

Manual Redis Migration Using redis-cli and RDB Files

Manual migrations using Redis's built-in tools, such as `redis-cli` and RDB (Redis Database) files, are ideal for users who require full control over data export and import particularly during transitions between providers, Redis version upgrades, or importing existing self-managed Redis datasets into Elestio's managed environment. This guide walks through the process of performing a manual migration to and from Elestio Redis services using command-line tools, ensuring data portability, consistency, and transparency at every step.

When to Use Manual Migration

Manual migration using native Redis tools is well-suited for scenarios that demand complete control over the migration process. It is especially useful when transferring data from a self-hosted Redis instance, an on-premises server, or another cloud provider into Elestio's managed Redis service. This method supports one-time imports without requiring persistent connections between source and destination systems.

It also provides a reliable approach for performing version upgrades. Because RDB files contain a snapshot of the dataset in a portable format, they can be restored into newer Redis versions with minimal compatibility issues. When Elestio's built-in tools are not applicable such as in migrations from isolated environments or selective key transfers manual migration becomes the preferred option. It also enables offline backup archiving, providing users with transportable and restorable datasets independent of platform-specific formats.

Performing the Migration

Prepare the Environments

Before starting the migration, ensure that Redis is properly installed on both the source system and your Elestio service. The source Redis server must allow access (if remote) and have a user with sufficient privileges to export the dataset, including read access to all relevant keys and data types.

On the Elestio side, provision a Redis service through the dashboard. Once it's active, retrieve the connection credentials from the **Database Info** section. This includes host, port, and password. Verify that your public IP is allowed under **Cluster Overview > Security > Limit access per IP**, or the Redis port will not be reachable.

Create a Backup Using RDB

Use Redis's RDB snapshotting method to create a backup of the dataset. This process serializes the current state of your Redis database into a binary .rdb file.

To trigger a manual snapshot, run:

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

Once the command completes, locate the resulting dump.rdb file on the source system. This is typically stored in /var/lib/redis/ or a path defined in your Redis configuration.

Alternatively, you can generate an RDB file using:

```
redis-cli --rdb backup.rdb
```

This creates a portable snapshot of the entire dataset without modifying the source instance's configuration.

Transfer the Dump File to the Target

If your local system differs from the one with access to Elestio's Redis service, transfer the dump file using a secure file transfer tool such as SCP:

```
scp backup.rdb user@host:/path/to/restore-system/
```

Ensure the file is available on the system you will use to perform the restore. You do not need to upload the RDB file directly to the Elestio service restores are performed remotely using Redis commands.

Restore the Dataset to Elestio

To restore data into Elestio, start a temporary local Redis instance using the dump file:

```
redis-server --dbfilename dump.rdb --dir /path/to/rdb/
```

This allows you to access the original dataset locally. Then, connect to both the local and Elestio Redis instances and copy keys using redis-cli. For example:

```
redis-cli -h <source_host> --scan | while read key; do
  redis-cli -h <source_host> DUMP "$key" | \
  redis-cli -h <elestio_host> -p <elestio_port> -a <elestio_password> RESTORE "$key" 0 -
done
```

This approach reads each key from the source instance and restores it to the Elestio-managed Redis instance. Ensure that both instances are reachable and that no firewall or access rules block communication.

For large datasets or environments with complex key structures, consider using community tools like redis-copy or redis-migrate-tool to streamline key transfers.

Validate the Migration

After completing the import, verify that the migration was successful by connecting to the Elestio Redis instance and inspecting the dataset.

Start by checking the total key count:

```
redis-cli -h <elestio_host> -p <elestio_port> -a <elestio_password> DBSIZE
```

Review specific keys to confirm data consistency:

```
redis-cli -h <elestio_host> -p <elestio_port> -a <elestio_password> KEYS *
```

Also verify the integrity of sets, hashes, lists, and sorted sets if used in your application. Ensure that your application connects to the new Redis instance without issues and performs expected operations.

If you've updated environment variables or configuration files, confirm that your changes are reflected in the application deployment.

Benefits of Manual Migration

Manual Redis migration using redis-cli and RDB files offers several important advantages:

- **Portability and Compatibility:** RDB files are standard Redis snapshot formats that can be restored into any Redis-compatible instance, whether hosted locally, in containers, or in the cloud.
- **Version Flexibility:** Migrate across Redis versions using forward-compatible RDB snapshots, without relying on binary compatibility or replication.
- **Offline Storage:** Backup files can be stored offline, versioned, and archived as part of disaster recovery or compliance processes.
- **Platform Independence:** Elestio does not enforce proprietary formats. Native Redis tools give you complete control over export, transfer, and restoration operations.

Revision #1

Created 16 May 2025 12:32:18 by kaiwalya

Updated 16 May 2025 12:36:01 by kaiwalya