ООО «Астра Консалтинг»

Астра ИС МД (Инфраструктурные Сервисы)

Модуль Астра ИС МД

Модуль Astra Redis

ДОКУМЕНТАЦИЯ, СОДЕРЖАЩАЯ ИНФОРМАЦИЮ, НЕОБХОДИМУЮ ДЛЯ ЭКСПЛУАТАЦИИ ЭКЗЕМПЛЯРА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ, ПРЕДОСТАВЛЕННОГО ДЛЯ ПРОВЕДЕНИЯ ЭКСПЕРТНОЙ ПРОВЕРКИ

Москва, 2026

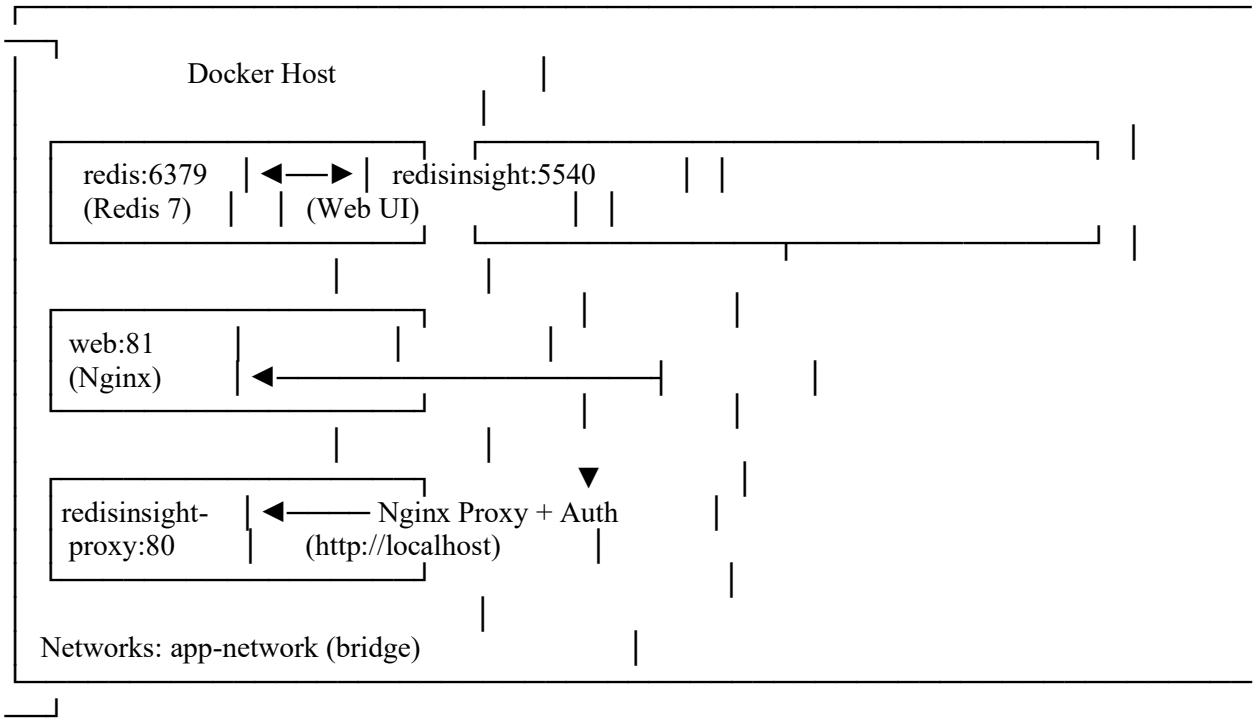
СОДЕРЖАНИЕ

1. Область применения

Настоящий документ содержит описание функциональных характеристик программного обеспечения Астра ИС МД и модуля Astra Redis, развернутого через docker-compose.

Документ предназначен для системных администраторов и разработчиков, осуществляющих эксплуатацию и обслуживание системы управления базами данных Astra Redis.

1.1. Архитектура системы



1.2. Характеристики системы

Таблица 1 — Характеристики системы Astra Redis

Параметр	Значение
Redis версия	7 (alpine)
Порт	6379
Аутентификация	Включена
Пароль	mysecretpassword
Персистентность	AOF (Append Only File)
Сетевой драйвер	bridge

2. Подключение к Astra Redis

2.1. Подключение через redis-cli

2.1.1. Из контейнера

Подключение с аутентификацией:

```
docker exec -it redis redis-cli -a mysecretpassword
```

Подключение с использованием URI:

docker exec -it redis redis-cli -u redis://:mysecretpassword@localhost:6379

2.1.2. С хост-машины

Установка redis-tools (если не установлен):

sudo apt install redis-tools

Подключение:

redis-cli -h localhost -p 6379 -a mysecretpassword

2.2. Аутентификация

При подключении необходимо использовать пароль.

Команда AUTH:

redis-cli -h localhost -p 6379 -a mysecretpassword AUTH mysecretpassword

Проверка подключения командой ping:

redis-cli -h localhost -p 6379 -a mysecretpassword ping

Ожидаемый ответ: PONG

При некорректном пароле вернется ошибка:

(error) WRONGPASS invalid username-password pair

2.3. Подключение через RedisInsight

1. Откройте http://localhost
2. Введите учетные данные:
    - Логин: admin
    - Пароль: mysecretpassword
3. RedisInsight автоматически подключится к Redis

2.4. Подключение из кода

Python (redis-py):

```
import redis

r = redis.Redis(
    host='localhost',
    port=6379,
    password='mysecretpassword',
    decode_responses=True
)

r.ping()  # True
```

Node.js (ioredis):

```
const Redis = require('ioredis');

const redis = new Redis({
  host: 'localhost',
  port: 6379,
  password: 'mysecretpassword'
});
```

```
redis.ping().then(result => {
  console.log(result); // PONG
});
```

---

3. Основные команды Redis

3.1. Получение информации о сервере

Общая информация о сервере:

```
redis-cli -h localhost -p 6379 -a mysecretpassword INFO
```

Информация о сервере:

```
redis-cli -h localhost -p 6379 -a mysecretpassword INFO server
```

Информация о клиентах:

```
redis-cli -h localhost -p 6379 -a mysecretpassword INFO clients
```

Информация о памяти:

```
redis-cli -h localhost -p 6379 -a mysecretpassword INFO memory
```

Информация о персистентности:

```
redis-cli -h localhost -p 6379 -a mysecretpassword INFO persistence
```

Статистика:

```
redis-cli -h localhost -p 6379 -a mysecretpassword INFO stats
```

Пример вывода INFO SERVER:

```
# Server
redis_version:7.0.15
redis_mode:standalone
os:Linux 6.8.0-49-generic x86_64
arch_bits:64
multiplexing_api:epoll
tcp_port:6379
uptime_in_seconds:3600
uptime_in_days:0
```

3.2. Работа со строками (String)

Установка значения:

```
SET key value
SET user:1000 "John Doe"
```

Установка сроком жизни (секунды):

```
SETEX session:abc 3600 "user_data"
```

Получение значения:

```
GET key
GET user:1000
```

Установка только если ключ не существует:

SETNX newkey "value"

Получение и изменение значения:

GETSET counter "100"

Проверка существования:

EXISTS key
EXISTS user:1000

Удаление ключа:

DEL key
DEL oldkey

Примеры:

SET product:1001 "Laptop"
GET product:1001 *# Laptop*
EXISTS product:1001 *# 1*
DEL product:1001
GET product:1001 *# (nil)*

3.3. Работа с хешами (Hash)

Установка полей:

HSET user:1000 name "John Doe"
HSET user:1000 email "john@example.com"

Установка нескольких полей:

HMSET user:1000 field1 value1 field2 value2

Получение одного поля:

HGET user:1000 name

Получение нескольких полей:

HMGET user:1000 name email

Получение всех полей и значений:

HGETALL user:1000

Получение всех полей:

HKEYS user:1000

Получение всех значений:

HVALS user:1000

Удаление поля:

HDEL user:1000 email

Проверка существования поля:

HEXISTS user:1000 email

3.4. Работа со списками (List)

Добавление в начало списка:

LPUSH queue:tasks task1

Добавление в конец списка:

RPUSH queue:tasks task2 task3

Извлечение из начала:

LPOP queue:tasks

Извлечение из конца:

RPOP queue:tasks

Получение по индексу:

LINDEX messages 0

Получение диапазона:

LRANGE queue:tasks 0 -1

Установка по индексу:

LSET messages 1 "updated"

Удаление элементов:

LREM queue:tasks 1 "task1"

Длина списка:

LLEN queue:tasks

3.5. Работа с множествами (Set)

Добавление элементов:

SADD tags "redis" "database" "cache"

Получение всех элементов:

SMEMBERS tags

Проверка принадлежности:

SISMEMBER tags "redis"

Удаление элементов:

SREM tags "cache"

Количество элементов:

SCARD tags

Случайный элемент:

SRANDMEMBER tags

Случайный элемент с удалением:

SPOP tags

3.6. Работа с упорядоченными множествами (Sorted Set)

Добавление со счетом:

ZADD leaderboard 100 "player1" 85 "player2" 95 "player3"

Количество элементов:

ZCARD leaderboard

Получение по порядку:

ZRANGE leaderboard 0 -1 WITHSCORES

Получение в обратном порядке:

ZREVRANGE leaderboard 0 -1 WITHSCORES

Получение счета элемента:

ZSCORE leaderboard "player1"

Удаление:

ZREM leaderboard "player2"

Увеличение счета:

ZINCRBY leaderboard 10 "player1"

---

4. Мониторинг и статистика

4.1. Мониторинг в реальном времени

Мониторинг всех команд (выполняется внутри контейнера):

docker exec -it redis redis-cli -a mysecretpassword MONITOR

4.2. Список подключенных клиентов

redis-cli -h localhost -p 6379 -a mysecretpassword CLIENT LIST

Пример вывода:

id=3 addr=192.168.1.100:54321 fd=8 name= age=120 idle=0 flags=N db=0
id=4 addr=192.168.1.101:54322 fd=9 name= age=60 idle=5 flags=N db=0

4.3. Статистика сервера
redis-cli -h localhost -p 6379 -a mysecretpassword INFO STATS

Таблица 2 — Ключевые метрики

Метрика	Описание
total_connections_received	Всего подключений
total_commands_processed	Всего команд выполнено
instantaneous_ops_per_sec	Операций в секунду
keyspace_hits	Успешных обращений к ключам
keyspace_misses	Промахов кэша

4.4. Медленные команды

Получение последних 10 медленных команд:

redis-cli -h localhost -p 6379 -a mysecretpassword SLOWLOG GET 10

Количество медленных команд:

redis-cli -h localhost -p 6379 -a mysecretpassword SLOWLOG LEN

4.5. Использование памяти

redis-cli -h localhost -p 6379 -a mysecretpassword INFO MEMORY

Пример вывода:

```
# Memory
used_memory:2097152
used_memory_human:2.00M
used_memory_rss:4194304
used_memory_peak:3145728
used_memory_peak_human:3.00M
mem_fragmentation_ratio:2.00
```

4.6. Мониторинг через RedisInsight

1. Откройте http://localhost
2. Войдите с учетными данными (admin/mysecretpassword)
3. Используйте встроенные дашборды для мониторинга

---

5. Резервное копирование и восстановление

5.1. Создание резервной копии

5.1.1. RDB Snapshot (синхронно)

docker exec redis redis-cli -a mysecretpassword SAVE

5.1.2. RDB Snapshot (асинхронно, в фоне)

docker exec redis redis-cli -a mysecretpassword BGSAVE

5.1.3. Проверка последнего сохранения

redis-cli -h localhost -p 6379 -a mysecretpassword LASTSAVE

5.2. Восстановление из резервной копии

Остановите Redis:

docker compose stop redis

Скопируйте файл dump.rdb в volume:

docker cp dump.rdb redis:/data/

Запустите Redis:

docker compose start redis

5.3. Доступ к файлам данных

Просмотр содержимого volume:

docker volume ls | grep redis

Монтирование volume для доступа:

docker run --rm -v redis_redis_data:/data -v $(pwd):/backup alpine cp /data/dump.rdb /backup/

---

6. Персистентность

6.1. Текущие настройки

redis-cli -h localhost -p 6379 -a mysecretpassword INFO PERSISTENCE

Пример вывода:

```
# Persistence
loading:0
rdb_changes_since_last_save:100
rdb_bgsave_in_progress:0
rdb_last_save_time:1625098000
rdb_last_bgsave_status:ok
aof_enabled:1
aof_last_write_status:ok
```

6.2. Типы персистентности

6.2.1. RDB (Snapshotting)

Периодическое создание снимков базы данных.

Преимущества: * Компактные файлы * Быстрое восстановление

Недостатки: * Возможна потеря данных между снимками

6.2.2. AOF (Append Only File)

Логирование всех операций записи.

Преимущества: * Большая надежность * Только добавляется в файл

Недостатки: * Больший размер файла

6.3. Текущая конфигурация

В docker-compose используется:

```
command: >
  sh -c "redis-server --appendonly yes --requirepass $$REDIS_PASSWORD"
```

Это означает: * --appendonly yes — включен режим AOF * --requirepass $$REDIS_PASSWORD — пароль mysecretpassword

6.4. Проверка AOF

redis-cli -h localhost -p 6379 -a mysecretpassword INFO persistence | grep aof

---

Приложение A. Часто задаваемые вопросы

A.1. Как подключиться к Redis извне?

Redis слушает на всех интерфейсах (0.0.0.0). Подключение по IP сервера:

redis-cli -h <IP-сервера> -p 6379 -a mysecretpassword

A.2. Как изменить пароль?

1. Остановите контейнеры:

   docker compose down

2. Измените пароль в docker-compose.yml:

   environment**:**
   REDIS_PASSWORD**:** новый_пароль

3. Также обновите пароль в nginx-auth/.htpasswd

4. Запустите заново:

   docker compose up -d

A.3. Как сбросить все данные?

Удаление всех ключей:

docker exec redis redis-cli -a mysecretpassword FLUSHALL

Удаление с очисткой AOF и RDB:

docker exec redis redis-cli -a mysecretpassword FLUSHALL ASYNC

A.4. Как сделать дамп базы?

Создание дампа:

docker exec redis redis-cli -a mysecretpassword SAVE

Копирование из контейнера:

docker cp redis:/data/dump.rdb ./backup-dump.rdb
docker cp redis:/data/appendonly.aof ./backup-aof.aof