

## 1 Introducing ange-ftp.

Ange-ftp is a system for transparent file-transfer between remote UNIX, VMS, CMS or MTS hosts using FTP. This means that you can edit, copy and otherwise manipulate files on any machine you have access to from within GNU Emacs as if it were a local file. Ange-ftp works by introducing an extended filename syntax, and overloading functions such as **insert-file-contents** so that accessing a remote file causes appropriate commands to be sent to an FTP process. Ange-ftp works with Dired (and in particular Sebastian Kremer's Tree Dired) to facilitate directory browsing and multiple file transfer from remote hosts.

The author of ange-ftp is Andy (Ange) Norman ([ange@hplb.hpl.hp.com](mailto:ange@hplb.hpl.hp.com)).

Finally, this info file was written by Dave Smith ([dsmith@stats.adelaide.edu.au](mailto:dsmith@stats.adelaide.edu.au)), although large chunks of it are plagiarised straight out of the extensive comments section of `ange-ftp.el`.

## 2 Installing ange-ftp

If you don't already have a copy of ange-ftp, or you want a later version, ange-ftp is pretty easy to get hold of. FTP is the probably the simplest method, but other options such as mail are available.

Once you have the Emacs-Lisp source, there are a few customisations you might need to make. The ideal configuration is to have the FTP process running on the same machine as you are running Emacs on, but this is not always possible since some machines cannot access hosts outside the local network. In this case, the FTP process needs to be run on a machine which *does* have access to the local world — this is called the **gateway host**. Ange-ftp has facilities to make use of a gateway host when accessing remote hosts.

### 2.1 How to get the ange-ftp source code

The latest version of ange-ftp should always be available for anonymous FTP from

`alpha.gnu.ai.mit.edu`

in the file

`ange-ftp/ange-ftp.tar.Z`

(which includes both `ange-ftp.el` and this texinfo file.) In ange-ftp notation, that's

`/anonymous@alpha.gnu.ai.mit.edu:ange-ftp/ange-ftp.tar.Z`

Alternatively, ange-ftp is also part of the Emacs-Lisp Archive on `archive.cis.ohio-state.edu`. The latest version should always be available on this site, but the Lisp-Code Directory entry is not always up to date; it currently reads:

```
ange-ftp (3.112)  91-08-12
  Andy Norman, <ange@hplb.hpl.hp.com>
  archive.cis.ohio-state.edu:
    /pub/gnu/emacs/elisp-archive/as-is/ange-ftp.el.Z
  transparent FTP Support for GNU Emacs
```

Ange-ftp can also be found at:

`ugle.unit.no:/pub/gnu/emacs-lisp/ange-ftp.el.Z`

Failing these, someone on the ange-ftp mailing list (See Chapter 4 [Getting help], page 16) or the author himself (See Chapter 1 [What is ange-ftp?], page 1) may be able to help you find the latest version.

If you intend to do a lot of browsing through archive sites it is definitely worth your while installing Sebastian Kremer's Tree Dired along with ange-ftp (if you haven't done it already). Tree Dired will work with ange-ftp without any modifications: all you need to do is follow the installation instructions that come with the package. The Tree Dired package comes complete with the latest version of ange-ftp, and is available for anonymous FTP from the following sites:

```
ftp.thp.Uni-Koeln.DE:/pub/gnu/emacs/diredall.tar.Z  (134.95.64.1)
ftp.cs.buffalo.edu:pub/Emacs/diredall.tar.Z
```

Alternatively, you can get in touch with Sebastian himself using his e-mail address: `sk@thp.Uni-Koeln.DE`.

## 2.2 Installing the source

Installation is simply a matter of copying the file `ange-ftp.el` to a directory in your load-path. If you don't already have a load-path, this is probably a good time to make one. Just create a directory (say, `~/elisp`) in which you plan to keep your Emacs-Lisp files. Then place the following line in your `.emacs`:

```
(setq load-path (cons (expand-file-name "~/elisp") load-path))
```

The `expand-file-name` is *important* — omitting it is a common reason why load-paths do not work.

Once you've copied `ange-ftp.el` to the appropriate directory, it is recommended to byte-compile it, with *M-x byte-compile-file*. Then place the line

```
(require 'ange-ftp)
```

in your `.emacs` (*after* the line which modifies your load-path, of course!) It's that simple.

The above instructions should allow you to access all hosts that your local machine can access. If your local host has limited access, however, you may wish to have ange-ftp working through a gateway machine. If so, read on. Otherwise, See Chapter 3 [Using ange-ftp], page 9, to get started using ange-ftp.

## 2.3 Using a gateway

Suppose you are running Emacs (and ange-ftp, of course) on a machine X (let's call it the 'local host') and you want to access a file on a machine Z (which we will call the 'remote host'). Unfortunately, X does not have FTP access to Z: when you try a manual FTP something like the following happens:

```
X$ ftp Z.foo.bar.com
ftp: connect: Host is unreachable
```

However, X *does* have access to a machine Y (the 'gateway machine') which *can* access Z. Fortunately, you have an account on the gateway machine, and so the solution is to login to Y, ftp to Z, download the file you want from Z to Y, and then copy it from Y to the local host, X. This can get a bit tedious, to say the least, but fortunately ange-ftp can do all the hard work for you.

Firstly, you need to set the variable `ange-ftp-gateway-host` to the name of the gateway machine. The name should be the one that the local host recognises, that is, the name you use with `login` so that it works.

```
(setq ange-ftp-gateway-host "Y.local.lan.edu")
```

Since you only need to go through these convolutions for remote hosts that the local host can't access directly, you can set the variable `ange-ftp-local-host-regexp` to a regular expression which matches those hostnames that X can access, but does not match those hosts that only the gateway can access. In other words, if the host you are trying to access matches `ange-ftp-local-host-regexp`, the FTP process will be run on the local machine, otherwise it will be run on the gateway machine. For example

```
(setq ange-ftp-local-host-regexp "\\\\.hp\\.com$\\|^\\[\\.]*$")
```

will match all hosts that are in the `'hp.com'` domain, or don't have an explicit domain in their name, but will fail to match hosts with explicit domains or that are specified by their IP address.

The next step is to determine whether you have a smart gateway, that is, if the FTP process on the gateway will accept commands of the form `USER user@host`. You can test this by trying a manual FTP:

```
X$ ftp -n Y.local.lan.edu
Connected to Y.local.lan.edu
220 Y.local.lan.edu FTP server (Version ???? some-date) ready.
ftp> user myname@Z.foo.bar.com
```

If you then got a message like:

```
331 Password required for myname@Z.foo.bar.com
Password:
530 Login incorrect.
Login failed.
```

then you *don't* have a smart gateway. If you do, then something else happens – but since it doesn't work for me I can't say what! Anyway, if you do have a smart gateway, put the line

```
(setq ange-ftp-smart-gateway t)
```

in your `.emacs`. You may also wish to set the variable `ange-ftp-smart-gateway-port` to the port of the gateway machine to use when smart gateway is in operation, but the default of 21 will probably be fine. In any case, your installation has finished, so See Chapter 3 [Using ange-ftp], page 9, now – the rest of this section is of no use to you. If on the other hand you don't have a smart gateway, put the line

```
(setq ange-ftp-smart-gateway nil) ; this is the default
```

in your `.emacs` and read on.

Since to get files from Z to X we need to copy from Z to Y, and then from Y to X, we need a place to store files on Y which is also accessible by X, i.e. we need a directory which is mounted on both X and Y. Since we are assuming that the local host and the gateway machine are on the same local network, it's fairly likely that this is the case thanks to NFS. If such a directory exists, then ange-ftp can transfer files from Z to X simply by FTP'ing from Z to the temporary directory on Y, and then using a normal (local) copy from the image of the temporary directory on X to the destination directory. Unfortunately, ange-ftp requires that this temporary directory has the *same* name on both the local and gateway machines, so you might need to do some twiddling with symbolic links, or ask your sysadmin to set something up with NFS. Once you have found such a directory, set the variable `ange-ftp-gateway-tmp-name-template` to the name of this directory plus an identifying filename prefix. For example:

```
(setq ange-ftp-gateway-tmp-name-template "/nfs/hplose/ange/ange-ftp")
```

where `/nfs/hplose/ange` is a directory that is shared between the gateway machine Y and the local machine X.

The next step is to find a means of getting an FTP process running on the gateway machine. The simplest method is to spawn a remote shell using `remsh` or `rsh` or their equivalent. Unfortunately, this doesn't always work — try the following:

```
X$ rsh Y.local.lan.edu ftp
```

On my system, this command simply hangs. On others, it might be disallowed for security reasons. If it doesn't work for you, then skip the rest of this paragraph. If it does, then you

should set then variable `ange-ftp-gateway-program` to the name of the program used to spawn a remote shell. The default is `"remsh"` on HP-UX machines, and `"rsh"` otherwise. You should also set `ange-ftp-gateway-program-interactive` to `nil`:

```
(setq ange-ftp-gateway-program "rsh")
(setq ange-ftp-gateway-program-interactive nil)
```

and now your installation is complete.

Since spawning a remote shell didn't work, `ange-ftp` needs to actually login to the gateway machine to run `ftp`, the same as you would do if you were running `ftp` manually. So you need to set the variable `ange-ftp-gateway-program` to the name of the program that lets you log onto the gateway machine — probably `"rlogin"` or `"telnet"`:

```
(setq ange-ftp-gateway-program "rlogin")
```

Now set the variable `ange-ftp-gateway-prompt-pattern` to a regular expression that matches the prompt you get when you login to the gateway machine. Be very specific here; this regexp must not match *anything* in your login banner except this prompt. `shell-prompt-pattern` is far too general as it appears to match some login banners from Sun machines. For example:

```
(setq ange-ftp-gateway-prompt-pattern "[^$]*\\$ *")
```

You also need to set the variable `ange-ftp-gateway-program-interactive` to `t` to let `ange-ftp` know that it has to "hand-hold" the login to the gateway machine:

```
(setq ange-ftp-gateway-program-interactive t)
```

Now comes a slightly tricky bit. You need to set the variable `ange-ftp-gateway-setup-term-command` to a UNIX command that will put the pty connected to the gateway machine into a no-echoing mode, and will strip off carriage-returns from output from the gateway machine. The default is `"stty -onlcr -echo\n"` for HP-UX machines, and `"stty -echo nl\n"` otherwise. So

```
(setq ange-ftp-gateway-setup-term-command "stty -echo nl\n")
```

will probably work. If it does, then you're done. There's a bit of a problem for `tcsh` users, though: in some versions of `tcsh`, the "tty sanity checking" feature prevents the above commands from working. In this case, an easy fix is to invoke `csch` first, and then run the `stty`:

```
(setq ange-ftp-gateway-setup-term-command "exec csch\nstty -echo nl\n")
```

or maybe

```
(setq ange-ftp-gateway-setup-term-command "(stty -echo nl; csch)\n")
```

may well do the trick. When using this method, synchronisation may be a problem: if your gateway machine is slow in responding, when `ange-ftp` is ready for (and indeed has already sent) FTP commands, your machine may still be setting up. This can cause `ange-ftp` to think that the FTP has had an error, and abort. One solution is to set `ange-ftp-skip-msgs` (a regular expression matching messages from the `ftp` process that can be ignored) so that any line ending in `^M` (carriage-returns) will be ignored (since the `stty` hasn't come into effect yet) and also to ignore any lines beginning with your prompt (since this means the terminal setup is still in progress):

```
(setq ange-ftp-skip-msgs
  (concat "\\|^\\.\\C-M$\\|^\\|^" ange-ftp-gateway-prompt-pattern))
```

```
ange-ftp-skip-msgs))
```

Unfortunately, this can also mean that sometimes ange-ftp won't recognise a *real* error, and simply hang – but if that ever happens *C-g* might get you out of it.

## 2.4 Other user options

Here are the other user options available in ange-ftp:

**ange-ftp-netrc-filename:** The name of a file in `netrc(5)` format that ange-ftp will use to match hostnames, users and their respective passwords. Hostnames specified here are also used for hostname completion. The default is `"~/.netrc"`.

**ange-ftp-default-user:** If this is a string, it is the username to use when none is specified in a filename. If `nil`, then the name under which the user is logged in is used. If non-`nil` but not a string, the user is prompted for the name. The default is `nil`.

**ange-ftp-default-password:** The password to use when the user is the same as **ange-ftp-default-user**. The default is `nil`.

**ange-ftp-default-account:** Account password to use when the user is the same as **ange-ftp-default-user**. The default is `nil`.

**ange-ftp-generate-anonymous-password:** If this is `t`, then ange-ftp will generate a password of the form `your_username@local_host` when you specify a username of `anonymous` in the filename (or if you are automatically logged in as `anonymous`). If this is a string, then that string is used instead. If it is `nil`, then the user is prompted for a password. The default is `nil`.

**ange-ftp-dumb-unix-host-regexp:** The FTP servers on some machines have problems if the `ls` command is used. The usual indication that something is wrong is when ange-ftp erroneously thinks that a directory is just a plain file. The routine **ange-ftp-add-dumb-unix-host** can be called to tell ange-ftp to limit itself to the `DIR` command and not `ls` for a given host (but this change will take effect for the current GNU Emacs session only). If a large number of machines with similar hostnames have this problem then it is easier to change the value of this variable to a regexp which matches hostnames which have this problem, particularly since ange-ftp cannot automatically detect such hosts. The default is `nil`.

**ange-ftp-binary-file-name-regexp:** By default ange-ftp will transfer files in ASCII mode. If a file being transferred matches the value of this regexp then the FTP process will be toggled into BINARY mode before the transfer and back to ASCII mode after the transfer. The default is:

```
(concat "\\Z$\\\\\\\\.lzh$\\\\\\\\.arc$\\\\\\\\.zip$\\\\\\\\.zoo$\\\\\\\\.tar$\\\\\\\\|"
        "\\dvi$\\\\\\\\.ps$\\\\\\\\.elc$\\\\\\\\|TAGS$\\\\\\\\|"
        "\\gif$\\\\\\\\.EXE\\\\\\\\(;[0-9]+\\\\\\\\)?$")
```

**ange-ftp-hash-mark-size:** Ange-ftp by default requests that the FTP process sends hash marks (just `#` characters) during transfers to keep track of how much data has been sent or received. This variable, if non-`nil`, should be the number of kilobytes represented by the FTP client's hash mark. The default value of 1 doesn't work for me — I use 2 instead.

**ange-ftp-process-verbose:** If this is `t` then ange-ftp will be chatty about interaction with the FTP process. The default is `t`.

**ange-ftp-ftp-program-name:** This should be the name of the FTP program to run on the local host. The default value of "ftp" should be fine for most systems.

**ange-ftp-gateway-ftp-program-name:** Same as above, but this time it's the name of the program to be used if a gateway is in use. The default is again "ftp", but some AT&T folks claim to use something called "pftp" here.

**ange-ftp-make-backup-files:** A list of operating systems for which ange-ftp will make Emacs backup files on the remote host. For example, '(unix)' makes sense, but '(unix vms)' or '(vms)' would be silly, since VMS makes its own backups. The host type is determined by the function **ange-ftp-host-type**. Possible host types are: **dumb-unix**; **vos**; **vms**; **mts**; and **unix**. The default of **nil** means make no backups on remote hosts.

**ange-ftp-path-format:** This variable dictates the the format of a fully expanded remote pathname. This is a cons (REGEXP . (HOST USER PATH)), where REGEXP is a regular expression matching the full remote pathname, and HOST, USER, and PATH are the numbers of parenthesised expressions in REGEXP for the components (in that order). The syntax can be customised with this variable to a certain extent, but there are limitations. The default is

```
'("^~/\\(\\([^\t@/:]*)\\)@\\)?\\(\\([^\t@/:]*)\\):\\(\\.\\(.*\\)\" . (3 2 4)).
```

**ange-ftp-multi-msgs:** A regular expression matching messages from the ftp process that start a multiline reply. The default is

```
"^220-\\|^230-\\|^226\\|^25.-\\|^221-\\|^200-\\|^530-\\|^4[25]1-"
```

**ange-ftp-good-msgs:** A regular expression matching messages from the ftp process that indicate that the action that was initiated has completed successfully. The default is "**^220 \\|^230 \\|^226\\|^25. \\|^221 \\|^200 \\|^\\[Hh\\]ash mark**".

**ange-ftp-skip-msgs:** A regular expression matching messages from the ftp process that can be ignored. The default is

```
(concat "^200 \\(PORT\\|Port\\) \\|^331 \\|^150 \\|^350 \\|^\\[0-9\\]+ bytes \\|^"
      "^Connected \\|^$\\|^Remote system\\|^Using\\|^ \\|^Password:\\|^"
      "^local:\\|^Trying\\|^125 \\|^550-")
```

but you might need to tweak it if ange-ftp is giving up when it shouldn't.

**ange-ftp-fatal-msgs:** A regular expression matching messages from the FTP process that indicate something has gone drastically wrong attempting the action that was initiated and that the FTP process should (or already has) been killed. The default is

```
(concat "^ftp: \\|^Not connected\\|^530 \\|^4[25]1 \\|^rcmd: \\|^"
      "^No control connection\\|^unknown host\\|^lost connection")
```

**ange-ftp-gateway-fatal-msgs:** Regular expression matching messages from the rlogin / telnet process that indicates that logging in to the gateway machine has gone wrong. The default is

```
"No route to host\\|^Connection closed\\|^No such host\\|^Login incorrect"
```

**ange-ftp-tmp-name-template:** This should be a directory and a filename prefix indicating where ange-ftp should make temporary files. The default of "/tmp/ange-ftp" should be fine for most systems.

**ange-ftp-retry-time:** Number of seconds to wait before retrying if a file or listing doesn't arrive. For slow connections, you might get a "listing unreadable" error messages or an empty buffer for a file that you know has something in it. The solution is to increase

the value of `ange-ftp-retry-time`. Its default value is 5 which is plenty for reasonable connections. However, for some transatlantic connections 20 might be a better value.



## 3 Using ange-ftp

Once you have ange-ftp installed, you never need worry about using FTP again. The interface is completely transparent, and you may now use Emacs commands such as **C-x C-f** (**find-file**) on *any* file that your local host (or, if you are using one) your gateway can access. That file may be a regular file (for editing, viewing etc.), a directory (for invoking **Dired**) or even a symbolic link (pointing to a directory or a regular file). All it takes is an extended filename syntax. For example, if you give the filename

```
/ange@anorman:/tmp/notes
```

to **find-file**, then ange-ftp will spawn an FTP process, connect to the host **anorman** as user **ange**, get the file **/tmp/notes** and pop up a buffer containing the contents of that file as if it were on the local filesystem. If ange-ftp needed a password to connect then it would prompt the user in the minibuffer. From then on you can edit that file as if it were any other file: saving is with **C-x C-s** as usual — in fact, everything is as usual.

Ange-ftp is also extremely useful for regular "file-transfer" FTP jobs. Since **Dired** also works on remote directories when using ange-ftp, you will be able to browse the filesystem on your favourite archive site with consummate ease.

### 3.1 Remote filename syntax

The general form of the extended filename syntax is

```
/user@host:path
```

which refers to the file pointed to by **path** on machine **host** when logging in as user **user**. When **path** is supplied as a relative file-name (that is, without a leading '/') it is relative to **user**'s home directory (although such relative filenames are ultimately converted to absolute ange-ftp pathnames). You may even refer to home directories of users on remote Unix sites using the standard tilde '~' notation. **host** needs to be the fully qualified pathname if the local or gateway machine requires it to be (however hostname completion is available if it is included in your **.netrc** — See Section 3.3 [Using a **.netrc**], page 10), or it may be an IP number if your nameserver can't find the site. The **user@** part may be omitted, in which case the username is chosen on the basis of the variable **ange-ftp-default-user** (See Section 2.4 [Other options], page 6) and your **.netrc**. If a password is requested by the FTP process, ange-ftp will either prompt you for it, or generate one on the basis of the variables **ange-ftp-default-password**, and your **.netrc**.

Thus the following are all valid ange-ftp filenames:

```
/anonymous@waldo.uranium.com:pub/games/wumpus
/root@127.44.3.1:/etc/passwd
/jbrown@freddie.ucla.edu:~mblack/
/alpha.gnu.ai.mit.edu:ange-ftp/ange-ftp.tar.Z
```

### 3.2 Using **Dired**

This feature of ange-ftp is particularly useful when file-transfers, as opposed to file-editing, are the order of the day. Simply run **find-file** on a directory to get a listing of the files in that directory. For example, you might run **find-file** on

```
/anonymous@archive.site.com:pub
```

to see what's in the `pub` directory of your favourite archive site. This brings up a `Dired` buffer of all the files in that directory. The `f` command is useful for looking at `README` files — if you then decide to save it `C-x C-w` is useful. You can also use this method to copy files, but the `c` command is easier. The `f` command can also be used to descend the directory tree by applying it to directories.

You can also use `Dired` to refresh `ange-ftp`'s internal cache. If you (or anybody else) has changed a remote directory since you first accessed it with `ange-ftp`, completion is not provided on any new files that `ange-ftp` does not know about. If you have (or create) a `Dired` buffer which contains the modified directory, executing `revert-buffer` with a prefix argument (`C-u g` in the `Dired` buffer) will force a refresh of both the the buffer *and also* *ange-ftp's internal cache*. If you find that filename completion isn't working on a file that you *know* is there, this is how to fix the problem.

The version of `Dired` supplied with Emacs version 18.58 (and earlier versions) does not include a capability for multiple file transfers. The `Tree Dired` package (See Section 2.1 [Obtaining source code], page 2), however, is ideal for this application. `Tree Dired` provides facilities for maintaining an entire directory tree in a `Dired` buffer, for marking files which match a certain regexp (or you can select files interactively) and then copying all those files to your local host (or even a different remote host). Another useful feature is `Virtual Dired`, which allows you to save `Dired` buffers of remote hosts, allowing you to browse them at a later date without actually needing to connect to the host. See the documentation for `Tree Dired` for more details.

Since `ange-ftp` is mostly transparent, modifying `Dired` or `Tree Dired` by means of hooks or keybindings should still work.

### 3.3 Using a .netrc file

Being prompted for passwords all the time can get rather annoying, but there is a way to fix the problem — a `.netrc` (but See Section 2.4 [Other options], page 6, and `ange-ftp-netrc-filename` if you want another filename) file in your home directory. Basically, this is a file (in the format of Unix `netrc(5)`) which contains the names of all the machines you regularly login to, as well as the username and password you use for that machine. You can also supply an account password, if required.

Your `.netrc` file consists of lines of the form

```
machine <machine-name> login <user-name> password <password>
```

It doesn't all have to be on the one line, though: any `login` or `password` commands in the file refer to the previous `machine` command. You can also have `account <account-passwd>` commands if you need special account passwords.

For example, you might have the following line in your `.netrc`:

```
machine Y.local.lan.edu login myname password secret
```

Then if you run `find-file` on the file `/Y.local.lan.edu:somefile` you will automatically be logged in as user `myname` with password `secret`. You can still login under another name and password, if you so desire: just include the `user@` part of the filename.

You may also include a default option, as follows:

```
default login <user-name> password <password>
```

which applies to any other machines not mentioned elsewhere in your `.netrc`. A particularly useful application of this is with anonymous logins:

```
default login myname password myname@myhost.edu
```

so that accessing `/anyhost:anyfile` will automatically log you in anonymously, provided the host is not mentioned in the `.netrc`. Note also that if the value of `ange-ftp-default-user` is non-nil, its value will have precedence over the username supplied in the default option of the `.netrc`.

The `.netrc` file is also useful in another regard: machines included in it are provided with hostname completion. That is, for any machine in the `.netrc`, you need only type a slash and the first few characters of its name and then press `TAB` to be logged in automatically with a username and password from the `.netrc` file. So it's a good idea to put hosts you use regularly in your `.netrc` as well:

```
machine archive.site.com login anonymous password myname@X.local.lan.edu
```

### 3.4 Ange-ftp commands

Ange-ftp supplies a few interactive commands to make connecting with hosts a little easier.

Command `ange-ftp-set-user`: Prompts for a hostname and a username. Next time access to the host is attempted, ange-ftp will attempt to log in again with the new username.

Command `ange-ftp-set-passwd`: Prompts for a hostname, user and password. Future logins to that host as that user will use the given password.

Command `ange-ftp-set-account`: Prompts for a hostname, user and account. Future logins to that host as that user will use the given account.

Note that the effects of the above three commands only last the duration of the current Emacs session. To make their effects permanent, you may include them as lisp code in your `.emacs`:

```
(ange-ftp-set-user HOST USER)
(ange-ftp-set-password HOST USER PASSWORD)
(ange-ftp-set-account HOST USER ACCOUNT)
```

This is an alternative to using a `.netrc`; See Section 3.3 [Using a `.netrc`], page 10.

Command `ange-ftp-kill-ftp-process`: kill the FTP process associated with a given buffer's filename (by default the current buffer). This is an easy way to achieve a resynch: any future accesses to the remote host will cause the FTP process to be recreated.

### 3.5 Descriptive directory listings

Some hosts (such as `cs.uwp.edu`) now use descriptive directory listings (which in fact contain *less* information than the standard listing!) when issued the `ls` command, and ange-ftp has been modified to cope with this. Ange-ftp can detect such listings, but if you regularly use a remote host which uses this extended listing format you should set the variable `ange-ftp-dl-dir-regexp` to a regular expression which matches directories using the extended listing format. You shouldn't anchor the regexp with `'$'` – that way the regexp will match subdirectories as well. Alternatively, you can use the interactive command `ange-ftp-add-dl-dir` to temporarily add a remote directory for this Emacs session only.

Tree Dired has been modified to work with such descriptive listings.

## 3.6 Using ange-ftp with non-Unix hosts

Ange-ftp also works with some non-Unix hosts, although not necessarily with all the features available with Unix hosts. VMS, CMS, and MTS systems will all now work with ange-ftp and Tree Dired (although Classical Dired may well be broken on such systems.) Filename completion also now works on these hosts.

Ange-ftp should be able to automatically detect which type of host you are using (VMS, CMS or MTS), but if it is unable to do so you can fix the problem by setting the appropriate `ange-ftp-TYPE-host-regexp` variable (where `TYPE` is one of `'vms'`, `'cms'` or `'mts'`) – see below. If ange-ftp is unable to automatically detect any VMS, CMS or MTS host, please report this as a bug: See Chapter 5 [Bugs], page 17.

In all cases the file-name conventions of the remote host are converted to a UNIX-ish format, and this is the format you should use to find files on such hosts.

### 3.6.1 VMS support

VMS filenames are of the form `FILE.TYPE;##`, where both `FILE` and `TYPE` can be up to 39 characters long, and `##` is an integer version number between 1 and 32,767. Valid characters in filenames are `'A'-'Z'`, `'0'-'9'`, `'_'`, `'-'` and `'$'`, however `'$'` cannot begin a filename and `'-'` cannot be used as the first or last character.

Directories in VMS are converted to the standard UNIX `'/'` notation. For example, the VMS filename

```
PUB$: [ANONYMOUS.SDSCPUB.NEXT]README.TXT;1
```

would be entered as

```
/PUB$$:/ANONYMOUS/SDSCPUB/NEXT/README.TXT;1
```

(The double `'$'` is required to prevent Emacs from attempting to expand an environment variable.) Similarly, to anonymously FTP the file `[.CSV.POLICY]RULES.MEM;1` from `ymir.claremont.edu` you would type `C-x C-f /anonymous@ymir.claremont.edu:CSV/POLICY/RULES.MEM;1`. You can always drop off the `';##'` part at the end of the filename to get the latest version.

Sandy Rutherford provides some tips for using VMS hosts:

- Although VMS is not case sensitive, EMACS running under UNIX is. Therefore, to access a VMS file, you must enter the filename with upper case letters.
- To access the latest version of file under VMS, you use the filename without the `';'` and version number. You should always edit the latest version of a file. If you want to edit an earlier version, copy it to a new file first. This has nothing to do with ange-ftp, but is simply good VMS operating practice. Therefore, to edit `FILE.TXT;3` (say 3 is latest version), do `C-x C-f /ymir.claremont.edu:FILE.TXT`. If you inadvertently do

```
C-x C-f /ymir.claremont.edu:FILE.TXT;3
```

you will find that VMS will not allow you to save the file because it will refuse to overwrite `FILE.TXT;3`, but instead will want to create `FILE.TXT;4`, and attach the buffer to this file. To get out of this situation, `M-x write-file /ymir.claremont.edu:FILE.TXT` will attach the buffer to latest version of the file. For this reason, in Tree Dired `f` (`dired-find-file`), always loads the file sans version, whereas `v`, (`dired-view-file`), always loads the explicit version number. The reasoning being that it is reasonable to view old versions of a file, but not to edit them.

- VMS filenames often contain ‘\$’ characters: make sure you always quote these as ‘\$\$’ and watch out for the Emacs bug which fails to quote ‘\$’s when defaults are presented in the minibuffer: see See Chapter 5 [Bugs], page 17.

Ange-ftp should automatically detect that you are using a VMS host. If it fails to do so (which should be reported as a bug) you can use the command `ange-ftp-add-vms-host` to inform ange-ftp manually. For a more permanent effect, or if you use a VMS host regularly, it’s a good idea to set `ange-ftp-vms-host-regexp` to a regular expression matching that host’s name. For instance, if use `yimir.claremont.edu` a lot, place the following in your `.emacs`:

```
(setq ange-ftp-vms-host-regexp "^ymir.claremont.edu$")
```

### 3.6.2 CMS support

Ange-ftp has full support, including Tree Dired support, for hosts running CMS.

CMS filenames are entered in a UNIX-y way. Minidisks are treated as UNIX directories; for example to access the file `READ.ME` in minidisk `*.311` on `cuvmb.cc.columbia.edu`, you would enter

```
/anonymous@cuvmb.cc.columbia.edu:/*.311/READ.ME
```

If `*.301` is the default minidisk for this account, you could access `FOO.BAR` on this minidisk as

```
/anonymous@cuvmb.cc.columbia.edu:FOO.BAR
```

CMS filenames are of the form `FILE.TYPE`, where both `FILE` and `TYPE` can be up to 8 characters. Again, beware that CMS filenames are always upper case, and hence must be entered as such.

Sandy Rutherford provides some tips on using CMS hosts:

- CMS machines, with the exception of anonymous accounts, nearly always need an account password. To have ange-ftp send an account password, you can either include it in your `.netrc` (See Section 3.3 [Using a .netrc], page 10), or use `ange-ftp-set-account`.
- Ange-ftp cannot send “write passwords” for a minidisk. Hopefully, we can fix this.

Ange-ftp should automatically detect that you are using a CMS host. If it fails to do so (which should be reported as a bug) you can use the command `ange-ftp-add-cms-host` to inform ange-ftp manually. For a more permanent effect, or if you use a CMS host regularly, it’s a good idea to set `ange-ftp-cms-host-regexp` to a regular expression matching that host’s name.

### 3.6.3 MTS support

Ange-ftp has full support, including Tree Dired support, for hosts running the Michigan terminal system, and should be able to automatically recognise any MTS machine.

MTS filenames are entered in a UNIX-y way. For example, if your account was `YYYY`, the file `FILE` in the account `XXXX` on `mtsg.ubc.ca` would be entered as

```
/YYYY@mtsg.ubc.ca:/XXXX:/FILE
```

In other words, MTS accounts are treated as UNIX directories. Of course, to access a file in another account, you must have access permission for it. If **FILE** were in your own account, then you could enter it in a relative path fashion as

```
/YYYY@mtsg.ubc.ca:FILE
```

MTS filenames can be up to 12 characters. Like UNIX, the structure of the filename does not contain a type (i.e. it can have as many ‘.’s as you like.) MTS filenames are always in upper case, and hence be sure to enter them as such! MTS is not case sensitive, but an EMACS running under UNIX is.

Ange-ftp should automatically detect that you are using an MTS host. If it fails to do so (which should be reported as a bug) you can use the command **ange-ftp-add-mts-host** to inform ange-ftp manually. For a more permanent effect, or if you use an MTS host regularly, it’s a good idea to set **ange-ftp-mts-host-regexp** to a regular expression matching that host’s name.

### 3.7 File- and host-name completion

Full filename completion is supported on all remote UNIX hosts and some non-Unix hosts. Hostnames also have completion if they are mentioned in the **.netrc** and no username is specified. However using the filename completion feature can be a bit of a two edged sword.

To understand why, we need to discuss how ange-ftp works. Whenever ange-ftp is asked to find a remote file (or directory) an **ls** command is sent to the FTP process to list all the files in the directory. This list is maintained in an internal cache, to provide filename completion for later requests on that directory. Ange-ftp keeps this cache up-to-date by monitoring Emacs commands which affect files and directories, but if a process outside Emacs (such as another user) changes a directory (e.g. a new file is added) completion won’t work on that file since ange-ftp doesn’t know about it yet. The solution is to force ange-ftp to reread the directory and update it’s cache, and the easiest way to do that is with **Dired** — See Section 3.2 [Using Dired], page 9, to see how.

Another problem is that the **ls** command can take a long time, especially when dealing with distant hosts over slow links. So if you’re after a file in the **pub/images** directory but nothing else, it’s a better idea to type **pub/images/file** TAB than **pub/im** TAB which will force a read of the **pub** directory (since ange-ftp needs to know how to complete **im**). A little extra typing can often save a lot of waiting. Don’t be afraid to use the TAB key once the directory is cached, though.

### 3.8 Accessing the FTP process buffer

The FTP process used to access the remote files is available for access if you wish. It will be in a buffer called “**\*ftp remote-file-name\***”, i.e. if you found the file

```
/anonymous@archive.site.com:pub/README
```

there will be a buffer

```
*ftp anonymous@archive.site.com*
```

where all the transfers are taking place. You can have a look at the buffer using **C-x b** as usual, and even type in commands to the FTP process under an interface very much like ‘**shell-mode**’. There are two instances when doing this can be very useful: one is accessing non-UNIX hosts, where **Dired** and filename completion may not work (if ange-ftp

even works at all). The other is multiple (i.e. wildcard) file transfers which the standard version of Dirent does not handle (but Tree Dirent *does*, and is worth installing for this feature alone.) If you are going to use `mget` or `mput`, make sure you type `glob` first: ange-ftp turns globbing off by default. Don't be afraid of changing directories, either — ange-ftp always uses absolute pathnames when communicating with the FTP process.

You can kill the FTP process at any time simply by killing this buffer. This won't cause ange-ftp any grief whatsoever — if you later make another request to that host, ange-ftp will simply fire up another process and create a new buffer to hold it.

## 4 Getting help

Ange-ftp has its own mailing list modestly called ange-ftp-lovers where ange-ftp users discuss new features, problems, bug fixes etc. There is also another list called ange-ftp-lovers-announce which is reserved exclusively for the announcement of new versions. All users of ange-ftp are welcome to subscribe (see below) to either of these lists. New versions of ange-ftp are posted periodically to these lists.

To [un]subscribe to ange-ftp-lovers or ange-ftp-lovers-announce, or to report mailer problems with the list, please mail one of the following addresses:

```

ange-ftp-lovers-request@anorman.hpl.hp.com
ange-ftp-lovers-request%anorman.hpl.hp.com@hplb.hpl.hp.com
hplb.hpl.hp.com!anorman.hpl.hp.com!ange-ftp-lovers-request
hplabs.hpl.hp.com!anorman.hpl.hp.com!ange-ftp-lovers-request

```

Please don't forget the '-request' part, and please make it clear in the request which mailing list you wish to join.

For mail to be posted directly to ange-ftp-lovers, send to one of the following addresses:

```

ange-ftp-lovers@anorman.hpl.hp.com
ange-ftp-lovers%anorman.hpl.hp.com@hplb.hpl.hp.com
hplb.hpl.hp.com!anorman.hpl.hp.com!ange-ftp-lovers
hplabs.hpl.hp.com!anorman.hpl.hp.com!ange-ftp-lovers

```

The ange-ftp-lovers mailing list is archived on

```
ftp.reed.edu:pub/mailling-lists/ange-ftp/
```

The newsgroup `gnu.emacs.help` also occasionally discusses ange-ftp.



## 5 Bugs and Wish List

Here is a list of the known bugs in ange-ftp:

- Be warned that files created by using ange-ftp will take account of the umask of the ftp daemon process rather than the umask of the creating user. This is particularly important when logging in as the root user. The way that I tighten up the ftp daemon's umask under HP-UX is to make sure that the umask is changed to 027 before I spawn `/etc/inetd`. I suspect that there is something similar on other systems.
- Some combinations of FTP clients and servers break and get out of sync when asked to list a non-existent directory. Some of the `ai.mit.edu` machines cause this problem for some FTP clients.
- Ange-ftp does not check to make sure that when creating a new file, you provide a valid filename for the remote operating system. If you do not, then the remote FTP server will most likely translate your filename in some way. This may cause ange-ftp to get confused about what exactly is the name of the file. The most common causes of this are using lower case filenames on systems which support only upper case, and using filenames which are too long.
- Null (blank) passwords confuse both ange-ftp and some FTP daemons.
- ange-ftp likes to use pty's to talk to its FTP processes. If GNU Emacs creates a FTP process that only talks via pipes (for example, if `process-connection-type` is `nil`) then ange-ftp won't be getting the information it requires at the time that it wants it since pipes flush at different times to pty's. One disgusting way around this problem is to talk to the FTP process via `rlogin` which does the 'right' things with pty's.
- You need to quote '\$' characters in filenames by using '\$\$' instead. This isn't actually a bug, but rather an Emacs convention (which allows environment variables in filenames.) What *is* a bug is that when filenames containing '\$'s are inserted in the minibuffer as defaults, the '\$' is not converted into the '\$\$' quoted form — hopefully this will be fixed in version 19. It doesn't usually bother Unix users, but VMS filenames often contain '\$'. Incidentally, Sebastian Kremer's `gmhist` package (which comes with the Tree Dired distribution: See Section 2.1 [Obtaining source code], page 2) fixes this bug.
- Some hosts (notably ULTRIX) mark symbolic links with a '@' character in an `ls -F` listing. The variable `dired-ls-F-marks-symlinks` when set to `t` (the default) alerts Dired to this behaviour and everything is OK. Enabling this behaviour by default is not generally a problem on hosts which does *not* mark symlinks in this way, but if you have `dired-ls-F-marks-symlinks` set to `t` while accessing a such a host, then Dired will think that a symbolic link whose name ends in '@' (a strange thing indeed!) is a regular file. The fix (other than setting `dired-ls-F-marks-symlinks` to `nil`, a bad idea if you regularly access hosts who mark symbolic links) is to remove 'F' from the `ls` listing switches (use `C-u s` in the Dired buffer.)

Another problem with symbolic links arises with hosts who do not show the linked file with '->' in the listing, meaning that Dired will not recognize the symlink. The solution here is to get a decent `ls` program on that machine.

- No classic dired support for non-UNIX systems. Tree dired was enough.

- If a directory listing is attempted for an empty directory on (at least some) VMS hosts, an ftp error is given. This is really an ftp bug, and I don't know how to get ange-ftp work to around it.
- Bombs on filenames that start with a space. Deals well with filenames containing spaces, but beware that the remote ftpd may not like them much.
- Doesn't autosave. Maybe someone could implement auto-saving on the local host ...
- The code to do compression of files over ftp is not as careful as it should be. It deletes the old remote version of the file, before actually checking if the local to remote transfer of the compressed file succeeds. Of course to delete the original version of the file after transferring the compressed version back is also dangerous, because some OS's have severe restrictions on the length of filenames, and when the compressed version is copied back the `-Z` or `.Z` may be truncated. Then, ange-ftp would delete the only remaining version of the file. Maybe ange-ftp should make backups when it compresses files?
- Remote to remote copying of files on non-Unix machines can be risky. Depending on the variable `ange-ftp-binary-file-name-regex`, ange-ftp will use binary mode for the copy. Between systems of different architecture, this still may not be enough to guarantee the integrity of binary files. Binary file transfers from VMS machines are particularly problematical.
- Some CMS machines do not assign a default minidisk when you ftp them as anonymous. It is then necessary to guess a valid minidisk name, and `cd` to it. This is (understandably) beyond ange-ftp; however Sebastian Kremer says:

It is beyond ange-ftp, but if the `init ftp` macro were supported, one could write the appropriate `cd` command into that. I used to do that on a CMS machine I had an account on because I never could remember the name of the minidisk. I think I even had to give an `account` command, too. Supporting `init` would be a very handy thing.

Hmm, why start `ftp(1)` with the `-n` flag at all?

- For CMS support, we send too many `cd`'s. Since `cd`'s are cheap, I haven't worried about this too much. Eventually, we should have some caching of the current minidisk.

If you find any bugs or problems with this package, **please** e-mail the author. Ideas and constructive comments are especially welcome. So are any enhancements to ange-ftp, preferably debugged and documented. Also welcome are any typo fixes, corrections or additions to this manual. And just so you don't forget, here's Ange's address again:

`ange@hplb.hpl.hp.com`

Enjoy!

# Concept Index

## A

anonymous FTP ..... 11  
 archive sites ..... 10  
 auto-saving ..... 18

## B

backup files ..... 7  
 buffers ..... 14

## C

CMS filenames ..... 13  
 CMS minidisks ..... 18  
 compressing files ..... 18  
 copying ..... 18

## D

descriptive directory listings ..... 11

## E

Emacs-Lisp Archive ..... 2  
 extended directory listings ..... 11

## F

filename completion ..... 10  
 FTP processes ..... 15

## H

home directories of other users ..... 9  
 hostname completion ..... 11  
 HP-UX ..... 5

## I

IP numbers ..... 9

## L

listing unreadable error ..... 7  
 load path ..... 3

## M

MTS filenames ..... 13  
 multiple file transfers ..... 10, 15

## N

NFS ..... 4  
 numeric Internet addresses ..... 9

## O

other users' home directories ..... 9

## P

process buffers ..... 14  
 processes ..... 15  
 pty ..... 17

## R

remote shell ..... 4

## S

smart gateways ..... 4  
 symbolic links ..... 9, 17

## T

tcsh ..... 5  
 temporary files ..... 4, 7  
 Tree Dired, source ..... 2

## U

ULTRIX ..... 17  
 umask ..... 17

## V

virtual dired ..... 10  
 VMS filenames ..... 12

## W

wildcards ..... 10, 15

## Variable and command index

### A

ange-ftp-add-cms-host ..... 13  
 ange-ftp-add-dl-dir ..... 11  
 ange-ftp-add-dumb-unix-host ..... 6  
 ange-ftp-add-mts-host ..... 14  
 ange-ftp-add-vms-host ..... 13  
 ange-ftp-binary-file-name-regexp ..... 6  
 ange-ftp-cms-host-regexp ..... 13  
 ange-ftp-default-account ..... 6  
 ange-ftp-default-password ..... 6, 9  
 ange-ftp-default-user ..... 6, 9, 11  
 ange-ftp-dl-dir-regexp ..... 11  
 ange-ftp-dumb-unix-host-regexp ..... 6  
 ange-ftp-fatal-msgs ..... 7  
 ange-ftp-ftp-program-name ..... 7  
 ange-ftp-gateway-fatal-msgs ..... 7  
 ange-ftp-gateway-ftp-program-name ..... 7  
 ange-ftp-gateway-host ..... 3  
 ange-ftp-gateway-program ..... 5  
 ange-ftp-gateway-program-interactive ..... 5  
 ange-ftp-gateway-prompt-pattern ..... 5  
 ange-ftp-gateway-setup-term-command ..... 5  
 ange-ftp-gateway-tmp-name-template ..... 4  
 ange-ftp-generate-anonymous-password ..... 6  
 ange-ftp-good-msgs ..... 7  
 ange-ftp-hash-mark-size ..... 6  
 ange-ftp-kill-ftp-process ..... 11  
 ange-ftp-local-host-regexp ..... 3  
 ange-ftp-make-backup-files ..... 7  
 ange-ftp-mts-host-regexp ..... 14  
 ange-ftp-multi-msgs ..... 7  
 ange-ftp-netrc-filename ..... 6, 10  
 ange-ftp-path-format ..... 7  
 ange-ftp-process-verbose ..... 6  
 ange-ftp-retry-time ..... 8

ange-ftp-set-account ..... 11, 13  
 ange-ftp-set-passwd ..... 11  
 ange-ftp-set-user ..... 11  
 ange-ftp-skip-msgs ..... 5, 7  
 ange-ftp-smart-gateway ..... 4  
 ange-ftp-smart-gateway-port ..... 4  
 ange-ftp-tmp-name-template ..... 7  
 ange-ftp-vms-host-regexp ..... 13

### D

dired-find-file ..... 12  
 dired-ls-F-marks-symlinks ..... 17  
 dired-view-file ..... 12

### F

find-file ..... 9

### G

gmhist ..... 17

### P

process-connection-type ..... 17

### R

revert-buffer ..... 10

### S

shell-prompt-pattern ..... 5

# Table of Contents

<b>1</b>	<b>Introducing ange-ftp.....</b>	<b>1</b>
<b>2</b>	<b>Installing ange-ftp .....</b>	<b>2</b>
2.1	How to get the ange-ftp source code .....	2
2.2	Installing the source .....	3
2.3	Using a gateway .....	3
2.4	Other user options.....	6
<b>3</b>	<b>Using ange-ftp .....</b>	<b>9</b>
3.1	Remote filename syntax .....	9
3.2	Using Dired .....	9
3.3	Using a .netrc file .....	10
3.4	Ange-ftp commands .....	11
3.5	Descriptive directory listings .....	11
3.6	Using ange-ftp with non-Unix hosts.....	12
3.6.1	VMS support .....	12
3.6.2	CMS support .....	13
3.6.3	MTS support .....	13
3.7	File- and host-name completion.....	14
3.8	Accessing the FTP process buffer .....	14
<b>4</b>	<b>Getting help .....</b>	<b>16</b>
<b>5</b>	<b>Bugs and Wish List .....</b>	<b>17</b>
	<b>Concept Index .....</b>	<b>19</b>
	<b>Variable and command index .....</b>	<b>20</b>