

Manual Migration Using keydb-cli and Dump Files

Manual migrations using KeyDB's native tools are ideal for users who prefer full control over data export and import, particularly during provider transitions, environment duplication, or when importing an existing self-managed KeyDB dataset into Elestio's managed environment. This guide walks through the process of performing a manual migration to and from Elestio KeyDB services using command-line tools, ensuring that your data remains portable, auditable, and consistent.

KeyDB is fully compatible with Redis tooling, including `redis-cli`, `redis.conf`, and snapshot formats (`dump.rdb`, `appendonly.aof`). For clarity, this guide uses `keydb-cli` and `keydb-server` where applicable. If you're using a Redis-compatible CLI installed with your system, the commands remain the same.

When to Use Manual Migration

Manual migration using `keydb-cli` is well-suited for scenarios where full control over the data export and import process is required. This method is particularly useful when migrating from an existing KeyDB or Redis-compatible setup, whether self-hosted, on-premises, or on another cloud provider, into Elestio's managed KeyDB service. It allows for one-time imports without requiring continuous connectivity between source and target systems.

This approach is also ideal when dealing with version upgrades, as KeyDB's dump-based backups can be restored into newer versions without compatibility issues. In situations where Elestio's built-in snapshot or replication tools aren't applicable, such as migrations from isolated environments or selective key imports, manual migration becomes the most practical option. Additionally, this method enables users to retain portable, versioned backups outside of Elestio's infrastructure, which can be archived, validated offline, or re-imported into future instances.

Performing the Migration

Prepare the Environments

Before initiating a migration, verify that KeyDB is properly installed and configured on both the source system and your Elestio service. On the source, you need an active KeyDB instance with the ability to persist data using either RDB or AOF files. The user must also be allowed to connect over TCP if the server is remote.

On the Elestio side, provision a KeyDB service from the dashboard. Once deployed, retrieve the connection information from the Database admin tab. This includes the hostname, port, and password. You'll use these credentials to connect during the restore step. Ensure that your IP is allowed to connect under the Cluster Overview > Security > Limit access per IP section; otherwise, the KeyDB port will be unreachable during the migration.

Create a Backup Dump

In this step, you generate a snapshot of the source KeyDB database. If persistence is enabled, the file `dump.rdb` will be automatically created in the working directory of the KeyDB server. You can trigger a manual save using the CLI to ensure the most recent in-memory state is written to disk.

```
keydb-cli -h <source_host> -p <source_port> SAVE
```

Alternatively, to extract the RDB file via CLI, use:

```
keydb-cli --rdb dump.rdb
```

This command generates an RDB file compatible with KeyDB versions and saves it to the current directory. The resulting file is portable and version-aware. If the source instance uses AOF instead of RDB, you may want to disable AOF temporarily and force an RDB snapshot before migration.

Transfer the Dump File to the Target

If your source and target environments are on different hosts, the dump file must be transferred securely. This step ensures the snapshot is available on the system from which you'll perform the restore. You can use secure copy (`scp`), `rsync`, or any remote file transfer method.

```
scp dump.rdb your_user@your_workstation:/path/to/local/
```

If restoring from your local machine to Elestio, ensure the dump file is stored in a location readable by your current shell user. Elestio does not require the file to be uploaded to its servers; the restore is performed by connecting over the network or using a compatible Docker container. At this point, your backup is isolated from the source environment and ready for import.

Create the Target Container or Instance

To perform the restore locally before importing into Elestio, you can start a KeyDB instance using Docker. Mount the directory containing `dump.rdb` as a volume so the server automatically loads the snapshot on boot.

```
docker run -v /path/to/backup:/data --name keydb-restore -p 6379:6379 eqalpha/keydb
```

This command runs a local KeyDB container with the dump file mounted at `/data`. When the container starts, it detects `dump.rdb` and loads the dataset into memory. This provides an environment where you can inspect the data or transfer it into Elestio over the network.

If you're directly importing into Elestio without this intermediate step, skip to the next section and connect using keydb-cli.

Restore the Data into Elestio

With the dump file available and the target instance deployed on Elestio, you can restore the data using keydb-cli and a live copy command. First, connect to the Elestio KeyDB instance using the credentials from the dashboard:

```
keydb-cli -h <elestio_host> -p <elestio_port> -a <elestio_password>
```

Once connected, you can either use redis-cli --pipe to import a key-by-key export, or if restoring from a container, use a sync-based migration. In most cases, however, you'll want to configure dump.rdb in a containerized KeyDB instance and then use replication to send data to Elestio:

```
keydb-cli -h localhost -p 6379 SLAVEOF <elestio_host> <elestio_port>
```

Then authenticate with:

```
keydb-cli -h localhost -p 6379 -a <elestio_password>
```

KeyDB will stream its contents into Elestio. Once the synchronization is complete, you can break replication and make Elestio the new primary.

```
keydb-cli -h localhost -p 6379 SLAVEOF NO ONE
```

This method is ideal for large datasets or when no direct dump import is supported.

Validate the Migration

Once the restore completes, you must validate the accuracy and completeness of the migration. Connect to the Elestio database using keydb-cli and inspect the dataset:

```
keydb-cli -h <elestio_host> -p <elestio_port> -a <elestio_password>
```

Begin by checking the total number of keys:

```
DBSIZE
```

Inspect specific keys or key patterns:

```
KEYS *  
GET some_key  
TYPE some_key
```

Validate TTLs, data structures, and expected values. Run application-specific health checks and confirm that the application can read and write to the new instance without errors.

If you made any changes to connection strings or credentials, update your environment variables or secret managers accordingly. Elestio also supports automated backups, which you should enable post-migration to protect the restored dataset.

Benefits of Manual Migration

Manual KeyDB migration using native dump files on Elestio provides several key advantages:

- **Compatibility and Portability:** Logical dumps allow you to migrate from any KeyDB- or Redis-compatible source into Elestio, including on-premises systems, Docker containers, or other clouds.
- **Version-Safe Upgrades:** The tools support migrating across KeyDB versions, which is ideal during controlled upgrades.
- **Offline Archiving:** Manual dumps serve as portable archives for cold storage, disaster recovery, or historical snapshots.
- **Platform Independence:** You retain full access to KeyDB native tools without being locked into Elestio-specific formats or interfaces.

This method complements Elestio's automated backup and migration features by enabling custom workflows and one-off imports with full visibility into each stage.

Revision #1

Created 25 June 2025 06:34:01 by kaiwalya

Updated 25 June 2025 06:39:52 by kaiwalya