Website Hosting - Support #85

Backing Up with Rsnapshot & Snapshots Of MySQL Databases

03/02/2013 04:59 PM - Daniel Curtis

Status:	Closed	**Start date:**	03/02/2013
Priority:	Normal	**Due date:**	
Assignee:	Daniel Curtis	**% Done:**	100%
Category:		**Estimated time:**	1.00 hour
Target version:		**Spent time:**	1.00 hour

Description

This howto will show you how to install and set up Rsnapshot, enable archiving of snapshots and how to back up MySQL databases on Debian.

1. Install rsnapshop

The following guide will be ran as root:

```
sudo su
```

```
apt-get install rsnapshot
```

2. Edit config file:

```
nano /etc/rsnapshot.conf
```

- Set snapshot_root to path where you want to keep the backups

```
snapshot_root   /var/cache/rsnapshot/
```

- Set up the list of directories/files to back up

```
backup  /etc/           localhost/
```

Backup Intervals

This section of the configuration file is used only to define labels for intervals and how many snapshots of each level to keep. How often the snapshots are made is configured and run via cron. The keyword interval is followed by an alphanumeric label, followed by a number, signifying how many intervals (snapshots) to keep. The interval labels must be unique and in ascending order, smallest interval must be listed first.

In following example, after 6 "hourly" snapshots the oldest "hourly" is deleted. The top entry (in this case "hourly") is copied from the source, while the remaining entries simply link to the latest snapshot from one level above.

In other words, to keep six backups a day (four-hour interval), seven daily backups (one week), and four weekly backups (one month), specify:

```
interval        hourly  6
interval        daily   7
interval        weekly  4
```

The interval labels "hourly", "daily", "weekly" can be changed to suit your needs, for example "daysago", "weeksago" and "monthsago".

Each time rsnapshot hourly is executed, by hand or by cron, it will create a new snapshot, rotate the old ones, and retain the 6 most recent (hourly.0 - hourly.5).

Backup Points

Example of a backup point inside configuration file:

```
backup      /etc/       localhost/
```

Backup points begin with the word backup, following /etc/ is the full path of the directory that will be backed up and localhost is a directory inside snapshot_root. You can change localhost to anything associating to the server like server's fully qualified domain name.

In addition to full paths on the local filesystem, you can also backup remote systems using rsync over ssh. If you have ssh installed and enabled (via the cmd_ssh parameter), you can specify a path like:

```
backup      root@example.com:/etc/     example.com/
```

Have in mind that you must have key-based logins enabled for the root user at example.com, without passphrases in order this to function properly.

Backup Scripts

You can find many examples in the utils directory which on Debian is located at /usr/share/doc/rsnapshot/examples/utils/.

Backup scripts are executed with each lowest interval.

Executing Backups Via Cron

The cron file is located at

- /etc/cron.d/rsnapshot

The default contents:

```
# 0 */4         * * *         root    /usr/bin/rsnapshot hourly
# 30 3       * * *         root    /usr/bin/rsnapshot daily
# 0  3       * * 1         root    /usr/bin/rsnapshot weekly
# 30 2       1 * *         root    /usr/bin/rsnapshot monthly
```

You have to uncomment lines to activate the backups.

The first tells cron to execute cron-apt every 4 hours which matches the hourly interval settings in the rsnapshot.conf.

Test configuration

Every time you make a change in the config file do a configtest

```
rsnapshot configtest
```

PLEASE BE AWARE OF THE FOLLOWING RULES:

- Configuration file requires tabs between elements, spaces represent argument for scripts
- Directories require a trailing slash, example:

  right: /home/

  wrong: /home

Archiving snapshots

```
cp /usr/share/doc/rsnapshot/examples/utils/rsnaptar /usr/local/bin/
```

Make sure your backup scripts are owned by root, and **not** writable by anyone else.

```
chown root.root /usr/local/bin/rsnaptar
chmod o-w /usr/local/bin/rsnaptar
```

Edit script and set directory paths

```
nano /usr/local/bin/rsnaptar
```

Set TAR_DIR to path where snapshot will be archived, and SNAPSHOT_DIR to where the daily snapshot is located:

```
TAR_DIR="/home/user/dvd_backup"
SNAPSHOT_DIR="/var/cache/rsnapshot/daily.0"
```

Please note that the hourly cycle needs to complete in order for the daily snapshot to be created.

This script supports gpg encrupting of files, to disable it comment out following fline in /usr/local/bin/rsnaptar:

```
GPG="/usr/bin/gpg"
```

Scheduling Execution Via Cron

```
nano /etc/cron.daily/rsnaptar
```

Add:

```
#!/usr/bin/env bash
/usr/local/bin/rsnaptar root@example.com
```

Make /etc/cron.daily/rsnaptar executable

```
chmod +x /etc/cron.daily/rsnaptar
```

Notes:
- This howto does not use the GPG option to encrypt files, if this is a necessity than I can update the howto to include file encryption for better security,
- By default script is run standalone, but it could be modified to run as a backup_script via rsnapshot itself,

Rsnapshot: Backing Up MySQL Databases

Install mysql-client, if not installed, this will provide the mysqldump utility:

```
apt-get install mysql-client
cp /usr/share/doc/rsnapshot/examples/utils/backup_mysql.sh /usr/local/bin/
```

Make sure your backup scripts are owned by root, and not writable by anyone else.

```
chown root.root /usr/local/bin/backup_mysql.sh
chmod o-w /usr/local/bin/backup_mysql.sh
```

This script is designed only to back up all databases to a single file.

Edit script:

```
nano /usr/local/bin/backup_mysql.sh
```

Replace

> /usr/bin/mysqldump --all-databases > mysqldump_all_databases.sql

with

> /usr/bin/mysqldump --defaults-file=/etc/mysql/debian.cnf --all-databases > mysqldump_all_databases.sql

Create debian-sys-maint user

- Log into MySQL as root MySQL user

```
mysql -u root -p
```

Enter password

- Create debian-sys-maint user

```
GRANT ALL PRIVILEGES ON *.* TO 'debian-sys-maint'@'localhost' IDENTIFIED BY 'SuperSecretPassword';
```

Relacing SuperSecretPassword with an appropriate password.

- Match the password in /etc/mysql/debian.cnf

```
# Automatically generated for Debian scripts. DO NOT TOUCH!
[client]
host     = localhost
user     = debian-sys-maint
password = SuperSecretPassword
socket   = /var/run/mysqld/mysqld.sock
[mysql_upgrade]
user     = debian-sys-maint
password = SuperSecretPassword
socket   = /var/run/mysqld/mysqld.sock
basedir  = /usr
```

Replacing SuperSecretPassword with the password used on the MySQL server

If you need to dump a single database use line below:

    /usr/bin/mysqldump --defaults-file=/etc/mysql/debian.cnf DATABASE > DATABASE.SQL

Add Script To Rsnapshot

Edit rsnapshot config:

```
nano /etc/rsnapshot.conf
```

Under **BACKUP POINTS** / **SCRIPTS** add the following:

```
backup_script   /usr/local/bin/backup_mysql.sh  localhost/mysqldump
```

Test config:

```
rsnapshot configtest
```

Resources:

http://www.howtoforge.com/set-up-rsnapshot-archiving-of-snapshots-and-backup-of-mysql-databases-on-debian

Related issues:

Related to Website Hosting - Bug #86: Error While Backing Up MySQL Using rsna...	**Closed**	**03/02/2013**

History

#1 - 03/02/2013 05:29 PM - Daniel Curtis

- Description updated

#2 - 03/02/2013 06:12 PM - Daniel Curtis

Is also Document Backing Up with Rsnapshot & Snapshots Of MySQL Databases for GNet Cyber Solutions

#3 - 03/02/2013 06:34 PM - Daniel Curtis

- Description updated

#4 - 03/02/2013 06:49 PM - Daniel Curtis

- Description updated

#5 - 03/02/2013 07:49 PM - Daniel Curtis

- Description updated

#6 - 03/02/2013 08:20 PM - Daniel Curtis

- Description updated

#7 - 07/25/2013 02:04 PM - Daniel Curtis

It is also nice to have a list of packages that are installed. This can be produced using:

```
dpkg --get-selections
```

Backup a list of installed application packages

As I am doing periodic updates I have created a simple bash script to dump the current packages installed to /var/backups/installed_packages.log and set the script to run in cron every day:

- dpkg-manifest.sh

```
dpkg --get-selections > /var/backups/installed_packages.log
```

Restore a list of install application packages

In the even I need to restore a server using installed_packages.log:

```
sudo dpkg --clear-selections
sudo dpkg --set-selections < ~/backup/installed_packages.log
sudo apt-get dselect-upgrade
```

sudo dpkg --clear-selections will mark all current packages installed for removal, that way when you restore your saved package list the packages that are not on the list will be removed from your system.