Advanced Enterprise Administration for Perforce

These exercises follow on from the Introduction to Administration Exercises and assume the same initial setup and local test repositories, including changes to configuration carried out during that course. If your environment has been refreshed, your instructor can advise additional steps required prior to carrying out the following exercises.

Lab Set E1: Advanced Maintenance

1. Create an archive depot called *archive.*

```
p4 depot archive

Set The 'Type:' value to: archive
```

2. The files in //depot/Misc/manuals/… are not used at the moment. Archive them in your new archive depot.

```
p4 archive -D archive //depot/Misc/manuals/...
```

3. We need //depot/Misc/manuals/triggers.doc again. Restore it from your archive depot.

```
p4 restore -D archive  //depot/Misc/manuals/triggers.doc
```


Lab Set E2: Offline Checkpoint

Your objective for this exercise:

- Create an offline checkpoint together with rotating the live journal
- Optionally restore the database files and replace the live server db.*

1. Ensure your current environment is pointing at the 1666 server – use “set” or ”p4 set” or “p4 set P4CONFIG” to ensure this.

2. Checkpoint your server on port 1666

   ```
   p4 admin checkpoint

       or

   p4d -r c:\p4train -jc
   ```

   Note the number of the resulting checkpoint to use below (NNN)

3. Create an offline directory

   ```
   mkdir c:\p4train\offline_db
   ```

4. Recover from the checkpoint into that offline directory

   ```
   p4d -r c:\p4train\offline_db -jr c:\p4train\checkpoint.NNN
   ```

   Where NNN is the number of the checkpoint previously created.

5. Submit a change to your 1666 server, and sync and “un-sync” the workspace – see steps below.

   Create a new workspace (e.g. **`test-ws`**) with a map for **`//depot/`**…

   ```
   p4 client test-ws
   ```

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

   ```
   Root: c:\test-ws
   View:
         //depot/... //test-ws/...
   ```

   In P4V do a Get Latest (or **`p4 sync`** on command line)

   ```
   p4 -c test-ws sync
   ```

   This will create a number of records in your db.have table.
   Note the size of your db.have file in c:\p4train

   ```
   dir c:\p4train\db.have
   ```

   “Unsync” your workspace files (sync to revision 0 or “Remove from workspace”)

   ```
   p4 -c test-ws sync #0
   ```

   Note the size of your db.have file in c:\p4train

   ```
   dir c:\p4train\db.have
   ```


6. Rotate the journal on the 1666 server (Hint: don't checkpoint it – so use "**-jj**" rather than "**-jc**" option to p4d) – and note the number of the resulting rotated journal

```
p4d -r c:\p4train -J journal -jj
```

7. Apply that journal to your offline database

```
p4d -r c:\p4train\offline_db -jr c:\p4train\journal.NNN
```

Where NNN is the number of the journal previously created.

8. Checkpoint the offline database – but without incrementing the journal counter (Hint: use "**-jd**" instead of "**-jc**")

```
p4d -r c:\p4train\offline_db -jd  myoffline.ckp
```

9. Backup the db.* files from your offline database and then recreate them from the checkpoint.

```
cd c:\p4train\offline_db
mkdir save
move db.* save
p4d -r . -jr myoffline.ckp
```

10. Stop your server on 1666

```
p4 admin stop
```

11. Backup the db.* files from **c:\p4train** and move the db.* files from your offline directory to **c:\p4train**

```
cd c:\p4train
mkdir save
move db.* save
move c:\p4train\offline_db\db.* .
```

12. Recover the live journal into the **c:\p4train** directory

```
p4d -r . -jr journal
```

13. Restart the 1666 server and check that everything looks OK.

```
server.bat
```

14. Compare the size of db.have with the size you noted previously

```
dir c:\p4train\db.have
```


Lab Set E3: Broker

Your objective for this exercise:

- Create a broker, and configure it to disallow the 'p4 obliterate' command entirely.

15. Check and if necessary set the security level for your server to 3.

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

16. Create a user account on your server for the broker to connect to: user type `service` named `svc_p4broker`. Then create a group named `Automation.G` with a timeout value set to `unlimited`.

    Then, create a folder to store the Broker executables, configuration and logfiles:

```
MKDIR C:\P4Broker
COPY C:\p4train\p4broker.exe C:\P4Broker\p4broker.exe
COPY C:\p4train\p4broker.exe C:\P4Broker\p4brokers.exe
CD C:\P4Broker
```

    Alternately, acquire the p4broker.exe executable from:

```
ftp://ftp.perforce.com/perforce/r14.1/bin.ntx86/p4broker.exe
      (replace r14.1 with required version if different)
```

    Generate the default configuration file, and adjust:

```
p4broker –C > p4broker.cfg
```

    Use Notepad to edit p4broker.cfg. Change these values in the file:

```
directory          = C:\P4Broker;
service-user       = "svc_p4broker";
ticket-file        = "C:\P4Broker\P4Tickets.txt";
```

    Review other settings. Then, add this block at the end of the file:

```
# Forbid the use of 'p4 obliterate'.
command: obliterate
{
    action  = reject;
    message = "p4 obliterate disabled. This will be reported!";
}
```

    Next, tell the Windows service where to find the broker config file:

```
p4 set –S P4Broker P4BROKEROPTIONS="-c
      C:\P4Broker\p4broker.cfg"
```

    Next, create, start, and test the Windows service for executing the broker:

```
svcinst create –n P4Broker –e C:\P4Broker\p4brokers.exe
net start P4Broker
p4 –p 1667 info
```


17. Accessing the **broker** (port 1667), try to obliterate the file

    ```
    //depot/Misc/Marketing/valuations.xls
    ```

    Note: it is **always** good practice to initially run the obliterate command in '*report*' mode

    ```
    p4 -p 1667 obliterate //depot/Misc/Marketing/valuation.xls
    ```

    Note the error message (as entered in the broker.cfg file)

18. Now try obliterating the above file directly in the server;

    Initially in report mode:

    ```
    p4 -p 1666 obliterate //depot/Misc/Marketing/valuation.xls
    ```

    Finally force the obliterate:

    ```
    p4 -p 1666 obliterate -y //depot/Misc/Marketing/valuation.xls
    ```

Lab Set E4: Replication

Your objective for this exercise:

- Create a read-only only replica of your 1666 server (see the "Do It Yourself" lab).
- OR, just play with a replica that is already configured in your virtual lab (see the "Just Playing Around" lab).

Do It Yourself

19. Create a user account of type **`service`** named **`replica1`**. Then create a group named **`replica_group`** with a timeout value set to **`unlimited`**, and that has the **`replica1`** in its user list. Set the password for the **`replica1`** user to: **`Replicapass!`**

    Give the **`replica_group`** user a 'super' access in the protection table of the master.

20. Set up the replica configuration variables for our new replication server called "Replica1". You will need to set several configuration variables (Hint: create a local .bat file with notepad and edit it before running it)

```
p4 configure set Replica1#P4TARGET=localhost:1666
p4 configure set Replica1#P4PORT=1999
p4 configure set Replica1#monitor=1
p4 configure set Replica1#P4TICKETS=C:\Replica1\tickets.txt
p4 configure set Replica1#journalPrefix=C:\Replica1\ckp
p4 configure set Replica1#db.replication=readonly
p4 configure set Replica1#lbr.replication=readonly
p4 configure set Replica1#serviceUser=replica1
p4 configure set Replica1#startup.1="pull -i 1"
p4 configure set Replica1#startup.2="pull -u -i 1"
p4 configure set server=3
p4 configure set P4LOG=log
```

21. Checkpoint your live server

```
p4 admin checkpoint
```

(can add –Z option to compress the checkpoint)

22. Create a new root directory for your replica here: **`C:\Replica1`**

    a. Restore the checkpoint in this directory
    b. Create a '**`ckp`**' subdirectory to hold checkpoints.

```
mkdir C:\Replica1 C:\Replica1\ckp
cd c:\Replica1
p4d –r c:\Replica1 -jr C:\p4train\checkpoint.n
```

where n is the latest available checkpoint in C:\p4train.
If checkpoint was compressed (in 21 above), add the –z parameter and a '.gz' extension to the checkpoint filename

23. Start the replica server; use the **–In** parameter to specify the server name

```
p4d –r C:\Replica1 –In Replica1 –J journal
```

24. Log the replica (service) user **replica1** into the **master** server to commence replication. (As the master and replica will typically be on separate hardware, this normally has to happen from the replica machine).

    Hint: open a new command prompt and set the environment variable P4TICKETS to point to a new tickets file in the **C:\Replica1** directory.

```
set P4TICKETS=C:\Replica1\tickets.txt
p4 –p 1666 –u replica1 login
```

25. As user **bruno**, check on the master server side that the replica has started the replica service by checking the log (hint: **p4 logtail**).

```
p4 –p 1666 logtail
```

26. Check on the replica that your pull processes are running. You will need to log in to the replica first. Hint: use the `p4 monitor show` command on the replica and look for the 'p4 pull' processes.

```
p4 –p 1999 monitor show -l
```

27. Update some data. For example, change/add the Description of the **bruno_ws** workspace spec on the master server, and then print out the same spec on the replica server.

```
p4 –p 1666 client bruno_ws

p4 –p 1999 print //specs/client/bruno_ws.p4s
      or
p4 –p 1999 client –o bruno_ws
```

28. Test that you cannot attempt to sync, or edit or submit any files to this replica server (but that you still can if you edit them on the master server).

```
p4 –p 1999 –c bruno_ws sync #0
p4 –p 1999 –c bruno_ws sync //depot/Jam/MAIN/…
p4 –p 1999 –c bruno_ws edit //depot/Jam/MAIN/src/jam.c
```

29. Check to see if there are any (version) files in the depot sub-directory under **c:\Replica1**

```
dir /ad c:\replica1
```

30. Run a verify with **–qt** flag on **//depot/jam/MAIN/…** to schedule the pulling of all appropriate versioned files. Wait a few moments, and then check that these files have now been created in your replica directory (either using Windows Explorer or cmd commands)

```
p4 –p 1999 verify –qt //depot/jam/MAIN/...
dir /s/b c:\replica1\depot
```

    (In a real installation you would copy all the versioned files from your master server to the replica server – not use **p4 verify –qt**)

Just Playing Around

A standard Perforce server with service named **RepFunMaster** has been set up on your virtual lab, running on port 10001. It has a replica with service named **RepFunHAReplica** running on port 10002.

To play with it, just point P4V to the different P4PORTs (10001 and 10002) and play around. Or, for command line experimentation, do this:

```
set P4CONFIG=P4Config.txt
cd C:\RepFun\master\ws
p4 info
cd ..\..\ha_replica
p4 info
```


Lab Set E5: Forwarding Replica and Build Replica

Your objective for this exercise:

- Turn your read-only replica into a forwarding replica and show that the behavior has changed
- Turn your forwarding replica into a build replica and show that you can create a new replica specific workspace

31. Change the Replica1 server specification (on the master server) to make it a forwarding-replica (see P4Dist manual)

```
p4 server Replica1
```

Set following line:

```
Services:  forwarding-replica
```

32. Set the appropriate configurable to make **`Replica1`** a forwarding replica (see P4Dist manual)

```
p4 configure set Replica1#rpl.forward.all=1
```

33. Restart the replica

```
p4 -p 1999 admin restart
```

34. Test that you can now edit and submit files when communicating with the replica server.

```
p4 -p 1999 -c bruno_ws sync #0
p4 -p 1999 -c bruno_ws sync //depot/Jam/MAIN/…
p4 -p 1999 -c bruno_ws edit //depot/Jam/MAIN/src/jam.c
```

35. Change the Replica1 server specification to make it a build-server (see P4Dist manual)

```
p4 server Replica1
```

Set following line:

```
Services:  build-server
```

36. Remove the configurable which makes **`Replica1`** a forwarding replica (see P4Dist manual)

```
p4 configure unset Replica1#rpl.forward.all
```

37. Restart the replica

```
p4 -p 1999 admin restart
```

38. Create a new workspace **`build-ws`** which is tied to the replica (Hint: set ServerID field) specifying appropriate root and view. Note: due to restrictions relating to client workspaces within the build replicas, your current client setting (p4 –p 1999) will impact the command required here.

```
p4 -p 1999 -c build-ws client build-ws
```


Set following line:

```
ServerID:  Replica1
```

39. Sync your new workspace

```
p4 -p 1999 -c build-ws syncls -la db.
```

40. Check if the workspace is visible on the main server and if you can see which files are synced within it by talking to the main server.

```
p4 -p 1666 clients
p4 -p 1666 files @build-ws
```

41. Check if there is a replica specific file called **db.have.rp** within the replica database directory (not in the sync'd workspace)

```
dir c:\Replica1\db.have.rp
```

*This file is used by the replica in place of **db.have** which is the database file that normally records information on which files are sync'd to each workspace.*

Lab Set E6: Security

Your objective for this exercise:

- Put some basic commercial security measures in place. This lab requires the command line interface.

42. Check and if necessary set the security level to 3.

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

43. Make it so new user accounts can be created only by super users.

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

44. Make the default changelist type be 'restricted', rather than 'public.'

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

45. Security Discussions:

    Select security topics of interest and discuss. Below are some sample topics:

- Perforce built-in SSL vs. Do-It-Yourself SSH Tunnels
- External Authentication (Active Directory/LDAP) and Password Snooping
- Single-Sign On Integrations
- Two-Factor Authentication
- Security Level Segregation and Air Drops

Lab Set E7: Structured Logging

Your objective for this exercise:

- Enable all structured logs. This isn't something you would normally do on a production server; instead you would select only those logs you intend to monitor.

46. Enable recording of all seven structured log types.

```
p4 configure set serverlog.file.1=all.csv
p4 configure set serverlog.file.2=commands.csv
p4 configure set serverlog.file.3=errors.csv
p4 configure set serverlog.file.4=audit.csv
p4 configure set serverlog.file.5=track.csv
p4 configure set serverlog.file.6=user.csv
p4 configure set serverlog.file.7=events.csv
```

47. Configure each log to grow no larger than 30Mb, and keep no more than 5 iterations of logs.

```
p4 configure set serverlog.maxmb.1=30
p4 configure set serverlog.maxmb.2=30
p4 configure set serverlog.maxmb.3=30
p4 configure set serverlog.maxmb.4=30
p4 configure set serverlog.maxmb.5=30
p4 configure set serverlog.maxmb.6=30
p4 configure set serverlog.maxmb.7=30
p4 configure set serverlog.retain.1=5
p4 configure set serverlog.retain.2=5
p4 configure set serverlog.retain.3=5
p4 configure set serverlog.retain.4=5
p4 configure set serverlog.retain.5=5
p4 configure set serverlog.retain.6=5
p4 configure set serverlog.retain.7=5
```

48. Rotate the logs manually once

```
CD C:\p4train
DIR /OD *.csv
p4 logrotate
DIR /OD *.csv
```

See how the logs are rotated.

49. Use the 'p4 logappend' command to write something to the user log.

```
p4 logappend -a This is my message to the log.
TYPE user.csv
```


Lab Set E8: Simple Admin Command Automation

Your objective for this exercise:

- Try a few script-like commands using the power of the p4 command line.

50. Print out a list of usernames and email addresses.

    Windows:

    ```
    for /f "tokens=1,2" %a in ('p4 users') do @echo %a %b
    ```

    Unix (bash):

    ```
    p4 users | cut -d" " -f1,2 | while read c; do echo $c; done
    ```

51. Create a client workspace (which is a form command and will normally start up and editor) from a text file.

    ```
    p4 –c new-ws client –o | p4 client –i
    ```

    Check that the client workspace **new-ws** has been successfully created.

52. Create a new group all_users and via the command line add all users to that group.

    Save a basic group specification file to g.txt

    ```
    p4 group -o all-users > g.txt
    ```

    Append a list of all current Perforce users to that file (remembering to add at least one space before each username):

    ```
    for /f "tokens=1" %a in ('p4 users') do @echo    %a >> g.txt
    ```

    Create the group by inputting from the file.

    ```
    p4 group -i < g.txt
    ```

    Check that the group has been successfully created.

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

