NAME

ucf − Update Configuration File: preserve user changes in configuration files

SYNOPSIS

ucf [*options*] *<New File> <Destination>*

ucf [*options*] *−−purge <Destination>*

DESCRIPTION

This utility provides a means of asking the user whether or not to accept new versions of configuration files provided by the package maintainer, with various heuristics designed to minimize interaction time. It uses debconf to interact with the user, as per Debian policy. In the SYNOPSIS above, *New file* is the configuration file as provided by the package (either shipped with the package, or generated by the maintainer scripts on the fly), and *Destination* is the location (usually under /etc) where the real configuration file lives, and is potentially modified by the end user. Since the files edited would be real files, and not symbolic links, **ucf** follows and resolves symbolic links before acting. As far as possible, ucf attempts to preserve the ownership and permission of the *New file* as it is copied to the new location.

This script attempts to provide conffile like handling for files installed under */etc* not shipped in a **Debian** package, but handled by the postinst instead. **Debian** policy states that files under */etc* which are configuration files **must** preserve user changes, and this applies to files handled by maintainer scripts as well. Using **ucf,** one may ship a bunch of default configuration files somewhere in */usr* (*/usr/share/<pkg>* is a good location), and maintain files in */etc,* preserving user changes and in general offering the same facilities while upgrading that **dpkg** normally provides for *"conffiles"*

Additionally, this script provides facilities for transitioning a file that had not been provided *conffile* like protection to come under this schema, and attempts to minimize questions asked at install time. Indeed, the transitioning facility is better than the one offered by **dpkg** while transitioning a file from a *non−conffile* to *conffile* status. The second form in the SYNOPSIS above is for purging information about the configuration file when the package is purged; and is critical for allowing smooth reinstallations.

During the course of operations, when working with configuration files, **ucf** optionally creates copies of versions of the configuration file in question. For example, a file with the suffix *ucf-old* holds the old version of a configuration file replaced by **ucf.** Also, copies of the configuration file with the suffixes *ucf-new* and *ucf-dist* may be created; and the maintainer scripts should consider purging copies of the configuration file with these extensions during purge.

OPTIONS

−h, −−help
: Print a short usage message

−n, −−no−action
: Dry run. Print the actions that would be taken if the script is invoked, but take no action.

−d[n], −−debug=[n]
: Set the debug level to the (optional) level *n* (n defaults to 1). Please note there must be no spaces before the optional digit n. This turns on copious debugging information.

−p, −−purge
: Removes all vestiges of the file from the state hashfile. This is required to allow a package to be reinstalled after it is purged; since otherwise, the real configuration file is removed, but it remains in the hash file; and on reinstall no action is taken, since the md5sum of the new file matches that in the hashfile. In short, remember to use this option in the postrm for every configuration file managed by ucf when the package is being purged (assuming ucf itself exists). *Note:* ucf does not actually touch the file on disk in this operation, so any file removals are still the responsibility of the calling package.

−v, −−verbose
: Make the script be very verbose about setting internal variables.

−s foo, −−src−dir foo
Set the source directory (historical md5sums are expected to live in files and sub directories of this directory) to foo. By default, the directory the new_file lives in is assumed to be the source directory. Setting this option overrides settings in the environment variable **UCF_SOURCE_DIR,** and in the configuration file variable **conf_source_dir.**

−−sum−file foo
Force the historical md5sums to be read from this file, rather than defaulting to living in the source directory. Setting this option overrides settings in the environment variable **UCF_OLD_MDSUM_FILE,** and in the configuration file variable **conf_old_mdsum_file.**

−−three−way
This turns on the option, during installation, for the user to be offered a chance to see a merge of the changes between old maintainer version and the new maintainer version into the local copy of the configuration file. If the user likes what they see, they can ask to have these changes merged in. This allows one to get new upstream changes merged in even while retaining local modifications to the configuration file. This is accomplished by taking the configuration file and stashing it in a cache area during registration, and using diff3 during the install (the stashed file name is a munged version of the full path of the configuration file to avoid name space clashes).

−−debconf−ok
Indicate that it is ok for *ucf* to use an already running debconf instance for prompting (it has always been ok to use ucf when debconf is not running -- it shall invoke debconf as needed).

−−debconf−template foo
Instruct ucf to use the named multiselect debconf template instead of the normal ucf-provided debconf template. The caller is responsible for ensuring that the named template exists and has a list of choices matching those for the default ucf template, and should set Choices−C: ${CHOICES} to ensure the returned values match those from the default template. Note that the choices must be different according to whether the **−−three−way** option is also set.

−−state−dir /path/to/dir
Set the state directory to /path/to/dir instead of the default */var/lib/ucf.* Used mostly for testing.

USAGE

The most common case usage is pretty simple: a single line invocation in the postinst on configure, and another single line in the postrm to tell **ucf** to forget about the configuration file on purge (using the −−purge option) is all that is needed (assuming ucf is still on the system).

It is recommended that you also register any file being managed by **ucf** with the ucf registry; this associates the configuration file with the package it belongs to. This is done with a simple call to **ucfr.** Users may then query the association between a configuration file and the package using the tool **ucfq.** Please see the appropriate manual pages for details.

If a file maintained by maintainer scripts is being transitioned from an unprotected status to the protection afforded by the script, the maintainer can help ease the transition by reducing the questions that may be asked at installation time. Specifically, questions should not be asked if the file in question is an unmodified version that was one shipped in a previous version of this package; and the maintainer can help by telling the script about the historical md5sums that published versions of this file contained.

The way to do this is to either create a file called **<New file>.md5sum,** with one md5sum on each line, (the file names you use are ignored, except for the entry named default), or create a directory, called **<New file>.md5sum.d,** which should contain any number of files, each containing a single line, namely, the md5sum of a previous version of **<New file>.** The names of these files are not important, with one exception: The file called default is treated specially. For example, the author personally uses either package version numbers or release code names, like *7.6.3,* or *potato.* If none of the historical md5sums match, we are almost certain that either the historical record of md5sums is not complete, or the user has changed the configuration file.

The default historical md5sum

The exception to the rule about names mentioned earlier is that if no md5sums match, and if the file **<New file>.md5sum.d/default** exists, or if there is a line corresponding to a *default* file in **<New file>.md5sum,** it shall be used as the default md5sum of the *previous* version of the package assumed to have been installed on this machine. As you can see, unless there are limited number of previously released packages (like just one), the maintainer is also making an informed guess, but the option is provided to the maintainer.

If the file **<New file>.md5sum,** or the directory **<New file>.md5sum.d** does not exist, or none of the md5sums match, we test the installed *<Destination>* file to see whether it is the same as the *<New file>*. If not, we ask the user whether they want us to replace the file.

An additional facility is also offered: optionally, ucf can store one old version of the maintainers copy of the configuration file, and, on upgrade, calculate the changes made in the maintainers version of the configuration file, and apply that patch to the local version of the file (on user request, of course). There is also a preview facility where the user can inspect the results of such a merge, before asking the action to be taken.

ENVIRONMENT VARIABLES

The variable **UCF_FORCE_CONFFNEW,** if set, forces the new file to always overwrite the installed destination file, while the variable **UCF_FORCE_CONFFOLD,** if set silently retains the installed file. **UCF_FORCE_CONFFMISS** is only applicable when the installed destination file does not exist (perhaps due to user removal),and forces ucf to recreate the missing file (the default behaviour is to honor the users wishes and not recreate the locally deleted file).

FILES

This script creates the file *new_file.md5sum,* and it may copy the file (presumably shipped with the package) *<New file>* to its destination, *<Destination>.*

/var/lib/ucf/hashfile, and */var/lib/ucf/hashfile.X,* where *X* is a small integer, where previous versions of the hashfile are stored.

/etc/ucf.conf

EXAMPLES

If the package *foo* wants to use ucf to handle user interaction for configuration file *foo.conf,* a version of which is provided in the package as */usr/share/foo/configuration,* a simple invocation of ucf in the post inst file is all that is needed:

ucf */usr/share/foo/configuration /etc/foo.conf*

On purge, one should tell ucf to forget about the file (see detailed examples in /usr/share/doc/ucf/examples):

ucf *−−purge /etc/foo.conf* Please note that purge can also be used to make ucf forget the previous state of the files, and when the package is next installed or updated, ucf will ask the user to replace the current cofiguration file. Do this if you want to change your decision to not update to a maintainer provided version of the configuration file.

The motivation for this script was to provide conffile like handling for start files for emacs lisp packages (for example, */etc/emacs21/site−start.d/50psgml−init.el*) These start files are not shipped with the package, instead, they are installed during the post installation configuration phase by the script */usr/lib/emacsen−common/emacs−package−install $package_name.*

This script is meant to be invoked by the packages install script at */usr/lib/emacsen−common/packages/install/$package_name* for each flavour of installed emacsen by calling it with the proper values of new file (*/usr/share/emacs/site−lisp/<pkg>/<pkg−init.el*), and dest file (*/etc/emacs21/site−start.d/50<pkg−init.el*), and it should do the rest.

SEE ALSO

ucf.conf(5), ucfr(1), ucfq(1), and diff3(1). The **Debian** Emacs policy, shipped with the package *emacsen−common.*

AUTHOR

This manual page was written Manoj Srivastava <srivasta@debian.org>, for the Debian GNU/Linux system.