SCVS Cheat Sheet

November 3, 1991

The SCVS Cheat Sheet explains some of the standard uses of **scvs**. If you want to do something that is not covered here you should refer to the man pages for **scvs**, **cvs**, and **rcs**. **Scvs** manages source directory trees called *modules*. **Scvs** modules for the kernel correspond to what we've traditionally called modules, e.g. net, timer, mach, etc. The "master" copy of the modules is stored in the *repository*, which is a directory tree of RCS files rooted at `/sprite/src/kernel/Cvsroot`. In order to modify the source for a module you must first get your own copy of the module. Make your changes in your copy, and when you are satisfied with them commit your changes to the repository. Your changes are not applied to the repository until you commit them, at which time they become the "latest" version of the sources. Throughout this cheat sheet the term "latest version" refers to the current version of the sources in the repository.

How do I get a copy of a module?

Copies of kernel modules are usually kept in your kernel build directory (`/sprite/src/kernel/`*username*). In this directory type **scvs co** *module*. A subdirectory called *module* will be created, and it will be populated with the current version of the module. You will be told about other users that have a copy of the module checked out, including any other copies you might have. You may want to check with these people to make sure that your changes will be compatible. If you don't intend to commit your changes to the repository then you should use the **−i** option so that other users are not told about your copy.

After the checkout is complete you need to run **mkmf** in your copy to create a Makefile, dependencies.mk, etc.

It is ok to re-checkout a module if you checked it out before and still have the source tree.

How do I get information about the status of my copy of a module?

Once you have a copy of a module you may want some information about the files in your copy, like which ones you've modified and which ones are out-of-date with the latest version. **scvs info** will tell you what you want to know. Here's what its output means:

U *file*	Your copy of *file* needs to be updated.
M *file*	You've modified *file*.
C *file*	You've modified *file*, and it is also out-of-date.
A *file*	You've added *file*.
R *file*	You've removed *file*.
D *file*	Somebody deleted *file* from the repository.

What if I want more information about my copy of a file?

The command **scvs status** *file* will give you RCS information about the your copy of a file and the latest version of the file. Here is some sample output for the file *timerInt.h*.

File:	timerInt.h	
From:	9.8	Fri Sep 13 15:04:52 1991 timerInt.h
RCS:	9.9	/sprite/src/kernel/Cvsroot/kernel/timer/timerInt.h,v

The *From:* line tells you that your copy of *timerInt.h* came from RCS version 9.8 of that file, and that you created the copy on Friday, September 13, 1991 at 15:04:52. The *RCS:* line tells you that the latest RCS version of the file is 9.9, and it gives you the full path to the RCS file, not that you would ever need it.

How do I get the RCS log for my copy of a file?

Scvs log *file* will print out the RCS log for the file.

How do I run diff on my copy of a file?

Scvs diff *file* will do a **rcsdiff** between your copy of a file and the version from which it came. If you want to do a diff between your version and the latest version you should do **scvs diff −R** *file*. You can also do diffs between various versions of the file by specifying the standard **−r** options.

How do I bring my copy up-to-date with the latest version?

If your copy of the sources is out-of-date with the latest version in the repository you can update it using **scvs update** [*modules*|*subdirs*|*files*]. If you don't specify any files it will update all files in the current directory and its subdirectories, otherwise it will only update the files or subdirectories you listed. If **scvs update** discovers that you have modified a file, and the copy of the file in the repository has changed also, it will use **rcsmerge** to try and merge the changes. If this happens you will want to look at the file to make sure the changes were compatible. The output from **scvs update** is similar to that of **scvs info**:

U *file*	Your copy of *file* was updated.
M *file*	Your changes to *file* were merged with the new version.
C *file*	Your changes to *file* were merged, but there was a conflict.
A *file*	You've added *file*.
R *file*	You've removed *file*.
D *file*	*file* was deleted from your sources.

An "update" target has been added to the Makefile for kernel modules so that **pmake update** will run **scvs update**. This is useful for updating all modules in `/sprite/src/kernel`.

Who else has a copy of the same module?

When you check out a module you will be told about other users who also have a copy. You can also find out by doing **scvs who** [*modules*].

What if I just want to look at the sources for a module?

Copies of all kernel modules are kept in `/sprite/src/kernel`. These copies are read-only so don't try to modify them. These copies can be used for browsing, as well as for building the object files that are used to produce official kernels.

How do I add a new file to a module?

Scvs only deals with files it knows about. If **scvs** hasn't been told about a file it will be ignored by any **scvs** commands. To add a file to a module create the file in your copy of the module, then type **scvs add** *file*. The file must exist in order to be added. The file will not be added to the repository until you **commit** your changes.

How do I add a new subdirectory to a module?

Subdirectories are added using **scvs add** in your copy of a module just like files, except that you will be asked if you really want to do this. If the subdirectory contains any files they must be added individually (**add** is not recursive).

How do I remove a file from a module?

In your copy of a module type **scvs remove** *file* to remove a file. If the file still exists it will be deleted. The file will not be removed from the repository until you **commit** your changes.

Oops. I just accidently removed a file from my copy!

If you haven't **commit**ted your changes yet, you can retrieve the file using **scvs add** *file*. It will give you the version of the file you originally checked out.

How do I add a new module to the repository?

Don't try this at home. Send mail to jhh@sprite.

How do I undo my changes to a file?

Currently the best way to do this is to use **scvs remove** *file*, followed by **scvs add** *file*. This will give you the version of the file you originally checked out.

How do I commit my changes to a module?

After you have made your changes to a module you need to commit them into the repository. There are several steps in committing your changes:

(1) Verify that your copy is up-to-date and it works properly.

(2) Lock the module(s) using **scvs lock** [*modules*]. Locking the modules prevents other users from committing changes at the same time.

(3) Use **scvs commit** [*files*] (**commit** is usually abbreviated as **ci**) to commit your changes. Each **scvs ci** command allows you to specify one log message (either using **−m** or by bringing up an editor). If you want all the files you modified to have the same message use **scvs ci** without any options, otherwise you must run **scvs ci** once for each message and corresponding list of files.

(4) As you commit your changes the copy of the sources in `/sprite/src/kernel/`*module* will be updated. These copies are used to build kernels, so you need to recompile any source files you may have changed. To do this cd to `/sprite/src/kernel/`*module*. Run **mkmf**, followed by **pmake**. Make sure the resulting object files produce a working kernel. Then run **pmake install**. Be sure to compile and install for all machine types. If the sources don't build a working kernel then you may have forgotten to **add** or **remove** a file.

(5) Unlock the modules using **scvs unlock** [*modules*]. Don't forget to do this.

What do I do when I'm done with my copy?

scvs done lets **scvs** know that you are done with your copy, so that other users will no longer be told about it. The **−d** option will delete your copy. Use the **−d** option with care. If you deleted your copy using **rm** without running **scvs done** first then **scvs** will think you still have a copy. You set it straight with **scvs done** *pathname*, where *pathname* is the full pathname of the copy you used to have.

How do I unlock a locked module?

If **svcs** is killed unexpectedly it will leave locks on any modules it was processing at the time. If you have an unwanted lock on a module you can remove it with **scvs unlock** [*modules*]. If you want to remove all locks on a module, even those owned by other people, you can do so with **scvs unlock -a** [*modules*].