

环境准备

切片集群

主从哨兵

1. 部署

1.1. 二进制方式

1.1.1. 下载二进制包

1.1.2. 部署

1.2. docker-compose 容器方式

1.3. 配置连接&认证参数

1.3.1. 连接认证参数

1.3.2. 配置服务控制 systemd

2. 配置到 Prometheus

3 Dashboard

4. 告警规则

4.1. Redis down

4.2. Redis missing master

4.2.1. 规则拆解分析

4.2.2. 完整规则

4.3. Redis too many masters

4.4. Redis disconnected slaves

4.4.1. 规则拆解分析

4.4.2. 完整规则

4.5. Redis replication broken Redis 复制已中断

4.5.1. 规则拆解分析

4.5.2. 完整规则

4.6. Redis cluster flapping Redis群集摆动

4.6.1. 规则拆解分析

4.6.2. 完整规则

4.7. Redis missing backup

4.8. Redis out of system memory

4.9. Redis out of configured maxmemory

4.10. Redis too many connections

4.11. Redis not enough connections

4.12. Redis rejected connections

4.13. Redis 使用内存过多

4.13.1 规则拆解分析

4.13.2 完整规则

4.14. Redis 已有内存碎片

命令行配置参数

环境准备

在一台服务器上部署 Redis 的主从、哨兵和切片集群

条件是 服务器上安装 docker。

切片集群

version: "3.9"

services:

redis7001:

```
labels:
  co.elastic.logs/module: redis
image: redis:6.2.6-alpine
restart: always
environment:
  REDISCLI_AUTH: 'FdMyKBIVv'
volumes:
  - "/etc/localtime:/etc/localtime"
  - "./data:/data/"
  - "./config/cluster-7001.conf:/usr/local/etc/redis/redis.conf:ro"
command: redis-server /usr/local/etc/redis/redis.conf
sysctls:
  - net.core.somaxconn=2048
redis7002:
labels:
  co.elastic.logs/module: redis
image: redis:6.2.6-alpine
restart: always
environment:
  REDISCLI_AUTH: 'FdMyKBIVv'
volumes:
  - "/etc/localtime:/etc/localtime"
  - "./data:/data/"
  - "./config/cluster-7002.conf:/usr/local/etc/redis/redis.conf:ro"
command: redis-server /usr/local/etc/redis/redis.conf
sysctls:
  - net.core.somaxconn=2048
redis7003:
labels:
  co.elastic.logs/module: redis
image: redis:6.2.6-alpine
restart: always
environment:
  REDISCLI_AUTH: 'FdMyKBIVv'
volumes:
  - "/etc/localtime:/etc/localtime"
  - "./data:/data/"
  - "./config/cluster-7003.conf:/usr/local/etc/redis/redis.conf:ro"
command: redis-server /usr/local/etc/redis/redis.conf
sysctls:
  - net.core.somaxconn=2048
redis7004:
labels:
  co.elastic.logs/module: redis
image: redis:6.2.6-alpine
restart: always
environment:
  REDISCLI_AUTH: 'FdMyKBIVv'
volumes:
  - "/etc/localtime:/etc/localtime"
  - "./data:/data/"
  - "./config/cluster-7004.conf:/usr/local/etc/redis/redis.conf:ro"
command: redis-server /usr/local/etc/redis/redis.conf
sysctls:
  - net.core.somaxconn=2048
redis7005:
labels:
  co.elastic.logs/module: redis
```

```

image: redis:6.2.6-alpine
restart: always
environment:
  REDISCLI_AUTH: 'FdMyKBIVv'
volumes:
  - "/etc/localtime:/etc/localtime"
  - "./data:/data/"
  - "./config/cluster-7005.conf:/usr/local/etc/redis/redis.conf:ro"
command: redis-server /usr/local/etc/redis/redis.conf
sysctls:
  - net.core.somaxconn=2048
redis7006:
  labels:
    co.elastic.logs/module: redis
  image: redis:6.2.6-alpine
  restart: always
  environment:
    REDISCLI_AUTH: 'FdMyKBIVv'
  volumes:
    - "/etc/localtime:/etc/localtime"
    - "./data:/data/"
    - "./config/cluster-7006.conf:/usr/local/etc/redis/redis.conf:ro"
  command: redis-server /usr/local/etc/redis/redis.conf
  sysctls:
    - net.core.somaxconn=2048

```

cluster-7001.conf

```

bind 0.0.0.0
port 6379
daemonize no
timeout 0
tcp-keepalive 0

#####
# ARM 架构设置
ignore-warnings ARM64-COW-BUG

#####
# 集群设置
cluster-enabled yes
cluster-config-file cluster-redis-do-not-edit-7001.conf

# 不需要集群的全部节点完好才提供服务
cluster-require-full-coverage no

#####

loglevel notice

databases 16
maxclients 20000

dir "/data/"
dbfilename "cluster-dump-7001.db"
stop-writes-on-bgsave-error yes
rdbcompression yes

```

```
rdbchecksum yes

appendonly yes
appendfilename "appendonly-7001.aof"
appendfsync everysec

aof-load-truncated yes
no-appendfsync-on-rewrite yes
auto-aof-rewrite-percentage 100
auto-aof-rewrite-min-size 64mb
aof-rewrite-incremental-fsync yes

lua-time-limit 5000
slowlog-log-slower-than 10000
slowlog-max-len 128
latency-monitor-threshold 0
hash-max-ziplist-entries 512
hash-max-ziplist-value 64
list-max-ziplist-entries 512
list-max-ziplist-value 64
set-max-intset-entries 512
zset-max-ziplist-entries 128
zset-max-ziplist-value 64
hll-sparse-max-bytes 3000
activerehashing yes
client-output-buffer-limit normal 0 0 0
client-output-buffer-limit slave 256mb 64mb 60
client-output-buffer-limit pubsub 32mb 8mb 60
hz 10

protected-mode yes

requirepass "FdMyKBIVv"
masterauth "FdMyKBIVv"
```

主从哨兵

```
version: "3.9"
services:
  redis:
    labels:
      co.elastic.logs/module: redis
    image: redis:6.2.6-alpine
    restart: always
    volumes:
      - "/etc/localtime:/etc/localtime"
      - "./data:/data/"
      - "./config:/usr/local/etc/redis/"
    command: redis-server /usr/local/etc/redis/redis.conf
    sysctls:
      - net.core.somaxconn=2048
    networks:
      sentinel-net:
        ipv4_address: "172.18.0.2"
```

```
redis-slave1:
  labels:
    co.elastic.logs/module: redis
  image: redis:6.2.6-alpine
  restart: always
  volumes:
    - "/etc/localtime:/etc/localtime"
    - "./data:/data/"
    - "./config:/usr/local/etc/redis/"
  command: redis-server /usr/local/etc/redis/redis-slave1.conf
  sysctls:
    - net.core.somaxconn=2048
  networks:
    sentinel-net:
      ipv4_address: "172.18.0.3"
redis-slave2:
  labels:
    co.elastic.logs/module: redis
  image: redis:6.2.6-alpine
  restart: always
  volumes:
    - "/etc/localtime:/etc/localtime"
    - "./data:/data/"
    - "./config:/usr/local/etc/redis/"
  command: redis-server /usr/local/etc/redis/redis-slave2.conf
  sysctls:
    - net.core.somaxconn=2048
  networks:
    sentinel-net:
      ipv4_address: "172.18.0.4"
sentinel-1:
  labels:
    co.elastic.logs/module: redis
  image: redis:6.2.6-alpine
  restart: always
  volumes:
    - "/etc/localtime:/etc/localtime"
    - "./data:/data/"
    - "./config:/usr/local/etc/redis/"
  command: redis-server /usr/local/etc/redis/sentinel.conf --sentinel
  sysctls:
    - net.core.somaxconn=2048
  networks:
    sentinel-net:
      ipv4_address: "172.18.0.5"
sentinel-2:
  labels:
    co.elastic.logs/module: redis
  image: redis:6.2.6-alpine
  restart: always
  volumes:
    - "/etc/localtime:/etc/localtime"
    - "./data:/data/"
    - "./config:/usr/local/etc/redis/"
  command: redis-server /usr/local/etc/redis/sentinel-2.conf --sentinel
  sysctls:
    - net.core.somaxconn=2048
  networks:
```

```

    sentinel-net:
      ipv4_address: "172.18.0.6"
sentinel-3:
  labels:
    co.elastic.logs/module: redis
  image: redis:6.2.6-alpine
  restart: always
  volumes:
    - "/etc/localtime:/etc/localtime"
    - "./data:/data/"
    - "./config:/usr/local/etc/redis"
  command: redis-server /usr/local/etc/redis/sentinel-3.conf --sentinel
  sysctls:
    - net.core.somaxconn=2048
  networks:
    sentinel-net:
      ipv4_address: "172.18.0.7"
networks:
  sentinel-net:
    ipam:
      driver: default
      config:
        - subnet: "172.18.0.0/24"

```

redis.conf

```

bind 0.0.0.0
port 6379
daemonize no
protected-mode no
timeout 0
tcp-keepalive 0

loglevel notice
databases 16
maxclients 20000

dir "/data/"
dbfilename "cluster-dump.db"
stop-writes-on-bgsave-error yes
rdbcompression yes
rdbchecksum yes

appendonly yes
appendfilename "appendonly.aof"
appendfsync everysec

aof-load-truncated yes
no-appendfsync-on-rewrite yes
auto-aof-rewrite-percentage 100
auto-aof-rewrite-min-size 64mb
aof-rewrite-incremental-fsync yes

lua-time-limit 5000
slowlog-log-slower-than 10000

```

```
slowlog-max-len 128
latency-monitor-threshold 0
hash-max-ziplist-entries 512
hash-max-ziplist-value 64
list-max-ziplist-entries 512
list-max-ziplist-value 64
set-max-intset-entries 512
zset-max-ziplist-entries 128
zset-max-ziplist-value 64
hll-sparse-max-bytes 3000
activeresharding yes
client-output-buffer-limit normal 0 0 0
client-output-buffer-limit slave 256mb 64mb 60
client-output-buffer-limit pubsub 32mb 8mb 60
hz 10

requirepass "FdMyKBIVv"
masterauth "FdMyKBIVv"
```

sentinel.conf

```
bind 0.0.0.0
port 6370
daemonize no

sentinel announce-ip 172.18.0.5
sentinel announce-port 6370

sentinel monitor epmsMaster 172.18.0.2 6379 2
sentinel down-after-milliseconds epmsMaster 3000
sentinel auth-pass epmsMaster FdMyKBIVvz
```

1. 部署

github 地址 https://github.com/oliver006/redis_exporter

1.1. 二进制方式

1.1.1. 下载二进制包

可以下载不同的版本

https://github.com/oliver006/redis_exporter/releases

```
wget
https://github.com/oliver006/redis_exporter/releases/download/v1.52.0/redis_exporter-v1.52.0.linux-amd64.tar.gz
```

1.1.2. 部署

```
tar -xf redis_exporter-v1.52.0.linux-amd64.tar.gz -C /usr/local/
ln -s /usr/local/redis_exporter-v1.52.0.linux-amd64/ /usr/local/redis_exporter
```

1.2. docker-compose 容器方式

docker 镜像下载网址 https://quay.io/repository/oliver006/redis_exporter?tab=tags&tag=latest

compose.yml

```
version: '3.9'
services:
  redis_exporter:
    environment:
      REDIS_ADDR: "ip:port"
      REDIS_PASSWORD: '密码'
    image: quay.io/oliver006/redis_exporter:alpine
    command: "-is-cluster"
    restart: always
    network_mode: "host"
    expose:
      - "9121"
```

1.3. 配置连接&认证参数

1.3.1. 连接认证参数

- `-redis.addr` 指定启动时连接的 Redis 服务的 IP 和端口，集群成员中任何一个都可以。格式：**ip:port**，示例：`192.168.3.100:6379`
- `-redis-user` 指定 Redis 服务的认证用户，高版本启用了 ACL 认证模式的 Redis 支持。
- `-redis.password` 指定 Redis 服务的认证密码，密码部分不要用任何单引号或双引号（会被判断为密码的一部分）。格式：**-redis.password=密码**，示例：`-redis.password=Uiasdf;k889s`
- `-is-cluster` 如果被监控的服务是以分片集群方式运行，需要加上这个参数，如果被监控的服务是以主/从或者哨兵模式运行，不加此参数。

1.3.2. 配置服务控制 systemd

分片集群模式

/etc/systemd/system/redis-exporter.service

```
[Unit]
Description=The Redis Cluster Exporter  监控程序
After=network-online.target
Wants=network-online.target

[Service]
# 集群模式
ExecStart=/usr/local/redis_exporter/redis_exporter -is-cluster -redis.addr
node1-redis-host:6379 -redis.password=密码 -include-system-metrics

KillSignal=SIGQUIT
Restart=always
RestartPreventExitStatus=1 6 SIGABRT
TimeoutStopSec=5
KillMode=process
PrivateTmp=true
LimitNOFILE=1048576
LimitNPROC=1048576
```



```
[Install]
WantedBy=multi-user.target
```

启动服务

```
systemctl enable --now redis-exporter
```

验证

```
curl localhost:9121/metrics |grep redis_up
```

```
[root@prometheus ~]# curl localhost:9121/metrics |grep redis_up
% Total    % Received % Xferd  Average Speed   Time    Time     Current
           Dload  Upload   Total     Spent    Left     Speed
100 29377    0 29377    0    0 2115k      0 --:--:-- --:--:-- --:--:-- 2206k
# HELP redis_up Information about the Redis instance
# TYPE redis_up gauge
redis_up 1
# HELP redis_uptime_in_seconds uptime_in_seconds metric
# TYPE redis_uptime_in_seconds gauge
redis_uptime_in_seconds 63616
```

CSDN @shark_西瓜甜

主从模式

/etc/systemd/system/redis-master-exporter.service

```
[Unit]
Description=The Redis Master Or Slave Exporter 监控程序
After=network-online.target
Wants=network-online.target

[Service]
ExecStart=/usr/local/redis_exporter/redis_exporter -redis.addr 127.0.0.1:27001 -
web.listen-address=0.0.0.0:9122 -redis.password=密码 -include-system-metrics

KillSignal=SIGQUIT
Restart=always
RestartPreventExitStatus=1 6 SIGABRT

TimeoutStopSec=5
KillMode=process
PrivateTmp=true
LimitNOFILE=1048576
LimitNPROC=1048576

[Install]
WantedBy=multi-user.target
```

哨兵模式

因为目前版本未找到对 **sentinel** 设置认证信息的配置，所以 **sentinel** 自身是不用配置密码的，否则会报错。

/etc/systemd/system/redis-sentinel-exporter.service

```
[Unit]
Description=The Redis Sentinel Exporter 监控程序
After=network-online.target
```

```

Wants=network-online.target

[Service]
ExecStart=/usr/local/redis_exporter/redis_exporter -redis.addr 127.0.0.1:27001 -
web.listen-address=0.0.0.0:9123

KillSignal=SIGQUIT
Restart=always
RestartPreventExitStatus=1 6 SIGABRT

TimeoutStopSec=5
KillMode=process
PrivateTmp=true
LimitNOFILE=1048576
LimitNPROC=1048576

[Install]
WantedBy=multi-user.target

```

2. 配置到 Prometheus

```


- job_name: 'redis_cluster'
  static_configs:
    - targets:
        # 集群模式
        - 172.20.0.2:6379
        - 172.20.0.3:6379
        - 172.20.0.4:6379
        - 172.20.0.5:6379
        - 172.20.0.6:6379
        - 172.20.0.7:6379
  metrics_path: /scrape
  relabel_configs:
    - source_labels: [__address__]
      target_label: __param_target
    - source_labels: [__param_target]
      target_label: instance
    - target_label: __address__
      replacement: localhost:9121 # redis_exporter 的 IP 和端口
- job_name: 'redis_master_slave'
  static_configs:
    - targets:
        # 主从模式
        - 172.18.0.2:6379
        - 172.18.0.3:6379
        - 172.18.0.4:6379
  metrics_path: /scrape
  relabel_configs:
    - source_labels: [__address__]
      target_label: __param_target
    - source_labels: [__param_target]
      target_label: instance
    - target_label: __address__
      replacement: localhost:9122
- job_name: 'redis_sentinel'

```

```
static_configs:
- targets:
  # 哨兵模式
  - 172.18.0.5:6370
  - 172.18.0.6:6370
  - 172.18.0.7:6370
metrics_path: /scrape
relabel_configs:
- source_labels: [__address__]
  target_label: __param_target
- source_labels: [__param_target]
  target_label: instance
- target_label: __address__
  replacement: localhost:9123
```

重新加载配置文件

```
curl -X POST --user 'shark:123456' localhost:9090/-/reload
```


Prometheus
Alerts
Graph
Status ▾
Help

Targets

All scrape pools ▾

All
Unhealthy
Expand All

beijing-servers (2/2 up) show more

mysql (3/3 up) show more

prometheus (1/1 up) show less

Endpoint	State	Labels
http://localhost:9090/metrics	UP	instance="localh

redis_cluster (6/6 up) show more

redis_master_slave (3/3 up) show more

redis_sentinel (3/3 up) show more

CSDN @shark_西瓜甜

为了实验目的，同时监控了 Redis 的多种模式。

3 Dashboard

ID: 11835

Import dashboard

Import dashboard from file or Grafana.com

Importing dashboard from Grafana.com

Published by	downager
Updated on	2020-03-02 18:42:02

Options

Name

Redis Dashboard for Prometheus Redis Exporter (helm stable/redis-ha)

Folder

General

Unique identifier (UID)

The unique identifier (UID) of a dashboard can be used to uniquely identify a dashboard between multiple Grafana installs. The UID allows having consistent URLs for accessing dashboards so changing the title of a dashboard will not break any bookmarked links to that dashboard.

xDLNRKUWz

Change uid

Prometheus

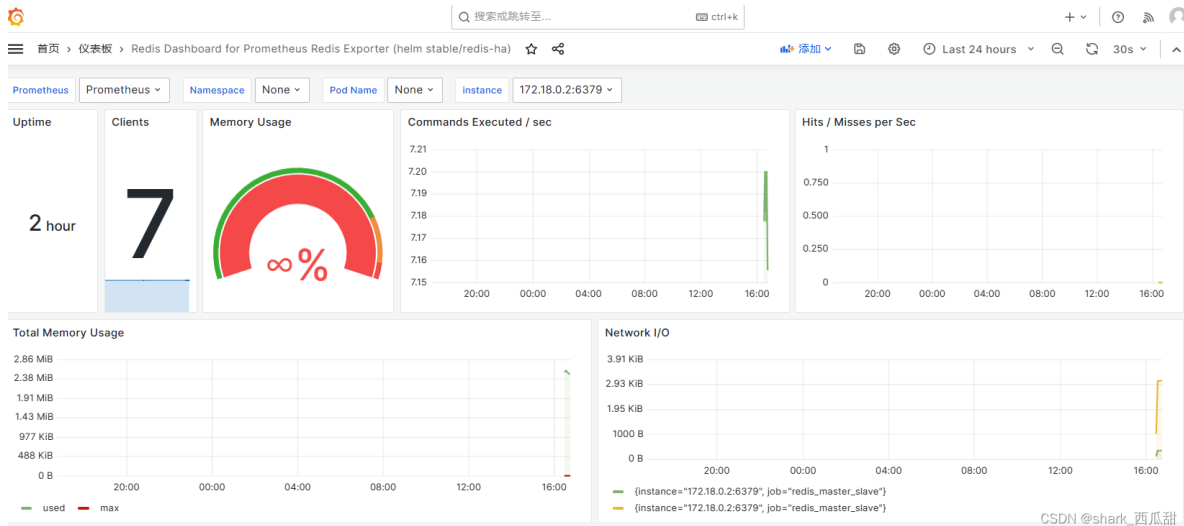
Prometheus

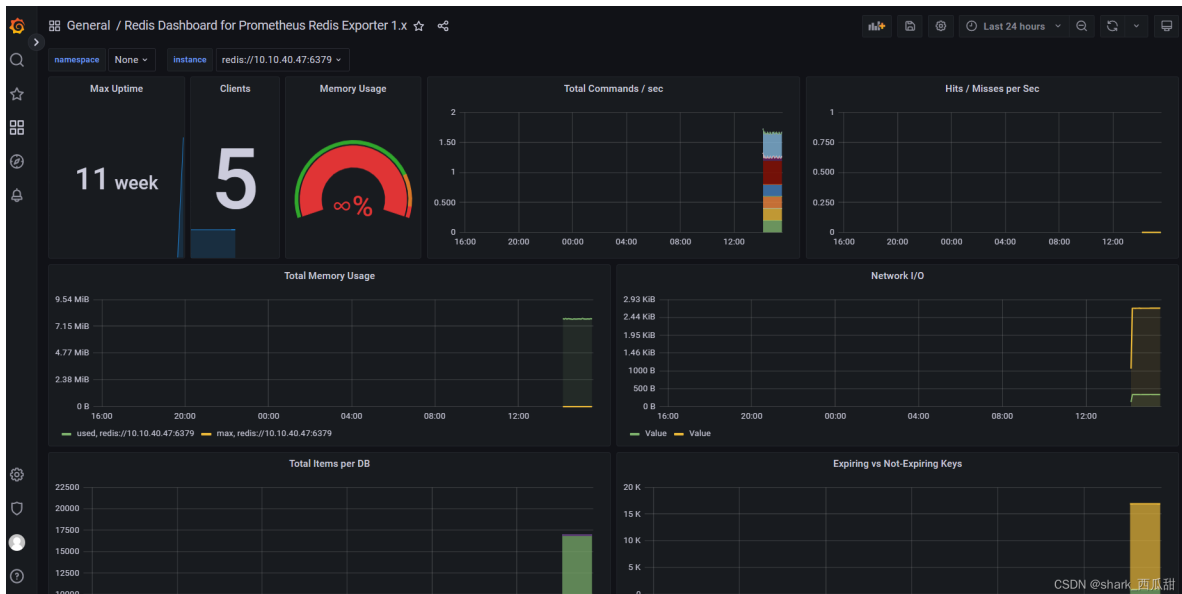
Import

Cancel

CSDN @shark_西瓜甜

效果图





4. 告警规则

4.1. Redis down

Redis instance is down

```
- alert: RedisDown
  expr: redis_up == 0
  for: 0m
  labels:
    severity: critical
  annotations:
    summary: Redis down (instance {{ $labels.instance }})
    description: "Redis instance is down\n VALUE = {{ $value }}\n LABELS = {{ $labels }}"
```

4.2. Redis missing master

Redis集群没有标记为master的节点。

4.2.1. 规则拆解分析

语法回归: `or`

`vector1 or vector2`

包含 `vector1` 的所有原始元素（标签集+值）以及另外包含 `vector2` 的在 `vector1` 中不具有匹配标签集的所有元素的向量。

因为如果集群中的确丢失了 **master**，表达式 `count(redis_instance_info{role="master"})` 值将会是空，这样就无法和 `1` 进行比较运算了。

因此，使用 `or`，再配合 `vector(0)`，使整个表达式左侧的最终结果是整数，以便于比较。

4.2.2. 完整规则

```
- alert: RedisMissingMaster
  expr: (count(redis_instance_info{role="master"}) or vector(0)) < 1
  for: 0m
  labels:
    severity: critical
  annotations:
    summary: Redis missing master (instance {{ $labels.instance }})
    description: "Redis cluster has no node marked as master.\n VALUE = {{ $value }}\n LABELS = {{ $labels }}"
```

4.3. Redis too many masters

Redis集群有太多标记为master的节点。

如果是 cluster 模式, 修改 (`> 1`) 为正确的 master 数量, 比如正常是 3 个master, 那就修改为: (`> 3`)

主从模式

```
- alert: RedisTooManyMasters
  expr: count(redis_instance_info{job="redis_master_slave",role="master"}) > 1
  for: 0m
  labels:
    severity: critical
  annotations:
    summary: Redis too many masters (instance {{ $labels.instance }})
    description: "Redis cluster has too many nodes marked as master.\n VALUE = {{ $value }}\n LABELS = {{ $labels }}"
```

集群模式

```
- alert: RedisTooManyMasters
  expr: count(redis_instance_info{job="redis_cluster",role="master"}) > 3
  for: 0m
  labels:
    severity: critical
  annotations:
    summary: Redis too many masters (instance {{ $labels.instance }})
    description: "Redis cluster has too many nodes marked as master.\n VALUE = {{ $value }}\n LABELS = {{ $labels }}"
```

4.4. Redis disconnected slaves

4.4.1. 规则拆解分析

检查 Redis 的 从服务是否连接正常。
指标 `redis_connected_slaves` 返回的是集群中每个主实例已经连接的从实例数量，如果本身自己就是从实例，其值则是 0



因此通过 `count` 函数可以计算出一个集群中总实例数，通过 `sum` 函数可以计算出已经处于连接状态的从实例数。

总实例数 - 从实例数 = 当前的主实例数

当前的主实例数 减去 集群中应该的主实例数，结果应该不能大于 0，否则就是当前的主实例数多于集群中应该的主实例数，也就是 从实例数少了。

加入 Prometheus 监控了多个 Redis 集群，可以在配置 Prometheus 的时候，将不同的 Redis 集群配置在不同的 `job_name` 下，就像本实例那样。这样在获取数据的时候可以利用标签 `job` 分别设置告警规则。



4.4.2. 完整规则

```
- alert: RedisDisconnectedSlaves
  expr: count(redis_connected_slaves) - sum(redis_connected_slaves) - 1 > 0
  for: 0m
  labels:
    severity: critical
  annotations:
    summary: Redis disconnected slaves (instance {{ $labels.instance }})
    description: "Redis not replicating for all slaves. Consider reviewing the redis replication status.\n VALUE = {{ $value }}\n LABELS = {{ $labels }}"
```

4.5. Redis replication broken Redis 复制已中断

在某一段时间范围内，复制曾经中断过。

4.5.1. 规则拆解分析

语法回顾: 函数 `delta` 计算范围向量中每个时间序列元素的第一个值和最后一个值之间的差值，返回一个具有给定增量和等效标签的即时向量。

指标 `redis_connected_slaves` 统计的是每个主实例和其已经连接的从实例个数。

`redis_connected_slaves[1m]` 返回的是每个主实例在最近 1 分钟内和其已经连接的从实例个数。这些数据差值等于 0，表示从实例很稳定；如果差值小于 0，则表示从实例丢失过，也就是复制中断过。

4.5.2. 完整规则

```
- alert: RedisReplicationBroken
  expr: delta(redis_connected_slaves[1m]) < 0
  for: 0m
  labels:
    severity: critical
  annotations:
    summary: Redis replication broken (instance {{ $labels.instance }})
    description: "Redis instance lost a slave\n  VALUE = {{ $value }}"
  LABELS = {{ $labels }}
```

4.6. Redis cluster flapping Redis群集摆动

4.6.1. 规则拆解分析

在Redis副本连接中检测到更改。当副本节点失去与主节点的连接并重新连接（也称为摆动）时，可能会发生这种情况。

语法回顾:

函数 `changes(v range-vector)` 统计出每个指标的范围向量中每个值更改的次数。

如果集群中和某个主实例连接的从实例个数在最近 1 分钟内，由 2 个变成了 1 个，之后由变成了 2 个。这种情况就是发生了变化，可以被函数 `changes` 统计出来。

4.6.2. 完整规则

```
- alert: RedisClusterFlapping
  expr: changes(redis_connected_slaves[1m]) > 1
  for: 2m
  labels:
    severity: critical
  annotations:
    summary: Redis cluster flapping (instance {{ $labels.instance }})
    description: "Changes have been detected in Redis replica connection. This can occur when replica nodes lose connection to the master and reconnect (a.k.a flapping).\n  VALUE = {{ $value }}\n  LABELS = {{ $labels }}"
  LABELS = {{ $labels }}
```

4.7. Redis missing backup

Redis has not been backed up for 24 hours

Redis已24小时未备份


```

- alert: RedisMissingBackup
  expr: time() - redis_rdb_last_save_timestamp_seconds > 60 * 60 * 24
  for: 0m
  labels:
    severity: critical
  annotations:
    summary: Redis missing backup (instance {{ $labels.instance }})
    description: "Redis has not been backed up for 24 hours\n VALUE = {{ $value }}\n LABELS = {{ $labels }}"

```

4.8. Redis out of system memory

Redis 当前使用的内存超出了系统内存(> 90%)

redis_exporter 需要设置启动参数: `-include-system-metrics` 或者环境变量

`REDIS_EXPORTER_INCL_SYSTEM_METRICS=true` 启动。

```

- alert: RedisOutOfSystemMemory
  expr: redis_memory_used_bytes / redis_total_system_memory_bytes * 100 > 90
  for: 2m
  labels:
    severity: warning
  annotations:
    summary: Redis out of system memory (instance {{ $labels.instance }})
    description: "Redis is running out of system memory (> 90%)\n VALUE = {{ $value }}\n LABELS = {{ $labels }}"

```

4.9. Redis out of configured maxmemory

Redis 当前使用的内存超出了配置的最大可用内存 maxmemory (> 90%)

Redis Server 需要再配置文件中添加 `maxmemory 4096mb` , 没有配置此参数, 指标

`redis_memory_max_bytes` 则返回 0

Q redis_memory_max_bytes Execute

Table Graph Load time: 6ms Resolution: 14s Result series: 8

	Evaluation time
redis_memory_max_bytes(instance="172.18.0.2:6379", job="redis_master_slave")	0
redis_memory_max_bytes(instance="172.18.0.3:6379", job="redis_master_slave")	0
redis_memory_max_bytes(instance="172.18.0.4:6379", job="redis_master_slave")	0
redis_memory_max_bytes(instance="172.20.0.2:6379", job="redis_cluster")	4294967296
redis_memory_max_bytes(instance="172.20.0.3:6379", job="redis_cluster")	0
redis_memory_max_bytes(instance="172.20.0.5:6379", job="redis_cluster")	0
redis_memory_max_bytes(instance="172.20.0.6:6379", job="redis_cluster")	0
redis_memory_max_bytes(instance="172.20.0.7:6379", job="redis_cluster")	0

CSDN @shark_西瓜甜

```

- alert: RedisOutOfConfiguredMaxmemory
  expr: redis_memory_used_bytes / redis_memory_max_bytes * 100 > 90
  for: 2m
  labels:
    severity: warning
  annotations:
    summary: Redis out of configured maxmemory (instance {{ $labels.instance }})
    description: "Redis is running out of configured maxmemory (> 90%)\n VALUE = {{ $value }}\n LABELS = {{ $labels }}"

```

4.10. Redis too many connections

Redis is running out of connections (> 90% used)

指标 `redis_config_maxclients` 返回配置的最大连接数，默认 **20000**。

```
- alert: RedisTooManyConnections
  expr: redis_connected_clients / redis_config_maxclients * 100 > 90
  for: 2m
  labels:
    severity: warning
  annotations:
    summary: Redis too many connections (instance {{ $labels.instance }})
    description: "Redis is running out of connections (> 90% used)\n VALUE = {{ $value }}\n LABELS = {{ $labels }}"
```

4.11. Redis not enough connections

连接到 Redis 的客户端过少。

这个应根据实际情况设置阈值，正常的生产环境，业务正常的情况下，这个连接数应该比较多，不会是个几个。

同时也好考虑，应用连接的是主实例，还是从实例。因为有的小公司或者小的系统，应用没有采用读写分离，只连接了主实例。

```
- alert: RedisNotEnoughConnections
  expr: redis_connected_clients < 5
  for: 2m
  labels:
    severity: warning
  annotations:
    summary: Redis not enough connections (instance {{ $labels.instance }})
    description: "Redis 实例的连接太少了 (> 5)\n VALUE = {{ $value }}\n LABELS = {{ $labels }}"
```

4.12. Redis rejected connections

某些到Redis的连接已被拒绝。

通常会监控一段时间内被拒绝的连接数是否在增长，所以可以使用函数 `increase`，此函数计算一段时间内的增长量。

```
- alert: RedisRejectedConnections
  expr: increase(redis_rejected_connections_total[1m]) > 0
  for: 0m
  labels:
    severity: critical
  annotations:
    summary: Redis rejected connections (instance {{ $labels.instance }})
    description: "Some connections to Redis has been rejected\n VALUE = {{ $value }}\n LABELS = {{ $labels }}"
```

4.13. Redis 使用内存过多

系统给的内存少，redis使用的内存多。

4.13.1 规则拆解分析

Redis 终端命令 `info memory` 返回的是当前实例的内存使用情况。其中有三个指标：

```
# Memory
used_memory:14640520
used_memory_human:13.96M
used_memory_rss:24936448
used_memory_rss_human:23.78M
...
mem_fragmentation_ratio:1.70
```

这里有一个 `mem_fragmentation_ratio` 的指标，它表示的就是 Redis 当前的内存碎片率。那么，这个碎片率是怎么计算的呢？其实，就是上面的两个指标 `used_memory_rss` 和 `used_memory` 相除的结果。

```
mem_fragmentation_ratio = used_memory_rss / used_memory
```

`used_memory_rss` 是操作系统实际分配给 Redis 的物理内存空间，里面就包含了碎片；而 `used_memory` 是 Redis 为了保存数据实际申请使用的空间。

`mem_fragmentation_ratio` 小于1，说明 `used_memory_rss` 小于了 `used_memory`，这意味着操作系统分配给 Redis 进程的物理内存，要小于Redis 实际存储数据的内存，也就是说 Redis 没有足够的物理内存可以使用了，这会导致 Redis 一部分内存数据会被换到 Swap 中，之后当Redis 访问 Swap 中的数据时，延迟会变大，性能下降。

`mem_fragmentation_ratio` 大于 1 但小于 1.5。这种情况是合理的。是存在一些正常的内存碎片。一个是由操作系统的内存分配器决定的，不会轻易修改；另一个是由 Redis 负载决定，也无法限制。所以，存在内存碎片也是正常的。

`mem_fragmentation_ratio` 大于 1.5。这表明内存碎片率已经超过了 50%。一般情况下，这个时候，我们就需要采取一些措施来降低内存碎片率了。

4.13.2 完整规则

```
- alert: RedisUsesTooMuchMemory
  expr: (redis_memory_used_rss_bytes / redis_memory_used_bytes) < 1
  for: 0m
  labels:
    severity: critical
  annotations:
    summary: Redis 可以使用的内存不够(instance {{ $labels.instance }})
    description: "系统分配的内存不够 Redis 使用\n VALUE = {{ $value }}\n LABELS = {{ $labels }}"
```

4.14. Redis 已有内存碎片

Redis 产生的内存碎片多。

```
- alert: RedisMemoryFragmentation
  expr: (redis_memory_used_rss_bytes / redis_memory_used_bytes) > 1.5
  for: 0m
  labels:
    severity: critical
  annotations:
    summary: Redis 内存碎片过多(instance {{ $labels.instance }})
    description: "Redis 已产生过多内存碎片 \n  VALUE = {{ $value }}\n LABELS = {{ $labels }}"
```

命令行配置参数

Name	Environment Variable Name	Description
-redis.addr	REDIS_ADDR	Redis实例的地址，默认为redis://localhost:6379。
-redis.user	REDIS_USER	用于身份验证的用户名（Redis 6.0及更新版本的Redis ACL）。
-redis.password	REDIS_PASSWORD	Redis实例的密码，默认为 ""（无密码）。
-redis.password-file	REDIS_PASSWORD_FILE	要抓取的Redis实例的密码文件，默认为 ""（无密码文件）。
-check-keys	REDIS_EXPORTER_CHECK_KEYS	要导出值和长度/大小的 key 模式的逗号分隔列表，例如：db3=user_count 将从db3导出 key user_count。如果省略，db默认为0。使用SCAN可以找到用这个标志指定的键模式。如果您需要glob模式匹配，请使用此选项；对于非模式key，check-single-keys的速度更快。警告：使用 --check-keys 来匹配大量的键可能会减慢导出程序的速度，使其无法完成对redis实例的抓取。
-check-single-keys	REDIS_EXPORTER_CHECK_SINGLE_KEYS	Comma separated list of keys to export value and length/size, eg: db3=user_count will export key user_count from db 3. db defaults to 0 if omitted. The -keys specified with this flag will be looked up directly without any glob pattern matching. Use this option if you don't need glob pattern matching; it is faster than check-keys.
-check-streams	REDIS_EXPORTER_CHECK_STREAMS	Comma separated list of stream-patterns to export info about streams, groups and consumers. Syntax is the same as check-keys.
-check-single-streams	REDIS_EXPORTER_CHECK_SINGLE_STREAMS	Comma separated list of streams to export info about streams, groups and consumers. The streams specified with this flag will be looked up directly without any glob pattern matching. Use this option if you don't need glob pattern matching; it is faster than check-streams.
-check-keys-batch-size	REDIS_EXPORTER_CHECK_KEYS_BATCH_SIZE	Approximate number of keys to process in each execution. This is basically the COUNT option that will be passed into the SCAN command as part of the execution of the key or key group metrics, see COUNT option. Larger value speeds up scanning. Still Redis is a single-threaded app, huge COUNT can affect production environment.

Name	Environment Variable Name	Description
-count-keys	REDIS_EXPORTER_COUNT_KEYS	Comma separated list of patterns to count, eg: db3=sessions:* will count all keys with prefix sessions: from db 3. db defaults to 0 if omitted. Warning: The exporter -runs SCAN to count the keys. This might not perform well on large databases.
-script	REDIS_EXPORTER_SCRIPT	Comma separated list of path(s) to Redis Lua script(s) for gathering extra metrics.
-debug	REDIS_EXPORTER_DEBUG	Verbose debug output
-log-format	REDIS_EXPORTER_LOG_FORMAT	Log format, valid options are txt (default) and json.
-namespace	REDIS_EXPORTER_NAMESPACE	Namespace for the metrics, defaults to redis.
-connection-timeout	REDIS_EXPORTER_CONNECTION_TIMEOUT	Timeout for connection to Redis instance, defaults to "15s" (in Golang duration format)
-web.listen-address	REDIS_EXPORTER_WEB_LISTEN_ADDRESS	Address to listen on for web interface and telemetry, defaults to 0.0.0.0:9121.
-web.telemetry-path	REDIS_EXPORTER_WEB_TELEMETRY_PATH	Path under which to expose metrics, defaults to /metrics.
-redis-only-metrics	REDIS_EXPORTER_REDIS_ONLY_METRICS	Whether to also export go runtime metrics, defaults to false.
-include-config-metrics	REDIS_EXPORTER_INCL_CONFIG_METRICS	Whether to include all config settings as metrics, defaults to false.
-include-system-metrics	REDIS_EXPORTER_INCL_SYSTEM_METRICS	是否包括 <code>total_system_memory_bytes</code> 等系统指标，默认为false。建议打开。
-redact-config-metrics	REDIS_EXPORTER_REDACT_CONFIG_METRICS	Whether to redact config settings that include potentially sensitive information like passwords.
-ping-on-connect	REDIS_EXPORTER_PING_ON_CONNECT	Whether to ping the redis instance after connecting and record the duration as a metric, defaults to false.
-is-tile38	REDIS_EXPORTER_IS_TILE	38 Whether to scrape Tile38 specific metrics, defaults to false.
-is-cluster	REDIS_EXPORTER_IS_CLUSTER	Whether this is a redis cluster (Enable this if you need to fetch key level data on a Redis Cluster).
-export-client-list	REDIS_EXPORTER_EXPORT_CLIENT_LIST	Whether to scrape Client List specific metrics, defaults to false.
-export-client-port	REDIS_EXPORTER_EXPORT_CLIENT_PORT	Whether to include the client's port when exporting the client list. Warning: including the port increases the number of metrics generated and will - make your Prometheus server take up more memory

Name	Environment Variable Name	Description
-skip-tls-verification	REDIS_EXPORTER_SKIP_TLS_VERIFICATION	Whether to skip TLS verification when the exporter connects to a Redis instance
-tls-client-key-file	REDIS_EXPORTER_TLS_CLIENT_KEY_FILE	Name of the client key file (including full path) if the server requires TLS client authentication
-tls-client-cert-file	REDIS_EXPORTER_TLS_CLIENT_CERT_FILE	Name of the client cert file (including full path) if the server requires TLS client authentication
-tls-server-key-file	REDIS_EXPORTER_TLS_SERVER_KEY_FILE	Name of the server key file (including full path) if the web interface and telemetry should use TLS
-tls-server-cert-file	REDIS_EXPORTER_TLS_SERVER_CERT_FILE	Name of the server certificate file (including full path) if the web interface and telemetry should use TLS
-tls-server-ca-cert-file	REDIS_EXPORTER_TLS_SERVER_CA_CERT_FILE	Name of the CA certificate file (including full path) if the web interface and telemetry should use TLS
-tls-server-min-version	REDIS_EXPORTER_TLS_SERVER_MIN_VERSION	Minimum TLS version that is acceptable by the web interface and telemetry when using TLS, defaults to TLS1.2 (supports - TLS1.0,TLS1.1,TLS1.2,TLS1.3).
-tls-ca-cert-file	REDIS_EXPORTER_TLS_CA_CERT_FILE	Name of the CA certificate file (including full path) if the server requires TLS client authentication
-set-client-name	REDIS_EXPORTER_SET_CLIENT_NAME	Whether to set client name to redis_exporter, defaults to true.
-check-key-groups	REDIS_EXPORTER_CHECK_KEY_GROUPS	Comma separated list of LUA regexes for classifying keys into groups. The regexes are applied in specified order to individual keys, and the group name is -generated by concatenating all capture groups of the first regex that matches a key. A key will be tracked under the unclassified group if none of the specified regexes matches it.
-max-distinct-key-groups	REDIS_EXPORTER_MAX_DISTINCT_KEY_GROUPS	Maximum number of distinct key groups that can be tracked independently per Redis database. If exceeded, only key groups with the highest -memory consumption within the limit will be tracked separately, all remaining key groups will be tracked under a single overflow key group.
-config-command	REDIS_EXPORTER_CONFIG_COMMAND	What to use for the CONFIG command, defaults to CONFIG.