A Database Story: Backup then Recovery [NG]

*Author: Daniel Killam*  
**Framework Category:** Operate and Maintain  
**Specialty Area:** Network Services  
**Work Role:** Network Operations Specialist  
**Task Description:** Develop and implement network backup and recovery procedures. (T0065)  
**Scenario**

It seems that our new intern has deleted some important database files needed for an upcoming project. This incident has caused us to realize that we are severely lacking in our standard recovery and backup procedures. As such , we need you to design and implement a plan for backing up and restoring data critical to our business. After designing these procedures, they need to be tested in order to ensure their functionality.

**Additional Information**

More details and objectives about this challenge will be introduced during the challenge meeting, which will start once you begin deploying the challenge.  
  
You will be able to check your progress during this challenge using the check panel within the workspace once the challenge is deployed. The checks within the check panel report on the state of some or all of the required tasks within the challenge.  
  
Once you have completed the requested tasks, you will need to document the methodology you used with as much detail and professionalism as necessary. This should be done on the documentation tab within the workspace once the challenge is deployed. Below the main documentation section be sure to include a tagged list of applications you used to complete the challenge.  
  
Your username/password to access all virtual machines and services within the workspace will be the following...  
Username: playerone  
Password: password123  
  
The username/password used to access the Firewall's web interface within the workspace will be the following...  
Username: admin  
Password: password123

haven't been backing anything up. In order to make sure we don't lose data, we should be backing up critical data often. We need you to write up some backup and restoration procedures with respects to database backups. I'd like this task done as soon as possible.

Create a sql dump of our 'wordpress' MySQL database and move it to the Backup server first.

When you've finished that go ahead and use the MySQL server on the Backup machine and restore the wordpress database to said server.

* Use the steps that you take while doing these things in order to write up a set of procedures that can be used by other employees to replicate this process.

@playerone, the MySQL server is hosted on Database, and I went ahead and created a folder on Backup at filepath /ftp/public/prodbackup/ in which I'd like you to put a sql dump file of the 'wordpress' database. Also, if you can name the file 'dump.sql' so we can take a look at it if we need to, that would be great. Make sure that the dump includes a backup of all tables. Also, don't do anything else to modify the file contents. Keep it a rather plain SQL dump.

Solution:

On Database Server:

* mysqldump -u playerone -p --default-character-set=utf8 --result-file=dump.sql wordpress (export database wordpress to dump.sql)
* scp dump.sql playerone@172.16.30.79:/home/playerone (copy dump.sql to home directory of playerone on backup server.

On Backup Server:

* mv /home/playerone/dump.sql /ftp/public/prodbackup (Move the dump.sql to /ftp/public/prodbackup)
* Create database wordpress on Backup Server:
  + sudo mysql
  + CREATE DATABASE wordpress;
* Import into wordpress database from dump.sql:
  + mysql --user=root --password=root --default\_character\_set utf8 wordpress < dump.sql

--default-character-set=utf8: This insures UTF8 is used for each field

--result-file=file.sql: This option prevents the dump data from passing through the Operating System which likely does not use UTF8. Instead it passes the dump data directly to the file specified.

Extra command to maneuver around mysql:

* mysql -u username -p
* SHOW DATABASES; (list all the databases)
* USE mydatabase; (choose a database)
* SHOW TABLES; (list all the table)
* SELECT \* from table;
* CREATE DATABASE mydatabase (create new database)
* DROP DATABASE mydatabase (delete a database)
* SELECT User, Host FROM mysql.user; (List all the users in databases)
* RENAME USER ‘root’@’location’ TO ‘newusername’@’location’; (Rename the root account)
  + Note: location could be ‘localhost’ ‘IP address’ ‘%’ (% for all hosts)
* ALTER USER ‘root’@’location’ IDENTIFIED BY ‘newP@ssword’;
* CREATE USER ‘username’@’location’ IDENTIFIED WITH mysql\_native\_password BY ‘p@ssword’ (Create user named ‘henry’ that can access databasae from ‘location’, that can be authenticated by authentication plugin ‘mysql\_native\_password’, with the password of ‘p@ssword’)
  + Note: authentication plugin ‘caching\_sha2\_password’ is recommended.
  + Authentication plugin may prevent the use of PhpMyAdmin to log in from remote location.
* GRANT ALL PRIVILEGES ON \*.\* TO 'username'@'location' with grant option; (grant all Permissions to user to whatever databases and tables, along with GRANT option: ability to grant or remove privileges to other users)
* FLUSH PRIVILEGES; (to reload GRANT tables to apply any change)

PhpMyAdmin: UI to manage SQL Server.