Water Data Project para Groots

Maintenance - Backups de base de datos (runbooks incluídos)

[**Alcance**](#_wu7ihw7e46bz) **1**

[**Configuración de Backups**](#_1s1wcb6fz8qg) **2**

[Instalación en kobo-install](#_bwx5wntt60de) 2

[Configuración en disco](#_jitwaevhd4az) 3

[Estructura del directorio Backups](#_va0qrhmgpegc) 3

[Archivo de configuración](#_f66s035p2nj7) 3

[Configuración con AWS S3](#_uy0jlhoooo2d) 4

[Requisitos](#_57onn8ui13s) 4

[Instalación en kobo-install](#_v2rgfnnhq6sq) 4

[Archivo de configuración](#_7beid6v4w4le) 5

[Revisión de logs](#_634ar0dyz5op) 6

[Desde kobo-docker](#_122p26qppmwm) 6

[**Restauración de Bases de Datos**](#_bcf8if96ih8i) **6**

[Acerca del archivo de backup](#_jklhqbsosyo) 6

[Restaurar MongoDB](#_vd97szbnfkiy) 7

[Restaurar PostgreSQL](#_b2eesu1bhgsq) 8

[Base de datos Postgres de KoboForms (KPI)](#_u1g3irn8s9v7) 8

[Base de datos Postgres de KoboCat](#_5rawvjffpoo6) 8

[Restaurar Redis](#_lxbglwf92k3o) 8

[Referencias adicionales](#_vhksg1a4e3xp) **9**

# Alcance

Proporciona la información de cómo configurar Backups y realizar Restores de las bases de datos manejadas por KoBoToolbox.

Se explican los procedimientos para instalaciones con backup a AWS S3 y con backup a disco local.

|  |
| --- |
| Se asume que la ruta de instalación (de ejemplo) está en:  [**/home/kobo\_user/kobotoolbox/]**  Sin embargo, la misma puede cambiar de acuerdo a la configuración final que se realice. |

# Configuración de Backups

## Instalación en kobo-install

La configuración de backups debe realizarse primero, independientemente de si el destino de los mismos es S3 o local.

El documento: ***Deployment - Runbook - Fase 1 (*** *Página 9****)*** es la referencia más completa de cómo realizar la configuración de backups.

sin embargo se muestran los pasos específicos en la configuración:

|  |
| --- |
| Do you want to activate backups?  1) Yes  2) No  [1]: 1 |

|  |
| --- |
| Do you want to activate backups?  1) Yes  2) No  [1]: 1 |

|  |
| --- |
| KoBoCat media backup schedule?  [0 0 \* \* 0]: 0 20 \* \* \* |

|  |
| --- |
| PostgreSQL backup schedule?  [0 2 \* \* 0]: 0 21 \* \* \*  En el ejemplo, cada día de cada mes de cada año a las 21:00. Se puede cambiar a cualquier valor que se ajuste a las necesidades de la entidad. |

|  |
| --- |
| MongoDB backup schedule?  [0 1 \* \* 0]: 0 22 \* \* \*  En el ejemplo, cada día de cada mes de cada año a las 22:00. Se puede cambiar a cualquier valor que se ajuste a las necesidades de la entidad. |

|  |
| --- |
| Redis backup schedule?  [0 3 \* \* 0]: 0 23 \* \* \*  En el ejemplo, cada día de cada mes de cada año a las 23:00. Se puede cambiar a cualquier valor que se ajuste a las necesidades de la entidad. |

## Configuración en disco

Si en el paso de si *desea usar AWS S3 como almacenamiento*, selecciona NO (2),

|  |
| --- |
| Do you want to use AWS S3 storage?  1) Yes  2) No  [2]: |

todos los backups de base de datos serán generados sólo en el directorio local:

[**/home/kobo\_user/kobotoolbox/kobo-docker/backups**]

(Aclaración: si este fué el directorio seleccionado para que KoBoToolbox sea instalado, este puede variar dependiendo de su configuración al momento de instalar)

El directorio **backups** siempre se encuentra bajo **kobo-docker**

### Estructura del directorio Backups

Se tienen cuatro subdirectorios:

* kobocat
  + Almacena los formularios subidos a la plataforma
* mongo
  + Almacena un backup de la base de datos de MongoDB en formato
  + mongo-3.4-<dominio.tld>-<año>.<mes>.<dia>.<hora>\_<minuto>.gz
* postgres
  + Almacena dos bases de datos de PostgreSQL:
    - KoboCAT
    - KoboForms (KPI)
  + Formato
    - postgres-**kobocat**-9.5-<dominio.tld>-<año>.<mes>.<dia>.<hora>\_<minuto>.pg\_dump
    - postgres-**koboform**-9.5-<dominio.tld>-<año>.<mes>.<dia>.<hora>\_<minuto>.pg\_dump
* redis
  + Almacena la base de datos de Redis Master
  + Formato
    - redis-3.2.12-<dominio.tld>-<año>.<mes>.<dia>.<hora>\_<minuto>.gz

### Archivo de configuración

Se encuentra en:

[**/home/kobo\_user/kobotoolbox/kobo-env/envfiles]**

Específicamente:

* databases.txt

## Configuración con AWS S3

### Requisitos

* Tener un acceso de API de AWS S3:
  + Access key.
  + Secret key.
* Contar con un bucket con los accesos de administrador.
* Bucket debe estar en **us-east-1**.
  + Se ha visto que KoBoToolbox utiliza componentes de acceso a AWS que están limitados a las Zonas de Disponibilidad más conocidas.

## Instalación en kobo-install

Si desea tener AWS como repositorio de los archivos de backup, se podrá configurar al momento de realizar la acción:

|  |
| --- |
| python3 run.py -s |

Una vez que llegue a la opción: *Do you want to use AWS S3 storage?* realice los siguientes pasos:

|  |
| --- |
| Do you want to use AWS S3 storage?  1) Yes  2) No  [1]: 1 |

|  |
| --- |
| AWS Access Key []:<Coloque el Access Key> |

|  |
| --- |
| AWS Secret Key []: <Coloque el Secret Key> |

|  |
| --- |
| AWS Bucket name []: <Coloque el Bucket para FORMULARIOS en la región us-east-1> |

|  |
| --- |
| Otras configuraciones... |

|  |
| --- |
| AWS Backups bucket name []:<Coloque el Bucket para BACKUPS en la región us-east-1> |

|  |
| --- |
| How many yearly backups to keep?  [2]: <Sugerido mantener> |

|  |
| --- |
| How many monthly backups to keep?  [12]: <Sugerido mantener> |

|  |
| --- |
| How many weekly backups to keep?  [4]: <Sugerido mantener> |

|  |
| --- |
| How many daily backups to keep?  [30]: <Sugerido mantener> |

|  |
| --- |
| PostgresSQL backup minimum size (in MB)?  Files below this size will be ignored when rotating backups.  [50]: <Sugerido mantener> |

|  |
| --- |
| MongoDB backup minimum size (in MB)?  Files below this size will be ignored when rotating backups.  [50]: <Sugerido mantener> |

|  |
| --- |
| Redis backup minimum size (in MB)?  Files below this size will be ignored when rotating backups.  [5]: <Sugerido mantener> |

|  |
| --- |
| Chunk size of multipart uploads (in MB)?  [15]: <Sugerido mantener> |

|  |
| --- |
| Use AWS LifeCycle deletion rule?  1) Yes  2) No  [2]: <Sugerido mantener> |

### Archivo de configuración

Se encuentra en:

[**/home/kobo\_user/kobotoolbox/kobo-env/envfiles]**

Específicamente:

* aws.txt

## Revisión de logs

### Desde kobo-docker

Vaya a:

[**/home/kobo\_user/kobotoolbox/kobo-docker/logs]**

En este directorio tiene los siguientes subdirectorios:

* kobocat
* kpi
* mongo
* nginx
* postgres
* redis\_cache
* redis\_main

Cada uno contiene información respecto a la especialidad del componente en cuestión

# Restauración de Bases de Datos

Como lo menciona la documentación (Ver [Referencias Adicionales](#_vhksg1a4e3xp)) la restauración del backup de una base de datos específica se realiza en su propio contenedor de Docker.

Se sugiere seguir el orden propuesto, ya que ese dió resultados positivos en las pruebas.

Se debe tener a mano los usuarios y contraseñas de cada base de datos. Los mismos se obtienen al momento de ejecutar:

|  |
| --- |
| python3 run.py -s |

### Acerca del archivo de backup

Ya que los archivos estén dispuestos, se deben hacer disponibles dentro del contenedor de cada base de datos. Para la conveniencia de este procedimiento, el archivo ***docker-compose.backend.template.yml*** (kobo-docker) describe el lugar de almacenamiento de backups en la sección ***volumes*** de cada servicio (container). Por ejemplo para mongo, se describe:

mongo:

image: mongo:3.4

hostname: mongo

environment:

- MONGO\_DATA=/data/db

env\_file:

- ../kobo-env/envfile.txt

- ../kobo-env/envfiles/databases.txt

- ../kobo-env/envfiles/aws.txt

volumes:

- ./.vols/mongo:/data/db

**- ./backups/mongo:/srv/backups**

- ./mongo:/kobo-docker-scripts

- ./log/mongo:/srv/logs

restart: always

command: "bash /kobo-docker-scripts/entrypoint.sh"

stop\_grace\_period: 5m

Esto significa que dentro del container, los backups se encuentran en **/srv/backups** y que el mapeo en la máquina host (la máquina donde se ejecutan todos los containers) tiene los backups de mongo en [**/home/kobo\_user/kobotoolbox]/kobo-docker/backups/mongo**

A continuación se describen las restauraciones para cada base de datos.

## Restaurar MongoDB

Ingrese al contenedor

|  |
| --- |
| docker exec -it kobobe\_mongo\_1 bash |

Debe contar con un backup específico. El mismo debería encontrarse en el path anteriormente indicado:

[**/home/kobo\_user/kobotoolbox]/kobo-docker/backups/mongo**

Ejecute:

|  |
| --- |
| mongorestore --username=root --archive=/srv/backups/mongo-3.4-nexion-dev.tk-2020.11.26.22\_00.gz --gzip |

|  |
| --- |
| Password: <Provea el password de Mongo> |

En el ejemplo el archivo:

**mongo-3.4-nexion-dev.tk-2020.11.26.22\_00.gz**

Es el que se quiere utilizar para la restauración

El output que debería esperarse es:

|  |
| --- |
| 2020-12-02T21:10:41.848+0000 preparing collections to restore from  2020-12-02T21:10:41.891+0000 reading metadata for formhub.instances from archive '/srv/backups/mongo-3.4-nexion-dev.tk-2020.11.26.22\_00.gz'  2020-12-02T21:10:41.895+0000 restoring formhub.instances from archive '/srv/backups/mongo-3.4-nexion-dev.tk-2020.11.26.22\_00.gz'  2020-12-02T21:10:41.900+0000 restoring indexes for collection formhub.instances from metadata  2020-12-02T21:10:41.913+0000 finished restoring formhub.instances (0 documents)  2020-12-02T21:10:41.920+0000 restoring users from archive '/srv/backups/mongo-3.4-nexion-dev.tk-2020.11.26.22\_00.gz'  2020-12-02T21:10:41.982+0000 done |

## Issues pip dentro de la imagen de Mongo

Al momento de realizar esta guía se tuvieron algunos casos en los que los backups de mongodb no se pueden realizar dado que la versión de python es 3.5.2 en la imagen de mongo provista por kobotoolbox (mongo:3.4). El problema es por DEPRECATION y la no instalación correcta de pip para python3.5. Eso evita que el archivo cron: /etc/cron.d/ backup\_mongo\_crontab.

La solución a este problema es la instalación manual de pip dentro del container de mongo:

* docker exec -it kobobe\_mongo\_1 bash
* apt install –y curl
* curl https://bootstrap.pypa.io/pip/3.5/get-pip.py -o get-pip.py
* python3 get-pip.py
* bash /kobo-docker-scripts/toggle-backup-activation.sh

Esto generará el archivo cron mencionado: /etc/cron.d/ backup\_mongo\_crontab y ya se podrán realizar los backups con normalidad.

## Restaurar PostgreSQL

Ingrese al contenedor

|  |
| --- |
| docker exec -it kobobe\_postgres\_1 bash |

Debe contar con un backup específico. El mismo debería encontrarse en el path anteriormente indicado:

[**/home/kobo\_user/kobotoolbox]/kobo-docker/backups/postgres**

#### Base de datos Postgres de KoboForms (KPI)

Ejecute:

|  |
| --- |
| pg\_restore -U kobo -d koboform -c "/srv/backups/postgres-koboform-9.5-nexion-dev.tk-2020.11.26.21\_00.pg\_dump" |

En el ejemplo el archivo:

**postgres-koboform-9.5-nexion-dev.tk-2020.11.26.21\_00.pg\_dump**

Es el que se quiere utilizar para la restauración

El output que debería esperarse es: vacío

#### Base de datos Postgres de KoboCat

Ejecute:

|  |
| --- |
| pg\_restore -U kobo -d kobocat -c "/srv/backups/postgres-kobocat-9.5-nexion-dev.tk-2020.11.26.21\_00.pg\_dump" |

En el ejemplo el archivo:

**postgres-kobocat-9.5-nexion-dev.tk-2020.11.26.21\_00.pg\_dump**

Es el que se quiere utilizar para la restauración

El output que debería esperarse es: vacío

## Restaurar Redis

Ingrese al contenedor

|  |
| --- |
| docker exec -it kobobe\_redis\_main\_1 bash |

Debe contar con un backup específico. El mismo debería encontrarse en el path anteriormente indicado:

[**/home/kobo\_user/kobotoolbox]/kobo-docker/backups/redis**

Ejecute:

|  |
| --- |
| cd /srv/backups/ |

|  |
| --- |
| gunzip redis-3.2.12-nexion-dev.tk-2020.11.26.23\_00.gz |

|  |
| --- |
| mv redis-3.2.12-nexion-dev.tk-2020.11.26.23\_00 /data/enketo-main.rdb |

En el ejemplo el archivo:

**redis-3.2.12-nexion-dev.tk-2020.11.26.23\_00.gz**

Es el que se quiere utilizar para la restauración (una vez se realiza la acción unzip)

El output que debería esperarse es: vacío

# Script para restaurar la base de datos de un servidor a otro

Los siguientes scripts se los debe ejecutar en una sesión SSH dentro del servidor destino (target server). Se debe contar con acceso al servidor del cual se quiere tener la base de datos origen (remote server).

Recuerde reemplazar los valores entre signos “<” y “>” ej. <DB-PASSWORD>, <TARGET-SERVER-IP>, <REMOTE-SERVER-IP>.

#Fix Postgres issue:

#--------------------

export postgres\_pass=<DB-PASSWORD>

export new\_db=support\_recovery

export support\_db=support

export target\_server=<TARGET-SERVER-IP>

export remote\_server=<REMOTE-SERVER-IP>

# get pg dump

PGPASSWORD=password pg\_dump -U user --port 5440 -h ${remote\_server} -d support --format=custom > recovery.dump

# Execute create DB

# Delete all sessions

PGPASSWORD=${postgres\_pass} psql -d postgres -U user --port 5440 -h ${target\_server} -c "SELECT pg\_terminate\_backend(pid) FROM pg\_stat\_activity WHERE pid <> pg\_backend\_pid() AND datname = '${new\_db}'";

# Drop DB

PGPASSWORD=${postgres\_pass} psql -d postgres -U user --port 5440 -h ${target\_server} -c "DROP DATABASE IF EXISTS ${new\_db};"

# Create temp

PGPASSWORD=${postgres\_pass} psql -d postgres -U user --port 5440 -h ${target\_server} -c "CREATE DATABASE ${new\_db} WITH OWNER 'user' ENCODING 'UTF8' CONNECTION LIMIT= -1;"

# Restore into temp db

PGPASSWORD=${postgres\_pass} pg\_restore -d ${new\_db} -U user --port 5440 -h ${target\_server} recovery.dump

# \* -- New DB is ready -- \*

#-- Healthcheck

PGPASSWORD=${postgres\_pass} psql -d ${new\_db} -U user --port 5440 -h ${target\_server} -c "\d"

PGPASSWORD=${postgres\_pass} psql -d ${new\_db} -U user --port 5440 -h ${target\_server} -c "select \* from public.\"Role\""

# Delete all sessions from support database

PGPASSWORD=${postgres\_pass} psql -d postgres -U user --port 5440 -h ${target\_server} -c "SELECT pg\_terminate\_backend(pid) FROM pg\_stat\_activity WHERE pid <> pg\_backend\_pid() AND datname = '${support\_db}'";

# Delete all sessions from support database

PGPASSWORD=${postgres\_pass} psql -d postgres -U user --port 5440 -h ${target\_server} -c "SELECT pg\_terminate\_backend(pid) FROM pg\_stat\_activity WHERE pid <> pg\_backend\_pid() AND datname = '${support\_db}\_OLD'";

# Drop DB - support\_OLD

PGPASSWORD=${postgres\_pass} psql -d postgres -U user --port 5440 -h ${target\_server} -c "DROP DATABASE IF EXISTS ${support\_db}\_OLD;"

# Renaming support database to support\_OLD

PGPASSWORD=${postgres\_pass} psql -d postgres -U user --port 5440 -h ${target\_server} -c "ALTER DATABASE ""${support\_db}"" RENAME TO ""${support\_db}\_OLD"""

# Terminating new db sessions

PGPASSWORD=${postgres\_pass} psql -d postgres -U user --port 5440 -h ${target\_server} -c "SELECT pg\_terminate\_backend(pid) FROM pg\_stat\_activity WHERE pid <> pg\_backend\_pid() AND datname = '${new\_db}'";

# Renaming new\_db database to support

PGPASSWORD=${postgres\_pass} psql -d postgres -U user --port 5440 -h ${target\_server} -c "ALTER DATABASE ""${new\_db}"" RENAME TO ""${support\_db}"""

# Healthcheck

PGPASSWORD=${postgres\_pass} psql -d ${support\_db} -U user --port 5440 -h ${target\_server} -c "\d"

PGPASSWORD=${postgres\_pass} psql -d ${support\_db} -U user --port 5440 -h ${target\_server} -c "select \* from public.\"Role\""

# Referencias adicionales

<https://github.com/kobotoolbox/kobo-docker#usage>