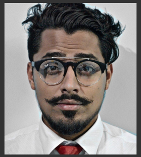
Connect.ONLINE

Automating Couchbase Monitoring

With Prometheus and Grafana Integrations Supplemented with Predictive Analytics



Ashish Rana
Data Engineering Specialist

Who will be benefited from this session ?

Industry professionals looking for scalable monitoring for multiple Couchbase VMs requiring minimal human interventions.

Key Takeaways from this Session

A Complete automation approach for monitoring solution

- Running integration tools as services
- In house developed tool orchestration
- Required customizations to these tools
- Total failure recovery and high availability
- Predictive maintenance and anomaly analysis

Where it all begins

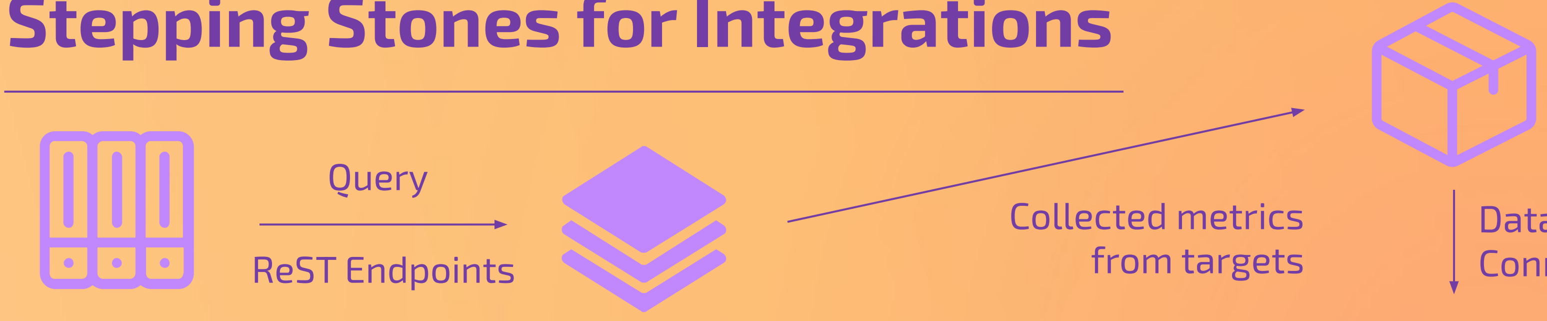
The Couchbase Blog

Couchbase Monitoring Integration with Prometheus and Grafana

by Karim Meliani

Metric Flow Sequence: Couchbase VM → CB-Exporter → Prometheus DB → Grafana Dashboards

Stepping Stones for Integrations



Query

ReST Endpoints

Collected metrics
from targets

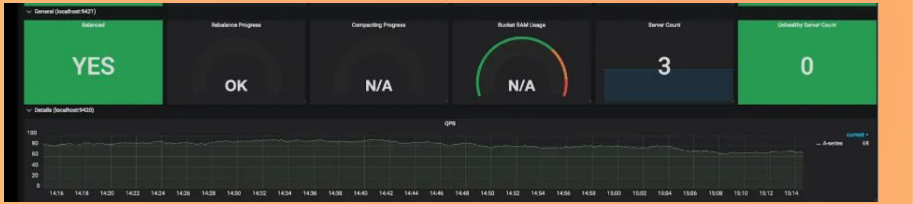
Prometheus
Database

Datasource
Connection

Couchbase
AWS EC2
Instances

Couchbase
Exporter

pools/default
[/bucket, node, tasks]



Steps Required for adding an Instance

- **Spinning up a couchbase exporter process for given instance to be monitored.**

```
Execute Command: $ ./couchbase-exporter --couchbase.username admin_user
--couchbase.password --web.listen-address=":9420" --couchbase.url="http://235.19.xx.xx:8091"
```

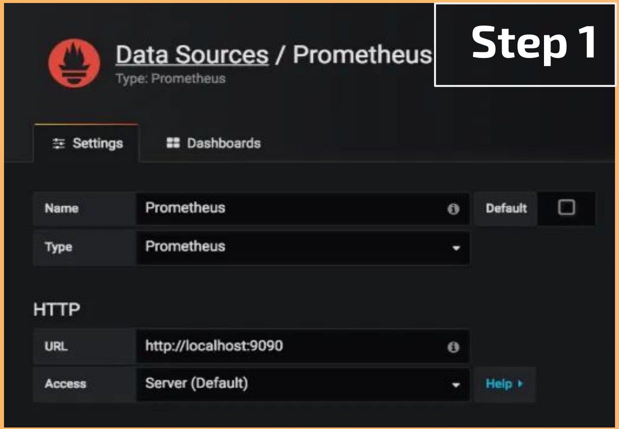
- **Adding scraping target instances in config file and running the prometheus instance.**

```
Inside configuration prometheus.yml file: - targets: [‘localhost:9420’, ‘localhost:9421’]
Execute Command: $ ./prometheus --config.file=prometheus.yml
```

- **Starting the grafana dashboard service & setting url for your Prometheus datasource.**

```
Execute Command: $ sudo service grafana-server start
Grafana UI Setup: Login → Settings → Data Sources → Set Prometheus datasource.
```

Grafana Data-Source Setup



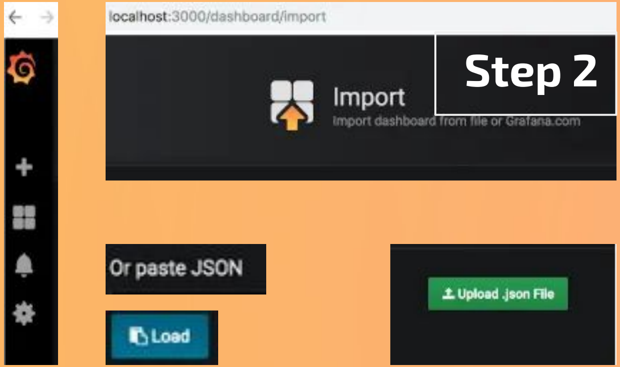
Step 1

Data Sources / Prometheus
Type: Prometheus

Name	Prometheus	Default	□
Type	Prometheus		

HTTP

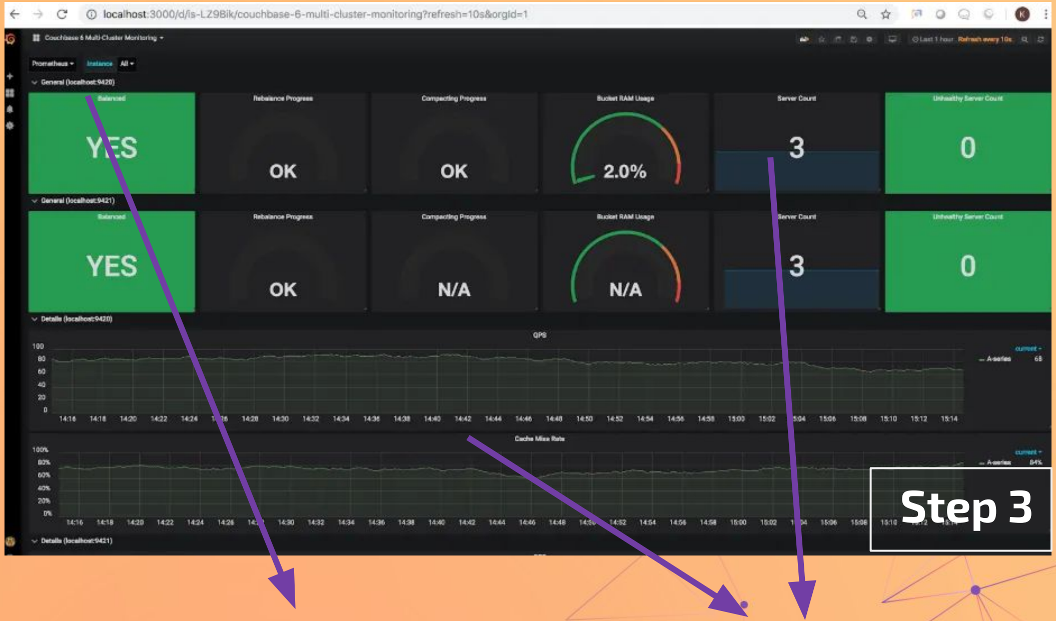
URL	http://localhost:9090	
Access	Server (Default)	Help ▸



localhost:3000/dashboard/import

Step 2

Import
Import dashboard from file or Grafana.com



localhost:3000/d/is-LZ9Bik/couchbase-6-multi-cluster-monitoring?refresh=10s&orgId=1

Step 3

General (localhost:9420)

Cluster Level Stats !!

Limitations

01 Manual Efforts required for new Couchbase VMs that limits scalability.

02 Combinations of tools not being in sync when new VM added.

03 No customizations for Grafana & CB-Exporter metrics captured.

04 Total system failure of monitoring solution & high availability.

05 Alerting, notifying and mitigation reactions for faulty instances.

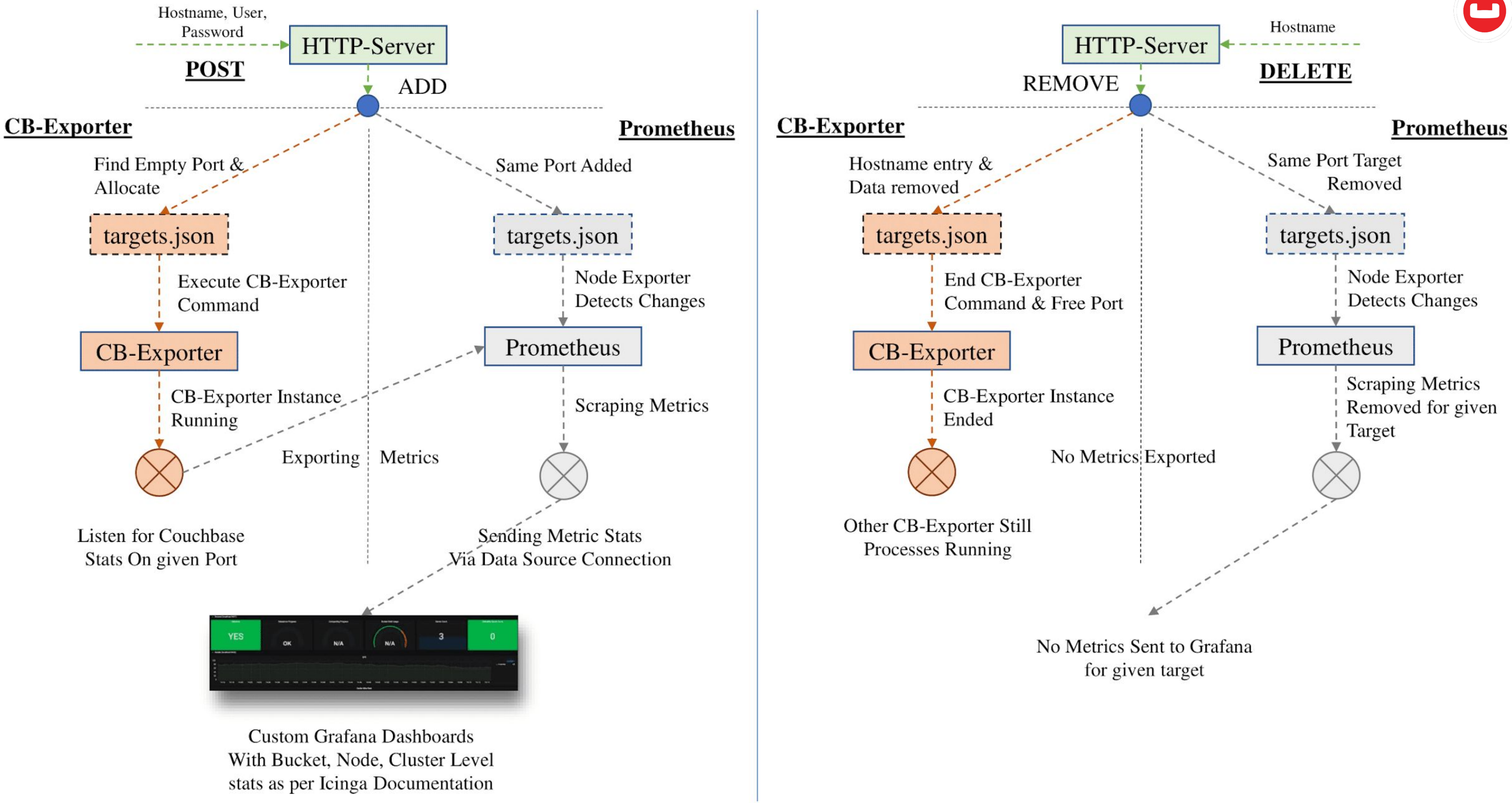
06 Predictive repairs & outlier analysis from metric data streams.

01

Automated Monitoring

ReST service architecture for adding Couchbase instances for monitoring

Cloud Automation: Prometheus Monitoring & Alerting Tool



Hostname, User, Password

POST

HTTP-Server

ADD

CB-Exporter

Prometheus

Find Empty Port & Allocate

targets.json

Execute CB-Exporter Command

CB-Exporter

CB-Exporter Instance Running

Listen for Couchbase Stats On given Port

Same Port Added

targets.json

Node Exporter Detects Changes

Prometheus

Scraping Metrics

Exporting Metrics

Sending Metric Stats Via Data Source Connection

Custom Grafana Dashboards With Bucket, Node, Cluster Level stats as per Icinga Documentation

Hostname

DELETE

HTTP-Server

REMOVE

CB-Exporter

Prometheus

Hostname entry & Data removed

targets.json

End CB-Exporter Command & Free Port

CB-Exporter

CB-Exporter Instance Ended

Other CB-Exporter Still Processes Running

Same Port Target Removed

targets.json

Node Exporter Detects Changes

Prometheus

Scraping Metrics Removed for given Target

No Metrics Exported

No Metrics Sent to Grafana for given target

02

Customizations

Add new features to Grafana dashboards, Prometheus database and Couchbase Exporter tool

3X CUSTOMIZATIONS

- **Adding new variables and graphs to observe metrics at bucket or node level.**

```
Query 1 : "label_values(couchbase_bucket_basicstats_dataused_bytes{cluster="$cluster"}, bucket)”
Query 2 : "label_values(couchbase_node_interestingstats_couch_spatial_data_size{cluster="$cluster"},node)"
```

- **Running Node Exporter for dynamic target analysis & AlertManager for managing alerts.**

```
- alert: CouchbaseNotBalanced
          expr: couchbase_cluster_balanced == 0 and couchbase_task_rebalance_progress == 0
          severity: critical
```

- **Creating your metrics to measure with different Couchbase ReST endpoints like /indexStatus**

Creating Go struct object for index metric ⟶ Then, coding Collector object and collector function for this metric as well. ⟶ Finally, adding index object to main.go file. ⟶ Build the exporter with *go* build.

03

High Availability & Total Recovery

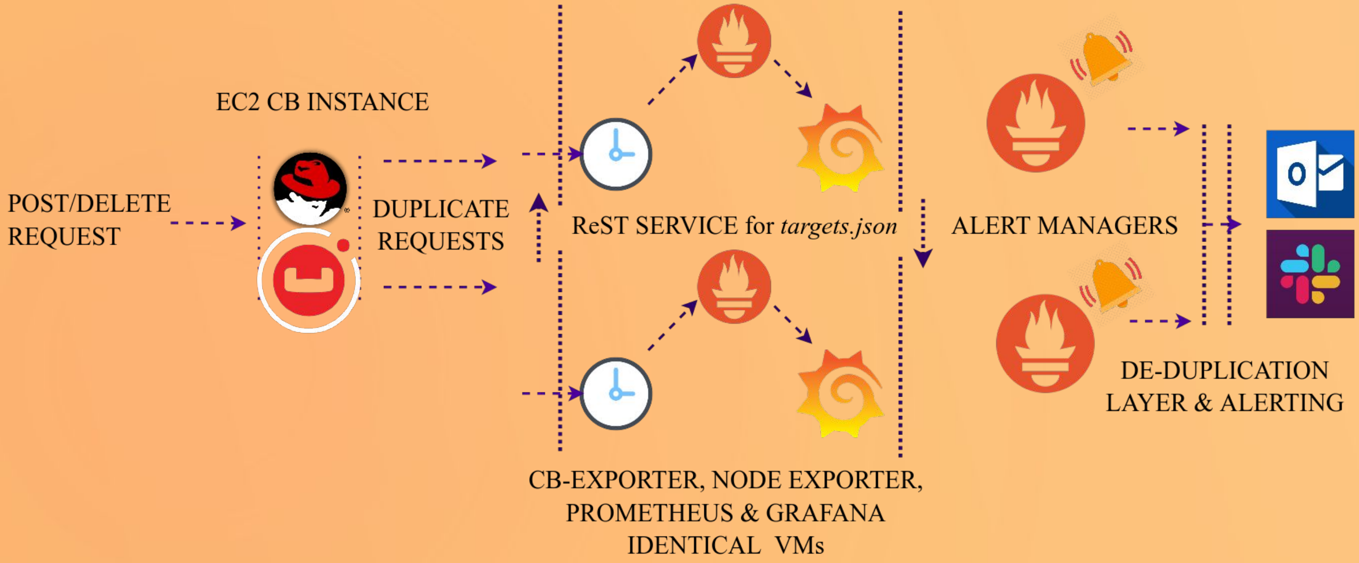
Prometheus runs as standalone instance on a VM.
Red Flag: What if that VM goes down or needs rebooting?

High Availability for Monitoring Solution

HIGH AVAILABILITY

CB-Exporter service duplicates request to both servers.

Both servers when live maintains same targets.json file.


EC2 CB INSTANCE
POST/DELETE REQUEST
DUPLICATE REQUESTS
ReST SERVICE for *targets.json*
ALERT MANAGERS
DE-DUPLICATION LAYER & ALERTING
CB-EXPORTER, NODE EXPORTER, PROMETHEUS & GRAFANA IDENTICAL VMs

Total Recovery Initiation

- CB-Exporter Recovery: Initiating CB-Exporter after Couchbase VM patching or Reboot.
- Prometheus, Grafana, Node Exporter and AlertManager Recovery: Initiating service after the reboot and starting the two ReST HTTP servers as well.

04

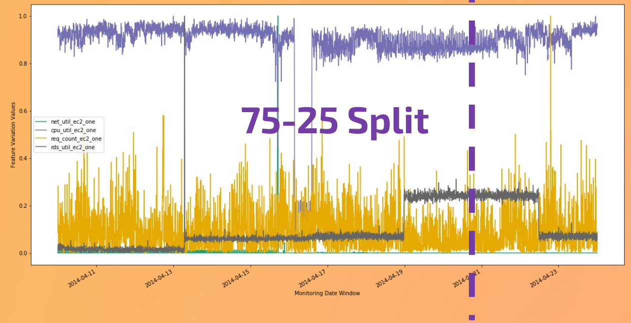
Predictive Maintenance

Analyzing stream of system health data Couchbase Instance & taking predictive actions with outlier analytics

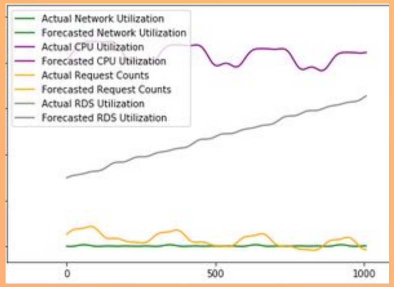
Predictive Maintenance from data streams

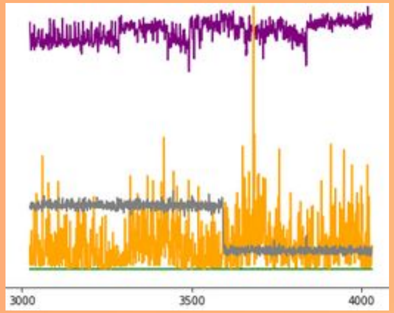
- **Unsupervised outlier detection analysis with *PyOD* on forecasted data with *fbprophet*.**

Couchbase VM

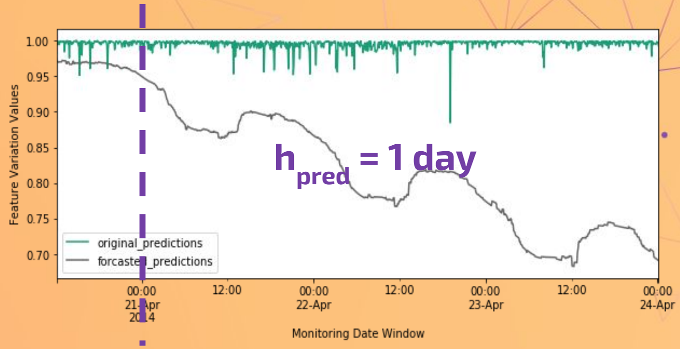


Step 1: Forecasting for CPU%, Request Counts, Network and DB consumption metrics.





Step 2: MAE evaluation for forecast model performance analysis.



Step 3: Sanity Checks on model predictions and deployment performance..

Exporting Prometheus Data

HTTP server scraping data with ReST API: /query resource endpoint. Beware, not to overload !!

Modelling Techniques

Forecasting: Additive models for each variable. Outlier analysis with LCSP: combine(LOF, LODA, CBLOF) models.

Important Resources

- *Karim Meliani's* article *'Couchbase Monitoring Integration with Prometheus and Grafana'*
- *Ashish Rana's* case study 'Dissecting the "*Couchbase Monitoring Integration with Prometheus & Grafana*" ' and all the discussed *artifacts.*
- Documentation for *Prometheus Database* & *AlertManager, mesh setup*; *Grafana dashboards PromQL queries* and *grafonnet*; *Couchbase Exporter package.*
- Documentation for */query API endpoints*, *PyOD Package* & *fbprophet* to proceed with predictive maintenanace implementations.

THANK YOU

Connect.ONLINE
DEVELOP YOUR PATH

Contact
prod.ashish.rana@gmail.com