IBM

Understanding Linux configuration files

Classifications and usage

Level: Introductory

Subodh Soni (subodh@in.ibm.com), Software Engineer, IBM

01 Dec 2001

This article explains configuration files on a Linux system that control user permissions, system applications, daemons, services, and other administrative tasks in a multi-user, multi-tasking environment. These tasks include managing user accounts, allocating disk quotas, managing e-mails and newsgroups, and configuring kernel parameters. This article also classifies the config files present on a Red Hat Linux system based on their usage and the services they affect.

Introduction

Every Linux program is an executable file holding the list of opcodes the CPU executes to accomplish specific operations. For instance, the ls command is provided by the file /bin/ls, which holds the list of machine instructions needed to display the list of files in the current directory onto the screen. The behaviour of almost every program can be customized to your preferences or needs by modifying its configuration files.

Is there a standard configuration file format in Linux?

In a word, no. Users who are new to Linux (rightly) feel frustrated that each configuration file looks like a new challenge to figure out. In Linux each programmer is free to choose the configuration file format he or she prefers. Format options range from the /etc/shells file, which contains a list of possible shells separated by a newline, to Apache's complex /etc/httpd.conf file.

What are system configuration files?

The kernel itself may be considered a "program." Why does the kernel need configuration files? The kernel needs to know the list of users and groups in the system, and manage file permissions (that is, determine if a file can be opened by a specific user, according to the permissions, UNIX_USERS). Note that these files are not specifically read by programs, but by a function provided by a system library, and used by the kernel. For instance, a program needing the (encrypted) password of a user should not open the /etc/passwd file. Instead, it should call the system library function getpw(). This kind of function is also known as a system call. It is up to the kernel (through the system library) to open the /etc/passwd file and after that, search for the password of the requested user.

Most of the configuration files in the Red Hat Linux system are in the /etc directory unless otherwise specified. The configuration files can be broadly classified into the following categories:

Access files

/etc/host.conf	Tells the network domain server how to look up hostnames. (Normally /etc/hosts, then name server; it can be changed through netconf.)
/etc/hosts	Contains a list of known hosts (in the local network). Can be used if the IP of the system is not dynamically generated. For simple hostname resolution (to dotted

	notation), /etc/hosts.conf normally tells the resolver to look here before asking the network nameserver, DNS or NIS.
/etc/hosts.allow	Man page same as hosts_access. Read by tcpd at least.
/etc/hosts.deny	Man page same as hosts_access. Read by tcpd at least.

Booting and login/logout

/etc/issue & /etc/issue.net	These files are read by mingetty (and similar programs) to display a "welcome" string to the user connecting from a terminal (issue) or through a telnet session (issue.net). They include a few lines stating the Red Hat release number, name, and Kernel ID. They are used by rc.local.
/etc/redhat-release	Includes one line stating the Red Hat release number and name. Used by rc.local.
/etc/rc.d/rc	Normally run for all run levels with level passed as argument. For example, to boot your machine in the Graphics mode (X-Server), run the following command from your command line: `init 5`. The runlevel 5 is starts the system in graphics mode.
/etc/rc.d/rc.local	Not official. May be called from rc, rc.sysinit, or /etc/inittab.
/etc/rc.d/rc.sysinit	Normally the first script run for all run levels.
/etc/rc.d/rc/rcX.d	Scripts run from rc (**X** stands for any number from 1 to 5). These directories are "run-level" specific directories. When a system starts up, it identifies the run-level to be initiated, and then it calls all the startup scripts present in the specific directory for that run-level. For example, the system usually starts up and the message "entering run-level 3" is shown after the boot messages; this means that all the init scripts in the directory /etc/rc.d/rc3.d/ will be called.

File system

The kernel provides an interface to display some of its data structures that can be useful for determining the system parameters like interrupts used, devices initialised, memory statistics, etc. This interface is provided as a separate but dummy filesystem known as the /proc filesystem. Many system utilities use the values present in this filesystemf or displaying the system statistics. For example, the file /proc/modules lists the currently loaded modules in the system. This information is read by the command lsmod, which then displays it in a human readable format. In the same way, the file mtab specified in the following table reads the /proc/mount file, which contains the currently mounted filesystems.

/etc/mtab	This changes continuously as the file /proc/mount changes. In other words, when filesystems are mounted and unmounted, the change is immediately reflected in this file.
/etc/fstab	Lists the filesystems currently "mountable" by the computer. This is important because when the computer boots, it runs the command `mount -a`, which takes care of mounting every file system marked with a "1" in the next-to-last column of fstab.
/etc/mtools.conf	Configuration for all the operations (mkdir, copy, format, etc.) on a DOS-type filesystem.

System administration

/etc/group	Contains the valid group names and the users included in the specified groups. A single user can be present in more than one group if he performs multiple tasks. For example, is a "user" is the administrator as well as a member of the project group "project 1", then his entry in the group file will look like: `user: * : group-id : project1`
/etc/nologin	If the file /etc/nologin exists, login(1) will allow access only to root. Other users will be shown the contents of this file and their logins refused.
etc/passwd	See "man passwd". Holds some user account info including passwords (when not "shadowed").
/etc/rpmrc	rpm command configuration. All the rpm command line options can be set together in this file so that all of the options apply globally when any rpm command is run on that system.
/etc/securetty	Contains the device names of tty lines (one per line, without leading /dev/) on which root is allowed to login.
/etc/usertty ***/etc/shadow***	Contains the encrypted password information for users' accounts and optionally the password aging information. Included fields are: • Login name • Encrypted password • Days since Jan 1, 1970 that password was last changed • Days before password may be changed • Days after which password must be changed • Days before password is to expire that user is warned • Days after password expires that account is disabled • Days since Jan 1, 1970 that account is disabled
/etc/shells	Holds the list of possible "shells" available to the system.
/etc/motd	Message Of The Day; used if an administrator wants to convey some message to all the users of a Linux server.

Networking

/etc/gated.conf	Configuration for gated. Used only by the gated daemon.
/etc/gated.version	Contains the version number of the gated daemon.
/etc/gateway	Optionally used by the routed daemon.
/etc/networks	Lists names and addresses of networks accessible from the network to which the machine is connected. Used by route command. Allows use of name for network.
/etc/protocols	Lists the currently available protocols. See the NAG (Network Administrators Guide) and man page. C interface is getprotoent. Should never change.
/etc/resolv.conf	Tells the kernel which name server should be queried when a program asks to "resolve" an IP Address.
/etc/rpc	Contains instructions/rules for RPC, which can be used in NFS calls, remote file system mounting, etc.
/etc/exports	The file system to be exported (NFS) and permissions for it.
/etc/services	Translates network service names to port number/protocol. Read by inetd, telnet, tcpdump, and some other programs. There are C access routines.
/etc/inetd.conf	Config file for inetd. See the inetd man page. Holds an entry for each network service for which inetd must control daemons or other servicers. Note that services will be running, but comment them out in /etc/services so they will not be available even if running. Format: <service_name> <sock_type> <proto> <flags>

	<user> <server_path> <args>
/etc/sendmail.cf	The Mail program sendmail's configuration file. Cryptic to understand.
/etc/sysconfig/network	Indicates NETWORKING=yes or no. Read by rc.sysinit at least.
/etc/sysconfig/network-scripts/if*	Red Hat network configuration scripts.

System commands

System commands are meant exclusively to control the system, and make everything work properly. All the programs like login (performing the authentication phase of a user on the console) or bash (providing the interaction between a user and the computer) are system commands. The files associated with them are therefore particularly important. This category has the following files of interest to users and administrators.

/etc/lilo.conf	Contains the system's default boot command line parameters and also the different images to boot with. You can see this list by pressing Tab at the LILO prompt.
/etc/logrotate.conf	Maintains the log files present in the /var/log directory.
/etc/identd.conf	Identd is a server that implements the TCP/IP proposed standard IDENT user identification protocol as specified in the RFC 1413 document. identd operates by looking up specific TCP/IP connections and returning the user name of the process owning the connection. It can optionally return other information instead of a user name. See the identd man page.
/etc/ld.so.conf	Configuration for the Dynamic Linker.
/etc/inittab	This is chronologically the first configuration file in UNIX. The first program launched after a UNIX machine is switched on is init, which knows what to launch, thanks to inittab. It is read by init at run level changes, and controls the startup of the main process.
/etc/termcap	A database containing all of the possible terminal types and their capabilities.

Daemons

A daemon is a program running in non-interactive mode. Typically, daemon tasks are related to the networking area: they wait for connections, so that they can provide services through them. Many daemons are available for Linux, ranging from Web servers to ftp servers.

/etc/syslogd.conf	The configuration file for the syslogd daemon. syslogd is the daemon that takes care of logging (writing to disk) messages coming from other programs to the system. This service, in particular, is used by daemons that would not otherwise have any means of signaling the presence of possible problems or sending messages to users.
/etc/httpd.conf	The configuration file for Apache, the Web server. This file is typically not in /etc. It may be in /usr/local/httpd/conf/ or /etc/httpd/conf/, but to make sure, you need to check the particular Apache installation.
/etc/conf.modules or /etc/modules.conf	The configuration file for kerneld. Ironically, it is not the kernel "as a daemon". It is rather a daemon that takes care of loading additional kernel modules "on the fly" when needed.

User programs

In Linux (and UNIX in general), there are countless "user" programs. A most common user program config file is /etc/lynx.cfg. This is the configuration file for lynx, the well-known textual browser. Through this file you can define the proxy server, the character set to use, and so on. The following code sample shows a part of the lynx.cfg file that can be modified to change the proxy settings of the Linux system. These settings apply (by default) to all the users running lynx in their respective shells, unless a user overrides the default config file by specifying `--cfg = "mylynx.cfg`.

Proxy settings in /etc/lynx.cfg

```
.h1 proxy
.h2 HTTP_PROXY
.h2 HTTPS_PROXY
.h2 FTP_PROXY
.h2 GOPHER_PROXY
.h2 NEWS_PROXY
.h2 NNTP_PROXY
# Lynx version 2.2 and beyond supports the use of proxy servers that can act as
# firewall gateways and caching servers. They are preferable to the older
# gateway servers. Each protocol used by Lynx can be mapped separately using
# PROTOCOL_proxy environment variables (see Lynx Users Guide). If you have
# not set them externally, you can set them at run time via this configuration file.
# They will not override external settings. The no_proxy variable can be used
# to inhibit proxying to selected regions of the Web (see below). Note that on
# VMS these proxy variables are set as process logicals rather than symbols, to
# preserve lowercasing, and will outlive the Lynx image.
#
.ex 15
http_proxy:http://proxy3.in.ibm.com:80/
ftp_proxy:http://proxy3.in.ibm.com:80/
#http_proxy:http://penguin.in.ibm.com:8080
#ftp_proxy:http://penguin.in.ibm.com:8080/

.h2 NO_PROXY
# The no_proxy variable can be a comma-separated list of strings defining
# no-proxy zones in the DNS domain name space.  If a tail substring of the
# domain-path for a host matches one of these strings, transactions with that
# node will not be proxied.
.ex
no_proxy:demiurge.in.ibm.com, demiurge
```

Changing configuration files

When changing a configuration file, make sure that the program using that configuration is restarted if it's not controlled by the system administrator or the kernel. A normal user doesn't usually have privileges to start or stop system programs and/or daemons.

The kernel

Changing configuration files in the kernel immediately affects the system. For example, changing the passwd file to add a user immediately enables that user. Also there are some kernel tunable parameters in the /proc/sys directory on any Linux system. The write-access to all these files is given only to the super-user; other users have only readonly access. The files in this directory are classified in the same manner as the Linux kernel source. Every file in this directory represents a kernel data structure that can be dynamically modified to change the system performance.

Note: Before changing any value in any of these files, make sure you know everything about the file to avoid irreparable damage to the system.

Files in the /proc/sys/kernel/ directory

File name	Description
threads-max	The maximum number of tasks the kernel can run.
ctrl-alt-del	If 1, then pressing this key sequence cleanly reboots the system.
sysrq	If 1, then Alt-SysRq is active.
osrelease	Displays the release of the operating system.
ostype	Displays the type of the operating system.
hostname	The host name of the system.
domainname	Network domain of which the system is a part.
modprobe	Specifies whether modprobe should be automatically run at startup, and load the necessary modules.

Daemons and system programs

A daemon is a program that is always running in background, quietly carrying out its task. Common ones are in.ftpd (ftp server daemon), in.telnetd (telnet server daemon), and syslogd (system logging daemon). Some daemons, while running, keep a close watch on the configuration file and reload it automatically when it changes. But most of the daemons do not reload automatically. We need to "tell" them somehow that the configuration file has changed and that it should be reloaded. This can be achieved (on Red Hat Linux systems) by restarting the services using the service command.

For example, if we have changed the network configuration, we need to issue:
`service network restart`.

Note: The services are most commonly the scripts present in the /etc/rc.d/init.d/* directory and are started by the init when the system is booted. So, to restart the service you can also do the following:
`/etc/rc.d/init.d/<script-for-the-service> start | stop | status`
start, stop, and status are the values that these scripts take as input to perform the action.

User programs

A user or system program reads its configuration file every time it is launched. Remember, though, that some system programs are spawned when the computer is turned on, and their behaviour depends on what they read in the configuration files in /etc/. So, the first time a user program is started, the default configuration is read from the files present in the /etc/ directory. Later, the user can customise the programs by using rc and . (dot) files as explained in the next section.

User configuration files: . (dot) files and rc files

We have seen how programs can be easily configured. But what if someone does not like the way a program has been configured in /etc/? A "normal" user cannot simply go into /etc and change the configuration files; they are owned -- from the filesystem's point of view -- by root! This is why most user programs define two configuration files: the first one at a "system" level, located in /etc/; and the other one, "private" to the user, that can be found in his or her home directory.

For example, in my system I have installed the very useful wget utility. In /etc/ there is an /etc/wgetrc file. In my home directory, there is a file named .wgetrc, which describes my customised configuration (which will be loaded only when I, the user run the wget command). Other users may also have the .wgetrc file in their home directory (/home/other); this file will be read, of course, only when the user runs the wget command. In other words, the /etc/wgetrc file provides "default" values for wget, while the /home/xxx/.wgetrc file lists the "customisations" for a certain user. It is important to understand that this is the "general rule," and is not necessarily true for all cases. A program like pine, for instance, does not have any files in /etc/, but only the

custom configuration in the users' home directory, in a file named .pinerc. Other programs may only have a default configuration file in /etc/, and may not let users "customize" them (it's the case with only a few of the config. files in the /etc dir.).

Commonly used rc and . (dot) files

Filename	Description
~/.bash_login	Look at "man bash". Treated by bash like ~/.bash_profile if that doesn't exist.
~/.bash_logout	Look at "man bash".Sourced by bash login shells at exit.
~/.bash_profile	Sourced by bash login shells after /etc/profile.
~/.bash_history	The list of commands executed previously.
~/.bashrc	Look at "man bash". Sourced by bash non-login interactive shells (no other files are). Non-interactive shells source nothing unless BASH_ENV or ENV are set.
~/.emacs	Read by emacs at startup.
~/.forward	If this contains an e-mail address, then all mail to owner of ~ will be forwarded to that e-mail address.
~/.fvwmrc ~/.fvwm2rc	Config files for fvwm and fvwm2 (the basic X Window manager).
~/.hushlogin	Look at "man login". Causes a "quiet" login (no mail notice, last login info, or MOD).
~/.mail.rc	User init file for mail program.
~/.ncftp/	Directory for ncftp program; contains bookmarks, log, macros, preferences, trace. See man ncftp. The purpose of ncftp is to provide a powerful and flexible interface to the Internet standard File Transfer Protocol. It is intended to replace the stock ftp program that comes with the system.
~/.profile	Look at "man bash". Treated by bash like ~/.bash_profile if that and ~/.bash_login don't exist, and used by other Bourn-heritage shells too.
~/.pinerc	Pine configuration
~/.muttrc	Mutt configuration
~/.exrc	Configuration of vi can be controlled by this file. Example: set ai sm ruler Writing the above line in this file makes vi set the auto-indentation, matching brackets and displaying line number and rows-columns options.
~/.vimrc	Default "Vim" configuration file. Same as .exrc.
~/.gtkrc	GNOME Toolkit.
~/.kderc	KDE configuration.
~/.netrc	Default login names and passwords for ftp.
~/.rhosts	Used by the r-tools: rsh, rlogin, etc. Very weak security since host impersonation is easy. 1. Must be owned by user (owner of ~/) or superuser. 2. Lists hosts from which users may access this account. 3. Ignored if it is a symbolic link.
~/.rpmrc	See "man rpm". Read by rpm if /etc/rpmrc is not present.
~/.signature	Message text that will be appended automatically to the mail sent from this account.

~/.twmrc	Config file for twm (**T**he **W**indow **M**anager).
~/.xinitrc	Read by X at startup (not by xinit script). Mostly starts some progs. Example: exec /usr/sbin/startkde If the above line is present in this file, then the KDE Window Manager is started in when the startx command is issued from this account.
~/.xmodmaprc	This file is passed to the xmodmap program, and could be named anything (~/.Xmodmap and ~/.keymap.km, for example).
~/.xserverrc	Run by xinit as the X server if it can find X to execute.
~/News/Sent-Message-IDs	Default mail history file for gnus.
~/.Xauthority	Read and written by xdm program to handle authorization. See the X, xdm, and xauth man pages.
~/.Xdefaults, ~/.Xdefaults-hostname	Read by X applications during startup on hostname. If the -hostname file can't be found, .Xdefaults is looked for.
~/.Xmodmap	Points to *.xmodmaprc*; Red Hat had (has) .xinitrc using this name.
~/.Xresources	Usually the name for the file passed to xrdb to load the X resources database, to avoid the need for applications to read a long .Xdefaults file. (~/.Xres has been used by some.)
~/mbox	User's old mail.

Resources

- Also on *developerWorks:*
  - Technical FAQ for Linux users
  - What good is a Linux client?
  - Using the xinetd program for system administration
- Browse more Linux resources on *developerWorks*.
- Browse more Open source resources on *developerWorks*.

About the author

Subodh Soni has a bachelor of engineering degree in Computer Science and Technology Regional Engineering from College Surat, India. He works for IBM Global Services (IBM Software Labs) in India; he is also a member of the IBM Linux Technology Center, where he concentrates on Linux RAS (Reliability, Availability, and Serviceability). Other areas of interest are operating system internals, Linux system administration, and troubleshooting. You can reach him at subodh@in.ibm.com.