Bacula Restores

Useful Bacula commands

Watch restore job:

```
watch -n1 'echo "status client client=\"srv-IP_IP_IP_IP\"" | bconsole'
```

Check backup client status via bconsole:

```
status client
```

==========

Bacula File Restore

---

1. SSH onto backup server.

---

2. Initiate a restore

Run the following commands to initiate a restore using the backup jobID:

Bconsole > restore > 3 > enter Job IDs, comma seperated.

---

3. Mark files to be restored

Once the directory tree has been built we need to mark the /var/lib/mysql directory, and initiate the restore:

```
mark /path/to/file
```

Run 'done' once all required files have been marked

```
done
```

Select option 9 (where):

```
9
```

Once here, you need to enter the directory we're restoring TO:

```
/root/restore_TICKETNUMBER
```

==========

Bacula DB Restore

https://kb.ukfast.net/Restore_MySQL_from_Bacula

---

1. SSH onto backup server.

---

2. Initiate a restore

Run the following commands to initiate a restore using the backup jobID:

Bconsole > restore > 3 > enter Job IDs, comma seperated.

---

3. Mark files to be restored

Once the directory tree has been built we need to mark the /var/lib/mysql directory, and initiate the restore:

```
mark /var/lib/mysql
```

Run 'done' once all required files have been marked

```
done
```

Select option 9 (where):

```
9
```

Once here, you need to enter the directory we're restoring to (on client server):

```
/root/restore_TICKETNUMBER
```

4A. Starting 2nd MySQL Instance

Once MySQL has been restored onto client server, we then need to start a 2nd instance of mysql so that we can dump the required databases.

The below command starts the 2nd MySQL instance (You need to replace /mnt/mysql with the path we've restored MySQL to; in my example this is /root/restore_TICKETNUMBER).

```
/usr/sbin/mysqld --socket=/tmp/mysql2.sock --datadir=/mnt/mysql --skip-networking --pid-file=/tmp/mysql2.pid --user=mysql --skip-grant-tables
```

Running this command will take over your session, meaning that you'll have to leave this running and open a fresh SSH session.

If you're encountering errors when attempting to start the 2nd instance, it would be worth having a google of the errors. If you're still having issues, please see below:

4B. Troubleshooting restore MySQL startup issues

If you're unable to start the 2nd MySQL instance after troubleshooting, there's 2 main options:

1. Delete the restored content on the client server and start the restore again
2. If this still doesn't work, you can try to start the 2nd MySQL instance using innodb_force_recovery.

   There are 6 levels of force recovery options, see here for full details.

```
/usr/sbin/mysqld --socket=/tmp/mysql2.sock --datadir=/mnt/mysql --skip-networking --pid-file=/tmp/mysql2.pid --user=mysql --skip-grant-tables --innodb_force_recovery=X
```

You'll need to replace 'x' on the above command with your chosen level, I'd advise starting with level 1 and moving up until the MySQL instance is started. Anything above level 4 can cause permanent data corruption, so it's ideal if we can avoid this.

If innodb_force_recovery is used, please note down the level used and tell the client about this.

https://dev.mysql.com/doc/refman/8.0/en/forcing-innodb-recovery.html

If errors are being shown regarding the existing MySQL configuration, you can attempt to launch the 2nd instance with the --no-defaults flag, this essentially tells MySQL to launch with the default settings :

```
/usr/sbin/mysqld --no-defaults --socket=/tmp/mysql2.sock --
datadir=/home/restore_4450884/mysql --skip-networking --pid-file=/tmp/mysql2.pid --
user=mysql --skip-grant-tables
```

---

5. Dumping a database

Now that we've got the restored MySQL instance running, we can look to validate the data we require is present, and dump out the database.

Connect to restored MySQL instance:

```
mysql -S /tmp/mysql2.sock
```

Check that the required database is present:

```
SHOW DATABASES;
```

Once we've confirmed that the databases we need is present, we can look to dump out the database into a file:

Run the following command, substituting 'databasename' with the name of your required database, and also updating 'database_restoreTICKETNUMBER' with the appropriate ticket number.

```
mysqldump -S /tmp/mysql2.sock databasename --events --triggers --routines >
/root/database_restoreTICKETNUMBER.sql
```

Other options for dumping a database/s

All data and databases :

```
mysqldump -S /tmp/mysql2.sock --all-databases --events --triggers --routines >
/root/database_restoreTICKETNUMBER.sql
```

To dump several but not all databases (substitute databasename1,databasename2 etc) :

```
mysqldump -S /tmp/mysql2.sock --databases databasename1 databasename2 --events --triggers --
routines > /root/database_restoreTICKETNUMBER.sql
```

For a specific table only (substitute databasename and tablename)

```
mysqldump -S /tmp/mysql2.sock databasename tablename > /root/database_restoreTICKETNUMBER.sql
```

If you're encountering errors when attempting to dump the required data, you can use the -f flag to ignore errors.

---

6. Cleaning up

Once the dump is complete, terminate the 2nd instance:

```
mysqladmin -S /tmp/mysql2.sock shutdown
```

Remove the restored /var/lib/mysql directory.

```
rm -rf /root/restore_TICKETNUMBER
```

Revision #9
Created 2024-02-26 22:16:09 UTC by Daniel
Updated 2024-06-05 01:13:26 UTC by Daniel