Introduction to Perforce for Administrators

You may complete these exercises using the command line, P4V, and the P4Admin GUI, as desired. Exercises that require using the command line only are noted as such. The answers use command line syntax. Notepad, the default text editor for Perforce forms on a Windows platform, will be used throughout these exercises for editing both Perforce forms, such as changelists, and for text files. The steps used to answer the questions in all exercise sets are more important than the results you get within this demo database, as an aid to adapting what you learn in class to your work situation at home.

In the scenario for these exercise sets, you are a new Perforce user, starting to work on files in an existing Perforce database. The prior system administrator (who you are taking the job over from) has already set up your local environment variables so you can connect to your Perforce server. Use the username "bruno," and the client workspace spec name "bruno_ws". Your initial password is: brunopass

You have a Perforce server running on your local machine on port 1666. For this training course, the default folder for the Perforce server installation is ***c:\p4train*** *. The path to this folder is mentioned in a number of the course exercises / answers – if the server is installed into a different location, replace c:\p4train with the appropriate path when carrying out the exercises.*

Each exercise set has the same name as the corresponding PowerPoint chapter title.

Your objectives for this initial exercise:

- Log in to Perforce.
- Sync your workspace

  Consider command line options, environment variables, and P4CONFIG

1. Log in to Perforce using bruno's password: brunopass

   Hint: use **p4 login**

```
p4 -u bruno login
```

2. Synchronize your workspace with the head revisions of files from your Perforce server.

   Hint: use **p4 sync**

```
set P4CLIENT=bruno_ws
p4 sync
```

or

```
p4 -c bruno_ws sync
```

Alternative solution:

```
cd %userprofile%
set P4CONFIG=.p4config.txt
echo P4USER=bruno > %P4CONFIG%
echo P4CLIENT >> %P4CONFIG%
p4 login
p4 sync
```


Lab Set 1: Installation

Your objectives for this exercise:

- To start another Perforce server running on your local machine. (As an administrator, you may wish to do this on occasion for testing purposes.)

3. Create a directory for a test server on port 1777, and start the server from that directory:

```
MKDIR C:\p1777
CD c:\p1777
p4d –p 1777 -r c:\p1777 -J c:\p1777\journal
```

If the server starts successfully, you will see a message "Perforce server starting..." and your session may not return to a DOS prompt. Minimize this window; the server will remain running as long as the window is not closed.

In class the server program is running on the same machine as the client program; therefore you will not need to specify a machine name in the following exercises that communicate with server port 1777.

At your local site, the server and client programs will most likely not be running on the same machine.

4. Open a second MS-DOS window, and use p4 usage flags to communicate with port 1777. Start by using:

```
p4 -p 1777 info
```

to make sure you can connect to that new server running on port 1777.

5. Create a new client workspace spec named bruno-mars in the 1777 server. Make sure the "bruno-mars" client workspace root doesn't overlap the "bruno_ws" client workspace root.

```
p4 -p 1777 client bruno-mars
```

In the spec, use something like this for the workspace root:

```
Root: c:\mars
```

6. What are the root directories for servers 1666 and 1777?

```
p4 -p 1666 info
p4 -p 1777 info
```

The 1777 server is needed for Lab 3 so leave it running. With server 1666 use the workspace "bruno_ws" with the root directory "C:\bruno_ws".

7. Create a new Perforce server instance, with its own root directory C:\p4_5\root, its own logs directory C:\p4_5\logs, and its own copy of p4s.exe (copied from C:\p4train\p4d.exe). Configure a Windows service named p4_5 to use the copied p4s.exe, running on port 5000 with a log file, C:\p4_5\logs\log and journal C:\p4_5\logs\journal.

We'll walk you through this one! Start by creating directories:

```
MKDIR C:\p4_5\root C:\p4_5\logs
```


Grab an instance-specific copy of p4s.exe:

```
COPY C:\p4train\p4d.exe C:\p4_5\root\p4s.exe
```

Set Perforce configuration variables specific to the p4_5 service (which does not yet exist):

```
p4 set –S p4_5 P4ROOT=C:\p4_5\root
p4 set –S p4_5 P4PORT=5000
p4 set –S p4_5 P4JOURNAL=C:\p4_5\logs\journal
p4 set –S p4_5 P4LOG=C:\p4_5\logs\log
```

Now, create the Windows service:

```
svcinst create –n p4_5 –e C:\p4_5\root\p4s.exe –a
```

Next, start the new service you just created:

```
net start p4_5
```

If everything went right, you'll see a successful start message. If you see an error, perhaps something about "System error 1067", then start troubleshooting. Start checking for typos, make sure the directories were created, etc.

Test that you can access the server:

```
p4 –p 5000 info
```

You can remove the service as well using svinst:

```
net stop p4_5
svcinst delete –n p4_5
```

You need only the 1666 server from here on; the port 5000 server is no longer needed.

Lab Set 2: Setup

8. There is a //depot/Misc/manuals directory, to which your workspace will need access for this exercise. Create a typemap so that documents (*.doc) files added to //depot/Misc/manuals/... cannot simultaneously be opened for edit by multiple workspaces. To test this, create a file hello_world.doc in c:\bruno_ws\Misc\manuals, with the line "hello world" in it. Add it to the depot and check to confirm its filetype.

   If your workspace does not already include the Misc/manuals directory, use "p4 client bruno_ws" to add a line to your client workspace `View:` field that maps the //depot/Misc/manuals/... files to your client workspace. Use "p4 sync //depot/Misc/manuals/..." to create the folders for you.

   ```
   p4 typemap
   ```

   Add a line to the "Typemap:" field.

   ```
   Typemap:
   +l    //depot/Misc/manuals/....doc
   ```

   ```
   cd c:\bruno_ws\Misc\manuals
   notepad hello_world.doc
   ```

   *Type in the line **hello world**, and save the file.*

   ```
   p4 add //depot/Misc/manuals/hello_world.doc
   p4 submit
   p4 files //depot/Misc/manuals/hello_world.doc
   ```

   The command output is:
   `//depot/Misc/manuals/hello_world.doc#1 - add change 834 (text+l)`

9. What value is your security counter set to?

   ```
   p4 configure show security
   ```

10. Set the security counter to level 3.

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

11. After the security settings are changed, the old unsecure password 'brunopass' is not valid anymore. What happens if you try to run any command (like 'p4 depots') against the server?

    Reset the password to something more secure: Brun0P@ss!
    After the password is changed, you need to log in again.

    ```
    p4 passwd
    p4 login
    ```

12. Disable automatic "on the fly" user creation, making it so that users must run the `p4 user` command to create accounts.

    ```
    p4 configure set dm.user.noautocreate=1
    ```

13. Disable automatic "on the fly" user creation, making it so that only Perforce super users can create new accounts.

    ```
    p4 configure set dm.user.noautocreate=2
    ```


Lab Set 3: Backup and Recovery

Your objectives for this exercise:

- Create, identify and rectify corrupt archive data
- To backup your 1777 server's metadata
- To restore your 1777 server's metadata from your backup.

As an Administrator, you will want to plan a regular schedule for backing up your Perforce data. For your production server you would also schedule an OS backup of your depot files and your checkpoint and journal archive files.

Once you have finished your restore procedure you will no longer need to keep your test server running. See instructions at the end of this exercise for checking to be sure that you are connecting to your server listening on port 1666.

14. Run a verify on your 1666 server and check for no errors

```
p4 verify -q //...
```

15. Now we will cause verify errors and "fix" them.

Go to the directory where RCS file for //depot/jam/MAIN/src/Build.com is stored, backup the archive file, edit it to provoke an error and then reset it.

```
cd c:\p4train\depot\jam\main\src
copy Build.com,v Build.com-bak
notepad Build.com,v
```

Add some text to the penultimate line in the file (the line before the single @). Run a verify to check that there are now errors for it – if done correctly the error message should say BAD!

```
p4 verify -q //...
```

Reset the file to the previous version and make sure that the *verify* is now correct again.

16. Now we will provoke verify errors for a binary file and "fix" them. This is similar to the previous exercise.

```
cd C:\p4train\depot\Misc\manuals\triggers.doc,d
rename 1.436.gz 1.436.gz.bak
```

Run a verify to check that there are now errors for //depot/Misc/Manuals/triggers.doc (error should say MISSING!)

```
p4 verify -q //...

rename 1.436.gz.bak 1.436.gz
```

Check that there are no verify errors any more.

17. **Using your 1777 server**, create a workspace and add 1 or 2 files to the server. Make sure that there are a couple of changelists that have been created as a result and note their numbers.

    Run a *verify* to make sure that everything is consistent.

18. Checkpoint your 1777 server, using p4 usage flags:

    Hint: Use a client program command.

    ```
    p4 -p 1777 admin checkpoint
    ```

19. Use the client program command to stop the 1777 server; then switch to the window where your 1777 server was running, and:

    a. move the db.* files (the database files) to a temp directory
    b. recover the database from the checkpoint and the current journal
    c. restart the server.

    Hints:

    - The checkpoint number has been printed by the server to the screen where the server was running and the current journal is truncated.
    - To ensure the server finds your checkpoint file use the '-r' flag to specify where the checkpoint file is located when you restore.

    ```
    p4 -p 1777 admin stop
    cd c:\p1777
    mkdir temp
    move db.* temp
    p4d -r c:\p1777 -jr checkpoint.1 journal
    p4d -r c:\p1777 -p 1777
    ```

    By using the '-r' flag you are telling the server to find the checkpoint file in the C:\p1777 folder.

    Run verify and use your workspace to make sure that your repository looks like it did before you stopped it and recreated it from the checkpoint.

You will no longer need a server running on port 1777. Stop that server again, close all of your MSDOS windows and open a new one. Check your user, client workspace spec name and port. They should be "bruno," "bruno_ws" and "<localhost>:1666," respectively.

Lab Set 4: Protections

Your objectives for this exercise:

- Create a protections table for your 1666 server.
- Make the username `bruno` the Perforce superuser.
- Goals:
    - Limit unauthorized access to Perforce superuser commands.

20. Create a group named `engineering`, and include users `bruno`, `earl`, `gale` and `raj`.

```
p4 group engineering
```

Edit the Users field.

```
Users:
bruno
earl
gale
raj
```

21. Create a group named `perfusers`, and include users `hera` , `quinn` , `ines` and `ona`. Include the `engineering` group as a subgroup.

```
p4 group perfusers
```

Edit the "Subgroups:" and "Users:" fields.

```
Subgroups:
engineering
Users:
hera
quinn
ines
ona
```

22. Set up permissions so that only the following rules exist:

    - The `perfusers` group has read access to all depot files.
    - The `engineering` group has write access to all files under //depot/Jam/…
    - User `bruno` is the only super user.

```
p4 protect
```

Add/modify two lines to the protections table.

```
Protections:
read  group perfusers     *  //...
write group engineering   *  //depot/Jam/...
super user  bruno         *  //...
```


23. Can user `ines` sync a file, open it for edit, change it and submit it?

    `p4 protects -u ines`

    User ines has read-only permission for all files, so can sync the file but cannot perform the other operations on it.

24. What permissions does user `bruno` have?

    `p4 protects -u bruno`

25. What groups, including subgroups, does user `earl` belong to?

    `p4 groups -i earl`

Lab Set 5: Depots

26. Add a new local depot called “td.”

    Hint: Use the ‘p4 depot’ command.

    ```
    p4 depot td
    ```

    Save the form.

27. Check to see if you have a “spec” depot. If not then created one called “specs”. Populate the spec depot with a baseline of current specs.

    ```
    p4 depots
    ```

    Search for the word “spec” in the output. To create:

    ```
    p4 depot specs
    ```

    Set the Type field:

    Type: spec

    Save the form.

    ```
    p4 admin updatespecdepot -a
    ```

28. Add the //td depot to your client workspace spec view. Make sure the files from your new depot do not mask the files from your //depot mapping.

    ```
    p4 client
    ```
    (use ‘p4 info’ to check if correct workspace is selected; if not set P4CLIENT or append workspace name to p4 client command)

    Add a line to your client workspace View:

    ```
    View:
    [existing View lines]
       //td/... //bruno_ws/td/...
    ```

29. What are the differences between your edited client workspace spec and the previous version stored in the “spec” depot?

    Hint: The depot path for your client workspace spec “bruno_ws” is: //spec/client/bruno_ws.p4s

    ```
    p4 filelog //spec/client/bruno_ws.p4s
    ```

    Select the revisions of bruno_ws.p4s to compare. This answer compares revisions #1 & #2

    ```
    p4 diff2 //spec/client/bruno_ws.p4s#1    \
    //spec/client/bruno_ws.p4s#2
    ```

30. Create a README.txt file in your client workspace, based on the mapping you created above, and add the file to the //td depot.

    ```
    MKDIR c:\bruno_ws\td
    notepad c:\bruno_ws\td\README.txt
    p4 add c:\bruno_ws\td\README.txt
    p4 submit
    ```


31. Create an unload depot named: `//unload`

    `p4 depot unload`

    Set the Type field:

    `Type: unload`

    Save the form.

Optional:

32. Unload the workspace **ines-rose (**owned by user **ines)**

    `p4 unload -f -c ines-rose`

33. Verify that the workspace has been unloaded

    `p4 clients -u ines`

    *verify **ines-rose** is not present in the client list*

34. Reload the **ines-rose** workspace and verify it is reloaded using p4 clients

    `p4 reload -f -c ines-rose`

Lab Set 6: Jobspec

Your objectives for this exercise:

- To show the fields in the current job specification form.
- To add a field to the Perforce job specification form, to allow specifying and then searching on operating system type.
- To use the new field in new and existing jobs.
- To find all jobs with a specific value in the new field.

35. What are the fields in the job specification form?

    ```
    p4 jobspec -o | more
    ```

    *The first few lines of **jobspec** output show the fields.*

36. Add a field called “OS”, whose values can be “Unix”, “NT”, “other”, or “unknown”, to the job specification form. The field’s preset is “default”, and default value is “unknown”.

    ```
    p4 jobspec
    ```

    *In the **jobspec** form, add a line for the “OS” field:*

    ```
    Fields:
          106 OS select 8 default
    ```

    and add these lines to define its values:

    ```
    Values:
          OS Unix/Windows/Mac/other/unknown
    Preset:
          OS unknown
    ```

    *Add notes about the “OS” field in the **Comments** section.*

37. Update job000004 and modify its OS field value.

    ```
    p4 job job000004
    ```

    Edit the OS field:

    ```
    OS: <add a valid value>
    ```

    Since this is a job that existed before the OS field was added to the jobspec, the field is there, but has no value.

38. Create a new job and change its OS field value to the same value you used in the previous exercise.

    ```
    p4 job
    ```

    Edit the OS field:

    ```
    OS: <add a valid value>
    ```

39. Search for all jobs containing the OS value you used in the previous exercises.

    ```
    p4 jobs -e "OS=<value you used>"
    ```


Lab Set 7: Maintenance

Your objectives for this exercise:

- To edit database records owned by other users
- To perform database cleanup operations
- To obliterate files from Perforce
- To archive some binary files

40. Delete the client workspace spec named `raj-fir`.

```
p4 client -f -d raj-fir
```

41. Modify the "FullName" field in gale's user record.

```
p4 user -f gale
```

Change the FullName field to something like:

```
FullName: Gale McMullen
```

42. List and then delete the client workspace specs belonging to user `ines`.

```
p4  clients -u ines
p4 client -f  -d  ines-rose
```

...etc.

43. List all Perforce users. Delete user `ines`.

```
p4 users
p4 user -f -d ines
```

44. List all entries in the protections table for user `ines` and make appropriate changes to delete user `ines` from any listed entries.

```
p4 protects -u ines
```

User ines is in the perfusers group.

```
p4 group perfusers
```

*Delete **ines** from the **Users:** field.*

45. Unload the label named `www-live`.

```
p4 labels
p4 unload -f -l www-live
p4 labels
```

46. Reload the unloaded label named `www-live`.

```
p4 labels
p4 reload -f -l www-live
p4 labels
```


47. Obliterate the //depot/Talkhouse/rel1.0/... files.

*First, preview the **obliterate** command:*

```
p4 obliterate //depot/Talkhouse/rel1.0/...
```

Then, run the command:

```
p4 obliterate -y  //depot/Talkhouse/rel1.0/...
```

48. Obliterate revisions 1 through 3 of //depot/Jam/MAIN/src/jamMR.html

View current revisions:

```
p4 filelog //depot/Jam/MAIN/src/jamMR.html
```

*Now preview the **obliterate** command:*

```
p4 obliterate //depot/Jam/MAIN/src/jamMR.html#1,3
```

Then, run the command:

```
p4 obliterate -y //depot/Jam/MAIN/src/jamMR.html#1,3
```

Re-check file revisions:

```
p4 filelog //depot/Jam/MAIN/src/jamMR.html
```


Lab Set 8: Monitoring

49. What value is your monitor counter set to?

```
p4 configure show monitor
```

50. Set the monitor counter to level 1.

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

51. After changing the monitor settings, find out what processes are currently running against the Perforce Server.

```
p4 monitor show –a –e -l
```

52. Check to see if any database tables are currently locked.

```
p4 lockstat
```

Congratulations!

You have completed the Perforce Training Course exercises. We hope this course material has prepared you to use Perforce with confidence and ease.