NAME

etcupdate - manage updates to system files not updated by installworld

SYNOPSIS

etcupdate [**-npBFN**] [**-d** *workdir*] [**-r** | **-s** *source* | **-t** *tarball*] [**-A** *patterns*] [**-D** *destdir*] [**-I** *patterns*] [**-L** *logfile*] [**-M** *options*] [**-m** *make*]
etcupdate build [**-BN**] [**-d** *workdir*] [**-s** *source*] [**-L** *logfile*] [**-M** *options*] [**-m** *make*] *tarball*
etcupdate diff [**-d** *workdir*] [**-D** *destdir*] [**-I** *patterns*] [**-L** *logfile*]
etcupdate extract [**-BN**] [**-d** *workdir*] [**-s** *source* | **-t** *tarball*] [**-D** *destdir*] [**-L** *logfile*] [**-M** *options*] [**-m** *make*]
etcupdate resolve [**-p**] [**-d** *workdir*] [**-D** *destdir*] [**-L** *logfile*]
etcupdate revert [**-d** *workdir*] [**-D** *destdir*] [**-L** *logfile*] *file ...*
etcupdate status [**-d** *workdir*] [**-D** *destdir*]

DESCRIPTION

The **etcupdate** utility is a tool for managing updates to files that are not updated as part of 'make installworld' such as files in */etc*. It manages updates by doing a three-way merge of changes made to these files against the local versions. It is also designed to minimize the amount of user intervention with the goal of simplifying upgrades for clusters of machines.

To perform a three-way merge, **etcupdate** keeps copies of the current and previous versions of files that it manages. These copies are stored in two trees known as the "current" and "previous" trees. During a merge, **etcupdate** compares the "current" and "previous" copies of each file to determine which changes need to be merged into the local version of each file. If a file can be updated without generating a conflict, **etcupdate** will update the file automatically. If the local changes to a file conflict with the changes made to a file in the source tree, then a merge conflict is generated. The conflict must be resolved after the merge has finished. The **etcupdate** utility will not perform a new merge until all conflicts from an earlier merge are resolved.

MODES

The **etcupdate** utility supports several modes of operation. The mode is specified via an optional command argument. If present, the command must be the first argument on the command line. If a command is not specified, the default mode is used.

Default Mode

The default mode merges changes from the source tree to the destination directory. First, it updates the "current" and "previous" trees. Next, it compares the two trees merging changes into the destination directory. Finally, it displays warnings for any conditions it could not handle automatically.

If the **-r** option is not specified, then the first step taken is to update the "current" and "previous" trees. If

a "current" tree already exists, then that tree is saved as the "previous" tree. An older "previous" tree is removed if it exists. By default the new "current" tree is built from a source tree. However, if a tarball is specified via the **-t** option, then the tree is extracted from that tarball instead.

Next, **etcupdate** compares the files in the "current" and "previous" trees. If a file was removed from the "current" tree, then it will be removed from the destination directory only if it does not have any local modifications. If a file was added to the "current" tree, then it will be copied to the destination directory only if it would not clobber an existing file. If a file is changed in the "current" tree, then **etcupdate** will attempt to merge the changes into the version of the file in the destination directory. If the merge encounters conflicts, then a version of the file with conflict markers will be saved for future resolution. If the merge does not encounter conflicts, then the merged version of the file will be saved in the destination directory. If **etcupdate** is not able to safely merge in changes to a file other than a merge conflict, it will generate a warning.

For each file that is updated a line will be output with a leading character to indicate the action taken. The possible actions follow:

A Added
C Conflict
D Deleted
M
Merged
U Updated

Finally, if any warnings were encountered they are displayed after the merge has completed.

Note that for certain files **etcupdate** will perform post-install actions any time that the file is updated. Specifically, pwd_mkdb(8) is invoked if */etc/master.passwd* is changed, cap_mkdb(1) is invoked to update */etc/login.conf.db* if */etc/login.conf* is changed, newaliases(1) is invoked if */etc/mail/aliases* is changed, services_mkdb(8) is invoked if */etc/services* is changed, tzsetup(8) is invoked if */etc/localtime* is changed and if */var/db/zoneinfo* exists, and */etc/rc.d/motd* is invoked if */etc/motd* is changed. One exception is that if */etc/mail/aliases* is changed and the destination directory is not the default, then a warning will be issued instead. This is due to a limitation of the newaliases(1) command. Similarly, if */etc/motd* is changed and the destination directory is not the default, then */etc/rc.d/motd* will not be executed due to a limitation of that script. In this case no warning is issued as the result of */etc/rc.d/motd* is merely cosmetic and will be corrected on the next reboot.

Build Mode

The **build** mode is used to build a tarball that contains a snapshot of a "current" tree. This tarball can be used by the default and extract modes. Using a tarball can allow **etcupdate** to perform a merge without

requiring a source tree that matches the currently installed world. The *tarball* argument specifies the name of the file to create. The file will be a tar(5) file compressed with bzip2(1).

Diff Mode

The **diff** mode compares the versions of files in the destination directory to the "current" tree and generates a unified format diff of the changes. This can be used to determine which files have been locally modified and how. Note that **etcupdate** does not manage files that are not maintained in the source tree such as */etc/fstab* and */etc/rc.conf*.

Extract Mode

The **extract** mode generates a new "current" tree. Unlike the default mode, it does not save any existing "current" tree and does not modify any existing "previous" tree. The new "current" tree can either be built from a source tree or extracted from a tarball.

Resolve Mode

The **resolve** mode is used to resolve any conflicts encountered during a merge. In this mode, **etcupdate** iterates over any existing conflicts prompting the user for actions to take on each conflicted file. For each file, the following actions are available:

(p) postpone — Ignore this conflict for now.

(df) diff-full — Show all changes made to the merged file as a unified diff.

(e) edit — Change the merged file in an editor.

(r) resolved — Install the merged version of the file into the destination directory.

(mf) mine-full — Use the version of the file in the destination directory and ignore any changes made to the file in the "current" tree.

(tf) theirs-full — Use the version of the file from the "current" tree and discard any local changes made to the file.

(h) help — Display the list of commands.

Revert Mode

The **revert** mode is used to restore the stock versions of files. In this mode, **etcupdate** installs the stock version of requested files. This mode cannot be used to restore directories, only individual files.

Status Mode

The **status** mode shows a summary of the results of the most recent merge. First it lists any files for which there are unresolved conflicts. Next it lists any warnings generated during the last merge. If the last merge did not generate any conflicts or warnings, then nothing will be output.

OPTIONS

The following options are available. Note that most options do not apply to all modes.

-A *patterns* Always install the new version of any files that match any of the patterns listed in *patterns*. Each pattern is evaluated as an sh(1) shell pattern. This option may be specified multiple times to specify multiple patterns. Multiple space-separated patterns may also be specified in a single option. Note that ignored files specified via the IGNORE_FILES variable or the **-I** option will not be installed.

-B Do not build generated files in a private object tree. Instead, reuse the generated files from a previously built object tree that matches the source tree. This can be useful to avoid gratuitous conflicts in sendmail(8) configuration files when bootstrapping. It can also be useful for building a tarball that matches a specific world build.

-D *destdir* Specify an alternate destination directory as the target of a merge. This is analogous to the DESTDIR variable used with 'make installworld'. The default destination directory is an empty string which results in merges updating */etc* on the local machine.

-d *workdir* Specify an alternate directory to use as the work directory. The work directory is used to store the "current" and "previous" trees as well as unresolved conflicts. The default work directory is <*destdir*>*/var/db/etcupdate*.

-F Ignore changes in the FreeBSD ID string when comparing files in the destination directory to files in either of the "current" or "previous" trees. In **diff** mode, this reduces noise due to FreeBSD ID string changes in the output. During an update this can simplify handling for harmless conflicts caused by FreeBSD ID string changes.

Specifically, if a file in the destination directory is identical to the same file in the "previous" tree modulo the FreeBSD ID string, then the file is treated as if it was unmodified and the "current" version of the file will be installed. Similarly, if a file in the destination directory is identical to the same file in the "current" tree modulo the FreeBSD ID string, then the "current" version of the file will be installed to update the ID string. If the "previous" and "current" versions of the file are identical, then **etcupdate** will not change the file in the destination directory.

Due to limitations in the diff(1) command, this option may not have an effect if there are other changes in a file that are close to the FreeBSD ID string.

-I *patterns* Ignore any files that match any of the patterns listed in *patterns*. No warnings or other messages will be generated for those files during a merge. Each pattern is evaluated as an sh(1) shell pattern. This option may be specified multiple times to specify multiple

patterns. Multiple space-separated patterns may also be specified in a single option.

-L *logfile* Specify an alternate path for the log file. The **etcupdate** utility logs each command that it invokes along with the standard output and standard error to this file. By default the log file is stored in a file named *log* in the work directory.

-M *options* Pass *options* as additional parameters to make(1) when building a "current" tree. This can be used for to set the TARGET or TARGET_ARCH variables for a cross-build.

-m *make* Use *make* as the make(1) binary when building a "current" tree.

-n Enable "dry-run" mode. Do not merge any changes to the destination directory. Instead, report what actions would be taken during a merge. Note that the existing "current" and "previous" trees will not be changed. If the **-r** option is not specified, then a temporary "current" tree will be extracted to perform the comparison.

-N Perform a NO_ROOT build when building a "current" tree. The resulting tree will include a corresponding *METALOG* file at its root.

-p Enable "pre-world" mode. Only merge changes to files that are necessary to successfully run 'make installworld' or 'make installkernel'. When this flag is enabled, the existing "current" and "previous" trees are left alone. Instead, a temporary tree is populated with the necessary files. This temporary tree is compared against the "current" tree. This allows a normal update to be run after 'make installworld' has completed. Any conflicts generated during a "pre-world" update should be resolved by a "pre-world" **resolve**.

-r Do not update the "current" and "previous" trees during a merge. This can be used to "re-run" a previous merge operation.

-s *source* Specify an alternate source tree to use when building or extracting a "current" tree. The default source tree is */usr/src*.

-t *tarball* Extract a new "current" tree from a tarball previously generated by the **build** command rather than building the tree from a source tree.

CONFIG FILE

The **etcupdate** utility can also be configured by setting variables in an optional configuration file named */etc/etcupdate.conf*. Note that command line options override settings in the configuration file. The configuration file is executed by sh(1), so it uses that syntax to set configuration variables. The following variables can be set:

ALWAYS_INSTALL Always install files that match any of the patterns listed in this variable similar to the **-A** option.

DESTDIR Specify an alternate destination directory similar to the **-D** option.

EDITOR Specify a program to edit merge conflicts.

FREEBSD_ID Ignore changes in the FreeBSD ID string similar to the **-F** option. This is enabled by setting the variable to a non-empty value.

IGNORE_FILES Ignore files that match any of the patterns listed in this variable similar to the **-I** option.

LOGFILE Specify an alternate path for the log file similar to the **-L** option.

MAKE_CMD Specify the make(1) binary when building a "current" tree similar to the **-m** option.

MAKE_OPTIONS Pass additional options to make(1) when building a "current" tree similar to the **-M** option.

SRCDIR Specify an alternate source tree similar to the **-s** option.

WORKDIR Specify an alternate work directory similar to the **-d** option.

ENVIRONMENT

The **etcupdate** utility uses the program identified in the EDITOR environment variable to edit merge conflicts. If EDITOR is not set, vi(1) is used as the default editor.

FILES

/etc/etcupdate.conf Optional config file.
/var/db/etcupdate Default work directory used to store trees and other data.
/var/db/etcupdate/log Default log file.

EXIT STATUS

The **etcupdate** utility exits 0 on success, and >0 if an error occurs.

EXAMPLES

To compare the files in */etc* with the stock versions:

```
etcupdate diff
```

To merge changes after an upgrade via the buildworld and installworld process:

```
etcupdate
```

To resolve any conflicts generated during a merge:

```
etcupdate resolve
```

Bootstrapping

The **etcupdate** utility may need to be bootstrapped before it can be used. The **diff** command will fail with an error about a missing reference tree if bootstrapping is needed.

Bootstrapping **etcupdate** requires a source tree that matches the currently installed world. The easiest way to ensure this is to bootstrap **etcupdate** before updating the source tree to start the next world upgrade cycle. First, generate a reference tree:

```
etcupdate extract
```

Second, use the **diff** command to compare the reference tree to your current files in */etc*. Undesired differences should be removed using an editor, patch(1), or by copying files from the reference tree (located at */var/db/etcupdate/current* by default)

If the tree at */usr/src* is already newer than the currently installed world, a new tree matching the currently installed world can be checked out to a temporary location. The reference tree for **etcupdate** can then be generated via:

```
etcupdate extract -s /path/to/tree
```

The **diff** command can be used as above to remove undesired differences. Afterwards, the changes in the tree at */usr/src* can be merged via a regular merge.

DIAGNOSTICS

The following warning messages may be generated during a merge. Note that several of these warnings cover obscure cases that should occur rarely if at all in practice. For example, if a file changes from a file to a directory in the "current" tree and the file was modified in the destination directory, then a warning will be triggered. In general, when a warning references a pathname, the corresponding file in the destination directory is not changed by a merge operation.

Directory mismatch: <path> (<type>) An attempt was made to create a directory at *path* but an existing file of type "type" already exists for that path name.

Modified link changed: <file> (<old> became <new>) The target of a symbolic link named *file* was changed from "old" to "new" in the "current" tree. The symbolic link has been modified to point to a target that is neither "old" nor "new" in the destination directory.

Modified mismatch: <file> (<new> vs <dest>) A file named *file* of type "new" was modified in the "current" tree, but the file exists as a different type "dest" in the destination directory.

Modified <type> changed: <file> (<old> became <new>) A file named *file* changed type from "old" in the "previous" tree to type "new" in the "current" tree. The file in the destination directory of type "type" has been modified, so it could not be merged automatically.

Modified <type> remains: <file> The file of type "type" named *file* has been removed from the "current" tree, but it has been locally modified. The modified version of the file remains in the destination directory.

Needs update: /etc/localtime (required manual update via tzsetup(8)) The */var/db/zoneinfo* file does not exist, so **etcupdate** was not able to refresh */etc/localtime* from its source file in */usr/share/zoneinfo*. Running tzsetup(8) will both refresh */etc/localtime* and generate */var/db/zoneinfo* permitting future updates to refresh */etc/localtime* automatically.

Needs update: /etc/mail/aliases.db (required manual update via newaliases(1)) The file */etc/mail/aliases* was updated during a merge with a non-empty destination directory. Due to a limitation of the newaliases(1) command, **etcupdate** was not able to automatically update the corresponding aliases database.

New file mismatch: <file> (<new> vs <dest>) A new file named *file* of type "new" has been added to the "current" tree. A file of that name already exists in the destination directory, but it is of a different type "dest".

New link conflict: <file> (<new> vs <dest>) A symbolic link named *file* has been added to the "current" tree that links to "new". A symbolic link of the same name already exists in the destination directory, but it links to a different target "dest".

Non-empty directory remains: <file> The directory *file* was removed from the "current" tree, but it contains additional files in the destination directory. These additional files as well as the directory remain.

Remove mismatch: <file> (<old> became <new>) A file named *file* changed from type "old" in the "previous" tree to type "new" in the "current" tree, but it has been removed in the destination directory.

Removed file changed: <file> A file named *file* was modified in the "current" tree, but it has been removed in the destination directory.

Removed link changed: <file> (<old> became <new>) The target of a symbolic link named *file* was changed from "old" to "new" in the "current" tree, but it has been removed in the destination directory.

SEE ALSO

cap_mkdb(1), diff(1), make(1), newaliases(1), sh(1), mergemaster(8), pwd_mkdb(8), services_mkdb(8), tzsetup(8)

HISTORY

The **etcupdate** utility first appeared in FreeBSD 10.0.

AUTHORS

The **etcupdate** utility was written by John Baldwin <*jhb@FreeBSD.org*>.

BUGS

Rerunning a merge does not automatically delete conflicts left over from a previous merge. Any conflicts must be resolved before the merge can be rerun. It is not clear if this is a feature or a bug.

There is no way to easily automate conflict resolution for specific files. For example, one can imagine a syntax along the lines of

```
etcupdate resolve tf /some/file
```

to resolve a specific conflict in an automated fashion.

Bootstrapping **etcupdate** often results in gratuitous diffs in */etc/mail/*.cf* that cause conflicts in the first merge. If an object tree that matches the source tree is present when bootstrapping, then passing the **-B** flag to the **extract** command can work around this.