



APUSIC
固若长城
睿比世界

迁移手册

金蝶Apusic键值数据库 V2.0

版权声明

本文档所涉及的软件著作权、版权等知识产权已依法进行了注册，由金蝶天燕云计算股份有限公司合法拥有。受《中华人民共和国著作权法》《计算机软件保护条例》《知识产权保护条例》和相关国际版权条约、法律、法规以及其它知识产权法律和条约的保护。未经授权许可，不得非法使用。

免责声明

本文档包含的版权信息由金蝶天燕云计算股份有限公司合法拥有，受法律的保护，金蝶天燕云计算股份有限公司对本文档可能涉及到的非金蝶天燕云计算股份有限公司的信息不承担任何责任。在法律允许的范围内，您可以查阅并仅能够在《中华人民共和国著作权法》规定的合法范围内复制和打印本文档。任何单位和个人未经金蝶天燕云计算股份有限公司书面授权许可，不得使用、修改、再发布本文档的任何部分和内容，否则将被视为侵权，金蝶天燕云计算股份有限公司有依法追究其责任的权利。

本文档如有更新，不另行通知。对本文档中的问题您可向金蝶天燕云计算股份有限公司告知或查询。未经本公司明确授予的任何权利均予保留。

商标声明

 是深圳市金蝶天燕云计算股份有限公司向中华人民共和国国家商标局申请注册的注册商标，注册商标专用权由金蝶天燕合法拥有，受法律保护。未经金蝶天燕的书面许可，任何单位及个人不得以任何方式或理由对该商标的任何部分进行使用、复制、修改、传播、抄录或与其它产品捆绑使用销售。凡侵犯金蝶天燕商标权的，金蝶天燕将依法追究其法律责任。本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

目录

- 1 前言
 - 1.1 适用对象
 - 1.2 相关文档
 - 1.3 技术支持
- 2 概述
- 3 KVDB V2.0.2及以下版本迁移至V2.0操作说明
 - 3.1 准备工作
 - 3.1.1 环境检查
 - 3.1.2 工具与文件准备
 - 3.1.2.1 必备工具
 - 3.1.2.2 待迁移文件收集
 - 3.2 单机模式迁移
 - 3.2.1 迁移前提
 - 3.2.2 准备工作
 - 3.2.2.1 工具与配置前置
 - 3.2.2.2 备份数据并停止KVDB V2.0.2及以下版本单节点（全量停机）
 - 3.2.3 迁移步骤
 - 3.2.3.1 配置文件迁移
 - 3.2.3.2 缓存数据迁移
 - 3.2.3.3 启动Apusic KVDB V2.0服务
 - 3.2.3.4 验证启动
 - 3.2.4 注意事项
 - 3.3 主从模式迁移
 - 3.3.1 迁移前提
 - 3.3.2 准备工作
 - 3.3.2.1 工具与配置前置
 - 3.3.2.2 备份数据并停止KVDB V2.0.2及以下的本主从集群（全量停机）
 - 3.3.3 迁移步骤（按「主节点->所有从节点」顺序执行）
 - 3.3.3.1 主节点迁移
 - 3.3.3.2 从节点迁移（所有从节点同步执行）
 - 3.3.4 注意事项
 - 3.4 哨兵模式迁移

- 3.4.1 迁移前提
- 3.4.2 准备工作
 - 3.4.2.1 工具与配置前置
 - 3.4.2.2 备份数据并全量停止KVDB V2.0.2及以下版本的哨兵 + 主从集群
- 3.4.3 迁移步骤（按「Apusic KVDB V2.0主从集群->Apusic KVDB V2.0哨兵集群」顺序执行）
 - 3.4.3.1 重建Apusic KVDB V2.0主从集群（核心，同上一章完整步骤）
 - 3.4.3.2 哨兵节点迁移（所有哨兵节点同步执行）
- 3.4.4 注意事项
- 3.5 集群模式迁移
 - 3.5.1 迁移前提
 - 3.5.2 准备工作
 - 3.5.2.1 工具与配置前置
 - 3.5.2.2 统一环境与记录源拓扑
 - 3.5.2.3 备份数据并全量停止KVDB V2.0.2及以下版本的集群（禁止任何写入）
 - 3.5.2.4 数据文件预处理（所有源节点执行）
 - 3.5.3 迁移步骤（按「配置转换->Apusic KVDB V2.0集群初始化->分片数据导入->集群状态验证」执行）
 - 3.5.3.1 初始化Apusic KVDB V2.0空集群
 - 3.5.3.1.1 验证所有节点启动成功（日志显示Ready to accept connections）
 - 3.5.3.2 分片数据导入
 - 3.5.3.3 集群状态修复与数据校验
 - 3.5.4 注意事项
- 4 Redis 8.6及以下版本迁移至Apusic KVDB V2.0操作说明
 - 4.1 准备工作
 - 4.1.1 环境检查
 - 4.1.2 工具与文件准备
 - 4.2 单机模式迁移
 - 4.2.1 迁移前提
 - 4.2.2 准备工作
 - 4.2.2.1 数据与配置备份
 - 4.2.2.2 数据备份并停止Redis单节点
 - 4.2.3 迁移步骤
 - 4.2.3.1 缓存数据迁移

- 4.2.3.2 修改redis.conf配置
- 4.2.3.3 启动KVDB服务
- 4.2.3.4 验证启动
- 4.2.4 注意事项
- 4.3 主从模式迁移
 - 4.3.1 迁移前提
 - 4.3.2 准备工作
 - 4.3.2.1 工具与文件准备
 - 4.3.2.2 数据备份并停止Redis主从集群（全量停机）
 - 4.3.3 迁移步骤（按「主节点->所有从节点」顺序执行）
 - 4.3.3.1 缓存数据迁移
 - 4.3.3.2 配置文件迁移（redis.conf）
 - 4.3.3.3 启动KVDB主从节点服务
 - 4.3.3.4 主从关系验证
 - 4.3.4 注意事项
- 4.4 哨兵模式迁移
 - 4.4.1 迁移前提
 - 4.4.2 准备工作
 - 4.4.2.1 工具与文件准备
 - 4.4.2.2 数据备份并全量停止Redis哨兵 + 主从集群
 - 4.4.3 迁移步骤
 - 4.4.3.1 缓存数据迁移
 - 4.4.3.2 配置文件迁移（redis.conf）
 - 4.4.3.3 启动KVDB主从节点服务
 - 4.4.3.4 启动KVDB哨兵节点服务
 - 4.4.3.5 哨兵功能验证
 - 4.4.4 注意事项
- 4.5 集群模式迁移
 - 4.5.1 迁移前提
 - 4.5.2 准备工作
 - 4.5.2.1 工具与文件准备
 - 4.5.2.2 数据备份并全量停止Redis集群（禁止任何写入）
 - 4.5.3 迁移步骤
 - 4.5.3.1 数据文件预处理（所有源节点执行）

- 4.5.3.2 配置文件迁移 (redis.conf, node.conf)
- 4.5.3.3 启动KVDB集群节点
- 4.5.3.4 集群状态验证
- 4.5.4 注意事项
- 5 使用kvdb-shake进行迁移
 - 5.1 概述
 - 5.1.1 适用场景
 - 5.1.2 兼容性
 - 5.2 工具准备
 - 5.3 基本使用
 - 5.4 配置文件说明
 - 5.4.1 Reader 配置 (数据来源)
 - 5.4.1.1 sync_reader: 通过 PSync 协议实时同步
 - 5.4.1.2 scan_reader: 通过 SCAN 命令全量遍历
 - 5.4.1.3 rdb_reader: 从 RDB 文件离线读取
 - 5.4.1.4 aof_reader: 从 AOF 文件离线读取
 - 5.4.2 Writer 配置 (数据写入)
 - 5.4.3 Filter 配置 (可选)
 - 5.4.4 Advanced 配置 (可选)
 - 5.5 命令行参数
 - 5.6 典型场景配置示例
 - 5.6.1 场景一: Redis 单机迁移至 KVDB 单机 (基于 PSync)
 - 5.6.2 场景二: Redis 主从迁移至 KVDB 主从
 - 5.6.3 场景三: Redis 哨兵/主从集群迁移至 KVDB 哨兵/主从集群
 - 5.6.4 场景四: Redis Cluster 迁移至 KVDB Cluster
 - 5.6.5 场景五: 基于 SCAN + KSN 的全量 + 增量迁移
 - 5.6.6 场景六: 从 RDB 文件离线迁移至 KVDB
 - 5.7 数据一致性校验
 - 5.8 注意事项
- 6 常见问题排查
 - 6.1 工具无法运行
 - 6.2 校验或修复文件失败
 - 6.3 kvdb-cli 无法连接KVDB服务
 - 6.4 kvdb-shake 迁移常见问题

•7 总结

- 7.1 KVDB V2.0.2及以下版本向Apusic KVDB V2.0迁移流程
- 7.2 Redis 8.6及以下版本向Apusic KVDB V2.0迁移流程
- 7.3 基于 kvdb-shake 的在线迁移流程

1 前言

本文档为金蝶Apusic键值数据库（KVDB）V2.0的迁移手册，详细介绍了如何将Redis迁移到KVDB，以及KVDB老版本迁移到V2.0版本。

1.1 适用对象

本文档适用于KVDB产品运维工程师、IT系统运维工程师、开发工程师、软件架构师及研发经理等人员。

1.2 相关文档

了解更多Apusic KVDB V2.0产品相关的信息，请参阅以下Apusic KVDB V2.0产品手册文档集：

序号	手册文档	说明
1	金蝶Apusic键值数据库 V2.0 快速使用手册	简单介绍了如何快速上手使用KVDB。
2	金蝶Apusic键值数据库 V2.0 安装手册	详细介绍如何在各操作系统上安装KVDB，以及KVDB服务启停操作，产品的注册过程。
3	金蝶Apusic键值数据库 V2.0 键值核心用户手册	详细介绍 KVDB 相关功能的使用、配置、管理及配套工具的使用方法。
4	金蝶Apusic键值数据库 V2.0 管控台用户手册	详细介绍KVDB管控台相关功能的使用和操作说明。
5	金蝶Apusic键值数据库 V2.0 开发手册	详细介绍基于各开发语言进行KVDB客户端应用开发的说明。
6	金蝶Apusic键值数据库 V2.0 迁移手册	详细介绍KVDB历史版本迁移升级到V2.0版本的说明，以及Redis迁移到KVDB的说明。
7	金蝶Apusic键值数据库 V2.0 运维手册	详细介绍KVDB的监控、运维、安全加固等运维说明。
8	金蝶Apusic键值数据库 V2.0 性能优化手册	详细介绍KVDB性能调优的说明。

1.3 技术支持

KVDB产品提供全面的技术支持服务，您可以通过以下方式获得技术支持：

- 网址: www.apusic.com
- 电话: 400-855-5800
- 邮箱: support@apusic.com
- 金蝶云社区: <https://vip.kingdee.com/?productId=73&productLineId=14&lang=zh-CN>

您在取得技术支持时, 请提供如下信息:

1. 您的姓名
2. 公司与联系方式
3. 操作系统及其版本
4. 产品版本号
5. 出现异常及错误的日志、截图等详细信息

2 概述

本手册介绍两种迁移至Apusic KVDB V2.0的操作说明：

1. KVDB V2.0.2及以下版本迁移至Apusic KVDB V2.0：使用kvdb-conf-conv转换KVDB V2.0.2及以下版本的配置文件为Apusic KVDB V2.0兼容格式，复用原有配置及数据文件，启动后验证功能即可；
2. Redis 8.6及以下版本迁移至Apusic KVDB V2.0：核心依托KVDB可直接加载Redis配置（redis.conf等）及数据文件（RDB/AOF等）的特性。

3 KVDB V2.0.2及以下版本迁移至V2.0操作说明

3.1 准备工作

3.1.1 环境检查

- 已安装Apusic KVDB V2.0（需与工具版本匹配），且安装目录权限正常（可读可执行）；
- 待迁移的KVDB V2.0.2及以下版本的服务可正常访问，或已获取其RDB缓存文件、配置文件（YAML格式）；
- 迁移目标服务器（部署的Apusic KVDB V2.0服务器）具备足够的磁盘空间，确保能存储迁移后的缓存数据及配置文件；

3.1.2 工具与文件准备

3.1.2.1 必备工具

- [kvdb-check-rdb](#)：用于校验和修复KVDB V2.0.2及以下版本的RDB缓存文件（随Apusic KVDB V2.0安装包部署，位于/kvdb/目录下）；
- [kvdb-check-aof](#)：用于校验和修复KVDB V2.0.2及以下版本的AOF文件（同目录部署）；
- [kvdb-conf-conv](#)：用于转换KVDB V2.0.2及以下版本的配置文件为Apusic KVDB V2.0兼容格式（同目录部署）；

3.1.2.2 待迁移文件收集

- KVDB V2.0.2及以下版本的配置文件：服务端配置（如conf.yaml）或哨兵配置（如sentinel_go.yaml）；
- KVDB V2.0.2及以下版本的RDB快照文件：默认名为dump.rdb，路径通常在安装目录下（需确认实际路径）；
- 若KVDB V2.0.2及以下版本启用了AOF持久化，则优先使用AOF文件作为迁移数据源，RDB文件作为备份迁移数据源；

3.2 单机模式迁移

单机模式迁移适用于仅部署单台KVDB服务的场景，核心流程为「文件准备 -> 配置转换 -> 数据校验修复 -> 部署启动 -> 验证」，操作简单且无依赖关系，具体步骤如下：

3.2.1 迁移前提

- 目标服务器已安装Apusic KVDB V2.0，安装目录权限正常（可读可执行）；

- 已收集KVDB V2.0.2及以下版本的配置文件（如conf.yaml）、RDB快照文件(dump.rdb)，若启用AOF持久化则优先获取AOF文件；
- 目标服务器磁盘空间充足，至少为源数据文件大小的2倍（预留备份与运行空间）。

3.2.2 准备工作

3.2.2.1 工具与配置前置

- [kvdb-check-rdb](#)：用于校验和修复KVDB V2.0.2及以下版本的RDB缓存文件（随Apusic KVDB V2.0安装包部署，位于/kvdb/目录下）；
- [kvdb-check-aof](#)：用于校验和修复KVDB V2.0.2及以下版本的AOF文件（同目录部署）；
- [kvdb-conf-conv](#)：用于转换KVDB V2.0.2及以下版本的配置文件为Apusic KVDB V2.0兼容格式（同目录部署）；

3.2.2.2 备份数据并停止KVDB V2.0.2及以下版本单节点（全量停机）

- 停止KVDB V2.0.2及以下版本单节点的写入操作后，再次执行bgsave命令，生成最新RDB文件；备份源端KVDB配置；
- 必须先停止KVDB V2.0.2及以下版本的单节点，禁止任何数据写入，确保源数据文件最终一致性。

#节点执行：停止KVDB V2.0.2及以下版本的KVDB进程

```
ps -ef | grep kvdb-server | grep -v grep | awk '{print $2}' | xargs kill -9
```

#验证进程已停止（无输出则表示成功）

```
ps -ef | grep kvdb-server | grep -v grep
```

3.2.3 迁移步骤

3.2.3.1 配置文件迁移

使用 [kvdb-conf-conv](#) 工具将KVDB V2.0.2及以下版本的服务端配置转换为Apusic KVDB V2.0兼容格式，操作如下：

1. 备份目标服务器的Apusic KVDB V2.0默认配置文件（避免覆盖丢失）：

```
cp /kvdb/kvdb.yaml /kvdb/kvdb.yaml.bak
```

2. 执行配置转换（指定--server参数，适配服务端配置）：

```
kvdb-conf-conv --server --go-yaml /path/to/conf.yaml --c-yaml
/kvdb/kvdb.yaml
```

3. 验证转换结果：打开/kvdb/kvdb.yaml，确认核心配置项（如端口、密码、数据目录、内存限制等）已正确继承配置，无语法错误；

3.2.3.2 缓存数据迁移

1. 校验并修复源RDB或AOF文件（优先处理AOF文件，数据更完整）：

```
#若使用AOF文件
kvdb-check-aof /path/to/appendonlydir/appendonly.aof.manifest

#若使用RDB文件
kvdb-check-rdb /path/to/dump.rdb

#若文件损坏，执行修复
kvdb-check-aof --fix
/path/to/appendonlydir/appendonly.aof.1.base.rdb /
kvdb-check-aof --fix
/path/to/appendonlydir/appendonly.aof.1.base.aof /
kvdb-check-aof --fix
/path/to/appendonlydir/appendonly.aof.1.incr.aof

kvdb-check-rdb --fix /path/to/dump.rdb
```

2. 复制修复后的文件到Apusic KVDB V2.0的安装目录：

```
#查看Apusic KVDB V2.0数据目录（在kvdb.yaml中通过dir配置项确认，默认路径如下），复制RDB文件
cp /path/to/dump.rdb/ kvdb/

#若使用AOF文件，需先启用AOF配置（在kvdb.yaml中设置appendonly yes），再复制
cp /path/to/appendonlydir /kvdb/
```

3.2.3.3 启动Apusic KVDB V2.0服务

1. 启动Apusic KVDB V2.0服务，加载转换后的配置文件：

```
kvdb-server /kvdb/kvdb.yaml
```

2. 验证服务启动状态：

```
#查看服务进程
ps -ef | grep kvdb-server

#查看日志（默认日志路径 / kvdb/server.log）
tail -100f /kvdb/server.log
```

3.2.3.4 验证启动

1. 连接Apusic KVDB V2.0服务，验证数据完整性：

```
#使用kvdb-cli连接（默认端口6359，若配置密码需加-a选项）
kvdb-cli -p 6359 -a <密码># 示例值，非真实数据

#查看键空间统计信息
info keyspace
```

2. 验证功能可用性：执行写入、修改、删除等操作，确认服务正常响应，无配置或数据相关报错：

3.2.4 注意事项

- 迁移过程中需停止KVDB V2.0.2及以下版本的服务，避免数据写入导致源文件损坏；
- 迁移后观察服务运行 10-20 分钟，确认无日志报错等异常。

3.3 主从模式迁移

主从模式迁移适配下，KVDB V2.0.2及以下版本与Apusic KVDB V2.0无法跨版本建立主从的场景，核心原则为全量停机 -> 先迁主节点 -> 后迁从节点 -> 数据一致验证，通过RDB或AOF全量数据导入完成迁移，全程需短暂停机（无跨版本同步环节），具体步骤如下：

3.3.1 迁移前提

- 主、从目标服务器均已安装Apusic KVDB V2.0，安装目录权限正常（可读、可写、可执行），且主从节点网络互通（无防火墙拦截6359端口）；
- 已收集KVDB V2.0.2及以下版本主、从节点的配置文件（conf.yaml）、AOF文件（优先，数据更完整）或RDB快照文件，并做好全量备份；
- 提前告知业务方停机窗口，建议选择业务低峰期操作，停机时长约为「数据文件拷贝 + 服务启动」耗时（根据数据量调整，一般5-30分钟）；
- 目标服务器磁盘空间充足，至少为源数据文件大小的2倍（预留备份与运行空间）。

3.3.2 准备工作

3.3.2.1 工具与配置前置

- [kvdb-check-rdb](#)：用于校验和修复KVDB V2.0.2及以下版本的RDB缓存文件（随Apusic KVDB V2.0安装包部署，位于/kvdb/目录下）；
- [kvdb-check-aof](#)：用于校验和修复KVDB V2.0.2及以下版本的AOF文件（同目录部署）；
- [kvdb-conf-conv](#)：用于转换KVDB V2.0.2及以下版本的配置文件为Apusic KVDB V2.0兼容格式（同目录部署）；

3.3.2.2 备份数据并停止KVDB V2.0.2及以下的本主从集群（全量停机）

- 停止KVDB V2.0.2及以下版本单节点的写入操作后，再次执行bgsave命令，生成最新RDB文件；备份源端KVDB配置；
- 必须先停止所有KVDB V2.0.2及以下版本的主、从节点，禁止任何数据写入，确保源数据文件最终一致性。

```
#主节点 + 所有从节点统一执行：停止KVDB V2.0.2及以下版本的KVDB进程
ps -ef | grep kvdb-server | grep -v grep | awk '{print $2}' | xargs
kill -9

#验证进程已停止（无输出则表示成功）
ps -ef | grep kvdb-server | grep -v grep
```

3.3.3 迁移步骤（按「主节点->所有从节点」顺序执行）

3.3.3.1 主节点迁移

主节点为数据核心，需优先完成配置转换、数据校验导入和服务启动，作为后续从节点的同步源。

步骤1: 配置文件转换与优化

1. 备份Apusic KVDB V2.0主节点默认配置（避免覆盖）：

```
cp /kvdb/kvdb.yaml /kvdb/kvdb-master.yaml.bak
```

2. 将KVDB V2.0.2及以下版本的主节点配置转换为Apusic KVDB V2.0主节点专用配置（命名为kvdb-master.yaml，便于区分）：

```
kvdb-conf-conv --server --go-yaml /path/to/conf.yaml --c-yaml  
/kvdb/kvdb-master.yaml
```

3. 检查Apusic KVDB V2.0主节点配置，启用主节点身份（无replicaof配置），确认核心配置项：

```
vi /kvdb/kvdb-master.yaml
```

关键配置确认 / 修改：

- port:6359（端口统一，若需修改则所有节点同步）；
- requirepass: 你的密码（若源主节点有密码，需保留，后续从节点需配置masterauth）；
- dir:/kvdb/（数据目录，与实际规划一致）；
- appendonly:yes（建议启用AOF，与源端保持一致，若源端仅用RDB则设为no）；
- logfile:kvdb-master.log（主节点独立日志，便于排查）。

步骤2: 数据文件校验、修复与导入

1. 校验并修复源KVDB V2.0.2及以下版本的主节点数据文件（主节点执行，若文件无损坏则跳过修复）：

#若使用AOF文件（推荐）

```
kvdb-check-aof /path/to/appendonlydir/appendonly.aof.manifest
```

#若使用RDB文件

```
kvdb-check-rdb /path/to/dump.rdb
```

#若文件损坏，执行修复

```

kvdb-check-aof --fix
/path/to/appendonlydir/appendonly.aof.1.base.rdb /
kvdb-check-aof --fix
/path/to/appendonlydir/appendonly.aof.1.base.aof /
kvdb-check-aof --fix
/path/to/appendonlydir/appendonly.aof.1.incr.aof

kvdb-check-rdb --fix /path/to/dump.rdb

```

2. 将修复后的干净数据文件拷贝到Apusic KVDB V2.0主节点数据目录，并修改权限：

```

#拷贝AOF文件（推荐，若用RDB则执行下一行）
cp /path/to/appendonlydir/ /kvdb/

#拷贝RDB文件
cp /path/to/dump.rdb /kvdb/

#赋权（避免服务无权限读取）
chmod 644 /kvdb/{appendonlydir/, dump.rdb}

```

步骤3：启动Apusic KVDB V2.0主节点服务

```

#启动主节点，指定专用配置文件
kvdb-server /kvdb/kvdb-master.yaml

#验证主节点启动状态
ps -ef | grep kvdb-server | grep master # 查看进程
tail -100f /kvdb/kvdb-master.log # 查看日志

```

日志显示 Ready to accept connections 表示主节点启动成功；

连接主节点验证数据完整性（核心，确保数据导入成功）：

```
kvdb-cli -p 6359 -a 密码 # 连接主节点
```

```
info keyspace # 查看键空间统计（确认数据量）
```

3.3.3.2 从节点迁移（所有从节点同步执行）

从节点无需单独导入数据，启动后通过Apusic KVDB V2.0内部主从同步，从已启动的Apusic KVDB V2.0主节点拉取全量数据，确保与主节点数据完全一致。

步骤1：配置文件转换与主从关联

1. 备份Apusic KVDB V2.0从节点默认配置：

```
cp /kvdb/kvdb.yaml /kvdb/kvdb-slave.yaml.bak
```

2. 将KVDB V2.0.2以下版本从节点配置转换为Apusic KVDB V2.0从节点专用配置（命名为 kvdb-slave.yaml）：

```
kvdb-conf-conv --server --go-yaml /path/to/conf.yaml --c-yaml  
/kvdb/kvdb-slave.yaml
```

3. 检查Apusic KVDB V2.0从节点配置，指定主节点信息，启用从节点身份：

```
vi /kvdb/kvdb-slave.yaml
```

关键配置确认 / 修改：

- port:6360（端口统一，若需修改则所有节点同步）；
- requirepass: 你的密码（若源主节点有密码，需保留，后续从节点需配置 masterauth）；
- dir:/kvdb/（数据目录，与实际规划一致）；
- appendonly:yes（建议启用AOF，与源端保持一致，若源端仅用RDB则设为no）；
- replicaof: 主节点IP主节点Port # 指向Apusic KVDB V2.0主节点的IP和端口；
- logfile:kvdb-slave.log（主节点独立日志，便于排查）。

步骤2：启动Apusic KVDB V2.0从节点服务

无需向从节点数据目录拷贝任何源数据文件，Apusic KVDB V2.0从节点会自动从主节点同步：

```
#启动从节点，指定专用配置文件
```

```
kvdb-server /kvdb/kvdb-slave.yaml
```

```
#验证从节点启动状态
```

```
ps -ef | grep kvdb-server # 查看进程
```

```
tail -100f /kvdb/kvdb-slave.log # 查看日志
```

日志显示 MASTER <-> REPLICA sync: Finished with success 表示从节点已成功连接主节点并开始同步。

步骤3：验证从节点同步状态

连接从节点，确认同步完成且数据与主节点一致：

```
kvdb-cli -p 6360 -a 密码 # 连接从节点
```

```
info replication # 查看同步状态
```

关键状态确认：

- role:slave：从节点身份正常；
- master_host：主节点IP、masterport:6359：主节点指向正确；
- master_link_status: up：主从连接正常。

3.3.4 注意事项

- 迁移全程必须先全量停止KVDB V2.0.2及以下版本的主从节点，禁止数据写入，否则会导致源数据文件不一致，导入后数据丢失；
- 主节点数据导入后必须先验证数据完整性，再启动从节点，避免从节点同步错误数据；
- 从节点无需拷贝任何源数据文件，由Apusic KVDB V2.0内部主从同步完成，手动拷贝会导致同步冲突；
- 主、从节点的核心配置必须一致（AOF开关、端口、密码等），否则会导致连接失败或同步异常；
- 迁移后观察集群运行10-20分钟，确认主从同步无延迟、日志无报错、业务访问正常。

3.4 哨兵模式迁移

KVDB V2.0.2及以下版本与Apusic KVDB V2.0无法跨版本主从同步、哨兵无法监控跨版本节点的场景，核心原则为全量停机->先迁主从节点（同第六章）->后迁哨兵节点->重新关联监控，通过「Apusic KVDB V2.0主从集群重建 + 哨兵重新监控」完成迁移，全程需短暂业务停机，具体步骤如下：

3.4.1 迁移前提

- 所有主从节点、哨兵节点均已安装Apusic KVDB V2.0，网络互通（开放6359服务端口、26359哨兵端口）；
- 已收集KVDB V2.0.2及以下版本的主从节点配置、数据文件、哨兵配置文件(sentinel_go.yaml)，并做好全量备份；
- 提前规划停机窗口（低峰期），停机时长为「KVDB V2.0.2及以下版本停机 + Apusic KVDB V2.0主从启动 + 哨兵启动」耗时（一般10-40分钟）；
- 所有目标节点时间同步，磁盘空间充足（主节点数据目录预留2倍源数据空间）。

3.4.2 准备工作

3.4.2.1 工具与配置前置

- [kvdb-check-rdb](#)：用于校验和修复KVDB V2.0.2及以下版本的RDB缓存文件（随Apusic KVDB V2.0安装包部署，位于/kvdb/目录下）；
- [kvdb-check-aof](#)：用于校验和修复KVDB V2.0.2及以下版本的AOF文件（同目录部署）；
- [kvdb-conf-conv](#)：用于转换KVDB V2.0.2及以下版本的配置文件为Apusic KVDB V2.0兼容格式（同目录部署）；

3.4.2.2 备份数据并全量停止KVDB V2.0.2及以下版本的哨兵 + 主从集群

- 停止KVDB V2.0.2及以下版本单节点的写入操作后，再次执行bgsave命令，生成最新RDB文件；备份源端KVDB配置；
- 必须先停止所有KVDB V2.0.2及以下版本的哨兵节点和主从节点，禁止数据写入和哨兵监控，确保源数据一致性。

#所有节点统一执行：停止KVDB V2.0.2及以下版本的KVDB主从 + 哨兵进程

```
ps -ef | grep -E 'kvdb-server | kvdb-sentinel' | grep -v grep | awk '{print $2}' | xargs kill -9
```

#验证进程已停止（无输出则成功）

```
ps -ef | grep -E 'kvdb-server | kvdb-sentinel' | grep -v grep
```

3.4.3 迁移步骤（按「Apusic KVDB V2.0主从集群->Apusic KVDB V2.0哨兵集群」顺序执行）

3.4.3.1 重建Apusic KVDB V2.0主从集群（核心，同上一章完整步骤）

- 严格按照上一章主从模式迁移的所有步骤，完成Apusic KVDB V2.0主从集群的搭建：
- 迁移Apusic KVDB V2.0主节点：配置转换 -> 数据校验导入 -> 启动 -> 验证数据完整性；
- 迁移所有Apusic KVDB V2.0从节点：配置转换（关联主节点） -> 启动 -> 验证主从同步；
- 完成主从集群整体验证：数据一致性、主从功能、日志无异常。

此步骤为哨兵迁移的基础，必须确保Apusic KVDB V2.0主从集群完全正常运行，再进行后续哨兵迁移。

3.4.3.2 哨兵节点迁移（所有哨兵节点同步执行）

哨兵节点无需导入数据，仅需将KVDB V2.0.2及以下版本的哨兵配置转换为Apusic KVDB V2.0兼容格式，并修改监控目标为已搭建完成的Apusic KVDB V2.0主节点。

步骤1：配置文件转换与监控目标修改

1. 备份Apusic KVDB V2.0哨兵默认配置：

```
cp /kvdb/sentinel.yaml /kvdb/sentinel-c.yaml.bak
```

2. 将KVDB V2.0.2及以下版本的哨兵配置转换为Apusic KVDB V2.0哨兵专用配置（命名为sentinel-c.yaml）：

```
kvdb-conf-conv --sentinel --go-yaml /path/to/sentinel_go.yaml --c-yaml /kvdb/sentinel.yaml
```

3. 检查Apusic KVDB V2.0哨兵配置，确认哨兵关键配置：

```
vi /kvdb/sentinel.yaml
```

关键配置修改或确认（核心为monitor配置项，指向Apusic KVDB V2.0主节点）：

#哨兵核心监控配置（替换为Apusic KVDB V2.0主节点信息，其余参数与源端保持一致）

SENTINEL:

```
sentinel monitor: mymaster 10.0.1.100 (示例IP, 请替换为实际IP) 6359 2 #
```

```

mymaster: 监控名称（与源端一致），后接C主节点IP、端口、quorum阈值

sentinel down-after-milliseconds: mymaster 30000 # 主观下线超时时间（与源端一致）

sentinel failover-timeout: mymaster 180000 # 故障转移超时时间（与源端一致）

sentinel auth-pass: mymaster 密码 # 与C主节点的requirepass一致，无密码则省略

```

其他配置确认：

- port:26359（哨兵默认端口，与源端一致）；
- logfile:sentinel.log（哨兵独立日志）；
- dir:/kvdb/（哨兵数据目录，存储监控状态）。

```

#所有哨兵节点同步启动，指定专用配置文件
kvdb-sentinel /kvdb/sentinel.yaml

#验证哨兵启动状态
ps -ef | grep kvdb-sentinel # 查看进程
tail -100f /kvdb/sentinel.log # 查看日志

```

步骤2：启动Apusic KVDB V2.0哨兵服务

日志显示 Sentinel ID is xxxxxxxxxxxxxxxxxxxxxxxx 表示哨兵启动成功。

关键验证：哨兵已成功发现Apusic KVDB V2.0主从集群的所有节点

```

#连接哨兵端口（26359），查询监控状态

kvdb-cli -p 26359

info sentinel # 查看哨兵监控信息

```

```
sentinel slaves mymaster # 查看监控的从节点列表
```

验证结果要求：

- 从节点列表中显示的IP和端口均为Apusic KVDB V2.0从节点信息，状态为online。

3.4.4 注意事项

- 迁移前必须全量停止KVDB V2.0.2及以下版本的哨兵 + 主从节点，避免哨兵持续监控已停机节点，或数据写入导致源文件不一致；
- 必须先完成Apusic KVDB V2.0主从集群的搭建和验证，再启动哨兵节点，否则哨兵会监控不到有效节点，导致迁移失败；
- 哨兵配置中监控名称（如mymaster）、quorum阈值、超时时间建议与源端保持一致，减少客户端配置修改；
- 所有哨兵节点的监控配置必须完全一致（监控目标、密码、阈值等），否则会导致哨兵集群分裂，故障转移异常；
- 迁移后观察10-20分钟，重点监控：哨兵日志无报错、主从切换无异常、同步延迟在可接受范围、业务访问无中断。

3.5 集群模式迁移

集群模式迁移适配下，KVDB V2.0.2及以下版本与Apusic KVDB V2.0无法跨版本建立集群、无法跨版本同步分片数据的场景，核心原则为全量停机->分片数据全量导出->Apusic KVDB V2.0集群初始化->分片数据批量导入->集群验证，因集群为多主多从分片架构，无跨版本同步可能，需通过「全量数据导出-导入」完成整集群迁移，全程需业务停机，具体步骤如下：

3.5.1 迁移前提

- 目标集群所有节点（与源集群节点数一致：M主N从）均已安装Apusic KVDB V2.0，网络互通（开放6359服务端口、16359集群gossip通信端口）；
- 已收集KVDB V2.0.2及以下版本的集群所有节点的配置文件、AOF、RDB数据文件，并记录源集群拓扑信息（执行cluster nodes 获取槽位分配、主从关系）；
- 提前规划长停机窗口（根据数据量调整，建议低峰期），停机时长为「数据导出 + 集群初始化 + 数据导入 + 验证」耗时（数据量10G内约30-60分钟）；
- 所有目标节点磁盘空间充足（每节点预留2倍源节点数据空间），时间同步，防火墙放行6359、16359端口。

3.5.2 准备工作

3.5.2.1 工具与配置前置

- [kvdb-check-rdb](#): 用于校验和修复KVDB V2.0.2及以下版本的RDB缓存文件（随Apusic KVDB V2.0安装包部署，位于/kvdb/目录下）；
- [kvdb-check-aof](#): 用于校验和修复KVDB V2.0.2及以下版本的AOF文件（同目录部署）；
- [kvdb-conf-conv](#): 用于转换KVDB V2.0.2及以下版本的配置文件为Apusic KVDB V2.0兼容格式（同目录部署）；

3.5.2.2 统一环境与记录源拓扑

1. 所有目标节点创建统一目录并赋权：

2. 观察KVDB V2.0.2及以下版本的集群核心拓扑：

连接KVDB V2.0.2及以下版本的集群任意节点，执行以下命令，保存输出结果（后续Apusic KVDB V2.0集群初始化完全复用）：

```
kvdb-cli cluster nodes # 记录：主从节点IP和端口、节点ID、槽位分配（如0-5460）
```

或者查看node.conf文件，其中记录有集群节点IP，端口，节点ID，槽位分配等信息。

```
29755a26cc60c056b3af39eb66dc5ccbfbdd20c1 127.0.0.1:6383@16383 slave
882b57434ef8c361fcea44c327379880181af1a6 0 1769505692837 2 connected

882b57434ef8c361fcea44c327379880181af1a6 127.0.0.1:6380@16380 master
- 0 1769505692432 2 connected 5461-10922

980c1df981902c2df617ca81d4df7219b1e86a8b 127.0.0.1:6384@16384
myself,slave 2793d6ad527eee582aa30890d7c8407e9c112b4f003 connected

43d25d48fac16f9b8f6b519f88a889179b8a6e87 127.0.0.1:6382@16382 slave
ac9f0c10e742b4570f6436db8af281c1d51ea570 0 1769505692331 1 connected

2793d6ad527eee582aa30890d7c8407e9c112b4f 127.0.0.1:6381@16381 master
- 0 1769505692332 3 connected 10923-16383
```

```
ac9f0c10e742b4570f6436db8af281c1d51ea570 127.0.0.1:6359@16359 master
- 0 1769505692228 1 connected 0-5460
```

记录每个KVDB V2.0.2及以下版本的每个主节点IP与端口号，备份对应机器上的AOF或RDB文件用于后续将数据迁移到Apusic KVDB V2.0集群：

3.5.2.3 备份数据并全量停止KVDB V2.0.2及以下版本的集群（禁止任何写入）

- 停止KVDB V2.0.2及以下版本单节点的写入操作后，再次执行bgsave命令，生成最新RDB文件；备份源端KVDB配置；

```
#所有KVDB V2.0.2及以下版本的集群节点统一执行：停止进程
ps -ef | grep kvdb-server | grep -v grep | awk '{print $2}' | xargs
kill -9

#验证进程已停止（无输出则成功）
ps -ef | grep kvdb-server | grep -v grep
```

3.5.2.4 数据文件预处理（所有源节点执行）

对每个KVDB V2.0.2及以下版本的主节点的AOF或RDB文件进行校验 + 修复，确保为干净数据文件，避免导入Apusic KVDB V2.0后出现数据损坏：

```
#优先使用AOF文件（每个节点单独处理，集群为分片存储，各节点数据独立）
kvdb-check-aof /path/to/nodeX/appendonlydir/appendonly.aof.manifest

#若无AOF则使用RDB文件
kvdb-check-rdb /path/to/nodeX/dump.rdb

#若文件损坏，执行修复
kvdb-check-aof --fix
/path/to/appendonlydir/appendonly.aof.1.base.rdb /
kvdb-check-aof --fix
/path/to/appendonlydir/appendonly.aof.1.base.aof /
kvdb-check-aof --fix
/path/to/appendonlydir/appendonly.aof.1.incr.aof
```

```
kvdb-check-rdb --fix /path/to/nodeX/dump.rdb
```

3.5.3 迁移步骤（按「配置转换->Apusic KVDB V2.0集群初始化->分片数据导入->集群状态验证」执行）

3.5.3.1 初始化Apusic KVDB V2.0空集群

步骤1: 所有Apusic KVDB V2.0节点配置转换与集群启用

1. 配置转换:

```
kvdb-conf-conv --server --go-yaml /path/to/nodeX/conf.yaml --c-yaml  
/kvdb/kvdb.yaml
```

2. 检查配置文件:

```
vi /kvdb/kvdb.yaml
```

关键配置:

- port:6359（集群服务端口，与源端一致）；
- logfile: kvdb-cluster-XXX.log（节点独立日志）；
- dir:/kvdb/（统一数据目录）；
- requirepass: 密码（与源端一致，所有节点相同）；
- cluster-enabled:yes（启用集群模式）；
- cluster-config-file:nodes-XXX.conf（每个节点单独的集群配置文件）；
- cluster-node-timeout:15000（集群节点超时时间，与源端一致）；
- cluster-require-full-coverage:yes（开启全槽位覆盖检查）。

3. 所有节点配置完成后，启动所有Apusic KVDB V2.0节点（仅启动服务，未形成集群）：

```
kvdb-server /kvdb/kvdb-cluster-XXX.yaml
```

3.5.3.1.1 验证所有节点启动成功（日志显示Ready to accept connections）

步骤2：执行集群初始化（槽位 + 主从分配）

选择任意一个Apusic KVDB V2.0主节点，执行kvdb-cli --cluster create命令，严格按照源拓扑指定节点IP、端口和槽位分配，示例以3主3从（源拓扑槽位 0-5460、5461-10922、10923-16383）为例：

```
#命令格式: kvdb-cli --cluster create 主1 主2 主3 从1 从2 从3 --cluster-replicas 1
#--cluster-replicas 1: 表示1主1从，与源端一致

kvdb-cli --cluster create 10.0.1.10（示例IP，请替换为实际IP）:6359
10.0.1.11（示例IP，请替换为实际IP）:6359 192.168.1.12:6359
192.168.1.13:6359 192.168.1.14:6359 192.168.1.15:6359 --cluster-replicas 1
```

执行成功后，验证集群初始化状态：

```
kvdb-cli -p 6359 -a 密码 cluster nodes
```

要求：全部槽位均已合理分配至集群各节点，所有节点状态为 myself，master 或 slave，无 fail 状态。

3.5.3.2 分片数据导入

先启动Apusic KVDB V2.0单机节点用于加载RDB或AOF文件，再使用Apusic KVDB V2.0 kvdb-cli --cluster import 命令，将单机节点中的数据自动重新映射导入到集群中，命令为集群标准化导入方式，无需手动操作数据文件，避免文件权限、格式不兼容等问题。

```
#核心命令格式（一行执行，换行符\仅为排版）

kvdb-cli --cluster import 集群主节点IP: 集群主节点Port \

--cluster-from 单节点IP: 单节点Port \
```

```
--cluster-copy \

--cluster-replace \

-a 集群密码
```

参数详解：

- --cluster-copy：必加参数，复制数据时保留源单节点的原始数据，避免源数据被删除；
- --cluster-replace：可选参数，若集群中存在与源节点同名的键，直接覆盖（建议迁移时添加，确保数据与源端一致，根据业务需求选择）。

因集群为分片存储，每个KVDB V2.0.2及以下版本的源分片仅存储对应槽位的数据，需按「源分片主节点->Apusic KVDB V2.0对应分片主节点」的顺序，逐个分片执行导入命令，全程在Apusic KVDB V2.0集群任意可操作节点执行即可，无需在目标主节点本地操作。

示例：以3主3从集群为例（源主1 -> C主1、源主2 -> C主2、源主3 -> C主3）

```
#导入分片1：源主1(192.168.1.1:6359) -> Apusic KVDB V2.0主1(10.0.1.10 (示例IP, 请替换为实际IP) :6359)
```

```
kvdb-cli --cluster import 10.0.1.10 (示例IP, 请替换为实际IP) :6359 \
```

```
--cluster-from 192.168.1.1:6359 \
```

```
--cluster-copy \
```

```
--cluster-replace \
```

```
-a 集群密码
```

```
#导入分片2：源主2(192.168.1.2:6359) -> Apusic KVDB V2.0主2(10.0.1.11 (示例IP, 请替换为实际IP) :6359)
```

```
kvdb-cli --cluster import 10.0.1.11 (示例IP, 请替换为实际IP) :6359 \
```

```
--cluster-from 192.168.1.2:6359 \
```

```

--cluster-copy \

--cluster-replace \

-a 集群密码

#导入分片3: 源主3(192.168.1.3:6359) -> Apusic KVDB V2.0主
3(192.168.1.12:6359)

kvdb-cli --cluster import 192.168.1.12:6359 \

--cluster-from 192.168.1.3:6359 \

--cluster-copy \

--cluster-replace \

-a 集群密码

```

3.5.3.3 集群状态修复与数据校验

因手动导入数据，需执行集群状态修复，确保槽位与数据匹配，无槽位丢失或数据错乱：

```

#验证槽位覆盖（必须显示16384个槽位全部分配）
kvdb-cli -p 6359 -a 密码 cluster info
cluster info 输出中 cluster_slots_assigned:16384 表示槽位全部分配，无遗
漏。

```

3.5.4 注意事项

- 集群迁移必须全量停机，且源集群所有节点禁止写入，因集群为分片存储，单个节点数据写入会导致分片数据不一致；
- 集群 gossip通信端口 16359(6359+10000) 必须放行，否则节点无法发现集群拓扑，导致集群分裂；
- 执行kvdb-cli --cluster import命令时，--cluster-copy为必加参数，省略该参数会直接删除KVDB V2.0.2及以下版本的源节点的原始数据，造成不可逆的源数据丢失；--cluster-replace 建议迁移时添加，确保覆盖Apusic

KVDB V2.0集群内的同名测试键，与源端数据完全一致；

- 若源集群存在数据倾斜（某分片数据量过大），导入前需提前评估Apusic KVDB V2.0集群对应节点的磁盘、内存资源；迁移后及时对倾斜分片进行扩容，避免集群出现性能瓶颈、影响业务访问速度；
- 迁移完成后建议观察集群运行10-20分钟，重点监控：各分片数据无丢失、主节点故障转移自动触发、跨分片操作正常执行、主从同步无持续延迟、业务访问延迟符合预期。

4 Redis 8.6及以下版本迁移至Apusic KVDB V2.0操作说明

4.1 准备工作

4.1.1 环境检查

- 目标服务器已安装Apusic KVDB V2.0，安装目录权限正常（可读可执行）；
- 已收集Redis的RDB快照文件(dump.rdb)，若启用AOF持久化则优先获取AOF文件；
- 目标服务器磁盘空间充足，至少为源数据文件大小的2倍（预留备份与运行空间）。

4.1.2 工具与文件准备

- [kvdb-check-rdb](#)：用于校验和修复KVDB V2.0.2及以下版本的RDB缓存文件（随Apusic KVDB V2.0KVDB安装包部署，位于/kvdb/目录下）；
- [kvdb-check-aof](#)：用于校验和修复KVDB V2.0.2及以下版本的KVDB的AOF文件（同目录部署）；

4.2 单机模式迁移

单机模式迁移适用于仅部署单台KVDB服务的场景，核心流程为「文件准备 -> 数据校验修复 -> 部署启动 -> 验证」，具体步骤如下：

4.2.1 迁移前提

- 目标服务器已安装Apusic KVDB V2.0，安装目录权限正常（可读可执行）；
- 已收集Redis的RDB快照文件(dump.rdb)，若启用AOF持久化则优先获取AOF文件；
- 目标服务器磁盘空间充足，至少为源数据文件大小的2倍（预留备份与运行空间）。

4.2.2 准备工作

4.2.2.1 数据与配置备份

- 检查Apusic KVDB V2.0安装目录、数据存储目录权限（确保KVDB用户可读写）；
- RDB文件和Redis配置文件复制至目标端KVDB指定备份目录（如/kvdb 644），确保KVDB可读取。

4.2.2.2 数据备份并停止Redis单节点

- 停止业务对Redis的写入操作，再次执行bgsave命令，生成最新RDB文件；备份源端Redis配置(redis.conf)；

- 必须先停止Redis单节点，禁止任何数据写入，确保源数据文件最终一致性。

#节点执行：停止Redis进程

```
ps -ef | grep kvdb-server | grep -v grep | awk '{print $2}' | xargs
kill -9
```

#验证进程已停止（无输出则表示成功）

```
ps -ef | grep kvdb-server | grep -v grep
```

4.2.3 迁移步骤

4.2.3.1 缓存数据迁移

1. 校验并修复源RDB或AOF文件（优先处理AOF文件，数据更完整）：

#若使用AOF文件

```
kvdb-check-aof /path/to/appendonly.aof
```

#若使用RDB文件

```
kvdb-check-rdb /path/to/dump.rdb
```

#若文件损坏，执行修复

```
kvdb-check-aof --fix /path/to/appendonly.aof
```

```
kvdb-check-rdb --fix /path/to/dump.rdb
```

2. 复制修复后的文件到Apusic KVDB V2.0的安装目录：

#查看Apusic KVDB V2.0数据目录（在kvdb.yaml中通过dir配置项确认，默认路径如下），复制RDB文件

```
cp /path/to/dump.rdb/ kvdb/
```

#若使用AOF文件，需先启用AOF配置（在kvdb.yaml中设置appendonly yes），再复制

```
cp /path/to/appendonly.aof /kvdb/
```

4.2.3.2 修改redis.conf配置

1. 修改Redis服务配置文件(redis.conf)，追加以下配置项到配置文件最下方：

```
worker-threads 1
```

4.2.3.3 启动KVDB服务

1. 启动Apusic KVDB V2.0服务，直接加载Redis的配置文件：

```
kvdb-server /kvdb/redis.conf
```

2. 验证服务启动状态：

```
#查看服务进程
ps -ef | grep kvdb-server

#查看日志（默认日志路径 / kvdb/server.log）
tail -100f /kvdb/server.log
```

4.2.3.4 验证启动

1. 连接Apusic KVDB V2.0服务，验证数据完整性：

```
#使用kvdb-cli连接（Redis默认端口6359，若配置密码需加-a选项）
kvdb-cli -p 6359 -a <密码># 示例值，非真实数据

#查看键空间统计信息
info keyspace
```

2. 验证功能可用性：执行写入、修改、删除等操作，确认服务正常响应，无配置或数据相关报错：

4.2.4 注意事项

- 迁移过程中需停止Redis服务，避免数据写入导致源文件损坏；
- 迁移后观察服务运行 10-20 分钟，确认无日志报错等异常。

4.3 主从模式迁移

主从模式迁移适用于部署一主多从KVDB服务的场景，核心流程为「文件准备 -> 数据校验修复 -> 部署启动 -> 验证」，具体步骤如下：

4.3.1 迁移前提

- 主、从目标服务器均已安装Apusic KVDB V2.0，安装目录权限正常（可读、可写、可执行），且主从节点网络互通（无防火墙拦截6359端口）；
- 已收集Redis主、从节点的配置文件（redis.conf）、AOF文件（优先，数据更完整）或RDB快照文件，并做好全量备份；
- 提前告知业务方停机窗口，建议选择业务低峰期操作，停机时长约为「数据文件拷贝 + 服务启动」耗时（根据数据量调整，一般5-30分钟）；
- 目标服务器磁盘空间充足，至少为源数据文件大小的2倍（预留备份与运行空间）。

4.3.2 准备工作

4.3.2.1 工具与文件准备

- [kvdb-check-rdb](#)：用于校验和修复KVDB V2.0.2及以下版本的RDB缓存文件（随Apusic KVDB V2.0KVDB安装包部署，位于/kvdb/目录下）；
- [kvdb-check-aof](#)：用于校验和修复KVDB V2.0.2及以下版本的KVDB的AOF文件（同目录部署）；

4.3.2.2 数据备份并停止Redis主从集群（全量停机）

- 停止业务对Redis的写入操作，再次执行bgsave命令，生成最新RDB文件；备份源端Redis配置（redis.conf）；
- 必须先停止所有Redis主、从节点，禁止任何数据写入，确保源数据文件最终一致性。

#主节点 + 所有从节点统一执行：停止Redis进程

```
ps -ef | grep kvdb-server | grep -v grep | awk '{print $2}' | xargs kill -9
```

#验证进程已停止（无输出则表示成功）

```
ps -ef | grep kvdb-server | grep -v grep
```

4.3.3 迁移步骤（按「主节点->所有从节点」顺序执行）

4.3.3.1 缓存数据迁移

1. 校验并修复源Redis主节点数据文件（主节点执行，若文件无损坏则跳过修复）：

#若使用AOF文件（推荐）

```
kvdb-check-aof /path/to/appendonly.aof
```

#若使用RDB文件

```
kvdb-check-rdb /path/to/dump.rdb
```

#若文件损坏，执行修复

```
kvdb-check-aof --fix /path/to/appendonly.aof
```

```
kvdb-check-rdb --fix /path/to/dump.rdb
```

2. 将修复后的干净数据文件拷贝到Apusic KVDB V2.0主节点数据目录，并修改权限：

#拷贝AOF文件（推荐，若用RDB则执行下一行）

```
cp /path/to/appendonly.aof /kvdb/
```

#拷贝RDB文件

```
cp /path/to/dump.rdb /kvdb/
```

#赋权（避免服务无权限读取）

```
chmod 644 /kvdb/{appendonly.aof, dump.rdb}
```

4.3.3.2 配置文件迁移 (redis.conf)

1. 将Redis主/从节点的redis.conf拷贝至KVDB主/从节点的根目录。
2. 修改Redis服务配置文件(redis.conf)，追加以下配置项到配置文件最下方：

```
worker-threads 1
```

4.3.3.3 启动KVDB主从节点服务

1. 先启动KVDB主节点：切换至主节点KVDB安装目录，执行启动命令，确认主节点服务启动成功；

```
#启动节点，指定专用配置文件
kvdb-server /kvdb/redis.conf

#验证从节点启动状态
ps -ef | grep kvdb-server    # 查看进程
tail -100f /kvdb/server.log  # 查看日志
```

2. 再启动所有KVDB从节点：逐个启动从节点KVDB服务，查看每个从节点的启动日志，确认无报错。

```
#启动从节点，指定专用配置文件
kvdb-server /kvdb/redis.conf

#验证从节点启动状态
ps -ef | grep kvdb-server    # 查看进程
tail -100f /kvdb/server.log  # 查看日志
```

4.3.3.4 主从关系验证

- 在KVDB主节点执行命令：info replication命令，查看从节点列表，确认所有从节点已正常连接主节点（master_link_status为up）；
- 在每个KVDB从节点执行命令：info replication命令，确认与主节点同步正常（offset一致）；
- 执行数据读写测试，主节点set key，从节点get key，确认数据同步正常。

4.3.4 注意事项

- 迁移全程必须先全量停止Redis主从节点，禁止数据写入，否则会导致源数据文件不一致，导入后数据丢失；
- 主节点数据导入后必须先验证数据完整性，再启动从节点，避免从节点同步错误数据；
- 从节点无需拷贝任何源数据文件，由Apusic KVDB V2.0内部主从同步完成，手动拷贝会导致同步冲突；
- 主、从节点的核心配置必须一致（AOF开关、端口、密码等），否则会导致连接失败或同步异常；
- 迁移后观察集群运行10-20分钟，确认主从同步无延迟、日志无报错、业务访问正常。

4.4 哨兵模式迁移

Apusic KVDB V2.0支持加载Redis配置redis.conf和sentinel.conf配置文件，核心流程为全量停机->先迁主从节点->后迁哨兵节点->重新关联监控，通过「主从集群重建 + 哨兵重新监控」完成迁移，全程需短暂业务停机，具体步骤如下：

4.4.1 迁移前提

- 所有主从节点、哨兵节点均已安装Apusic KVDB V2.0，网络互通（开放6359服务端口、26359哨兵端口）；
- 已收集Redis主从节点配置(redis.conf)、数据文件、哨兵配置文件(sentinel.conf)，并做好全量备份；
- 提前规划停机窗口（低峰期），停机时长为「Redis停机 + Apusic KVDB V2.0主从启动 + 哨兵启动」耗时（一般10-40分钟）；
- 所有目标节点时间同步，磁盘空间充足（主节点数据目录预留2倍源数据空间）。

4.4.2 准备工作

4.4.2.1 工具与文件准备

- [kvdb-check-rdb](#)：用于校验和修复KVDB V2.0.2及以下版本的RDB缓存文件（随Apusic KVDB V2.0KVDB安装包部署，位于/kvdb/目录下）；
- [kvdb-check-aof](#)：用于校验和修复KVDB V2.0.2及以下版本的KVDB的AOF文件（同目录部署）；

4.4.2.2 数据备份并全量停止Redis哨兵 + 主从集群

- 停止业务对Redis的写入操作，再次执行bgsave命令，生成最新RDB文件；备份源端Redis配置(redis.conf)；
- 必须先停止所有Redis哨兵节点和主从节点，禁止数据写入和哨兵监控，确保源数据一致性。

```
#所有节点统一执行：停止Redis主从 + 哨兵进程
```

```
ps -ef | grep -E 'redis-server | redis-sentinel' | grep -v grep |  
awk '{print $2}' | xargs kill -9
```

```
#验证进程已停止（无输出则成功）
```

```
ps -ef | grep -E 'redis-server | redis-sentinel' | grep -v grep
```

4.4.3 迁移步骤

4.4.3.1 缓存数据迁移

1. 校验并修复源Redis主节点数据文件（主节点执行，若文件无损坏则跳过修复）：

#若使用AOF文件（推荐）

```
kvdb-check-aof /path/to/appendonly.aof
```

#若使用RDB文件

```
kvdb-check-rdb /path/to/dump.rdb
```

#若文件损坏，执行修复

```
kvdb-check-aof --fix /path/to/appendonly.aof
```

```
kvdb-check-rdb --fix /path/to/dump.rdb
```

2. 将修复后的干净数据文件拷贝到Apusic KVDB V2.0主节点数据目录，并修改权限：

#拷贝AOF文件（推荐，若用RDB则执行下一行）

```
cp /path/to/appendonly.aof /kvdb/
```

#拷贝RDB文件

```
cp /path/to/dump.rdb /kvdb/
```

#赋权（避免服务无权限读取）

```
chmod 644 /kvdb/{appendonly.aof, dump.rdb}
```

4.4.3.2 配置文件迁移 (redis.conf)

1. 将Redis主/从节点的redis.conf拷贝至KVDB主/从节点的根目录。
2. 修改Redis服务配置文件(redis.conf)，追加以下配置项到配置文件最下方：

```
worker-threads 1
```

3. 哨兵配置文件(sentinel.conf)无需修改，可拷贝至KVDB哨兵节点的根目录。

4.4.3.3 启动KVDB主从节点服务

1. 先启动KVDB主节点：切换至主节点KVDB安装目录，执行启动命令，确认主节点服务启动成功；

```
#启动节点，指定专用配置文件
kvdb-server /kvdb/redis.conf

#验证从节点启动状态
ps -ef | grep kvdb-server    # 查看进程
tail -100f /kvdb/server.log  # 查看日志
```

2. 再启动所有KVDB从节点：逐个启动从节点KVDB服务，查看每个从节点的启动日志，确认无报错。

```
#启动从节点，指定专用配置文件
kvdb-server /kvdb/redis.conf

#验证从节点启动状态
ps -ef | grep kvdb-server    # 查看进程
tail -100f /kvdb/server.log  # 查看日志
```

4.4.3.4 启动KVDB哨兵节点服务

1. 再启动所有KVDB哨兵节点，查看哨兵服务进程与日志状态，确保启动成功。

```
#启动节点，指定专用配置文件
kvdb-sentinel /kvdb/sentinel.conf

#验证从节点启动状态
ps -ef | grep kvdb-sentinel  # 查看进程
tail -100f /kvdb/sentinel.log # 查看日志
```

4.4.3.5 哨兵功能验证

- 在任意KVDB哨兵节点执行命令：info sentinel命令，查看哨兵监控的主从节点数量、状态，确认哨兵能正常识别KVDB主从节点；
- 测试业务是否能通过哨兵自动获取新主节点地址，读写业务正常；测试完成后，恢复原KVDB主节点服务

4.4.4 注意事项

- 迁移前必须全量停止Redis哨兵 + 主从节点，避免哨兵持续监控已停机节点，或数据写入导致源文件不一致；

- 拷贝sentinel.conf后，若源端与目标端KVDB主节点IP不同，必须修改sentinel monitor参数中的主节点IP，否则哨兵无法监控到目标主节点；
- 必须先启动主从节点，再启动哨兵节点，确保哨兵启动时能正常发现主从节点，避免监控失败。
- 迁移后观察10-20分钟，重点监控：哨兵日志无报错、主从切换无异常、同步延迟在可接受范围、业务访问无中断。

4.5 集群模式迁移

Apusic KVDB V2.0支持加载Redis配置redis.conf和node.conf配置文件，核心流程为「文件准备 -> 数据校验修复 -> 部署启动 -> 验证」，具体步骤如下：

4.5.1 迁移前提

- 目标集群所有节点（与源集群节点数一致：M主N从）均已安装Apusic KVDB V2.0，网络互通；
- 已收集Redis集群所有节点的配置文件、AOF、RDB数据文件，并记录源集群拓扑信息（执行cluster nodes 获取槽位分配、主从关系）；
- 提前规划长停机窗口（根据数据量调整，建议低峰期），停机时长为「数据导出 + 集群初始化 + 数据导入 + 验证」耗时（数据量10G内约30-60分钟）；
- 所有目标节点磁盘空间充足（每节点预留2倍源节点数据空间）。

4.5.2 准备工作

4.5.2.1 工具与文件准备

- [kvdb-check-rdb](#)：用于校验和修复KVDB V2.0.2及以下版本的RDB缓存文件（随Apusic KVDB V2.0KVDB安装包部署，位于/kvdb/目录下）；
- [kvdb-check-aof](#)：用于校验和修复KVDB V2.0.2及以下版本的KVDB的AOF文件（同目录部署）；

4.5.2.2 数据备份并全量停止Redis集群（禁止任何写入）

- 停止业务对Redis的写入操作，再次执行bgsave命令，生成最新RDB文件；备份源端Redis配置（redis.conf）；

#所有Redis集群节点统一执行：停止进程

```
ps -ef | grep kvdb-server | grep -v grep | awk '{print $2}' | xargs kill -9
```

```
#验证进程已停止（无输出则成功）
```

```
ps -ef | grep kvdb-server | grep -v grep
```

4.5.3 迁移步骤

4.5.3.1 数据文件预处理（所有源节点执行）

对每个Redis主节点的AOF或RDB文件进行校验 + 修复，确保为干净数据文件，避免导入KVDB后出现数据损坏：

```
#优先使用AOF文件（每个节点单独处理，集群为分片存储，各节点数据独立）
```

```
kvdb-check-aof /path/to/nodeX/appendonly.aof
```

```
#若无AOF则使用RDB文件
```

```
kvdb-check-rdb /path/to/nodeX/dump.rdb
```

```
#若文件损坏，执行修复
```

```
kvdb-check-aof --fix /path/to/appendonly.aof
```

```
kvdb-check-rdb --fix /path/to/nodeX/dump.rdb
```

4.5.3.2 配置文件迁移（redis.conf, node.conf）

1. 将Redis每个集群节点的redis.conf拷贝至KVDB对应集群节点的根目录。
2. 修改Redis服务配置文件(redis.conf)，追加以下配置项到配置文件最下方：

```
worker-threads 1
```

3. 将Redis每个集群节点(node.conf)拷贝至KVDB对应集群节点的根目录。

4.5.3.3 启动KVDB集群节点

- 逐个启动所有KVDB集群节点，执行启动命令，查看每个节点的启动日志，确认无报错（重点关注集群连接及槽位加载情况）。

```
#启动节点，指定专用配置文件
```

```
kvdb-server /kvdb/redis.conf
```

```
#验证从节点启动状态
```

```
ps -ef | grep kvdb-server    # 查看进程  
tail -100f /kvdb/server.log  # 查看日志
```

4.5.3.4 集群状态验证

- 在任意KVDB节点执行命令：cluster info，确认集群状态为ok，所有槽位分配完整、无未分配槽位；
- 执行命令：cluster slots，查看槽位分配与源端一致，每个节点负责的槽位正确；
- 执行数据读写测试，跨分片查询测试，确认数据完整、业务正常；

4.5.4 注意事项

- 集群迁移必须全量停机，且源集群所有节点禁止写入，因集群为分片存储，单个节点数据写入会导致分片数据不一致；
- 若源集群存在数据倾斜（某分片数据量过大），导入前需提前评估Apusic KVDB V2.0集群对应节点的磁盘、内存资源；迁移后及时对倾斜分片进行扩容，避免集群出现性能瓶颈、影响业务访问速度；
- 迁移完成后建议观察集群运行10-20分钟，重点监控：各分片数据无丢失、主节点故障转移自动触发、跨分片操作正常执行、主从同步无持续延迟、业务访问延迟符合预期。

5 使用kvdb-shake进行迁移

5.1 概述

kvdb-shake 是 KVDB（兼容 Redis 协议）生态下的数据迁移与同步工具，主要用于在 Redis/KVDB 实例之间进行数据迁移、增量同步、备份恢复等操作。

5.1.1 适用场景

- Redis 实例数据迁移至 Apusic KVDB V2.0（支持单机、主从、哨兵、集群各类部署模式）；
- Apusic KVDB V2.0 不同实例之间的数据迁移或扩容；
- 在线不停机迁移（基于 PSync 的全量 + 增量同步）；
- 基于 RDB 文件的离线迁移；
- 基于 SCAN + Keyspace Notification 的全量 + 增量迁移。

5.1.2 兼容性

- 源端：Redis 2.8 ~ 8.x、各主流云厂商的 Redis-like 数据库；
- 目标端：Apusic KVDB V2.0、Redis、及各类兼容 Redis 协议的数据库；
- 部署模式：单机、主从、哨兵、集群（Cluster）。

5.2 工具准备

kvdb-shake 工具随 Apusic KVDB V2.0 安装包一同发布，位于 KVDB 安装目录的 `tools/kvdb-shake/` 子目录下，主要包含：

文件	说明
kvdb-shake / kvdb-shake.exe	主程序二进制文件（Linux/Windows）
shake.toml	默认配置文件（TOML 格式）
LICENSE	许可证文件
README.md	说明文件

使用前请确认二进制文件具备可执行权限（Linux 平台）：

```
chmod +x kvdb-shake
```

5.3 基本使用

以「将源端 Redis (127.0.0.1:6359) 数据迁移至目标端 KVDB (127.0.0.1:6359)」为例：

1. 编辑配置文件 `shake.toml`：

```
[sync_reader]
cluster = false
address = "127.0.0.1:6359"
username = ""
password = ""
tls = false
sync_rdb = true
sync_aof = true

[redis_writer]
cluster = false
address = "127.0.0.1:6359"
username = ""
password = "your_password"
tls = false

[advanced]
rdb_restore_command_behavior = "rewrite"
```

2. 启动迁移任务：

```
./kvdb-shake shake.toml
```

3. 观察输出日志，当出现持续递增的同步进度以及 "rdb done" / "sync done" 等提示时，表示迁移/同步正常进行。

5.4 配置文件说明

kvdb-shake 使用 TOML 格式的配置文件，主要由四个部分组成：`[xxx_reader]`、`[xxx_writer]`、`[filter]`、`[advanced]`。

5.4.1 Reader 配置（数据来源）

根据源端类型选择对应的 Reader 段，同一任务中只能启用一种 Reader。

5.4.1.1 sync_reader: 通过 PSync 协议实时同步

通过模拟从节点连接到主节点，接收源端 RDB 全量数据和 AOF 增量数据，是数据一致性最佳、对源端影响最小的方式，支持不停机迁移。

```
[sync_reader]
cluster = false           # 源端是否为集群
address = "127.0.0.1:6359" # 源端地址，集群模式填任意节点
username = ""             # ACL 用户名，传统鉴权留空
password = ""             # 密码，无鉴权留空
tls = false               # 是否启用 TLS/SSL，无需配置证书
sync_rdb = true           # 是否同步 RDB 全量数据
sync_aof = true           # 是否同步 AOF 增量数据
```

参数说明：

参数	说明
cluster	源端是否为集群模式，true 表示按集群处理
address	源端地址，集群模式下填写任意节点地址即可
username	ACL 鉴权用户名（按需填写）
password	鉴权密码（按需填写）
tls	是否启用 TLS/SSL，启用时无需配置证书
sync_rdb	是否同步 RDB 全量数据
sync_aof	是否同步 AOF 增量数据；关闭后仅做全量迁移

注意：若 kvdb-shake 与源端连接断开，建议调大源端 `client-output-buffer-limit replica` 参数。

5.4.1.2 scan_reader: 通过 SCAN 命令全量遍历

适用于无法使用 sync_reader（如目标端无 PSync 协议）的场景。包含 SCAN 全量阶段和 KSN（基于 Keyspace Notification）增量阶段。

```
[scan_reader]
cluster = false
address = "127.0.0.1:6359"
username = ""
```

```

password = ""
tls = false
dbs = []           # 指定需要同步的 DB, 例如 [0,2,5]
scan = true        # 是否进行 SCAN 全量同步
ksn = false        # 是否启用基于 Keyspace Notification 的增量
同步
count = 1          # 每次 SCAN 拉取的 key 数量, 建议从 1 起逐步
增加

```

参数说明:

参数	说明
cluster	源端是否为集群
address	源端地址, 集群模式下填任意节点
username/password	鉴权配置
tls	是否启用 TLS/SSL
dbs	指定需要同步的 DB 库列表, 空数组表示所有 DB
scan	是否执行 SCAN 全量同步
ksn	是否启用 KSN 增量同步
count	每次 SCAN 拉取的 key 数量, 数值越大效率越高但对源端压力越大

注意: `scan_reader` 的性能与一致性弱于 `sync_reader`, 建议优先选择 `sync_reader`。使用 KSN 时需先在源端开启 `notify-keyspace-events`, 且配置中需包含 `AE`。

5.4.1.3 rdb_reader: 从 RDB 文件离线读取

用于从已导出的 RDB 文件中读取数据进行迁移/恢复, 无需源端 Redis 运行中。

```

[rdb_reader]
filepath = "/path/to/dump.rdb"

```

5.4.1.4 aof_reader: 从 AOF 文件离线读取

用于从 AOF 文件中解析并恢复数据。

```
[aof_reader]
filepath = "/path/to/appendonly.aof"
```

5.4.2 Writer 配置（数据写入）

kvdb-shake 当前仅支持 `redis_writer` 一种 Writer，负责将处理后的数据写入目标端 KVDB/Redis 实例：

```
[redis_writer]
cluster = false           # 目标端是否为集群
address = "127.0.0.1:6359" # 目标端地址，集群模式填任意节点
username = ""             # ACL 用户名，传统鉴权留空
password = ""             # 密码，无鉴权留空
tls = false               # 是否启用 TLS/SSL
```

参数说明：

参数	说明
cluster	目标端是否为集群模式
address	目标端地址，集群模式下填写集群中任意节点地址
username/password	鉴权配置：ACL 体系同时填写，传统体系仅填写 password
tls	是否启用 TLS/SSL

注意：当目标端为集群时，多键命令必须满足同一 slot 的要求，否则写入会失败。建议目标端版本不低于源端版本，否则可能出现命令不兼容；如确需降级，可在 `[advanced]` 中将

`target_redis_proto_max_bulk_len` 设为 0，避免使用 `RESTORE` 命令。

5.4.3 Filter 配置（可选）

通过 `[filter]` 配置段可对同步的数据进行过滤和改写，包含两层能力：

- 规则过滤：通过 `allow_*`、`block_*` 等列表控制允许/阻止同步的 Key、DB、命令或命令组；
- Lua function 钩子：通过 `function` 选项编写 Lua 代码，对命令进行改写或拆分。

```
[filter]
allow_key_prefix = ["order:", "user:"]
```

```
block_key_prefix = ["tmp:"]  
function = "return redis.call('SET', KEYS[1], ARGV[1])"
```

5.4.4 Advanced 配置（可选）

高级配置段 [advanced] 提供若干调优参数，常用项包括：

参数	说明
rdb_restore_command_behavior	目标端存在相同 Key 时的处理策略：panic（默认，遇到重复 Key 即停止） / rewrite（覆盖目标 Key） / skip（跳过）
target_redis_proto_max_bulk_len	单条命令允许的最大字节数，超过将回退到普通命令（如 SET、HSET）
parallel	并发同步协程数
batch_size	批量写入的批大小
log_level	日志级别：debug / info / warn / error

注意：rdb_restore_command_behavior 仅在使用 RESTORE 命令时生效，覆盖范围包括 rdb_reader、sync_reader 的 RDB 阶段、scan_reader 的全量/KSN 阶段；sync_reader 的 AOF 阶段因直接转发命令不生效。

5.5 命令行参数

kvdb-shake 启动时仅需传入一个参数——配置文件的路径：

```
./kvdb-shake <配置文件路径>
```

例如：

```
./kvdb-shake shake.toml
```

其他说明：

- 不要在同一目录并行运行两个 kvdb-shake 进程，运行时产生的临时文件可能被覆盖，导致异常行为；
- 不要降低 Redis 版本（例如从 Redis 6.0 降到 5.0），kvdb-shake 会引入新的命令与编码方式，降级可能导致不兼容。

5.6 典型场景配置示例

5.6.1 场景一：Redis 单机迁移至 KVDB 单机（基于 PSync）

适用于「源端 Redis 与目标端 KVDB 均为单机实例」的在线迁移，配置简单、影响最小。

shake.toml：

```
[sync_reader]
cluster = false
address = "127.0.0.1:6359"
username = ""
password = ""
sync_rdb = true
sync_aof = true

[redis_writer]
cluster = false
address = "127.0.0.1:6359"
username = ""
password = "your_password"

[advanced]
rdb_restore_command_behavior = "rewrite"
```

执行迁移：

```
./kvdb-shake shake.toml
```

迁移过程中通过日志观察 RDB 全量同步与 AOF 增量同步进度，业务可在切换前进行数据校验。

5.6.2 场景二：Redis 主从迁移至 KVDB 主从

仅需在 Reader 段指定主节点地址即可（kvdb-shake 会自动从主节点拉取全量数据）。目标端 KVDB 主从应提前按照「主从模式迁移」章节完成部署与配置。

shake.toml：

```

[sync_reader]
cluster = false
address = "192.168.1.10:6359"    # 源端 Redis 主节点地址
username = ""
password = "source_password"
sync_rdb = true
sync_aof = true

[redis_writer]
cluster = false
address = "192.168.1.20:6359"    # 目标 KVDB 主节点地址
username = ""
password = "target_password"

[advanced]
rdb_restore_command_behavior = "rewrite"

```

5.6.3 场景三：Redis 哨兵/主从集群迁移至 KVDB 哨兵/主从集群

哨兵模式下数据迁移的本质仍为「主从到主从」的迁移，仅需配置主节点信息，kvdb-shake 通过 PSync 直接对接源端主节点：

```

[sync_reader]
cluster = false
address = "192.168.1.10:6359"    # 源端 Redis 主节点
username = ""
password = "source_password"
sync_rdb = true
sync_aof = true

[redis_writer]
cluster = false
address = "192.168.1.20:6359"    # 目标 KVDB 主节点
username = ""
password = "target_password"

```

[advanced]

```
rdb_restore_command_behavior = "rewrite"
```

哨兵节点本身无需迁移，目标 KVDB 哨兵按「哨兵模式迁移」章节配置并指向新部署的 KVDB 主节点即可。

5.6.4 场景四：Redis Cluster 迁移至 KVDB Cluster

集群模式下需将 `cluster` 字段设置为 `true`，Reader 与 Writer 均填写集群中任一节点地址：

[sync_reader]

```
cluster = true
address = "192.168.1.10:6359"    # 源端 Redis Cluster 任意节点
username = ""
password = "source_password"
sync_rdb = true
sync_aof = true
```

[redis_writer]

```
cluster = true
address = "192.168.1.20:6359"    # 目标 KVDB Cluster 任意节点
username = ""
password = "target_password"
```

[advanced]

```
rdb_restore_command_behavior = "rewrite"
```

注意：集群间迁移要求源、目标两端的 slot 分布能覆盖所有 key；若源数据中存在跨 slot 的多键命令，需确保目标端也能正确处理，否则写入会失败。

5.6.5 场景五：基于 SCAN + KSN 的全量 + 增量迁移

适用于源端无法开启 PSync、或者需要兼容老版本 Redis 的场景：

[scan_reader]

```
cluster = false
address = "127.0.0.1:6359"
```

```

username = ""
password = ""
scan = true
ksn = true
count = 10
dbs = []

[redis_writer]
cluster = false
address = "127.0.0.1:6359"
username = ""
password = "your_password"

[advanced]
rdb_restore_command_behavior = "rewrite"

```

使用前请先在源端执行以下命令开启 Keyspace Notification:

```
127.0.0.1:6359> CONFIG SET notify-keyspace-events "AKE"
```

5.6.6 场景六：从 RDB 文件离线迁移至 KVDB

适用于「源端 Redis 无法保持在线、只能提供 RDB 快照文件」的场景：

```

[rdb_reader]
filepath = "/path/to/dump.rdb"

[redis_writer]
cluster = false
address = "127.0.0.1:6359"
username = ""
password = "your_password"

[advanced]
rdb_restore_command_behavior = "rewrite"

```

执行迁移：

```
./kvdb-shake shake.toml
```

5.7 数据一致性校验

迁移完成后，建议通过以下方式校验两端数据一致性：

1. 目标端 KVDB 启动后，使用 `kvdb-cli` 连接目标实例，执行 `info keyspace` 查看键空间统计信息，与源端比对；
2. 抽样比对关键 Key：在源端与目标端分别执行 `DBSIZE`、随机抽样若干 Key 比对 Value；
3. 对于集群模式，分别在各分片主节点上抽样比对，确认每个 slot 数据一致；
4. 建议业务切换前保留源端 Redis 一段时间作为回滚备份。

5.8 注意事项

- 同一个目录中不要并行运行多个 `kvdb-shake` 进程，运行时产生的临时文件可能被覆盖；
- 建议目标端 KVDB 版本不低于源端 Redis 版本，避免命令编码不兼容；
- `sync_reader` 模式下，建议调大源端 `client-output-buffer-limit replica` 参数，防止同步过程中连接被强制断开；
- `scan_reader` 模式下，`count` 参数从 1 起逐步调大，避免对源端造成过大压力；KSN 阶段还需调大源端 `client-output-buffer-limit pubsub` 参数；
- 集群模式迁移时，源端多键命令需满足同一 slot 的要求，否则向目标端写入会失败；
- 迁移完成后建议保留源端 Redis 运行 10-20 分钟进行观察，确认无异常后再停服回收。

6 常见问题排查

6.1 工具无法运行

问题1: 终端输入kvdb-check-rdb提示"command not found" (Linux);

- 原因: 工具路径未添加到系统环境变量;
- 解决: 输入完整路径运行 (如 /usr/local/kvdb/kvdb-check-rdb) , 或添加环境变量 (echo "export PATH=/usr/local/kvdb/:\$PATH" >> /etc/profile, source /etc/profile) 。

6.2 校验或修复文件失败

问题1:kvdb-check-rdb或kvdb-check-aof 提示"Permission denied";

- 原因: 无文件读取或写入权限;
- 解决: 使用chmod修改文件权限 (chmod 644 dump.rdb), 或使用sudo运行工具 (sudo kvdb-check-rdb --fix dump.rdb)。

问题2: 修复后KVDB仍无法启动;

- 原因: 文件损坏过于严重, 或工具版本与KVDB版本不匹配;
- 解决: 使用备份文件恢复, 升级或降级工具版本, 与KVDB服务版本一致。

6.3 kvdb-cli 无法连接KVDB服务

问题1: 连接提示 "Could not connect to KVDB at 127.0.0.1:6359: Connection refused";

- 原因: KVDB服务未启动;
- 解决: 启动KVDB服务 (kvdb-server /path/to/kvdb.yaml, Linux) 。

问题2: 远程连接提示 "Could not connect to KVDB at 10.0.1.100 (示例IP, 请替换为实际IP) :6359: Connection timed out";

- 原因: 防火墙未开放6359端口, 或KVDB配置不允许远程连接;
- 解决: 开放防火墙端口 (firewall-cmd --add-port=6359/tcp --permanent, Linux), 修改KVDB配置 (bind 0.0.0.0, protected-mode no), 重启KVDB服务。

问题3：连接后执行命令提示 "NOAUTH Authentication required";

- 原因：KVDB服务设置了密码，连接时未输入；
- 解决：连接时用 -a 选项输入密码，或交互模式输入 AUTH 密码。

6.4 kvdb-shake 迁移常见问题

问题1：kvdb-shake 启动后长时间卡在 "rdb downloading" 阶段；

- 原因：源端数据量较大，或源端 `client-output-buffer-limit replica` 限制过小导致同步连接被强制断开；
- 解决：在源端执行 `CONFIG SET client-output-buffer-limit replica 1024mb 128mb 60` 调大缓冲区限制后重启 kvdb-shake。

问题2：kvdb-shake 启动报错 "auth failed" 或反复重连源端/目标端；

- 原因：源端或目标端配置了 ACL/传统密码鉴权，但 shake.toml 中的 `username` / `password` 未正确填写；
- 解决：ACL 鉴权需同时配置 `username` 与 `password`；传统鉴权仅需配置 `password`；无鉴权两者均留空。

问题3：向目标端 KVDB 集群写入时报 "CROSSSLOT" 错误；

- 原因：集群模式下，多键命令涉及的 Key 不在同一个 slot 中；
- 解决：源端避免使用跨 slot 的多键命令，或在源端通过 `{tag}` 形式将相关 Key 归入同一 slot；必要时改用单 Key 操作。

问题4：使用 scan_reader 时源端 CPU 占用飙升；

- 原因：DUMP 命令为 CPU 密集型，`count` 参数过大导致请求压力过高；
- 解决：将 `count` 从 1 起逐步调大，并在业务低峰期执行；优先使用 sync_reader 替代。

7 总结

本文档详细介绍了KVDB V2.0.2及以下版本与Redis 8.6及以下版本向Apusic KVDB V2.0的适配迁移流程，核心总结如下：

7.1 KVDB V2.0.2及以下版本向Apusic KVDB V2.0迁移流程

- 迁移前需完成环境检查、工具与文件准备，确保权限充足、文件完整；
- 配置文件迁移通过 kvdb-conf-conv 工具实现，需指定配置类型、源或目标文件路径，转换前务必备份目标配置文件；
- 缓存数据迁移基于RDB文件，需先通过 kvdb-check-rdb 校验文件完整性，修复损坏文件后部署到Apusic KVDB V2.0安装目录；
- 迁移后需验证服务启动状态、数据完整性和功能可用性，确保迁移成功；
- 迁移过程中若遇到工具运行、文件处理、服务启动等问题，可参考常见问题排查章节快速解决。

7.2 Redis 8.6及以下版本向Apusic KVDB V2.0迁移流程

- 安装KVDB，解压安装包至目标目录，完成基础安装；
- 停止源端Redis服务，收集对应模式核心文件（配置文件+数据文件）并备份；
- 使用Redis的配置文件按对应模式顺序启动KVDB节点，确保无报错；
- 验证数据及功能正常，切换业务流量，保留源端用于回滚。

7.3 基于 kvdb-shake 的在线迁移流程

- 在独立的迁移服务器上解压 Apusic KVDB V2.0 安装包中的 `tools/kvdb-shake/` 目录；
- 准备 shake.toml 配置文件，根据源端和目标端的部署模式选择对应的 Reader/Writer，并配置地址、鉴权、cluster 等参数；
- 执行 `./kvdb-shake shake.toml` 启动任务，观察 RDB 全量同步与 AOF/SCAN 增量同步进度；
- 同步完成后进行数据一致性校验，业务低峰期切换流量至 KVDB，保留源端作为回滚备份。

遵循本文档操作流程，可实现版本KVDB V2.0.2及以下版本与Redis 8.6及以下版本向Apusic KVDB V2.0的平滑迁移，保障缓存服务的连续性和数据安全性。

全国统一服务热线
4008-555-800

金蝶天燕云计算股份有限公司(简称“金蝶天燕云”)成立于2000年，前身为“金蝶中间件公司”，是金蝶集团旗下新一代软件基础云平台服务商，云计算国家标准制定企业，国家信创产业核心软件企业。金蝶天燕是国家863重点研发计划与核高基重大专项承接企业，也是“两网一站四库十二金”国家重点工程的基础平台提供商，产品广泛应用于政府、军工、金融、能源等关键行业，累计服务客户总数超过10万家。



云计算国家标准制定企业
金蝶集团旗下基础软件企业
信息技术应用创新核心企业
官网：www.apusic.com

