The Environment Maintenance Tool Installation Guide

Computing Services
Carnegie Mellon University
February 25, 1993
Edition 0.0

DRAFT

Table of Contents

About this Guide

This guide is a reference for the installation of the Environment Maintenance Tool (EMT) software. It will identify hardware and software components necessary to install EMT, explain how to create EMT servers and clients, and give an overview of the structure used with the Andrew System at Carnegie Mellon University.

These conventions are followed in the *EMT Installation Guide*:

- **Bold** is used for new terms or for emphasis.
- *Italics* are used for titles.
- `Typewriter` is used for text that appears on the screen.
- **`Bold Typewriter`** is used for text or commands that you type.
- Text inside angle brackets **< >** describes a variable that you type as part of a command.
- This font is used for pathnames to a specific directory or file.
- A bulleted list shows a set of items or choices.

CHAPTER 1 *EMT Installation*

1.1 Overview

EMT was written to simply the process of maintaining and releasing applications in the AFS environment. The task of software management has traditionally been a time consuming one. The goal of EMT is to decrease mundane and tedious work by automating the process of software management and standardizing software collections.

1.2 Installation Requirements

In order to install EMT, there are several resources the installer must have:

- A server machine for EMT to run on. The server should have at least <?> MB of free hard drive space for the EMT daemon.
- Root access on the server.
- AFS privileges. You will need access to the servtab to install EMT. <?>
- Client machines. <Flesh out>

1.3 What the Installer Should Know

In addition to hardware and access requirements, there are many things the installer should know before beginning to install EMT.

- Understand your current software management system.
- Be proficient with AFS, particularly with mount points, volumes and the commands vos and pts.
- Be familiar with what EMT can do for you. For more information, read *Software Management in the Andrew System*, by Colyer, Held, Markley and Wong. This technical paper is available via anonymous FTP from **`export.acs.cmu.edu`**. The file is /pub/SoftMgt/AFSUG-Fall-92.ps.Z.

1.4 Other Software to be Installed

There are several pieces of software that are required in order to build EMT. Before installing EMT, you must have the following installed:

- AFS libraries (common error and command line).
- Compilation tools (e.g., make, build, or smake).
- **ADM**. ADM is available via anonymous FTP from **`export.acs.cmu.edu`**. Files are in /pub/adm.

In addition, there are programs that add functionality to EMT:

- **Site**, which supplies EMT with server/partition pairs when EMT needs to know where to create, replicate or unreplicate volumes. Site provides an automated generation of the correct locations.
- **Depot**, which links released software collections based on the name of the application. Depot is available via anonymous FTP from **`export.acs.cmu.edu`**. Files are in /pub/depot.

<Should depot actually be required?>

- **Oplog**, which sends information from EMT logs to a bboard, email address, or a zephyr instance.

1.5 Creating a Directory Structure for EMT

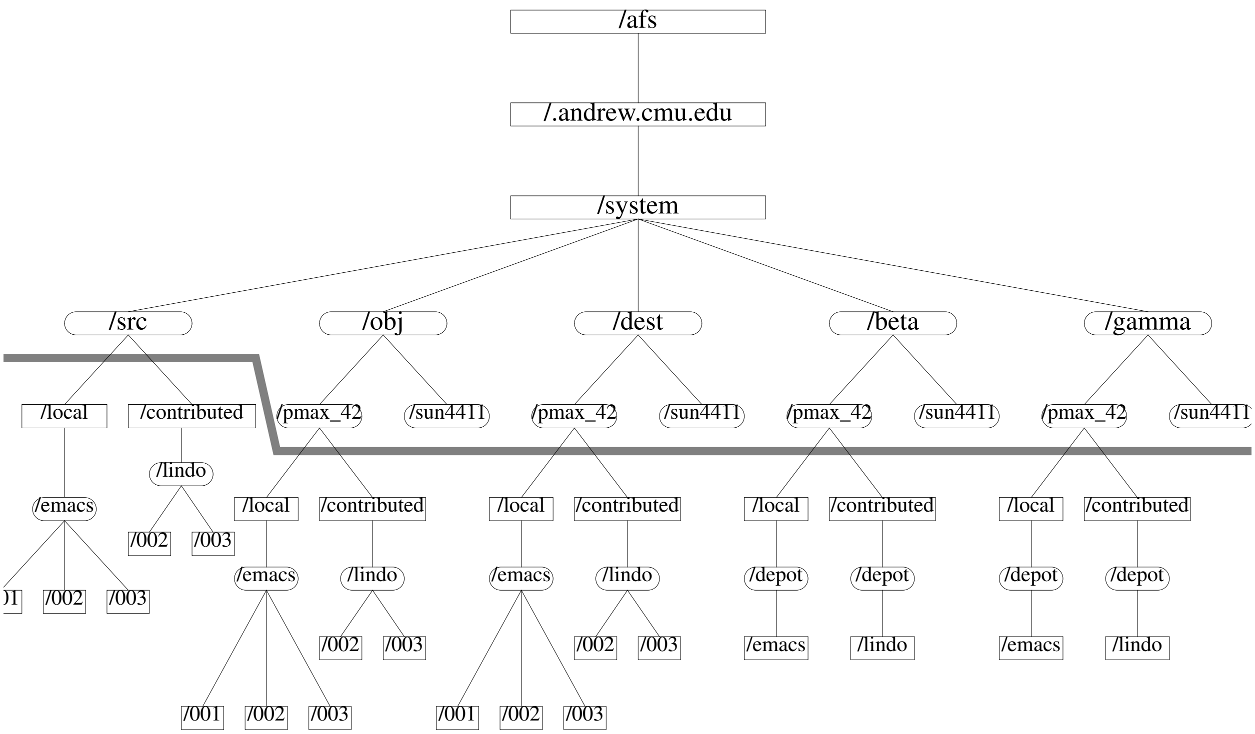
Before compiling EMT, you will need to establish a software management structure at your site to conform to a structure EMT will be able to manage. EMT makes software management easier by imposing a standard structure. Before using EMT, you will have to create the various areas it requires "by hand."

1.6 Sample Directory Structure for EMT

The illustration below shows an example of how the structure of your software management area might look. Items that are in boxes are mountpoints for a volume. Items that are in ovals are simply directories.

EMT needs to know the exact path to each of the volumes that appear just below the thick gray line. These paths are defined in **emtd.config**. Once these areas are created and defined, EMT can then maintain the area below the gray line. You can define the area above the gray line in any way you choose. This is merely an example.

Please note that this illustration is not complete. For example, the same structure should appear under the boxes with "/sun4411" in them as appears under the boxes with "/pmax_42" in them. "emacs" and "lindo" are only examples of possible collection names, and more versions than those shown can be mounted in each collection's area.



1.6.1 Rational for the Structure

In order to keep the large amount of software on the Andrew File Servers organized and easily accessible, Computing Services enforces a software management structure.

- Each application has its own **collection**. The collection name must be unique, and this name will be associated with the software permanently. For example, *gnu-emacs* and *batmail* are collection names. The collection name must have 9 characters or fewer.
- Each application's collection contains **versions** of that software. Each version is given a three number designation.The first version is 001, the second is 002, and so forth.

Each software collection is part of a **tree**. There are currently three trees supported at Carnegie Mellon: *local*, *contributed*, and *host*. The type of software you are developing and who it is for will determine the tree your software will be a part of. *Local* contains software developed or supported by Computing Services. *Contributed* contains software developed and supported by users, and *Host* contains software which is used only by systems or machines

Within each tree, difference **system** types may be supported. Currently, two system types are supported on Andrew, pmax_42 and sun4411.

When software is released, it is released to an **environment**. There are two environments at Carnegie Mellon — **gamma** and **beta**.

- *Gamma* is the normal Andrew environment that is used on cluster machines and most personal workstations.
- *Beta* is the environment used by Computing Services employees to test new software before the campus community can access it.

1.6.2 Development Areas

The areas set aside for software development are in the form of **volumes**. Volumes are self-contained areas of disk space. Once created, a volume is **mounted** as a directory in the Andrew File System. To users, volumes look like any other directory. However, volumes can be mounted from more than one path name, and volumes have a set *quota* of storage space. An example of a volume is your home directory.

•>> The Source Area

The **source** area is where you place the sources for your software.

The mountpoint for a source volume is:

/afs/andrew/system/src/<tree>/<collection>/<version>

For example:

/afs/andrew/system/src/local/gnu-emacs/001

•>> The Dest Area

The **dest** area is where **destination volumes** are kept. These are also referred to as **release volumes**. The mountpoint for a destination volume is:

/afs/andrew/system/dest/<system>/<tree>/<collection>/<version>

For example:

/afs/andrew/system/dest/pmax_42/local/gnu-emacs/001

If you have large files that are shared between the system types you support, you may create a common volume. This is a single volume that contains files for your software that are not system dependent — for example, help files. This volume appears as the mountpoint common within each of your release volumes.

•>> The Object Area

The **object** area is where the objects and executables are built when you compile software. By default, this area will be on your local disk, in

/usr/obj/<tree>/<collection>/<version>

For example:

/usr/obj/local/gnu-emacs/001

The mountpoint for object volumes on AFS is:

/afs/andrew/system/obj/<system>/<tree>/<collection>/<version>

For example:

/afs/andrew/system/obj/pmax_42/local/gnu-emacs/001

1.7 Compiling EMT

You will need working binaries for the server and client of EMT. This will require modifying a few files contained in the sources you have. After you have completed these modifications, you may use **make, build**, or **smake** to compile the EMT sources.

1.7.1 Modify rxemt.xg

Contained in the rx subdirectory, this file contains the defines that determine tell EMT a great deal about its working environment.

•>> EMTD_RPC_PORT

This defines the port that the EMT server will bind to. Use the default, or pick a port that no other service on the host machine will use.

•>> EMTD_RPC_SERVICE

This defines the service number that the EMT server will use in binding its socket.

•>> EMTD_RPC_NAME

This defines the name of the service that EMT will use in binding its socket.

•>> EMTD_WORK_NAME

This is the name of the user that EMT will authenticate as when doing all of its work. This should be an account that has been specifically created for EMT. A real user should not be using this account. If you are not using MIT kerberos, you will need to set the key for the user and place this in the KeyFile within the CONF directory.

•>> EMTD_WORK_INSTANCE

This is the instance of the user that EMT will authenticate as. There is no need for EMT to use any special instance (such as an admin instance), so you may simply leave this as an empty string.

•>> EMTD_WORK_CELL

This is the cell that the account EMT will be using resides in. This should be set to your local cell name. You may set this to be NULL as well.

•>> EMTD_WORK_ID

This is the ID of the account that EMT will be using.

•>> EMTD_SERVER_CELL

This is the name of the cell that the EMT server will be working in. Set this to be the name of your cell. This should not be set to NULL.

•>> EMTD_WORKING_DIRECTORY

This is the path to the directory from which EMT will read its configuration files, and under which it will write its logs.

•>> EMTD_CONF_PATH_DEFAULT

This is the default configuration path. It is where EMT will look for its KeyFile and other standard AFS configuration files. If you are using MIT kerberos, you can set this to be an empty string.

•>> EMTD_DB_DIRECTORY

This is the path to the database directory that EMT uses. This should point to a database volume for EMT that is mounted somewhere in AFS.

•>> EMTD_SRVR_DUMP_PATH

This is the path to a directory containing nightly dumps from the file servers. These dumps are done through a local modification to AFS at CMU. Since this is only used in the **vclean** command, it is unlikely that you will need to set this to anything other than an empty string. If you would like more information about this, please contact EMT support at CMU (markley+@cmu.edu).

•>> EMTD_SCRIPTS_DIRECTORY

This is the path to the area containing the scripts used by EMT in its execution. These scripts came with the EMT source, and are initially located in the **scripts** subdirectory. It doesn't matter whether you place these on the local disk or on AFS as long as the path is correctly set here.

•>> EMTD_RSH_PATH

This is the path to the **rsh** executable. If you are not using MIT kerberos or don't plan to use **rsh**, you can set this to be an empty string.

•>> EMTD_OPLOG_COMMAND

This is the command line invoked for logging operations that EMT has performed. At CMU, we use an application named **oplog**. EMT execs this command line, and writes to it through its stdin. You may use any application to log the completion of EMT commands in this way. You could set this to be a dummy shell script, or simply set it to send mail to someone or some bboard.

•>> EMTD_DEFAULT_ADM_SERVER

This is the name of the default ADM server host machine. Set this to be the name of the machine that the ADM server is running on.

•>> EMTD_DEFAULT_ADM_PORT

This is the default name of the port that the ADM server will be listening to.

•>> **EMTD_DEFAULT_HOST**

This is the name of the default EMT server host machine. Set this to be the name of the machine that the EMT server will be running on.

•>> **EMTD_MAX_MONITORS**

Monitors are the connections to EMT that are watching a single command's execution. This is the maximum number of monitors that EMT will allow on each command. Each monitor only takes about 8 bytes, but 10 monitors seems to be more than enough.

1.7.2 Modify the top-level Makefile

If all goes well, you will only have to modify the top-level Makefile. This Makefile contains several variable definitions that determine the functionality of the EMT server and client.

•>> **GNU Readline**

Allowing USE_GNURL to be defined will enable the client to take advantage of the gnu-readline functionality.

•>> **MIT Kerberos**

If you do not wish to use MIT kerberos, you must comment out the definitions of both USE_MITKERB and KRBLIBS.

•>> **Remote Shell (RSH)**

If you are using MIT kerberos, and wish to take advantage of **rsh** through EMT, you must allow USE_RSH to be defined. If you don't want this feature comment it out.

•>> **Operation Logging**

If you want EMT to log the operations it performs through the utility you defined with EMTD_OPLOG_COMMAND in rxemt.xg, you must allow USE_OPLOG to be defined. Comment it out if you only want things loggedto a file on the local disk.

•>> **Timestamps**

If you plan to use timestamps on links in your depot areas under your software management scheme, you must allow USE_TIMESTAMPS to be defined. Comment it out if you don't want to use timestamps.

•>> **Destination Directory**

DESTDIR is the path that will be used when installing EMT. You should set it to point to the volume you wish to install EMT into. This volume should be mounted in the destination area initially, according to your software management directory structure.

•>> **Root Directory**

ROOTDIR must be set to the root path of the area under which common include and library directories will be found. At CMU, this is /usr/local/.

•>> Library Directories

LIBDIRS must be set to point to the directories that your libraries will be found in.

•>> auth Library

LIBAUTH must be set to indicate the correct "auth" library for your site. If your are running AFS 3.2, this should be libauth.krb.a.

•>> ADM Library

LIBADM must be set to the path for the ADM library at your site.

1.7.3 Make the executables

After having edited the necessary files and setting things the way that your site requires them, you should simply need to run make in the top level of the EMT source area. If you run into errors along the way, you may have to check the way you have modified the files. If it seems to be a more general problem, you may wish to contact EMT support at CMU (markley+@cmu.edu).

1.8 Configuration Files for EMT

In order for EMT to maintain your software, it requires knowledge about your software management structure. EMT reads this information from the file named **emtd.config**, which must be placed in the directory you indicated by setting EMTD_WORKING_DIRECTORY (in rxemt.xg). This file describes the structure of your software management in terms of the trees, environments, and system types you will be using. A sample file is contained in the **local** subdirectory of the EMT source area and in Appendix 1. This describes the structure at CMU, which you might use as an example of how to structure your site.

The emtd.config file has a very simple format. Each line starts with a key word followed by a colon. There ordering of these key words is important in defining your structure. The following is a list of the key words, and the information that should be on the lines that follow them.

1.8.1 Tree Description

tree: <tree name>

The largest overall structure in our software management scheme is called a tree. To start the definition of a tree, you must first have a line with the **tree** key word, followed by the name of the tree. Everything following this line in the configuration file will be read as part of this tree, until another **tree** key word is encountered.

1.8.2 Source Path

src_path: <source path>

EMT needs to know where the source area is for the tree. You must place the path to the top level of your source area after this key word.

1.8.3 Database Path

db_path: <database path>
EMT must have an area that where it can store and update information. You must place the path to the database area you have created for EMT after this keyword.

1.8.4 Group Ownership

owner: <owning pts group>
You should place the name of the **pts** group that should own all other **pts** groups created for this tree after this key word.

1.8.5 Environment Description

environment: <environment name>
This is the next largest structure of software management after the tree. You must place the name of the environment you are defining after this keyword. Every line that follows this key word will be associated with the environment named, until another environment or tree key word is reached. Note that there are two special environments that must be defined for every tree: **obj** and **dest**. These environments are defined in the same way as all other environments, but they have special meaning to EMT. They specify the object and destination areas that EMT will use for the current tree in all its operations on that tree.

1.8.6 System Description

system: <system name>,<path>,<depot type>,<replicated>,<unrep days>,<expire days>
This key word describes the smallest part of our software management: the system. Information contained all the lines following this key word will be associated with the system being defined, until another system, environment, or tree key word is reached.

<system name>
Should be filled with the name of the system being defined.

<path>
Should be filled with the path to the top level for this system, under the current tree and environment. When working with the object and destination environments, this will be the path to the working areas for these environments. For all other environments, this should be the path to the depot area, without "depot" on the end.

<depot type>
This is used to indicate which version of depot is to be run. This is only really significant at CMU. You should always set this to be non-zero.

<replicated>
Should be set to indicate whether this system should have collections replicated, when they are released to this environment and tree. Setting this to be non-zero will indicate that collections should be replicated. This has no meaning within the object and destination environments.

<unrep days>
Should be set to indicate the number of days that will be allowed to pass before old, unreleased replicated volumes will be unreplicated (in order to save disk

space).

<expire days>
Should be set to indicate the number of days that will be allowed to pass before old mountpoints in the depot area will be removed. These old mountpoints result when EMT is using the timestamp routines.

1.8.7 Replicated Volumes

volume_rep: <volume name>
This key word indicates the name of a volume that should be released when changes are made to the current system's area. Any number of lines with this key word can come after a system key word. This has no meaning for systems within the destination or object environments.

1.8.8 Hosts used by rsh

rsh_host: <host name>
This defines the host that should be used when performing an **rsh** operation for some function operating on the current system. Any number of these key word lines can be listed after a system key word. The host names will be stored with the system in the order they are supplied. If **rsh** fails to connect with a host, it will try the next host on the list. This has no meaning in the object and destination environments, or if you do not plan to use **rsh**.

1.9 Tree Configuration Files

In order to allow for different configurations for each tree, EMT reads a configuration file for each tree. This file is named **config.<tree name>**, and must be placed in the same directory as **emtd.config** (see previous section). There is an example config.local found in Appendix 2.

1.9.1 Limiting Quota

quota.<collection name>: <maximum quota>
Should you decide to allow collection maintainers to perform certain commands on their collection, this is how you can limit the amount of quota they can have for each volume in the collection. If they want more than this amount, they will have to contact the Gatekeeper. Setting <maximum quota> to be zero will give the maintainer access to unlimited quota for their collection. The <collection name> is the name of the collection which this quota restriction will affect. If "*" is used as the collection name, or no collection name is given, the maximum quota specified will be the default for every collection within the tree that is not explicitly defined.

1.9.2 Limiting the number of Versions

versions: <maximum versions>
To limit the number of versions that can exist for a collection at any one time, you must set <maximum versions>. This is useful in ensuring that maintainers delete old versions before allocating space for new versions. If necessary, the Gatekeeper can override this by using the "-force" commands on creates in the EMT client.

1.9.3 Limiting Command Access

auth.<command name>: <security level>+

As mentioned earlier, it is possible to allow the maintainer to perform certain functions on their collection. Through the auth keyword, the level of security that can call individual commands can be specified.

<command name>

This must be the command name for which authorization is being set. If "*" is used as the command name, or no command name is specified, the security levels listed become the default for any command not otherwise explicitly defined. The current command names that the EMT server recognizes are: create, delete, release, remove, update, setquota, access, migrate, and expunge.

<security level>+

There are three levels of security: global, gatekeeper, and maintainer. The global security level is authorized to perform any command, so it should not be listed. The other levels should be listed, separated by whitespace.

1.9.4 Limiting the Times Commands are allowed

time.<command name>.<environment name>: <restricted time>+

This keyword allows the software manager to restrict the time that commands can be run under certain environments. This is useful if large releases of volumes cause a significant and noticeable slowdown of access to the file servers.

<command name>

This must be the command for which a time restriction is being set. If "*" is used as the command name, the time restriction becomes the default for all commands within the environment that are not explicitly defined for that environment. The current command names that the EMT server recognizes are: create, delete, release, remove, update, setquota, access, migrate, and expunge.

<environment name>

This must be the environment in which the time restriction for the command is being set. If "*" is used as the environment name, the time restriction becomes the default for the command in any environment where it is not explicitly set. If both the command and environment names are set to "*" or not present, the time restriction becomes the default for any command in any environment for which time restrictions have not been explicitly set.

1.9.5 <restricted time>+

This must be set to indicate the restricted time(s) for the command and environment specified. The format is as follows:

(<day>+, <start time>-<end time>)

`<day>+` is the number of the day for which the restriction should apply. If it should occur on more than one day, the number for each day should appear separated by whitespace. Zero is Sunday, etc.

`<start time>` is the time the restriction should begin. It should be the hour and minute of the 24 hour clock separated by a colon.

`<end time>` is the time the restriction should end. It should be the hour and minute of the 24 hour clock separated by a colon.

More than one of these restrictions can be listed after the time keyword. They should be placed on the same line, separated by whitespace.

APPENDIX 1 *Example emtd.config file*

```
tree: local
src_path:/afs/andrew.cmu.edu/system/src/local
db_path:/afs/.andrew.cmu.edu/data/db/emt
owner: sysmaint:localboss
environment: obj
system: pmax_42,/afs/andrew.cmu.edu/system/obj/pmax_42/local,1,0,0,0
system: sun4411,/afs/andrew.cmu.edu/system/obj/sun4411/local,7,0,0,0
environment: dest
system: pmax_42,/afs/andrew.cmu.edu/system/dest/pmax_42/local,1,0,0,0
system: sun4411,/afs/andrew.cmu.edu/system/dest/sun4411/local,1,0,0,0
environment: beta
system: pmax_42,/afs/.andrew.cmu.edu/system/beta/pmax_42/local,1,0,10,30
volume_rep: pmax_42.blocal.011
system: sun4411,/afs/.andrew.cmu.edu/system/beta/sun4411/local,1,0,10,30
environment: gamma
system: pmax_42,/afs/.andrew.cmu.edu/system/gamma/pmax_42/
local,1,1,10,30
volume_rep: pmax_42.local.011
rsh_host: pmax.wash.acs.cmu.edu
```

system: sun4411,/afs/.andrew.cmu.edu/system/gamma/sun4411/local,1,1,10,30

volume_rep: sun4411.local.011

rsh_host: sun4c.wash.acs.cmu.edu

tree: contributed

src_path: /afs/andrew.cmu.edu/system/src/contributed

db_path: /afs/.andrew.cmu.edu/data/db/emt

owner: sysmaint:contriboss

environment: obj

system: pmax_42,/afs/andrew.cmu.edu/system/obj/pmax_42/contributed,1,0,0,0

system: sun4411,/afs/andrew.cmu.edu/system/obj/sun4411/contributed,1,0,0,0

environment: dest

system: pmax_42,/afs/andrew.cmu.edu/system/dest/pmax_42/contributed,1,0,0,0

system: sun4411,/afs/andrew.cmu.edu/system/dest/sun4411/contributed,1,0,0,0

environment: gamma

system: pmax_42,/afs/.andrew.cmu.edu/system/gamma/pmax_42/contributed,1,1,10,30

volume_rep: pmax_42.cont.012

rsh_host: pmax.wash.acs.cmu.edu

system: sun4411,/afs/.andrew.cmu.edu/system/gamma/sun4411/contributed,1,1,10,30

volume_rep: sun4411.cont.012

rsh_host: sun4c.wash.acs.cmu.edu

APPENDIX 2 *Example config.local file*

```
quota.atk-atk: 80000
quota.gnu-emacs: 80000
quota.tsdb: 5000
quota.emt: 0
quota: 20000
versions: 4
auth.create: gatekeeper maintainer
auth.setquota: gatekeeper maintainer
auth.access: gatekeeper
auth: gatekeeper
time.release.gamma: (1 2 3 4 5, 10:00-17:00)
time.update.gamma: (1 2 3 4 5, 10:00-17:00) (0, 2:00-6:00)
```