Linux Daemon

What is a daemon?

A daemon is a kind of program over UNIX-like OS that executes in the background unobtrusively, instead of upon the direct access of a user. It waits to be triggered by the appearance of a particular condition or event.

Typically, UNIX-like systems execute numerous daemons, primarily for accommodating requests for services through other systems on the network, to hardware activity, and for responding to other programs as well.

Examples of conditions or actions that could activate daemons into the activity can be a particular date or time, passage of a described time interval, receipt of a web request or email created from a specific communication line, and a file landing in a specific directory.

It is not mandatory that a perpetrator of a condition or action be known that the daemon is listening.

However, programs frequently would implement an action just because they are known that it would arouse a daemon implicitly.

A daemon is also called background processes. It is a UNIX or Linux program that executes inside the background. Almost every daemon contains names that finish with "d" the letter. For example, sshd, this manages connections of SSH remote access, or the httpd daemon that manages the Apache server. Often, Linux begins daemons at starting time.

Various shell scripts are saved inside the directory that is /etc/init.d. These scripts are used for starting and stopping daemons.

Linux Processes

Usually, daemons are instantiated as the processes. These processes are the running or executing instances of a program. A process is handled by the kernel which is the operating system's core and it assigns all the special process identification numbers.

In Linux, there are mainly three common kinds of processes which are as follows:

- ❍Batch
- ❍Interactive
- ❍Daemon

The batch process is submitted through a processes queue and is not related to the command line. These processes are well suited to perform recurring operations if the usage of the system is low.

The interactive process is interactively executed by the user on the command line.

The daemon is identified by a system like those processes whose parent process contains a PID of one.

Always, it defines the process init. The init process is the initial process that is begun when a Linux system is started up and it remains over the system until the system is shut down.

The init can adopt any type of process whose parent process terminates or dies without waiting for the status of the child process.

So, the basic technique to launch the daemon is dividing or forking twice or once and also enabling the parent processes to terminate while the child process starts implementing its general function.

History of Daemons

A few daemons are published by init scripts of System V. These are scripts or short programs that can be executed automatically if the system is starting up. They might either be reproduced at intervals or survive for the session duration.

Now, several daemons are begun only as needed and by one daemon (xinetd) instead of executing continuously. The xinetd is known as TCP/IP super server.

It is begun at starting time and also it listens to various ports which are assigned to those processes listed inside the configuration file, i.e., /etc/xinetd.conf or /etc/inetd.conf.

Manually, a few daemons could also be started in inclusion for being launched by the application programs and operating system. All the daemons have an individual script in several UNIX-like OS including Linux with which it could be re-started and terminated.

The management of these scrips is implemented according to runlevels. The runlevel can be defined as an operating or configuration state of a system that only permits some selected processes to available. Starting into a distinct runlevel can support solving certain problems or issues including rectifying system errors.

Key Points of Daemons

Some important key points of Daemons are explained below:

❍The word daemon is taken from the Greek methodology daemon. These were supernatural beings that lie between mortals and gods and which possessed unique power or knowledge.

❍In 1963, the term daemon was initially applied inside a system context on the pioneering project MAC with the help of IBM 7094.

❍It was inspired by the daemon of thermodynamics and physics of Maxwell which was an abstract agent that supported sort molecules of distinct speeds and worked in the background tirelessly.

❍After that, the term was used for describing the processes of the background which worked implement system chores tirelessly.

❍The first daemon computer was a program that created tape backups automatically.

❍This term was utilized for computer use. It was as a short form for Disk and Execution MONitor.

❍Various programs called services the daemons functions on the Microsoft Windows operating system. However, the word daemon is also sometimes being applied with those systems.

Implementation of Daemons

Unix like Systems

The process of Unix-like system is a daemon if its parent process dies and this daemon is appointed the init process (number 1 process) as the parent process and contains no controlling terminal in the strictly technical way.

A daemon might be however any process of background whether the init process's child or not.

The basic technique for a procedure becomes a daemon on the UNIX like system when the procedure is begun through the command line or startup scripts like the System Starter script or the init script, involves:

❍Deleting unnecessary variables through the environment optionally.

❍Running as the background task by exiting and forking. It allows the parent of the daemon (startup or shell process) for receiving exit notifications and continue the normal execution.

- ❍Detaching through the invoking section, accomplished by an individual operation usually, setsid():
  - ❍Dissociating through the tty controlling.
  - ❍Making a newer session and becoming that session's session leader.
  - ❍Becoming the leader of the process group.
- ❍If the daemon wishes to make sure that it would not inherit a new tty controlling, it might exit and fork again. It means that it's no longer any session leader inside the new session and cannot inherit any tty controlling.
- ❍Setting the current working directory as the root directory so the process doesn't take any directory in use that might be over a mounted file system.
- ❍Modifying the umask to 0 for allowing create(), open(), and other calls of operating system to facilitate their permission mask and not to rely on all the caller umask.
- ❍Redirecting the file descriptors 0, 1, and 2 for the standard streams (stderr, stdout, stdin) to a logfile or /dev/null, and closing every other descriptor file acquired through the parent process.

When the process begins by any super server daemon like systemd, launchd, or inetd, the super server will implement those functions for this process, except those old-style daemons not transformed into executing under systemd and described as multithreaded and Type=forking datagram servers upon inetd.

MS-DOS

The daemon-like program was executed as terminate and stay resident (in short TSR) software inside the Microsoft DOS platform.

Windows NT

Programs known as Windows services implement the functions of these daemons on the Microsoft Windows NT systems. They execute as processes and usually don't interact with the mouse, keyboard, and monitor. They might be launched with the help of the operating system at the time of boot.

Windows services are manually stopped, started, and configured by the Control Panel (a dedicated configuration/control program), the PowerShell scripting system, or net stop and net start commands, the service controller element of the service control manager.

Any Windows application however can implement the responsibilities of a daemon not just like a service and a few windows daemons contain the option of executing as the normal process.

Classic macOS and Mac OS

Various optional services and features were facilitated by those files loaded at the boot-up time that rebuild the operating system on the classic Mac OS.

These were called control panels and system extensions. The later versions of standard Mac OS enlarged these with completely fledged faceless background applications.

These applications are regular applications that execute inside the background. These were still specified as the regular system extensions to the user.

macOS is a Unix system and it uses daemons. The macOS applies the term of the service for designating software that implements functions chosen through the services menu instead of applying that term as windows do for daemons.

Typical daemons functions

❍Execute scheduled actions like cron.

❍Monitor systems like RAID array or hard disk health.

❍Respond to the request of the network and open network port (like port 80).

How do we start, restart, or stop daemons for a shell prompt?

We need to apply their service commands like below:

1. service daemon-name-here start
2. service daemon-name-here stop
3. service daemon-name-here restart

In the following example, stars, restart and stop.

1. service httpd start
2. service httpd stop
3. service httpd restart

How do we check the list of each running daemon?

To check the status of every installed daemon, type:

1. service - -status-all

Planning Our Daemon

What daemon going to do?

The daemon must implement one thing, and implement it well. That single thing might be as complicated as handling a lot of mailboxes over more than one domain or as easy as calling sendmail for mailing it out towards an admin and specifying a report.

We must have a better idea of what a daemon must do in any case. It's going to interact with a few other daemons that we might specify or not. It is also something else to examine.

Interaction

Daemons must never have any communication with the user directly by a terminal. Every communication passes by a few interface sort (which we might or might not have to specify), which could be as complicated as the GUI+GTK or as easy as an individual set.

The basic structure of daemon

Daemon has to implement a few low-level housework for getting itself ready for the real job when it starts up. It involves some steps which are as follows:

❍Fork off a super process (parent process)

❍Modify mask of file mode (umask)

❍Open logs to write

❍Make a special Session ID (in short SID)

❍Modify the working directory (current) to a secure place

❍Close classing descriptors of file

❍Enter original daemon code

List of daemon services for Unix and Linux like systems

❍**anacron:** It runs delayed cron actions at the start-up time.

❍**amd:** It stands for Auto Mount Daemon.

❍**atd:** It executes jobs queued applying the at the tool.

❍**apmd:** It stands for Advanced Power Management Daemon.

❍**crond:** It is a task scheduler daemon.

❍**autofs:** It helps the automounter daemon permitting unmount and mount of devices over demand.

❍**dhcpd:** It stands for Dynamic Host Configuration Protocol. Also, it is an Internet Bootstrap Protocol Server.

❍**cupsd:** It stands for CUPS printer daemon.

❍**ftpd:** It stands for FTP Server Daemon.

❍**httpd:** It is a Web Server Daemon.

❍**gated:** It can route the daemons that replace egpup and routed and manage more than one routing protocol.

❍**lpd:** It stands for Line Printer Daemon.

❍**imapd:** It is the imap server daemon.

❍**inetd:** It stands for Internet Superserver Daemon.

❍**memchached:** It is an in-memory distributed object caching daemon.

❍**mysql:** It is a database server daemon.

❍**mountd:** It is a mount daemon.

❍**nfsd:** It stands for Network File Sharing Daemon.

❍**named:** It is a DNS server daemon.

❍**nflock:** It is applied for starting and stopping the locking services of nfs files.

❍**ntpd:** It stands for Network Time Protocol service daemon.

❍**nmbd:** It stands for Network Message Block Daemon.

❍**postgresql:** It is a database server daemon.

❍**postfix:** It is a mail transport agent and used as a substitution for Sendmail.

❍**rpcbind:** It stands for Remote Procedure Call Bind Daemon.

❍**routed:** It handles routing tables.

❍**smbd:** It is a Samba Daemon.

❍**sendmail:** It is a mail transfer agent daemon.

❍**smtpd:** It stands for Simple Mail Transfer Protocol Daemon.

❍**squid:** It is a web page caching proxy server daemon.

❍**snmpd:** It stands for Simple Network Management Protocol Daemon.

❍**syncd:** It can keep various file systems synchronized along with system memory.

❍**sshd:** It is a Secure Shell Server Daemon.

❍**syslogd:** It stands for system logging daemon.

❍**telnetd:** It is a Telnet Server Daemon.

❍**tcpd:** It has a service wrapper that can restrict the authorization to inetd-based services from hosts.deny and hosts.allow.

❍**vsftpd:** It stands for Very Secure FTP Daemon.

❍**webmin:** It is a web-based administration server daemon.

❍**xntd:** It is a Network Time Server Daemon.

❍**xinetd:** It is an Enhanced Internet Superserver Daemon.