PERFORCE

Best Practices

Perforce Software, Inc.

Preface

This document provides best practices for the following Perforce administrative tasks:

- Configuring the Perforce Server
- Backup, Replication, and Recovery
- Server Maintenance
- Security
- Maximizing Server Performance

This guide assumes some familiarity with Perforce and does not duplicate the basic information in the Perforce user documentation. For basic information on Perforce, consult *Introducing Perforce*. For system administrators, the *Perforce System Administrator's Guide* is essential reading. All documentation is available from the Perforce web site at http://www.perforce.com.

Please Give Us Feedback

Perforce welcomes feedback from our users. Please send any suggestions for improving this document or the SDP to consulting@perforce.com.

Configuring the Perforce Server

This chapter tells you how to configure a Perforce server machine and an instance of the Perforce Server. These topics are covered more fully in the System Administrator's Guide and in the Knowledge Base.

Volume Layout and Hardware

To ensure maximum data integrity and performance, use three different physical volumes for each server instance. Three volumes can be used for all instances hosted on one server machine, but using three volumes per instance reduces the chance of hardware failure affecting more than one instance.

- **Perforce metadata (database files):** Use the fastest volume possible, ideally RAID 1+0 on a dedicated controller with the maximum cache available on it. This volume is normally referred to as /metadata.
- **Journals and logs:** Use a fast volume, ideally RAID 1+0 on its own controller with the standard amount of cache on it. This volume is normally referred to as /logs. If a separate logs volume is not available, put the logs on the depotdata volume.
- **Depot data, archive files, scripts, and checkpoints**: Use a large volume, with RAID 5 on its own controller with a standard amount of cache or you can user SAN or NAS storage. This volume is the only volume that must be backed up (if you use the layout recommend in this guide). Back up scripts should place the checkpoints on this volume. This volume can be backed up to tape or another long term backup device. This volume is normally referred to as /depotdata.

For optimal performance on UNIX machines, use the XFS file system.

If three controllers are not available, put the `/logs` and /depotdata volumes on the same controller. Do not run anti-virus tools or backup tools against the /metadata volume(s) or /logs volume(s), because they can interfere with the operation of the Perforce server.

The recommendations above are for absolute maximum performance on a bare metal installation. If you are installing Perforce in a VM and using SAN/Network storage, you do not need all of these volumes. However, it is still recommended that you do NOT put your metadata on the VM virtualized disk or on a NAS volume unless you are running a small Perforce server.

Memory, CPU and OS

Make sure the server has enough memory to cache the **db.rev** database file and to prevent the server from paging during user queries. Maximum performance is obtained if the server has enough memory to keep all of the database files in memory. The fastest I/O is no I/O on the database.

Below are some approximate guidelines for allocating memory.

- 1.5 kilobyte of RAM per file stored in the server.
- 32 MB of RAM per user.

Use the fastest processors available with the fastest available bus speed. Faster processors with a lower number of cores provide better performance for Perforce. Quick bursts of computational speed are more important to Perforce's performance than the number of processors, but you should have a minimum of two processors so that the offline checkpoint and backup processes do not interfere with your Perforce server. The bottleneck for Perforce is the lock time held on the db files during the compute phase of a command. The faster your CPU is, the shorter the lock times are.

Linux with the XFS file system provides the best performance for Perforce.

Protections, file types, monitoring, security and the configure table

After the server is installed, most sites will want to modify server permissions (protections) and security settings. Other common configuration steps include modifying the file type map and enabling process monitoring. To configure permissions, perform the following steps:

1. To set up protections, issue the `p4 protect` command. The protections table is displayed.
2. Delete the following line:
   `write user * * //depot/...`
3. Define protections for your server using groups. Perforce uses an inclusionary model. No access is given by default; you must specifically grant access to users and groups in the protections table. It is best for performance to grant users specific access to the areas of the depot that they need rather than granting everyone open access and then trying to remove access via exclusionary mappings, even if that means you end up generating a larger protections table.
4. To set the server's default file types, run the p4 typemap command and define your typemap to override Perforce's default behavior.

   Add any file type entries that are specific to your site. Suggestions include:

   - For already-compressed file types (such as `.zip`, `.gz`, `.avi`, `.gif`), assign a file type of `binary+Fl` to prevent the server from attempting to compress them again before storing them.
   - For regular binary files, add binary+l so that only one person at a time can check them out.
   - For large, generated text files, assign the `text+C` file type, to avoid causing server memory issues.

5. To enable monitoring on your server using the p4 monitor command, set the monitor counter as follows:

   ```
   p4 configure set monitor=1
   ```

6. To set your security level to the highest level, issue the following command:

   ```
   p4 configure set security=3
   ```

7. Set the following recommended configuration settings (See the *Perforce System Administrator's Guide* for more details):
   - `p4 configure set journalPrefix=/path/to/checkpoints/prefix_of_checkpoints` (Defines location and prefix for rotated journals)
   - `p4 configure set dm.user.noautocreate=`2 (Turns off auto user creation)
   - `p4 configure set dm.user.resetpassword=1` (Requires new user's to reset their password. Only set if you are using internal authentication.)
   - `p4 configure set filesys.P4ROOT.min=1G` (Sets minimum free space to 1 Gigabyte for $P4ROOT)
   - `p4 configure set filesys.depot.min=1G` (Sets minimum free space to 1 Gigabyte for the depots)
   - `p4 configure set filesys.P4JOURNAL.min=1G` (Sets minimum free space to 1 Gigabyte for the journal volume)
   - `p4 configure set server=3` (Sets the log file to record all user activity)
   - `p4 configure set lbr.autocompress=1` (turns off RCS file storage for text files)
   - `p4 configure set lbr.bufsize=1M` (Sets the librarian buffer size to 1 Meg)
   - `p4 configure set server.commandlimits=2` (Prevents users from overriding group limts on the command line.)
   - `p4 configure set serverlog.file.3=logpath/errors.csv` (Turns on structured errors log)
   - `p4 configure set serverlog.file.7=logpath/events.csv` (Turns on structured events log)
   - `p4 configure set serverlog.file.8=logpath/integrity.csv` (Turns on structured replica integrity log)
   - `p4 configure set serverlog.retain.3=7` (Maintains only the last 7 error logs)
   - `p4 configure set serverlog.retain.7=7` (Maintains only the last 7 events logs)
   - `p4 configure set serverlog.retain.7=8` (Maintains only the last 7 integrity logs)

See
http://www.perforce.com/perforce/doc.current/manuals/p4dist/chapter.replication.html#replication.verifying if you are also setting up a replica server.

```
p4 configure set rpl.checksum.auto=1
p4 configure set rpl.checksum.change=2

p4 configure set rpl.checksum.table=1
```

8. To make your changelists default to restricted, set the following (for higher security environments):

```
p4 configure set defaultChangeType=restricted
```

Backup, Replication, and Recovery

Perforce servers maintain *metadata* and *versioned files*. The metadata contains all the information about the files in the depots. Metadata resides in database (db.*) files in the server's root directory (P4ROOT). The versioned files contain the file changes that have been submitted to the server. Versioned files reside on the depotdata volume.

This section assumes that you understand the basics of Perforce backup and recovery. For more information, consult the *Perforce System Administrator's Guide* and the Knowledge Base articles about replication.

Typical Backup Procedure

Maintenance scripts typically run as *cron* jobs on Unix/Linux or as Windows *scheduled tasks* that periodically back up the metadata. The weekly sequence for offline checkpoints is described below (This sequence comes from the Perforce Server Deployment Package sold through Perforce Consulting).

Six nights a week, perform the following tasks.

1. Truncate the active journal.
2. Replay the journal to the offline database.
3. Create a checkpoint from the offline database.
4. Recreate the offline database from the last checkpoint.

Once a week, perform the following tasks.

1. Stop the live server.
2. Truncate the active journal.
3. Replay the journal to the offline database.
4. Archive the live database.
5. Move the offline database to the live database directory.
6. Start the live server.
7. Create a new checkpoint from the archive of the live database.
8. Recreate the offline database from the last checkpoint.
9. Verify all depots.

This normal maintenance procedure should put the checkpoints (metadata snapshots) on the /depotdata volume, which contains the versioned files. Backing up the /depotdata volume with a normal backup utility provides you with all the data necessary to recreate the server.

To ensure that the backup does not interfere with the metadata backups (checkpoints), coordinate backup of the /depotdata volume to run after the checkpoints are created. The preceding maintenance procedure minimizes server downtime, because checkpoints are created from offline or saved databases while the server is running.

With no additional configuration, the normal maintenance prevents loss of any more than one day's metadata changes. To provide an optimal Recovery Point Objective (RPO), Perforce provides additional tools for replication.

Full One-Way Replication

Perforce supports a full one-way replication of data from a master server to a replica, including versioned files. The p4 pull command is the replication mechanism, and a replica server can be configured to know it is a replica and use the replication command. The p4 pull mechanism requires very little configuration and no additional scripting. As this replication mechanism is simple and effective, we recommend it as the preferred replication technique. Replica servers can also be configured to contain only metadata, which can be useful for reporting or offline checkpointing purposes. See the *Distributing Perforce Guide* for details on setting up replica servers. http://www.perforce.com/perforce/doc.current/manuals/p4dist/p4dist.pdf

If you wish to use the replica as a read-only server, you can use the P4Broker to direct read-only commands to the replica or you can use a forwarding replica. The broker can do load balancing to a pool of replicas if you need more than one replica to handle your load. Use of the broker may require use of a P4AUTH server for authentication.

Server Maintenance

This section describes typical maintenance tasks and best practices for administering server machines.

Unloading and Reloading labels

To use the unload and reload commands for archiving clients and labels, you must first create an unload depot using the "p4 depot" command. Run:

```
p4 depot unload
```

Set the '`Type:`' field of the depot to `unload` and save the form.

After the depot is created, you can use the following command to archive all of the clients and labels that have been accessed since the given date:

```
p4 unload -f -L -z -a -d date
```

For example, to unload all clients and labels that haven't been accessed since Jan. 1, 2013, you would run:

```
p4 unload -f -L -z -a -d 2013/01/01
```

Users can reload their own clients/labels using the reload command. They can run:

```
p4 reload -c clientname
```

or

```
p4 reload -l labelname
```

As a super user, you can reload an unloaded item by adding the -f flag to the reload command as follows:

```
p4 reload -f -c|l specname
```

In addition, you can avoid having to unload or reload labels by creating a trigger to set the autoreload option as the default on all new labels. That will cause the server to use the unload depot for storing the labels rather than storing them in db.label. This helps with server performance by not increasing the size of the database for label storage.

Managing users

P4Admin has a window on the home page that allows you to list users that haven't accessed the server in a specified time period. It is a good idea to remove those users and their workspaces to keep the database clean and keep your licenses free for active users.

Removing empty changelists

It is a best to delete empty pending changelists periodically. Users create numbered changelists as part of their daily work and revert the files out of them, leaving empty pending changelists behind. To delete those changelists, do the following:

```
p4 changes -s pending | cut -d " " -f 2 > changes.txt
```

Then, just loop over all of the changelists in changes.txt and attempt to delete them with:

```
p4 change -f -d change_number
```

Only the empty changelists will be deleted.

Security

This section describes security best practices for administering server machines.

SSL Encryption - Data on the wire

Perforce supports SSL encryption on the communications between the client and the server. All clients and API tools must be 2012.1 or higher in order to support SSL communications. You can phase in encryption by setting up a broker that supports SSL encryption on a new port on the Perforce server and then slowly switch everyone over to the new server address and port. Once the switch is complete, you can disable access to the unencrypted port. See section "Encrypting connections to a Perforce server" in the *Perforce System Administrator's Guide* for more details:

http://www.perforce.com/perforce/doc.current/manuals/p4sag/03_superuser.html

Encryption - Data at rest

If you choose to encrypt your data at rest, make sure that you use a separate volume for the Perforce metadata (db.*) and do not encrypt that volume. The data encryption and decryption process adds latency to the I/O to the database and Perforce performance depends on fast I/O to the database. No source code is stored in the Perforce database, so it is not necessary to encrypt the database.

Encryption - Backups

Your backup systems should all be encrypted as well, particularly tape backups. Tapes are the most vulnerable since they are easily moved to another system where they can be restored.

VLANs

For maximum security, you can place your Perforce server inside of a virtual lan environment and only open access to the ports that are necessary for accessing your server. It is also good practice to run your server on something other than the default port so that people cannot guess at it being on port 1666.

Passwords

It is recommended that you use Perforce internal passwords for maximum security. While Perforce does support authentication to external systems via an authentication trigger, the passwords are passed via stdin to the trigger, making it possible for a Perforce admin to capture those passwords. It is understood that the Perforce admin is a trusted individual since they have full access to the source code, but giving them full access to people's AD/LDAP passwords could expose problems since those passwords allow access to user's private company records.

Perforce supports password minimum strength, password expiration, forced password resets, and minimum passwords lengths internally as of release 2012.2, so the majority of IT password requirements can be met using internal passwords.

Operator User

Use the operator user for running your Perforce server. The operator user can perform Checkpoints, verifications, etc. This prevents you from having a logged in regular user on the Perforce server all of the time.

Init Scripts

Make sure the init scripts that stop and start your Perforce services are only modifiable by the root user, and that they do not call out to other scripts that can be modified by a non-root user. Doing so would allow a hacker to run commands on the system as the root user giving himself or herself access to the entire server.

Use the Perforce Broker

It gives you the ability to control and block many things the server cannot do by itself. One example is the ability to block unauthorized proxy servers. You can implement a filter script that checks the broker IP address and only allows the command to go through if it is coming from an authorized proxy server. The command handling feature of the broker has largely been replaced by command triggers in the server, but the broker can still be useful for certain thing depending on your needs.

Maximizing Server Performance

The following sections provide some guidelines for maximizing the performance of the Perforce Server. More information on this topic can be found in the System Administrator's Guide and in the Knowledge Base.

Optimizing the database files

The Perforce Server's database is composed of b-tree files. The server does not fully rebalance and compress them during normal operation. To optimize the files, you must checkpoint and restore the server.

To minimize the size of back up files and maximize server performance, minimize the size of the db.have and db.label files.

Limiting large requests

To prevent large requests from overwhelming the server, you can limit the amount of data and time allowed per query by setting the maxresults, maxscanrows and maxlocktime parameters to the lowest setting that does not interfere with normal daily activities. As a good starting point, set these parameters as follows:

- maxscanrows to maxresults * 4
- maxresults to slightly larger than the maximum number of files the users need to be able to sync to do their work
- maxlocktime to 30000 milliseconds.

These values must be adjusted up as the size of your server and the number of revisions of the files grow. To simplify administration, assign limits using a single group called limits that contains all of the users on your server and do not use that group in the protections table. Leave all other group values as the default of unset. You may need a second group for power users that provides a higher set of limits.

To prevent users from inadvertently accessing large numbers of files, define their client view to be as narrow as possible, considering the requirements of their work. Similarly, limit users' access in the protections table to the smallest number of directories that are required for them to do their job.

Finally, keep triggers simple. Complex triggers increase load on the server.

P4V performance settings

See:

http://answers.perforce.com/articles/KB/2878

and

http://answers.perforce.com/articles/KB/14844