redislabs
home of redis

# Monitoring Redis Enterprise with DataDog

## 09/13/2019

Presented by Chris Mague

redislabs
home of redis

# Agenda

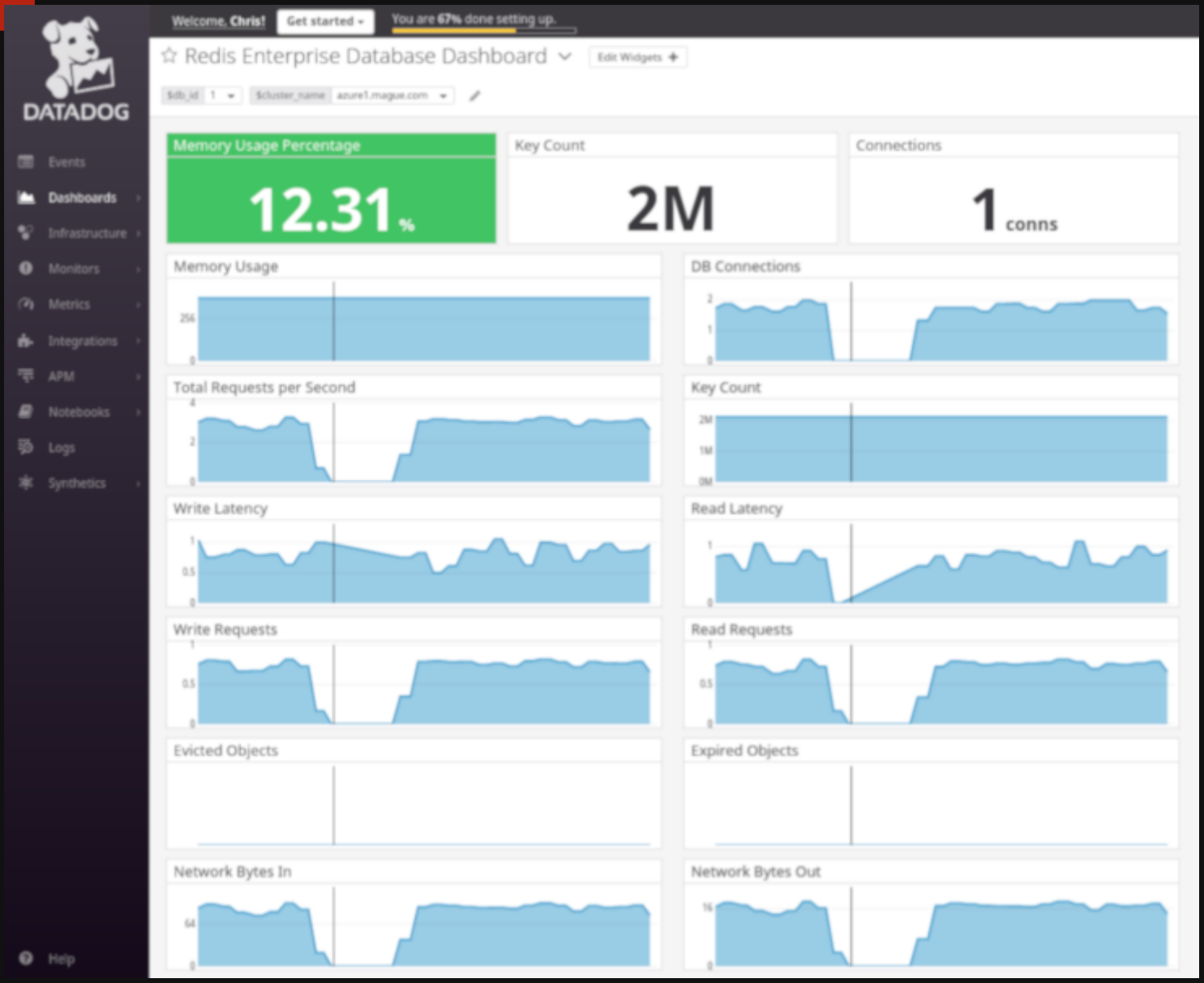
- End Goal
- Prerequisites
- Configuring Datadog for Programatic Access
- Setting up Datadog Agent
- Collecting data from Redis Enterprise
- Configuring Metadata
- Building dashboards
- Adding monitors
- Collected Data
- Example Monitors

redislabs
home of redis

# Questions we want to answer

- How full are my databases?
- What's the latency of my queries?
- Are clients connected?
- Are keys being evicted from my database?

redislabs
home of redi



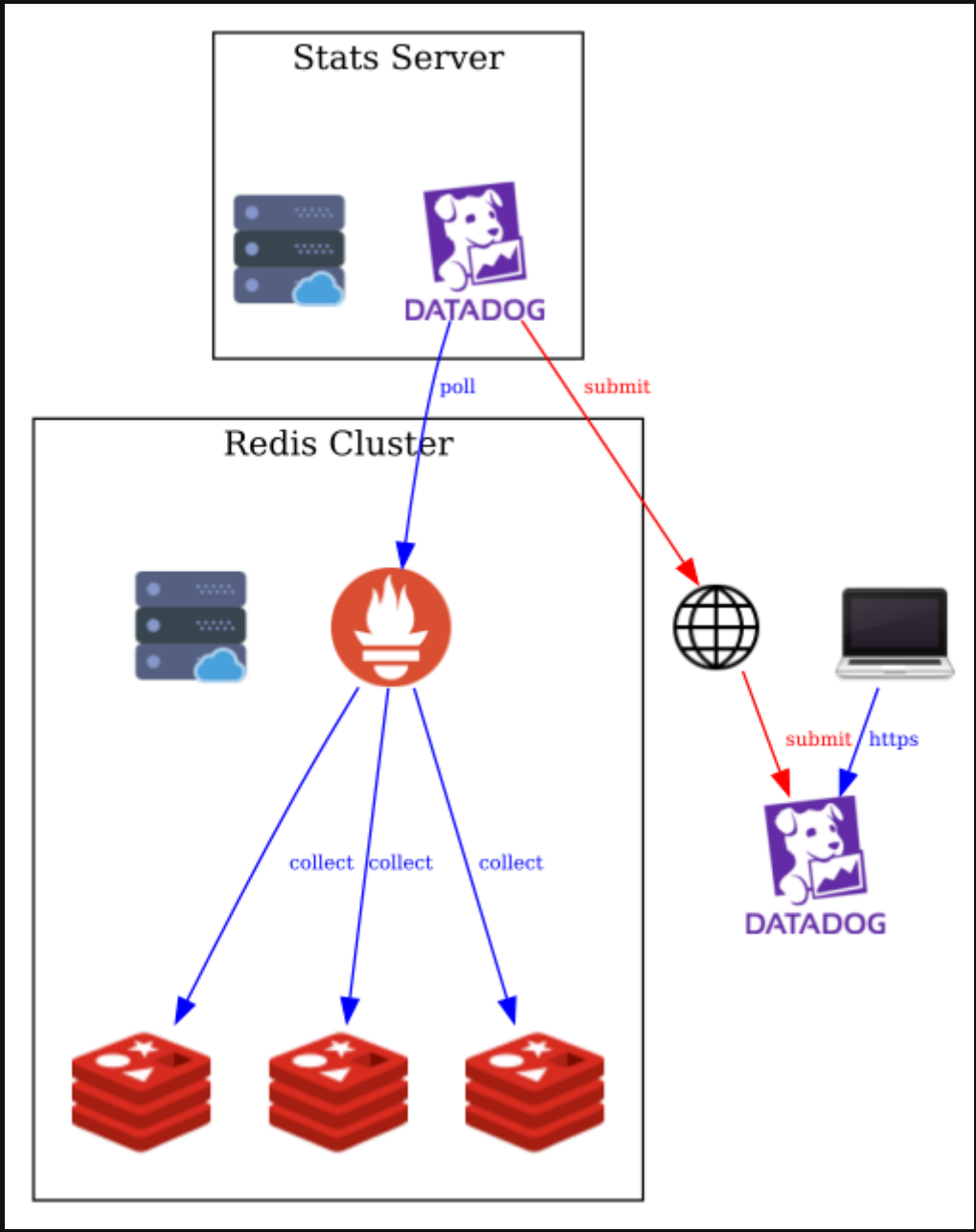
redislabs
home of redis

tag:"redise"

Hide Controls | Showing all 4 results

| Status | | |
|---|---|---|
| Triggered | | 0 |
| Alert | | 0 |
| Warn | | 0 |
| No Data | | 0 |
| OK | | 4 |

| | STATUS | NAME ↑ | TAGS |
|---|---|---|---|
| | OK | Redis Enterprise Eviction Alert | redise |
| | OK | Redis Enterprise Latency Alert | redise |
| | OK | Redis Enterprise Usage Alert | redise |
| | OK | Redis Enterprise has no connections | redise |


Stats Server
DATADOG
poll
submit
Redis Cluster
submit
https
collect
collect
collect
DATADOG

redislabs
home of redis

# Prerequisites

| Requirement | Usage |
|---|---|
| Datadog API Key | Needed to send data to Datadog |
| Datadog Application Key | Needed to create dashboards and alerts |
| Terraform | for automatically configuring the Datadog dashboards and metadata |
| Ansible | (optional) for automatically configuring the Datadog agent |

redislabs
home of redis

# Datadog Config

We need to create API and Application keys for programatic access

https://app.datadoghq.com/account/settings#api

redislabs
home of redis

# Datadog Config

Welcome, **Chris!**

DATADOG

- New Stuff!
- Events
- Dashboards
- Infrastructure
- Monitors
- Metrics
- Integrations
- APM
- Notebooks
- Logs
- Synthetics
- Help

Integrations | APIs | Agent | Embeds

## API Keys

Your **API keys** are unique to your organization. An API key is required by the **Datadog Agent** to submit metrics and events to Datadog.

| Name | Key | Created by | Created at (UTC) | |
|---|---|---|---|---|
| | [redacted] | chris.mague@[redacted] | 2019-08-16 17:03:56 | Revoke |

**New API key**

API key name | Create API key

** Your org must have at least one key and at most 5 keys*

## Application Keys

**Application keys** , in conjunction with your org's API key, give you **full access to Datadog's programmatic API** . Application keys are associated with the user account that created them and can be named. The application key is used to log all requests made to the API.

| Name | Key | Created by | |
|---|---|---|---|
| terraform | [redacted] | chris.mague@[redacted] | Revoke |

**New application key**

App key name | Create Application Key

redislabs
home of redis

# Setting up the Datdog Agent

- The datadog agent needs to be installed on a node preferably outside of the cluster
- Can be done manually with a bash script
- Also automated via Ansible, Puppet or Chef

redislabs
home of redis

# Datadog Config with Ansible

```yaml
- hosts: all
  become: yes
  become_user: root
  become_method: sudo
  gather_facts: yes
  vars_files:
    - vars/main.yml

  handlers:
    - name: restart_datadog_agent
      service:
        name: datadog-agent
        state: restarted

  pre_tasks:
    - name: Update Apt Cache
      apt: update_cache=yes cache_valid_time=86400
      when: ansible_os_family == "Debian"

  roles:
    - Datadog.datadog
```

redislabs
home of redis

# Datadog agent

- Configuration files
  - /etc/datadog-agent/datadog.yaml (main config - contains API key)
  - /etc/datadog-agent/conf.d/ (integrations)
- Processes
  - stop: sudo systemctl stop datadog-agent
  - start: sudo systemctl start datadog-agent
  - status: sudo systemctl status datadog-agent

redislabs
home of redis

# Collecting Data from Redis Enterprise Manually

- Configuration files
  - /etc/datadog-agent/conf.d/prometheus.d/conf.yaml
- Processes
  - restart: sudo systemctl restart datadog-agent

redislabs
home of redis

# Collecting Data from Redis Enterprise Automated

```yaml
handlers:
  - name: restart_datadog_agent
    service:
      name: datadog-agent
      state: restarted

post_tasks:
  - name: Setup Prometheus Scraper Config for Datadog
    template:
      src: prometheus_conf.yaml.j2
      dest: /etc/datadog-agent/conf.d/prometheus.d/conf.yaml
      owner: root
      group: root
      mode: 0644
    notify:
      - restart_datadog_agent
```

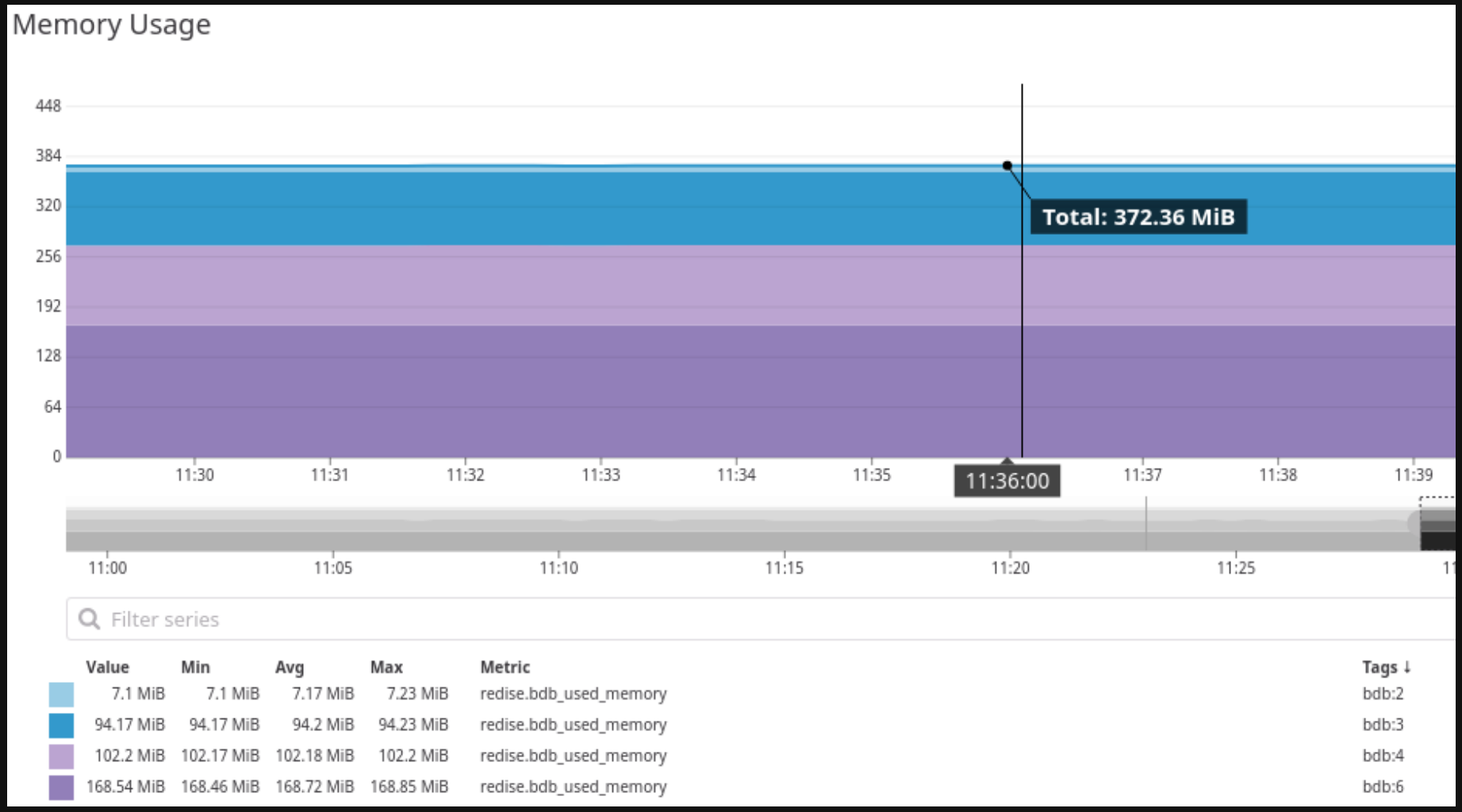
Template is available for download here

redislabs
home of redis

# Prometheus Config

```
- prometheus_url: http://re.example.com:8070/metrics
  ssl_ca_cert: false
  namespace: redise
  max_returned_metrics: 2000
  metrics:
    - bdb_avg_latency
    - bdb_avg_latency_max
    - bdb_avg_other_latency
    - bdb_avg_read_latency
    - listener_acc_latency
    - listener_conns
    ...
```

# Why do we need Metadata?




that's better than 7.1 e6

redislabs
home of redis

# Configuring Metadata with Terraform

```
resource "datadog_metric_metadata" "bdb_used_memory" {
  metric      = "redise.bdb_used_memory"
  short_name  = "Redis Enterprise Database Used Memory"
  description = "Amount of memory in use for the database"
  type        = "gauge"
  unit        = "byte"
}
```

# Configuring Dashboards with Terraform

```
provider "datadog" {
  api_key = "${var.datadog_api_key}"
  app_key = "${var.datadog_app_key}"
}

resource "datadog_dashboard" "dbd_dashboard" {
  title         = "Redis Enterprise Database Dashboard"
  description   = "Created using the Datadog provider in Terraform"
  layout_type   = "free"
  is_read_only  = true

  widget {
     query_value_definition{
      request {
        q = "(avg:redise.bdb_used_memory{$db_id,$cluster_name} by {bdb}/avg:redise.bdb_memory_limit{$c
        conditional_formats {
          comparator = "<"
          value = "80"
          palette = "white_on_green"
        }
        conditional_formats {
          comparator = ">"
          value = "80"
          palette = "white_on_red"
        }
```

# Configuring Monitors

```
resource "datadog_monitor" "no_conns" {
  name               = "Redis Enterprise has no connections"
  type               = "metric alert"
  message            = "There are no connections the the Redis Enterprise Database
  escalation_message = "Some escalation notification goes here"

  query = "avg(last_15m):avg:redise.bdb_conns{*} by {bdb} < 1"

  thresholds = {
    ok                = 0
    warning           = 2
    warning_recovery  = 4
    critical          = 1
    critical_recovery = 3
  }
  notify_no_data    = false
  renotify_interval = 60
  notify_audit      = false
  timeout_h         = 60
  include_tags      = true
  tags              = ["redise"]
}
```

redislabs
home of redis

# Using terraform to create dashboards, monitors, and metadata

```
cd terraform
cp dashboard.tfvars.example dashboard.tfvars

# modify dasboard.tfvars to your API/App keys

terraform init
terraform apply -var-file=dashboard.tfvars
```

redislabs
home of redis

# Collected Data - Latency

| metric | definition |
|---|---|
| bdb_avg_latency | average latency of all requests |
| bdb_avg_latency_max | maximum latency |
| bdb_avg_read_latency | average latency of read requests |
| bdb_avg_write_latency | average latency of write requests |
| bdb_avg_other_latency | average latency of other requests |

Latency is measure from time the request hits the proxy until the response is sent back over the network

redislabs
home of redis

# Collected Data - Requests

| metric | definition |
|---|---|
| bdb_write_req | number of write operations per database |
| bdb_read_req | number of read operations per database |
| bdb_other_req | number of read operations per database |
| bdb_total_req | number of total operations per database |
| bdb_total_req_max | maximum number of total operations per database |

redislabs
home of redis

# Collected Data - Keys

| metric | definition |
|---|---|
| bdb_no_of_keys | number of keys in the database |
| bdb_expired_objects | number of keys that have exceeded their TTL |
| bdb_evicted_objects | number of keys that have been pushed out to make room for new keys |

redislabs
home of redis

# Collected Data - Resources

| metric | definition |
|---|---|
| bdb_used_memory | Amount of memory used by the DB |
| bdb_ingress_bytes | Network traffic coming into the DB |
| bdb_egress_bytes | Network traffic coming out the DB |
| bdb_fork_cpu_system | % cores utilization in system mode for all redis shard fork child processes of this database |
| bdb_main_thread_cpu_system | % cores utilization in system mode for all redis shard main threas of this database |
| bdb_main_thread_cpu_system_max | maximum % cores utilization in system mode for all redis shard main threas of this database |

# Example Monitors

## No Active Database Connections

```
"avg(last_15m):avg:redise.bdb_conns{*} by {bdb} < 1"
```

## High Database Request Latency (10ms)

```
"avg(last_15m):avg:redise.bdb_avg_latency{*} by {bdb} > 0.0
```

## Database Key Eviction

```
"avg(last_5m):avg:redise.bdb_evicted_objects{*} by {bdb} >
```

## Database Usage

```
"avg(last_5m):( avg:redise.bdb_used_memory{*} by {bdb}
 / avg:redise.bdb_memory_limit
{*} by {bdb} ) * 100 > 85"
```

redislabs
home of redis

# Thank you

Contact: Chris Mague - christian@redislabs.com