How to Backup And Restore Uptime Kuma Database in Docker?

written by sysadmin | 7 January 2026

The previous article explained how to install Uptime Kuma using Docker on Linux. This article will explain how to back up and restore the Kuma uptime database in Docker.

Problem

How to back up and restore the Uptime Kuma database in Docker?

Solution

Below are the steps to back up and restore the Uptime Kuma database in Docker:

A. Database SQLite

Here is the method to back up and restore a SQLite database in Docker:

1. Backup database

If you want to back up the Uptime Kuma database in Docker, you can run the command below to get the Kuma database on your host:

```
docker run --rm
  -v uptime-kuma:/data
  -v $(pwd):/backup
  alpine tar czf /backup/kuma-backup.tar.gz /data
```

After that, look in your current folder; the database should appear like the image below:

```
sysadmin@docker:~$ docker run --rm \
  -v uptime-kuma:/data \
  -v $(pwd):/backup \
  alpine tar czf /backup/kuma-backup.tar.gz /data
Unable to find image 'alpine:latest' locally
latest: Pulling from library/alpine
1074353eec0d: Pull complete
5c1f58ba4e0d: Download complete
644afed44dca: Download complete
Digest: sha256:865b95f46d98cf867a156fe4a135ad3fe50d2056aa3f25ed31662dff6da4eb62
Status: Downloaded newer image for alpine:latest
tar: removing leading '/' from member names
sysadmin@docker:~$ ls
crud  get-docker.sh  kuma-backup.tar.gz  main.zip
```

Back up the uptime Kuma database

However, if you want to back up automatically, then follow the steps below:

a. Create a backup folder on the host

Run the commands below to create a backup folder in the host:

```
sudo mkdir -p /opt/kuma-backup
sudo chown root:root /opt/kuma-backup
sudo chmod 700 /opt/kuma-backup
```

b. Run the backup script

Create a file, for example, backup-kuma.sh, in the root folder containing the bash script below:

```
#!/bin/bash
set -e

# CONFIG
CONTAINER_NAME="uptime-kuma"
VOLUME_NAME="uptime-kuma"
BACKUP_DIR="/opt/kuma-backup"
RETENTION_DAYS=14
DATE=$(date +"%Y%m%d-%H%M%S")
BACKUP_FILE="$BACKUP_DIR/kuma-backup-$DATE.tar.gz"

# Backup
docker run --rm \
  -v $VOLUME_NAME:/data:ro \
```

```
  -v $BACKUP_DIR:/backup \
  alpine \
  tar czf /backup/$(basename $BACKUP_FILE) /data

# Cleanup old backups
find $BACKUP_DIR -type f -name "kuma-backup-*.tar.gz" -mtime +$RETENTION_DAYS
-delete
```

Save the file and make it executable using the command:

```
sudo chmod +x /root/backup-kuma.sh
```

If you run the script, the backup database file should be in the **/opt/backup** folder.

c. Set up a Cron Job

If you want to run the script every 2:30 AM, then on the crontab, write the script as shown in the image below

```
0 2 * * *       /root/backup-kuma.sh >> /root/kuma-backup.log 2>&1
```

2. Restore database

Before you restore the database, make sure the container has been running first. If you want to restore the Uptime Kuma database that you have previously backed up, then you can run the command below on the host:

```
docker run --rm \
  -v uptime-kuma:/data \
  -v /opt/kuma-backup:/backup \
  alpine \
  tar xzf /backup/kuma-backup-YYYYMMDD-HHMMSS.tar.gz -C /
```

If the restore process is complete, the hosts that were monitored in the previous container should be monitored again by the new container.

B. Database MariaDB

Here is the method to back up and restore a MariaDB database

in Docker:

1. Backup database

If you want to back up the MariaDB database in Docker, you can run the command below to get the database on your host:

```
docker exec mariadb \
  mysqldump -u root -p kuma > kuma.sql
```

Change the kuma with your database name. Or use the command below if you want to compress the result:

```
docker exec mariadb \
  mysqldump -u root -p kuma | gzip > kuma.sql.gz
```

You can use the script below to back up the database:

```
#!/bin/bash

# ==========================
# CONFIGURATION
# ==========================
CONTAINER_NAME="mariadb"
DB_USER="backup_user"
DB_PASSWORD="PASSWORD_DB"
DB_NAME="appdb"

BACKUP_DIR="/opt/backup/mariadb"
RETENTION_DAYS=7
DATE=$(date +"%Y-%m-%d_%H-%M-%S")
BACKUP_FILE="$BACKUP_DIR/${DB_NAME}_${DATE}.sql.gz"
LOG_FILE="$BACKUP_DIR/backup.log"

# ==========================
# PREPARATION
# ==========================
mkdir -p "$BACKUP_DIR"

echo "[$(date)] Backup started" >> "$LOG_FILE"

# ==========================
# BACKUP DATABASE
# ==========================
docker exec "$CONTAINER_NAME" \
  mysqldump -u"$DB_USER" -p"$DB_PASSWORD" "$DB_NAME" \
  --single-transaction \
```

```
  --quick \
  --routines \
  --triggers \
  --events \
  | gzip > "$BACKUP_FILE"

# ========================
# VALIDATION
# ========================
if [ $? -eq 0 ]; then
  echo "[$(date)] Backup SUCCESS: $BACKUP_FILE" >> "$LOG_FILE"
else
  echo "[$(date)] Backup FAILED" >> "$LOG_FILE"
  exit 1
fi

# ========================
# BACKUP ROTATION (DELETE OLD FILE)
# ========================
find "$BACKUP_DIR" -type f -name "*.sql.gz" -mtime +$RETENTION_DAYS -delete

echo "[$(date)] Old backups cleaned (>${RETENTION_DAYS} days)" >> "$LOG_FILE"
echo "[$(date)] Backup finished" >> "$LOG_FILE"
```

And you can insert the script above in the crontab.

2. Restore database

Before you restore the database, make sure the container has been running first. If you want to restore the database that you have previously backed up, then you can run the command below on the host:

```
docker exec -i mariadb \
  mysql -u root -p -e "CREATE DATABASE kuma;"

docker exec -i mariadb \
  mysql -u root -p kuma < kuma.sql
```

Note

If you have a failure to back up the database, you can see the logs in the **/root/backup-kuma.log** file.

References

fishparts.net
homelab.anita-fred.net