What is a port

The Ports Collection is essentially a set of Makefiles, patches, and description files placed in /usr/ports.
The port includes instructions on how to build source code, but does not include the actual source code. You can get the source code from a CD-ROM or from the Internet. Source code is distributed in whatever manner the software author desires. Frequently this is a tarred and gzipped file, but it might be compressed with some other tool or even uncompressed. The program source code, whatever form it comes in, is called a ``distfile''
A port skeleton is a minimal set of files that tell your FreeBSD system how to cleanly compile and install a program. Each port skeleton includes:

•A Makefile. The Makefile contains various statements that specify how the application should be compiled and where it should be installed on your system.

A distinfo file. This file contains information about the files that must be downloaded to build the port and their checksums, to verify that files have not been corrupted during the download using md5

A files directory. This directory contains patches to make the program compile and install on your FreeBSD system. Patches are basically small files that specify changes to particular files. They are in plain text format, and basically say ``Remove line 10'' or ``Change line 26 to this ...''. Patches are also known as ``diffs'' because they are generated by the diff program.

This directory may also contain other files used to build the port.

A pkg-descr file. This is a more detailed, often multiple-line, description of the program.

A pkg-plist file. This is a list of all the files that will be installed by the port. It also tells the ports system what files to remove upon

Installing using Ports

In our example we are going to install the lsof package which is a nice systems administration utility that does not come with FreeBSD by default.
However before we do this we need to make a small change to our /etc/make.conf.
As root type

```
vi /etc/make.conf
```

We are going to insert a line in here stating thus:

```
MASTER_SITE_OVERRIDE=ftp://noc.ws.afnog.org/pub/FreeBSD/ports/distfiles/
```

In some rare cases, users may need to acquire the tarballs from a site other than the MASTER_SITES (the location where files are downloaded from). In our case, we are doing this so that we may download the package from our local ftp server rather than from the internet. You can override the MASTER_SITES option by editing /etc/make.conf as we have done or on the command line with the following example command:

```
# cd /usr/ports/sysutils/lsof (this should be changed to
the particular port you're installing at the time)
# make MASTER_SITE_OVERRIDE=ftp://noc.ws.afnog.org/pub/FreeBSD/ports/distfiles/
```

In this example we change the MASTER_SITES option to noc.ws.afnog.org/pub/FreeBSD/ports/distfiles/.

Thus ensuring that the make command will attempt to download the file from our local server first rather than the internet. This is useful if you have many machines on your network and you want to preserve bandwidth by having one master server contain all the files and the rest download from this one machine.

The first method of editing /etc/make.conf is useful in that It makes the change persistent while doing it on the command line implies tat you have to add this option every time you try to install a port.

Installing the lsof port

Now that we have edited our /etc/make.conf, let us install the lsof port:

```
cd /usr/ports/sysutils/lsof
make
```

This should show you output similar to:

```
===>  Vulnerability check disabled, database not found
=> lsof_4.75C.freebsd.tar.bz2 doesn't seem to exist in /usr/ports/distfiles/.
=> Attempting to fetch from ftp://noc.ws.afnog.org/pub/FreeBSD/distfiles/.
lsof_4.76.freebsd.tar.bz2                    100% of  437 kB 1429  Bps 00m00s
===>  Extracting for lsof-4.76
=> Checksum OK for lsof_4.76.freebsd.tar.bz2.
===>  Patching for lsof-4.76
===>  Applying FreeBSD patches for lsof-4.76
===>  Configuring for lsof-4.76
```

Notice that once the compile is complete you are returned to your prompt. The next step is to install the port. In order to install it, you simply need to tack one word onto the make command, and that word is install:

```
noc# make install
===>  Installing for lsof-4.74.2
===>   Generating temporary packing list
===>  Checking if sysutils/lsof already installed
install  -s -o root -m 2755 -g kmem
/usr/ports/sysutils/lsof/work/lsof_4.75C.freebsd/lsof /usr/local/sbin
install  -o root -g wheel -m 444
/usr/ports/sysutils/lsof/work/lsof_4.75C.freebsd/lsof.8 /usr/local/man/man8/lsof.8
install  -o root -g wheel -m 555
/usr/ports/sysutils/lsof/work/lsof_4.75C.freebsd/scripts/* /usr/local/share/lsof
cd /usr/local &&  /usr/bin/find -s share/lsof -type f -o -type l
>>/usr/ports/sysutils/lsof/work/.PLIST.mktmp ; /usr/bin/find -d share/lsof -type d
 | /usr/bin/sed -e 's/^/@dirrm /g' >>/usr/ports/sysutils/lsof/work/.PLIST.mktmp
===>   Compressing manual pages for lsof-4.74.2
===>   Registering installation for lsof-4.74.2
===> SECURITY REPORT:
      This port has installed the following binaries which execute with
      increased privileges.
/usr/local/sbin/lsof
      If there are vulnerabilities in these programs there may be a security
      risk to the system. FreeBSD makes no guarantee about the security of
      ports included in the Ports Collection. Please type 'make deinstall'
      to deinstall the port if this is a concern.
      For more information, and contact details about the security
      status of this software, see the following webpage:
http://people.freebsd.org/~abe/
```

Once you are returned to your prompt, you should be able to run the application you just installed. Since lsof is a program that runs with increased privileges, a security warning is shown. During the building and installation of ports, you should take heed of any other warnings that may appear.

Note: You can save an extra step by just running make install instead of make and make install as two separate steps.

Note: Some shells keep a cache of the commands that are available in the directories listed in the PATH environment variable, to speed up lookup operations for the executable file of these commands. If you are using one of these shells, you might have to use the rehash command after installing a port, before the newly installed commands can be used. This command will work for shells like tcsh. Use the hash -r command for shells like sh or bash. Look at documentation for your shell for more information.

The ports system uses fetch to download the files, which honors various environment variables, including FTP_PASSIVE_MODE, FTP_PROXY, and FTP_PASSWORD. You may need to set one or more of these if you are behind a firewall, or need to use an FTP/HTTP proxy.

For users which cannot be connected all the time, the make fetch option is provided. Just run this command at the top level directory (/usr/ports) and the required files will be downloaded for you. This command will also work in the lower level categories, for example: /usr/ports/net. Note that if a port depends on libraries or other ports this will not fetch the distfiles of those ports too. Replace fetch with fetch-recursive if you want to fetch all the dependencies of a port too.

Note: You can build all the ports in a category or as a whole by running make in the top level directory, just like the aforementioned make fetch method. This is dangerous, however, as some ports cannot co-exist. In other cases, some ports can install two different files with the same filename.

Using the Ports Collection will use up disk space over time. Because of this tendency of the ports tree to grow in size, after building and installing software from the ports, you should always remember to clean up the temporary work directories using the make clean command. This will remove the work directory after a port has been built and installed. You can also remove the source distribution files from the distfiles directory, and remove the installed ports if the need for them has passed.

Removing Installed Ports

```
# cd /usr/ports/sysutils/lsof
# make deinstall
```

That was easy enough. You have removed lsof from your system. If you would like to reinstall it, you can do so by running make reinstall from the /usr/ports/sysutils/lsof directory.

Note: The make deinstall and make reinstall sequence does not work once you have run make clean. If you want to deinstall a port after cleaning, use pkg_delete

UPDATING YOUR PORTS

This is a quick method for getting the Ports Collection using CVSup. Install the net/cvsup port. We did this ON Monday and we installed cvsup-without-gui.

As root, edit /usr/share/examples/cvsup/ports-supfile
Change CHANGE_THIS.FreeBSD.org to a CVSup server near you.
In our case this will be noc.ws.afnog.org

Run cvsup:

```
# cvsup -g -L 2 /usr/share/examples/cvsup/ports-supfile
```

It is wise to do this regularly in order to ensure that you install the latest version of a port each time you use ports for installation.

UPGRADING PORTS

Keeping your ports up to date can be a tedious job. For instance, to upgrade a port you would go to the ports directory, build the port, deinstall the old port, install the new port, and then clean up after the build. The sysutils/portupgrade utility will do everything for you! Just install it like you would any other port, using the make install clean command.

```
# cd /usr/ports/sysutils/portupgrade
# make install
```

Now create a database with the pkgdb -F command. This will read the list of installed ports and create a database file in the /var/db/pkg directory.

```
#pkgdb -F
```

Now when you run portupgrade -a, it will read this and the ports INDEX file. Finally, portupgrade will begin to download, build, backup, install, and clean the ports which have been updated. portupgrade comes with a lot of options for different use cases, the most important ones will be presented here:

Portupgrade Options

If you want to upgrade only a certain application, not the complete database, use portupgrade pkgname, include the flags -r if portupgrade should act on all those packages depending on the given package as well, and -R to act on all packages required by the given packages. Let
To use packages instead of ports for installation, provide -P. With this option portupgrade searches the local directories listed in PKG_PATH, or fetches packages from remote site if it is not found locally. If packages can not be found locally or fetched remotely, portupgrade will use ports. To avoid using ports, specify -PP.
To just fetch distfiles (or packages, if -P is specified) without building or installing anything, use -F.

Note: It is important to regularly update the package database using pkgdb -F to fix inconsistencies, especially when portupgrade asks you to. Do not abort portupgrade while it is updating the package database, this will leave you an inconsistent database.

Another nifty tool for managing your ports collection is portsnap. More information on using the portsnap tool can be found in the FreeBSD handbook at http://www.freebsd.org/handbook

Post installation activities

After installing a new application you will normally want to read any documentation it may have included, edit any configuration files that are required, ensure that the application starts at boot time (if it is a daemon), and so on.

Use pkg_info to find out which files were installed, and where. For example, we have just installed lsof-4.76 thus this command

```
# pkg_info -L lsof-4.76 | less
```

will show all the files installed by the package. Pay special attention to files in man/ directories, which will be manual pages, etc/ directories, which will be configuration files, and doc/, which will be more comprehensive documentation.

If you are not sure which version of the application was just installed, a command like this

```
# pkg_info | grep -i lsof
```

will find all the installed packages that have lsof in the package name. Replace lsof in your command line as necessary.

Once you have identified where the application's manual pages have been installed, review them using man Similarly, look over the sample configuration files, and any additional documentation that may have been provided.

Ports that should start at boot (such as Internet servers like apache) will usually install a sample script in /usr/local/etc/rc.d. You should review this script for correctness and edit or rename it if needed.