【Linux Programming】Day2

Tags	
Date	@May 14, 2022
Summary	

【Ch2】Shell Programming

2.1 What is a shell?

A shell is a program that acts as the interface between us and the Linux, enabling us to enter commands for the operating system to execute.

For example, input and output can be redirected using `<` and `>`, data piped between simultaneously executing programs using `|`, and output from a subprocess grabbed by using `$(...)`.

It is quite feasible to have multiple shells installed on Linux, with different users able to pick the one they prefer.

The following figure shows how the shell and other programs sit around the Linux kernel.

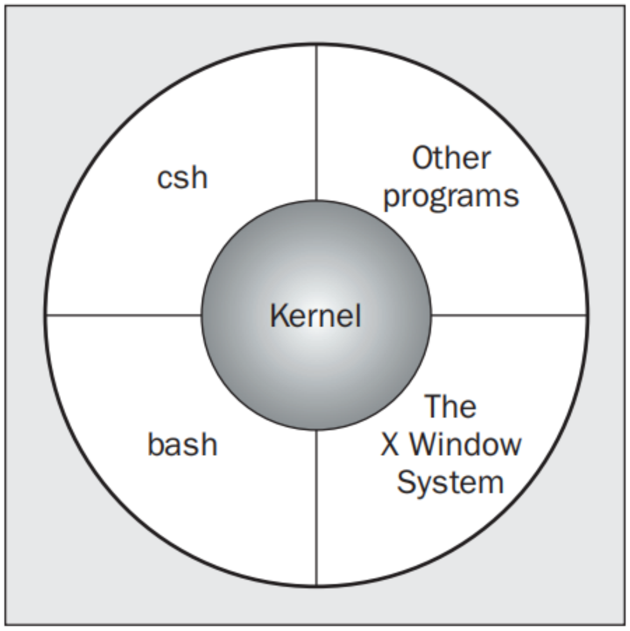
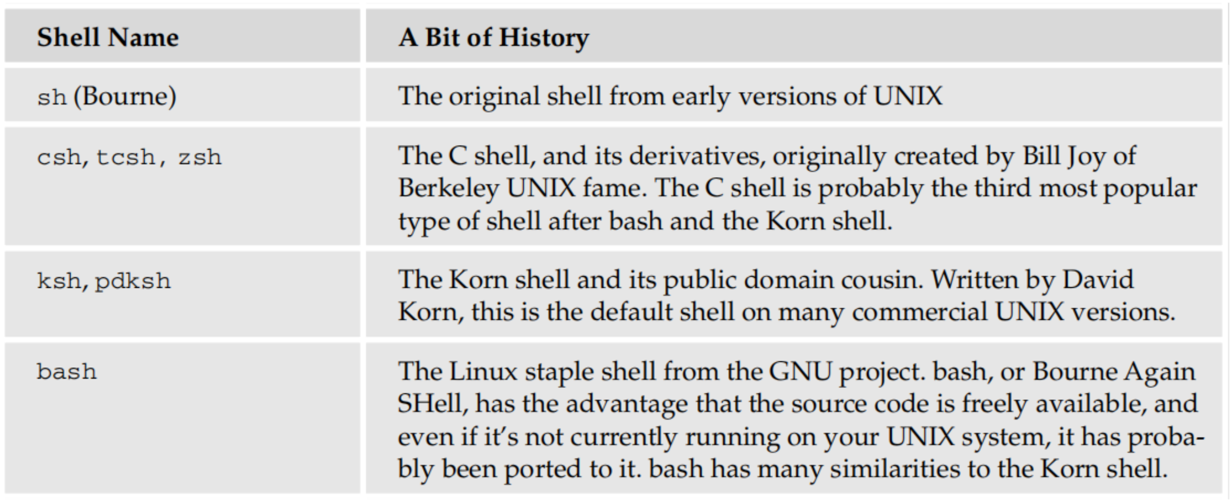


Figure 2-1

On Linux, the standard shell that is always installed as `/bin/sh` is called bash.

The following table offers summary of some of the more common shells available:



Shell Name	**A Bit of History**
`sh` (Bourne)	The original shell from early versions of UNIX
`csh, tcsh, zsh`	The C shell, and its derivatives, originally created by Bill Joy of Berkeley UNIX fame. The C shell is probably the third most popular type of shell after bash and the Korn shell.
`ksh, pdksh`	The Korn shell and its public domain cousin. Written by David Korn, this is the default shell on many commercial UNIX versions.
`bash`	The Linux staple shell from the GNU project. bash, or Bourne Again SHell, has the advantage that the source code is freely available, and even if it's not currently running on your UNIX system, it has probably been ported to it. bash has many similarities to the Korn shell.

2.2 Pipes and Redirection

Redirecting Output

We might already be familiar with some redirection, such as

```
$ ls -l > lsoutput.txt
```

which saves the output of the ls command into a file called `lsoutput.txt`.

File descriptor 0 is the stnandard input to a program, file descriptor 1 is the standard output, and file descriptor 2 is the standard error output.

We can use the command `set -o noclobber`(or `set -C`) to set the `noclobber` option to prevent a file from being overwritten using redirection.

We can cancel this option using `set +o noclobber`

To append to this file, use the `>>` operator. For example

```
$ ps >> lsoutput.txt
```

will append the output of the `ps` command to the end of the specified file.

To redirect the standard error output, preface the `>` oeprator with the number of the file descriptor we wish to redirect. Because the standard error is on file descriptor 2, use the `2>` operator.

The command

```
$ kill -HUP 1234 >killout.txt 2>killerr.txt
```

will put the output and error information into separate files.

If we prefer to capture both sets of output into a single file, we can use the `>&` operator to combine the two outputs. Therefore

```
$ kill -1 1234 >killerr.txt 2>&1
```

will put both the output and error outputs into the same file.

If we don’t want to save either standard output or standard error. We can use the Linux universal “bit bucket” of `/dev/null` to efficiently discard the entire output

```
$ kill -1 1234 >/dev/null 2>&1
```