Backup PostgreSQL to Amazon S3

PostgreSQL # AWS

2015-01-13 · 1 min read

This article describes an easy way to backup PostgreSQL database to Amazon S3 using `s3cmd` and `cron`.

Install and configure `s3cmd`

For Debian/Ubuntu

```
apt-get install s3cmd
```

For RedHat/CentOS

```
yum install s3cmd
```

For OSX

```
brew install s3cmd
```

Set Amazon access & secret keys using `--configure` option. Enable encryption for data transferring as well as `HTTPS`.

```
s3cmd --configure
```

Backup the database

```
PGPASSWORD=YOUR_PASSWORD pg_dump -Fc --no-acl -h localhost -U YOUR_USER
```

Push to Amazon S3

Create a bucket

```
s3cmd mb s3://YOUR_NAME
```

Push created backup to that bucket

```
s3cmd put backup.dump s3://YOUR_NAME --encrypt
```

Automate

```
#!/usr/bin/env bash

APP=$1

DB_NAME=$2
DB_USER=$3
DB_PASS=$4

BUCKET_NAME=<YOUR_NAME_HERE>

TIMESTAMP=$(date +%F_%T | tr ':' '-')
TEMP_FILE=$(mktemp tmp.XXXXXXXXXX)
S3_FILE="s3://$BUCKET_NAME/$APP/$APP-backup-$TIMESTAMP"

PGPASSWORD=$DB_PASS pg_dump -Fc --no-acl -h localhost -U $DB_USER $DB_NA
s3cmd put $TEMP_FILE $S3_FILE --encrypt
rm "$TEMP_FILE"
```

Save the file to `backup.sh` and make it executable with `chmod +x`. Test if it works with properly set positional parameters, e.g.

```
./backup.sh acme acme_db acme_user acme_pass
```

Add a cron entry using `crontab -e` and set it to run daily at 11pm

```
* 23 * * * /path/to/backup.sh acme acme_db acme_user acme_pass
```

TABLE OF CONTENTS