NAME sccs – front end for the Source Code Control System (SCCS)

SYNOPSIS **/usr/ccs/bin/sccs** [**–r**] [**–d** *rootprefix*] [**–p** *subdir*] *subcommand* [*option* . . .] [*file* . . .]

AVAILABILITY SUNWsprot

DESCRIPTION

The **sccs** command is a comprehensive, straightforward front end to the various utility programs of the Source Code Control System (SCCS).

sccs applies the indicated *subcommand* to the history file associated with each of the indicated files.

The name of an SCCS history file is derived by prepending the '**s.**' prefix to the filename of a working copy. The **sccs** command normally expects these '**s.**files' to reside in an **SCCS** subdirectory. Thus, when you supply **sccs** with a *file* argument, it normally applies the subcommand to a file named **s.***file* in the **SCCS** subdirectory. If *file* is a path name, **sccs** looks for the history file in the **SCCS** subdirectory of that file's parent directory. If *file* is a directory, however, **sccs** applies the subcommand to every **s.**file file it contains. Thus, the command:

example% sccs get program.c

would apply the **get** subcommand to a history file named:

SCCS/s.program.c

while the command:

example% sccs get SCCS

would apply it to every **s.**file in the **SCCS** subdirectory.

Options for the **sccs** command itself must appear before the *subcommand* argument. Options for a given subcommand must appear after the *subcommand* argument. These options are specific to each subcommand, and are described along with the subcommands themselves (see **Subcommands**, below).

Running Setuid

The **sccs** command also includes the capability to run ''setuid'' to provide additional protection. However this does not apply to subcommands such as **sccs-admin**(1), since this would allow anyone to change the authorizations of the history file. Commands that would do so always run as the real user.

OPTIONS

The following options are supported:

–d *rootprefix* Define the root portion of the path name for SCCS history files. The default root portion is the current directory. Note: *rootprefix* is prepended to the entire *file* argument, even if *file* is an absolute path name. **–d** overrides any directory specified by the **PROJECTDIR** environment variable (see **ENVIRONMENT**, below).

–p *subdir* Define the (sub)directory within which a history file is expected to reside. **SCCS** is the default. (See **EXAMPLES**, below).

−r Run **sccs** with the real user ID, rather than set to the effective user ID.

OPERANDS

The following operands are supported:

subcommand An SCCS utility name or the name of one of the pseudo-utilities listed in **USAGE**.

options An option or option-argument to be passed to *subcommand*.

operands An operand to be passed to *subcommand*.

USAGE

Subcommands

Many of the following **sccs** subcommands invoke programs that reside in **/usr/ccs/bin**. Many of these subcommands accept additional arguments that are documented in the reference page for the utility program the subcommand invokes.

admin Modify the flags or checksum of an SCCS history file. Refer to **sccs-admin**(1) for more information about the **admin** utility. While **admin** can be used to initialize a history file, you may find that the **create** subcommand is simpler to use for this purpose.

cdc **−r** *sid* [**−y** [*comment*]]
cdc **−r***sid* [**−y** [*comment*]]

Annotate (change) the delta commentary. Refer to **sccs-cdc**(1). Note: The **fix** subcommand can be used to replace the delta, rather than merely annotating the existing commentary.

−r *sid* | **−r***sid* Specify the SCCS delta ID (SID) to which the change notation is to be added. The SID for a given delta is a number, in Dewey decimal format, composed of two or four fields: the *release* and *level* fields, and for branch deltas, the *branch* and *sequence* fields. For instance, the SID for the initial delta is normally **1.1**.

−y [*comment*] Specify the comment with which to annotate the delta commentary. If **−y** is omitted, **sccs** prompts for a comment. A null *comment* results in an empty annotation.

check [**−b**] [**−u** [*username*] | **−U**]

Check for files currently being edited. Like **info** and **tell**, but returns an exit code, rather than producing a listing of files. **check** returns a non-zero exit status if anything is being edited.

−b Ignore branches.

−u [*username*] | **−U**
Only check files being edited by you. When *username* is specified, only check files being edited by that user.

clean [**−b**]

Remove everything in the current directory that can be retrieved from an SCCS history. Does not remove files that are being edited.

−b Do not check branches to see if they are being edited.

'**clean** −**b**' is dangerous when branch versions are kept in the same directory.

comb Generate scripts to combine deltas. Refer to **sccs-comb**(1).

create Create (initialize) history files. **create** performs the following steps:

- Renames the original source file to **,program.c** in the current directory.
- Create the history file called **s.program.c** in the **SCCS** subdirectory.
- Performs an '**sccs get**' on **program.c** to retrieve a read-only copy of the initial version.

deledit [−**s**] [−**y** [*comment*]]

Equivalent to an '**sccs delta**' and then an '**sccs edit**'. **deledit** checks in a delta, and checks the file back out again, but leaves the current working copy of the file intact.

−**s** Silent. Do not report delta numbers or statistics.

−**y** [*comment*] Supply a comment for the delta commentary. If −**y** is omitted, **delta** prompts for a comment. A NULL *comment* results in an empty comment field for the delta.

delget [−**s**] [−**y** [*comment*]]

Perform an '**sccs delta**' and then an '**sccs get**' to check in a delta and retrieve read-only copies of the resulting new version. See the **deledit** subcommand for a description of −**s** and −**y**. **sccs** performs a **delta** on all the files specified in the argument list, and then a **get** on all the files. If an error occurs during the **delta**, the **get** is not performed.

delta [−**s**] [−**y** [*comment*]]

Check in pending changes. Records the line-by-line changes introduced while the file was checked out. The effective user ID must be the same as the ID of the person who has the file checked out. Refer to **sccs-delta**(1). See the **deledit** subcommand for a description of −**s** and −**y**.

diffs [−**C**] [−**c** *date-time*] [−**r** *sid*] *diff-options*
diffs [−**C**] [−**c***date-time*] [−**r***sid*] *diff-options*

Compare (in **diff**(1) format) the working copy of a file that is checked out for editing, with a version from the SCCS history. Use the most recent checked-in version by default. The **diffs** subcommand accepts the same options as **diff**, with the exception that the −**c** option to **diff** must be specified as −**C**.

−**C** Pass the −**c** option to **diff**.

−**c** *date-time* | −**c***date-time*

Use the most recent version checked in before the indicated date and time for comparison. *date-time* takes the form: *yy*[*mm*[*dd*[*hh*[*mm*[*ss*]]]]]. Omitted units default to their maximum possible values; that is −**c7502** is equivalent to −**c750228235959**.

−r *sid* | **−r***sid* Use the version corresponding to the indicated delta for comparison.

edit Retrieve a version of the file for editing. '**sccs edit**' extracts a version of the file that is writable by you, and creates a **p.**file in the **SCCS** subdirectory as lock on the history, so that no one else can check that version in or out. ID keywords are retrieved in unexpanded form. **edit** accepts the same options as **get**, below.

enter Similar to **create**, but omits the final '**sccs get**'. This may be used if an '**sccs edit**' is to be performed immediately after the history file is initialized.

fix **−r** *sid*
fix **−r***sid* Revise a (leaf) delta. Remove the indicated delta from the SCCS history, but leave a working copy of the current version in the directory. This is useful for incorporating trivial updates for which no audit record is needed, or for revising the delta commentary. **fix** must be followed by a **−r** option, to specify the SID of the delta to remove. The indicated delta must be the most recent (leaf) delta in its branch. Use **fix** with caution since it does not leave an audit trail of differences (although the previous commentary is retained within the history file).

get [**−ekmps**] [**−c** *date-time*] [**−r** *sid*]
get [**−ekmps**] [**−c***date-time*] [**−r***sid*]

Retrieve a version from the SCCS history. By default, this is a read-only working copy of the most recent version; ID keywords are in expanded form. Refer to **sccs-get**(1).

−e Retrieve a version for editing. Same as **sccs edit**.

−k Retrieve a writable copy but do not check out the file. ID keywords are unexpanded.

−m Precede each line with the SID of the delta in which it was added.

−p Produce the retrieved version on the standard output. Reports that would normally go to the standard output (delta ID's and statistics) are directed to the standard error.

−s Silent. Do not report version numbers or statistics.

−c *date-time* | **−c***date-time*
Retrieve the latest version checked in prior to the date and time indicated by the *date-time* argument. *date-time* takes the form: *yy*[*mm*[*dd*[*hh*[*mm*[*ss*]]]]].

−r *sid* | **−r***sid* Retrieve the version corresponding to the indicated SID.

help *message-code* | *sccs-command*
help stuck

Supply more information about SCCS diagnostics. **help** displays a brief explanation of the error when you supply the code displayed by an SCCS diagnostic message. If you supply the name of an SCCS command, it prints a usage line. **help** also recognizes the keyword **stuck**. Refer to **sccs-help**(1).

info [–**b**] [–**u** [*username*] | –**U**]
Display a list of files being edited, including the version number checked out, the version to be checked in, the name of the user who holds the lock, and the date and time the file was checked out.

–**b** Ignore branches.

–**u** [*username*] | –**U**
List only files checked out by you. When *username* is specified, only list files checked out by that user.

print Print the entire history of each named file. Equivalent to an '**sccs prs** –**e**' followed by an '**sccs get** –**p** –**m**'.

prs [–**el**] [–**c** *date-time*] [–**r** *sid*]
prs [–**el**] [–**c***date-time*] [–**r***sid*]
Peruse (display) the delta table, or other portion of an **s.**file. Refer to **sccs-prs**(1).

–**e** Display delta table information for all deltas earlier than the one specified with –**r** (or all deltas if none is specified).

–**l** Display information for all deltas later than, and including, that specified by –**c** or –**r**.

–**c** *date-time* | –**c***date-time*
Specify the latest delta checked in before the indicated date and time. The *date-time* argument takes the form: *yy*[*mm*[*dd*[*hh*[*mm*[*ss*]]]]].

–**r** *sid* | –**r***sid* Specify a given delta by SID.

prt [–**y**]
Display the delta table, but omit the MR field (see **sccsfile**(4) for more information on this field). Refer to **sccs-prt**(1).

–**y** Display the most recent delta table entry. The format is a single output line for each file argument, which is convenient for use in a pipeline with **awk**(1) or **sed**(1).

rmdel –**r** *sid*
rmdel –**r***sid* Remove the indicated delta from the history file. That delta must be the most recent (leaf) delta in its branch. Refer to **sccs-rmdel**(1).

sccsdiff –**r***old-sid* –**r***new-sid diff-options*
Compare two versions corresponding to the indicated SIDs (deltas) using **diff**. Refer to **sccs-sccsdiff**(1).

tell [–**b**] [–**u** [*username*] | –**U**]
Display the list of files that are currently checked out, one file per line.

Ignore branches.

–**u** [*username*] | –**U**
Only list files checked out to you. When *username* is specified, only list files check out to that user.

unedit "Undo" the last **edit** or '**get** –**e**', and return the working copy to its previous condition. **unedit** backs out all pending changes made since the file was checked out.

unget Same as **unedit**. Refer to **sccs-unget**(1).

val Validate the history file. Refer to **sccs-val**(1).

what Display any expanded ID keyword strings contained in a binary (object) or text file. Refer to **what**(1) for more information.

EXAMPLES

sccs converts the command:

```
example% sccs –d/usr/src/include get stdio.h
```

to:

```
/usr/ccs/bin/get /usr/src/include/SCCS/s.stdio.h
```

The command:

```
example% sccs –pprivate get include/stdio.h
```

becomes:

```
/usr/ccs/bin/get include/private/s.stdio.h
```

To initialize the history file for a source file named **program.c**: make the SCCS subdirectory, and then use '**sccs create**':

```
example% mkdir SCCS
example% sccs create program.c
program.c:
1.1
14
```

After verifying the working copy, you can remove the backup file that starts with a comma:

```
example% diff program.c ,program.c
example% rm ,program.c
```

To check out a copy of **program.c** for editing, edit it, and then check it back in:

```
example% sccs edit program.c
1.1
new delta 1.2
14
example% vi program.c
your editing session
```

```
example% sccs delget program.c
comments? clarified cryptic diagnostic
1.2
3 inserted
2 deleted
12 unchanged
1.2
15
```

To retrieve a file from another directory into the current directory:

example% sccs get /usr/src/sccs/cc.c

or:

example% sccs −p/usr/src/sccs/ get cc.c

To check out all files under SCCS in the current directory:

example% sccs edit SCCS

To check in all files currently checked out to you:

example% sccs delta \`sccs tell −u\`

ENVIRONMENT

See **environ**(5) for descriptions of the following environment variables that affect the execution of **sccs**: **LC_CTYPE**, **LC_MESSAGES**, and **NLSPATH**.

PROJECTDIR If contains an absolute path name (beginning with a slash), **sccs** searches for SCCS history files in the directory given by that variable.

If **PROJECTDIR** does not begin with a slash, it is taken as the name of a user, and **sccs** searches the **src** or **source** subdirectory of that user's home directory for history files. If such a directory is found, it is used. Otherwise, the value is used as a relative path name.

EXIT STATUS

The following exit values are returned:

0 Successful completion.

>0 An error occurred.

FILES

SCCS	SCCS subdirectory
SCCS/d.*file*	temporary file of differences
SCCS/p.*file*	lock (permissions) file for checked-out versions
SCCS/q.*file*	temporary file
SCCS/s.*file*	SCCS history file
SCCS/x.*file*	temporary copy of the **s.**file
SCCS/z.*file*	temporary lock file
/usr/ccs/bin/∗	SCCS utility programs

SEE ALSO

awk(1), **diff**(1), **sccs-admin**(1), **sccs-cdc**(1), **sccs-comb**(1), **sccs-delta**(1), **sccs-get**(1), **sccs-help**(1), **sccs-prs**(1), **sccs-rmdel**(1), **sccs-sact**(1), **sccs-sccsdiff**(1), **sccs-unget**(1), **sccs-val**(1), **sed**(1), **what**(1), **sccsfile**(4)

Programming Utilities Guide

BUGS

There is no **sact** subcommand to invoke **/usr/ccs/bin/sact** (see **sccs-sact**(1)). However, the **info** subcommand performs an equivalent function.