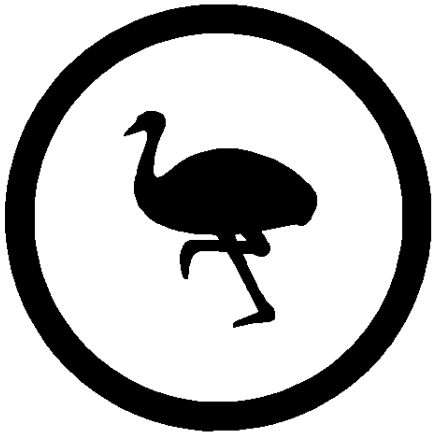
EMU Webmail Software Manual



Document Version 2.0

Contents

Chapter 1

Introduction

1.1 What is EMU Webmail?

EMU Webmail is a software package that runs on your web server, allowing your users access to their own POP or IMAP email accounts from any web browser. EMU Webmail has many advantages, including the following:

EMU Webmail is a universal client.

Other desktop-based email clients, like Eudora and Pegasus, require users to have configured software on their local machine. Not the case with EMU Webmail. The only requirement for using EMU Webmail is a web browser.

EMU Webmail is email security.

You shouldn't be forced to rely on someone you don't trust to safeguard sensitive data. Other service-based Webmail companies make you store your email on their computers. Why should you to trust them to make backups of your important data? How easy is it for them to view your files? Very easy. But not with EMU Webmail. EMU Webmail answers these security concerns by running on your own system and allowing you to store email locally, so you never have to worry that your data could get into the wrong hands. And when it is run off a secure webserver, EMU Webmail provides a very high level of security for email in transit.

EMU Webmail is email for everyone.

EMUmail wasn't designed for computer engineers. It was designed to be a simple yet powerful tool for everyone to use. Type in a URL and before you can say "easy, global and secure," you're checking your email.

EMU Webmail is simplicity for system administrators.

EMUmail makes life simple for system administrators. A system administrator needs only to install EMU Webmail once for everyone on the network to use the program. When upgrades become available, one install is all that's needed for every user to enjoy the latest enhancements. No client-side software worries means fewer headaches for system administrators.

This document contains all the information you need to know in order to install, run, configure, and customize EMU Webmail.

This document is intended for mainly for system administrators and web designers wishing to install, maintain, or customize EMU Webmail. Users who check their email with EMU Webmail and have questions about their accounts should consult the online help pages, or their system administrator.

Chapter 2

Installing EMU Webmail

This chapter will guide you through the installation of EMU Webmail. Follow the instructions appropriate for your platform (Unix or Windows NT). Unix installs come in two flavors: with RPM and without. Installing with RPM is easier as most of the hard work is done for you. Some administrators may still favor Tarball installations as they offer the greatest flexibility.

After you have installed the system, you may wish to tweak the configuration according to your needs. Consult the chapter on Configuration for guidelines.

2.1 System Requirements

We recommand that your system meet the following requirements before you download and install EMU Webmail:

- 128 MB of RAM. 10 MB per instantaneous access with FastCGI (optional).
- 100 MB of available disk space, 2 MB per user for storage.
- Required for Unix only: Perl 5.005_03. (Test version with `perl -v`)
- Existing POP3 or IMAP mail server populated with email accounts.
- Existing SMTP server.
- WWW server capable of CGI.

- Internet connectivity.

2.2 Unix Install Instructions

2.2.1 Instructions for RPM install

This section contains instructions for installing EMU Webmail via the RPM package management system. These instructions apply to all architectures compatible the RPM format. Note that the installation steps must be executed as the administrator (root) user. You may wish to download a binary distribution or a source distribution, depending on availability. If an RPM is not an option on your operating system, try the tar package format instead.

The EMU Webmail Distribution will install the core EMU files, update or add some Perl modules inside the EMU directory. Two different RPMs are required to run EMU. One contains necessary Perl modules, while the other contains the core EMU cgi code.

The main EMU Webmail installation by default resides under:

`/home/emumail` With the webserver Document Root assumed to be at `/home/emumail/html`, and the EMU data directory at `/home/emumail/data`.

1. *Download the EMU Webmail Distribution from the Internet or Online Store.*

   EMU Webmail is available through the Internet at

   `<http://www.emumail.com/download.html>`

   Download the distribution to a temporary directory, and install the distribution files:

   ```
   rpm -ivh emumail-modules-4.5_NNN-N.i386.rpm
   ```

   ```
   rpm -ivh emumail-4.5_NNN-N.i386.rpm
   ```

Note that the NNN-N may vary depending on your architecture and the current version of EMU Webmail.

2. *Install License Keys.*

   If you have a licensed version of the EMU software, you will next need to install license keys. If you do not have a licensed version of the software, it will run in demo mode for a period of 30 days, after which the program will cease to function. Contact sales for a full demo key.

   To install a license key:

   ```
   cd /home/emumail/data
   ```

   Edit the site configuration file (`site.emu`), adding the license key.

   ```
   vi site.emu
   ```

   Add your license key to the file by copy and pasting into the `site.emu` file. A sample license key looks like:

   ```
   KEY__contact=user@emumail.com
   KEY__expire=970339698
   KEY__full_license=yes
   KEY__version=demo
   KEY__master=632e0e3f324f2b8239g55cf72cb5cb67
   ```

   Note that one license is required per CPU.

3. *Configure Web Server*

   Every web server configuration is different, so you may wish to consult your web server's manual for exact details of this step. In general you will need to setup CGI as a file type, provide the ability to access emumail.cgi from the web, and optionally setup fast cgi access.

Setting up .CGI as a file type is synonymous with "allowing emumail.cgi to be executed instead of downloaded." Some web servers require setting an execute permission, others require a mapping such that any files that end in `.cgi` will get passed to a cgi handler instead of being downloaded.

Under the Apache web server, this is done via the `AddHandler` configuration directive:

```
AddHandler cgi-script .cgi
```

to the Apache configuration file (httpd.conf). You should also set `Options +ExecCGI` in the configuration file under the scope of the directory in which the emumail.cgi file is installed. For example:

```
<Directory /home/emumail/html>
Options Indexes FollowSymLinks ExecCGI
AllowOverride None
</Directory>
```

Other web server will require different steps to install CGI capabilities.

4. *Verify Installation*

   After this is done you're ready to point a web browser to the EMU Webmail Installation . The EMU installation can be accessed at:

   ```
   http://[yourmachinename]/[webpathtoemu]/emumail.cgi
   ```

   If you get a login page for EMU Webmail, and the title bar in your window says "EMU Webmail - Login," then everything is successful and you can continue on to the Configuration section of this document.

   If you don't get a login page, or if the title of the login page contains an error, corrective steps will need to be taken, including re-running and double checking the steps in the installation process. See the Troubleshooting section of this document for more information.

2.2.2 Instructions for TarBall install

This section contains instructions for installing EMU Webmail on a Unix platform. These instructions apply to all flavors of Unix, including Linux, HP-UX, SunOS, Solaris, AIX, and others. Note that the installation steps must be executed as the administrator (root) user.

1. *Download and unpack the EMU Webmail Distribution.*

   EMU Webmail is available through the Internet at

   `<http://www.emumail.com.>`

   Download the distribution to a temporary directory, and unpack the distribution file:

   ```
   tar -xvf distfile.tar
   ```

2. *Run the install script.*

   Move to the top directory of the distribution and execute the install script:

   ```
   sh emuinstall.sh
   ```

   The script will ask you where you would like to install EMU Webmail. The default is `/home/emumail`, but you may choose a different location if you wish.

   For EMU Webmail to run properly, your system must have several supporting files. If files are missing or outdated, the install script will fetch and attempt to install the latest files for you.

   If you are asked for configuration information, typing RETURN to take the defaults should suffice for most questions. After the install script finishes you will be returned to the command prompt, ready for the next step.

3. *Create and populate the directory structure*

   The EMU Webmail distribution contains two main directories: `html/` and `data/`. The `html/` directory contains all the files that are accessible from the web server, including the main CGI file and all the images and help files. The `data/` directory is composed of user directories, configuration files, and HTML template data that should not be accessible from the web server. Both the `html/` and `data/` directories should be

readable and writeable by the user your web server is running as (often the user `nobody`–see a later section of this Chapter for information on finding this out).

The contents of the distribution's `html/` directory need to be copied to be in your web server's hierarchy. Determine where your web servers main directory is (the directory that contains the pages that first come up at `http://www.yourserver.com/`), then create a directory inside this directory:

```
mkdir /usr/local/apache/htdocs/emu
```

Note that you may pick another location as well provided that your web server is configured to serve files from this location. Next, copy the distributions `html/` directory to the web server's directory that you just created:

```
cp -R html/* /usr/local/apache/htdocs/emu
```

Now perform a similar set of steps for the data directory. First decide where to put the data directory. The data directory should have sufficient room for each of your users files. Make the directory by typing something similar to the following:

```
mkdir /emudata
```

Then copy the contents of the distribution `data/` directory to the newly created directory:

```
cp -R data/* /emudata
```

The final step to populating these directories is to setup the security permissions. The `html/` directory should have the same file permissions as your other files in your web server. After verifying this, set the file permissions on the `data/` directory to restrict access to only the web server user:

```
chown -R WEBSERVERUSER data
```

```
chmod -R 750 data
```

4. *Setup the init.emu file.*

   A file called `init.emu` unites the `data/` and `html/` portions of EMU Webmail by telling the CGI program where to find the data directory:

   ```
   page_root=/home/emumail/data
   ```

This line needs to be edited to reflect the absolute path of your `data/` directory. The `init.emu` file can be found in the `html/` directory.

5. *Testing the CGI program.*

   Now it is time to test the program. Move to the `html/` directory and execute the following command:

   ```
   ./emumail.cgi test
   ```

   If everything is working correctly, HTML should scroll by and fill up the screen.

   If you get a Perl error instead, you need to make sure that `emuinstall.sh` correctly installed all the modules (running `emuinstall.sh` again won't hurt), that the file permissions are correct, and that the `page_root=` line in the `init.emu` file properly points to the absolute path of your `data/` directory. After you have checked all of the above, see the Troubleshooting section of this document, visit the EMUmail web site at `http://www.emumail.com`, or email a support technician at `emu-tech@emumail.com`.

6. *View EMU Webmail from a web browser.*

   The final step is to view the program in action from a web browser.

   Delete the `/tmp/emuerror` file if it exists:

   ```
   rm /tmp/emuerror
   ```

   After this is done you're ready to point a web browser to the location you placed the `html/` directory. If you get a login page for EMU Webmail, and the title bar in your window says "EMU Webmail - Login," then everything is successful and you can continue on to the Configuration section of this document.

   If you don't get a login page, or if the title of the login page contains an error, corrective steps will need to be taken, including rerunning and double checking the steps in the installation process. See the Troubleshooting section of this document, visit the EMUmail web site at `http://www.emumail.com`, or e-mail a support technician at `emu-tech@emumail.com`.

2.3 Windows NT Install Instructions

To install EMU Webmail on Windows NT, first run the InstallShield wizard, which will guide you though most of the steps. After running the wizard, there are a few things that need to be done to complete your EMU Webmail installation:

1. EMU Webmail contains two main directories. One is `emumail` in your web server root. The other is `emudata` (used for storing users' mail). You need to set the permissions on both these directories to allow 'Network' access. Both directories need 'Change (RWXD)' access.

2. Setup your web server to execute CGI scripts. For Microsoft IIS 4.x do the following:

   (a) Open the Internet Service Manager.

   (b) Right click on the `emumail` folder and select 'Properties'

   (c) In the 'Directory' tab set the following:

   - Access Permission: check 'Read', uncheck 'Write'
   - Permissions: select 'Execute (including script)'
   - Content Control: uncheck 'Directory browsing allowed'

3. Configure the `site.emu` file in the `emudata` directory. The defaults should work for most installations. The most common settings that you may need to change would be:

   ```
   default_pop=pop.mydomain.com
   ```

   ```
   smtp_host=smtp.mydomain.com
   ```

4. You are now ready to run EMU Webmail. Load the the page in your browser using the URL `http://mydomain.com/emumail`.

2.4 Identifying the Web Server Account

The most accurate way to find out which account your web server is running under is to look at its configuration file (usually `httpd.conf`, for Apache, or through the Server Administration Page for Netscape servers). However,

a useful shortcut (though slightly more prone to error) is to look at who the web server is running under in your systems process list. Under Linux, FreeBSD, Digital UNIX, SunOS, and other BSD/OSF versions of Unix, type:

```
ps -aux | grep httpd
```

Under Solaris, IRIX, HP-UX, and other System V versions of Unix, type:

```
ps ef | grep http
```

In either case, you will get a listing of processes, similar to this (the first and last columns may give different names):

```
nobody 2487 0.0 1.7 2056 1076 ?  S May 10 0:00 httpd
nobody 2488 0.0 1.6 2036 1056 ?  S May 10 0:00 httpd
nobody 2489 0.0 1.4 2048  924 ?  S May 10 0:00 httpd
```

The left-hand column is what you want to look at. It may say `nobody`, or it may say something else. The name that you see in this column, regardless of what it is, is the name of the account that your web server is running as.

If, when you execute the `ps` command, nothing is returned, you can't use the shortcut; your web server is probably not named on a variation of "http". Youll have to dig up the configuration file to find out who your web servers user is. Usually Cobalt 4 runs as "apache", while most standard Apache installs run as "nobody".

2.5 Multiple Domains

Recall that the `init.emu` file tells the EMU Webmail engine where to find the `data/` directory. In fact, it is possible to use different `data/` directories when EMU Webmail is accessed through different URLs.

A file named `domain.init` in the `html/` directory, containing the line `page_root=dir` will tell the EMU Webmail engine to use the data directory `dir` when accessed through the URL `domain`. For example, if you wish to use the directory `data2/` specifically when the EMU Webmail engine is accessed through the URL `www.mysite2.com`, you should create a file named `www.mysite2.com.init` in the `html/` directory. This file should contain the

same thing as your `init.emu` file, except it should give the absolute path of your `data2/` directory.

You may create as many `domain.init` files as you wish. The `init.emu` file will serve as a default for domains not covered by a `domain.init` file.

Chapter 3

System Structure

This chapter will familiarize you with the structure of the EMU Webmail system. An installation of EMU Webmail includes the following directories:

As the diagram shows, the system is divided into two main directories: `data/` and `html/`. The `html/` directory contains all the files that are accessible from the web server, including the main CGI file and all the images and help files. The `data/` directory is composed of user directories, configuration files, and HTML template data that should not be accessible from the web server.

Below, we will take a look at a sample user home directory, located in the `emumail/data/homes` path. A user home directory contains the following files and subdirectories:

`DB_VERSION`	database version for this account
`LASTSESSION`	timestamp of last login
`files/`	files stored by the user
`folders/`	folder information database files
`folders-ordered/`	temporary files with folders with sorted contents
`foldmap`	database mapping messages to folders
`messages/`	raw mail messages, 1 file per message
`tmp/`	temporary session information
`userdb`	main database with option settings and addressbook

Chapter 4

Configuring EMU Webmail

There are many options that can be set in the EMU Webmail configuration files. Most users will need to change few, if any, of the configuration options–the defaults are meant to work in most cases. Some of the options are meaningful only in the licensed version of EMU Webmail. For more information on purchasing EMU Webmail, see our website at `http://emumail.com`, or contact `sales@emumail.com`.

4.1 Configuration File Overview

There are three main configuration files that control the operation of EMU Webmail: `site.emu`, `lang.emu`, and `conf.emu`. The files `site.emu`, and `lang.emu` are contained in your `data/` directory. A copy of `conf.emu` is contained in each interface directory. Additional copies of `lang.emu` may also be located in the interface directories.

- The `site.emu` file is a general configuration file. It is used to control system-wide options, including mail hosts, security, access restrictions, user options, disk quotas, and internal EMU Webmail settings. These settings affect all users.
- The `conf.emu` file focuses on interface-specific options. Since EMU Webmail supports multiple user interfaces, it is possible to set options on a per-interface basis, affecting only users using that particular interface. This is the purpose of the `conf.emu` file. Options such as background and link colors, maximum lengths for different elements,

and other interface-related options are in this file. Note: setting in `site.emu` and `conf.emu` can be interchanged, with `conf.emu` taking a higher precedence than `site.emu`.

- The `lang.emu` file contains the phrases that EMU Webmail uses to communicate error, warning, and other messages to the user that may need to be modified because of geographic, aesthetic, or linguistic needs. The `lang.emu` file can be present in each interface, allowing you to have different messages for each interfacefor example, one for the Spanish version and one for the English. The default `lang.emu` file is located at the same level as the `site.emu` file and will be the default for all interfaces. Per interface `lang.emu` files will override the default.

4.2 Configuration File Format

All configuration files are syntactically similar.

A line of the form `NAME=VAL` is used to assign the value `VAL` to the configuration option specified by `NAME`.

Each statement must occur on a single line. The order of the statements is not important.

The syntax of the configuration files is case-sensitive, so you will want to make sure that the cases of your options are correct. To define an option, at the beginning of the line type the name of the option, then an equals symbol, and then the value. It is not necessary to enclose the value in quotes. For example: `MSG_AddressEmpty = Empty Addressbook`.

Many options, such as `mail_hosts` and `allowed_domains` allow multiple values. You may separate multiple values by placing a space, or a comma, between values. Only one delimiting character, space or comma, may be present per option. Other options, such as `default_signature` and almost all entries in `lang.emu`, accept only a single value, but allow you to place spaces between words.

4.3 Special Values in Configuration Files

Some values take special meaning in these three files. The words `false` and `true` (which evaluate to 0 and 1, respectively) are Boolean values which can

be used to define options that take on only a true or false value. The words `no` and `yes` may be substituted for `false` and `true`.

The `\n` character may be used to enter a newline character into a value. For example, if the `default_signature` option was set to:

`default_signature=EMU\n4.0`,

the resulting signature would be:

```
EMU
4.0
```

You may set similar values in a configuration file by using `[variablename]`. `[variablename]` will get replaced by the value of `variablename`. For example if you set your `default_pop` to be `pop.mydomain.com` then any subsequent occurrence of `[default_pop]` will be replaced with `pop.mydomain.com`.

If an option is left with a blank value or is commented using the `#` character, then that options will be left unset, and at its default internal value. Thus, to have no default signature for any of your users, you would remove anything after the equals symbol: `default_signature=`.

4.4 Adding Custom Variables

If you are customizing your interface, you may sometimes find it useful to create variables whose values are set in `conf.emu`, that you can then access from your interface. To create a variable, simply create an entry for it in your interface's `conf.emu` file. For example, to create a variable called `foo`, and have its value be 1, add this line to `conf.emu`:

```
foo=1
```

You can then access this variable from an HTML template by writing `$foo`.

4.5 Precedence Among Files

Since EMU Webmail allows for multiple versions of the `lang.emu` and `conf.emu` files, there is an order of precedence for these values. Generally stated, interface specific configuration settings override system-wide configuration set-

tings. The `lang.emu` file located in an interface's directory contains settings with the highest precedence, and will override any site-wide `lang.emu` settings. Similarly, the `conf.emu` file read in from the current interface would override any settings that are found in the site-wide `site.emu` file. The `site.emu` file is always loaded and serves as a catchall if variables have not been redefined in other configuration files.

4.6 Licensing the Product

To license EMU Webmail so that you can customize the HTML, add your 5-line licensing key to the `site.emu` file. Licensing keys may be obtained by purchasing online at `http://www.intel.com/netstructure/store`. The licensed key may be placed anywhere in the `site.emu` file and looks like the following:

```
KEY__contact=user@emumail.com
KEY__expire=970339698
KEY__full_license=yes
KEY__version=demo
KEY__master=632e0e3f324f2b8239g55cf72cb5cb67
```

4.7 Restricting Access

Through parameters in the `site.emu` or `conf.emu` files, EMU Webmail allows you to control who accesses their email through your server, as well as where they access from. The simplest method of controlling access is to edit the `mail_hosts=` line in `site.emu`. Here, you can enter a space-delimited list of hostnames that you will allow users to check their email on. For example, if you wanted to allow only users with accounts on `pop.x.com` and on `pop.y.com`, your `mail_hosts=` line would look like this:

```
mail_hosts=pop.x.com pop.y.com
```

After defining the list of allowable servers, make `mail_host_input_box=` equal to `false`. This will prevent the "Hostname" box from appearing on the EMU Webmail Login page, and will instead present the user with a select box from which he or she may choose one of your pre-defined mail hosts.

A more sophisticated method of access control is to define a set of mail hosts from which your users may log on to EMU Webmail. This is controlled from the `allowed_domains=` line, where you can place a space-delimited list of domains that you will allow your users to connect from. For example, an `allowed_domains=` line that looks like this:

```
allowed_domains=x.com z.y.com
```

would allow logins only for users connecting from computers with hostnames ending in either `x.com` or `z.y.com`, e.g., `abc.x.com`, `tom.z.y.com`. This is useful in an environment such as an Intranet, where you might not want to allow connections from the outside world, but wish to provide access to users within your domain.

4.8 User Quotas

EMU Webmail allows you to set limits on disk usage by your users. Quotas may be enabled for all users, for a set of domains, or on an individual user basis. The `quota_default=` line allows you to set a default quota for all users, measured in bytes. A user whose account does not fall into any other category will have the default quota enforced. To enforce a quota for users with email accounts in a specific domain, you can create a domain-specific quota by adding a line to the `site.emu` file:

```
quota_@domainname = N
```

where `domainname` is the domain you wish to place a quota on, and `N` is the size of the quota, in bytes.

For example, to place a quota of 2 megabytes on all users whose e-mail accounts are in the domain `x.com`, e.g. `user1@x.com`, or `user2@mail.x.com`, you would enter this line in `site.emu`:

```
quota_@x.com = 2048000
```

To make individual exceptions to the system-wide and domain-wide quotas, use the users e-mail address to define their quota, similar to the domain-wide quotas. For example, to give `joe@x.com` a quota of 1 megabyte, add

this line to `site.emu`:

```
quota_joe@x.com = 1024000
```

Note: Assigning a quota of 0 will disable quotas for that class of users. For example, if you had a system-wide quota of 1 megabyte, but wish to give `jane@x.com` unlimited disk space, you would assign her a quota of 0:

```
quota_jane@x.com = 0
```

4.9 site.emu variables

The most common options that affect system-wide operation of EMU Webmail are found in `site.emu`, and are summarized here:

4.9.1 Common Options

`add_domain_to_user`

If set, the users e-mail address will be calculated to be `username@` this value. Use this when your pop server is not necessarily your e-mail hostname. So `user@pop.yourdomain.com` could get translated into `user@yourdomain.com`.

`allow_user_config`

If set to true, this will allow a conf.emu file to be read in from a users home directory. This allows the configuration to be tweaked down to a per user granularity.

`allow_user_lang`

Same as the allow_user_config, except with the lang.emu file.

`cache_emupages`

When using a persistent CGI like under mod_perl and FastCGI, cache the template files in memory to eliminate disk access.

`compose_dont_save_outgoing`

Set this to true to disable the saving of outgoing messages by default.

cookie_path

When running multiple copies of EMU Webmail from the same domain name, this will be the path that the cookies will be valid for. This option allows multiple CGI files to be run on the same server and have two concurrent EMU Webmail sessions open at the same time. This is an advanced option and is NOT the preferred method of running multiple interfaces. The default of /, should be sufficient for most installations.

daylight_savings

Set to true if daylight savings time is observed.

db_perms

Default permissions for the user databases.

dbm_isa

Specifies the Perl routine to use for the internal hashes.

default_autoload

Sets the default autoload of images in messages.

default_encoding

Default encoding of MIME messages parts.

default_filter1...N

Sets up default filtering options when new users login for the FIRST time. Format is:

default_imap_prefix

Some IMAP servers put a users folder hierarchy in a place other than the message root, such as mail/. This option allows the administrator to set this so their users dont have to.

default_mail_local

If true sets the default to download mail messages to the EMU Webmail 4 server and remove them from the POP/IMAP server.

default_outbox_host

When allowing users to save outgoing messages on an IMAP server, this option tells the EMU Webmail engine which host to save on, if it is different from the POP3 server.

default_pop

Use this option to specify the default POP3 or IMAP server that you will connect to when your users do not enter a hostname. The default is localhost, but you should change this to the correct hostname: default_pop=pop.yourdomain.com

default_send_host

You may set the default host to send mail to when mail is addressed only with a user name, and not a full email address. This is useful when you have a lot of users sending a lot of mail to different people on a single mail server. default_send_host=popular.x.com

dictionary

CGI to use as the dictionary for word lookups. The word to lookup is appended to the end of the URL.

disable_account_persistence

Similar to remote_only, however the entire home directory will be deleted on login and logout.

disable_caching

If not enabled [cache_headers] will be placed in all headers or Cache-Control: no-store, private if [cache_headers] doesnt exist.

disable_dictionary

An online dictionary can be used on the spell check page. This dictionary resides on a remote location and is queried by passing the unknown word to the end of GET HTTP request. To disable the dictionary support, set this value to true.

disable_forwarding

Support for writing a forwarding instruction for a users mail is available. Contact EMU Webmail technical support for advanced details.

disable_ipaddr_check

If set to false will mandate that each time during a single session that a user connects to the EMU Webmail server it connects using the same IP address. Causes problems if users come in through a proxy server or other means in which the IP addresses are masqueraded, however adds extra security on closed networks.

disable_msword

When a user receives an MS Word file, it is attempted to be converted to text. Setting this value to true can disable this behavior.

disable_nph

Forces the EMU Webmail engine to return only partial HTTP headers. This is useful for compatibility with FastCGI, and for renaming the emumail.cgi EMU Webmail engine.

disable_outbox

Disables users from saving outgoing messages.

disable_waitscreen

Optionally, when the EMU Webmail engine is delayed, it can present a temporary, animated wait screen. This behavior may be removed by setting this option to false.

do_realtime_filter

When set to true this will automatically filter messages as they come into the server. Set to false will force the user to press the filter messages button to trigger the filtering.

failed_login_counter

When set to true, EMU Webmail will keep track of the number of incorrect logins a user has had. This option normally is used in conjunction with the failed_login_max option.

failed_login_max

When the total number of incorrect logins a user has had exceeds this number, the user is presented with another URL, possibly directing them to further assistance or to a password retrieval page. The failed_login_counter option must be set to true in order for this option to work. Default is 6 attempts.

failed_login_url

If the failed_login_counter option is enabled, when a user attempts to login after exceeding the failed_login_max they will be taken to the page defined in this option.

force_mail_local

To force that all mail be downloaded from the POP/IMAP server and deleted from the mail spool, set this value to true.

force_protocol

To force the EMU Webmail engine to use a particular login protocol, set this to the protocol name.

forward_file

The path to the forward file, if it exists.

emu_debug

Setting this option to true forces EMU Webmail to create a log file in the data/ directory or to the directory that debug_path points, named emudebug. This file contains an entry for most internal actions EMU Webmail performs during a session. Event logging is very useful for investigating problems with EMU Webmail. Since emubebug can grow very quickly, it is recommended that you leave debugging off when you do no absolutely need it not using it.

keepdomains

When logging into the POP server if the domain name is one of these domains, it will be entered to the POP server as part of the user name. This is useful for POP servers that handle virtual hosts.

lax_mailbox_sync

Under IMAP dont synchronize when the mailboxes differ.

login_protocol

Allows you to select which protocols are used by default for logging a user in, and in which order the connections are attempted. The allowed protocols are pop3 and imap. The protocols are used in the order they appear in the options line, and are delimited with a space.

map2pop_DOMAINNAME

Will map a DOMAIN name to a POP server so the users dont have to remember it.

max_time

Maximum time in seconds that a session can remain inactive and not expire.

outbox_protocol

Allows you to select which method to use to save users outgoing messages, and in which order the methods are attempted. The allowed methods are imap and local. An IMAP outbox allows your

users to store their outgoing mail on another server. A local outbox stores outgoing messages in users EMU Webmail home directories.

perlsub_user_home

Perl subroutine to calculate where a users home directory should be. Must return a valid path. This option is useful for placing users home directories on various partitions or remote mount points based on some administrator defined load balancing function. The e-mail address will be given to the sub as a parameter.

publisher_name

The publisher_name is the string that is printed in the title bar of the Web browser. The value of this option may be changed to anything you like. By default it is set to EMUmail.

quota_type

Setting this value to emumail will allow the EMU Webmail administrator to program their own routine in the Custom.pm module of the data/lib/EMU directory in the quota_check subroutine. Other values are ignored. This is useful for implementations where the quota is defined by an outside interface. The quota_check routine should return (in : delimited fashion) the amount allowed, amount used, and a percentage of usage. EMU Webmail will compare to the used (in bytes) to the quota_allowed variable to determine if the user is over quota.

redirect_login

Set this to a URL where the user will be taken when they logout. Useful to switch to https:// for login. Can add perlsub_ to the beginning to make it into a Perl subroutine that returns the URL.

redirect_logout

Set this to a URL where the user will be taken when they logout.

remote_only

Setting this to true will disallow the creation of long term storage on the EMU Webmail server. When a user logs in and logs out their home directories will be deleted. Use this option if you wish to provide occasional use of the webmail services, but dont wish to have long term storage. The users options and personal settings will be retained on the EMU Webmail server, but the mail messages and folders wont be.

smtp_host

If your outgoing mail server is different from your POP3 server, enter the SMTP servers hostname here.

success_login_sub

Allows the administrator to execute a subroutine in the Custom.pm module whenever a user successfully logs in. This will allow the administrator to set their own options, do advanced reporting, or whatever they like. The value of this variable corresponds to an actual Perl subroutine in the Custom.pm, and returns no value.

sysfile_dir

When users attach files, they can also attach files from this system-wide directory. This gives you the option of specifying system and company wide logos, documents, and shared information to all your webmail users.

sysfile_dir2

A second sysfile directory can be specified using this value.

timezone

Designates your time zone. Use the standard abbreviation for your local time zone.

umask

Default umask for created files.

use_fastcgi

Setting to true will enable the FastCGI hooks to speed up program response. Must have a FastCGI compatible web server.

use_modperl

Setting to true will enable mod_perl hooks to speed up program response. Must have a mod_perl compatible web server like Apache.

xadvert

Sets an X-Advert: field in all outbound messages.

4.9.2 User Quotas

User Quotas EMU Webmail allows you to set limits on disk usage by your users. Quotas may be enabled for all users, for a set of domains, or on an individual user basis. The quota_default= line allows you to set a default quota for all users, measured in bytes. A user whose account does not fall into any other category will have the default quota enforced. To enforce a quota for users with e-mail accounts in a specific domain, you can create a domain-specific quota by adding a line to the site.emu file: quota_@domainname = N where domainname is the domain you wish to place a quota on, and N is the size of the quota, in bytes. For example, to place a quota of 2 megabytes on all users whose e-mail accounts are in the domain x.com, e.g. user1@x.com, or user2@mail.x.com, you would enter this line in site.emu: quota_@x.com = 2048000. To make individual exceptions to the system-wide and domain-wide quotas, use the users e-mail address to define their quota, similar to the domain-wide quotas. For example, to give joe@x.com a quota of 1 megabyte, add this line to site.emu: quota_joe@x.com = 1024000 Note: Assigning a quota of 0 will disable quotas for that class of users. For example, if you had a system-wide quota of 1 megabyte, but wish to give jane@x.com unlimited disk space, you would assign her a quota of 0: quota_jane@x.com = 0

4.9.3 User Defaults

The site.emu file allows you to set the defaults for many user options. Users may override these defaults, unless you have customized the interface to prevent users from changing the default options. The configurable options are listed here:

`default_email`

Allows you to form the outgoing mail address using the different parts of the users login address. Using a sample e-mail address, joe@mail.x.com, %u represents joe, %h represents mail.x.com, %1 represents com, %2 represents x, and %3 represents mail. The standard form for an e-mail address is %u@%h, which will result in the users complete e-mail address. A common form is to have %u@%2.%1 (dont forget the period!), which would result in our sample e-mail address above looking like this on outgoing mail: joe@x.com.

`default_signature`

Allows you to select a default signature to be appended to your users outgoing mail. Not to be confused with the MSG_Tagline option in lang.emu, which may not be overridden; the

default_signature may be overridden by the user. If you have multiple interfaces, you may, at your option, delete this line from site.emu and place it in the conf.emu file for each of your interfaces, thereby allowing you different signatures for different interfaces.

`default_real_name`

Allows you to set the default Full Name for new users.

`default_organization`

Allows you to set the default organization for new users. Useful within Intranets for placing your company name on outgoing mail.

`default_checkmail`

Sets the default frequency, in seconds, that EMU Webmail checks a users POP3 or IMAP accounts for new mail while the user is at the Index page.

`default_compose`

The Compose page has a number of header lines that can be hidden or shown as the user wishes. This sets the header lines that are available initially for the user to fill out. Acceptable values for this options include: attach, cc, bcc, from, replyto, and priority. The To header is always available. To have just the Cc line available by default on the Compse page: default_compose=cc or, compose_display_OPTION=true.

`default_address1...N`

Default addressbook entries for new users. The order of an entry is the nickname, then two vertical bars (——), then the full name, then two more vertical bars, and then the e-mail address. Starting with the number 1, you can have as many default entries as you desire, each on its own line. default_address1=joe——Joe Smith——joe@x.com default_address2=jane——Jane Doe——jane@x.com

4.9.4 Basic Interface Configuration

Each directory that contains a user interface (data/iface/normal, for instance), may contain a file named conf.emu, which allows different interfaces to specify different settings for certain options.

`addressbook_length`

The maximum number of characters to print in the Addressbook pull-down menus on the Compose page.

default_folder1...N

Allows you to specify folders that are automatically created when a user logs into EMU Webmail for the first time. For example, to have every new user have a folder, Company Mail, place this entry in conf.emu: default_folder1=Company Mail.

default_max_messages

The maximum number of messages to show in a single page. If the user exceeds this number of messages in the current folder, a link to a next page of messages appears in the Index page. If your users receive a large number of messages frequently, it might be advisable to raise the value of this option a bit so that users have to switch pages less often.

folder_namelen

The maximum allowable length for a new folders name. Folder names over this length will be truncated.

index_subject_length

The maximum number of characters to print for the messages subject in the Index page.

index_sender_length

The maximum number of characters to print for the senders address in the Index page.

post_login_page

If this line is entered into the conf.emu file, the file specified by this line will be loaded as an intermediary. This is useful for creating a frame-based interface for EMU Webmail.

post_login_page=framebase.html status0...2

These three variables allow you to specify what is printed for the three possible states of a message. Status0 is an unread message; Status1 is a read message; and Status2 is a message thats been replied to. HTML may be placed here. This is useful for placing small images, such as checkmarks, next to messages.

Chapter 5

Customizing the Interface

EMU Webmail allows you to customize the user interface by modifying the HTML template files, the interface configuration file (`conf.emu`), and the language definition file, (`lang.emu`). Template files are written in a superset of the HTML markup language, containing standard HTML code, plus specialized commands called EMUcode. Additionally, you may script templates using Embedded Perl for advanced presentation options.

Template files are found under EMU's data directory, inside the iface direictory.

```
/home/emumail/data/iface/normal
```

.

To customize EMU Webmail, you will need to have a licensed "professional" version of the software. Licenses are installed in the `site.emu` file and will allow all the customizations described in this document.

5.1 Adding a New Interface

EMU Webmail allows multiple user interfaces to be defined and used by one `emumail.cgi` file. All interfaces must be installed as subdirectories of the `/emumail/data/iface/` directory.

The default user interface name is `normal`. Changing the `site.emu` pa-

rameter `default_interface=` will change the default interface:

```
default_interface=french
```

To allow access to additional interfaces, edit the `site.emu` file, adding interfaces to the `ifaces=` line, as well as setting `multi_interfaces=true` to denote that multiple interfaces are in use.

```
multi_interfaces=true
ifaces=normal french
```

Note that there is a site-wide `lang.emu` file in the `data/` directory that contains default values for the interface's language (error) messages. These may be overridden by interface-specific `lang.emu` files in the appropriate `iface/` directory.

5.2 Modifying the User Interface

5.2.1 Introduction

EMU Webmail can be customized in one of two ways, either by using the powerful Embedded Perl, or the easier to learn EMU code. The default templates contain Embedded Perl.

5.2.2 Embperl

Embperl (embedded Perl) is a powerful scripting language that allows you to execute Perl code from within HTML templates.

There are three main ways that Embperl can be called:

1. *[- ... -] Execute code*

   ```
   [- $a = 5 -] [- $b = 6 if ($a == 5) -]
   ```

   The code between the [- and the -] is executed. No output will be generated. This is mainly for assignments, function calls, database queries, etc.

2. *[+ ... +]Output the result*

```
[+ $a +] [+ $array[$b] +] [+ "A is $a" +]
```

   The code between the [+ and the +] is executed and the return value (the value of the last expression evaluated) is output (sent to the browser).

3. *[! ... !] Execute code once*

```
[!  sub foo { my ($a, $b) = _ ; $a * $b + 7} !]
```

   Same as [- ... -], but the code is only executed for the first request. This is mainly for function definitions and one-time initialization.

Meta Commands

Embperl support some meta commands to control the "program flow" within the Embperl document. This can be compared to preprocessor commands in C. The meta commands take the following form:

```
[$ <cmd> <arg> $]
```

if, elsif, else, endif

The if command is just the same as in Perl. It is used to conditionally output/process parts of the document.

Example:

```
[$ if $EMU::protocol eq 'imap' $]
Using IMAP
[$ elsif $EMU::protocol eq 'POP' $]
Using POP
[$ else $]
Not POP or IMAP, must be Local?
[$ endif $]
```

foreach, endforeach

Create a loop iterating over every element of an array/list.
Example:

[$ foreach $msg (1..10) $]
[+ $msg +]
[$ endforeach $]

Embperl Resources

More information on Embperl can be found at:
`http://perl.apache.org/embperl/`

5.2.3 Guide for Template Background Colors

Templates can be controlled from a master configuration file to allow for themes to be developed and put into play without drastic HTML changes. In conf.emu you will see the key below for controlling the background colors for the "normal" interface. These themes are implemented only in the "normal" interface currently, located in the emudata/iface directory.

Match the described color to the screenshot's colors to determine which area to modify. Example, in the screenshot, the DARK GRAY color matches the MENU_BGCOLOR property. To change the color of the MENU_BGCOLOR, enter a valid hexcode for its value.

Key for screenshot:

```
DARK GRAY = MENU_BGCOLOR
LIGHT GRAY = BODY_BGCOLOR
RED = BODY_TOP_BGCOLOR
GREEN = BODY_BOT_BGCOLOR
LIGHT BLUE = BODY_HEADER_BGCOLOR
YELLOW = BODY_GRID_BGCOLOR
DARK BLUE = BODY_GRID_OUTLINE
```

Example of usage in conf.emu:

MENU_BGCOLOR = 006699

Another property you can change via conf.emu is the page background, margins, and even a background image. Do this via the PAGE_BODY property. An example of its usage is below:

```
PAGE_BODY = bgcolor="#FFFFFF" leftmargin="10"
```

Other parameters can be setup in a similar fashion to create themes that are tied to a conf.emu but which share a common set of HTML templates.

5.2.4 EMUcode

EMUcode is embedded within standard HTML code to produce a template file, which, when passed through the EMU Webmail engine, allows basic loops, conditionals, and variable substitutions. EMUcode provides the tools by which you may modify the user interface. EMUcode can be decomposed into two main types: variables and statements.

Variables are substituted with the values that are given by the EMU Webmail engine. Variables are represented in the EMUcode by a "`$`" followed by the variable name, for example, "`$myvariable`". Variables may derive their values in one of three ways:

- Through entries in the configuration files (`site.emu` or `conf.emu`).
- Through data gathered by the EMU Webmail engine (for example, an email message body).
- Through WWW form element names from the previous page submission.

The syntax of EMUcode statements has been designed so it stands out away from the surrounding HTML code. Statements are tagged with the EMUcode double-brace notation, `({ function() })`. A "`({`" starts the statement, and a "`})`" ends it, with values in the middle.

There is an important limitation in that an EMUcode statement, `({ statement })`, must not contain line wraps in the template files! That is, every `({ statement })` must be on ONE line. This limitation makes parsing the template files easier for the EMU Webmail engine.

`({ })` statements may be of three types: functions, loops, and conditionals. There are a fixed number of pre-defined functions, such as `({ print_addrbook() })` which access a certain data element from the EMU Webmail engine. Loop statements like `({ for address in @addrs })` are provided for further flexibility and to keep the code compact. Conditional statements, like `({ if ($have_results) })` are used to access HTML on an as needed basis, depending on the state of the program and data.

5.2.5 Pre-defined Functions

Pre-defined functions are pieces of EMUcode that get substituted with a dynamically generated piece of HTML code. Examples of pre-defined functions include listing address book entries, building CGI links, and printing a list of postponed messages. These functions may appear on any page and may be used accordingly.

EMU Code Pre-defined Functions:

({ attachments() })	Displays list of attachments.
({ get_status() })	Displays status (error) messages from the last action.
({ iface_select() })	Displays a list of interfaces for selecting. Needs to be wrapped around a <SELECT> statement.
({ linkpage(SECTION, HTML) })	Links to the given section of EMU Webmail. SECTION = { index, compose, address, options, folders, help, logout }
({ print_addrbook() })	Displays an <OPTION> list of address book entries. Needs to be wrapped around a <SELECT> statement.
({ print_emufiles(N) })	Displays an <OPTION> list of EMUfile entries. Needs to be wrapped around a <SELECT> statement. N can be 0 or 1. N = 1 means print draft messages as well.
({ print_folders() })	Displays an <OPTION> list of folder entries. Needs to be wrapped around a <SELECT> statement.
({ print_message() })	Gets message body text from postponed message, forwarded message, etc.
({ print_nextpage_options() })	Displays Jump box. Needs to be wrapped around a <SELECT> statement.
({ print_postponed() })	Displays postponed (draft) message list in select box.
({ print_priority() })	Displays message priority <OPTION> list. Needs to be wrapped around a <SELECT> statement.
({ showindex(SORTTYPE, HTML) })	Creates a link around HTML that will set the sort key of an index. SORTTYPE = { date, select, sender, size, subject }
({ top_image() })	Prints out a banner ad.

5.2.6 Conditional and Loop Statements

Conditional and loop statements are used in conjunction with pre-defined functions and variables to add power to EMUcode. The two basic statements of this type that EMUcode supports are the conditional, `({ if () })` and the loop statement, `({ for counter in @array })`. These two statements allow repetitive or branching functionality within your template file.

The IF statement

The `({ if () })` statement comes in two varieties: the simple and compound. Simple if statements will be print the left hand side of the statement if the right hand side is true:

```
({ (stuff to print) if (condition true) })
```

Compound `({ if })` statements are similar to simple `({ if })` statements, however they may span multiple lines:

```
({ if (condition true) })
Print this here.<P>
This can be multiple lines, and even contain embedded EMUcode.
({ endif })
```

Note that in compound `({ if })` statements the left hand side is embedded between the start and end of the statement. This `({ if })` ... `({ endif })` combination allows for more complex HTML to be generated while not getting in the way of the multi-line restriction of EMUcode.

The condition part of the `({ if })` loop may be of the following forms:

English	Operator	Example	True When
Equality	`==`	`($variable == 0)`	Value of `$variable` is `0`
Inequality	`!=`	`($variable != fred)`	Value of `$variable` isn't `fred`
Exists		`($variable)`	Value of `$variable` is defined

The FOR loop

Unlike the `({ if })` statement, the `({ for })`, `({ for value in @values })`, statement comes in only one form. Like the compound `({ if })` state-

ment though the ({ for }) loop begins with a start token ({ for value in @values }), contains code to be displayed for each iteration, and is finalized by a ({ done }) token.

```
({ for value in @values })
<B> $value </B>
({ done })
```

Note that in this example the internal loop variable "value" is accessed through $value in the loop. Any word may be substituted for "value".

5.2.7 Variables

As mentioned above, variables get substituted for something else when passed through the EMU Webmail engine. Variables allow a generic EMUcode template to be customized to a user's preferences or data, adding elements such as a user's message, name, or folder data. Variables are all denoted with a "$" followed by a variable name. For example: $variable is a variable whose name is variable. A less confusing example would be the variable $fish whose name is fish, who is written as $fish, and whose value may be something such as COD, CARP, or TROUT. EMU Webmail will interpret your variable names into their corresponding values. Instead of $fish showing up to the user, COD would be displayed.

The following is a list of variables recognized by EMU Webmail. You may add your own to form macro substitutions by adding them to the site.emu, conf.emu, or lang.emu files. Alternatively you may pass hidden form elements that will be interpreted as $form_variable_name on the NEXT page. Note that not all variables are accessible from every page and the last page you were on dictates the accessibility of some variables-some variables have smaller scope than others. Some variables may actually be a list of values, and are accessed as an array of values: $variable[array element], where array element is an integer.

The following list is organized by pages from the default interface that these variables show up in. If a variable repeats itself in further pages it is not duplicated. By looking up the list you can find the definition of a variable that exists in multiple pages.

address.html:

$addr	email address for addresses page. Used with a loop.
$addrs	Array of addresses d$count Used on address page to keep track of addresses. D1 is anchored to the first address, and if the value is TRUE is deleted when submitted to address_parse.
e$count	email for address COUNT
f$count	Fullname for address COUNT
$full	Fullname for addresses page
$have_results	Results from the LDAP search on the address page
$img_url	URL to the image files for this interface
$ldap_search_results	Data returned from LDAP search
n$count	Nickname for address COUNT
$nick	Nickname for addresses page
$publisher_name	The name given to the webmail engine. Defaults to EMUmail
$searched_for_name	Name searched for in addressbook
$searched_for_org	Organization searched for in addressbook
$url	The URL to the CGI, e.g., http://.../emumail.cgi

errors.html:

$FONT_ERROR	Font HTML to use for error codes
$logintext	Helpful login hints from login screen
$phrase	Error message

compose.html:

$attach.show	Show attachments box on Compose screen
$bcc	Blind Carbon Copy Value (from postponed, continued, spell check)
$bcc.show	1 if user has selected to show the BCC box
$cc	Carbon Copy Value
$cc.show	1 if user has selected to show the CC box
$ds	Don't Save outgoing mail. 1 if you're not supposed to save
$dsn	Delivery Status Notification. 1 if DSN is selected
$from	Value of From input box (from postponed, continued, spell check)
$from.show	Show From input box on compose screen
$here_atts	List of files currently attached to message
$msg_continued	Internal variable to point to a message unique ID (spell check, postpone)
$organization	Organization of sender
$org.show	Show organization input box
$priority.show	Show priority select box
$replyto	Value for Reply To input box
$replyto.show	Show Reply To input box if true
$sender	The Sender of the message
$sender.show	Show Sender input box if true
$subj	Value of the Subject input box for message
$to	Value of the To input box for message

login.html:

$error_login	Login error messages, helpful hints
$multi_interfaces	Used to select the interfaces names available
$user_name	Username derived from failed login, or cookies between sessions

folder.html:

`$answered_msgs`	Number of messages in the selected folder that have been answered
`$folder_selected`	The currently selected folder name
`$new_foldername`	The folder to be created
`$total_msgs`	Count of message in currently selected folder
`$unread_msgs`	Unread messages in currently selected folder

lookup.html:

`$bad_word`	The current, possibly incorrect, word being spell checked
`$def`	Possible definition text from remote dictionary
`$digest`	Internal message identification number
`$wordnum`	Current word number in spell checker

msgindex.html:

`$folder`	Current folder name
`$messages`	Total messages
`$moremsgs`	True if index spans multi pages
`$msg`	Current message number in index loop
`$myfold`	Alias for current folder name
`$next_page`	Next Page Number
`$reload`	Seconds to reload index page
`$tmp`	Next URL to load when refreshing
`$total_pages`	Maximum number of pages

msgview.html:

$address_add	URL to add current address to address book
$attachments	True if there are attachments
$basic_header	True if basic headers selected
$date	Value of date of current message
$full_header	True if full headers selected
$fullname	Full name of sender of message
$message	Internal message ID
$prologue	Inline HTML, attachements, etc.
$show_html	True if show_html option is selected - doesn't format < into <
$the_message	Body of message

options.html:

$autoload	True if autoloading inline HTML is on
$checkmail	Value for seconds to reload Index page
$disable_forwarding	True if forwarding feature to be disabled
$email	Current users e-mail address
$forward	Value for forwarding address
$max_messages	Maximum number of messages per page on index
$organization	Value for organization
$quota_used	Amount of quota user has used
$quota_allowed	Maximum quota for user
$real_name	Value of user's real full name
$real_time_spam	True if realtime spam (RBL) detection is to be used
$signature	Value for user's signature file

wordview.html:

$attachurl	URL to retrieve unaltered attachment
$word_body	Interpreted MS word document as HTML

5.3 Form Elements

EMUcode gets interpreted by the EMU Webmail engine, allowing dynamic elements and data to be put into normal template files. The user converses with the EMU Webmail engine through a series of HTML form submissions. The form elements are contained within the HTML tags `<a href>`, `<input>`, `<select>`, `<textarea>`, as well as through direct manipulation through the URL. Any form element may be put into the URL of inside an `<a href=>`. Example: `<a href="$url?passed=folder_parse&folder=myvalue">`.

The following is a list of form elements broken down a page at a time.

msgindex.html:

NAME	VALUE	DESC
d	$messages[hash@msg]	checkbox
delete.x	bool	img/submit
filter.x	bool	img/submit
folder		Select
index_jump.x	Bool	Img/submit
move.x	Bool	img/submit
narrow		
passed	select	hidden
position		select
refresh.x	bool	img/submit
reload.x	bool	img/submit
search.x	bool	img/submit

compose.html:

NAME	VALUE	DESC
attach.x	BOOL	submit/img
attached	$here_atts	hidden
bcc	$bcc	
bcc_s	UID	
bc	$cc	
cc_s	UID	Select
continue.x	BOOL	submit/img
from	$from	
message		textarea
msg_continued	$msg_continued	hidden
organization	$organization	
options	MULTIselect	MULTIselect
	attach	MULTIselect
	bcc	MULTIselect
	cc	MULTIselect
	from	MULTIselect
	org	MULTIselect
	priority	MULTIselect
	replyto	MULTIselect
outbox	$ds	
passed	compose_parse	hidden
postpone.x	BOOL	submit/img
priority		
properties.x	BOOL	submit/img
replyto	$replyto	
returnreceipt	$dsn	checkbox
selected_msg	UID	hidden
send.x	BOOL	submit/img
sender	$sender	
spelling.x	BOOL	submit/img
subject	$subj	
tmpupload		file upload
to	$to	

options.html:

NAME	VALUE	DESC
action$count		select
autoload	{0}|{1}	
bRegex$count	{0,1}	checkbox
checkmail	$checkmail	
data$count		select
delete.x	Bool	Submit/IMG
forward	$forward	
mailloc	{0|1}	radio/hidden
max_messages	$max-messages	
modifier$count		select
organization	$organization	
passed	options_parse	Hidden
prefix	$prefix	
protocol	{Imap|pop3|}	
real_time_spam	$real_time_spam	checkbox
reset.x	Bool	img/submit
save.x	Bool	img/submit
selected_file		Select
signature	$signature	textarea
total	$countersave	hidden
type$count		select
upload.x	Bool	File
uploaded_file	file contents	File
view.x	Bool	submit/img
yourmail	$email	Submit/Img
yourname	$real_name	

lookup.html:

NAME	VALUE	DESC
b	1	hidden
lookat	$bad_word	
lookup.x	BOOL	img/submit
passed	spelling_parse	hidden
selected_msg	$digest	hidden
spelling.x	BOOL	img/submit
wordnum	$wordnum	hidden

spelling.html:

NAME	VALUE	DESC
continue.x	Bool	Img/Submit
end_pos	$end_pos	Hidden
lookup.x	Bool	Submit/Img
passed	spelling_parse	Hidden
replace.x	Bool	Submit/Img
replacement	$bad_word	
replacements	(replacement list)	Select
selected_msg	$digest	Hidden
skip.x	Bool	Submit/Img
spelling.x	$wordnumb	Hidden
start_pos	$start_pos	Hidden
word_num	$word_num	Hidden

login.html:

NAME	VALUE	DESC
c	encoded pw	hidden/URL
first	1	hidden
next	nextsubonauth	hidden
passed	login_parse	hidden
password		password
type		interface type
user_name	$user_name	

address.html:

NAME	VALUE	DESC
d$count		checkbox
e$count	$addrs[n$count]	input
email	$addr	input
f$count	$addrs[n$count]	input
full	$full	input
n$count	$addrs[n$count]	input
nick	$nick	input
passed	address_passed	hidden
search.x		submit/img
search_name	$searched_for_name	input
search_org	$searched_for_org	input

folder.html:

NAME	VALUE	DESC
delete.x	Bool	submit/img
edit.x	Bool	submit/img
export.x	Bool	submit/img
folder_selected		
fold_fold		
fold_host		
fold_pass		
fold_type	{imap,local,pop}	hidden/radio
fold_user		
new_fold	$new_foldername	
passed	folders_parse	hidden
save.x	Bool	submit/img

msgview.html:

NAME	VALUE	DESC
add2folder	(foldername)	select
addto.x	Bool	Submit/img
delete.x	Bool	Submit/img
display	{showhtml, full_header, basic_header, vrml, print}	
display.x		submit/img
forward.x	Bool	Submit/img
passed	multi	Hidden
quick.x	Bool	Submit/img
rdate	$rdate	Hidden
reply_how	{reply, qreply, replyall, qreplyall}	Select
reply_x	Bool	Submit/img
variable	$message	Hidden
variable2		Select

5.4 Plugins

EMU Webmail has full support for programmers to create their own plug-ins. Plug-ins are Perl subroutines that are made accessible to the EMU Webmail engine; they are called when a URL submitted to the engine specifies that the plug-in should be used.

Every plug-in performs one main function: it supplies values for variables to the EMU Webmail engine. The EMU Webmail engine in turn performs substitutions on an EMUcode template which uses those variables. Of course, your plug-in may perform any number of functions, but the essential exchange between a plug-in and the EMU Webmail engine is in the passing of those values.

Plug-ins are placed in the `Custom.pm` file, which is in the `data/lib/EMU/` directory. This is the only location where you may place your plug-ins. When accessing the plug-in, EMU Webmail will look here to find the subroutine.

EMU Webmail defines its current state through the value of the `passed=` parameter in the URL. Standard states include `compose`, `compse_parse`, `address`, `address_parse`, `options`, etc. The special value, `passed=parse`, is reserved for accessing custom pages. When specifying the `parse` parameter, use the `variable=` parameter to specify which EMUcode template to load as the next page.

The `passed=parse` and `variable=filename` parameters may be used to tell the EMU Webmail engine to load any page present in the interface's directory. However, the variables that a custom page has access to is limited; any variables that are specific to any particular section of EMU Webmail will not be available to your custom page.

When entering the `parse` state, an additional parameter becomes available: `c_sub=`, whose value defines a custom subroutine–a plug-in-for the EMU Webmail engine to execute. This is where your plug-in will be called from. When creating custom pages, it will often be necessary to submit values for custom variables. To do this, simply create form elements with unique names. When the custom form is submitted, the values will automatically be placed in the URL. Your plug-in may then access these variables by referencing their names.

Here's an example of what a URL to an EMU Webmail plug-in might look like in an EMUcode template:

```
$url?folder=$folder&passed=parse&variable=newaddr.html&c_sub=
newaddress
```

(Please note that there should be no spaces or line breakes in the URL.)

The `$url` variable will be substituted for the URL of your EMUmail installation. The `folder=$folder` name/value pair must be present in any EMU Webmail internal URL; this n/v pair maintains the currently open mail folder for this session.

The `passed=parse` and `variable=newaddr.html` pair tell the EMU Webmail engine to load the file `data/iface/INTERFACENAME/newaddr.html`. Finally, the `c_sub=newaddress` pair tells the EMU Webmail engine to run the `newaddress` subroutine in `Custom.pm` before loading `newaddr.html`, and to substitute values for EMUcode variables in `newaddr.html` with values supplied by `newaddress`.

To construct the above URL from an HTML form in an EMUcode template, a form such as the following should be used:

```
<FORM METHOD="POST" ACTION="$url">
<INPUT TYPE="HIDDEN" NAME="FOLDER" VALUE="$folder">
<INPUT TYPE="HIDDEN" NAME="PASSED" VALUE="parse">
<INPUT TYPE="HIDDEN" NAME="VARIABLE" VALUE="newaddr.html">
<INPUT TYPE="HIDDEN" NAME="C_SUB" VALUE="newaddress">
...
</FORM>
```

Here is a simple plug-in that demonstrates the variable reading and writing mechanism. Assume that the form above has a line like this:

```
...
<INPUT TYPE="TEXT" NAME="EMAIL_NAME">
...
```

This would be a very simple plug in:

```
sub myaddress {
package EMU;
$name = $query->param('email_name');
write_tmp("myaddr1", $name."emumail.com", 1);
}
```

All this plug-in does is to read the value of the `email_name` param-

eter given to it by the page that loaded the plug-in, append the string `@emumail.com`, and then write that string to the variable `myaddr1`. The result is that any occurence of `$myaddr1` in the `newaddr.html` EMUcode template will be replaced with the value that was submitted in the `email_name` text box from the previous page.

There is virtually no limit to the number of extensions to EMU Webmail using the plug-in interface. Connecting to a backend database, integrating into existing applications, adding team-friendly features, and more, are all possible through EMU Webmail plug-in interface.

Chapter 6

Troubleshooting FAQ

This chapter contains some Frequently Asked Questions about troubleshooting EMU Webmail. The EMU Webmail FAQ is constantly being updated and expanded. The most current and complete version of the FAQ, in searchable form, is available at our web page: `http://www.emumail.com`.

Why do I receive an error "email address invalid" when I send mail?

Your mailserver is probably not accepting mail relay from the webserver (even though they may be the same machine). In order to open the relay to allow mail to be sent offsite, you need to consult your mailserver documentation for instructions. Be sure to open relay to only the webserver's IP address, and not to the whole Internet.

How can I set EMU Webmail to use another SMTP port?

By default, EMU Webmail will use 25 for SMTP. If you need to change the port number, remove the `#` at the beginning of the `#smtp_port=1025` line in your `site.emu`file and set it to the appropriate port.

What format does EMU Webmail keep its address books in?

The addressbook files are in GDBM database format, in a simple hash table. The addressbook addresses are stored alongside other data in the `userdb` file. A hash table entry looks like this: the key is

addresses.NICKNAME and the value is EMAIL:FULLNAME where NICKNAME, EMAIL, and FULLNAME are replaced by appropriate data. The hash table can be viewed by running the showgdbm tool available for download from http://emumail.com/support.html.

How do I set up CGI in Apache?

Add, in access.conf or httpd.conf (which are probably in /etc/httpd/conf/):

```
Options Indexes FollowSymLinks ExecCGI
AllowOverride None
```

Of course, change

/PATH/TO/EMUMAIL.CGI

to the appropriate directory path.

In srm.conf or httpd.conf, add or uncomment:

```
AddHandler cgi-script .cgi
```

On the first click after I log in, the server returns a "Session Expired" error.

It's likely that your web browser is not accepting cookies. To fix this:

In Netscape, go to the Edit menu: Edit -> Preferences -> Advanced: Cookies: "Accept all cookies"

In Internet Explorer, go to the Tools menu: Tools -> Internet Properties -> Security -> Custom Level -> Cookies -> Allow Per-session cookies -> "Enable"

How do I set up a frames-based interface for EMU Webmail?

In order to implement a frames-based interface after logging, you'll have to set a field in your conf.emu file (which is in the directory where your interface is located, e.g., data/iface/normal/conf.emu). It should look like this (note that we use frames.html as a generic filename–feel free to choose your own):

```
post_login_page=frames.html
```

In this manner, your file, frames.html, can set up a frames-based interface.

If you want to use alternate filenames, you can use the "parse" mode of EMU Webmail to access the templates directly. For example, in the default `msgindex.html` file, you can access the `compose.html` page by placing the following link in the page:

```
<a href="$url?folder=$folder&passed=compose">
```

Alternately, you can use the Parse mode to access the page directly:

```
<a href="$url folder=$folder&passed=parse&variable=compose.html">
```

Either method accomplishes the same thing. Note that you declare the name of the file that you wish to access through the `variable=` name/value pair.

To configure EMU Webmail for multiple interfaces, in `site.emu`, set

```
multi_interfaces=false
```

```
ifaces =  iface1 iface2 iface3
```

Place the relevant HTML files you create in `iface/iface1`, `iface/iface2`, etc.

I get an access violation right after installation.

If you have Activeperl installed, it may be causing a conflict. Try moving that directory to `c:\perl2` or something similar.

How do I configure EMU Webmail to use IMAP only?

In `site.emu`file set:

```
pure_imap=true
```

```
login_protocol=imap
```

```
outbox_protocol=imap
```

I got an "Improper login sequence" error.

It's likely that your web browser does not have Javascript enabled.

- For Netscape: `Edit -> Preferences -> Advanced:  Enable Java & Enable Javascript`
- For Internet Explorer: `Tools -> Internet Options -> Security -> Custom Level -> Scripting -> Enable`

How do I get emails to show the correct address in the From

In your `site.emu`file, find the line that says `default_email = something`. In that string, the codes are:

- `%u` = username (before @).
- `%h` = full host (e.g. `server.treverton.co.za`)
- `%1 ... %N` = reverse order domain, so if you had `mail.emumail.com`, `%1` would be `com`, `%2` would be `emumail`, and `%3` would be `mail`

If EMU is running on `mail.yourdomain.com`, and you want messages to appear to be coming from `yourdomain.com`, you might want to try something like:

```
default_email = %u@%2.%1
```

or

```
default_email = %u@yourdomain.com
```

How do I restrict what servers users can get mail from in EMU Webmail?

For EMUmail 3.x or Webmail 4.x, edit your `site.emu`, and set:

```
default_pop=popservername.com
```

```
allowed_domains=popservername.com
```

```
hostname_input_box=false
```

and edit the html template to remove the box.

How do I send mail to multiple people?

You can send mail to a group of people by placing a comma ',' between the addresses in either the To:, CC:, or BCC: fields. You can setup a group of people in your addressbook using this same method.

Everything looks to be set up, but the browser returns "document contains no data" and web server error logs say: "CGI error: QUIET / Compilation aborted"

Try deleting the `/tmp/emu-error` file, because it may have permissions which disallow write access by the webserver user. For example, if this

file is generated by root, then web browser tries to append it, it cannot since this file is owned by root. Note that this bug has been fixed in EMU Webmail 4.x.

How do I stop EMU Webmail from showing my name and mailhost everytime I log in?

This can be prevented by modifying the `login.html` template file so as not to retrieve this information from the cookie. Delete the references to `$username` and `$hostname` from the `login.html` file.

How can I send special MIME-types (like HTML) in my messages?

You can allow users to set the MIME-type of their message by adding an INPUT element to the compose page template. You can make all messages HTML messages HTML messages by adding this tag:

```
<input type=hidden name="message_ct" value="text/html">
```

Or you could put a drop down select box with multiple ones:

```
<select name="message_ct">
<option>text/html<option>text/plain</select>
```

or you can create a open text field:

```
<input type=text name="message_ct" value="text/plain">
```

We recommended *not* making text/html the default, as this might confuse users who use the caret (arrow) symbols in their messages.

Chapter 7

Additional Support

If you have technical question regarding EMU Webmail that is not answered in this document, please consult the searchable FAQ on our web page: `http://www.emumail.com`.

If you are unable to find the answer to your question on our web page, you may email `emu-tech@emumail.com` for further information. When contacting us with a problem, please include a detailed description of the problem you are having, including any error messages that EMU Webmail returns.