recycle via TCP/IP help   */
  '| command'
```

Figure 98. Sample RECYCLE EXEC Fragment

2. Recycle through the Web interface.

   If the Web server requires restart after configuration variables are changed, the server may decide to recycle itself.

   Recycle is a server-initiated restart of itself and other Web servers connected to the same port or configuration data. This is the case, for example, with the EnterpriseWeb/VM Interface.

   If the Web server decides to update changed directives without the need for a restart, the commercial products try to do so. For example, VM:Webgateway tries to reload changed configuration data as much as possible without a Web server restart.

When using SFS to maintain multiple Web servers, you should define a pre-scheduled maintenance program that runs in the Web servers' virtual machine and determines and starts changes.

9.7 Setup for a New Web Application

After a Web application provider has chosen an application directory layout, the Web administrator will create the required SFS directories or minidisk space. At this time the CGIs that need to be active are identified.

If the data resides in the SFS, and the CGIs are trusted, a file called ORDER DIRIDS A, as shown in Figure 99, could be created and used.

```
00001 WWWINTRA.ORDER
00002 WWWINTRA.ORDER.CGIBIN
00003 WWWINTRA.ORDER.IMAGES
00004 WWWINTRA.ORDER.JAVA
00005 WWWINTRA.ORDER.PIZZA
00006 WWWINTRA.ORDER.PIZZA.AMERICAN
00007 WWWINTRA.ORDER.PIZZA.ITALIAN
00008 WWWINTRA.ORDER.PIZZA.TOPPINGS
00009 WWWINTRA.ORDER.PASTA
00010 WWWINTRA.ORDER.PASTA.TOPPINGS
00011 WWWINTRA.ORDER.PASTA.TOPPINGS
```

Figure 99. Create ORDER DIRIDS A

Assume that you are either logged on to the WWWINTRA user ID, or that the user ID you use has SFS ADMIN authority. Then execute an EXEC similar that shown in Figure 100 to create the directories.

```
/* Application enrollment sample */
Address 'COMMAND'
Arg tousers '(' opts

Auth='WRITE NEWWRITE'
If WORDPOS('READ'  ,opts)>0 Then auth='READ  NEWREAD'
If WORDPOS('WRITE' ,opts)>0 Then auth='WRITE NEWWRITE'
If WORDPOS('PUBLIC',opts)>0 Then auth='READ  NEWREAD'

'PIPE (end ?) < ORDER DIRIDS',          /* Read directory file         */
  '| strip both',
  '| nfind *'||',                       /* Remove comments             */
  '| f1: fanout',
  '| spec /CREATE DIRECTORY /1 w1 nw', /* Create directory first      */
  '| f2: faninany',
  '| cons',                            /* Show command to be executed */
  '| command',                         /* Execute command             */
  '| cons',                            /* Show results if any         */
  '? f1:',                            /* Build stage for each user   */
  '| specs /callpipe (stagesep %) var tousers',
  '% split',
  '% specs ;GRANT AUTH/ 1 w1 nw /TO; 1 w1 nw ;('auth'; nw write',
     ';GRANT AUTH * */ nw w1 nw /TO; 1 w1 nw ;('word(auth,1)'; nw',
  '% *:/ nw',
  '| pipcmd',                          /* Execute stage to build GRANT*/
  '| f2:'
Exit rc
```

Figure 100. APPLENR EXEC Sample

9.7.1 Verify and Activate CGI Allowance

If you trust your Web application providers, there are no other tasks you have to perform at this point. If you need to enforce a kind of CGI enrollment process, refer to 9.1.1, “Delegation of Authority” on page 170.

9.7.2 Provide Navigation through INDEX HTML

Entry navigation is reduced to maintaining an entry per application in the Web servers root directory INDEX HTML file. This file will list services available on a server for users who access the default server page.

9.8 Internet or Intranet Setup Implications

Most likely you will use a VM/ESA Web server to serve intranet needs. This does not require many security considerations, as the data to be provided will be mainly for company-wide internal needs.

After gaining experience, you may decide to provide data to Internet users as well. Security issues are then of greater concern, as you do not want internal information exposed to just anyone.

The IBM Endicott site, home of the VM/ESA home page, has provided external access to their Web server. The layout of the VM/ESA Endicott servers is shown in Figure 101.

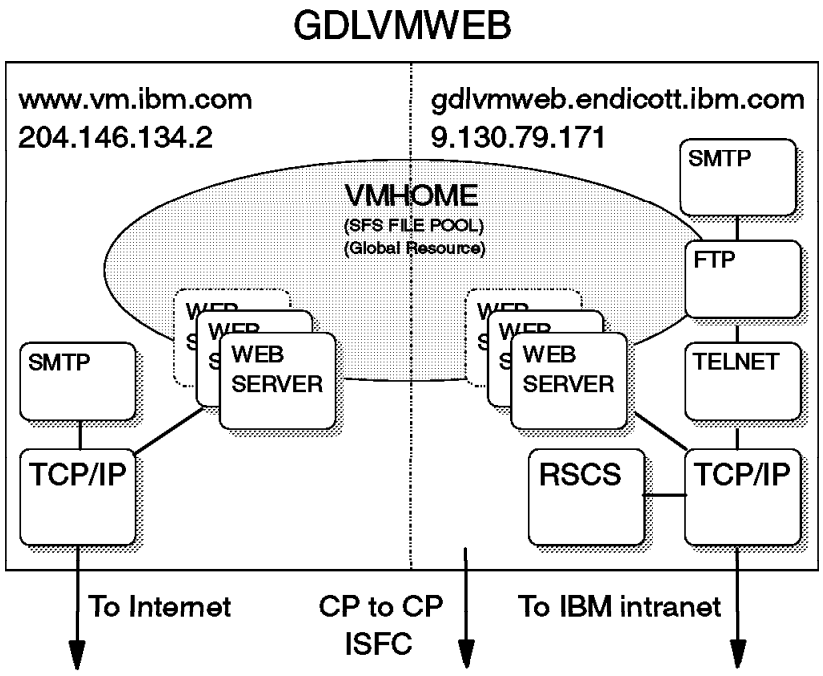


Figure 101. Endicott Site Setup

9.8.1 The Intranet

The IBM Endicott intranet site has its own TCP/IP service machine, which is accessible only to internal users by either going through a company firewall node if coming from outside, or by belonging to the internal TCP/IP network, clearly identified.

A name server would relate the IP address 9.130.8.40 to the logical server name `gdlvmweb.endicott.ibm.com`. The intranet TCP/IP also supports services such as TELNET, SMTP, FTP, and RSCS. This causes no problems, as the data available through the web can be accessed by any employee. TCP/IP services such as TELNET or FTP have their own authentication.

The data served by the server group is stored in the shared file system. All internal available data is granted read access to the Web servers for the intranet.

9.8.2 The Internet

The Endicott Internet site has its own TCP/IP service machine, which is accessible by external users. Internal users can connect there as well, but only by going outside the firewall.

Using two servers for intranet and Internet services has several advantages, namely:

1. Using the same SFS allows for direct data sharing between the two Web server groups.
2. Maintenance for the various Web servers, as well as all server code, can be done centrally.
3. Access control exploits the basic SFS GRANT capability. This can easily be extended to any External Security Manager (ESM) such as RACF, ACF2, or VM:Secure if needed.
4. Since the Web servers run in XC virtual machine mode, performance can be optimized by exploiting data spaces for DIRCONTROL directories, and by data caching in control units, and expanded and main storage.
5. The administration effort is nearly zero, as it can be delegated to the providers of Web data and applications, providing they place their data into the common Shared File System pool VMHOME.
6. To avoid data replication on work in progress, the SFS ALIAS feature can be exploited to serve only completed work. To do so, the Web servers get a global read permit through the CMS GRANT command (or ESM equivalent) for a subdirectory from which to serve. The owner of the data creates aliases using the CMS ALIAS command. Changes to the base file, even in a different directory, are automatically reflected to the Web server with no additional effort.

9.8.3 TCP/IP Concerns

You may ask why this solution cannot be achieved through a single TCP/IP. The answer lies in the way TCP/IP currently works and what the Web servers “know.”

For example, assume you want to connect Internet visitors at IP address 204.146.134.2, and intranet visitors at address 9.130.8.40.

Since TCP/IP can optionally easily forward requests, communications could potentially intermix. You would need an external router or a firewall function if a single TCP/IP is a requirement to prevent unauthorized access.

The Web server must be able to act as a multinode server. This means that the server has to query from which local adapter address it was called, and then adjust the security accordingly.

While VM:Webgateway has the needed support with full support for virtual homing and a robust security model, not all of the Web servers reviewed have these capabilities. These capabilities are necessary to support a multinode operation using a single VM TCP/IP server. In addition, corporate policy often requires that a multi-LAN operation as we have described requires the usage of segregated TCP/IP servers. As a result, the easiest and cleanest way to implement both intranet and Internet servers on the same node is to separate the traffic, as was done at the Endicott site (by using two TCP/IP machines). If, for economic or other reasons, you wish to provide services such as we have described here with a single TCP/IP server and network interface, then you should invest in a Web server that has full virtual homing support and a robust security model.

9.9 Consolidating Web Services

At some point, you may need to combine multiple Web services onto a single VM system.

Assume that all applications were written to the same standards. In this way, the VM/ESA Web servers have the same setup. Aside from naming conflicts, the data of the Web servers can be combined onto single directories. Assume further that your administration policies ensure that no namespace overlaps exist.

Also assume the Web services are written with relative URLs only. Using them, there should be no problem putting the directories under the same server root directory. If the same naming was used for different applications, simply rename the major user root subdirectory. Update the servers INDEX HTML, and the server work is complete. The CMS RELOCATE command may help move a directory structure for you.

The real problem in consolidating Web services are the bookmark entries which exist on the various client workstations. The users must be guided through the change so that they can adjust their stored bookmarks. Furthermore, this process needs to be extended over a long period of time to warn infrequent browsers.

Three basic implementation scenarios can be envisioned:

- Two TCP/IP service machines run on the target VM system, with the Web servers running totally independently.

  You might have to rename the Web servers on one of the merged services if it is already in use on the target system. This does no harm to stored bookmarks, as the IP address of the server and the underlying structure remains the same.

- One TCP/IP service machine running on the target VM system, with two Web servers running independently of each other.

For this, the Web servers would have to connect to an unshared IP address to identify where the request wanted to go. This is not possible with VM Web servers. As a result, such a merge should not be done when the home IP address is used to distinguish which data is served.

When eliminating an IP address, the name server must be updated to provide requests for the eliminated IP address with the new target address.

- One TCP/IP service machine running on the target VM system, with one Web server running.

  For this, the services of both Web servers must have had the same structure and all conflicts must have been resolved. Also, this merge should not be done when the home IP address is used to distinguish which data is to be served.

  As a result, you may deactivate an IP address if performance allows (performing the same name server change required as above), or keep the previous two.

The last two scenarios require guidance to the browsers to update their bookmarks. You may choose redirection as described in 9.3.2, “Moving Applications to Another Server” on page 173.

Tip

It is advisable to run multiple TCP/IP service machines and multiple sets of VM/ESA Web servers after system merges, or to take advantage of TCP/IP′s support for virtual hosting. This avoids extra work for the clients and is easy to achieve and manage.

If the service machines must be merged, then add redirection for the browsers, so that the relocated application is “just a click away.”

Avoid merging services where the Web server takes different paths based on the home IP address through which it was contacted.

9.10 Logging

For reference, statistical, and debugging functions you might want to keep a complete transaction log. All VM Web servers allow for this.

The base logging function for a VM Web server is the virtual machine console. It should be collected in a LOG service machine. You might want to change the LOGGING directive in the Webshare configuration to what is shown in Figure 102.

```
LOGPIPE=CONSOLE | SPEC /MSG WWWILOG/1 1-* NW | CHOP 240 | CP
```

Figure 102. Base Logging Function

You may want to deactivate the IDENT function in Webshare to speed up the Web servers′ processing. In the log service virtual machine, you will likely perform functions such as:

- Name resolution (simulate IDENT=YES)

- Statistic and report generation
- Log archiving

At this time, only EnterpriseWeb/VM provides a log handling service machine. This must be considered part of the framework which requires some modification to fulfill all the needs of a production service.

Take a look at the samples written for the Endicott site. You find them in the VM DOWNLOAD LIBRARY as PACKAGE VMHP at the following address:
`http://www.vm.ibm.com/download/packages`

9.11 Tuning Your Web Site

Even though this may not be needed until very high hit levels are reached, you have the following tuning possibilities:

- Data spaces if data is in DIRCONTROL

  If you require quick access to data that infrequently changes, you should consider using data spaces and defining your SFS subdirectories as DIRCONTROL. The default when defining an SFS directory is FILECONTROL.

  You have to weigh the ease of administration and possible aliases of a FILECONTROL directory against the performance benefits of a DIRCONTROL directory. As VM/ESA also provides automatic caching in storage, you should be able to serve many concurrent Web requests for data in FILECONTROL directories nearly as fast as with DIRCONTROL directories. The FILECONTROL directories will, however, be slower to access by Web servers that use the CMS ACCESS command (such as Webshare, EnterpriseWeb/VM and VM:Webgateway in Webshare-compatibility mode) when serving a URL.

- Minidisk caching in expanded and main storage

  Use data in storage techniques wherever possible.

- CGIs always in server disk access

  For large disks with many files, keeping a disk permanently accessed by the server will enhance performance of Web servers that use the CMS ACCESS command (such as Webshare, EnterpriseWeb/VM and VM:Webgateway in Webshare-compatibility mode).

- CGIs written in efficient CMS Pipelines

  Make use of CMS Pipelines to process large amounts of data quickly and with little programming time.

- Multiple Webshare-based Web Servers connected to a single TCP/IP port and multiple workers on VM:Webgateway

  For VM:Webgateway, start with multiple worker machines. For Webshare and EnterpriseWeb/VM, start with multiple servers. If they are not used, optionally reduce them to cover your maximum hit rates.

- Moving applications to another server or another TCP/IP stack

  One of the present limitations on a Web server is throughput of TCP/IP. Creating another TCPIP virtual machine will help eliminate this problem, though may introduce administration overhead.

- Standard CP performance settings such as QUICKDSP or SET SHARE

9.12 User Root Maintenance

A user root is identified by a tilde (~) character in a URL address. This is sometimes called a user hack.

Creating user roots is possible with all Web servers. You might want to consider disallowing user roots on your system to provide greater control over changes in service.

If you decide to allow user roots, the various Web servers handle the roots differently. The following list discusses these differences.

Webshare

A Webshare user root is limited to the same file pool as the server root directory, or to users' 191 minidisk. The search logic is as follows:

1. Is ACCESS to "fp:user.WEBSHARE" possible?
   If Yes, take it as user root.
2. Is ACCESS to "fp:user." possible?
   If Yes, search for file "WEBSHARE FILELIST."
   If Yes, take it as user root.
3. Is ACCESS to "user 191" possible?
   If Yes, search for file "WEBSHARE FILELIST."
   If Yes, take it as user root.
4. Is ACCESS to "user 192" possible?
   If Yes, search for file "WEBSHARE FILELIST."
   If Yes, take it as user root.
5. If all checks result in NO, return an error message.

To enable for user roots, the directive USERWEBS must be set to ON.

EnterpriseWeb/VM

An EnterpriseWeb user root can reside in any file pool or user minidisk. The logic is the same as with Webshare, except that a search is done for an EWEB subdirectory in addition to a WEBSHARE subdirectory, and in addition to a WEBSHARE FILELIST, an EWEB FILELIST is looked up as well.

To enable for user roots, the directive USERWEBS must be set to ON.

The default minidisks to be searched are specified with the directive USERWEBDISKS.

VM:Webgateway

VM:Webgateway uses a command interface to define user roots. The user root can be any SFS or BFS directory level (this includes a subdirectory to be set as a user root), or user minidisks.

Since this information is stored in the VM:Webgateway's database, which resides on a minidisk, all commands have to be replicated to all Web servers if multiple Web servers are used.

Note: As long as the Web server permits using user roots, and a user permits the Web server virtual machine to read his data, the user roots can be served. Usually if no FILELIST file is found, an automatic index list is generated with all files on the target object identified as user roots.

Chapter 10. How to Set Up a Web Application

Providing data and applications through the Web is an easy task, and this topic is best left to the many publications on this subject.

The following sections discuss the VM specifics for setting up an application. These depend heavily on the topology of the Web server chosen by the administrator.

Aspects of this topic are also covered extensively in

- *Web-Enabling VM Resources*, SG24-5347 (which will be available at a later date).

10.1 Select an Application Topology

When setting up a Web application, the first thing to do is to clearly define how you want to provide your data. You are free to choose how to present the data remembering that it should address the intended audience. So the basic question you have is how to handle the entry navigation of the visitor. The decision points are:

1. What should be put into the INDEX HTML file by the administrator as the anchor point to your application?

   Supply an easy-to-remember, but still clearly matching, application title. Accompany it with a nice graphic icon, if you wish.

2. Should access be through a user root, or should it stay below the server root?

   The user root requires a tilde character (~) in the URL path definition. For an application, this may not seem appropriate.

   Staying below the server root may require more coordination with your Web administrator, but it does ease the integration of your Web application into management services such as backup and recovery or security processes.

3. What is the application hierarchical topology?

   To make it easy for the visitor to navigate through your application, you should define the URL directory levels with significant naming. It is wise to stick with CMS limits as outlined in 7.3, “Naming Conventions” on page 151. This will allow you to stay away from navigation maintenance using techniques such as the FILELIST files.

After this topology design, create a file containing the SFS subdirectories you need and make it available to your Web administrator. The administrator will ask you about the anchor content in the main INDEX HTML, as well as which user IDs are allowed to have write access to your application data.

10.2 Select the Navigation

This section provides details on specific navigation issues.

For entry navigation, you might want to create an INDEX HTML file in your application root. If this is found, it will be presented to the visitor as your *homepage*. Make the homepage appealing, but try to avoid using fancy graphics, because that may delay processing and slow down the visitor′s progress.

You may want to place your CGIs into the server′s CGIBIN directory. This may even be a requirement if the administrator has to enforce a CGI approval process to ensure stability of the Web service. If you are not familiar with VM′s way to code CGIs, you may want to ask the administrator to proofread your action routines.

If the administrator is allowed to apply the “principle of trust,” which gives you the freedom to place CGIs wherever you want, consider placing it in your application CGIBIN directory. This makes it easy for you to reuse existing CGIs and is also a single way to look up CGIs. Your HTML files will always refer to the same place for CGIs (using indirect URL addressing). This lets you create an HTML skeleton for your application that has reuse capability.

Another consideration is the use of Server Side Includes. As the name suggests, this is a service the server provides to you by including pieces of common interest into your HTML pages. Refer to 7.7, “Server Side Includes” on page 160 for more details.

10.3 Select the Access Security

In the VM Web servers, access security is performed through HTACCESS directives (EnterpriseWeb) or DIRMAP rules (VM:Webgateway). As a high-level example, you may want to look at 7.5.3, “Protect an Application” on page 158. For detailed information, refer to the documentation of the Web server product installed.

If visitor authentication is required, contact your Web administrator about the most usable method (such as protection through the VM directory, External Security Managers, or simple user or group lists).

Also, stick with other REALM names already in use at this Web site to avoid asking visitors too often for their user ID/password combinations.

10.4 Exploit Shared File System Capabilities

We have already detailed the positive aspects of using the Shared File System as the base for Web serving. Try to exploit its features as much as you can. In this way, you can reduce the need for additional navigation files to a minimum.

As a short checklist, the main points to remember when placing data into an SFS to be served with the Web are the following:

- Subdirectory naming should match the URL address hierarchy.
- Subdirectory naming is limited to sixteen characters.
- CMS file name and file type naming are limited to eight characters each.

- Inside an SFS file pool you can use CMS ALIAS definitions to make files visible to a Web server, instead of replicating data out of work directories (such as your own).

10.5 Manage Application Data Changes

As an application owner, you should consider how to provide consistent service to application users. In case of problems, you must guarantee such items as:

- Backup and recovery
- 24-hour-a-day, 7-day-a-week production service
- Change control management

Even if you are working with a pilot service of test servers, these items should be considered. Actions should be clearly described (for example, for a backup person or operations).

One way to achieve change management is to use the TOOLSRUN PACKAGE, which has many necessary functions and controls. Find it in the VM download library as PACKAGE TOOLSRUN at the following address:

`http://www.vm.ibm.com/download/packages`

Finally, whenever you change a CGI, try it out on a spare Web server first. This is also a good place to do debugging with the powerful REXX interactive TRACE facility. Ask your Web administrator for this capability.

10.6 Common Appearance of All Applications

Your application should have a style, which should be preserved across all the views you provide.

Remember that there are things like “Server Side Includes,” which help you to change repeated information only once.

At least for externals, you should preserve a common corporate appearance. When designing your application′s appearance, keep existing rules and guidelines in mind.

10.7 Tuning Your Web Application

You should provide your Web service without wasting the resources of the operating system. Some reminders from previous sections are listed here:

- Get started with CMS PIPELINE/REXX power development.

  If you are unfamiliar with the CMS programming tools REXX and CMS Pipelines, try them out before you start programming in other languages. This will take little time, and the results are impressive.

  See 7.8.1, “REXX and CMS Pipelines” on page 161 for more information.

- Use fast loading graphical images.

  You should use low depth (that is, one with a URL as close to the root as possible), fast loading (that is, small, in both record and byte counts) graphics. Rendering will be much faster, but the user may come in through a slow line, and graphic loading may keep him waiting too long.

- Remember that there are browsers that do not display graphics.

  Always add good explanatory alternate text to identify to non-graphic browsers what kind of graphic they are missing. Never assume that navigation through graphics (such as imagemap) is possible for all clients; add alternative text navigation as well.

  An example of a fast non-graphical browser is Carl Forde′s Charlotte, running on VM, available from the Beyond Software Page at the following address:

  `http://www.beyond-software.com/software/charlott.vmarc`

- Do not overload pages with pictures and audio and video.

  Pictures, audio, video, background music, animated icons, JAVA, and more are supported. Remember two things before overloading a page.

  1. The user may sit on a slow line, so transferring all these objects may take a long time.
  2. A “busy” page design may make the visitor focus on the features you provide rather than on the information you want to pass on to him. (However, this should not stop you from creating a fun page for a serious application.)

- Be aware of “server push” and “client pull” functions.

  These allow a Web server (server push) or a browser (client pull) to continuously send or request new information. In some cases, this is exactly the function you want to provide (such as a stock price application displaying at a TV monitor with public access or an alert monitor). However, in other cases it is just network overhead and adds little extra value.

10.8 Communicate Changes

Inform your Web administrator if your application is moved or eliminated. Anchors pointing to “nowhere” does not build confidence in your application. In case of application elimination, keep a “This application has been removed” HTML in the server′s root to inform possible visitors.

Appendix A. TCP/IP Configuration Notes

The following chapter gives some helpful tips to help you understand why it is necessary to customize TCP/IP properly.

A.1 Port Assignment

Any TCP/IP application that needs to receive incoming requests must connect to a TCP/IP port.

A.1.1 Some Thoughts about Port Reservation and Security

If you start a Web server *without* reserving a port for its user ID in the TCP/IP configuration, you might experience problems such as:

- *Any* user ID on the system may connect to *any* port that is not reserved.
  - The port you want to use might get blocked by another user ID.
  - *Any* user ID might identify itself as the Web server and capture the client's password if user ID and password protection is active.
- If the Web server is restarted, the port might get blocked for several minutes, until all outstanding or pending sessions are dropped; therefore, the Web server cannot connect to the port.

These problems will be avoided if the user ID of the Web server is bound to a specific port in the TCP/IP configuration.

Note: TCPIP starts all virtual machines on its current AUTOLOG list when it starts execution. If a virtual machine on the AUTOLOG list has reserved a TCP port with the PORT statement, but is not accepting connections on that port, TCP/IP attempts to cancel that virtual machine and start it again. An exception is the case where the PORT statement specifies NOAUTOLOG.

TIP

If there are problems, achieve automated Web server recycle by defining it in the TCP/IP configuration (PROFILE TCPIP).

A.1.2 Choosing the Correct Port Number

For a regular Web server, port 80 should be used. It is the default TCP/IP port for HTTP servers. For the HTTPS protocol, the default is port 443.

If you want to start the Web server in test or maintenance mode, use a port number other than 80 or 443. Otherwise, any user can access incomplete data, and programs that you do not want to serve.

After you have decided which port number to use, define this port in the PROFILE TCPIP file (or *system_name* TCPIP respectively, if the same disk is shared among multiple systems), which is usually located on TCPIP 191.

```
;
; Autolog the following server machines
  AUTOLOG
;   FTPSERVE password    ; FTP     SERVER
;   SMTP     password    ; SMTP    SERVER
    SNALNKA  password    ; SNA     LINK SERVER
;   SNMPD    password    ; SNMP    VM AGENT  VIRTUAL MACHINE
;   SNMPQE   password    ; SNMP    VM CLIENT VIRTUAL MACHINE
;   REXECD   password    ; REXEC   SERVER
;   PORTMAP  password    ; PORTMAP SERVER
;   VMNFS    password    ; NFS     SERVER
    LPSERVE  password    ; LP      SERVER
;   NAMESRV  password    ; DOMAIN  NAME SERVER
;   NCSGLBD  password    ; NCS     GLBD SERVER
;   NCSLLBD  password    ; NCS     LLBD SERVER
;   ROUTED   password    ; ROUTED  SERVER
    WEBSHARE password    ; WEBSHARE                    1
  ENDAUTOLOG
;
; Reserve the following ports for specific servers
; values from RFC 1060, "Assigned numbers"
  PORT
    20 TCP FTPSERVE  NOAUTOLOG ; FTP Server
    21 TCP FTPSERVE            ; FTP Server
    23 TCP INTCLIEN            ; TELNET Server
    25 TCP SMTP                ; SMTP Server
    53 TCP NAMESRV             ; DOMAIN NAME Server
    53 UDP NAMESRV             ; DOMAIN NAME Server
    80 TCP WEBSHARE            ; WEBSHARE                2
   111 TCP PORTMAP             ; PORTMAP Server
   111 UDP PORTMAP             ; PORTMAP Server
   135 UDP NCSLLBD             ; NCS LLBD SERVER
   161 UDP SNMPD               ; SNMP AGENT
   162 UDP SNMPQE              ; SNMPQE AGENT
   512 TCP REXECD              ; REXECD SERVER (REXEC)
   514 TCP REXECD              ; REXECD SERVER (RSH)
   515 TCP LPSERVE             ; LP SERVER
   520 UDP ROUTED              ; ROUTED SERVER
   750 TCP VMKERB              ; KERBEROS SERVER
   750 UDP VMKERB              ; KERBEROS SERVER
   751 TCP ADMSERV             ; KERBEROS DATABASE SERVER
   751 UDP ADMSERV             ; KERBEROS DATABASE SERVER
  2049 UDP VMNFS               ; NFS Server
```

Figure 103. Part of a Sample PROFILE TCPIP for a Single Web Server

Notes:

1. User ID WEBSHARE will be started when TCP/IP starts execution.
2. In this example, user ID WEBSHARE is bound to port 80.

A.1.3 Multiple Servers on the Same Port Number

For performance reasons, you may want to use multiple Web servers on the same port number. In this case, you have to specify an additional entry for each of the Web server user IDs in the PROFILE TCPIP file. All of these Web server user IDs will be able to connect to the same port at the same time. The first Web server that reacts to an incoming request will take it, and the other Web servers will ignore that request.

```
;
; Autolog the following server machines
  AUTOLOG
;   FTPSERVE password    ; FTP     SERVER
;   SMTP     password    ; SMTP    SERVER
    SNALNKA  password    ; SNA     LINK SERVER
;   SNMPD    password    ; SNMP    VM AGENT  VIRTUAL MACHINE
;   SNMPQE   password    ; SNMP    VM CLIENT VIRTUAL MACHINE
;   REXECD   password    ; REXEC   SERVER
;   PORTMAP  password    ; PORTMAP SERVER
;   VMNFS    password    ; NFS     SERVER
    LPSERVE  password    ; LP      SERVER
;   NAMESRV  password    ; DOMAIN  NAME SERVER
;   NCSGLBD  password    ; NCS     GLBD SERVER
;   NCSLLBD  password    ; NCS     LLBD SERVER
;   ROUTED   password    ; ROUTED  SERVER
    WEBSHAR1 password    ; WEBSHARE                    1
    WEBSHAR2 password    ; WEBSHARE                    1
    WEBSHAR3 password    ; WEBSHARE                    1
  ENDAUTOLOG
;
; Reserve the following ports for specific servers
; values from RFC 1060, "Assigned numbers"
  PORT
    20 TCP FTPSERVE  NOAUTOLOG ; FTP Server
    21 TCP FTPSERVE            ; FTP Server
    23 TCP INTCLIEN            ; TELNET Server
    25 TCP SMTP                ; SMTP Server
    53 TCP NAMESRV             ; DOMAIN NAME Server
    53 UDP NAMESRV             ; DOMAIN NAME Server
    80 TCP WEBSHAR1            ; WEBSHARE                2
    80 TCP WEBSHAR2            ; WEBSHARE                2
    80 TCP WEBSHAR3            ; WEBSHARE                2
   111 TCP PORTMAP             ; PORTMAP Server
   111 UDP PORTMAP             ; PORTMAP Server
   135 UDP NCSLLBD             ; NCS LLBD SERVER
   161 UDP SNMPD               ; SNMP AGENT
   162 UDP SNMPQE              ; SNMPQE AGENT
   512 TCP REXECD              ; REXECD SERVER (REXEC)
   514 TCP REXECD              ; REXECD SERVER (RSH)
   515 TCP LPSERVE             ; LP SERVER
   520 UDP ROUTED              ; ROUTED SERVER
   750 TCP VMKERB              ; KERBEROS SERVER
   750 UDP VMKERB              ; KERBEROS SERVER
   751 TCP ADMSERV             ; KERBEROS DATABASE SERVER
   751 UDP ADMSERV             ; KERBEROS DATABASE SERVER
  2049 UDP VMNFS               ; NFS Server
```

Figure 104. Part of a Sample PROFILE TCPIP for Multiple Web Servers

Notes:

1 User IDs WEBSHAR1, WEBSHAR2, and WEBSHAR3 will be started when TCP/IP starts execution.

2 In this example, user IDs WEBSHAR1, WEBSHAR2, and WEBSHAR3 are all bound to port 80.

A.2 TCP/IP Performance Considerations

If your Web server delivers increasingly higher transaction rates, you might run out of TCP/IP resources. It is also possible that your Web server has very slow response times compared with other TCP/IP applications on the same system (for example, TELNET or FTP). If this is the case, the following section offers solutions for common TCP/IP performance problems.

A.2.1 Buffer Pool Sizes

Each request needs one session; for example, getting a page, getting an image, or executing a CGI. A session is not terminated immediately after the request is satisfied, but stays in a pending state for a limited time. This can be easily shown with the NETSTAT ALLCONN command (see Figure 106 on page 192).

This behavior is not only true for Web servers, but for *any* TCP/IP application (for example, TELNET, FTP, LPR and so on).

Therefore, the number of Socket Control Blocks (SCBs) and Transmission Control Blocks (TCBs) needed by TCP/IP may be higher than the number of sessions active at any one time.

You can easily verify the amount of buffers that were needed in the past by using the NETSTAT POOLSIZE command. For an example, see Figure 107 on page 193 and Figure 108 on page 194.

Socket Control Blocks (SCBs)

The number of allocated SCBs specifies the maximum number of sockets that can be active at the same time.

SCBs are not used exclusively for Web servers. They are shared with other TCP/IP services such as TELNET and FTP.

The default number of SCBs is 256, which quickly becomes too low if you are serving many pages or images within a short period. You can increase the value of this default up to 2000.

The SCB default number is specified in the PROFILE TCPIP (or *system_name* TCPIP respectively, if the same disk is shared among multiple systems) file, which is usually located on TCPIP 191 (see Figure 105 on page 191). The keyword for SCBs is *SCBPOOLSIZE*.

After installation, check the low-water values in the NETSTAT POOLSIZE command response more often, to prevent server-not-available conditions. See Figure 107 on page 193 and Figure 108 on page 194 for more details.

Transmission Control Blocks (TCBs)

The number of allocated TCBs specifies the maximum number of sessions which can be active at the same time.

The TCBs are not used exclusively for Web servers. They are shared with other TCP/IP services such as TELNET and FTP.

The default number of TCBs is 256, which quickly becomes too low. If you are serving many pages or images within a short time, you can increase the default up to 2000.

It is specified in the PROFILE TCPIP (or *system_name* TCPIP respectively, if the same disk is shared among multiple systems) file, which is usually located on TCPIP 191 (see Figure 105 on page 191). The keyword for TCBs is *TCBPOOLSIZE*.

After installation, check the low-water values in the NETSTAT POOLSIZE command response more often, to prevent server-not-available conditions. See Figure 107 on page 193 and Figure 108 on page 194 for details.

Sample PROFILE TCPIP

```
;
; Use statements below to alter sizes of free pools.
; See section in this manual on TCP/IP Configuration
; Commands for more information.
  ACBPOOLSIZE                 1000    ; Default = 1000
  ADDRESSTRANSLATIONPOOLSIZE  1500    ; Default = 1500
  CCBPOOLSIZE                 150     ; Default = 150
  DATABUFFERPOOLSIZE          160     ; Default = 160
  ENVELOPEPOOLSIZE            750     ; Default = 750
  IPROUTEPOOLSIZE             300     ; Default = 300
  LARGEENVELOPEPOOLSIZE       50      ; Default = 50
  RCBPOOLSIZE                 50      ; Default = 50
  SCBPOOLSIZE                 256     ; Default = 256          1
  SKCBPOOLSIZE                256     ; Default = 256
  SMALLDATABUFFERPOOLSIZE     0       ; Default = 0
  TCBPOOLSIZE                 256     ; Default = 256          2
  UCBPOOLSIZE                 100     ; Default = 100
;
```

Figure 105. Sample PROFILE TCPIP

Notes:

1. You can increase the default value of 256 for SCBPOOLSIZE up to the limit of 2000 (which was removed in TCP/IP for VM V2R4).
2. You can increase the default value of 256 for TCBPOOLSIZE up to the limit of 2000 (which was removed in TCP/IP for VM V2R4).

Sample NETSTAT Command Displays

```
netstat allconn
VM TCP/IP Netstat V2R3

Active Transmission Blocks
User Id  Conn  Local Socket            Foreign Socket          State
---- --  ----  ----- ------            ------- ------          -----
INTCLIEN 1000  *..TELNET               *..*                    Listen
INTCLIEN 1004  9.164.195.15..TELNET    9.12.14.31..1029        Established
INTCLIEN 1083  9.164.195.15..TELNET    9.12.14.94..1025        Established
EWEB002  1001  *..84                  *..*                    Listen
EWEB003  1002  *..84                  *..*                    Listen
EWEB001  1003  *..84                  *..*                    Listen
WEBSHARE 1023  9.164.195.15..80        9.12.14.94..1503        Closed
WEBSHARE 1025  9.164.195.15..80        9.12.14.94..1504        Closed
WEBSHARE 1026  9.164.195.15..80        9.12.14.94..1506        Closed
WEBSHARE 1030  *..80                   *..*                    Listen
WEBSHARE UDP   *..1029                 *..*                     UDP
WEBSHARE 1096  9.164.195.15..80        9.12.14.94..1494        Time-wait
WEBSHARE 1099  9.164.195.15..80        9.12.14.94..1495        Time-wait
WEBSHARE 1107  9.164.195.15..80        9.12.14.94..1496        Time-wait
WEBSHARE 1037  9.164.195.15..80        9.12.14.94..1498        Time-wait
WEBSHARE 1132  9.164.195.15..80        9.12.14.94..1502        Time-wait
WEBSHARE 1127  9.164.195.15..80        9.12.14.94..1510        Time-wait
WEBSHARE 1103  9.164.195.15..80        9.12.14.94..1512        Time-wait
WEBSHARE 1019  9.164.195.15..80        9.12.14.94..1513        Established 1
AFPMGR   1020  *..999                  *..*                    Listen
DSMSERV  1007  *..1500                 *..*                    Listen
Ready; T=0.04/0.08 21:48:05
```

Figure 106. Sample NETSTAT ALLCONN Command Output

Notes:

1 Only one session is established to port 80, but there are also many sessions in Closed and Time-wait state, which increases the number of SCBs and TCBs needed.

```
netstat poolsize
VM TCP/IP Netstat V2R2
TCPIP Free pool status:
Object     # alloc     # free      Lo-water    Permit size
======     =======     ======      ========    ===========
ACB          1000         978         788         100
CCB           150          86          79          10
Dat buf       160          71          34          32
Sm dat buf      0           0           0           1
Env           750         748         726          75
Lrg env        50          49          32          10
RCB            50          49          49           3
SCB           256         185          44          17          1
SKCB          256         231         223          17
TCB           256         133           0          17          2
UCB           100          86          84           6
Ready; T=0.01/0.02 15:52:34
```

Figure 107. Sample NETSTAT POOLSIZE Command Output (Low). In this sample the number of free SCBs and TCBs is too low.

Notes:

1 The number of allocated Socket Control Blocks (SCBs) is set to the default value of 256. The minimum number of free sessions left is 44 (low-water). Even if 185 sessions are free right now, the SCB limit should be increased to ensure that TCP/IP does not run out of free SCBs.

2 The amount of allocated Transmission Control Blocks (TCBs) has been set to the default value of 256. The minimum number of free sessions left is 0 (low-water). Even if 133 sessions are free right now, the TCB limit *must* be increased to ensure that TCP/IP does not run out of free TCBs.

```
netstat poolsize
VM TCP/IP Netstat V2R3
TCPIP Free pool status:
Object       # alloc     # free      Lo-water    Permit size
======       =======     ======      ========    ===========
ACB             1000        985           836          100
CCB              150        118           116           10
Dat buf          160        144           129           32
Sm dat buf         0          0             0            1
Tiny dat buf       0          0             0            0
Env              750        750           741           75
Lrg env           50         49            40           10
RCB               50         50            50            3
SCB             2000       1960          1834          133        1
SKCB             256        245           243           17
TCB             2000       1990          1864          133        2
UCB              100         95            94            6
Ready; T=0.03/0.06 21:50:08
```

In this sample, the number of free SCBs and TCBs is high enough. The limit should not be too high, because each control block needs some additional virtual storage in the TCP/IP Service Virtual Machine.

Figure 108. Sample NETSTAT POOLSIZE Command Output (High)

Notes:

1. The number of allocated Socket Control Blocks (SCBs) is set to the maximum value of 2000 (which was removed in TCP/IP for VM V2R4).
2. The number of allocated Transmission Control Blocks (TCBs) is set to the maximum value of 2000 (which was removed in TCP/IP for VM V2R4).

A.3 Domain Name Server Resolution, Security, and Performance Issues

The Web server may use a Domain Name Server (DNS) for name resolution. (For example, it resolves an IP address of 9.12.14.1 to wtscpok.itso.ibm.com.) This information is then stored in log files or is used for authentication of the client.

A.3.1 Security

Some Web servers protect the URL by IP address or host name and domain name.

This can cause security problems, as IP addresses are not secure (any workstation can change the IP address to whatever it likes), and therefore the resolved host name and domain name may also point to an incorrect client.

TIP

You should not use protection of critical resources based on IP addresses or domain names, because they can be easily faked.

See *Web-Enabling VM Resources*, SG24-5347 (which will be available at a later date) for a more extensive discussion of this topic.

A.3.2 Performance

If the Domain Name Server (DNS) is slow, or not available at all, the Web server waits until the timeout value, defined in TCPIP DATA, is reached. The TCPIP DATA file is usually located on TCPMAINT 592.

The default value for RESOLVERTIMEOUT is 30 seconds. Therefore, the Web server will wait 30 seconds for each request if the DNS is unavailable.

To prevent this situation, use the following solutions:

- If you do not need the client user ID and domain information in a CGI or authentication mechanism, comment out any NSINTERADDR statement in TCPIP DATA.
- If you need the client user ID and domain information in a CGI or authentication mechanism, set the RESOLVERTIMEOUT value very low (for example, to three).

Do not use the IDENT protocol to get the additional user ID authentication. It may slow down the Web server, especially if the client is connected through a slow network. In Webshare only the default setting for IDENT is *ON*.

```
;
; NSINTERADDR specifies the Internet address of the name server.
; LOOPBACK (14.0.0.0) is the default value (your local name server).
; If a name server will not be used, then do not code an NSINTERADDR
; statement (Comment out the NSINTERADDR line below).  This will cause
; all names to be resolved via site table lookup.
;
NSINTERADDR  14.0.0.0               1
;
; NSPORTADDR specifies the foreign port of the name server.
; 53 is the default value.
;
NSPORTADDR 53
;
; RESOLVEVIA specifies how the Resolver is to communicate with the
; name server.  TCP indicates use of TCP virtual circuits.  UDP
; indicates use of UDP datagrams.  The default is UDP.
;
RESOLVEVIA UDP
;
; RESOLVERTIMEOUT specifies the time in seconds that the Resolver
; will wait to complete an open to the name server (either UDP or TCP).
; The default is 30 seconds.
;
RESOLVERTIMEOUT 3                   2
;
```

Figure 109. Part of Sample TCPIP DATA File

Notes:

1. If you do not use a local DNS, comment out this line by inserting a semicolon in the first column, or change the IP address to the nearest DNS you want to use.
2. The resolver timeout limit should be as short as possible to avoid delays if the DNS is not reachable.

Appendix B. Quick Start to SFS

If you are already familiar with the VM/ESA Shared File System (SFS), there is no need for you to read this section. It was written for those who want to jump onto the SFS “train” right now, but are intimidated by the number of pages in the IBM manual *VM/ESA CMS File Pool Planning, Administration, and Operation*, SC24-5751.

Note: This IBM publication is very comprehensive and worth reading.

The following recommendations result from the experiences of hands on use, collected from internal IBM sites.

B.1 SFS Highlights

The VM/ESA Shared File System (SFS) is the next generation file server that improves upon the standard CMS minidisk file system.

Some of the advantages of using SFS are:

- A DASD saving of up to 40%, because the empty space on a minidisk necessary to work with CMS files is not preallocated
- Easy cross-system access to data, similar to that of the Network File System (NFS) of UNIX
- Centralized data management due to the SFS server carrying out the data access
- The use of a hierarchical file system to organize data better, similar to that of workstation OSs and MVS
- The use of a hierarchical Byte File System (BFS) to provide workstation or Open-Edition oriented UNIX-type files (fully integrated into CMS)
- Multiple READ/WRITE sharing of data

You should consider the following if using this method of storing data in VM:

- How to define file pools with correct parameters to satisfy security
- How to migrate user data from minidisks to file pools
- How to set up processes around file pools, such as auditing, accounting, and health checking
- How to use the APPC/VM Support (AVS machine) for cross-system access to file pools
- What conventions are necessary to ensure unique file pool names
- What must be done to satisfy security requirements against the SFS subsystem
- What logic will be needed for additional necessary tools

In this chapter, answers are given to some of these questions.

B.2 SFS - The Full Picture

For full production use you might want to consider installing the complete spectrum of SFS file pools, which includes the suite of DFSMS/VM servers, TSAF (in case of non-ISFC clusters), local file pools, and remote access.

Figure 110 shows the complete structure of this configuration.

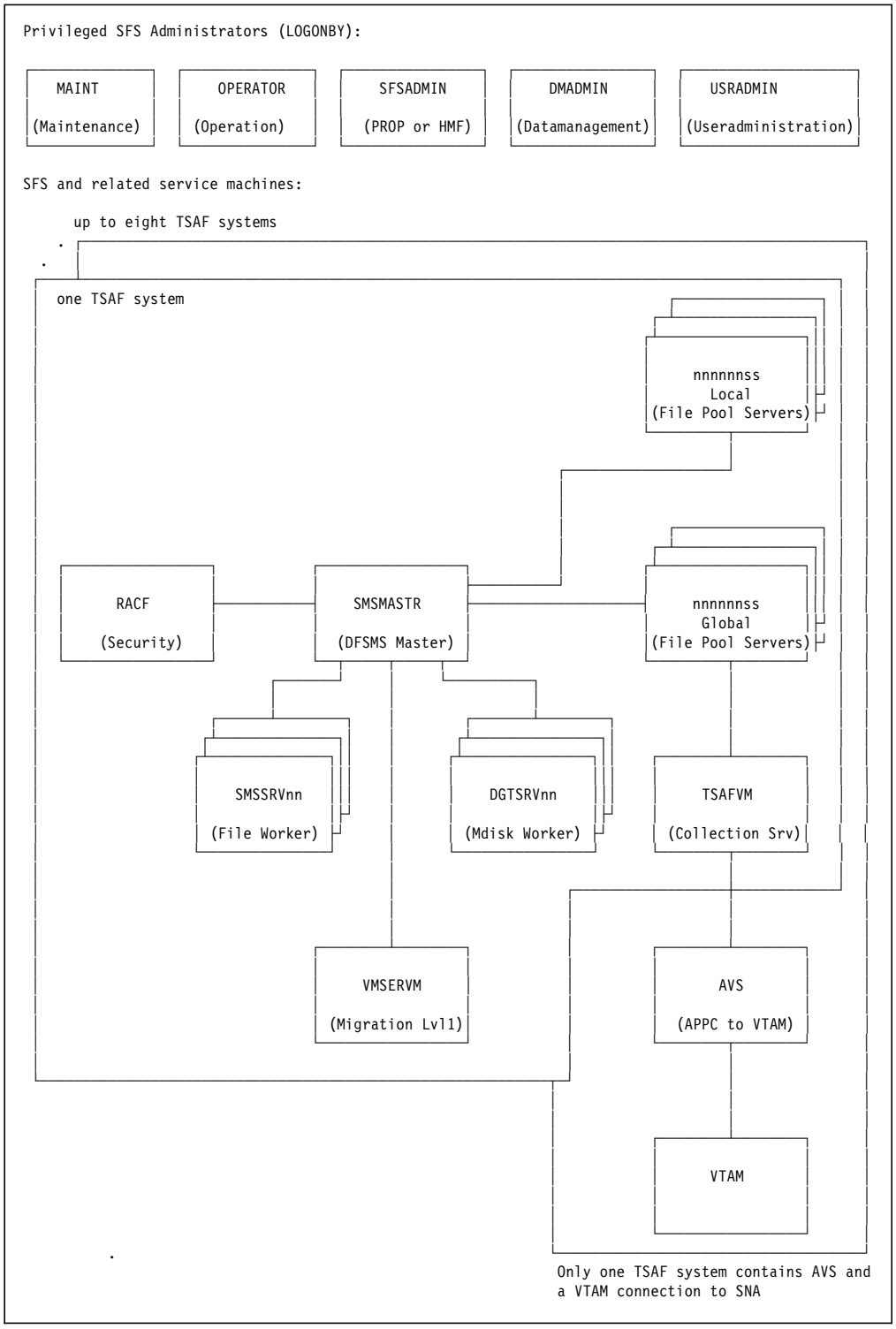
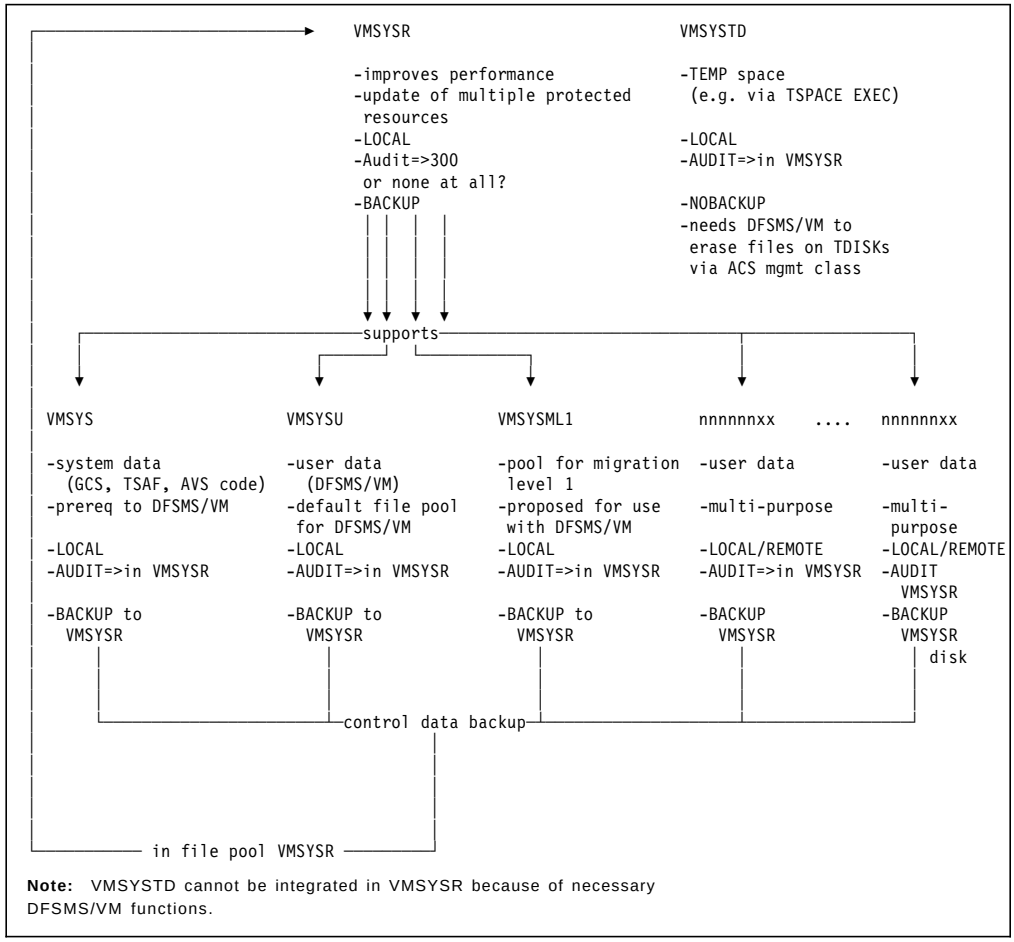


Figure 110. Overall Service Machine Structure



Note: VMSYSTD cannot be integrated in VMSYSR because of necessary DFSMS/VM functions.

Figure 111. Overview of Shared File Pools on a System

Note that Figure 111 represents the *full* picture of shared file pools on a system.

To simply get started, however, you can use the VMSYSU file pool that was automatically installed when you installed VM/ESA.

Alternatively, you can set up a remote accessible user file pool, as discussed in the following sections.

B.3 SFS Commands

SFS Administration is straightforward. The commands are intuitive and easy to remember. For a quick list of these commands, use the VM HELP command to point to either of these areas:

`HELP TASKS SFSADMIN` and `HELP SFSADMIN`.

The user ID enrolled as Shared File System administrator requires access to the delivered sample programs on MAINT 193. Access to this disk is mandatory if DFSMS/VM is installed. DFSMS/VM is a no-cost feature of VM/ESA that allows you to use system-managed storage.

B.4 SFS Servers and File Pools

The following sections provide an overview of the building blocks that make up an SFS server.

B.4.1 SFS Servers

An SFS server is an XAUTOLOGged virtual machine that executes code (through the CMSFILES segment) that maintains files on a set of control, log, catalog, and data disks for several users.

File pools that are not named VMSYS can be made available remotely. This can be done in a VM cluster of up to eight VM systems using local connections (TSAF or CTC), or to any other system connected by AVS and VTAM. If access to the file pools through VTAM is not desired, the VM systems in the cluster can be connected using ISFC.

B.4.2 File Pools

A file pool is a collection of minidisks managed by SFS. It contains user directories, files, and associated control information. Many user's directories and files can be contained in a single file pool.

Storage Groups 2 and up	A subset of minidisks within a file pool containing user directories and files
Catalog Storage Group	The first storage group containing information about the user directories and files (formerly FST entries)
Log Data	A set of two minidisks containing log information in dual write fashion
Control Data	A minidisk containing information about the space of this file pool, especially allocation and availability

B.4.3 File Pool Names

You can name your SFS file pools whatever you want. However, keep the following points in mind when naming file pools:

- SFS file pools starting with VMSYS can only be made available locally, even if they are defined as remote.

  In fact, if you want to prevent a file pool from becoming available outside your system, begin its name with VMSYS. The suggested file pool naming conventions are:

VMSYS	Control information for DFSMS, GCS, TSAF, AVS library
VMSYSU	System work disk, log for DFSMS
VMSYSR	CRR recovery
VMSYSML1	DFSMS migration level 1
VMSYSTD	Temporary disk space

- Names of file pools that are to be made available remotely (using TSAF or TSAF/AVS/VTAM) must be unique. The best method is to link the file pool name closely to the well-known RSCS node ID or PVM/VTAM log on node ID. We recommend that you use that name and drop the characters "VM" (that are usually part of a VM node) to obtain a string not longer than six characters, and then concatenate a numeric suffix of 00 to 99 to that name.

  Of course, not all node IDs contain the character string "VM." In this case, abbreviate that name to six characters to try to retain its typicality so it is

recognized as being a file pool belonging to its related node. Just make sure the name is unique.

Using this method, node ID STUTVM1, for example, could provide file pools STUT101 to STUT199, and node ID CBEPROFS could provide file pools CBEPRF01 to CBEPRF99.

We also found it handy, in production use, to have the user ID serving the file pool be identical to the name of the file pool. Even if there is no technical requirement to do so, it makes management easier.

Table 19 shows the file pools needed for the fully-managed SFS scenario, with the addition of VMSYSTD (temporary space) and user file pools.

Table 19. File Pools Necessary for IBM Products

File pool	SVM	Description	Scope
VMSYS:	*VMSYS*	Control files for DFSMS, GCS, TSAF, AVS required, if DFSMS is used	Local
VMSYSU:	*VMSYSU*	DFSMS work files, log for DFSMS default	Local
VMSYSR:	*VMSYSR*	CRR recovery server pool required, for performance improvements and multiple updated resources	Local
VMSYSML1:	*VMSYSML1*	DFSMS master server for migration purposes proposed, if DFSMS/VM is used	Local
VMSYSTD:	*VMSYSTD*	Temporary disk space proposed SFS usage	Local
nnnnnss:	*nnnnnnss*	User file pools	Remote or local

Note: File pools starting with VMSYS are local file pools. You cannot access data in these file pools remotely.

File Pool Disk Sizes

Table 20 shows starting minidisk sizes. If you want to calculate specific storage requirements, refer to *VM/ESA CMS File Pool Planning, Administration, and Operation*, SC24-5751.

Table 20 (Page 1 of 2). Disk Sizes

Mdisk	Size_{3380}	Size_{3390}	Description
191	01 cyl	01 cyl	Work disk containing PROFILE, DMSPARM, and POOLDEF
300	20 cyl	16 cyl	Audit data disk
301	18 cyl	15 cyl	Control disk containing a map of used and unused blocks

Table 20 (Page 2 of 2). Disk Sizes			
Mdisk	**Size$_{3380}$**	**Size$_{3390}$**	**Description**
302	50 cyl	40 cyl	Log 1, containing a list of recent actions
303	50 cyl	40 cyl	Log 2, same as above (dual log)
304	221 cyl	184 cyl	Group 1, containing catalog information about user files
310 to 3nn	442 cyl	368 cyl	Group 2 to n, containing user data

Note: Groups (1 to n) can be added as needed.

Rules

The following rules apply when allocating space for the various minidisks:

- Make sure the size of the control disk is allocated large enough to accommodate the required number of 512 byte blocks for the size of the database your server is to serve. It cannot be increased at a later time.
- The control disk, catalog disk, and one log disk should not be on the same pack as any user data disk.
- The two logs must be on different packs.
- The sizes for 310 to 3*nn* should be 442 (3380) and 368 (3390) cylinders.
- Minidisk parts for the storage group minidisks should be large enough to use the capacity of the devices, and yet hold about the same amount of megabytes of data. This makes movement and administration easier.
  - The sizes for 310 to 3*nn* should be 442 (3380) and 368 (3390) cylinders.
  - For smaller file pools, the storage group 1 minidisk size should be defined in multiples of 221 (3380) or 184 (3390).

B.4.4 CP Directory Statements

The following statements defined in the CP directory affect the work of an SFS server:

IUCV * IDENT File_pool GLOBAL

In this case, file_pool can be either the file pool name (*nnnnnnss*) or RESANY.

RESANY allows the server to start up any file pool; if the file pool name is given explicitly, only that file pool can be started.

The difference between GLOBAL and PRIVATE resources is important for APPC/VM communication and cross-system access of file pools. See *VM/ESA Connectivity Planning, Administration, and Operation*, SC24-5756 for a detailed discussion.

IUCV ALLOW

This entry is necessary, since it permits any user to connect to the file pool without explicit authorization.

OPTION MAXCONN xxxx NOMDCFS APPLMON QUICKDSP

For details concerning the MAXCONN parameter, see the discussion in B.4.5, “Size Allocations” on page 203.

The NOMDCFS parameter allows the server machine to use minidisk caching at a rate that is not limited by the Fair Share Limit.

APPLMON is necessary for DIAGNOSE X′DC′ to generate information for performance monitoring.

QUICKDISP is necessary to dispatch the server immediately without waits in the eligible list.

Note: The ACCT option is not used, because we account for allocated space with an outside program.

POSIXOPT SETIDS ALLOW

This statement allows a BFS server to have CP change its POSIX security value in support of opening executable files.

SHARE RELATIVE 1500

It is written that “SET SHARE REL will place the server machine in a more favorable position in the dispatch queue. Why 1500? The default setting for a user is 100. The server supports multiple users, such as 15, so we recommend 1500. This should be set in line with other server settings, such as VTAM.”

This is the value recommended in the IBM documentation.

NAMESAVE DFSMSSEG

This statement allows access to a restricted segment of DFSMS. If you want to use the DFSMS migration functions, you must have this access. If you do not plan to use DFSMS this statement is unnecessary, but it does not cause any problem.

MINIOPT NOMDC

This statement must follow the minidisk definition that you do not want to be cached in the expanded storage. It is recommended that you do not cache SFS, so use this statement for:

1. SFS control file disk (301 disk)
2. SFS log disks (302 and 303 disks)
3. All CRR File Pool minidisks (300, 301, 302, 303, 304, 310 upwards)

B.4.5 Size Allocations

Normally, each value defined for a specific file pool should be considered when more detailed information about the business of the file pool is given.

In the following example, we assume the file pool contains data of OFFICE users, where every user has a 191 minidisk of a default size and all other user data is

held in file pools. It is also assumed that each user ID is enrolled in a file pool by default.

MAXUSER

The documented formula for maximum enrolled users is

MAXIMUM enrolled users=300×
(the number of system defined users÷the number of system active users)

For OFFICE-type systems, the quotient "number of system-defined users÷the number of system active users" is typically between three and five. This allows you to enroll 900 to 1500 users per file pool server.

We recommend that you start each file pool with MAXUSER 1000.

Note: You can enroll more than 1000 users. This value determines how much logical catalog space is available.

MAXCONN

The documented formula for the MAXCONN value of the OPTION statement is:
For user file pools

MAXCONN=(USERS×3)+DISKS

For CRR recovery

MAXCONN=(USERS+RESOURCES+20)

where

USERS The number of logged-on SFS users expected during peak time (or the total number of logged-on CMS users)

DISKS The number of minidisk containing user data (>= 310 addresses)

RESOURCES The number of resources (such as SFS file pools) participating in CRR

Setting this value too high may cause virtual storage to be exhausted in the server machine.

We recommend you set MAXCONN=3000 for user file pools and MAXCONN=1100 for CRR recovery.

MAXDISKS

This value defines the maximum number of minidisks that will ever be allocated in the file pool.

We recommended a value of 500.

Size of the Control Minidisk 301

The size of this disk is critical.

It is recommended that you overestimate this value. Assuming that all user data of a node ID (such as image and logical system) will fit in one file pool (because the user does not need to specify the file pool ID explicitly in each ACCESS command), then the size of a file pool can reach 50 GB (estimated from BARS/VM backup sizes for the largest German OFFICE system).

Therefore, we recommend you use 12105 blocks for the control minidisk. This corresponds to 18 cylinders of 3380 or 17 cylinders of 3390 with 512-byte formatting.

Note: The size is estimated for file pools containing user data of OFFICE-type systems. For products other or other data, this size can be different.

Log Disks 302 and 303

These are two identically-sized disks. They must be on same type of device. The log disk must be able to hold all changes to the control data between backups, and should not be filled to more than 80% capacity because then the automatic backup will start. *This must be prevented during prime shift.*

You will probably be safe if you assume a nightly backup run, but you should allow for one night per week with no backup because of system shutdown, space problems, or for other reasons.

The formula to calculate the log size is: LOGSIZE=HOURS×ACTIVE_USERS×0.08 (the result unit is in MB) where

HOURS Number of hours between backup runs

ACTIVE_USERS Average number of logged-on users with system interactions within a minute interval

0.08 Estimate of the rate at which SFS log data is generated

We assume HOURS=48 and ACTIVE_USERS=1/8 of enrolled users. We also assume enrolled users=MAXUSER=133 per minute.

This means a log disk size of 855 cylinders of 3380 or 710 cylinders 3390 is reasonable.

B.4.6 Backup

The data backup support provided by SFS is not reflected in this appendix. The BACKUP statement in the DMSPARMS file specifies that control data backup is to be done.

Internally we recommend that an external backup program such as BARS/VM or VM:BACKUP be used for data backup/restore and disaster backup.

Note: You might want to examine the function provided by the FILESPACE UNLOAD, and FILESPACE RELOAD commands provided by VM/ESA Version 2.

B.4.7 Data Security Considerations

The SFS encompasses a full set of security-relevant features. By default, a directory or file is owned by the user that creates it. It is visible only to the owner. However, owners can make their directories and files sharable in any conceivable mode. This is controlled by the CMS command GRANT.

Owners of directories or files can grant public access to these objects. This is a function that requires monitoring.

B.5 SFS Performance Considerations

The performance options discussed are included in the sample directories. Some options should be set in server startup files as well.

However, two additional performance options should be observed when migrating to SFS.

1. Use a BUFFSIZE=64 parameter in the DEFNUC macro in the DMSNGP ASSEMBLE file used to create your CMS Nucleus.
2. Always install and start the CRR server file pool, even if you make no use of it. This improves the performance of the other servers by avoiding extra code provided to bypass the CRR machine.

B.6 Transparent Service Access Facility/Inter-System Facility for Communication

User IDs within the TSAF/ISFC collection must be unique, meaning that a user ID (for example, JB) must be owned by the same person (Joe Bloggs) on *all* systems within the TSAF/ISFC collection.

B.7 APPC/VM to VTAM Support (AVS)

If you want to grant access to directories and files in your file pools to users outside your TSAF/ISFC collection, you must install AVS.

Refer to the appropriate VM publications about the meanings of statements in the AVS server′s control files (AGWPROF) and the end-users control files (*COMDIR NAMES).

B.8 Data Facility Storage Management Subsystem (DFSMS/VM)

When DFSMS/VM is shipped, it comes with three servers which perform directory and file operations, and three servers which perform minidisk operations. You have to decide if three is enough, or if you want to use these functions at all and do without those servers.

B.9 Installation

The following discussion assumes you are performing directory maintenance through DIRMAINT, and that your ESM (external security manager) is RACF. However, it should be easy to apply the same steps also to any other way of maintaining VM.

1. Log on to your usual maintenance user ID (MAINT).

   Assume that this user ID has DIRMAINT DIRM_STAFF and RACF SPECIAL privileges.

   It is also assumed that you have an active CMSFILES segment that contains a PSEG CMSFILES which, in turn, contains the two LSEGs DMSDAC and DMSSAC.

2. Create the SFS server user ID.

User ID *xxxxxxnn* is the file pool server for file pool, where *xxxxxx* is a unique character string derived from the node ID and *nn* is a numeric value of 00 to 99 (see B.4.3, “File Pool Names” on page 200).

```
USER     XXXXXXNN xxxxxxxx 036M 036M G
 INCLUDE  DEFAULT
 AUTOLOG  AUTOLOG2 MAINT
 OPTION   MAXCONN  1000 ACCT NOMDCFS  APPLMON QUICKDSP
 SHARE    RELATIVE 1500
 NAMESAVE DFSMSSEG                              1
 POSIXOPT SETIDS ALLOW
 IUCV     ALLOW
 IUCV     *IDENT   RESANY   GLOBAL
 CONSOLE  0009     3215     T        SFSADMIN
 MDISK    0191 x    AUTOG 0003 VMSSCSCR MR      2
 LINK     $SMS2212 0200     0192     RR          3
 LINK     $MAINT   0193     0193     RR          4
 MDISK    0301 x    AUTOG 0018 VMSSCSCR MR      5
 MINOPT   NOMDC
 MDISK    0302 x    AUTOG 0050 VMSSCSCR MR      6
 MINOPT   NOMDC
 MDISK    0303 x    AUTOG 0050 VMSSCSCR MR      6
 MINOPT   NOMDC
 MDISK    0304 x    AUTOG 0221 VMSSCSCR MR      7
 MDISK    0310 x    AUTOG 0442 VMSSCSCR MR      8
 MDISK    0311 x    AUTOG 0442 VMSSCSCR MR      8
```

Note: The 191 minidisk needs be formatted.

Figure 112. Sample Directory Entry for xxxxxxnn

Notes:

1. This allows connection to the DFSMS/VM shared segment.
2. This is the file pool server 191 (A-) minidisk.
3. This is the DFSMS/VM code disk to be linked if installed.
4. This is the CMS auxiliary disk, which is required.
5. This is the control disk, similar to the CMS Minidisk allocation map.
6. These are the two LOG file disks, which keep track of changes since the last LOG file backup.
7. This forms storage group 1, which contains the catalog information.
8. This forms storage group 2, which is the first storage group holding user data. Remember that you can have multiple of user storage groups. You should use multiple storage groups because this can improve performance.

Use the directory sample *xxxxxxnn* as shown in Figure 112 to initially create one user ID, *xxxxxx*01. You can add up to 98 additional user IDs at a later time if you want to provide additional file pools. Provide minidisks extents for disks 310, 311, and as many disks as you want, depending on how large you want the pool to be. If the space you want to provide extends over several volumes specify additional minidisks, such as 312, 313, and so on.

3. Define all resources (user IDs and minidisks) to RACF (or to your external security manager product, if any).
4. Ensure that the $MAINT 193 minidisk (and, optionally, your DFSMS/VM code disk) are generally accessible.

5. Link and format the 191 minidisks of the previously-created server user ID.
6. Add the service machine user ID in your AUTOLOG2 virtual machine's XAUTOLOG procedure in the following sequence:

Startup Sequence for File Pool Servers

```
VMSYSR
Wait until it is up
VMSYSML1
VMSYSTD
VMSYSU
VMSYS
xxxxxxnn
Wait until VMSYSML1 is up
SMSMASTR
RMSMASTR
TSAFVM
AVSVM
```

Note: This is the startup sequence for a full-blown installation. For now, without DFSMS and the other file pools, just add the user ID you have generated.

B.10 Bring Up SFS

For the first setup, logon to user ID *xxxxxx*01 and perform the following steps, as shown in Figure 113.

1. Edit a *xxxxxx*01 DMSPARMS file similar to the following:

```
ADMIN      DFSMS    MAINT    DMADMIN  SFSADMIN USRADMIN
ADMIN      SMSMASTR RMSMASTR
ADMIN      SMSSRV01 SMSSRV02 SMSSRV03
ADMIN      DGTSRV01 DGTSRV02 DGTSRV03
AUDIT
BACKUP
DFSMS
FILEPOOLID XXXXXXnn
REMOTE
SAVESEGID  CMSFILES
USERS      1000
```

Figure 113. Sample DMSPARMS File

Note:

- You may want to change the first line to include user IDs as SFS administrators other than the ones provided.

  Also ensure that the SAVESEGID CMSFILES statement matches the name on your system.

- Change the pool name in the FILEPOOL statement to a file pool name you want.

  File the new statement.

2. Create a PROFILE EXEC, as shown in Figure 114 on page 209.

```
/* PROFILE EXEC for Shared File System Server */
Address Command
Parse upper arg parm
'SET AUTOREAD OFF'
'CP SET EMSG ON'
'CP SET RUN ON'
'CP SET PF11 RETRIEVE FORWARD'
'CP SET PF12 RETRIEVE BACKWARD'
'CP SET PF24 RETRIEVE'
'SET CMSTYPE HT'
'ACCESS 193 C'              /*$MAINT 193 disk; SFS system code       */
'ACCESS 192 D'              /*DFSMS/VM code disk                     */
'SYNONYM SYSPROG'
'SET CMSTYPE RT'
Select
 When parm='SETUP' then start = 0
 When parm='GO' then start = 1
 When parm='' & Linesize() = 0 then start = 1
 Otherwise
  Do
   Say "Wrong parameter was issued|"
   Say "Valid parameters are:"
   Say "SETUP = Do not start the Fileserver"
   Say "GO = Start the Fileserver"
   Exit
  End
End
'SEGMENT RESERVE DFSMSSEG'  /*only when using DFSMS                  */
'SET CMSTYPE HT'
'ACCESS 301 E/E * CONTROL'
 arc=rc
'SET CMSTYPE RT'
If arc¬=0 then Exit
'RELEASE E'
nnnnnnss = Userid()

'ESTATE SFSLIB CSLLIB *' /* Storage policy exit available?         */
If rc = 0
 then 'RTNLOAD DMSSFSEX (FROM SFSLIB SYSTEM'
'ESTATE AUDIT EXEC *'
If rc = 0 & start=1
 then 'EXEC AUDIT'          /* call audit file handling            */
If nnnnnnss <> 'VMSYSR'
 then 'EXEC FILESERV DEFBACKUP DISK 'nnnnnnss' BACKUP VMSYSR:'nnnnnnss'.BACKUP'
 else 'EXEC FILESERV DEFBACKUP DISK 'nnnnnnss' BACKUP E'

If start = 1 then 'EXEC FILESERV START'
frc = rc
If frc > 4 & ,                      /* only in error cases         */
   Substr(Diag(24,-1),13,1) = 2     /* we are disconnected         */
 then push 'CP IPL'                 /*    rerun the previous IPL   */
Exit frc
```

Figure 114. Sample PROFILE EXEC of User Data File Pool Servers

3. Create a *xxxxxxnn* PDOOLDEF file, as shown in Figure 115.

nnnnnnss POOLDEF for a general file pool server

```
00001 MAXUSERS=1000
00002 MAXDISKS=500
00003 DDNAME=AUDIT    DISK   FN=nnnnnnss   FT=AUDIT    FM=E
00004 DDNAME=BACKUP   DISK   FN=nnnnnnss   FT=BACKUP   FM=
00005 BKDIRID=VMSYSR:nnnnnnss.BACKUP
00006 DDNAME=CONTROL         VDEV=301
00007 DDNAME=LOG1            VDEV=302
00008 DDNAME=LOG2            VDEV=303
00009 DDNAME=MDK00001        VDEV=304      GROUP=1
00010 DDNAME=MDK00002        VDEV=310      GROUP=2
00011 DDNAME=MDK00003        VDEV=311      GROUP=2
```

Figure 115. Sample POOLDEF File for the File Pool Server Managing User Data

4. Execute the FILESERV GENERATE command to format and initialize the SFS catalog, log, and data disks.

   You will find the command enters XEDIT with file $$TEMP $POOLDEF shown on the screen. Enter the following XEDIT subcommands:

   - DEL*
   - GET *xxxxxxnn* POOLDEF A

   Note: If you have provided additional minidisks, such as 311, you have to add DDNAME statements for them. Just count the MDK0000x and GROUPs and provide a minidisk for each.

   Then FILE the file.

   At this point, the screen will remain unchanged for some time while formatting takes place. After some time, the following statements will appear:

```
DMS4PD3400I Initializing begins for DDNAME = CONTROL
DMS4PD3400I Initializing ends for DDNAME = CONTROL
DMS4PD3400I Initializing begins for DDNAME = MDK00001
DMS4PD3400I Initializing ends for DDNAME = MDK00001
DMS4PD3400I Initializing begins for DDNAME = MDK00002
DMS4PD3400I Initializing ends for DDNAME = MDK00002
DMS4PD3400I Initializing begins for DDNAME = LOG1
DMS4PD3400I Initializing ends for DDNAME = LOG1
DMS4PD3400I Initializing begins for DDNAME = LOG2
DMS4PD3400I Initializing ends for DDNAME = LOG2
DMS5FD3032I Filepool server has terminated
DMSWFV1120I File NODEIDnn POOLDEF A1 has been created or replaced
DMSWFV1117I FILESERV processing ended at nn:nn:nn on dd mmm yyyy
Ready;
```

   Enter PROFILE GO to start the file pool server.

   After the server is ready for operator communication, enter #CP DISC.

You may want to repeat these steps for each additional user pool server you want to activate.

The Shared File System is now up and running.

Appendix C. Sample Universal CGI for Use with All VM Web Servers

```
 1  /* REXX ---------------------------------------------------------------*/
 2  /* VM/ESA Version 2                                                   */
 3  /* Licensed Materials - Property of IBM Corp.                         */
 4  /* (c) Copyright IBM Corp. 1996, 1998                                 */
 5  /* All rights reserved.                                               */
 6  /* US Government Users Restricted Rights -                            */
 7  /* Use, duplication or disclosure restricted by                       */
 8  /* GSA ADP Schedule Contract with IBM Corp.                           */
 9  /* Sample CGI REXX code to process a WWW form                         */
10  /*                                                                    */
11  /*--------------------------------------------------------------------*/
12  /*                                                                    */
13  /* Sample CGI to serve the available VM Web Server.                   */
14  /*                                                                    */
15  /* Currently supported interfaces:                                    */
16  /*                                                                    */
17  /* VM/ESA:                                                            */
18  /*                                                                    */
19  /* Webshare      (Richard M. Troth - freeware)                        */
20  /* EnterpriseWeb (Beyond Software Incorporated)                       */
21  /* VM:Webgateway  (Sterling Software)                                 */
22  /* VM:Webgateway in Webshare compatibility mode                       */
23  /*                                                                    */
24  /*--------------------------------------------------------------------*/
25  /*                                                                    */
26  /* To write a CGI you only need to call the following sequence:       */
27  /*                                                                    */
28  /* call Init                                                          */
29  /* call Read                                                          */
30  /*                                                                    */
31  /* Thereafter, the variable stem post. contains all data posted to    */
32  /* the request (for example all data posted from a form).             */
33  /* See the comments in the Read routine for the format.               */
34  /* The variable query_string will contain the query_string (if        */
35  /* defined) from the url that referenced this cgi program             */
36  /*                                                                    */
37  /* For any line you wish to return to the client just call:           */
38  /*                                                                    */
39  /* call write 'STRING' 'whatever you like'                            */
40  /* call write 'STRING' 'It is now' time()                             */
41  /*                                                                    */
42  /* It is also possible to write a single variable or a complete       */
43  /* variable stem:                                                     */
44  /*                                                                    */
45  /* call write 'VAR varname'                                           */
46  /* or                                                                 */
47  /* call write 'STEM stemname.'                                        */
48  /*                                                                    */
49  /* You can also write server directivies:                             */
50  /*                                                                    */
51  /* call header 'STRING' 'Status: 200 OK'                              */
52  /*                                                                    */
53  /*--------------------------------------------------------------------*/
54  /*--------------------------------------------------------------------*/
55  /*                                                                    */
```

```
 56  /* The following variables are available for                          */
 57  /* Webshare                                                           */
 58  /*                                                                    */
 59  /* path: /htbin/test?test=yes                                         */
 60  /*                                                                    */
 61  /* the following variables are set in this universal cgi:             */
 62  /* ( the '.' in HTTP_ACCEPT=*.*; should be translated to '/',         */
 63  /*   but is treated as 'end of comment' here)                         */
 64  /*                                                                    */
 65  /* ARG=test=yes                                                       */
 66  /* ARGS=test=yes                                                      */
 67  /* CGIUSERS=TROTH TWOOLBRI JFORD CWHITE BHUNTER MAINT VETTER D1MATTW  */
 68  /* CLIENT=@9.12.14.94                                                 */
 69  /* CONTENT_LENGTH=0                                                   */
 70  /* FILE=/htbin/test                                                   */
 71  /* FILEPOOLS=VMSYSER                                                  */
 72  /* GATEWAY_INTERFACE=CGI/1.1                                          */
 73  /* HTTP_ACCEPT= *.*; q=0.300, application/octet-stream; q=0.100, text */
 74  /* HTTP_USER_AGENT=IBM-WebExplorer-DLL/v1.1f                          */
 75  /*     Other HTTP_* variables will be set as sent from the browser.   */
 76  /* HTTP=HTTP/1.0                                                      */
 77  /* LOCALHOST=wtscpok.itso.ibm.com                                     */
 78  /* LOCALPORT=83                                                       */
 79  /* LOGPIPE=>> httpd logfile a                                         */
 80  /* OPS_ENV=CMS                                                        */
 81  /* PATH_TRANSLATED=TEST *CGI *                                        */
 82  /* PATH=/htbin/test                                                   */
 83  /* PORT=83                                                            */
 84  /* POST.0=TEST                                                        */
 85  /* POST.TEST=yes                                                      */
 86  /* QUERY_STRING=test=yes                                              */
 87  /* REQUEST_METHOD=GET                                                 */
 88  /* SCRIPT_NAME=/htbin/test                                            */
 89  /* SERVER_NAME=wtscpok.itso.ibm.com                                   */
 90  /* SERVER_PORT=83                                                     */
 91  /* SERVER_PROTOCOL=HTTP/1.0                                           */
 92  /* SERVER_SOFTWARE_LIST=Webshare/1.2.3 VM_ESA/2.2.9507 CMS/11.507 REX */
 93  /* SERVER_SOFTWARE=Webshare/1.2.3                                     */
 94  /* SERVER_URL=SERVER_URL                                              */
 95  /* SOCKET=2                                                           */
 96  /* THREADSELECT=                                                      */
 97  /* USERWEBS=ON                                                        */
 98  /* VERB=GET                                                           */
 99  /* VERBOSE=VERBOSE                                                    */
100  /* VERSION=VM/CMS Webshare 2.4                                        */
101  /* VM_WEBSERVER=0                                                     */
102  /* VRM=1.2.4                                                          */
103  /*                                                                    */
104  /*--------------------------------------------------------------------*/
105  /*--------------------------------------------------------------------*/
106  /*                                                                    */
107  /* The following variables are available for                          */
108  /* EnterpriseWeb                                                      */
109  /*                                                                    */
110  /* path: /htbin/test?test=yes                                         */
111  /*                                                                    */
112  /* the following variables are set in this universal cgi:             */
113  /* ( the '.' in HTTP_ACCEPT=*.*; should be translated to '/',         */
114  /*   but is treated as 'end of comment' here)                         */
```

```
115  /*                                                                  */
116  /* ACCESSINT=60                                                     */
117  /* ACCESSMODES=*                                                    */
118  /* ARG=test=yes                                                     */
119  /* ARGS=test=yes                                                    */
120  /* AUTOGROUP=OFF                                                    */
121  /* AUTOINDEX=ON                                                     */
122  /* CLIENT=9.12.14.94                                                */
123  /* CONFIG_FILE=STEVE CONFIG *                                       */
124  /* CONTENT_LENGTH=0                                                 */
125  /* COOKIES=0                                                        */
126  /* DEFAULT_REALM=EnterpriseWeb                                      */
127  /* DEFAULTBLOCK=61440                                               */
128  /* DEFAULTMEDIA=Text/Plain 8BIT - - -                               */
129  /* ENTRYFILTER=OFF                                                  */
130  /* EWEB.THREAD=2                                                    */
131  /* FASTDISKS=*                                                      */
132  /* FASTMODE=OFF                                                     */
133  /* FASTOVERRIDE=ON                                                  */
134  /* FASTPATH=/VM/                                                    */
135  /* FASTPUBDISKS=*                                                   */
136  /* FILE=/htbin/test.cgi                                             */
137  /* FILEPOOL=SFSLSY4:                                                */
138  /* FQDN=OFF                                                         */
139  /* GATEWAY_INTERFACE=CGI/1.1                                        */
140  /* GMTOFFSET=-5.00                                                  */
141  /* HEADER_DIGESTION=ON                                              */
142  /* HEADERS=1                                                        */
143  /* HOSTADDR=9.12.14.94                                              */
144  /* HTTP_ACCEPT= *.*; q=0.300, application/octet-stream; q=0.100, text */
145  /* HTTP_USER_AGENT=IBM-WebExplorer-DLL/v1.1f                        */
146  /*     Other HTTP_* variables will be set as sent from the browser.  */
147  /* HTTP=HTTP/1.0                                                    */
148  /* HTTPS=OFF                                                        */
149  /* IDENT=NOIDENT                                                    */
150  /* IF_MODIFIED_SINCE=ON                                             */
151  /* IPFILTER=OFF                                                     */
152  /* LAST_MODIFIED=ON                                                 */
153  /* LISTF_ALLFILE=ALLFILE ALLOC                                      */
154  /* LOCALHOST=wtscpok.itso.ibm.com                                   */
155  /* LOCALPORT=90                                                     */
156  /* LOGFORMAT=STANDARD                                               */
157  /* LOGLINE=@STD 9.12.14.94 - - [12/Nov/1996:14:30:04 -5.00] "GET /bon */
158  /* LOGPIPE=CONSOLE                                                  */
159  /* MOZILLA_SUPPORT=ON                                               */
160  /* OPS_ENV=CMS                                                      */
161  /* PATH_TRANSLATED=TEST CGI J                                       */
162  /* PATH=/htbin/test.cgi                                             */
163  /* PORT=90                                                          */
164  /* POST.0=TEST                                                      */
165  /* POST.TEST=yes                                                    */
166  /* QUERY_STRING=test=yes                                            */
167  /* REMOTE_ADDR=9.12.14.94                                           */
168  /* REQUEST_METHOD=GET                                               */
169  /* REQUEST=GET /htbin/test.cgi?test=yes HTTP/1.0                    */
170  /* RESPECT_FM=J                                                     */
171  /* REVERSEDNS=OFF                                                   */
172  /* ROOT=SFSLSY4:VETTER.R96LS2505                                    */
173  /* ROOTTYPE=SFS                                                     */
```

```
174  /* SCRIPT_NAME=/htbin/test.cgi                                        */
175  /* SERVER_COOKIES=ON                                                  */
176  /* SERVER_HEADERS=ON                                                  */
177  /* SERVER_MODES=ABCDEFGHISXYZ                                         */
178  /* SERVER_NAME=wtscpok.itso.ibm.com                                   */
179  /* SERVER_PORT=90                                                     */
180  /* SERVER_PROTOCOL=HTTP/1.0                                           */
181  /* SERVER_SOFTWARE_LIST=EnterpriseWeb/1.1.1 VM_ESA/2.2.9507 CMS/11.50 */
182  /* SERVER_SOFTWARE=EnterpriseWeb/1.1.1                                */
183  /* SERVER_URL=http://wtscpok.itso.ibm.com:90/                         */
184  /* SFSPATH=SFSLSY4:VETTER.R96LS2505.BONUSPAK2.CGIBIN                  */
185  /* SIGL=155                                                           */
186  /* SOCKET=2                                                           */
187  /* SSI=ON                                                             */
188  /* SSIDISKS=*                                                         */
189  /* THREAD=2                                                           */
190  /* THREADING=OFF                                                      */
191  /* THREADSELECT=SELECT THREAD2                                        */
192  /* USERWEB_FM=J                                                       */
193  /* USERWEBLINKP=CHALLENGE                                             */
194  /* USERWEBS=ON                                                        */
195  /* USERWEBSPECIFY=ON                                                  */
196  /* VERB=GET                                                           */
197  /* VERBOSE=TERSE                                                      */
198  /* VERSION=EnterpriseWeb 1.1.1 VM/CMS                                 */
199  /* VM_WEBSERVER=0                                                     */
200  /* VRM=1.1.1                                                          */
201  /*                                                                    */
202  /*--------------------------------------------------------------------*/
203  /*--------------------------------------------------------------------*/
204  /*                                                                    */
205  /* The following variables are available for                          */
206  /* VM:Webgateway (Webshare compatibility mode)                        */
207  /*                                                                    */
208  /* path: /htbin/test?test=yes                                         */
209  /*                                                                    */
210  /* the following variables are set in this universal cgi:             */
211  /* ( the '.' in HTTP_ACCEPT=*.*; should be translated to '/',         */
212  /*   but is treated as 'end of comment' here)                         */
213  /*                                                                    */
214  /* ARGS=test=yes                                                      */
215  /* CGIUSERS=                                                          */
216  /* CONTENT_LENGTH=0                                                   */
217  /* FILE=/HTBIN/test.cgi                                               */
218  /* GATEWAY_INTERFACE=CGI/1.1                                          */
219  /* HOSTADDR=9.12.14.94                                                */
220  /* HTTP_ACCEPT=*.*; q=0.300,application/octet-stream; q=0.100,text/pl */
221  /* HTTP_USER_AGENT=IBM-WebExplorer-DLL/v1.1f                          */
222  /*     Other HTTP_* variables will be set as sent from the browser.   */
223  /* HTTP=HTTP/1.0                                                      */
224  /* LOCALHOST=WTSCPOK.ITSO.IBM.COM                                     */
225  /* LOCALPORT=81                                                       */
226  /* LOGPIPE=CONSOLE                                                    */
227  /* OPS_ENV=CMS                                                        */
228  /* PATH_USER=                                                         */
229  /* PATH=/HTBIN/test.cgi                                               */
230  /* PORT=81                                                            */
231  /* POST.0=TEST                                                        */
232  /* POST.TEST=yes                                                      */
```

```
233  /* QUERY_STRING=test=yes                                              */
234  /* REMOTE_ADDR=9.12.14.94                                             */
235  /* REQUEST_METHOD=GET                                                 */
236  /* SCRIPT_NAME=/HTBIN/test.cgi                                        */
237  /* SERVER_NAME=WTSCPOK.ITSO.IBM.COM                                   */
238  /* SERVER_PORT=81                                                     */
239  /* SERVER_PROTOCOL=HTTP/1.0                                           */
240  /* SERVER_SOFTWARE=VM:Webserver/2.2                                   */
241  /* THREADSELECT=                                                      */
242  /* USER=                                                              */
243  /* USERWEBS=ON                                                        */
244  /* VERB=GET                                                           */
245  /* VERBOSE=TERSE                                                      */
246  /* VERSION=VM/CMS VM:Webserver 2.2                                    */
247  /* VM_WEBSERVER=1                                                     */
248  /* VRM=2.2                                                            */
249  /* X_PATH_TRANSLATED_OUTCOME=NO_PATH_INFO                             */
250  /* X_SCRIPT_NAME_TRANSLATED=TEST CGI SFS SFSLSY4:VETTER.R96LS2505.BON */
251  /* X_SERVER_SCHEME=http                                               */
252  /*                                                                    */
253  /*--------------------------------------------------------------------*/
254  /*--------------------------------------------------------------------*/
255  /*                                                                    */
256  /* The following variables are available for                          */
257  /* VM:Webgateway (not Webshare compatibility mode)                    */
258  /*                                                                    */
259  /* path: /htbin/test?test=yes                                         */
260  /*                                                                    */
261  /* the following variables are set in this universal cgi:             */
262  /* ( the '.'  in HTTP_ACCEPT=*.*; should be translated to '/',        */
263  /*   but is treated as 'end of comment' here)                         */
264  /*                                                                    */
265  /* AUTH_TYPE=                                                         */
266  /* CONTENT_LENGTH=                                                    */
267  /* CONTENT_TYPE=                                                      */
268  /* GATEWAY_INTERFACE=CGI/1.1                                          */
269  /* HTTP_ACCEPT=*.*; q=0.300,application/octet-stream; q=0.100,text/pl */
270  /* HTTP_USER_AGENT=IBM-WebExplorer-DLL/v1.1f                          */
271  /*     Other HTTP_* variables will be set as sent from the browser.   */
272  /* OPS_ENV=CMS                                                        */
273  /* PATH_INFO=                                                         */
274  /* PATH_TRANSLATED=                                                   */
275  /* POST.0=TEST                                                        */
276  /* POST.TEST=yes                                                      */
277  /* QUERY_STRING=test=yes                                              */
278  /* REMOTE_ADDR=9.12.14.94                                             */
279  /* REMOTE_HOST=                                                       */
280  /* REMOTE_IDENT=                                                      */
281  /* REMOTE_USER=                                                       */
282  /* REQUEST_METHOD=GET                                                 */
283  /* SCRIPT_NAME=/HTBIN/TEST.CGIEXEC                                    */
284  /* SERVER_NAME=WTSCPOK.ITSO.IBM.COM                                   */
285  /* SERVER_PORT=81                                                     */
286  /* SERVER_PROTOCOL=HTTP/1.0                                           */
287  /* SERVER_SOFTWARE=VM:Webserver/2.2                                   */
288  /* VM_WEBSERVER=1                                                     */
289  /* X_AUTH_PRIVATE_INFO=                                               */
290  /* X_AUTH_VERIFIED=0                                                  */
291  /* X_PATH_TRANSLATED_OUTCOME=NO_PATH_INFO                             */
```

```
292 /* X_SCRIPT_NAME_TRANSLATED=TEST CGIEXEC SFS SFSLSY4:VETTER.R96LS2505 */
293 /* X_SERVER_SCHEME=http                                               */
294 /*                                                                    */
295 /*--------------------------------------------------------------------*/
296 /*--------------------------------------------------------------------*/
297
298 /* Overview
299   When a CGI REXX script is executing, anything written out to the
300   standard out will be sent to the browser as part of the HTML page
301   AFTER the beginning HTML header is generated.  This makes it very
302   easy to generate any type of page.*/
303
304 /*--------------------------------------------------------------------*/
305 /*                                                                    */
306 /* To write a CGI you only need to call the following sequence:       */
307 /*                                                                    */
308 /* call Init                                                          */
309 /* call Read                                                          */
310 /*                                                                    */
311 /* Thereafter, the variable stem post. contains all data posted to    */
312 /* the request (for example all data posted from a form).             */
313 /*                                                                    */
314 /* For any line you wish to return to the client just call:           */
315 /*                                                                    */
316 /* call write 'STRING' 'whatever you like'                            */
317 /* call write 'STRING' 'It is now' time()                             */
318 /*                                                                    */
319 /* It is also possible to write a single variable or a complete       */
320 /* variable stem:                                                     */
321 /*                                                                    */
322 /* call write 'VAR varname'                                           */
323 /* or                                                                 */
324 /* call write 'STEM stemname.'                                        */
325 /*                                                                    */
326 /*--------------------------------------------------------------------*/
327
328 trace o
329
330 parse arg arg
331
332 call init                  /* Initialize variable environment for CGI */      ➞219
333
334 call read                 /* Get additional information passed to CGI */      ➞222
335
336 /*--------------------------------------------------------------------*/
337 /*- local processing start -------------------------------------------*/
338 /*- This is just sample code that shows you all the variables set by -*/
339 /*- the 'init' and 'read' routines and where they came from.         -*/
340 /*--------------------------------------------------------------------*/
341
342 pipe '(end ? name testcgi)',
343      'rexxvars',
344   '|  buffer',
345   '|  var sourcestring',          /* Rexx "parse source" string     */
346   '|  drop 1',
347   '|  change 1.2 /v /=/',         /* Decode 'rexxvars' format       */
348   '|  change 1.2 /n //',
349   '|  join',                      /* Bring name and value together  */
350   '|  sort',
```

```
351   '|  change /</&lt;/',           /* Change these so they don't     */
352   '|  change />/&gt;/',           /*   confuse the web browser!     */
353   '|h:nlocate 1.5 "HTTP_"',       /* All HTTP header variables      */
354   '|p:nlocate 1.5 "POST."',       /* Any data POSTed                */
355   '|c:lookup fs = f1 w1 details', /* find CGI environment vars      */
356   '|z:faninany',                  /* Include HTTP_ variables        */
357   '|  stem cgivars.',             /* List of CGI variables & values */
358   '|  locate 1.13 "QUERY_STRING="', /* Find the QUERY_STRING var    */
359   '|  append literal',
360   '|  var qstring',               /* The QUERY_STRING from browser  */
361   '?h:',                          /* HTTP_ header variables         */
362   '|z:',
363   '?p:',                          /* Any data POSTed from client    */
364   '|  stem postvars.',
365  ,         /* List the CGI standard environment variables           */
366   '?  literal AUTH_TYPE CONTENT_LENGTH CONTENT_TYPE GATEWAY_INTERFACE',
367               'PATH_INFO PATH_TRANSLATED QUERY_STRING REMOTE_ADDR',
368               'REMOTE_HOST REMOTE_IDENT REMOTE_USER REQUEST_METHOD',
369               'SCRIPT_NAME SERVER_NAME SERVER_PORT SERVER_PROTOCOL',
370               'SERVER_SOFTWARE',
371   '|  split',
372   '|c:',
373   '|l:lookup fs = f1 w1 details', /* find local environment vars    */
374   '|  stem localvars.',           /* List of local vars and values  */
375  ,         /* List the variables set by this exec                   */
376   '?  literal $HEADER_WRITTEN $SERVER ARG OPS_ENV PIPE PIPE_OUTPUT',
377               'RC SIGL VM_WEBSERVER',
378   '|  split',
379   '|l:',
380   '|x:nlocate 1.7 "X_HTTP_"',     /* Header vars set by this program*/
381   '|  stem envvars.',              /* Other environment variables    */
382   '?x:',
383   '|  stem xtraheaders.'
384
385 /* Write standard CGI server directives */
386 If RC=0 then do
387    Call header 'STR Status: 200 OK'           /* response status    */        →224
388    Call header 'STR Content-type: text/html'  /* The data type sent */        →224
389 end
390 else do
391    error = '500 Internal error RC=' RC
392    call header 'STR Status:' error                                           →224
393    call header 'STR Content-type: text/plain'    /* Just plain text */        →224
394    call write  'VAR error'           /* Put the error msg on screen */       →225
395    exit
396 end
397
398 /*- Create an HTML document to display on the browser ----------------*/
399 call write 'STR <HTML>'                                                      →225
400 /*- Un comment the following line to test an ISINDEX tag -------------*/
401 /* call write 'STR <HEAD><ISINDEX></HEAD>' */
402 call write 'STR <BODY><B>The QUERY_STRING:</B><PRE>'                         →225
403 call write 'VAR qstring'                                                    →225
404 call write 'STR </PRE><HR><B>Any POST or decoded QUERY_STRING data:</B>'    →225
405 call write 'STR <BR>(QUERY_STRING is decoded by this CGI only when no',     →225
406               'data is POSTed from your browser.)<PRE>'
407 call write 'STEM postvars.'                                                 →225
408 call write 'STR </PRE><HR><B>Standard CGI environment variables:</B><PRE>' →225
409 call write 'STEM cgivars.'                                                 →225
```

```
410  call write 'STR </PRE><HR><B>Other server environment variables:</B><PRE>'    ➡225
411  call write 'STEM envvars.'                                                    ➡225
412 ┌If xtraheaders.0 > 0 then do
413 │   call write 'STR </PRE><HR><B>Variables set from request fields:</B><PRE>'  ➡225
414 │   call write 'STEM xtraheaders.'                                             ➡225
415 └end
416  call write 'STR </PRE><HR><B>Other variables set by the CGI program:</B>'     ➡225
417  call write 'STR <PRE>Rexx Source string:' substr(sourcestring,3)              ➡225
418  /*- Show how PIPE and PIPE_OUTPUT variables are used: ----------------*/
419  pipe 'stem localvars. |' pipe_output
420  call write 'STR </PRE></BODY></HTML>'                                         ➡225
421
422  /*---------------------------------------------------------------------*/
423  /*- local processing end ----------------------------------------------*/
424  /*---------------------------------------------------------------------*/
425  exit
426
```

```
427 /*--------------------------------------------------------------------*/
428 /*- Initialize variable environment for CGI --------------------------*/
429 /*--------------------------------------------------------------------*/
430 /*                                                                    */
431 /* Init:                                                              */
432 /*   - First find out where I am (VM/ESA, OS/390 ......)              */
433 /*   - On VM/ESA find out what Server Software I am running on        */
434 /*     (Webshare, EnterprisWeb or VM:Webgateway)                      */
435 /*   - Load variable environment                                      */
436 /*                                                                    */
437 /*--------------------------------------------------------------------*/
438 Init:                                                          EntryPoint
439 $header_written = 0          /* Have any header fields been written?  */
440 parse source ops_env .
441 select
442   when ops_env = 'CMS' then do                       /* o.k. - it's VM */
443     /* If we are a Webshare based CGI, this is how we get CGI vars.   */
444     address command 'GLOBALV SELECT HTTPD GET SERVER_SOFTWARE'
445     /* If we are VM:Webgateway based, this nucleus extension exists.  */
446     address command 'NUCEXT CGI'
447     vm_webserver = (rc = 0)
448
449     /* Determine what style of CGI we are.                            */
450     select
451       /* If server_software is set, we are Webshare based CGI.        */
452       when left(server_software,13) = 'EnterpriseWeb',   /* Beyond   */
453         then $server = 'ENTERPRISEWEB'
454       when left(server_software,8)  = 'CMSHTTPD', /* Webshare V1.2.0 */
455          | left(server_software,8)  = 'Webshare', /* Webshare V1.2.3 */
456         then $server = 'WEBSHARE'
457       when left(server_software,12) = 'VM:Webserver',    /* Sterling */
458          | left(server_software,13) = 'VM:Webgateway',   /* Sterling */
459         then $server = 'VMWEBSERVER_COMPAT'
460       /* Not Webshare based, perhaps in VM:Webgateway?                */
461       when vm_webserver,                                 /* Sterling */
462         then $server = 'VMWEBSERVER'
463       /* We do not know where we are running.                         */
464       otherwise $server = '?????'
465     end
466
467     /* Determine how to address commands to PIPEs.                    */
468     address command 'SET CMSTYPE HT'
469     'callpipe hole'
470     pipe_rc = rc
471     address command 'SET CMSTYPE RT'
472     if pipe_rc = 0
473       then pipe = 'callpipe'
474       else pipe = 'PIPE'
475     drop pipe_rc
476
477     select
478       when $server = 'ENTERPRISEWEB',
479          | $server = 'WEBSHARE',
480          | $server = 'VMWEBSERVER_COMPAT',
481         then do
482
483 /*- now get the general server information available anytime ---------*/
484
485       'callpipe command GLOBALV SELECT HTTPD LIST',
```

```
                '| drop 1',
                '| strip leading',
                '| change //=/',
                '| nfind =GTYPE.'||,
                '| nfind =MTYPE.'||,
                '| nfind =PIPE.'||,
                '| nfind =TYPE.'||,
                '| varload'

/*- check for additional information ---------------------------------*/

      'streamstate input 1'         /* Stream with header records */
      if rc>=0 & rc<12 then
         /* Note: this may incorrectly combine some fields..      */
         'callpipe *.input.1:',       /* They should be variables */
               '| xlate w1 upper - _',
               '| spec fs : f1 1 f2-* strip 21',
               '| sort 1-20',
               '| join keylength 20 /, /',
               '| spec /=X_HTTP_/ 1 w1 next /=/ next 21-* next',
               '| varload'             /* Load as extension vars */

      'streamstate input 2'          /* Additional EWEB variables */
      if rc>=0 & rc<12 then
         'callpipe *.input.2: | varload'

      if datatype(eweb.thread,'W'),
         then threadselect = "SELECT THREAD" || eweb.thread
         else threadselect = ""                    /* Try default */

/*- and load if available --------------------------------------------*/

      'callpipe command GLOBALV' threadselect 'LIST',
               '| drop 1',
               '| strip leading',
               '| specs /=/ 1 1-* n',
               '| nfind =GTYPE.'||,
               '| nfind =MTYPE.'||,
               '| nfind =PIPE.'||,
               '| nfind =TYPE.'||,
               '| varload'

/*- Specify the pipe stage to use for pipelines output ---------------*/

      pipe_output = '*.output.0:'
   end
/*-------------------------------------------------------------------*/
   when $server = 'VMWEBSERVER' then do
      /* Sterling in non-compatibility mode */

      'CGI GETVAR * (STEM GETVAR.'

      address command 'PIPE rexxvars',
                        '| drop 1',
                        '| specs /=/ 1 3-* n',
                        '| join',
                        '| nfind =GETVAR.0'||,
                        '| find =GETVAR.'||,
                        '| specs fs . /=/ 1 f2-* n',
```

```
                        '| buffer',
                        '| varload'

/*- Specify the pipe stage to use for pipelines output ---------------*/

   pipe_output ='join * x0D25 65535',
               '| change ""CGI WRITE DOCUMENT',
                         '(TRANSLATE USENGLISH CRLF STRING "',
               '| command'

   drop getvar.

   end
/*--------------------------------------------------------------------*/
   otherwise do
      say 'Unknown Webserver software ('server_software')'
   end
 end
 otherwise do
    say 'Operating System not supported ('ops_env')'
    $server = '?????'
 end
end
 return
```

```
572 /*--------------------------------------------------------------------*/
573 /*- Get additional information for CGI --------------------------------*/
574 /*- This routine will obtain any posted data and place it in the     -*/
575 /*- stem POST.  POST.0 will contain the name of the variables and    -*/
576 /*- each POST.<var> will contain the value.  If no data was posted,  -*/
577 /*- but the QUERY_STRING contains values, they will be decoded into  -*/
578 /*- the POST. stem in the same way.  (Webshare and EWEB do this      -*/
579 /*- automatically, the code below will do it for VM:Webserver.)      -*/
580 /*- Note: This routine does not handle forms with repeated field     -*/
581 /*-       names or names will invalid Rexx symbol characters in them.-*/
582 /*- Consult the documentation for your web server for more help.     -*/
583 /*--------------------------------------------------------------------*/
584 Read:                                                          EntryPoint
585 select
586    when $server = 'ENTERPRISEWEB',
587       | $server = 'WEBSHARE',
588       | $server = 'VMWEBSERVER_COMPAT' then do
589
590    'callpipe (end ?)',
591          '*.input.0:',            /* Read input from the form      */
592          '|  xlate 1-* 05 40',     /* Convert any tab char to space */
593          '|  strip',               /* Ignore any empty ones         */
594          '|  locate 1',
595          '|  xlate fieldsep = f1 upper', /* Upper case field name   */
596          '|f:fanout',
597          '|  specs "=POST." 1 1-* next', /* Put it in varload fmt.  */
598          '|  varload',
599          '?f:',
600          '|  chop before string "="',   /* Save just the field name */
601          '|  join * " "',               /* Make 1 string of names   */
602          '|  append literal',           /* Always set a value       */
603          '|  var post.0'
604
605    end
606    when $server = 'VMWEBSERVER' then do
607
608       /* Read the posted document into the variable BLOCK.  */
609       /* Perform U.S. English ASCII to EBCDIC translation.  */
610       'CGI READ 1 (VAR BLOCK TRANSLATE USENGLISH'
611
612       If rc = 12 then do  /* RC=12 means EOF - nothing was read */
613          rc=0
614          /* If no post data read, but the QUERY_STRING is set */
615          /*   then decode it (like Webshare would do..)       */
616          if symbol('QUERY_STRING') = 'VAR' then
617             block=query_string
618          else
619             block=''
620       end
621
622       /* If any information was read, assume it is actually   */
623       /*   form encoded data (as Webshare based products do), */
624       /*   and break it down into the stem variable POST.     */
625       /* "SYMCHARS $." tells it to allow all valid Rexx       */
626       /*   symbol chars in the stem.  See the help for CGI    */
627       /*   URLDECODE for more information.                    */
628       if Length(block) <> 0 then
629          'CGI URLDECODE (VAR BLOCK MODE TRANSFORMED' ,
630                         'INTO POST. SYMCHARS $.',
```

```
631 │  │                      'TRANSLATE USENGLISH'
632 │  │   else
633 │  │      post.0=''
634 │  │
635 │  │   drop block
636 │  │
637 │  └end
638 │   otherwise
639 └end
640  return
641
642
```

```
/*-------------------------------------------------------------------*/
/*- Write CGI server directives (HTTP response fields) --------------*/
/*-------------------------------------------------------------------*/
Header:                                                    EntryPoint

parse arg ctl text
ctl = translate(ctl)
select
   when $server = 'ENTERPRISEWEB',
      | $server = 'WEBSHARE',
      | $server = 'VMWEBSERVER_COMPAT' then do

      /* Headers are written to the primary output stream before  */
      /* any data is written.                                     */
      /* The "Status" field is not an HTTP header, but a server   */
      /* directive that must be changed into the real HTTP        */
      /* response record for "Webshare" I/O interface web servers.*/
      If $header_written = 0 then    /* First field not written?  */
        select
          when abbrev('VAR',ctl,1) then do
            Parse value value(text) with ?first ?rest
            If ?first = 'Status:' then
              call value text,'HTTP/1.0' ?rest
          end
          when abbrev('STEM',ctl,3) then do
            Parse value value(text||'1') with ?first ?rest
            If ?first = 'Status:' then
              call value text||'1','HTTP/1.0' ?rest
          end
          when abbrev('STRING',ctl,3) then do
            Parse var text ?first ?rest
            If ?first = 'Status:' then
              text = 'HTTP/1.0' ?rest
          end
          otherwise
        end
      select
        when abbrev('VAR',ctl,1) then
          'output' value(text)
        when abbrev('STEM',ctl,3) then
          'callpipe stem' text'| *.output.0:'
        when abbrev('STRING',ctl,3) then
          'output' text
        otherwise
      end
   end
   when $server = 'VMWEBSERVER' then do
      select
        when abbrev('VAR',ctl,1) then
          'CGI WRITE HEADER (VAR' translate(text)
        when abbrev('STEM',ctl,3) then
          'CGI WRITE HEADER (STEM' translate(text)
        when abbrev('STRING',ctl,3) then
          'CGI WRITE HEADER (STRING' text
        otherwise
      end
   end
   otherwise
end
```

```
$header_written = 1     /* We wrote at least 1 header field */
return


/*--------------------------------------------------------------------*/
/*- Write records to CGI output ---------------------------------------*/
/*--------------------------------------------------------------------*/
Write:                                                          EntryPoint

parse arg ctl text
ctl = translate(ctl)
select
   when $server = 'ENTERPRISEWEB',
      | $server = 'WEBSHARE',
      | $server = 'VMWEBSERVER_COMPAT' then do

    /* Was a header written before we wrote any data? */
    if $header_written = 1 then do
       'output'               /* Need a blank line before the data */
       $header_written = 0    /* ..but only write it one time.     */
    end

    select
       when abbrev('VAR',ctl,1) then
          'output' value(text)
       when abbrev('STEM',ctl,3) then
          'callpipe stem' text'| *.output.0:'
       when abbrev('STRING',ctl,3) then
          'output' text
       otherwise
    end
   end
   when $server = 'VMWEBSERVER' then do
    /* Note: VM:Webgateway automatically writes the header seperator */
    select
       when abbrev('VAR',ctl,1) then
          'CGI WRITE DOCUMENT (TRANSLATE USENGLISH CRLF VAR',
          translate(text)
       when abbrev('STEM',ctl,3) then
          'CGI WRITE DOCUMENT (TRANSLATE USENGLISH CRLF STEM',
          translate(text)
       when abbrev('STRING',ctl,3) then
          'CGI WRITE DOCUMENT (TRANSLATE USENGLISH CRLF STRING' text
       otherwise
    end
   end
   otherwise
end
return
```

Appendix D. Special Notices

This publication is intended to assist technical professionals who wish to learn more about the different Web servers available for the VM/ESA platform. The information in this publication is not intended as the specification of any programming interfaces that are provided by VM/ESA Version 2 Release 2.0 or later Release. See the PUBLICATIONS section of the IBM Programming Announcement for VM/ESA Version 2 Release 2.0 for more information

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM′s product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM′s intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The information about non-IBM (″vendor″) products in this manual has been supplied by the vendor and IBM assumes no responsibility for its accuracy or completeness. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer′s ability to evaluate and integrate them into the customer′s operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Any pointers in this publication to external Web sites are provided for convenience only and do not in any manner serve as an endorsement of these Web sites.

Any performance data contained in this document was determined in a controlled environment, and therefore, the results that may be obtained in other operating environments may vary significantly. Users of this document should verify the applicable data for their specific environment.

Reference to PTF numbers that have not been released through the normal distribution process does not imply general availability. The purpose of including these reference numbers is to alert IBM customers to specific information relative to the implementation of the PTF when it becomes available to each customer according to the normal IBM PTF distribution process.

The following terms are trademarks of the International Business Machines Corporation in the United States and/or other countries:

AIX	APPC/VM
AS/400	BookManager
DFSMS	DFSMS/VM
IBM	ISSC
OfficeVision	OfficeVision/VM
OS/2	OS/390
PROFS	RACF
RISC System/6000	S/390
System/390	Virtual Machine/Enterprise Systems Architecture
VM/ESA	VTAM

The following terms are trademarks of other companies:

C-bus is a trademark of Corollary, Inc.

PC Direct is a trademark of Ziff Communications Company and is used by IBM Corporation under license.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

Microsoft, Windows, and the Windows 95 logo are trademarks or registered trademarks of Microsoft Corporation.

Java and HotJava are trademarks of Sun Microsystems, Inc.

CADAM	Cadam Incorporated
CATIA	Dassault Systemes
DCE	The Open Software Foundation
Helvetica	Linotype Company
NCS	Apollo Computer, Incorporated
Netscape	Netscape Communications Corporation
Network File System	Sun Microsystems, Incorporated
NFS	Sun Microsystems, Incorporated
NDIS	3Com Corporation and Microsoft Corporation
PC Magazine	Ziff Communications Company
Sun Microsystems	Sun Microsystems, Incorporated
X Window System	Massachusetts Institute of Technology

Other trademarks are trademarks of their respective companies.

Appendix E. Web Samples on the Net

Additional materials related to this publication are available for download.

Sample Programs

Program source, object, and additional programs are available through the Internet.

The programs will be available in VMARC and packed CMS format. If you have an Internet connection you can access them by anonymous FTP:

```
ftp ftp.almaden.ibm.com
user: anonymous
password: your ip address
dir
cd redbooks\SG244874
binary
mget *.*
```

These commands retrieve the README.1ST file, which you can review for additional information.

Note: This is an external FTP site. IBM internals will need to login to the firewall (tollbooth) to access the FTP site.

Internal IBMers may receive the code through the SG244874 and SAMPCGI PACKAGE located on the VMTOOLS repository. You may receive a copy of the package by entering the following CMS command:

```
TOOLS TO VMTOOLS GET SG244874 PACKAGE
TOOLS TO VMTOOLS GET SAMPCGI  PACKAGE
   -or-
TOOLS SENDTO RALVM17 VMTOOLS VMTOOLS GET SG244874 PACKAGE
TOOLS SENDTO RALVM17 VMTOOLS VMTOOLS GET SAMPCGI  PACKAGE
```

Appendix F. Related Publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

F.1 International Technical Support Organization Publications

For information on ordering these ITSO publications see “How to Get ITSO Redbooks” on page 235.

- *Accessing the Internet*, SG24-2597
- *Building the Infrastructure for the Internet*, SG24-4824
- *A Guide to the Internet Connection Servers*, SG24-4805
- *The Internet & the World Wide Web: A Time-Saving Guide for New Users*, SG24-2499
- *Java Network Security*, SG24-2109
- *Safe Surfing: How to Build a Secure WWW Connection*, SG24-4564
- *TCP/IP Tutorial and Technical Overview*, GG24-3376
- *VM/ESA Network Computing with Java and NetRexx*, SG24-5148
- *Web-Enabling VM Resources*, SG24-5347 (available at a later date)

F.2 Redbooks on CD-ROMs

Redbooks are also available on CD-ROMs. **Order a subscription** and receive updates 2-4 times a year at significant savings.

CD-ROM Title	Subscription Number	Collection Kit Number
System/390 Redbooks Collection	SBOF-7201	SK2T-2177
Networking and Systems Management Redbooks Collection	SBOF-7370	SK2T-6022
Transaction Processing and Data Management Redbook	SBOF-7240	SK2T-8038
Lotus Redbooks Collection	SBOF-6899	SK2T-8039
Tivoli Redbooks Collection	SBOF-6898	SK2T-8044
AS/400 Redbooks Collection	SBOF-7270	SK2T-2849
RS/6000 Redbooks Collection (HTML, BkMgr)	SBOF-7230	SK2T-8040
RS/6000 Redbooks Collection (PostScript)	SBOF-7205	SK2T-8041
RS/6000 Redbooks Collection (PDF Format)	SBOF-8700	SK2T-8043
Application Development Redbooks Collection	SBOF-7290	SK2T-8037

F.3 Other IBM Publications

These publications are also relevant as further information sources:

- *VM/ESA Performance*, SC24-5782
- *VM/ESA CMS File Pool Planning, Administration, and Operation*, SC24-5751
- *VM/ESA Connectivity Planning, Administration, and Operation*, SC24-5756

F.4 On the Web

The following are electronic publications, available from the World Wide Web at the URLs indicated.

- *The Internet Engineering Task Force (IETF) Home Page*

  http://www.ietf.org/

  There you will find links to all of the Internet standards, experimental and informational RFCs and draft documents of proposed Internet standards. Documents of interest include:
  - *RFC 1945 - Hypertext Transfer Protocol -- HTTP/1.0*
  - *RFC 2068 - Hypertext Transfer Protocol -- HTTP/1.1*
  - *RFC 1866 - Hypertext Markup Language - 2.0*
  - *RFC 1867 - Form-based File Upload in HTML*
  - *RFC 1942 - HTML Tables*
  - *RFC 1980 - A Proposed Extension to HTML: Client-Side Image Maps*
  - *RFC 1738 - Uniform Resource Locators (URL)*
  - *RFC 1630 - Universal Resource Identifiers (URI)*
  - *RFC 1808 - Relative Uniform Resource Locators*
  - *RFC 1413 - The WWW Common Gateway Interface Version 1.1 Identification Protocol (IDENT)*
- *An Introduction to Writing Webshare CGI Scripts*

  http://www.beyond-software.com/Products/Presentations/Webshare/WebshareCGIs.html
- *A Guide to URLs*

  http://www.netspace.org/users/dwb/url-guide.html
- *Hypertext Transfer Protocol Working Group*

  http://www.ics.uci.edu/pub/ietf/http/
- *The WWW Common Gateway Interface Version 1.1*

  http://www.ics.uci.edu/pub/ietf/http/related/draft-robinson-www-interface-01.txt
- Drop in for a visit at Melinda Varian's homepage at the following location. There you will find many links and information concerning REXX and Pipelines.

  http://pucc.princeton.edu/~Melinda/
- *Internet Software for VM* page. It is provided by the Beyond Software Corporation at URL:

  http://www.beyond-software.com/Software/Software.html
- Further information about Beyond Software Inc. and EnterpriseWeb/VM is found on the Web at the following address:

  http://www.beyond-software.com
- *VM and the Internet* page. Contains articles, transcripts, helpful links, and product information.

  http://www.vm.sterling.com/general/documents/webindex.html
- For further information about Sterling Software, Inc., the VM Software Division, and VM:Webgateway, go to:

  http://www.sterling.com/
  http://www.vm.sterling.com/
- *VM/ESA Operating System Home Page*

http://www.vm.ibm.com/

How to Get ITSO Redbooks

This section explains how both customers and IBM employees can find out about ITSO redbooks, redpieces, and CD-ROMs. A form for ordering books and CD-ROMs by fax or e-mail is also provided.

- **Redbooks Web Site** http://www.redbooks.ibm.com/

  Search for, view, download or order hardcopy/CD-ROMs redbooks from the redbooks Web site. Also read redpieces and download additional materials (code samples or diskette/CD-ROM images) from this redbooks site.

  Redpieces are redbooks in progress; not all redbooks become redpieces and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

- **E-mail Orders**

  Send orders via e-mail including information from the redbook order form to:

	IBMMAIL	**Internet**
In United States:	usib6fpl at ibmmail	usib6fpl@ibmmail.com
In Canada:	caibmbkz at ibmmail	lmannix@vnet.ibm.com
Outside North America:	dkibmbsh at ibmmail	bookshop@dk.ibm.com

- **Telephone Orders**

United States (toll free)	1-800-879-2755	
Canada (toll free)	1-800-IBM-4YOU	
Outside North America	(long distance charges apply)	
(+45) 4810-1320 - Danish	(+45) 4810-1220 - French	(+45) 4810-1270 - Norwegian
(+45) 4810-1420 - Dutch	(+45) 4810-1020 - German	(+45) 4810-1120 - Spanish
(+45) 4810-1540 - English	(+45) 4810-1620 - Italian	(+45) 4810-1170 - Swedish
(+45) 4810-1670 - Finnish		

This information was current at the time of publication, but is continually subject to change. The latest information for customers may be found at http://www.redbooks.ibm.com/ and for IBM employees at http://w3.itso.ibm.com/.

IBM Intranet for Employees

IBM employees may register for information on workshops, residencies, and redbooks by accessing the IBM Intranet Web site at http://w3.itso.ibm.com/ and clicking the ITSO Mailing List button. Look in the Materials repository for workshops, presentations, papers, and Web pages developed and written by the ITSO technical professionals; click the Additional Materials button. Employees may also view redbook, residency and workshop announcements at http://inews.ibm.com/.

IBM Redbook Fax Order Form

Fax your redbook orders to:

United States (toll free)	1-800-445-9269
Canada	1-403-267-4455
Outside North America	(+45) 48 14 2207 (long distance charge)

Please send me the following:

Title	Order Number	Quantity

First name Last name

Company

Address

City Postal code Country

Telephone number Telefax number VAT number

- Invoice to customer number
- Credit card number

Credit card expiration date Card issued to Signature

We accept American Express, Diners, Eurocard, Master Card, and Visa. Payment by credit card not available in all countries. Signature mandatory for credit card payment.

List of Abbreviations

ACI	access control interface
ACS	access control system
ADMIN	administrative/administration
AIC	advanced information change (Sterling Software)
AIM	Automated Install Manager (Sterling Software)
ALT	alternate
API	application program interface
APPC/VM	advanced program-to-program communication/virtual machine (IBM)
ARP	address resolution protocol
ASCII	American National Standard Code for Information Interchange
AVI	Audio Video Interlaced
AVS	APPC/VM VTAM support
BARS/VM	Backup Archive & Retrieval System (VMBARS)
CADAM	computer augmented design and manufacturing
CD-ROM	(optically read) compact disk - read only memory
CERN	Conseil Europeen pour la Recherche Nucleaire (European organization for nuclear research)
CGI	Common Gateway Interface (programs that provide services on the WWW)
CMS	conversational monitor system (VM-based software, IBM)
COMPID	component identifier
CONFIG	configuration/configure
CP	control program
CPU	central processing unit
CPUID	CPU identification
CRLF	carriage return/line feed
CRR	coordinated resource recovery (VM/ESA)
DASD	direct access storage device
DCE	data communication equipment
DCF	document composition facility (program product)
DDNAME	data definition name
DFSMS	Data Facility Storage Management Subsystem (MVS and VM)
DIR	directory
DIRMAINT	directory maintenance program
DNS	domain name service
DTE	data transmission exchange
EBCDIC	extended binary coded decimal interchange code
EPM	enhanced editor for PM (OS/2)
ESM	external security manager (VM/CMS SFS authorization exits)
EXEC	execution program
FBA	fixed block architecture
FST	file status table
FTP	file transfer protocol
GA	general availability
GB	gigabyte (10**9 bytes or 1,000,000,000 bytes) case should be GB
Gb	gigabit (10**9 bits or 1,000,000,000 bits) case should be Gb
GIF	graphic interchange format
GML	generalized markup language (text format language)
HMF	host management facilities (IBM program product)
HQ	headquarters
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
ICMP	Internet control message protocol
ID	identification/identifier
IETF	Internet Engineering Task Force
INEWS	information news facility (IBM)
INTERNET	a worldwide network of TCP/IP-based networks
IP	Internet protocol (ISO)
IPL	initial program load
IS	information services
ISBN	international standard book number
ISFC	inter-system facility for communications (VM/ESA)
ISO	International Organization for Standardization
ISSC	Integrated Systems Solutions Company Ltd.
IT	information technology
ITSO	International Technical Support Organization

IUCV inter-user communication vehicle

IVP installation verification procedure/program

JPEG Joint Photographic Experts Group (CCITT/ISO, multimedia standards)

KERBEROS security system used in IBM′s TCP/IP products

LAN local area network

LPD line printer daemon (AIX)

LRECL logical record length

MAC medium access control (token-ring)

MAINT maintenance

MIDI musical instrument digital interface

MIME Multipurpose Internet Mail Extensions (RFC 1344)

MPEG Moving Pictures Experts Group (CCITT/ISO, multimedia standards)

MTU maximum transmission unit (Internet protocols)

NCS network computing system

NCSA The National Center for Supercomputing Applications at the University of Illinois at Urbana-Champaign

NDIS network driver interface specification

NFS network file system (USA, Sun Microsystems Inc.)

OV OfficeVision product

PARM parameter

POP Post Office Protocol (Internet)

PSEG page segment

PTF program temporary fix

RACF resource access control facility

RECFM record format

REXEC remote execution protocol

REXX restructured extended executor language

RFC Request for Comment

RISC reduced instruction set computer/cycles

RSCS remote spooling communications subsystem (VM′s counterpart to MVS JES NJE)

RSH remote shell execution on a remote host (AIX)

SCB session control block

SFS shared file system (hierarchical sharable VM/CMS file system)

SHARE an association of IBM engineering/scientific customers with large computing systems

SMSG special message (VM)

SMTP simple mail transfer protocol (Ethernet, based on TCP/IP)

SNA systems network architecture (IBM)

SNMP simple network management protocol (a TCP/IP protocol)

SPOOL system peripheral operation-off-line

SSI server side include

SSL Secure Sockets Layer

SVM service virtual machine

TCB transfer command block

TCP transmission control protocol (USA, DoD)

TCP/IP Transmission Control Protocol/Internet Protocol (USA, DoD, ARPANET; TCP=layer 4, IP=layer 3, UNIX-ish/Ethernet-based system-interconnect protocol)

TCPIP Transmission Control Protocol Internet Protocol

TELNET U.S. Dept. of Defense′s virtual terminal protocol, based on TCP/IP

TIP totally integrated plan

TSAF transparent services access facility

UCB unit control block

UDP user datagram protocol (TCPIP)

UNIX an operating system developed at Bell Laboratories

URL Uniform Resource Locator

VADDR virtual address

VM virtual machine (IBM System 370 & 390)

VM/CMS virtual machine/conversational monitor system (IBM)

VM/ESA virtual machine/enterprise systems architecture (IBM)

VMNFS virtual machine network file system (see NFS)

VMTOOLS VM programs conference facility (IUO)

VOLSER volume serial

VTAM virtual telecommunications access method (IBM) (runs under MVS, VM, & DOS/VSE)

WWW World Wide Web (Internet)

WYSIWYG what you see is what you get

Index

D

E

F

G

H

I

J

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y

ITSO Redbook Evaluation

Web Server Solutions for VM/ESA
SG24-4874-01

Your feedback is very important to help us maintain the quality of ITSO redbooks. **Please complete this questionnaire and Fax it to: USA International Access Code + 1 914 432 8264 or:**

- Use the online evaluation form found at http://www.redbooks.ibm.com
- Send your comments in an Internet note to redbook@us.ibm.com

Which of the following best describes you?
__Customer __Business Partner __Solution Developer __IBM employee
__None of the above

Please rate your overall satisfaction with this book using the scale:
(1 = very good, 2 = good, 3 = average, 4 = poor, 5 = very poor)

Overall Satisfaction _____

Please answer the following questions:

Was this redbook published in time for your needs? Yes____ No____

If no, please explain:

_____

_____

_____

_____

What other redbooks would you like to see published?

_____

_____

_____

Comments/Suggestions: **(THANK YOU FOR YOUR FEEDBACK!)**

_____

_____

_____

_____

_____

SG24-4874-01
Printed in the U.S.A.