



安裝手冊

金蝶Apusic分布式缓存 V2.0.6

版權所有 © 深圳市金蝶天燕雲計算股份有限公司2026。保留所有權利。

版權聲明

本文件所涉及的軟件著作權、版權等知識產權已依法進行了註冊，由金蝶天燕雲計算股份有限公司合法擁有。受《中華人民共和國著作權法》《計算機軟件保護條例》《知識產權保護條例》和相關國際版權條約、法律、法規以及其它知識產權法律和條約的保護。未經授權許可，不得非法使用。

免責聲明

本文件包含的版權資訊由金蝶天燕雲計算股份有限公司合法擁有，受法律的保護，金蝶天燕雲計算股份有限公司對本文件可能涉及到的非金蝶天燕雲計算股份有限公司的資訊不承擔任何責任。在法律允許的範圍內，您可以查閱並僅能夠在《中華人民共和國著作權法》規定的合法範圍內複製和打印本文件。任何單位和個人未經金蝶天燕雲計算股份有限公司書面授權許可，不得使用、修改、再發佈本文件的任何部分和內容，否則將被視為侵權，金蝶天燕雲計算股份有限公司有依法追究其責任的權利。

本文件如有更新，不另行通知。對本文件中的問題您可向金蝶天燕雲計算股份有限公司告知或查詢。未經本公司明確授予的任何權利均予保留。

商標聲明

 是深圳市金蝶天燕雲計算股份有限公司向中華人民共和國國家商標局申請註冊的註冊商標，註冊商標專用權由金蝶天燕合法擁有，受法律保護。未經金蝶天燕的書面許可，任何單位及個人不得以任何方式或理由對該商標的任何部分進行使用、複製、修改、傳播、抄錄或與其它產品捆綁使用銷售。凡侵犯金蝶天燕商標權的，金蝶天燕將依法追究其法律責任。本文件提及的其他所有商標或註冊商標，由各自的所有人擁有。

目錄

- 1 前言
 - 1.1 適用對象
 - 1.2 相關文檔
 - 1.3 技術支持
- 2 相關概念
- 3 安裝說明
 - 3.1 概述
 - 3.2 環境支持
 - 3.3 推薦配置
- 4 AMDC緩存安裝
 - 4.1 緩存核心安裝包命名規範
 - 4.2 單機模式
 - 4.2.1 驗證安裝
 - 4.3 主從模式
 - 4.4 哨兵模式
 - 4.4.1 哨兵模式運維 FAQ
 - 4.5 集群模式
 - 4.5.1 集群安裝步驟
 - 4.6 Docker容器環境安裝
 - 4.6.1 安裝包說明
 - 4.6.2 安裝步驟
 - 4.7 RPM包安裝
 - 4.7.1 安裝包安裝
- 5 停止緩存核心服務
 - 5.1 停止方式
 - 5.2 緊急停止方式
 - 5.3 停止方式對比
- 6 License（許可）授權管理
 - 6.1 授權模式概述
 - 6.2 授權配置詳解
 - 6.2.1 金蝶天燕本地授權
 - 6.2.2 金蝶KBC授權

- 6.2.3 金蝶統一授權中心
- 6.3 授權文件管理
 - 6.3.0.1 授權文件位置
 - 6.3.0.2 授權熱更新
 - 6.3.0.3 授權驗證優先級
- 6.4 故障排查
 - 6.4.1 常見錯誤信息
 - 6.4.2 處理方案
- 7 緩存核心服務相關參數說明
 - 7.1 網絡配置參數
 - 7.1.1 監聽地址配置
 - 7.1.2 端口配置
 - 7.1.3 保護模式
 - 7.2 後台運行配置
 - 7.2.1 守護進程模式
 - 7.2.2 PID文件配置
 - 7.3 安全認證配置
 - 7.3.1 密碼認證
 - 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 哨兵監控配置
 - 7.6 主從模式
 - 7.6.1 主從配置
 - 7.7 存儲配置
 - 7.7.1 數據文件配置
 - 7.7.2 AOF持久化配置
 - 7.8 重要配置示例
 - 7.8.1 單機模式配置示例
 - 7.8.2 主從模式配置示例
 - 7.8.3 哨兵模式配置示例

- 7.8.4 集群模式配置示例
- 8 Web管控台安裝
 - 8.1 安裝包介紹
 - 8.2 安裝包目錄結構
 - 8.3 安裝步驟
 - 8.3.1 驗證安裝
 - 8.3.2 停止Web管控台
 - 8.4 管控台核心配置說明
 - 8.5 管控台常用運維操作
 - 8.5.1 查看運行日誌
 - 8.5.2 重置預置管理員密碼
 - 8.5.3 備份與遷移

1 前言

本文檔為金蝶Apusic分佈式緩存（AMDC）V2.0.6產品的安裝手冊，提供產品在各種部署模式（含單機、哨兵、集群）下，於各操作系統及容器環境中的安裝部署過程指導，以及相關配置說明。

1.1 適用對象

本文檔適用於AMDC運維工程師、IT系統運維工程師、開發工程師、軟件架構師等人員。

1.2 相關文檔

了解更多AMDC V2.0.6產品相關的信息，請參閱以下AMDC V2.0.6產品手冊文檔集：

序號	手冊文檔	說明
1	金蝶Apusic分佈式緩存 V2.0.6 快速使用手冊	簡單介紹了如何快速上手使用AMDC。
2	金蝶Apusic分佈式緩存 V2.0.6 安裝手冊	詳細介紹如何在各操作系統上安裝AMDC，以及AMDC服務啟停操作和產品註冊過程。
3	金蝶Apusic分佈式緩存 V2.0.6 緩存核心用戶手冊	詳細介紹 AMDC 相關功能的使用、配置、管理及配套工具的使用方法。
4	金蝶Apusic分佈式緩存 V2.0.6 管控台用戶手冊	詳細介紹AMDC管控台相關功能的使用和操作說明。
5	金蝶Apusic分佈式緩存 V2.0.6 開發手冊	詳細介紹基於各開發語言進行AMDC客戶端應用開發的說明。
6	金蝶Apusic分佈式緩存 V2.0.6 遷移手冊	詳細介紹AMDC歷史版本遷移升級到V2.0.6版本的說明，以及Redis遷移到AMDC的說明。
7	金蝶Apusic分佈式緩存 V2.0.6 運維手冊	詳細介紹AMDC的監控、運維、安全加固等運維說明。
8	金蝶Apusic分佈式緩存 V2.0.6 性能優化手冊	詳細介紹AMDC性能調優的說明。

1.3 技術支持

AMDC產品提供全面的技術支持服務，您可以通過以下方式獲得技術支持：

- 網址: www.apusic.com
- 電話: 400-855-5800
- 郵箱: support@apusic.com
- 金蝶雲社區: <https://vip.kingdee.com/?productId=73&productLineId=14&lang=zh-CN>

您在取得技術支持時，請提供如下信息：

1. 您的姓名
2. 公司信息與聯繫方式
3. 操作系統及其版本
4. 產品版本號
5. 出現異常及錯誤的日誌、截圖等詳細信息

2 相關概念

名詞	含義	使用說明
單機	存儲數據，負責數據讀寫，負責數據同步	AMDC的默認模式，單機模式下只有一個節點
主從	從主節點複製數據，負責數據讀取，不參與寫操作	主節點數據的實時副本。它主動連接到指定的主節點，並接收主節點發送的數據流，從而保持與主節點的數據同步。默認情況下，從節點是只讀的，所有寫操作必須發送到主節點。
哨兵	監控主從節點，負責主節點故障轉移，負責主從節點同步	哨兵模式，當主節點掛掉時，自動將從節點切換為主節點，實現故障自動轉移。
集群	由多台相互獨立的計算機組成的計算機服務系統	在本文中，主從/哨兵/集群三種模式都可以算是集群，cluster集群模式特指多主節點數據分片存儲的模式。

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	華為鲲鹏、海思、飛騰、兆芯等X86/ARM架構芯片
操作系统	銀河麒麟系列、統信UOS、中標麒麟等
其他Linux系列	RedHat系列、CentOS系列、Ubuntu系列等

注意：AMDC V2.0.6 提供 *compat*（通用版）與 *perf*（全功能版）兩種緩存核心安裝包。*perf* 安裝包包含 MPTCP 多路徑傳輸與冷熱數據置換功能，要求 Linux 內核版本 ≥ 5.15 ，且使用 MPTCP 功能需服務器具備 2 個及以上物理網卡；不滿足上述條件的環境請選擇 *compat* 安裝包。詳見[緩存核心安裝包命名規範](#)。

3.3 推薦配置

部署模式	操作系统	安裝內容	硬件規格 (CPU/內存/硬盤)	服務器台數
單機	Linux	AMDC服務	8核/16G/100G	1
主從	Linux	AMDC服務	8核/16G/100G	2
哨兵	Linux	AMDC哨兵服務	8核/16G/100G	3
集群	Linux	AMDC控制台、AMDC服務	8核/16G/100G	3

4 AMDC緩存安裝

4.1 緩存核心安裝包命名規範

AMDC V2.0.6 緩存核心安裝包按功能集分為 **compat（通用版）** 與 **perf（全功能版）** 兩種變體：

變體	功能範圍	部署要求
compat（通用版）	包含除 MPTCP、冷熱數據置換外的全部功能	無特殊要求，兼容各主流 Linux 發行版
perf（全功能版）	包含全部功能（含 MPTCP 多路徑傳輸、冷熱數據置換）	Linux 內核版本 ≥ 5.15 ；使用 MPTCP 功能需服務器具備 2 個及以上物理網卡

選型建議： 需要使用 MPTCP 傳輸加速或冷熱數據置換能力，且部署環境滿足內核與網卡要求時，選擇 *perf* 安裝包；其餘場景建議使用 *compat* 安裝包。

各形態安裝包命名如下：

- 解壓安裝包： `AMDC-V2.0.6-構建日期-[compat|perf]-CPU架構.tar.gz` 解壓到目標機器任意路徑即可使用，例如：
 - `AMDC-V2.0.6-20260921-compatible-amd64.tar.gz` （x86 通用版）
 - `AMDC-V2.0.6-20260921-compatible-arm64.tar.gz` （ARM 通用版）
 - `AMDC-V2.0.6-20260921-perf-amd64.tar.gz` （x86 全功能版）
 - `AMDC-V2.0.6-20260921-perf-arm64.tar.gz` （ARM 全功能版）
- docker安裝包： `AMDC-V2.0.6-Docker-構建日期-compatible-CPU架構-基礎操作系統.tar.gz`，例如：
 - `AMDC-V2.0.6-Docker-20260921-compatible-amd64-kylin.tar.gz`
 - `AMDC-V2.0.6-Docker-20260921-compatible-arm64-kylin.tar.gz`
- RPM包： `AMDC-V2.0.6-構建日期-compatible-1.系統版本.CPU架構.rpm`，使用 `rpm -ivh 安裝包名` 安裝，安裝完成後，其安裝路徑為 `/opt/amdc`，例如：
 - `AMDC-V2.0.6-20260921-compatible-1.el7.x86_64.rpm`
 - `AMDC-V2.0.6-20260921-compatible-1.el7.aarch64.rpm`

註：構建日期為AMDC產品包具體構建日期，同一個版本可能存在多個構建日期，獲取安裝包以最新日期為準。

4.2 單機模式

安裝步驟：

1. 上傳AMDC緩存核心安裝包(`AMDC-V2.0.6-構建日期-[compat|perf]-CPU架構.tar.gz`)至目標服務器的安裝目錄下 (如: `/opt`目錄下)
2. 解壓安裝包: `tar -zxvf AMDC-V2.0.6-構建日期-[compat|perf]-CPU架構.tar.gz`
3. 進入解壓後的文件夾: `cd amdc`
4. 上傳授權文件 `license.lic` 至每個節點安裝目錄 `/opt/amdc` 下。授權可參考本文檔的[AMDC 授權管理指南](#)

```
[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      # 授權文件
└── sentinel.yaml
```

5. 執行啟動命令啟動AMDC緩存核心: `./amdc-server amdc.yaml`

4.2.1 驗證安裝

```
./amdc-cli
```

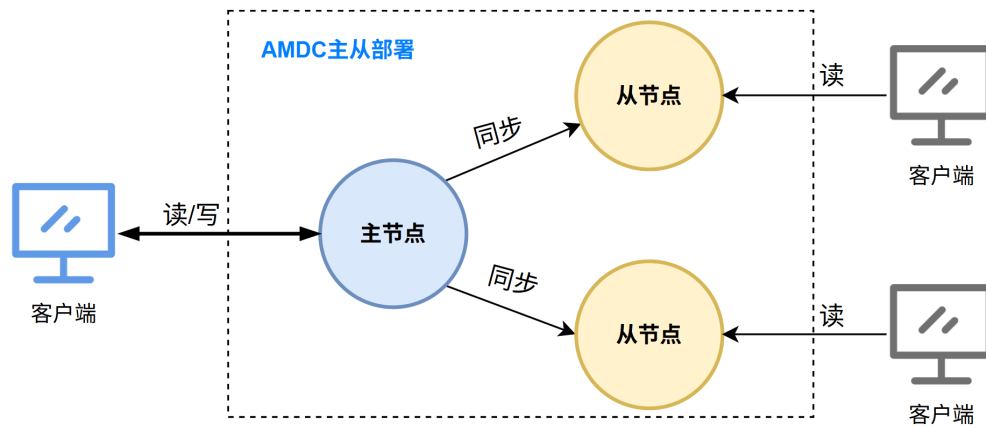
若正常打出info信息，即連接成功。

```
./amdc-cli
127.0.0.1:6359> info
# Server
amdc_version:2.0.6
amdc_git_sha1:61420466
```

```
amdc_git_dirty:0
amdc_build_id:905134ad61530534
amdc_build_date:2026-06-04
amdc_mode:standalone
os:Linux 5.10.0-281.0.0.184.oe2203sp4.aarch64 aarch64
arch_bits:64
monotonic_clock:ARM CNTVCT @ 100 ticks/us
multiplexing_api:epoll
atomicvar_api:atomic-builtin
gcc_version:10.3.1
process_id:321475
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
lru_clock:2168302
executable:/opt/amdc/amdc-server
config_file:/opt/amdc/amdc.yaml
worker_threads:1
mptcp_enabled:0
```

4.3 主從模式

主從部署圖



如上圖，將部署三個AMDC緩存核心，其中一個主節點，兩個從節點，其部署步驟是：

- 1.準備三個服務器節點（以下節點為示例）
 - 主節點： 10.0.1.46（示例IP，請替換為實際IP）
 - 從節點： 10.0.1.66（示例IP，請替換為實際IP）和10.0.1.32（示例IP，請替換為實際IP）
- 2.上傳AMDC緩存核心安裝包(`AMDC-V2.0.6-構建日期-[compat|perf]-CPU架構.tar.gz`)至目標服務器的安裝目錄下(如： /opt目錄下)
- 3.解壓安裝包： `tar -zxvf AMDC-V2.0.6-構建日期-[compat|perf]-CPU架構.tar.gz`
- 4.進入解壓後的文件夾： `cd amdc`
- 5.上傳授權文件 `license.lic` 至每個節點安裝目錄 `/opt/amdc` 下。授權可參考本文檔的[AMDC 授權管理指南](#)

```
[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      # 授權文件
└─ sentinel.yaml
```

- 6.配置amdc.yaml文件

- 主節點：修改 `amdc.yaml` 中的bind為當前機器IP

```
NETWORK:
# 監聽的網絡地址，默認僅監聽本地回環地址，註釋此行可監聽所有接口（暴露
在公網有安全風險）
bind:
  - "10.0.1.46 (示例IP, 請替換為實際IP) " # 修改為本機IP地址
  - "-::1"
# 保護模式，當未指定 bind 且無密碼時，僅允許本地回環地址和 Unix 域
套接字連接
```

- 從節點：

- 10.0.1.66 (示例IP, 請替換為實際IP)：修改 `amdc.yaml` 中的 `bind` 為當前機器IP和 `replicaof` 為主節點IP和端口

```
NETWORK:
# 監聽的網絡地址，默認僅監聽本地回環地址，註釋此行可監聽所有接
口（暴露在公網有安全風險）
bind:
  - "10.0.1.66 (示例IP, 請替換為實際IP) " # 添加本機IP
  - "-::1"

# 省略非配置項
REPLICATION:
  replicaof: "10.0.1.46 (示例IP, 請替換為實際IP) 6359"
# 添加主節點IP和端口
```

- 10.0.1.32 (示例IP, 請替換為實際IP) : 修改 `amdc.yaml` 中的 `bind` 為當前機器IP和 `replicaof` 為主節點IP和端口

```
NETWORK:
# 監聽的網絡地址, 默認僅監聽本地回環地址, 註釋此行可監聽所有接口 (暴露在公網有安全風險)
bind:
  - "10.0.1.32 (示例IP, 請替換為實際IP) " # 添加本機IP
  - "-::1"

# 省略非配置項
REPLICATION:
  replicaof: "10.0.1.46 (示例IP, 請替換為實際IP) 6359"
# 添加主節點IP和端口
```

- 7.在每個節點上啟動AMDC緩存核心
 - 啟動主節點: `./amdc-server amdc.yaml`
 - 啟動從節點: `./amdc-server amdc.yaml`
- 8.驗證安裝

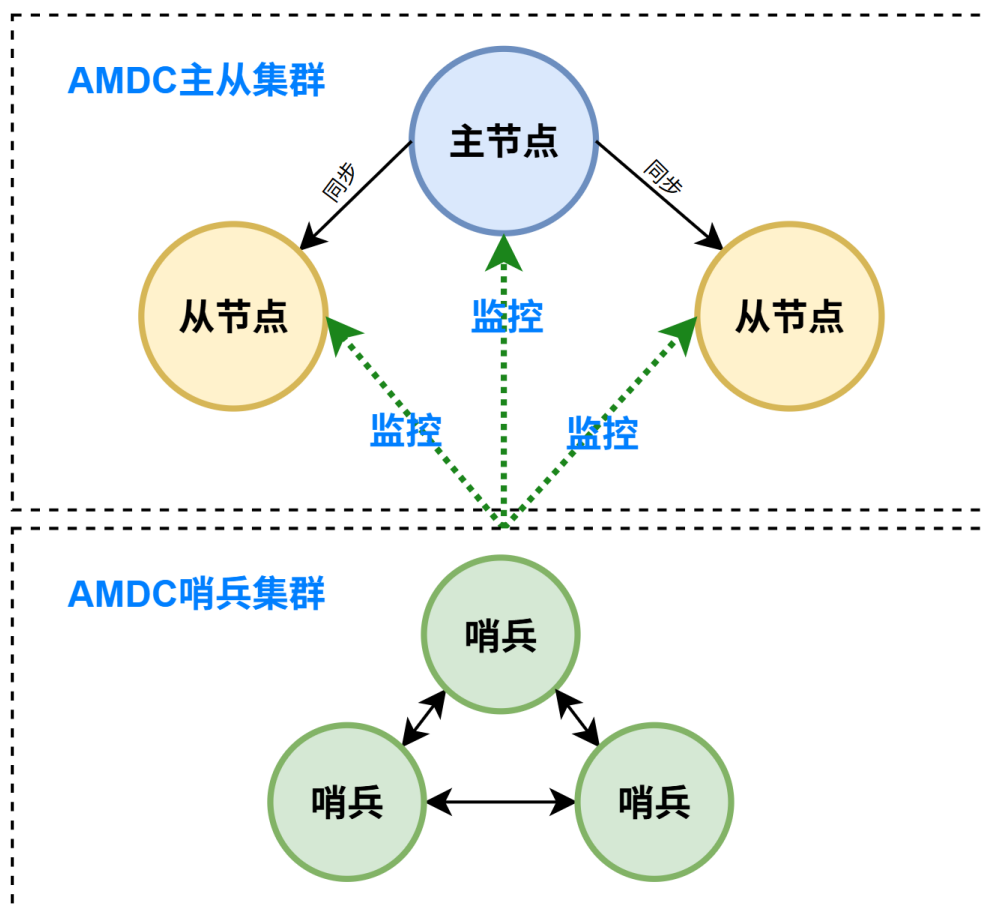
```
# 在主節點上執行
./amdc-cli -h 10.0.1.46 (示例IP, 請替換為實際IP) -p 6359 然後輸入
info replication
```bash
10.0.1.46 (示例IP, 請替換為實際IP) :6359> info replication
Replication
role:master
connected_slaves:2
slave0:ip=10.0.1.66 (示例IP, 請替換為實際IP) ,port=6359,state=online,offset=322,lag=0
slave1:ip=10.0.1.32 (示例IP, 請替換為實際IP)
```

```
IP) ,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
```

- 9. 如上述輸出所示，成功出現兩個從節點信息，標誌啟動三節點的主從模式成功

## 4.4 哨兵模式

### 哨兵模式部署圖



如上圖，將部署三個AMDC緩存核心+三個哨兵，其部署步驟是：

- 1.準備三個服務器節點（以下節點為示例）

- 10.0.1.46 (示例IP, 請替換為實際IP) : 主節點+哨兵
- 10.0.1.66 (示例IP, 請替換為實際IP) : 從節點+哨兵
- 10.0.1.32 (示例IP, 請替換為實際IP) : 從節點+哨兵
- 2.上傳AMDC緩存核心安裝包( `AMDC-V2.0.6-構建日期-[compat|perf]-CPU架構.tar.gz` )至目標服務器的安裝目錄下(如: `/opt`目錄下)
- 3.解壓安裝包: `tar -zxvf AMDC-V2.0.6-構建日期-[compat|perf]-CPU架構.tar.gz`
- 4.進入解壓後的文件夾: `cd amdc`
- 5.上傳授權文件 `license.lic` 至每個節點安裝目錄 `/opt/amdc` 下。授權可參考本文檔的[AMDC 授權管理指南](#)
- 6.緩存配置 `amdc.yaml` 文件
  - 主節點: 修改 `amdc.yaml` 中的`bind`為當前機器IP

**NETWORK:**

# 監聽的網絡地址, 默認僅監聽本地回環地址, 註釋此行可監聽所有接口 (暴露在公網有安全風險)

**bind:**

- "10.0.1.46 (示例IP, 請替換為實際IP) " # 修改為本機IP地址
- "-::1"

# 保護模式, 當未指定 `bind` 且無密碼時, 僅允許本地回環地址和 Unix 域套接字連接

- 從節點:
  - 10.0.1.66 (示例IP, 請替換為實際IP) : 修改 `amdc.yaml` 中的 `bind` 為當前機器IP和 `replicaof` 為主節點IP和端口

**NETWORK:**

# 監聽的網絡地址, 默認僅監聽本地回環地址, 註釋此行可監聽所有接口 (暴露在公網有安全風險)

**bind:**

- "10.0.1.66 (示例IP, 請替換為實際IP) " # 添加本機IP
- "-::1"

```
省略非配置項
REPLICATION:
 replicaof: "10.0.1.46 (示例IP, 請替換為實際IP) 6359"
添加主節點IP和端口
```

- 10.0.1.32 (示例IP, 請替換為實際IP) : 修改 `amdc.yaml` 中的 `bind` 為當前機器IP和 `replicaof` 為主節點IP和端口

```
NETWORK:
監聽的網絡地址, 默認僅監聽本地回環地址, 註釋此行可監聽所有接口 (暴露在公網有安全風險)
bind:
 - "10.0.1.32 (示例IP, 請替換為實際IP) " # 添加本機IP
 - "-::1"

省略非配置項
REPLICATION:
 replicaof: "10.0.1.46 (示例IP, 請替換為實際IP) 6359"
添加主節點IP和端口
```

- 7.哨兵配置

- 10.0.1.46 (示例IP, 請替換為實際IP) 節點哨兵修改bind為當前IP, 以及主節點的IP

```
NETWORK:
監聽的網絡地址, 默認僅監聽本地回環地址, 註釋此行可監聽所有接口 (暴露在公網有安全風險)
bind:
 - "10.0.1.46 (示例IP, 請替換為實際IP) "
 - "-::1"

省略非配置項
SENTINEL:
```

```
定義要監控的主節點，格式為 <master-name> <ip> <amdc-port>
<quorum> (如 mymaster 127.0.0.1 6359 2 表示至少 2 個
Sentinel 認為主節點下線才判定為客觀下線)

sentinel monitor: mymaster 10.0.1.46 (示例IP, 請替換為實際
IP) 6359 2
```

- 10.0.1.66 (示例IP, 請替換為實際IP) 節點哨兵修改bind為當前IP, 以及主節點的IP

NETWORK:

```
監聽的網絡地址，默認僅監聽本地回環地址，註釋此行可監聽所有接口
(暴露在公網有安全風險)
```

```
bind:
```

```
- "10.0.1.66 (示例IP, 請替換為實際IP) "
- "-::1"
```

```
省略非配置項
```

SENTINEL:

```
定義要監控的主節點，格式為 <master-name> <ip> <amdc-port>
<quorum> (如 mymaster 127.0.0.1 6359 2 表示至少 2 個
Sentinel 認為主節點下線才判定為客觀下線)
```

```
sentinel monitor: mymaster 10.0.1.46 (示例IP, 請替換為實際
IP) 6359 2
```

- 10.0.1.32 (示例IP, 請替換為實際IP) 節點哨兵修改bind為當前IP, 以及主節點的IP

NETWORK:

```
監聽的網絡地址，默認僅監聽本地回環地址，註釋此行可監聽所有接口
(暴露在公網有安全風險)
```

```
bind:
```

```
- "10.0.1.32 (示例IP, 請替換為實際IP) "
- "-::1"
```

```
省略非配置項
```

SENTINEL:

```
定義要監控的主節點，格式為 <master-name> <ip> <amdc-port>
```

```
<quorum> (如 mymaster 127.0.0.1 6359 2 表示至少 2 個
Sentinel 認為主節點下線才判定為客觀下線)

sentinel monitor: mymaster 10.0.1.46 (示例IP, 請替換為實際
IP) 6359 2
```

- 8. 在每個節點上啟動緩存和哨兵

- 啟動主節點: `./amdc-server amdc.yaml`
- 啟動從節點: `./amdc-server amdc.yaml`
- 啟動哨兵: `./amdc-sentinel sentinel.yaml`

- 9. 驗證哨兵模式

- 使用 `amdc-cli` 客戶端工具通過指定-p和-h參數, 指定哨兵的IP和端口

```
./amdc-cli -p 26359 -h 10.0.1.46 (示例IP, 請替換為實際IP)
10.0.1.46 (示例IP, 請替換為實際IP) :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=10.0.1.46 (示例IP,
請替換為實際IP) :6359,slaves=2,sentinels=3
```

- 如出現主節點信息、從節點數量和哨兵數量均正確, 即表示部署成功。

#### 4.4.1 哨兵模式運維 FAQ

##### 一. 哨兵集群狀態錯亂、節點信息異常 (主從 / 哨兵互相識別紊亂)

##### 現象

- 哨兵頻繁切換主節點、切換邏輯混亂;
- `sentinel info` 查看發現存在大量已下線、已刪除的舊主庫、從庫、哨兵記錄;
- 新舊節點混雜, 判定主從關係錯誤, 頻繁誤觸發故障轉移;
- 手動下線節點後哨兵仍殘留該節點信息, 無法自動清理。

## 根因

哨兵運行時會自動在 sentinel.yaml 文件底部追加持久化記錄：

- known-replica：記錄關聯主庫的所有從節點
- known-sentinel：記錄同集群其他哨兵節點

known-replica 與實際主從節點關係存在差異，導致主從關係識別錯誤。

## 修復步驟

1. 停止所有哨兵進程

```
amdc-cli -h 哨兵Ip -p 哨兵Port -a 密碼 shutdown
```

```
systemctl stop amdc_26359 (在一些機器上會配置自動重啟，需要用systemctl stop
命令手動停止節點)
```

2. 編輯所有哨兵配置文件 sentinel.yaml

刪除配置文件末尾config rewrite自動生成的全部 known-sentinel、known-replica 等配置行，保留手動編寫的基礎配置（sentinel monitor、認證、故障切換參數等不動）。

### 刪除前：

SENTINEL：

# 定義要監控的主節點，格式為 <master-name> <ip> <amdc-port> <quorum>  
(如 mymaster 127.0.0.1 6359 2 表示至少 2 個 Sentinel 認為主節點下線才判定為客觀下線)

```
sentinel monitor: mymaster 127.0.0.1 6111 2
```

# 主從節點的認證用戶名，格式為 <master-name> <username>

```
sentinel auth-user: mymaster ""
```

# 主從節點的認證密碼，格式為 <master-name> <password>

```
sentinel auth-pass: mymaster ""
```

# 判定節點主觀下線的超時時間（毫秒），默認 30000（30 秒）

```
sentinel down-after-milliseconds: mymaster 30000
```

# Sentinel 之間通信的認證用戶名

```
sentinel sentinel-user: ""
```

# Sentinel 之間通信的認證密碼其他設置

```
sentinel sentinel-pass: ""
```

```

故障轉移時從節點同步新主節點的並發數量，默認 1
sentinel parallel-syncs: mymaster 1
故障轉移超時時間（毫秒），默認 180000（3 分鐘）
sentinel failover-timeout: mymaster 180000
發生警告級事件時執行的通知腳本（如郵件、短信通知），格式為 <master-name>
<script-path>
sentinel notification-script: mymaster ""
故障轉移後通知客戶端重新配置的腳本，格式為 <master-name> <script-path>
sentinel client-reconfig-script: mymaster ""
是否禁止運行時修改腳本配置，默認 yes
sentinel deny-scripts-reconfig: yes
重命名命令，格式為 <master-name> <old-command-name> <new-command-
name>
sentinel rename-command: mymaster "" ""
是否支持主機名（默認 no，需依賴 DNS）
sentinel resolve-hostnames: no
是否對外公佈主機名（默認 no）
sentinel announce-hostnames: no
NAT/Docker 環境下對外公佈的 IP
sentinel announce-ip: ""
NAT/Docker 環境下對外公佈的端口
sentinel announce-port: 0
控制主節點重啟加載數據時，Sentinel 要不要觸發故障轉移，默認 0 主節點只要在
加載數據，Sentinel 絕對不切換主庫
sentinel master-reboot-down-after-period: mymaster 1

```

#### CONFIG REWRITE:

```

sentinel myid: 7354c3cc4fb8faa5fc5de06781f8b497e60d25dc
sentinel config-epoch: mymaster 0
sentinel leader-epoch: mymaster 0
sentinel current-epoch: 0
sentinel known-replica: mymaster 127.0.0.1 6112
sentinel known-sentinel: mymaster 127.0.0.1 26359
a2d89d7c88f0975fb61dcb5df7f78b297e2099ca

```

刪除後：

SENTINEL：

```
定義要監控的主節點，格式為 <master-name> <ip> <amdc-port> <quorum>
(如 mymaster 127.0.0.1 6359 2 表示至少 2 個 Sentinel 認為主節點下線才判定為客觀下線)
sentinel monitor: mymaster 127.0.0.1 6111 2
主從節點的認證用戶名，格式為 <master-name> <username>
sentinel auth-user: mymaster ""
主從節點的認證密碼，格式為 <master-name> <password>
sentinel auth-pass: mymaster ""
判定節點主觀下線的超時時間（毫秒），默認 30000（30 秒）
sentinel down-after-milliseconds: mymaster 30000
Sentinel 之間通信的認證用戶名
sentinel sentinel-user: ""
Sentinel 之間通信的認證密碼其他設置
sentinel sentinel-pass: ""
故障轉移時從節點同步新主節點的並發數量，默認 1
sentinel parallel-syncs: mymaster 1
故障轉移超時時間（毫秒），默認 180000（3 分鐘）
sentinel failover-timeout: mymaster 180000
發生警告級事件時執行的通知腳本（如郵件、短信通知），格式為 <master-name>
<script-path>
sentinel notification-script: mymaster ""
故障轉移後通知客戶端重新配置的腳本，格式為 <master-name> <script-path>
sentinel client-reconfig-script: mymaster ""
是否禁止運行時修改腳本配置，默認 yes
sentinel deny-scripts-reconfig: yes
重命名命令，格式為 <master-name> <old-command-name> <new-command-name>
sentinel rename-command: mymaster "" ""
是否支持主機名（默認 no，需依賴 DNS）
sentinel resolve-hostnames: no
是否對外公佈主機名（默認 no）
sentinel announce-hostnames: no
```

## 檢查哨兵節點 sentinel monitor 配置項

SENTINEL:

- # 定義要監控的主節點，格式為 <master-name> <ip> <amdc-port> <quorum>  
(如 mymaster 127.0.0.1 6359 2 表示至少 2 個 Sentinel 認為主節點下線才判定為客觀下線)

```
sentinel monitor: mymaster 127.0.0.1 6111 2
```

```
檢查主從節點信息
```

```
127.0.0.1:6111> info replication
Replication
role:master
connected_slaves:1
slave0:ip=127.0.0.1,port=6112,state=online,offset=102202,lag=1,worker-
thread=0
master_failover_state:no-failover
master_replid:b4a214a5cecfbd69098c2bcdcb8237a58770095ef
master_replid2:000000000000000000000000000000000000000000000000
master_repl_offset:102202
second_repl_offset:-1
repl_backlog_active:1
repl_backlog_size:1048576
repl_backlog_first_byte_offset:1
```

```
repl_backlog_histlen:102202
repl_mptcp_enabled:0
```

#### 4. 逐個重啟哨兵節點

```
amdc-sentinel sentinel.yaml
```

```
systemctl start amdc_26359 (在一些機器上會配置自動重啟，需要用systemctl
start 命令手動啟動節點)
```

#### 5. 校驗哨兵集群拓撲

```
連接任意哨兵
amdc-cli -h 哨兵Ip -p 哨兵Port -a 密碼
SENTINEL MASTERS
SENTINEL REPLICAS mymaster
SENTINEL SENTINELS mymaster
```

確認主從/哨兵節點關係是否正常。

## 二. 哨兵誤觸發故障轉移

### 現象

主節點網絡短暫抖動、負載高響應延遲，哨兵判定主庫下線，自動切主。

### 排查 & 解決

- 調整故障轉移投票等待時長，避免瞬時抖動觸發切換 `sentinel down-after-milliseconds mymaster 50000`;
- 檢查主庫 CPU、磁盤、網絡帶寬，優化主節點性能;
- 多哨兵分散部署在不同機房 / 服務器，避免單點網絡故障全員誤判。

## 三. 哨兵之間無法互相發現、哨兵投票不生效

### 現象

多哨兵部署，但 `SENTINEL SENTINELS` 只能看到自身，無法完成故障轉移投票。

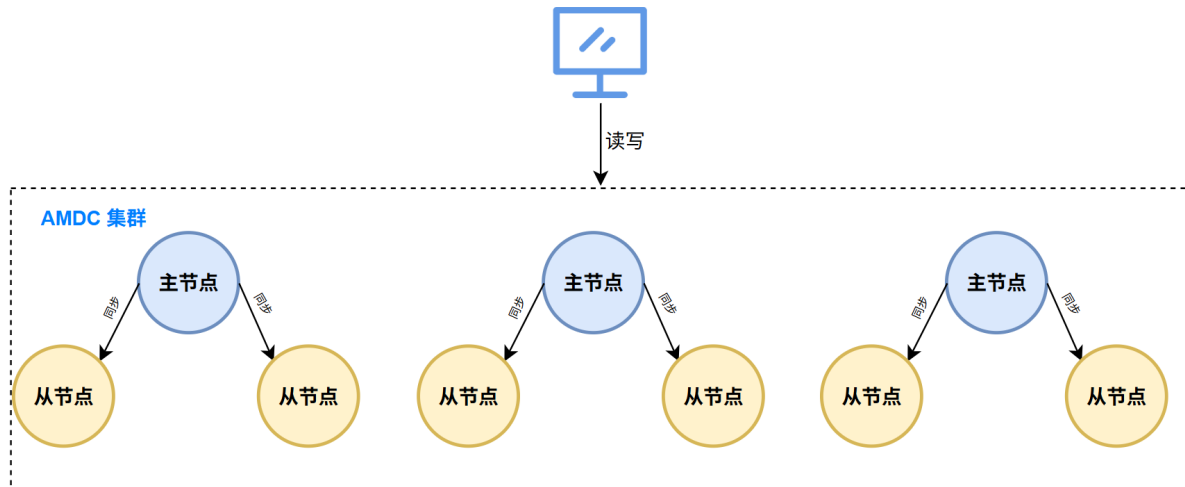
### 排查點

- 防火牆放行 26379 哨兵端口，節點間互通;
- 配置文件 `sentinel monitor` 集群名稱完全一致;

- 清理配置文件底部冗餘 known-sentinel 記錄後重啟哨兵；
- 檢查密碼配置，主從、哨兵認證密碼統一。

## 4.5 集群模式

集群部署圖



集群節點規劃

服務器IP	角色	端口
10.0.1.46 (示例IP, 請替換為實際IP)	集群節點1+從節點1	6359, 6360
10.0.1.66 (示例IP, 請替換為實際IP)	集群節點2+從節點2	6359, 6360
10.0.1.32 (示例IP, 請替換為實際IP)	集群節點3+從節點3	6359, 6360

### 4.5.1 集群安裝步驟

- 1.準備服務器節點
- 2.上傳AMDC緩存核心安裝包( `AMDC-V2.0.6-構建日期-[compat|perf]-CPU架構.tar.gz` )至上述目標服務器的安裝目錄下(如: `/opt`目錄下)
- 3.解壓安裝包: `tar -zxvf AMDC-V2.0.6-構建日期-[compat|perf]-CPU架構.tar.gz`
- 4.複製一份安裝包為 slave 節點: `cp -r amdc amdc-slave`
- 5.上傳授權文件 `license.lic` 至每個節點安裝目錄 `/opt/amdc/` 下, (包括slave)。授權可參考本文檔的 [AMDC 授權管理指南](#)

```
[root@private opt]# tree -L 2
.
├── amdc
│ ├── acls.properties
│ ├── amdc-benchmark
│ ├── amdc-check-aof
│ ├── amdc-check-rdb
│ ├── amdc-cli
│ ├── amdc-conf-conv
│ ├── amdc-sentinel
│ ├── amdc-server
│ ├── amdc.yaml
│ ├── license.lic #授權文件
│ └── sentinel.yaml
└── amdc-slave # 從節點目錄
 ├── acls.properties
 ├── amdc-benchmark
 ├── amdc-check-aof
 ├── amdc-check-rdb
 ├── amdc-cli
 ├── amdc-conf-conv
 ├── amdc-sentinel
 ├── amdc-server
 ├── amdc.yaml
 ├── license.lic #授權文件
 └── sentinel.yaml
```

- 6. 集群配置

- 集群節點1 (10.0.1.46 (示例IP, 請替換為實際IP)) : 修改 `amdc.yaml` 中的bind為當前機器IP和開啟集群模式, 並開啟後台運行模式

**NETWORK:**

# 監聽的網絡地址，默認僅監聽本地回環地址，註釋此行可監聽所有接口  
(暴露在公網有安全風險)

**bind:**

```
- "10.0.1.46 (示例IP, 請替換為實際IP) " # 添加本機IP
- "-::1"
```

**GENERAL:**

# 是否以守護進程模式運行

```
daemonize: yes
```

# 省略非配置項

**CLUSTER:**

# 是否啟用集群模式

```
cluster-enabled: yes
```

- 集群節點2 (10.0.1.66 (示例IP, 請替換為實際IP)) : 修改 `amdc.yaml` 中的bind為當前機器IP和開啟集群模式，並開啟後台運行模式

**NETWORK:**

# 監聽的網絡地址，默認僅監聽本地回環地址，註釋此行可監聽所有接口  
(暴露在公網有安全風險)

**bind:**

```
- "10.0.1.66 (示例IP, 請替換為實際IP) " # 添加本機IP
- "-::1"
```

# 省略非配置項

**CLUSTER:**

# 是否啟用集群模式

```
cluster-enabled: yes
```

- 集群節點3 (10.0.1.32 (示例IP, 請替換為實際IP)) : 修改 `amdc.yaml` 中的bind為當前機器IP和開啟集群模式，並開啟後台運行模式

```

NETWORK:

監聽的網絡地址，默認僅監聽本地回環地址，註釋此行可監聽所有接口
(暴露在公網有安全風險)

bind:
 - "10.0.1.32 (示例IP, 請替換為實際IP) " # 添加本機IP
 - "-::1"

省略非配置項

CLUSTER:

是否啟用集群模式

cluster-enabled: yes

```

- 7.從節點配置

- 由於主節點和從節點在同一台機器上，這裡修改監聽端口為： 6360
- 下面操作進入 `amdc-slave` 目錄下操作
- 從節點1 (10.0.1.46 (示例IP, 請替換為實際IP) )：修改 `amdc.yaml` 中的bind為當前機器IP、Port和開啟集群模式，並開啟後台運行模式

```

NETWORK:

監聽的網絡地址，默認僅監聽本地回環地址，註釋此行可監聽所有接口
(暴露在公網有安全風險)

bind:
 - "10.0.1.46 (示例IP, 請替換為實際IP) " # 添加本機IP
 - "-::1"

port: 6360

GENERAL:

是否以守護進程模式運行

daemonize: yes

pid 文件路徑

pidfile: "/var/run/amdc-server-slave.pid"

省略非配置項

CLUSTER:

```

```
是否啟用集群模式
cluster-enabled: yes

集群配置文件路徑，由節點自動維護
cluster-config-file: "node-6360.conf"
```

- 從節點2 (10.0.1.66 (示例IP, 請替換為實際IP)) : 修改 `amdc.yaml` 中的bind為當前機器IP、Port 和開啟集群模式，並開啟後台運行模式

```
NETWORK:

監聽的網絡地址，默認僅監聽本地回環地址，註釋此行可監聽所有接口
(暴露在公網有安全風險)

bind:
 - "10.0.1.66 (示例IP, 請替換為實際IP) " # 添加本機IP
 - "-::1"

port: 6360

GENERAL:

是否以守護進程模式運行
daemonize: yes

pid 文件路徑
pidfile: "/var/run/amdc-server-slave.pid"

省略非配置項

CLUSTER:

是否啟用集群模式
cluster-enabled: yes

集群配置文件路徑，由節點自動維護
cluster-config-file: "node-6360.conf"
```

- 從節點3 (10.0.1.32 (示例IP, 請替換為實際IP)) : 修改 `amdc.yaml` 中的bind為當前機器IP、Port 和開啟集群模式，並開啟後台運行模式

```
NETWORK:

監聽的網絡地址，默認僅監聽本地回環地址，註釋此行可監聽所有接口
(暴露在公網有安全風險)
```

```

bind:
 - "10.0.1.32 (示例IP, 請替換為實際IP) " # 添加本機IP
 - "-::1"

port: 6360

GENERAL:
 # 是否以守護進程模式運行
 daemonize: yes
 # pid 文件路徑
 pidfile: "/var/run/amdc-server-slave.pid"
省略非配置項

CLUSTER:
 # 是否啟用集群模式
 cluster-enabled: yes
 # 集群配置文件路徑, 由節點自動維護
 cluster-config-file: "node-6360.conf"

```

- 8.在每個節點上啟動緩存和從節點

- 啟動主節點: `./amdc-server amdc.yaml`
- 啟動從節點 (切換至amdc-slave) : `./amdc-server amdc.yaml`

- 9.使用 `amdc-cli` 客戶端工具創建集群

- `amdc-cli` : `./amdc-cli --cluster create 10.0.1.46 (示例IP, 請替換為實際IP) :6359 10.0.1.66 (示例IP, 請替換為實際IP) :6359 10.0.1.32 (示例IP, 請替換為實際IP) :6359 10.0.1.46 (示例IP, 請替換為實際IP) :6360 10.0.1.66 (示例IP, 請替換為實際IP) :6360 10.0.1.32 (示例IP, 請替換為實際IP) :6360 --cluster-replicas 1`

註: `amdc-cli --cluster` 命令使用手冊可參考用戶手冊。

- 創建過程中, 要輸入 [yes] , 確認集群節點安排, 如下:

```

[root@private amdc]# ./amdc-cli --cluster create 10.0.1.46 (示例IP, 請替換為實際IP) :6359 10.0.1.66 (示例IP, 請替換為實際IP) :6359 10.0.1.32 (示例IP, 請替換為實際IP) :6359 10.0.1.46 (示例IP, 請替換為實際IP) :6360 10.0.1.66 (示例IP, 請替換為實際IP) :6360 10.0.1.32 (示例

```

```

IP, 請替換為實際IP) :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 10.0.1.66 (示例IP) :6360 to 10.0.1.46 (示例
IP) :6359
Adding replica 10.0.1.32 (示例IP) :6360 to 10.0.1.66 (示例
IP) :6359
Adding replica 10.0.1.46 (示例IP) :6360 to 10.0.1.32 (示例
IP) :6359
M: a11b0a053f596321a4e310046e2b69bca8c96d4a 10.0.1.46 (示例
IP) :6359
 slots:[0-5460] (5461 slots) master
M: ab6dcd5d434ce3ec8ad199ac1a3d610ac33f3a7f 10.0.1.66 (示例
IP) :6359
 slots:[5461-10922] (5462 slots) master
M: 70451b10ef6f7793083b47a994c900cccc218b39 10.0.1.32 (示例
IP) :6359
 slots:[10923-16383] (5461 slots) master
S: 37ffc8538d24b332d45d6732b1df7e9386114522 10.0.1.46 (示例
IP) :6360
 replicates 70451b10ef6f7793083b47a994c900cccc218b39
S: 3ffc63268838f02dd4e1d66f11b5a6ffdc83cdfe 10.0.1.66 (示例
IP) :6360
 replicates a11b0a053f596321a4e310046e2b69bca8c96d4a
S: 4862dfb143cf5de0bebb64370f77c8bbe5bfc917 10.0.1.32 (示例
IP) :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 10.0.1.46 (示例
IP) :6359)
M: a11b0a053f596321a4e310046e2b69bca8c96d4a 10.0.1.46 (示例
IP) :6359
 slots:[0-5460] (5461 slots) master
 1 additional replica(s)
S: 4862dfb143cf5de0bebb64370f77c8bbe5bfc917 10.0.1.32 (示例
IP) :6360
 slots: (0 slots) slave
 replicates ab6dcd5d434ce3ec8ad199ac1a3d610ac33f3a7f
S: 37ffc8538d24b332d45d6732b1df7e9386114522 10.0.1.46 (示例
IP) :6360
 slots: (0 slots) slave
 replicates 70451b10ef6f7793083b47a994c900cccc218b39
S: 3ffc63268838f02dd4e1d66f11b5a6ffdc83cdfe 10.0.1.66 (示例
IP) :6360
 slots: (0 slots) slave
 replicates a11b0a053f596321a4e310046e2b69bca8c96d4a
M: 70451b10ef6f7793083b47a994c900cccc218b39 10.0.1.32 (示例
IP) :6359
 slots:[10923-16383] (5461 slots) master
 1 additional replica(s)
M: ab6dcd5d434ce3ec8ad199ac1a3d610ac33f3a7f 10.0.1.66 (示例
IP) :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
10.0.1.46 (示例IP) :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壓縮安裝包名稱

- x86\_64: `AMDC-V2.0.6-Docker-20260921-compatible-amd64-kylin.tar.gz`
- arm64: `AMDC-V2.0.6-Docker-20260921-compatible-arm64-kylin.tar.gz`

註：安裝包日期可能不同，請根據實際情況選擇。

## 2. 安裝包目錄結構

```
tree -L 1 AMDC-V2.0.6-Docker-20260921-compatible-amd64-kylin
AMDC-V2.0.6-Docker-20260921-compatible-amd64-kylin # docker安
裝包解壓後目錄
|-- AMDC-V2.0.6-Docker-20260921-compatible-amd64-kylin.tar.gz # docker鏡
像文件
|-- amdc
| |-- amdc.conf # AMDC配置文件
| |-- license.xml # 授權文件
|-- amdc-dockercompose.yaml # docker compose配置文件
```

### 4.6.2 安裝步驟

下面的操作以AMDC-V2.0.6-Docker-20260921-compatible-amd64-kylin.tar.gz為例說明。

#### 1. 解壓和導入鏡像

- 解壓 `tar -zxvf AMDC-V2.0.6-Docker-20260921-compatible-amd64-kylin.tar.gz`
- 進入解壓後的文件夾 `cd AMDC-V2.0.6-Docker-20260921-compatible-amd64-kylin`
- 加載鏡像: `docker load < AMDC-V2.0.6-Docker-20260921-compatible-amd64-kylin.tar.gz`

#### 2. 修改授權文件

AMDC授權文件需要掛載到容器內部安裝目錄，將不同授權文件放入當前amdc目錄下，啟動容器前，請確認授權文件有效，按需檢查或替換授權文件。

##### 授權文件掛載說明

- `license.xml` 舊授權掛載目標路徑: `/usr/local/bin/license.xml`
- `license.lic` KBC授權掛載目標路徑: `/usr/local/bin/license.lic`
- `acls.properties` 統一授權配置文件掛載目標路徑: `/usr/local/bin/acls.properties`

#### 3. 鏡像啟動與停止

- 啟動命令: `docker-compose -f amdc-dockercompose.yaml up -d`

- 停止命令: `docker-compose -f amdc-dockercompose.yaml down`

#### 4. amdc-dockercompose.yaml 配置說明

```
version: '3'
services:
 amdc:
 image: docker_image_tag
 ports:
 - 6359:6359
 volumes:
 - ./amdc:/opt/amdc:z # 將宿主機 ./amdc 目錄掛載至容器 /opt/amdc
 - ./amdc/license.xml:/usr/local/bin/license.xml # 將授權文件掛載
到目標位置, lic授權文件也同樣掛載
 working_dir: /opt/amdc # 將容器啟動默認工作目錄配置為掛載路徑
/opt/amdc, 該路徑與業務配置文件 amdc.yaml 中 dir 參數填寫路徑統一, 保證程序讀
取目錄一致。
 command: /usr/local/bin/amdc-server # amdc-cli / amdc-server /
amdc-sentinel 等其他可執行文件都在docker內部 /usr/local/bin/ 目錄下
```

## 4.7 RPM包安裝

RPM (Red Hat Package Manager) 安裝包是一種在 Linux 系統中用於軟件包管理的格式, 最初由 Red Hat 開發, 現在廣泛用於基於 Red Hat 的發行版, AMDC提供arm和x86的RPM安裝包, 可通過 `rpm` 命令安裝。其支持的操作系統有:

- Kylin
- OpenEuler
- CentOS

### 4.7.1 安裝包安裝

- x86\_64: `AMDC-V2.0.6-20260921-compat-1.el7.x86_64.rpm`
- arm64: `AMDC-V2.0.6-20260921-compat-1.el7.aarch64.rpm`

#### 1. 安裝命令:

```
rpm -ivh AMDC-V2.0.6-20260921-compat-1.el7.x86_64.rpm
Preparing...
```

```
[100%]
Updating / installing...
 1:amdc-2.0.6-1.el7
[100%]
```

2. 安裝完後，產品介質安裝目錄在 `/opt/amdc-2.0.6` 目錄下
3. 將授權文件導入到 `/opt/amdc-2.0.6/license.xml` 文件中
4. 啟動命令：`/opt/amdc-2.0.6/amdc-server`

## 5 停止緩存核心服務

AMDC 緩存核心提供多種停止方式，為確保數據安全和系統穩定性，建議按照以下優先級順序選擇合適的停止方法：

### 5.1 停止方式

1. 通過客戶端發送 SHUTDOWN 命令

這是最安全的停止方式，AMDC 會在完成數據持久化後再退出，確保數據完整性。

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

2. 交互式客戶端停止

當已連接到 AMDC 客戶端時，可直接執行 shutdown 命令：

```
127.0.0.1:6359> shutdown
```

執行流程：

1. 斷開所有客戶端連接
2. 根據配置執行 RDB/AOF 持久化操作
3. 安全退出服務進程

### 5.2 緊急停止方式

當 AMDC 服務無響應或無法正常停止時，可使用此方式。**注意：此操作可能導致數據丟失！**

操作步驟：

1. 查找 AMDC 進程 PID：

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

2. 強制終止進程：

```
kill -9 [进程ID]
```

### 5.3 停止方式對比

停止方式	數據安全性	適用場景	推薦度
客戶端 SHUTDOWN 命令	高	正常維護、計劃停機	推薦
交互式 shutdown	高	調試、測試環境	推薦
強制 kill -9	低	服務無響應、緊急故障處理	不推薦

**重要提示：**生產環境中請優先使用優雅停機方式，避免因強制終止導致的數據丟失風險。只有在 AMDC 服務完全無響應且無法通過正常方式停止時，才考慮使用強制終止方式。

## 6 License（許可）授權管理

### 6.1 授權模式概述

金蝶Apusic分佈式緩存（AMDC）V2.0.6 支持四種License許可授權驗證模式，用戶可根據實際需求選擇合適的授權方式：

授權模式	適用場景	配置文件	特點
金蝶天燕本地授權	獨立部署、離線環境	license.xml	本地文件授權，支持綁定IP、MAC地址
金蝶KBC授權	標準生產環境	license.lic	本地授權文件，通過授權碼方式驗證
金蝶統一授權中心	多租戶、雲環境	acls.properties	集中式授權管理，支持多租戶
金蝶統一授權中心環境變量	多租戶、雲環境	acls.properties env	集中式授權管理，支持多租戶

**默認授權模式：** 金蝶天燕本地授權和金蝶KBC授權驗證

**注意：**AMDC V2.0.6 版本優化授權文件加載邏輯，授權文件僅需放置在 amdc-server 同級目錄，無需再通過 amdc.yaml 配置授權路徑，簡化部署配置流程。

### 6.2 授權配置詳解

#### 6.2.1 金蝶天燕本地授權

**適用場景：** 離線環境、獨立部署、測試環境

**配置步驟：**

1. 獲取 `license.xml` 授權文件
2. 將授權文件放置在AMDC安裝目錄下

#### 6.2.2 金蝶KBC授權

**適用場景：** 標準生產環境、需要設備綁定的場景、離線環境

**配置步驟：**

1. 獲取設備特徵碼

執行AMDC服務啟動命令，系統將自動輸出設備特徵碼：

```
./amdc-server
```

輸出示例：

```
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 # o000o000o000o AMDC is
starting o000o000o000o
400987:C 05 Feb 2026 17:33:59.232 * AMDC KBC authorization
code: 'SZTY-1486655719' for mac: 00-00-00-00-00-00 (示例MAC地址)
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!
```

特徵碼說明：

- 格式： `SZTY-XXXXXXXXXX` （以SZTY開頭）
- 綁定信息：MAC地址或其他硬件特徵
- 注意：每個設備的特徵碼具有唯一性，不可跨設備使用

## 2. 申請授權文件

1. 訪問金蝶KBC授權系統（此操作由金蝶天燕銷售、實施或研發人員操作）
2. 使用獲取的特徵碼申請授權
3. 下載生成的 `license.lic` 文件

## 3. 配置授權文件

1. 將 `license.lic` 文件放置在AMDC安裝目錄下

### 6.2.3 金蝶統一授權中心

適用場景：多租戶環境、雲平台部署、集中式授權管理

配置步驟：

1. 創建 `acls.properties` 配置文件

2. 配置授權中心連接信息：

```
是否開啟統一授權中心
apusic_acls_enable=true

授權中心地址（多個地址用逗號分隔）
apusic_acls_authUrls=10.0.1.100:8080,10.0.1.101:8080（示例IP，請
替換為實際IP）

命名空間
apusic_acls_ns=public

租戶名稱（如不確定可填寫為public）
apusic_acls_tenant=public
```

3. 將 `acls.properties` 文件放置在AMDC安裝目錄下

## 6.3 授權文件管理

### 6.3.0.1 授權文件位置

所有授權文件均需放置在AMDC安裝目錄下，與 `amdc-server` 可執行文件同級目錄。

### 6.3.0.2 授權熱更新

AMDC支持授權文件熱更新，無需重啟服務：

- 直接替換現有的授權文件內容
- 系統會自動檢測並加載新的授權信息
- **注意：**確保新授權文件格式正確，避免授權失效

### 6.3.0.3 授權驗證優先級

當多種授權文件同時存在時，AMDC按照以下優先級進行驗證：

1. `acls.properties env`（統一授權中心環境變量）
2. `acls.properties`（統一授權中心）
3. `license.lic`（KBC授權）

#### 4. `license.xml` （本地授權）

## 6.4 故障排查

### 6.4.1 常見錯誤信息

- `**"No valid license file found"**`：未找到有效的授權文件
- `**"Local license verification failed"**`：授權文件無效或已過期
- `**"Authorization code mismatch"**`：特徵碼與設備不匹配

### 6.4.2 處理方案

1. 確認授權文件放置在正確目錄
2. 驗證授權文件格式和內容是否正確
3. 檢查設備特徵碼是否與授權文件匹配
4. 確認授權文件是否在有效期內

## 7 緩存核心服務相關參數說明

### 7.1 網絡配置參數

#### 7.1.1 監聽地址配置

參數名稱	默認值	說明	使用方式
bind	127.0.0.1 -::1	監聽的網絡地址，指定客戶端訪問緩存的IP地址，除綁定地址外無法連接	可綁定多個地址，如：bind： - "127.0.0.1" - "192.168.x.x" (示例IP，請替換為實際IP)

#### 7.1.2 端口配置

參數名稱	默認值	說明	使用方式
port	6359	監聽的TCP端口，設為0則不監聽TCP套接字	port: 6359

#### 7.1.3 保護模式

參數名稱	默認值	說明
protected-mode	yes	保護模式，當未指定bind且無密碼時，僅允許本地回環地址和Unix域套接字連接

## 7.2 後台運行配置

#### 7.2.1 守護進程模式

參數名稱	默認值	說明	使用方式
daemonize	no	是否以守護進程模式運行，yes則後台運行並生成pid文件	daemonize: yes, 註：logfile參數需要填寫，以便查看日誌

#### 7.2.2 PID文件配置

參數名稱	默認值	說明	使用方式
------	-----	----	------

pidfile	/var/run/amdc-sentinel.pid	pid文件路徑，守護進程模式下默認生成	pidfile: "/var/run/amdc-server.pid"
---------	----------------------------	---------------------	-------------------------------------

## 7.3 安全認證配置

### 7.3.1 密碼認證

參數名稱	說明	使用方式
requirepass	設置密碼，可以為空。從節點密碼建議與主節點密碼保持一致	requirepass: "123456" 註：在users.acl存在的情況下，server優先使用users.acl中的密碼
masterauth	主節點的密碼，若主節點無密碼則不需要填，若有密碼，不分主從所有節點都需要填寫該參數	masterauth: "123456"

## 7.4 集群配置

### 7.4.1 集群模式

參數名稱	默認值	說明	使用方式
cluster-enabled	-	是否開啟集群模式	cluster-enabled: "yes"

### 7.4.2 集群配置文件

參數名稱	默認值	說明
cluster-config-file	-	集群配置文件路徑，由節點自動維護

### 7.4.3 集群端口

參數名稱	默認值	說明	使用方式
port	6359	監聽的TCP端口，設為0則不監聽TCP套接字	port: 6359
集群總線	16359	集群節點間Gossip通信端口（=緩存端口+10000）	

## 7.5 哨兵配置

### 7.5.1 哨兵端口

參數名稱	默認值	說明
------	-----	----

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: "10.0.1.46 (示例IP, 请替换为实际IP) 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、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自動重寫的觸發百分比	auto-aof-rewrite-percentage: 100
auto-aof-rewrite-min-size	64mb	AOF自動重寫的最小文件大小	auto-aof-rewrite-min-size: 64mb
aof-load-truncated	yes	啟動時是否加載被截斷的AOF文件	aof-load-truncated: yes
aof-use-rdb-preamble	yes	AOF重寫時是否使用RDB前綴	aof-use-rdb-preamble: yes

## 7.8 重要配置示例

### 7.8.1 單機模式配置示例

```

NETWORK:
 bind:
 - "0.0.0.0" # 允許所有IP訪問
 port: 6359
 protected-mode: yes

GENERAL:
 daemonize: yes # 後台運行
 pidfile: "/var/run/amdc-server.pid"

SECURITY:
 requirepass: "your_password" # 設置訪問密碼

```

### 7.8.2 主從模式配置示例

```

主節點配置
NETWORK:
 bind:
 - "10.0.1.46 (示例IP, 請替換為實際IP)" # 主節點IP
 port: 6359

```

```

GENERAL:
 daemonize: yes

SECURITY:
 requirepass: "master_password" # 主節點密碼

從節點配置
NETWORK:
 bind:
 - "10.0.1.66 (示例IP, 請替換為實際IP) " # 從節點IP
 port: 6359

GENERAL:
 daemonize: yes

REPLICATION:
 replicaof: "10.0.1.46 (示例IP, 請替換為實際IP) 6359" # 指定主節點

SECURITY:
 requirepass: "master_password" # 從節點密碼需與主節點一致
 masterauth: "master_password" # 主節點密碼

```

### 7.8.3 哨兵模式配置示例

```

NETWORK:
 bind:
 - "10.0.1.46 (示例IP, 請替換為實際IP) " # 哨兵節點IP
 port: 26359

GENERAL:
 daemonize: yes

SENTINEL:
 sentinel monitor: mymaster 10.0.1.46 (示例IP, 請替換為實際IP) 6359 2

```

```
監控主節點
sentinel auth-pass: mymaster master_password # 主節點密碼
```

#### 7.8.4 集群模式配置示例

```
NETWORK:
 bind:
 - "10.0.1.46 (示例IP, 請替換為實際IP) " # 集群節點IP
 port: 6359

GENERAL:
 daemonize: yes

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

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

## 8 Web管控台安裝

### 8.1 安裝包介紹

- 安裝包：`AMDC-Console-【版本】-【日期】-【硬件架構】.tar.gz`

### 8.2 安裝包目錄結構

解壓管控台安裝包後，目錄結構如下（以Linux x86\_64為例）：

```
amdc_console/
├── agent/ # agent 命令行客戶端（用於無圖形界面的自動化運維）
├── amdc-console # 管控台主程序
├── config.yaml # 管控台核心配置文件
├── console.db # SQLite 數據庫文件（首次啟動自動生成）
├── resource/ # 靜態資源目錄（RSA 密鑰、AMDC 模板配置等）
├── upload_files/ # 用戶上傳文件（License、安裝包等）
├── log/ # 運行日誌目錄（首次啟動自動生成）
└── web/ # 前端靜態資源
```

### 8.3 安裝步驟

安裝步驟如下：

1. 上傳AMDC控制台安裝包至目標服務器的安裝目錄下（如：/opt目錄下）
2. 解壓安裝包：`tar -zxvf AMDC-Console-【版本】-【日期】-【硬件架構】.tar.gz`
3. 進入解壓後的文件夾：`cd amdc_console`
4. 按需修改 `config.yaml`（詳見後文「管控台核心配置說明」）
5. 執行啟動命令（後台啟動）：`nohup ./amdc-console > nohup.out 2>&1 &`
6. 啟動完成後即可通過瀏覽器訪問AMDC控制台：`http://ip:port`（其中 ip 表示部署AMDC控制台服務器的IP，port 為控制台端口，默認 9001），如：`http://192.168.x.x:9001`（示例IP，請替換為實際IP）

#### 8.3.1 驗證安裝

打開瀏覽器，訪問 `http://ip:port`（如：`http://192.168.x.x:9001`），若能正常加載AMDC管控台登錄頁面，即表示Web管控台安裝成功。

首次啟動後，管控台會在 `./resource/.initial_passwords` 文件中寫入所有預置管理員賬號的初始密碼（含 `amdc_admin`、三員管理員、租戶管理員），請妥善保存該文件並在首次登錄後立即修改密碼。

### 8.3.2 停止Web管控台

步驟：

1. 通過查看端口進程命令找到控制台進程PID：`netstat -nltp | grep 9001` 或者 `netstat -nltp | grep amdc-console`
2. 優雅停止：`kill -15 <PID>`（推薦，等待數據落盤）
3. 緊急強制停止：`kill -9 <PID>`

## 8.4 管控台核心配置說明

管控台所有配置集中在安裝目錄下的 `config.yaml` 文件中，首次啟動前可根據實際部署環境修改。

配置節點	參數名稱	默認值	說明
system（服務監聽）	<code>system.addr</code>	9001	管控台 HTTP 服務監聽端口，修改此參數可改變管控台訪問端口；如 <code>addr: 8090</code>
	<code>system.pprof-port</code>	0	pprof 性能診斷端口，0 = 關閉；排查問題時臨時改為 9011，僅監聽 127.0.0.1，排查完務必改回 0
	<code>system.default-language</code>	zh-CN	管控台默認界面語言，可選： zh-CN / zh-TW / en
	<code>system.use-multipoint</code>	false	是否允許多點登錄（同賬號多終端）
jwt（鑒權）	<code>jwt.signing-key</code>	見配置文件	JWT 簽名密鑰，生產環境必須通過 <code>AMDC_JWT_SIGNING_KEY</code> 環境變量覆蓋，禁止使用默認值
	<code>jwt.expires-time</code>	86400	Token 有效期（秒），默認 24 小時
sqlite（數據庫）	<code>sqlite.path</code>	<code>./console.db</code>	SQLite 數據庫文件路徑；生產環境建議配置到獨立數據盤

zap (日誌)	zap.level	info	日誌級別: debug / info / warn / error
	zap.director	log	日誌文件輸出目錄 (相對路徑)
local / oss (存儲)	local.path	uploads/file	業務文件上傳根目錄
	oss-type	local	對象存儲類型: local (本地) / qiniu (七牛) 等
tls (HTTPS)	tls.isEnabled	false	是否啟用 HTTPS; 啟用後 HTTP 端口將轉為 HTTPS, 需配合 <code>tls.key-path</code> / <code>tls.cert-path</code> 配置證書
amdc-conf (緩存對接)	amdc-conf.conf-path	./resource/amdc_conf/conf.yaml	AMDC 緩存核心默認配置模板, 納管新集群時使用的 <code>amdc.yaml</code> 模板
	amdc-conf.resource-catalogue	./resource	AMDC 資源目錄 (部署包、License、配置文件等的統一管理目錄)
	amdc-conf.upload-files	./upload_files/	用戶通過管控台上傳的安裝包、License、補丁等文件存儲目錄
	amdc-conf.amdc-shake-path	./resource/amdc-shake/amdc-shake	amdc-shake 工具路徑 (升級、遷移使用)
keycloak (SSO)	keycloak.isEnabled	false	是否啟用 Keycloak SSO 單點登錄; 啟用後需配置 <code>configURL</code> / <code>clientId</code> / <code>clientSecret</code> / <code>redirectURL</code> 等
cli (agent 認證)	cli.ai_token	ApusicMDC	agent 命令行客戶端的 AI Token ( <code>X-AMDC-CLI-Token</code> ), 留空禁用; 建議生產環境使用足夠隨機的強密鑰
prometheus (監控)	-	-	已遷移至前端「監控中心 / Prometheus 配置」頁面維護, 管控台啟動時從數據庫讀取, 無需在此配置

修改監聽端口示例:

```
system:
 addr: 8090 # 修改為期望的端口
```

修改後重啟管控台：先 `kill -15 <PID>`，再 `nohup ./amdc-console > nohup.out 2>&1 &`，並同步放行防火牆端口。

## 8.5 管控台常用運維操作

### 8.5.1 查看運行日誌

```
tail -f log/amdc-console.log
```

### 8.5.2 重置預置管理員密碼

若 `.initial_passwords` 文件丟失或預置賬號被鎖定，可在管控台可執行文件所在目錄執行：

```
./amdc-console --reinit amdc_admin
```

重置後會在終端輸出新的 12 位隨機密碼（僅作用於 5 個預置賬號）。

### 8.5.3 備份與遷移

數據類型	路徑	說明
SQLite 數據庫	console.db	賬號、集群納管信息、監控配置、審計日誌等
上傳文件	upload_files/	安裝包、License、補丁等
靜態資源	resource/（不含 casbin_rule.csv 與 amdc-shake 等自動生成文件）	RBAC 模型、密鑰對、AMDC 模板配置
日誌	log/	視保留策略定期清理

遷移到新服務器時，將上述目錄整體打包恢復即可。

全國統一服務熱線  
4008-555-800

金蝶天燕雲計算股份有限公司（簡稱“金蝶天燕”）成立於2000年，前身為“金蝶中間件公司”，是金蝶集團旗下基礎軟件平台提供商，雲計算國家標準制定企業，國家信創產業核心軟件企業。金蝶天燕是國家863重點研發計劃與核高基重大專項承接企業，也是“兩網一站四庫十二金”國家重點工程的基礎平台提供商，產品廣泛應用於政府、軍工、金融、能源等關鍵行業，累計服務客戶總數超過10萬家。



雲計算國家標準制定企業  
金蝶集團旗下基礎軟件企業  
資訊技術應用創新核心企業  
官網：[www.apusic.com](http://www.apusic.com)

