Redis

19 Monitoring

$ redis-cli monitor

```
➜  ~ docker exec -it d4b14a834ba7 redis-cli monitor
OK
1682871818.259467 [0 127.0.0.1:54426] "HGETALL" "user:300"
1682871820.037704 [0 127.0.0.1:54426] "KEYS" "*"
1682871828.763765 [0 127.0.0.1:54426] "GET" "mylist"
1682871831.609374 [0 127.0.0.1:54426] "TYPE" "mylist"
1682871853.119967 [0 127.0.0.1:54426] "LRANGE" "mylist" "0" "-1"
```

$ redis-cli --stat

```
➜  ~ docker exec -it d4b14a834ba7 redis-cli --stat
------- data ------ -------------------- load -------------------- - child -
keys       mem      clients blocked requests            connections
15         1.93M    6       0       4134030 (+0)        2147
15         1.93M    6       0       4134031 (+1)        2147
15         1.93M    6       0       4134032 (+1)        2147
15         1.93M    6       0       4134033 (+1)        2147
15         1.93M    6       0       4134034 (+1)        2147
15         1.93M    6       0       4134035 (+1)        2147
15         1.93M    6       0       4134036 (+1)        2147
15         1.93M    6       0       4134037 (+1)        2147
15         1.93M    6       0       4134038 (+1)        2147
15         1.93M    6       0       4134039 (+1)        2147
15         1.93M    6       0       4134040 (+1)        2147
15         1.93M    6       0       4134041 (+1)        2147
15         1.93M    6       0       4134042 (+1)        2147
```

$ redis-cli --bigkeys

```
➜  ~ docker exec -it d4b14a834ba7 redis-cli --bigkeys

# Scanning the entire keyspace to find biggest keys as well as
# average sizes per key type.  You can use -i 0.1 to sleep 0.1 sec
# per 100 SCAN commands (not usually needed).

[00.00%] Biggest set    found so far '"user"' with 3 members
[00.00%] Biggest list   found so far '"mylist"' with 200000 items
[00.00%] Biggest string found so far '"counter:__rand_int__"' with 6 bytes
[00.00%] Biggest hash   found so far '"user:100"' with 3 fields

-------- summary -------

Sampled 15 keys in the keyspace!
Total key length in bytes is 217 (avg len 14.47)

Biggest   list found '"mylist"' has 200000 items
Biggest   hash found '"user:100"' has 3 fields
Biggest string found '"counter:__rand_int__"' has 6 bytes
Biggest    set found '"user"' has 3 members

1 lists with 200000 items (06.67% of keys, avg size 200000.00)
4 hashs with 10 fields (26.67% of keys, avg size 2.50)
3 strings with 12 bytes (20.00% of keys, avg size 4.00)
0 streams with 0 entries (00.00% of keys, avg size 0.00)
7 sets with 9 members (46.67% of keys, avg size 1.29)
0 zsets with 0 members (00.00% of keys, avg size 0.00)
```

$ redis-cli --memkeys

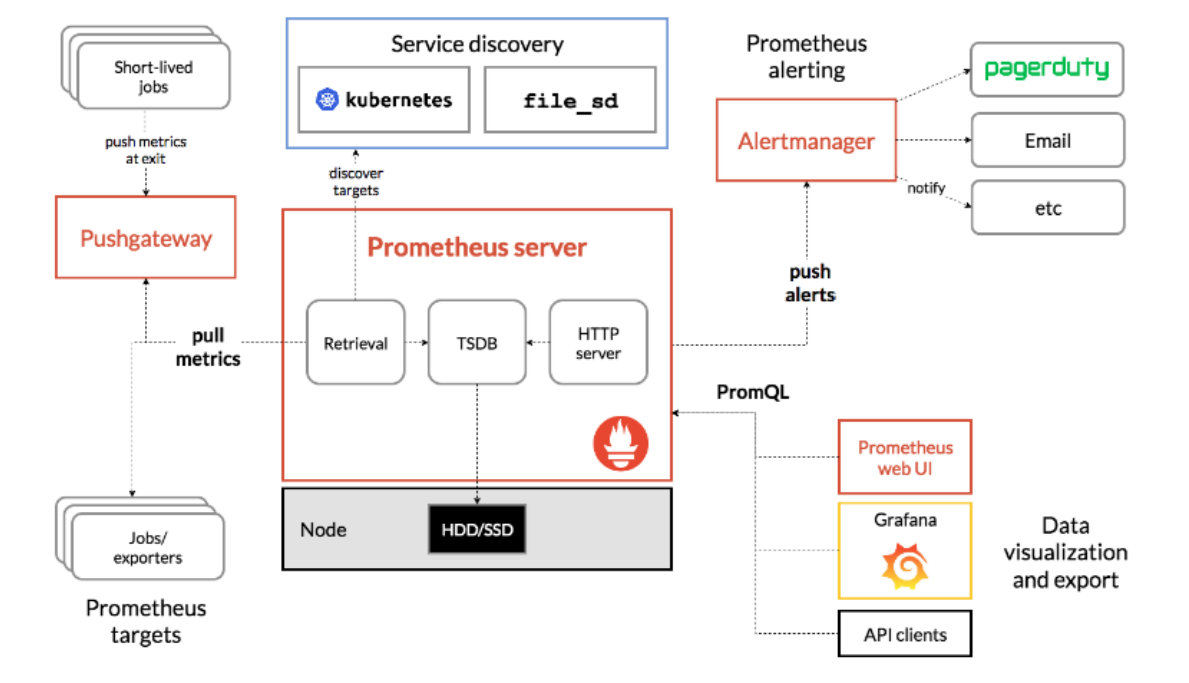
```
➜  ~ docker exec -it 9e25cfa12f59 redis-cli --memkeys

# Scanning the entire keyspace to find biggest keys as well as
# average sizes per key type.  You can use -i 0.1 to sleep 0.1 sec
# per 100 SCAN commands (not usually needed).

[00.00%] Biggest string found so far '"KEY2"' with 48 bytes
[00.00%] Biggest string found so far '"key:__rand_int__"' with 69 bytes
[00.00%] Biggest hash   found so far '"myhash"' with 86 bytes
[00.00%] Biggest list   found so far '"mylist"' with 860473 bytes

-------- summary -------

Sampled 7 keys in the keyspace!
Total key length in bytes is 60 (avg len 8.57)

Biggest   list found '"mylist"' has 860473 bytes
Biggest   hash found '"myhash"' has 86 bytes
Biggest string found '"key:__rand_int__"' has 69 bytes

1 lists with 860473 bytes (14.29% of keys, avg size 860473.00)
1 hashs with 86 bytes (14.29% of keys, avg size 86.00)
5 strings with 277 bytes (71.43% of keys, avg size 55.40)
0 streams with 0 bytes (00.00% of keys, avg size 0.00)
0 sets with 0 bytes (00.00% of keys, avg size 0.00)
0 zsets with 0 bytes (00.00% of keys, avg size 0.00)
```

$ redis-cli --latency

```
➜  ~ docker exec -it 9e25cfa12f59 redis-cli --latency
min: 0, max: 7, avg: 0.38 (472 samples)
```

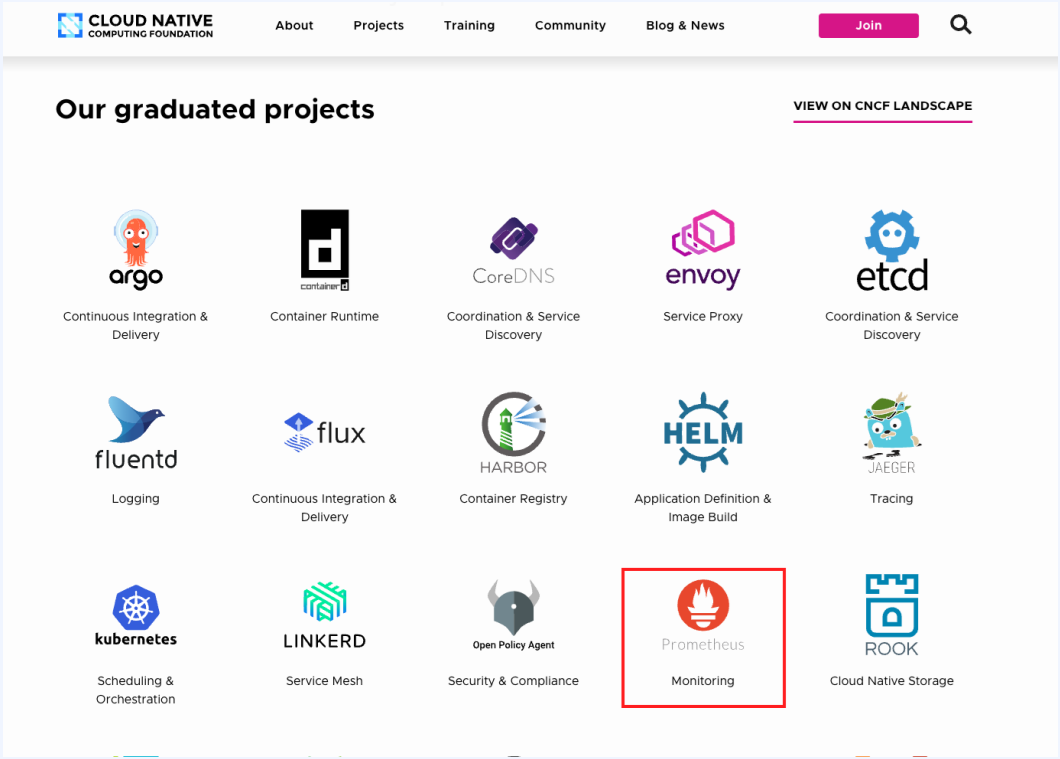
Prometheus / Grafana

Monitoring



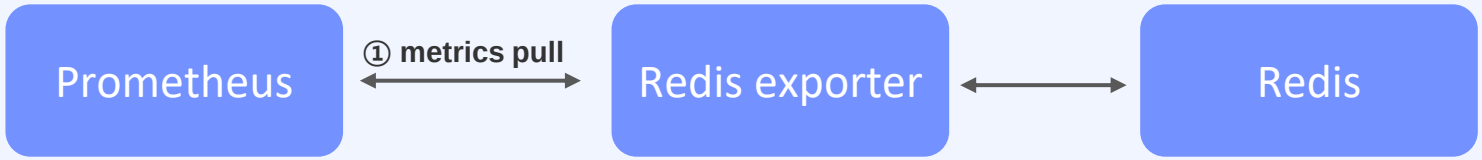
https://prometheus.io/docs/introduction/overview/

Monitoring

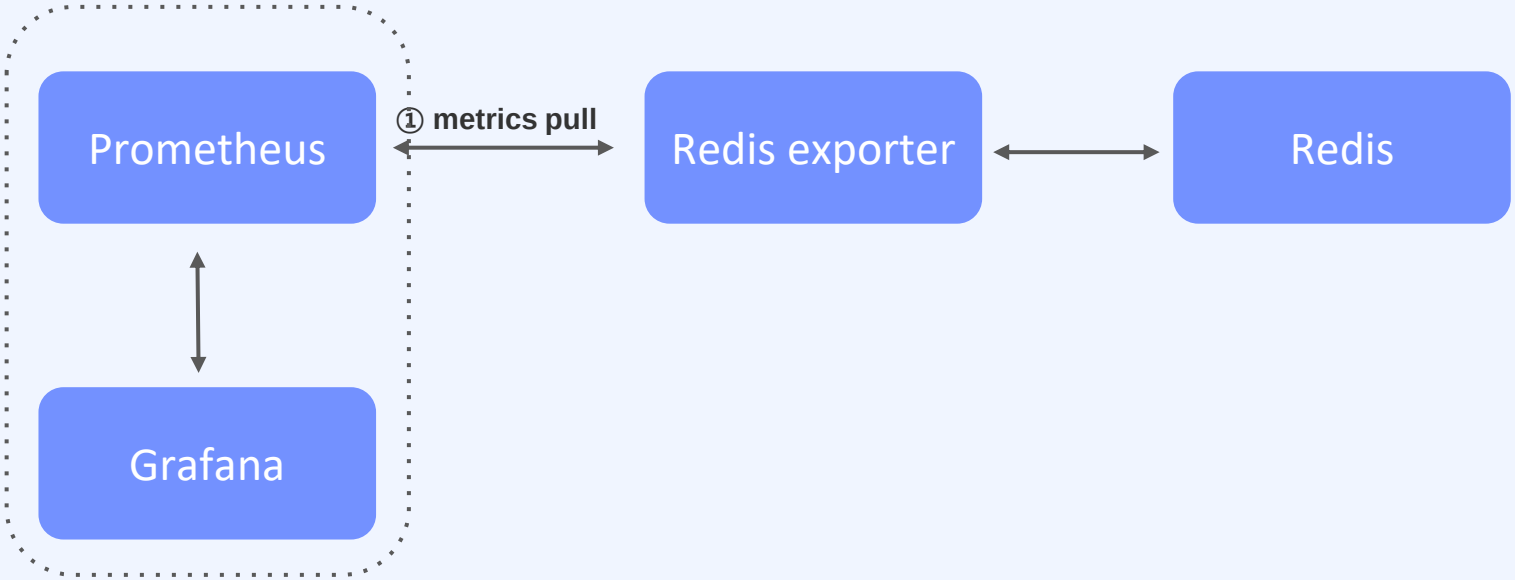


https://www.cncf.io/projects/

Monitoring



Monitoring



Monitoring

Memory Eviction

Monitoring

1. maxmemory
    a. **maxmemory 0 (default)**
2. maxmemory-policy
    a. **noeviction (default)**
    b. allkeys-lru
    c. allkeys-lfu
    d. volatile-lru
    e. volatile-lfu
    f. allkeys-random
    g. volatile-random
    h. volatile-ttl

Monitoring

1. maxmemory
    a. maxmemory 4G
2. maxmemory-policy
    a. noeviction
    b. allkeys-lru (least recently used)
    c. allkeys-lfu (least frequently used)
    d. volatile-lru
    e. volatile-lfu
    f. allkeys-random
    g. volatile-random
    h. volatile-ttl

Monitoring

1. maxmemory
    a. maxmemory 4G
2. maxmemory-policy
    a. noeviction
    b. allkeys-lru (least recently used)
    c. allkeys-lfu (least frequently used)
    d. volatile-lru
    e. volatile-lfu
    f. allkeys-random
    g. volatile-random
    h. volatile-ttl

Monitoring

1. maxmemory
    a. maxmemory 4G
2. maxmemory-policy
    a. noeviction
    b. allkeys-lru (least recently used)
    c. allkeys-lfu (least frequently used)
    d. volatile-lru
    e. volatile-lfu
    f. allkeys-random
    g. volatile-random
    h. volatile-ttl

Monitoring

1. maxmemory
    a. maxmemory 4G
2. maxmemory-policy
    a. noeviction
    b. allkeys-lru (least recently used)
    c. allkeys-lfu (least frequently used)
    d. volatile-lru
    e. volatile-lfu
    f. allkeys-random
    g. volatile-random
    h. volatile-ttl

실습