Apusic 金蝶天燕

生成时间：2026-05-07

安装手册

金蝶Apusic分布式缓存 V2.0.4

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

Apusic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 相关概念
- 3 安装说明
  - 3.1 概述
  - 3.2 环境支持
  - 3.3 推荐配置
- 4 AMDC缓存安装
  - 4.1 缓存核心安装包命名规范
  - 4.2 单机模式
    - 4.2.1 验证安装
  - 4.3 主从模式
  - 4.4 哨兵模式
  - 4.5 集群模式
    - 4.5.1 集群安装步骤
  - 4.6 Docker容器环境安装
    - 4.6.1 安装包说明
    - 4.6.2 安装步骤
  - 4.7 RPM包安装
    - 4.7.1 安装包安装
- 5 停止缓存核心服务
  - 5.1 停止方式
  - 5.2 紧急停止方式
  - 5.3 停止方式对比
- 6 License（许可）授权管理
  - 6.1 授权模式概述
  - 6.2 授权配置详解
    - 6.2.1 金蝶天燕本地授权
    - 6.2.2 金蝶KBC授权
    - 6.2.3 金蝶统一授权中心

- 6.3 授权文件管理
  - 6.3.0.1 授权文件位置
  - 6.3.0.2 授权热更新
  - 6.3.0.3 授权验证优先级
- 6.4 故障排查
  - 6.4.1 常见错误信息
  - 6.4.2 处理方案
- 7 缓存核心服务相关参数说明
- 7.1 网络配置参数
  - 7.1.1 监听地址配置
  - 7.1.2 端口配置
  - 7.1.3 保护模式
- 7.2 后台运行配置
  - 7.2.1 守护进程模式
  - 7.2.2 PID文件配置
- 7.3 安全认证配置
  - 7.3.1 密码认证
- 7.4 集群配置
  - 7.4.1 集群模式
  - 7.4.2 集群配置文件
  - 7.4.3 集群端口
- 7.5 哨兵配置
  - 7.5.1 哨兵端口
  - 7.5.2 哨兵监控配置
- 7.6 主从模式
  - 7.6.1 主从配置
- 7.7 存储配置
  - 7.7.1 数据文件配置
  - 7.7.2 AOF持久化配置
- 7.8 重要配置示例
  - 7.8.1 单机模式配置示例
  - 7.8.2 主从模式配置示例
  - 7.8.3 哨兵模式配置示例
  - 7.8.4 集群模式配置示例

- 8 Web管控台安装
  - 8.1 安装包介绍
  - 8.2 安装步骤
  - 8.3 停止Web管控台

1 前言

本文档为金蝶Apusic分布式缓存（AMDC）V2.0.4产品的安装手册，提供产品各种部署模式（含单机、哨兵、集群）在各操作系统、容器环境的安装部署过程指导以及相关配置说明。

1.1 适用对象

本文档适用于AMDC运维工程师、IT系统运维工程师、开发工程师、软件架构师等人员。

1.2 相关文档

了解更多AMDC V2.0.4产品相关的信息，请参阅以下AMDC V2.0.4产品手册文档集：

序号	手册文档	说明
1	金蝶Apusic分布式缓存 V2.0.4 快速使用手册	简单介绍了如何快速上手使用AMDC 。
2	金蝶Apusic分布式缓存 V2.0.4 安装手册	详细介绍如何在各操作系统上安装AMDC，以及AMDC服务启停操作，产品的注册过程。
3	金蝶Apusic分布式缓存 V2.0.4 缓存核心用户手册	详细介绍 AMDC 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分布式缓存 V2.0.4 管控台用户手册	详细介绍AMDC管控台相关功能的使用和操作说明。
5	金蝶Apusic分布式缓存 V2.0.4 开发手册	详细介绍基于各开发语言进行AMDC客户端应用开发的说明。
6	金蝶Apusic分布式缓存 V2.0.4 迁移手册	详细介绍AMDC历史版本迁移升级到V2.0.4版本的说明，以及Redis迁移到AMDC的说明。
7	金蝶Apusic分布式缓存 V2.0.4 运维手册	详细介绍AMDC的监控、运维、安全加固等运维说明。
8	金蝶Apusic分布式缓存 V2.0.4 性能优化手册	详细介绍AMDC性能调优的说明。

1.3 技术支持

AMDC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. 公司信息与联系方式
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 相关概念

名词	含义	使用说明
单机	存储数据，负责数据读写，负责数据同步	AMDC的默认模式，单机模式，单机模式下，主节点为单机模式
主从	从主节点复制数据，负责数据读写，不参与数据同步	主节点数据的实时副本。它主动连接到指定的主节点，并接收主节点发送的数据流，从而保持与主节点的数据同步。默认情况下，从节点是只读的，所有写操作必须发送到主节点。
哨兵	监控主从节点，负责主节点故障转移，负责主从节点同步	哨兵模式，当主节点挂掉时，自动切换为从节点，当从节点挂掉时，自动切换为主节点。
集群	由多台相互独立的计算机组成的计算机服务系统	在本文中，主从/哨兵/集群三种模式都可以算是集群，cluster集群模式特指多主节点数据分片存储的模式。

3 安装说明

3.1 概述

AMDC 产品包分为缓存和控制台两个部分，其中缓存是AMDC核心组件，控制台是AMDC管理组件。

3.2 环境支持

平台类型	系统类型
芯片类型	华为鲲鹏、海思、飞腾、兆芯等X86/ARM架构芯片
操作系统	银河麒麟系列、统信UOS、中标麒麟等
其他Linux系列	RedHat系列、CentOS系列、Ubuntu系列等

3.3 推荐配置

部署模式	操作系统	安装内容	硬件规格 (CPU/内存/硬盘)	服务器台数
单机	Linux	AMDC服务	8核/16G/100G	1
主从	Linux	AMDC服务	8核/16G/100G	2
哨兵	Linux	AMDC哨兵服务	8核/16G/100G	3
集群	Linux	AMDC控制台、AMDC服务	8核/16G/100G	3

4 AMDC缓存安装

4.1 缓存核心安装包命名规范

- 解压安装包：`AMDC-版本-构建日期-CPU架构.tar.gz` 解压到目标机器任意路径即可使用。
- docker安装包：`AMDC-Docker-构建日期-CPU架构-基础操作系统.tar.gz` 。
- RPM包：`AMDC-构建日期-CPU架构.rpm`，使用 `rpm -ivh 安装包名` 安装，安装完成后，其安装路径为 `/opt/amdc` 。

注：构建日期为AMDC产品包具体构建日期，同一个版本可能存在多个构建日期，获取安装包以最新日期为准。

4.2 单机模式

安装步骤：

1. 上传AMDC缓存核心安装包(`AMDC-版本-构建日期-CPU架构.tar.gz`)至目标服务器的安装目录下（如：/opt目录下）
2. 解压安装包：`tar -zxf AMDC-版本-构建日期-CPU架构.tar.gz`
3. 进入解压后的文件夹：`cd amdc`
4. 上传授权文件 `license.lic` 至每个节点安装目录 `/opt/amdc` 下。授权可参考本文档的AMDC 授权管理指南

```
[root@private amdc]# tree -L 2
.
├── acls.properties
├── amdc-benchmark
├── amdc-check-aof
├── amdc-check-rdb
├── amdc-cli
├── amdc-conf-conv
├── amdc-sentinel
├── amdc-server
├── amdc.yaml
├── license.lic       # 授权文件
└── sentinel.y
```

5. 执行启动命令启动AMDC缓存核心：`./amdc-server amdc.yaml`

4.2.1 验证安装

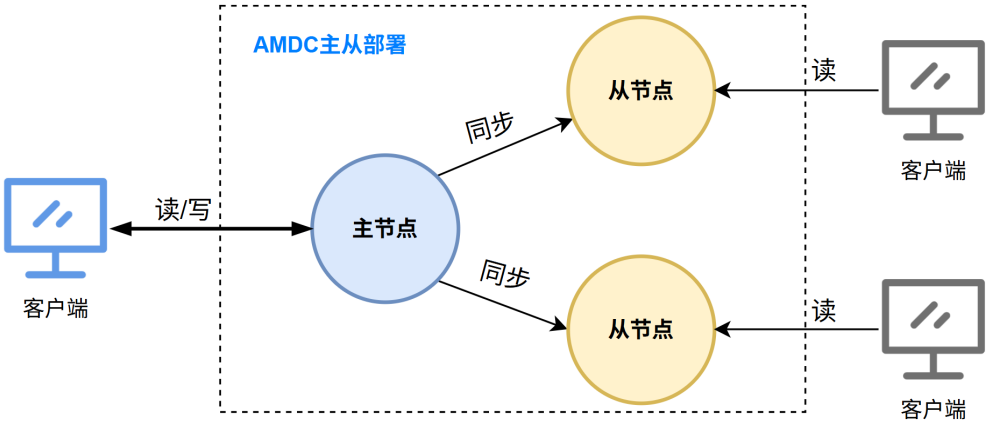
```
./amdc-cli
```

若正常打出info信息，即连接成功。

```
./amdc-cli
127.0.0.1:6359> info
# Server
amdc_version:2.0.4
amdc_git_sha1:61420466
amdc_git_dirty:0
amdc_build_id:905134ad61530534
amdc_build_date:2026-02-03
amdc_mode:cluster
os:Linux 3.10.0-1160.119.1.el7.x86_64 x86_64
arch_bits:64
monotonic_clock:POSIX clock_gettime
multiplexing_api:epoll
atomicvar_api:atomic-builtin
gcc_version:4.8.5
process_id:7075
process_supervised:no
run_id:69ffa6dd5f6bfa0250f5b0ee2bd38f8b9de143a3
tcp_port:6359
server_time_usec:1770275528738835
uptime_in_seconds:5201
uptime_in_days:0
hz:10
configured_hz:10
```

4.3 主从模式

主从部署图



如上图，将部署三个AMDC缓存核心，其中一个主节点，两个从节点，其部署步骤是：

- 1.准备三个服务器节点（以下节点为示例）
  - 主节点： 172.21.61.46
  - 从节点： 172.21.61.66和172.21.61.32
- 2.上传AMDC缓存核心安装包(`AMDC-版本-构建日期-CPU架构.tar.gz`)至目标服务器的安装目录下(如：/opt目录下)
- 3.解压安装包： `tar -zxf AMDC-版本-构建日期-CPU架构.tar.gz`
- 4.进入解压后的文件夹： `cd amdc`
- 5.上传授权文件 `license.lic` 至每个节点安装目录 `/opt/amdc` 下。授权可参考本文档的AMDC 授权管理指南

```
[root@private amdc]# tree -L 2
.
├── acls.properties
├── amdc-benchmark
├── amdc-check-aof
├── amdc-check-rdb
├── amdc-cli
├── amdc-conf-conv
├── amdc-sentinel
├── amdc-server
├── amdc.yaml
```

```
├── license.lic        # 授权文件
└── sentinel.yaml
```

- 5.配置amdc.yaml文件
  - 主节点： 修改 `amdc.yaml` 中的bind为当前机器IP

```
NETWORK:
# 监听的网络地址，默认仅监听本地回环地址，注释此行可监听所有接口（暴露在公网有安全风险）
bind:
    - "172.21.61.46"   # 修改为本机IP地址
    - "-::1"
# 保护模式，当未指定 bind 且无密码时，仅允许本地回环地址和 Unix 域套接字连接
```

  - 从节点：
    - 172.21.61.66: 修改 `amdc.yaml` 中的 `bind` 为当前机器IP和 `replicaof` 为主节点IP和端口

```
NETWORK:
# 监听的网络地址，默认仅监听本地回环地址，注释此行可监听所有接口（暴露在公网有安全风险）
bind:
  - "172.21.61.66"   # 添加本机IP
  - "-::1"

# 省略非配置项
REPLICATION:
  replicaof: "172.21.61.46 6359"   # 添加主节点IP和端口
```

    - 172.21.61.32: 修改 `amdc.yaml` 中的 `bind` 为当前机器IP和 `replicaof` 为主节点IP和端口

```
NETWORK:
# 监听的网络地址，默认仅监听本地回环地址，注释此行可监听所有接口（暴露在公网有安全风险）
bind:
  - "172.21.61.32"   # 添加本机IP
  - "-::1"

# 省略非配置项
REPLICATION:
  replicaof: "172.21.61.46 6359"   # 添加主节点IP和端口
```

- 6.在每个节点上启动AMDC缓存核心
  - 启动主节点：`./amdc-server amdc.yaml`
  - 启动从节点：`./amdc-server amdc.yaml`
- 7.验证安装

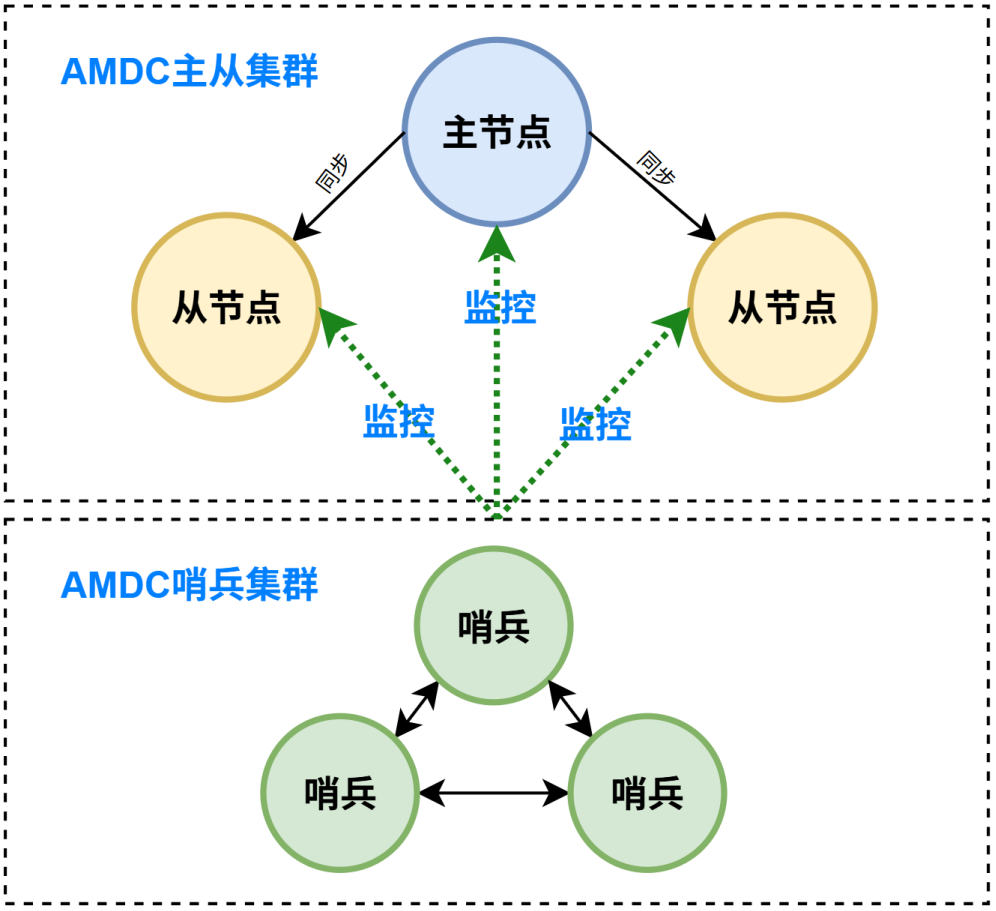
```
# 在主节点上执行
./amdc-cli -h 172.21.61.46 -p 6359 然后输入info replication
```bash
172.21.61.46:6359> info replication
# Replication
role:master
connected_slaves:2
slave0:ip=172.21.61.66,port=6359,state=online,offset=322,lag=0
slave1:ip=172.21.61.32,port=6359,state=online,offset=322,lag=1
master_failover_state:no-failover
master_replid:c246dd7c8902e4ca62fc72df6473d9386b502c8e
master_replid2:0000000000000000000000000000000000000000
master_repl_offset:322
second_repl_offset:-1
repl_backlog_active:1
```
```

```
repl_backlog_size:1048576
repl_backlog_first_byte_offset:1
```

- 8. 如上图，成功出现两个从节点信息，标识启动三节点的主从模式成功

4.4 哨兵模式

哨兵模式部署图



如上图，将部署三个AMDC缓存核心+三个哨兵，其部署步骤是：

- 1.准备三个服务器节点（以下节点为示例）
  - 172.21.61.46: 主节点+哨兵
  - 172.21.61.66: 从节点+哨兵
  - 172.21.61.32: 从节点+哨兵
- 2.上传AMDC缓存核心安装包(`AMDC-版本-构建日期-CPU架构.tar.gz`)至目标服务器的安装目录下(如：/opt目录下)
- 3.解压安装包： `tar -zxf AMDC-版本-构建日期-CPU架构.tar.gz`

- 4.进入解压后的文件夹: `cd amdc`
- 5.上传授权文件 `license.lic` 至每个节点安装目录 `/opt/amdc` 下。授权可参考本文档的AMDC 授权管理指南
- 6.缓存配置 `amdc.yaml` 文件
  - 主节点： 修改 `amdc.yaml` 中的bind为当前机器IP

```
NETWORK:
# 监听的网络地址，默认仅监听本地回环地址，注释此行可监听所有接口（暴露在公网有安全风险）
bind:
    - "172.21.61.46"   # 修改为本机IP地址
    - "-::1"
# 保护模式，当未指定 bind 且无密码时，仅允许本地回环地址和 Unix 域套接字连接
```

  - 从节点:
    - 172.21.61.66: 修改 `amdc.yaml` 中的 `bind` 为当前机器IP和 `replicaof` 为主节点IP和端口

```
NETWORK:
# 监听的网络地址，默认仅监听本地回环地址，注释此行可监听所有接口（暴露在公网有安全风险）
bind:
  - "172.21.61.66"   # 添加本机IP
  - "-::1"

# 省略非配置项
REPLICATION:
  replicaof: "172.21.61.46 6359"   # 添加主节点IP和端口
```

    - 172.21.61.32: 修改 `amdc.yaml` 中的 `bind` 为当前机器IP和 `replicaof` 为主节点IP和端口

```
NETWORK:
# 监听的网络地址，默认仅监听本地回环地址，注释此行可监听所有接口（暴露在公网有安全风险）
bind:
  - "172.21.61.32"   # 添加本机IP
  - "-::1"

# 省略非配置项
REPLICATION:
  replicaof: "172.21.61.46 6359"   # 添加主节点IP和端口
```

- 7.哨兵配置
  - 172.21.61.46 节点哨兵修改bind为当前IP，以及主节点的IP

```
NETWORK:
  # 监听的网络地址，默认仅监听本地回环地址，注释此行可监听所有接口（暴露在公网有安全风险）
  bind:
      - "172.21.61.46"
      - "-::1"
# 省略非配置项
SENTINEL:
  # 定义要监控的主节点，格式为 <master-name> <ip> <amdc-port> <quorum>（如 mymaster 127.0.0.1 6359 2 表示至少 2 个 Sentinel 认为主节点下线才判定为客观下线）
  sentinel monitor: mymaster 172.21.61.46 6359 2
```

  - 172.21.61.66 节点哨兵修改bind为当前IP，以及主节点的IP

```
NETWORK:
  # 监听的网络地址，默认仅监听本地回环地址，注释此行可监听所有接口
```

```
（暴露在公网有安全风险）
  bind:
      - "172.21.61.66"
      - "-::1"
# 省略非配置项
SENTINEL:
# 定义要监控的主节点，格式为 <master-name> <ip> <amdc-port>
<quorum>（如 mymaster 127.0.0.1 6359 2 表示至少 2 个
Sentinel 认为主节点下线才判定为客观下线）
  sentinel monitor: mymaster 172.21.61.46 6359 2
```

- 172.21.61.32 节点哨兵修改bind为当前IP，以及主节点的IP

```
NETWORK:
  # 监听的网络地址，默认仅监听本地回环地址，注释此行可监听所有接口
（暴露在公网有安全风险）
  bind:
      - "172.21.61.32"
      - "-::1"
# 省略非配置项
SENTINEL:
  # 定义要监控的主节点，格式为 <master-name> <ip> <amdc-port>
<quorum>（如 mymaster 127.0.0.1 6359 2 表示至少 2 个
Sentinel 认为主节点下线才判定为客观下线）
  sentinel monitor: mymaster 172.21.61.46 6359 2
```

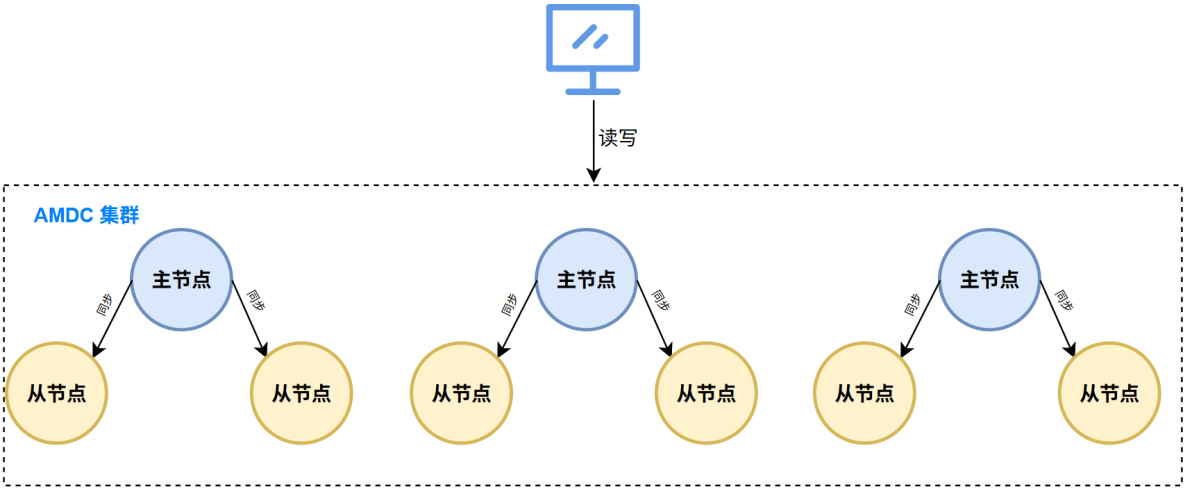
- 8.在每个节点上启动缓存和哨兵
  - 启动主节点：`./amdc-server amdc.yaml`
  - 启动从节点：`./amdc-server amdc.yaml`
  - 启动哨兵：`./amdc-sentinel sentinel.yaml`
- 9.验证哨兵模式
  - 使用 `amdc-cli` 客户端工具通过指定-p和-h参数，指定哨兵的IP和端口

```
./amdc-cli -p 26359 -h 172.21.61.46
172.21.61.46:26359> info sentinel
# Sentinel
sentinel_masters:1
sentinel_tilt:0
sentinel_running_scripts:0
sentinel_scripts_queue_length:0
sentinel_simulate_failure_flags:0
master0:name=mymaster,status=ok,address=172.21.61.46:6359,s
```

- 如上，出现主节点信息、从节点数量和哨兵数量都是正确的即部署成功。

4.5 集群模式

集群部署图



集群节点规划

服务器IP	角色	端口
172.21.61.46	集群节点1+从节点1	6359，6360
172.21.61.66	集群节点2+从节点2	6359，6360
172.21.61.32	集群节点3+从节点3	6359，6360

4.5.1 集群安装步骤

- 1.准备服务器节点
- 2.上传AMDC缓存核心安装包(`AMDC-版本-构建日期-CPU架构.tar.gz`)至上述目标服务器的安装目录下(如：/opt/amdc目录下)
- 3.解压安装包： `tar -zxf AMDC-版本-构建日期-CPU架构.tar.gz`
- 4.复制一份安装包为 `slave` 节点: `cp -r amdc amdc-slave`
- 5.上传授权文件 `license.lic` 至每个节点安装目录 `/opt/amdc/` 下，（包括slave）。授权可参考本文档的 AMDC 授权管理指南

```
[root@private opt]# tree -L 3
.
├── amdc
│   ├── amdc                    # 主节点目录
│   │   ├── acls.properties
│   │   ├── amdc-benchmark
│   │   ├── amdc-check-aof
│   │   ├── amdc-check-rdb
│   │   ├── amdc-cli
│   │   ├── amdc-conf-conv
│   │   ├── amdc-sentinel
│   │   ├── amdc-server
│   │   ├── amdc.yaml
│   │   ├── license.xml #授权文件
│   │   └── sentinel.yaml
│   └── amdc-slave              # 从节点目录
│       ├── acls.properties
│       ├── amdc-benchmark
│       ├── amdc-check-aof
│       ├── amdc-check-rdb
│       ├── amdc-cli
│       ├── amdc-conf-conv
│       ├── amdc-sentinel
```

```
|       ├── amdc-server
|       ├── amdc.yaml
|       ├── license.xml #授权文件
|       └── sentinel.yaml
```

- 6.集群配置
  - 集群节点1（172.21.61.46）： 修改`amdc.yaml`中的bind为当前机器IP和开启集群模式，并开启后台运行模式

```
NETWORK:
  # 监听的网络地址，默认仅监听本地回环地址，注释此行可监听所有接口（暴露在公网有安全风险）
  bind:
    - "172.21.61.46"   # 添加本机IP
    - "-::1"
GENERAL:
    # 是否以守护进程模式运行
    daemonize: yes
# 省略非配置项
CLUSTER:
  # 是否启用集群模式
  cluster-enabled: yes
```

  - 集群节点2（172.21.61.66）： 修改`amdc.yaml`中的bind为当前机器IP和开启集群模式，并开启后台运行模式

```
NETWORK:
  # 监听的网络地址，默认仅监听本地回环地址，注释此行可监听所有接口（暴露在公网有安全风险）
  bind:
    - "172.21.61.66"   # 添加本机IP
    - "-::1"
```

```
# 省略非配置项
CLUSTER:
  # 是否启用集群模式
  cluster-enabled: yes
```

- 集群节点3（172.21.61.32）： 修改 `amdc.yaml` 中的bind为当前机器IP和开启集群模式，并开启后台运行模式

```
NETWORK:
  # 监听的网络地址，默认仅监听本地回环地址，注释此行可监听所有接口（暴露在公网有安全风险）
  bind:
    - "172.21.61.32"   # 添加本机IP
    - "-::1"

# 省略非配置项
CLUSTER:
  # 是否启用集群模式
  cluster-enabled: yes
```

- 7.从节点配置
  - 由于主节点和从节点在同一台机器上，这里修改监听端口为： `6360`
  - 下面操作进入 `amdc-slave` 目录下操作
  - 从节点1（172.21.61.46）： 修改 `amdc.yaml` 中的bind为当前机器IP、Port和开启集群模式，并开启后台运行模式

```
NETWORK:
  # 监听的网络地址，默认仅监听本地回环地址，注释此行可监听所有接口（暴露在公网有安全风险）
  bind:
    - "172.21.61.46"   # 添加本机IP
```

```
    - "-::1"
  prot: 6360
GENERAL:
    # 是否以守护进程模式运行
    daemonize: yes
    # pid 文件路径
    pidfile: "/var/run/amdc-server-slave.pid"
# 省略非配置项
CLUSTER:
  # 是否启用集群模式
  cluster-enabled: yes
  # 集群配置文件路径，由节点自动维护
  cluster-config-file: "node-6360.conf"
```

- 从节点2（172.21.61.66）： 修改`amdc.yaml`中的bind为当前机器IP、Port和开启集群模式，并开启后台运行模式

```
NETWORK:
  # 监听的网络地址，默认仅监听本地回环地址，注释此行可监听所有接口（暴露在公网有安全风险）
  bind:
    - "172.21.61.66"   # 添加本机IP
    - "-::1"
  prot: 6360
GENERAL:
    # 是否以守护进程模式运行
    daemonize: yes
    # pid 文件路径
    pidfile: "/var/run/amdc-server-slave.pid"
# 省略非配置项
CLUSTER:
  # 是否启用集群模式
  cluster-enabled: yes
```

```
  # 集群配置文件路径，由节点自动维护
  cluster-config-file: "node-6360.conf"
```

- 从节点3（172.21.61.32）： 修改`amdc.yaml`中的bind为当前机器IP、Port和开启集群模式，并开启后台运行模式

```
NETWORK:
  # 监听的网络地址，默认仅监听本地回环地址，注释此行可监听所有接口（暴露在公网有安全风险）
  bind:
    - "172.21.61.32"   # 添加本机IP
    - "-::1"
  prot: 6360
GENERAL:
    # 是否以守护进程模式运行
    daemonize: yes
    # pid 文件路径
    pidfile: "/var/run/amdc-server-slave.pid"
# 省略非配置项
CLUSTER:
  # 是否启用集群模式
  cluster-enabled: yes
  # 集群配置文件路径，由节点自动维护
  cluster-config-file: "node-6360.conf"
```

- 8.在每个节点上启动缓存和从节点
  - 启动主节点：`./amdc-server amdc.yaml`
  - 启动从节点（切换至amdc-slave）：`./amdc-server amdc.yaml`
- 9.使用`amdc-cli`客户端工具创建集群
  - `amdc-cli`：`./amdc-cli --cluster create 172.21.61.46:6359 172.21.61.66:6359 172.21.61.32:6359 172.21.61.46:6360 172.21.61.66:6360 172.21.61.32:6360 --cluster-replicas 1`

> *注：* `amdc-cli --cluster` *命令使用手册可参考用户手册。*

- 创建过程中，要输入“yes”，确认集群节点安排，如下：

```
[root@private amdc]# ./amdc-cli --cluster create
172.21.61.46:6359 172.21.61.66:6359 172.21.61.32:6359
172.21.61.46:6360 172.21.61.66:6360 172.21.61.32:6360  --
cluster-replicas 1
>>> Performing hash slots allocation on 6 nodes...
Master[0] -> Slots 0 - 5460
Master[1] -> Slots 5461 - 10922
Master[2] -> Slots 10923 - 16383
Adding replica 172.21.61.66:6360 to 172.21.61.46:6359
Adding replica 172.21.61.32:6360 to 172.21.61.66:6359
Adding replica 172.21.61.46:6360 to 172.21.61.32:6359
M: a11b0a053f596321a4e310046e2b69bca8c96d4a 172.21.61.46:6359
  slots:[0-5460] (5461 slots) master
M: ab6dcd5d434ce3ec8ad199ac1a3d610ac33f3a7f 172.21.61.66:6359
  slots:[5461-10922] (5462 slots) master
M: 70451b10ef6f7793083b47a994c900cccc218b39 172.21.61.32:6359
  slots:[10923-16383] (5461 slots) master
S: 37ffc8538d24b332d45d6732b1df7e9386114522 172.21.61.46:6360
  replicates 70451b10ef6f7793083b47a994c900cccc218b39
S: 3ffc63268838f02dd4e1d66f11b5a6ffdc83cdfe 172.21.61.66:6360
  replicates a11b0a053f596321a4e310046e2b69bca8c96d4a
S: 4862dfb143cf5de0bebb64370f77c8bbe5bfc917 172.21.61.32:6360
  replicates ab6dcd5d434ce3ec8ad199ac1a3d610ac33f3a7f
Can I set the above configuration? (type 'yes' to accept): yes
>>> Nodes configuration updated
>>> Assign a different config epoch to each node
>>> Sending CLUSTER MEET messages to join the cluster
Waiting for the cluster to join
.
```

```
>>> Performing Cluster Check (using node 172.21.61.46:6359)
M: a11b0a053f596321a4e310046e2b69bca8c96d4a 172.21.61.46:6359
   slots:[0-5460] (5461 slots) master
   1 additional replica(s)
S: 4862dfb143cf5de0bebb64370f77c8bbe5bfc917 172.21.61.32:6360
   slots: (0 slots) slave
   replicates ab6dcd5d434ce3ec8ad199ac1a3d610ac33f3a7f
S: 37ffc8538d24b332d45d6732b1df7e9386114522 172.21.61.46:6360
   slots: (0 slots) slave
   replicates 70451b10ef6f7793083b47a994c900cccc218b39
S: 3ffc63268838f02dd4e1d66f11b5a6ffdc83cdfe 172.21.61.66:6360
   slots: (0 slots) slave
   replicates a11b0a053f596321a4e310046e2b69bca8c96d4a
M: 70451b10ef6f7793083b47a994c900cccc218b39 172.21.61.32:6359
   slots:[10923-16383] (5461 slots) master
   1 additional replica(s)
M: ab6dcd5d434ce3ec8ad199ac1a3d610ac33f3a7f 172.21.61.66:6359
   slots:[5461-10922] (5462 slots) master
   1 additional replica(s)
[OK] All nodes agree about slots configuration.
>>> Check for open slots...
>>> Check slots coverage...
[OK] All 16384 slots covered.
```

- 10.查看集群状态
  - `./amdc-cli --cluster info`

```
[root@private amdc]# ./amdc-cli --cluster info
172.21.61.46:6359> cluster info
cluster_state:ok
cluster_slots_assigned:16384
cluster_slots_ok:16384
```

```
cluster_slots_pfail:0
cluster_slots_fail:0
cluster_known_nodes:6
cluster_size:3
cluster_current_epoch:7
cluster_my_epoch:1
cluster_stats_messages_ping_sent:100
cluster_stats_messages_pong_sent:131
cluster_stats_messages_auth-ack_sent:1
cluster_stats_messages_sent:232
cluster_stats_messages_ping_received:131
cluster_stats_messages_pong_received:96
cluster_stats_messages_fail_received:1
cluster_stats_messages_auth-req_received:1
cluster_stats_messages_received:229
```

4.6 Docker容器环境安装

AMDC提供arm和x86的Docker系统安装包，可通过AMDC-Docker安装包直接部署AMDC.

4.6.1 安装包说明

1. AMDC-Docker压缩安装包名称

- x86_64: `AMDC-Docker-20260204-amd64-kylin.tar.gz`
- arm64: `AMDC-Docker-20260204-arm64-kylin.tar.gz`

注： 安装包日期可能不同，请根据实际情况选择。

2. 安装包目录结构

```
tree -l 1 AMDC-Docker-20260204-amd64-kylin
AMDC-Docker-20260204-amd64-kylin                # docker安装包解压后目录
|-- AMDC-Docker-20260204-amd64-kylin.tar.gz  # docker镜像文件
|-- amdc
|    |-- amdc.conf                                    # AMDC配置文件
```

```
|    |-- license.xml                                          # 授权文件
|-- amdc-dockercompose.yaml                          # docker compose配置文件
```

4.6.2 安装步骤

下面的操作以AMDC-Docker-20260204-amd64-kylin.tar.gz为例说明。

1. 解压和导入镜像

- 解压 `tar -zxvf AMDC-Docker-20260204-amd64-kylin.tar.gz`
- 进入解压后的文件夹 `cd AMDC-Docker-20260204-amd64-kylin`
- 加载镜像： `docker load < AMDC-Docker-20260204-amd64-kylin.tar.gz`

2. 修改授权文件

AMDC授权文件需要挂载到容器内部安装目录，当前样例是 `amdc/license.xml` ，在启动容器前，请更换授权文件。

> *授权文件挂载说明*
>
> - *license.xml 支持旧授权*
> - *license.lic 支持KBC授权，将申请的license.lic文件放到amdc目录，并在 `amdc-dockercompose.yaml` 配置挂载*
> - *acls.properties统一授权配置文件需要添加挂载到容器路径： `/opt/amdc`*

3. 镜像启动与停止

- 启动命令： `docker-compose -f amdc-dockercompose.yaml up -d`
- 停止命令： `docker-compose -f amdc-dockercompose.yaml down`

4. `amdc-dockercompose.yaml` 配置说明

```yaml
version: '3'
services:
  amdc:
    image: harbor.apusic.com/apusic/amdc:V2.0.4.20260204-kylin-amd64
    ports:
      - 6359:6359
    volumes:
      - ./amdc/amdc.yaml:/opt/amdc/amdc.yaml:z          # 配置文件挂载
      - ./amdc/license.xml:/opt/amdc/license.xml:z    # 授权文件挂载（若是licnese.lic，请修改）
```

```
      - ./amdc/acls.properties:/opt/amdc/acls.properties:z # 统一授权
文件挂载
      - ./amdc/dump.rdb:/opt/amdc/dump.rdb:z           # 缓存rdb文件挂
载（需要重新添加）
    working_dir: /opt/amdc
    command: ./amdc-server
```

4.7 RPM包安装

RPM（Red Hat Package Manager）安装包是一种在 Linux 系统中用于软件包管理的格式，最初由 Red Hat 开发，现在广泛用于基于 Red Hat 的发行版，AMDC提供arm和x86的RPM安装包，可通过 `rpm` 命令安装。其支持的操作系统有：

- Kylin
- OpenEuler
- CentOS

4.7.1 安装包安装

- x86_64: `AMDC-2.0.4-20260206-1.el7.x86_64.rpm`
- arm64: `AMDC-2.0.4-20260206-1.el7.aarch64.rpm`

1. 安装命令：

```
rpm -ivh AMDC-2.0.4-20260206-1.el7.x86_64.rpm
Preparing...
################################# [100%]
Updating / installing...
   1:amdc-2.0.4-1.el7
################################# [100%]
```

2. 安装完后，产品介质安装目录在 `/opt/amdc-2.0.4` 目录下
3. 将授权文件导入到 `/opt/amdc-2.0.4/license.xml` 文件中
4. 启动命令： `/opt/amdc-2.0.4/amdc-server`

5 停止缓存核心服务

AMDC 缓存核心提供多种停止方式，为确保数据安全和系统稳定性，建议按照以下优先级顺序选择合适的停止方法：

5.1 停止方式

1. 通过客户端发送 SHUTDOWN 命令

   这是最安全的停止方式，AMDC 会在完成数据持久化后再退出，确保数据完整性。

   ```
   ./amdc-cli -h [IP地址] -p [端口] -a [密码] shutdown
   ```

2. 交互式客户端停止

   当已连接到 AMDC 客户端时，可直接执行 shutdown 命令：

   ```
   127.0.0.1:6359> shutdown
   ```

执行流程：

1. 断开所有客户端连接
2. 根据配置执行 RDB/AOF 持久化操作
3. 安全退出服务进程

5.2 紧急停止方式

当 AMDC 服务无响应或无法正常停止时，可使用此方式。**注意：此操作可能导致数据丢失！**

操作步骤：

1. 查找 AMDC 进程 PID：

   ```
   ss -nltp | grep amdc
   # 或者
   ps -ef | grep amdc
   ```

2. 强制终止进程：

```
kill -9 [进程ID]
```

5.3 停止方式对比

停止方式	数据安全性	适用场景	推荐度
客户端 SHUTDOWN 命令	高	正常维护、计划停机	推荐
交互式 shutdown	高	调试、测试环境	推荐
强制 kill -9	低	服务无响应、紧急故障处理	不推荐

> ***重要提示***：*生产环境中请优先使用优雅停机方式，避免因强制终止导致的数据丢失风险。只有在 AMDC 服务完全无响应且无法通过正常方式停止时，才考虑使用强制终止方式。*

6 License（许可）授权管理

6.1 授权模式概述

金蝶Apusic分布式缓存（AMDC）V2.0.4 支持三种License许可授权验证模式，用户可根据实际需求选择合适的授权方式：

授权模式	适用场景	配置文件	特点
金蝶天燕本地授权	独立部署、离线环境	`license.xml`	本地文件授权，支持绑定IP、MAC地址
金蝶KBC授权	标准生产环境	`license.lic`	本地授权文件，通过授权码方式验证
金蝶统一授权中心	多租户、云环境	`acls.properties`	集中式授权管理，支持多租户

默认授权模式：金蝶天燕本地授权和金蝶KBC授权验证

6.2 授权配置详解

6.2.1 金蝶天燕本地授权

适用场景：离线环境、独立部署、测试环境

配置步骤：

1. 获取 `license.xml` 授权文件
2. 将授权文件放置在AMDC安装目录下
3. 在 `amdc.yaml` 中配置授权文件路径：

```
license: "license.xml"
```

6.2.2 金蝶KBC授权

适用场景：标准生产环境、需要设备绑定的场景、离线环境

配置步骤：

1. 获取设备特征码

   执行AMDC服务启动命令，系统将自动输出设备特征码：

```
./amdc-server
```

输出示例:

```
400987:C 05 Feb 2026 17:33:59.232 # Warning: no config file
specified, using the ./amdc.yaml
400987:C 05 Feb 2026 17:33:59.232 # oOOOoOOOoOOOo AMDC is
starting oOOOoOOOoOOOo
400987:C 05 Feb 2026 17:33:59.232 * AMDC KBC authorization
code: 'SZTY-1486655719' for mac: 00-15-5D-BC-65-3F
400987:C 05 Feb 2026 17:33:59.235 # No valid license file
found: ./license.xml or ./license.lic
400987:C 05 Feb 2026 17:33:59.235 # Local license verification
failed!
```

特征码说明:

- 格式：`SZTY-XXXXXXXXXX`（以SZTY开头）
- 绑定信息：MAC地址或其他硬件特征
- **注意**：每个设备的特征码唯一，不可跨设备使用

2. 申请授权文件

    1. 访问金蝶KBC授权系统（此操作由金蝶天燕销售、实施或研发人员操作）
    2. 使用获取的特征码申请授权
    3. 下载生成的 `license.lic` 文件

3. 配置授权文件

    1. 将 `license.lic` 文件放置在AMDC安装目录下
    2. 在 `amdc.yaml` 中配置授权文件路径：

```
license: "license.lic"
```

6.2.3 金蝶统一授权中心

适用场景：多租户环境、云平台部署、集中式授权管理

配置步骤：

1. 创建 `acls.properties` 配置文件
2. 配置授权中心连接信息：

```
# 是否开启统一授权中心
apusic_acls_enable=true

# 授权中心地址（多个地址用逗号分隔）
apusic_acls_authUrls=192.168.1.100:8080,192.168.1.101:8080

# 命名空间
apusic_acls_ns=public

# 租户名称（如不确定可填写为public）
apusic_acls_tenant=public
```

3. 将 `acls.properties` 文件放置在AMDC安装目录下

6.3 授权文件管理

6.3.0.1 授权文件位置

所有授权文件均需放置在AMDC安装目录下，与 `amdc-server` 可执行文件同级目录。

6.3.0.2 授权热更新

AMDC支持授权文件热更新，无需重启服务：

- 直接替换现有的授权文件内容
- 系统会自动检测并加载新的授权信息
- **注意**：确保新授权文件格式正确，避免授权失效

6.3.0.3 授权验证优先级

当多种授权文件同时存在时，AMDC按照以下优先级进行验证：

1. `acls.properties`（统一授权中心）

2. `license.lic`（KBC授权）
3. `license.xml`（本地授权）

6.4 故障排查

6.4.1 常见错误信息

- **"No valid license file found"**：未找到有效的授权文件
- **"Local license verification failed"**：授权文件无效或已过期
- **"Authorization code mismatch"**：特征码与设备不匹配

6.4.2 处理方案

1. 确认授权文件放置在正确目录
2. 验证授权文件格式和内容是否正确
3. 检查设备特征码是否与授权文件匹配
4. 确认授权文件是否在有效期内

7 缓存核心服务相关参数说明

7.1 网络配置参数

7.1.1 监听地址配置

参数名称	默认值	说明	使用方式
bind	127.0.0.1 -::1	监听的网络地址，指定客户端访问缓存的IP地址，除绑定地址外无法连接	可绑定多个地址，如： bind: - "127.0.0.1" - "192.168.0.190"

7.1.2 端口配置

参数名称	默认值	说明	使用方式
port	6359	监听的TCP端口，设为0则不监听TCP套接字	port: 6359

7.1.3 保护模式

参数名称	默认值	说明
protected-mode	yes	保护模式，当未指定bind且无密码时，仅允许本地回环地址和Unix域套接字连接

7.2 后台运行配置

7.2.1 守护进程模式

参数名称	默认值	说明	使用方式
daemonize	no	是否以守护进程模式运行，yes则后台运行并生成pid文件	daemonize: yes, 注：logfile参数需要填写，以便查看日志

7.2.2 PID文件配置

参数名称	默认值	说明	使用方式

pidfile	/var/run/amdc-sentinel.pid	pid文件路径，守护进程模式下默认生成	pidfile: "/var/run/amdc-server.pid"

7.3 安全认证配置

7.3.1 密码认证

参数名称	说明	使用方式
requirepass	设置密码，可以为空。从节点密码建议与主节点密码保持一致	requirepass: "123456" 注：在users.acl存在的情况下，server优先使用users.acl中的密码
masterauth	主节点的密码，若主节点无密码则不需要填，若有密码，不分主从所有节点都需要填写该参数	masterauth: "123456"

7.4 集群配置

7.4.1 集群模式

参数名称	默认值	说明	使用方式
cluster-enabled	-	是否开启集群模式	cluster-enabled: "yes"

7.4.2 集群配置文件

参数名称	默认值	说明
cluster-config-file	-	集群配置文件路径，由节点自动维护

7.4.3 集群端口

参数名称	默认值	说明	使用方式
port	6359	监听的TCP端口，设为0则不监听TCP套接字	port: 6359
集群总线	16359	集群节点间Gossip通信端口（=缓存端口+10000）	

7.5 哨兵配置

7.5.1 哨兵端口

参数名称	默认值	说明

port	26359	哨兵进程监听的TCP端口

7.5.2 哨兵监控配置

参数名称	说明	使用方式
sentinel monitor	监控的主节点配置	sentinel monitor: mymaster 127.0.0.1 6379 1
sentinel auth-pass	主节点密码校验	sentinel auth-pass: mymaster 123456

7.6 主从模式

7.6.1 主从配置

参数名称	说明	使用方式
replicaof	指定主节点地址和端口	replicaof: "172.21.61.46 6359"

7.7 存储配置

7.7.1 数据文件配置

参数名称	默认值	说明
dbfilename	"dump.rdb"	RDB文件名，不包括路径
dir	"./"	RDB文件、AOF文件夹、license文件、apusic-acls文件、aclfile文件、cluster-config-file文件等其他相对路径文件均受dir参数影响

7.7.2 AOF持久化配置

参数名称	默认值	说明	使用方式
appendonly	no	是否启用AOF持久化	appendonly: yes
appendfilename	"appendonly.aof"	AOF文件名称	appendfilename: "amdc.aof"
appendfsync	everysec	AOF刷盘策略: no（依赖系统） always（每次写入） everysec（每秒一次）	appendfsync: everysec
no-appendfsync-on-rewrite	no	AOF重写期间是否禁用主进程的fsync	no-appendfsync-on-rewrite: yes

auto-aof-rewrite-percentage	100	AOF自动重写的触发百分比	auto-aof-rewrite-percentage: 100
auto-aof-rewrite-min-size	64mb	AOF自动重写的最小文件大小	auto-aof-rewrite-min-size: 64mb
aof-load-truncated	yes	启动时是否加载被截断的AOF文件	aof-load-truncated: yes
aof-use-rdb-preamble	yes	AOF重写时是否使用RDB前缀	aof-use-rdb-preamble: yes

7.8 重要配置示例

7.8.1 单机模式配置示例

```
NETWORK:
  bind:
    - "0.0.0.0"   # 允许所有IP访问
  port: 6359
  protected-mode: yes

GENERAL:
  daemonize: yes   # 后台运行
  pidfile: "/var/run/amdc-server.pid"

SECURITY:
  requirepass: "your_password"   # 设置访问密码
```

7.8.2 主从模式配置示例

```
# 主节点配置
NETWORK:
  bind:
    - "172.21.61.46"   # 主节点IP
  port: 6359
```

```
GENERAL:
  daemonize: yes

SECURITY:
  requirepass: "master_password"   # 主节点密码

# 从节点配置
NETWORK:
  bind:
    - "172.21.61.66"   # 从节点IP
  port: 6359

GENERAL:
  daemonize: yes

REPLICATION:
  replicaof: "172.21.61.46 6359"   # 指定主节点

SECURITY:
  requirepass: "master_password"   # 从节点密码需与主节点一致
  masterauth: "master_password"    # 主节点密码
```

7.8.3 哨兵模式配置示例

```
NETWORK:
  bind:
    - "172.21.61.46"   # 哨兵节点IP
  port: 26359

GENERAL:
  daemonize: yes

SENTINEL:
  sentinel monitor: mymaster 172.21.61.46 6359 2   # 监控主节点
  sentinel auth-pass: mymaster master_password     # 主节点密码
```

7.8.4 集群模式配置示例

```
NETWORK:
  bind:
    - "172.21.61.46"  # 集群节点IP
  port: 6359

GENERAL:
  daemonize: yes

CLUSTER:
  cluster-enabled: yes  # 启用集群模式
  cluster-config-file: "nodes.conf"  # 集群配置文件

SECURITY:
  requirepass: "cluster_password"  # 集群密码
```

8 Web管控台安装

8.1 安装包介绍

- 安装包：`AMDC-Console-【版本】-【日期】-【硬件架构】.tar.gz`
-

8.2 安装步骤

其安装步骤主要有：

1. 上传AMDC控制台安装包至目标服务器的安装目录下(如：/opt目录下)
2. 解压安装包：`tar -zxf AMDC-Console-【版本】-【日期】-【硬件架构】.tar.gz`
3. 进入解压后的文件夹: `cd amdc-console`
4. 执行启动命令：`nohup ./amdc-console >nohup.out 2>&1 &`（后台启动）
5. 启动完成后即可通过浏览器访问AMDC控制台：用浏览器访问`http://ip:port`(其中ip表示部署AMDC控制台服务器的IP，port为控制台端口，如：`192.168.0.129:9001`

启动完成后即可通过浏览器访问控制台。

8.3 停止Web管控台

停止AMDC Web控制台的服务，需要强制结束AMDC控制台服务进程。

步骤：

1. 通过查看端口进程命令找到控制台进程PID：`netstat -nltp | grep 9001` 或者 `netstat -nltp | grep amdc-console`
2. 执行kill -9命令，强制杀死进程：`kill -9 进程ID`

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。

Apusic 金蝶天燕

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