# The Development of Shells

THE LINUX FOUNDATION

# Historical Order of Introduction

Most Linux users use the default **bash** shell, but those with long UNIX backgrounds with other shells may want to override the default; it is worth reviewing the main choices in the historical order of introduction:

- **sh** was written by Steve Bourne at AT&T in 1977, and is thus often known as the Bourne Shell; all other shells are descended from it in some fashion and it is available on all systems that have a UNIX bloodline
- **csh** was written by Bill Joy at UC Berkeley and released in 1978; the internal syntax is quite different that sh and is designed to resemble that of the C programming language, and hence the name
- **tcsh** was originally developed by Ken Greer at Carnegie Mellon University in the late 1970's; the t stands for TENEX, an operating system that was used on some DEC PDP-10s; it has many additional features as compared with csh and on virtually all modern systems csh is just a link to tcsh

# Historical Order of Introduction (Cont.)

- **ksh** was written by David Korn at AT&T and appeared in 1982, and is thus often known as the Korn shell; it was designed to be a major upgrade to sh, is backward compatible with it, and brings in some of the features of tcsh, such as command line history recall
  - This shell was long a favorite of many system administrators
- **bash** (the name stands for **B**ourne **A**gain **SH**ell) is a product of the GNU project and was created in 1987; it was designed as a major upgrade of sh; has full backward compatibility with sh and partial compatibility with ksh
- **zsh** was created by Paul Falstad at Princeton University in 1990; it is named after a professor, Zhong Shao, who used it as his login name; it is has a lot of extended features

# More on Shells

- On most Linux systems, sh is just a link to bash, scripts which are invoked as sh will only work without the bash extensions; a similar relationship exists between csh and tcsh
- Maximum portability is obtained by writing scripts that use only the older features, so you will see many scripts using sh in place of bash; but these days bash is certainly present on all Linux systems, and indeed on almost all UNIX system
- Porting of scripts between sh, ksh and bash is relatively easy and straightforward
- Porting from csh variants is more complicated because of the quite different syntax
- Linux systems make all of these shells available so porting is generally optional
- bash and ksh offer enhanced readability of scripts as compared to sh due to extensions; they also tend to run somewhat faster because the original philosophy of sh was to be minimal, have few internal commands, and rely on external commands for simple operations such as **`echo`**

# Newer Shells

- The newer shells replace some of these external commands with built-in versions
- While this is more efficient, sometimes subtle differences can wreck scripts
- Some of these snags can be avoided by using the full path in the scripts to the external command, such as **`/bin/echo`** instead of just **`echo`**, which would not search the path