p4dcfg – Configuration tool for UNIX

# Summary

p4dcfg is a p4d configuration tool for UNIX. It solves the common administration problem of the automatic start-up and shut-down of p4d and related servers.

On UNIX, servers are typically started by the *init.d* scripts. These scripts will store the process id (pid) in /var/run and use this information to shut down the application when the operating system is brought down. Scripting this for Perforce services is more complicated because these services should not run under the user *root*. p4dcfg solves this problem by setting the user id before starting the servers.

Services currently supported are:

|  |  |
| --- | --- |
| Service Type | Description |
| p4d | Perforce Server |
| p4p | Perforce Proxy |
| p4web | Perforce Web Client |
| p4ftp | Perforce FTP Client |

p4dcfg is designed to make the start-up and shut-down process easier. It uses a configuration file to identify which processes are configured and supports a set of administration commands:

|  |  |
| --- | --- |
| Command | Description |
| start | Starts the service |
| stop | Stops the service |
| status | Reports the status of the service |
| restart | Restarts the service |
| checkpoint | Performs a checkpoint (only p4d) |
| journal | Performs a journal rotation (only p4d) |

p4dcfg is designed to run as a setuid process owned by root to be able to update the /var/run directory. This makes it possible that the named owner of the named services can execute p4dcfg and keep the pid files (which can only be accessed by root) in sync.

# Configuration file

# Comment  
   
parameter = value  
  
type name   
{  
 parameter = value  
}

The configuration file is a text file. The default name is /etc/p4dcfg.conf. It has the following form

Environment variables assigned a value outside a server definition are global to all named services below the assignment.

Variables assigned within a server definition are only valid within the scope of the named service.

A named service is the type (*p4d*, *p4p*, *p4web* or *p4ftp*) followed by its name. Services are named to distinguish multiple services. The combination (type, name) is unique.

Each service type has a set of required variables that need to be defined before the service can start. Environment variables defined outside the configuration file are ignored by p4dcfg.

Required variables for each of the service are:

## p4d

|  |  |
| --- | --- |
| Name | Description |
| Service | Path to the executable p4d |
| Client | Path to the executable p4 |
| Owner | User name, owner of the p4d process |
| P4ROOT | Root directory of p4d |
| P4PORT | Port on which p4d will listen |

## p4p

|  |  |
| --- | --- |
| Name | Description |
| Proxy | Path to the executable p4p |
| Owner | User name, owner of the p4p process |
| P4PCACHE | Cache directory of p4p |
| P4TARGET | Port proxy uses to connect to the server |
| P4PORT | Port on which p4p will listen |

## p4web

|  |  |
| --- | --- |
| Name | Description |
| Web | Path to the executable p4web |
| Owner | User name, owner of the p4web process |
| P4WEB | Port on which p4web will listen |
| P4PORT | Port proxy uses to connect to the server |

## p4ftp

|  |  |
| --- | --- |
| Name | Description |
| Ftp | Path to the executable p4ftpd |
| Owner | User name, owner of the p4ftpd process |
| P4FTPPORT | Port on which p4ftpd will listen |
| P4PORT | Port proxy uses to connect to the server |

These variables can be defined globally (such as *Owner* or *Server*) or locally per server (such as *P4PORT*).

Additional environment variables (such as P4DEBUG or P4AUDIT) can be set in the configuration file; they will be defined the environment of the executed service.

In addition, you can define a variable called *Options*, which can contain any combinations of valid parameters and will be passed “as is” to the executable.

# Usage

## Command line

p4dcfg is invoked from the command line or a shell script with the following command line syntax:

**p4dcfg** [options] <command> <name>  
**p4dcfg** –h

**Options:**

-v verbose output  
-c <configfile> Config file used (only root)  
-t {p4d, p4p, p4web, p4ftp} Restrict to process type  
-z compress checkpoints  
-h print this message

**Commands:**

start starts the server(s)  
stop stops the server(s)  
restart restarts the server(s)  
status indicates the state of the server(s)  
checkpoint performs a checkpoint on the server(s) - only p4d  
journal rotates the journal of the server(s) - only p4d

If <name> is -a, all named services are affected.

Only root can use the option “-c” to use a different configuration file from the default “/etc/p4dcfg.conf”. Since p4dcfg is a setuid program, this rule is enforced to increase security.

## From init.d

Here is an example of how p4dcfg could be used from init.d. This is content of the file /etc/init.d/perforce:

**#! /bin/sh –e  
#### BEGIN INIT INFO  
# Provides: p4d  
# Required-Start: $syslog $time $local\_fs  
# Required-Stop: $syslog $time $local\_fs  
# Default-Start: 2 3 4 5  
# Default-Stop: S 0 1 6  
# Short-Description: Start script for Perforce Servers  
# Description: Starts a range of p4 servers  
#  
### END INIT INFO  
#  
# Author: Sven Erik Knop** [**sknop@perforce.com**](mailto:sknop@perforce.com) **#  
set -e  
  
PATH=/bin:/usr/bin:/sbin:/usr/sbin:/usr/local/bin  
STARTAPP=/usr/local/bin/p4dcfg  
CONFIGFILE=/etc/p4dcfg.conf  
  
test -x $STARTAPP || exit 0  
test -x $CONFIGFILE || exit 0  
  
case "$1" in  
 start)  
 $STARTAPP -c $CONFIGFILE start -a  
 ;;  
 stop)  
 $STARTAPP -c $CONFIGFILE stop -a  
 ;;  
 force-reload|restart)  
 $STARTAPP -c $CONFIGFILE restart -a  
 ;;  
 \*)  
 echo "Usage: /etc/init.d/p4dcfg {start|stop|restart|force-reload}"  
 exit 1  
 ;;  
esac  
  
exit 0**