The Bash Shell Startup Files

The shell program `/bin/bash` (hereafter referred to as just "the shell") uses a collection of startup files to help create an environment. Each file has a specific use and may affect login and interactive environments differently. The files in the `/etc` directory generally provide global settings. If an equivalent file exists in your home directory it may override the global settings.

An interactive login shell is started after a successful login, using `/bin/login`, by reading the `/etc/passwd` file. This shell invocation normally reads `/etc/profile` and its private equivalent `~/.bash_profile` (or `~/.profile` if called as **`/bin/sh`**) upon startup.

An interactive non-login shell is normally started at the command-line using a shell program (e.g., `[prompt]$`**`/bin/bash`**) or by the **`/bin/su`** command. An interactive non-login shell is also started with a terminal program such as **`xterm`** or **`konsole`** from within a graphical environment. This type of shell invocation normally copies the parent environment and then reads the user's `~/.bashrc` file for additional startup configuration instructions.

A non-interactive shell is usually present when a shell script is running. It is non-interactive because it is processing a script and not waiting for user input between commands. For these shell invocations, only the environment inherited from the parent shell is used.

The file `~/.bash_logout` is not used for an invocation of the shell. It is read and executed when a user exits from an interactive login shell.

Many distributions use `/etc/bashrc` for system wide initialization of non-login shells. This file is usually called from the user's `~/.bashrc` file and is not built directly into **`bash`** itself. This convention is followed in this section.

For more information see **`info bash`** -- **Nodes: Bash Startup Files and Interactive Shells**.

Note

Most of the instructions below are used to create files located in the `/etc` directory structure which requires you to execute the commands as the *root* user. If you elect to create the files in user's home directories instead, you should run the commands as an unprivileged user.

User Notes: http://wiki.linuxfromscratch.org/blfs/wiki/bash-shell-startup-files

/etc/profile

Here is a base `/etc/profile`. This file starts by setting up some helper functions and some basic parameters. It specifies some **bash** history parameters and, for security purposes, disables keeping a permanent history file for the *root* user. It also sets a default user prompt. It then calls small, single purpose scripts in the `/etc/profile.d` directory to provide most of the initialization.

For more information on the escape sequences you can use for your prompt (i.e., the `PS1` environment variable) see **info bash** -- **Node: Printing a Prompt**.

```
cat > /etc/profile << "EOF"
# Begin /etc/profile
# Written for Beyond Linux From Scratch
# by James Robertson <jameswrobertson@earthlink.net>
# modifications by Dagmar d'Surreal <rivyqntzne@pbzpnfg.arg>

# System wide environment variables and startup programs.

# System wide aliases and functions should go in /etc/bashrc.  Personal
# environment variables and startup programs should go into
# ~/.bash_profile.  Personal aliases and functions should go into
# ~/.bashrc.

# Functions to help us manage paths.  Second argument is the name of the
# path variable to be modified (default: PATH)
pathremove () {
        local IFS=':'
        local NEWPATH
        local DIR
        local PATHVARIABLE=${2:-PATH}
        for DIR in ${!PATHVARIABLE} ; do
                if [ "$DIR" != "$1" ] ; then
                  NEWPATH=${NEWPATH:+$NEWPATH:}$DIR
                fi
        done
        export $PATHVARIABLE="$NEWPATH"
}

pathprepend () {
        pathremove $1 $2
        local PATHVARIABLE=${2:-PATH}
        export $PATHVARIABLE="$1${!PATHVARIABLE:+:${!PATHVARIABLE}}"
}
```

```
pathappend () {
        pathremove $1 $2
        local PATHVARIABLE=${2:-PATH}
        export $PATHVARIABLE="${!PATHVARIABLE:+${!PATHVARIABLE}:}$1"
}

export -f pathremove pathprepend pathappend

# Set the initial path
export PATH=/bin:/usr/bin

if [ $EUID -eq 0 ] ; then
        pathappend /sbin:/usr/sbin
        unset HISTFILE
fi

# Setup some environment variables.
export HISTSIZE=1000
export HISTIGNORE="&:[bf]g:exit"

# Set some defaults for graphical systems
export XDG_DATA_DIRS=${XDG_DATA_DIRS:-/usr/share/}
export XDG_CONFIG_DIRS=${XDG_CONFIG_DIRS:-/etc/xdg/}
export XDG_RUNTIME_DIR=${XDG_RUNTIME_DIR:-/tmp/xdg-$USER}

# Setup a red prompt for root and a green one for users.
NORMAL="\[\e[0m\]"
RED="\[\e[1;31m\]"
GREEN="\[\e[1;32m\]"
if [[ $EUID == 0 ]] ; then
  PS1="$RED\u [ $NORMAL\w$RED ]# $NORMAL"
else
  PS1="$GREEN\u [ $NORMAL\w$GREEN ]\$ $NORMAL"
fi

for script in /etc/profile.d/*.sh ; do
        if [ -r $script ] ; then
                . $script
        fi
done

unset script RED GREEN NORMAL

# End /etc/profile
EOF
```

The /etc/profile.d Directory

Now create the `/etc/profile.d` directory, where the individual initialization scripts are placed:

```
install --directory --mode=0755 --owner=root --group=root /etc/profile.d
```

/etc/profile.d/bash_completion.sh

Note

Using the bash completion script below is controversial. Not all users like it. It adds many (usually over 1000) lines to the bash environment and makes it difficult to use the 'set' command to examine simple environment variables. Omitting this script does not interfere with the ability of bash to use the tab key for file name completion.

This script imports bash completion scripts, installed by many other BLFS packages, to allow TAB command line completion.

```
cat > /etc/profile.d/bash_completion.sh << "EOF"
# Begin /etc/profile.d/bash_completion.sh
# Import bash completion scripts

# If the bash-completion package is installed, use its configuration instead
if [ -f /usr/share/bash-completion/bash_completion ]; then

  # Check for interactive bash and that we haven't already been sourced.
  if [ -n "${BASH_VERSION-}" -a -n "${PS1-}" -a -z "${BASH_COMPLETION_VERSINFO-}" ]; then

    # Check for recent enough version of bash.
    if [ ${BASH_VERSINFO[0]} -gt 4 ] || \
       [ ${BASH_VERSINFO[0]} -eq 4 -a ${BASH_VERSINFO[1]} -ge 1 ]; then
       [ -r "${XDG_CONFIG_HOME:-$HOME/.config}/bash_completion" ] && \
            . "${XDG_CONFIG_HOME:-$HOME/.config}/bash_completion"
       if shopt -q progcomp && [ -r /usr/share/bash-completion/bash_completion ]; then
          # Source completion code.
          . /usr/share/bash-completion/bash_completion
       fi
    fi
  fi

else

  # bash-completions are not installed, use only bash completion directory
  if shopt -q progcomp; then
    for script in /etc/bash_completion.d/* ; do
      if [ -r $script ] ; then
        . $script
      fi
    done
  fi
fi
```

```
# End /etc/profile.d/bash_completion.sh
EOF
```

Make sure that the directory exists:

```
install --directory --mode=0755 --owner=root --group=root /etc/bash_completion.d
```

For a more complete installation, see **http://wiki.linuxfromscratch.org/blfs/wiki/bash-shell-startup-files#bash-completions**.

/etc/profile.d/dircolors.sh

This script uses the `~/.dircolors` and `/etc/dircolors` files to control the colors of file names in a directory listing. They control colorized output of things like **`ls --color`**. The explanation of how to initialize these files is at the end of this section.

```
cat > /etc/profile.d/dircolors.sh << "EOF"
# Setup for /bin/ls and /bin/grep to support color, the alias is in /etc/bashrc.
if [ -f "/etc/dircolors" ] ; then
        eval $(dircolors -b /etc/dircolors)
fi

if [ -f "$HOME/.dircolors" ] ; then
        eval $(dircolors -b $HOME/.dircolors)
fi

alias ls='ls --color=auto'
alias grep='grep --color=auto'
EOF
```

/etc/profile.d/extrapaths.sh

This script adds some useful paths to the PATH and can be used to customize other PATH related environment variables (e.g. LD_LIBRARY_PATH, etc) that may be needed for all users.

```
cat > /etc/profile.d/extrapaths.sh << "EOF"
if [ -d /usr/local/lib/pkgconfig ] ; then
        pathappend /usr/local/lib/pkgconfig PKG_CONFIG_PATH
fi
if [ -d /usr/local/bin ]; then
        pathprepend /usr/local/bin
fi
if [ -d /usr/local/sbin -a $EUID -eq 0 ]; then
        pathprepend /usr/local/sbin
fi
```

```
# Set some defaults before other applications add to these paths.
pathappend /usr/share/man  MANPATH
pathappend /usr/share/info INFOPATH
EOF
```

/etc/profile.d/readline.sh

This script sets up the default `inputrc` configuration file. If the user does not have individual settings, it uses the global file.

```
cat > /etc/profile.d/readline.sh << "EOF"
# Setup the INPUTRC environment variable.
if [ -z "$INPUTRC" -a ! -f "$HOME/.inputrc" ] ; then
        INPUTRC=/etc/inputrc
fi
export INPUTRC
EOF
```

/etc/profile.d/umask.sh

Setting the **umask** value is important for security. Here the default group write permissions are turned off for system users and when the user name and group name are not the same.

```
cat > /etc/profile.d/umask.sh << "EOF"
# By default, the umask should be set.
if [ "$(id -gn)" = "$(id -un)" -a $EUID -gt 99 ] ; then
  umask 002
else
  umask 022
fi
EOF
```

/etc/profile.d/i18n.sh

This script sets an environment variable necessary for native language support. A full discussion on determining this variable can be found on the **LFS Bash Shell Startup Files** page.

```
cat > /etc/profile.d/i18n.sh << "EOF"
# Set up i18n variables
export LANG=<ll>_<CC>.<charmap><@modifiers>
EOF
```

Other Initialization Values

Other initialization can easily be added to the `profile` by adding additional scripts to the `/etc/profile.d` directory.

/etc/bashrc

Here is a base `/etc/bashrc`. Comments in the file should explain everything you need.

```
cat > /etc/bashrc << "EOF"
# Begin /etc/bashrc
# Written for Beyond Linux From Scratch
# by James Robertson <jameswrobertson@earthlink.net>
# updated by Bruce Dubbs <bdubbs@linuxfromscratch.org>

# System wide aliases and functions.

# System wide environment variables and startup programs should go into
# /etc/profile.  Personal environment variables and startup programs
# should go into ~/.bash_profile.  Personal aliases and functions should
# go into ~/.bashrc

# Provides colored /bin/ls and /bin/grep commands.  Used in conjunction
# with code in /etc/profile.

alias ls='ls --color=auto'
alias grep='grep --color=auto'

# Provides prompt for non-login shells, specifically shells started
# in the X environment. [Review the LFS archive thread titled
# PS1 Environment Variable for a great case study behind this script
# addendum.]

NORMAL="\[\e[0m\]"
RED="\[\e[1;31m\]"
GREEN="\[\e[1;32m\]"
if [[ $EUID == 0 ]] ; then
  PS1="$RED\u [ $NORMAL\w$RED ]# $NORMAL"
else
  PS1="$GREEN\u [ $NORMAL\w$GREEN ]\$ $NORMAL"
fi

unset RED GREEN NORMAL

# End /etc/bashrc
EOF
```

~/.bash_profile

Here is a base `~/.bash_profile`. If you want each new user to have this file automatically, just change the output of the command to `/etc/skel/.bash_profile` and check the permissions after the command is run. You can then copy `/etc/skel/.bash_profile` to the home directories of already existing users, including *`root`*, and set the owner and group appropriately.

```
cat > ~/.bash_profile << "EOF"
# Begin ~/.bash_profile
# Written for Beyond Linux From Scratch
# by James Robertson <jameswrobertson@earthlink.net>
# updated by Bruce Dubbs <bdubbs@linuxfromscratch.org>

# Personal environment variables and startup programs.

# Personal aliases and functions should go in ~/.bashrc.  System wide
# environment variables and startup programs are in /etc/profile.
# System wide aliases and functions are in /etc/bashrc.

if [ -f "$HOME/.bashrc" ] ; then
  source $HOME/.bashrc
fi

if [ -d "$HOME/bin" ] ; then
  pathprepend $HOME/bin
fi

# Having . in the PATH is dangerous
#if [ $EUID -gt 99 ]; then
#  pathappend .
#fi

# End ~/.bash_profile
EOF
```

~/.profile

Here is a base `~/.profile`. The comments and instructions for using `/etc/skel` for `.bash_profile` above also apply here. Only the target file names are different.

```
cat > ~/.profile << "EOF"
# Begin ~/.profile
# Personal environment variables and startup programs.

if [ -d "$HOME/bin" ] ; then
  pathprepend $HOME/bin
fi

# Set up user specific i18n variables
#export LANG=<ll>_<CC>.<charmap><@modifiers>
```

```

# End ~/.profile
EOF
```

~/.bashrc

Here is a base `~/.bashrc`.

```
cat > ~/.bashrc << "EOF"
# Begin ~/.bashrc
# Written for Beyond Linux From Scratch
# by James Robertson <jameswrobertson@earthlink.net>

# Personal aliases and functions.

# Personal environment variables and startup programs should go in
# ~/.bash_profile.  System wide environment variables and startup
# programs are in /etc/profile.  System wide aliases and functions are
# in /etc/bashrc.

if [ -f "/etc/bashrc" ] ; then
  source /etc/bashrc
fi

# Set up user specific i18n variables
#export LANG=<ll>_<CC>.<charmap><@modifiers>

# End ~/.bashrc
EOF
```

~/.bash_logout

This is an empty `~/.bash_logout` that can be used as a template. You will notice that the base `~/.bash_logout` does not include a **`clear`** command. This is because the clear is handled in the `/etc/issue` file.

```
cat > ~/.bash_logout << "EOF"
# Begin ~/.bash_logout
# Written for Beyond Linux From Scratch
# by James Robertson <jameswrobertson@earthlink.net>

# Personal items to perform on logout.

# End ~/.bash_logout
EOF
```

/etc/dircolors

If you want to use the `dircolors` capability, then run the following command. The `/etc/skel` setup steps shown above also can be used here to provide a `~/.dircolors` file when a new user is set up. As before, just change the output file name on the following command and assure the permissions, owner, and group are correct on the files created and/or copied.

```
dircolors -p > /etc/dircolors
```

If you wish to customize the colors used for different file types, you can edit the `/etc/dircolors` file. The instructions for setting the colors are embedded in the file.

Finally, Ian Macdonald has written an excellent collection of tips and tricks to enhance your shell environment. You can read it online at **http://www.caliban.org/bash/index.shtml**.

Last updated on 2018-12-02 16:36:16 -0600