M3 and Prometheus

Monitoring at Planet Scale for Everyone

Let’s talk...

Monitoring an increasing number of things…

Metrics being used as a platform more than ever...

Operating in many regions or environments…

M3 and Prometheus...

Who am I? All round monitoring nerd obsessed with graphs...

@robskillington

Uber Staff Software Engineer

Creator of M3DB

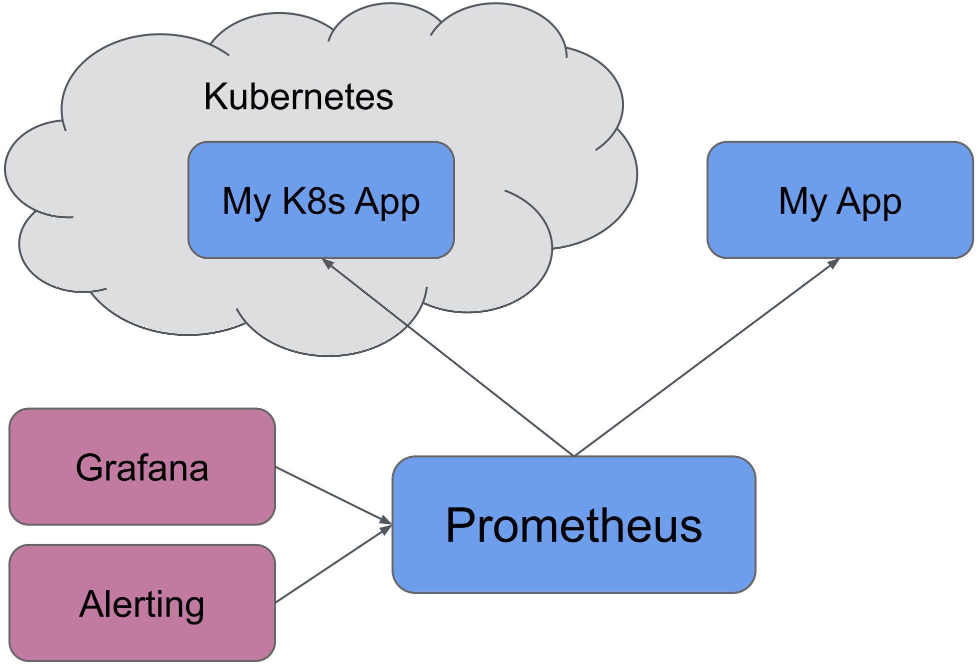
Member of OpenMetrics

What is Prometheus?

First built at SoundCloud (began 2014)

- An open source monitoring system and time series database.
- Essentially an industry standard for an all-in-one single node monitoring solution using metrics (explicitly not solving distributed storage of metrics).

What is Prometheus?



Single Region

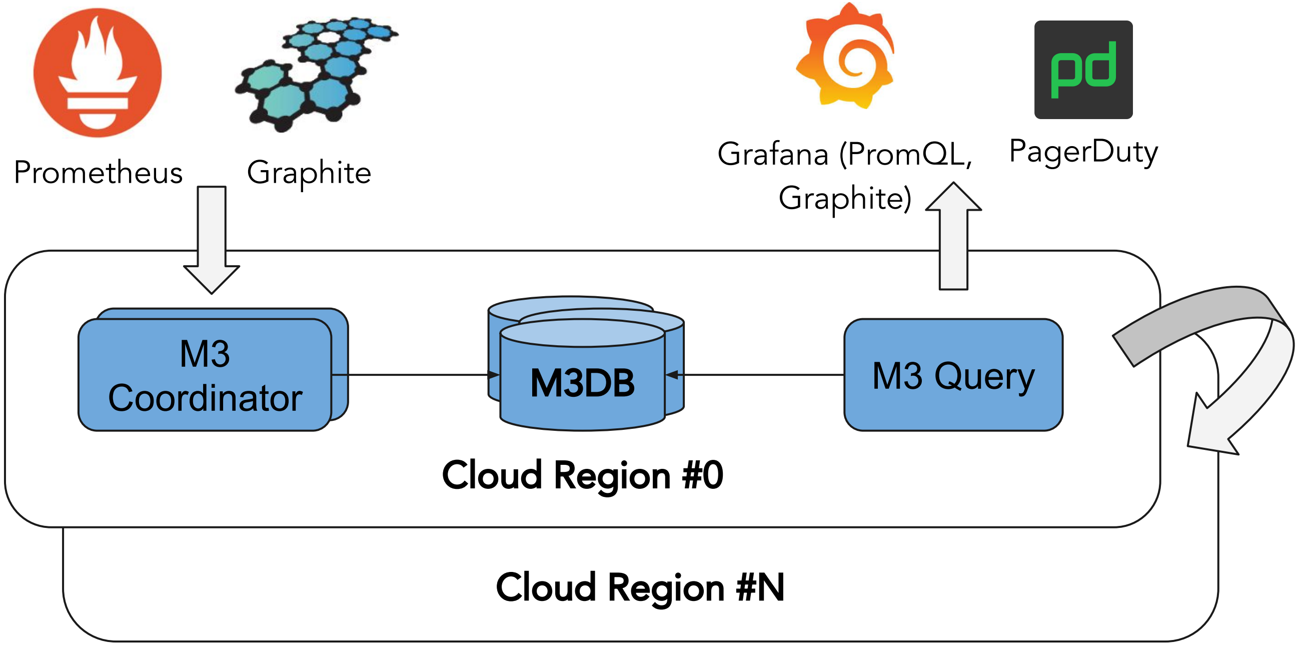
What is M3?

Built at Uber to scale monitoring horizontally and cost effective (began 2015)

- An open source monitoring system and distributed time series database, compatible with Prometheus as remote storage.
- First open source release in August 2018

What is M3?

- Monthly community meeting with attendees from small to large organizations
- Released every few weeks



1. Runs anywhere

2. Scalable to billions of metrics

3. Focus on simple operability

1. Runs anywhere

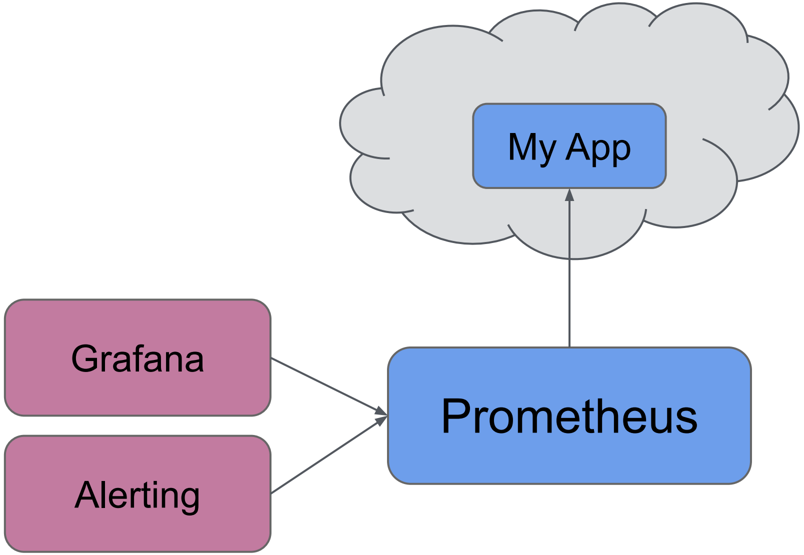
Cloud Native, Kubernetes or On Prem, Multi-Region, Prometheus and Graphite compatible

1. Runs anywhere

Why M3 and Prometheus

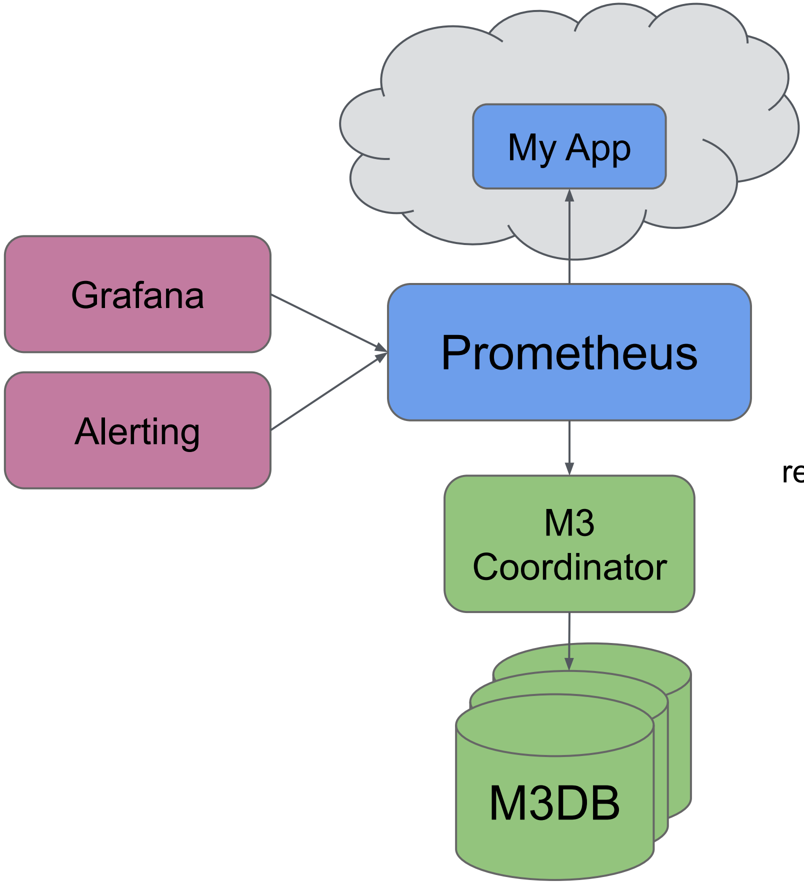
- Store metrics for weeks, months or years
- Store metrics at different retention based on mapping rules (e.g. app:nginx endpoints:/api*)
- Scale up storage just by adding nodes

Prometheus



Single Region

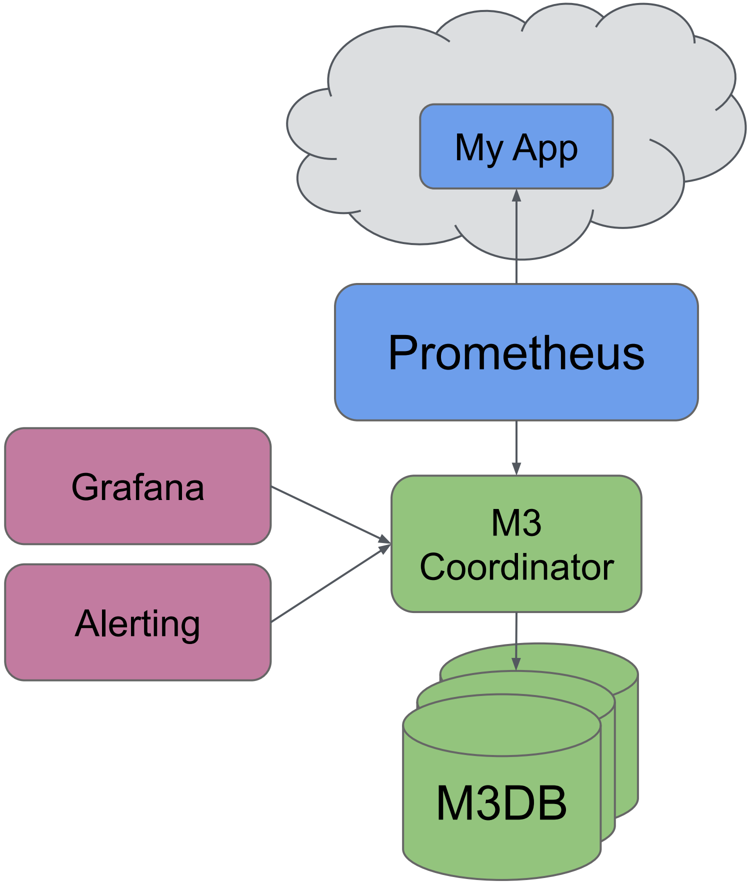
M3 and Prometheus (option 1)



Prometheus remote read and write with M3 Coordinator

Single Region

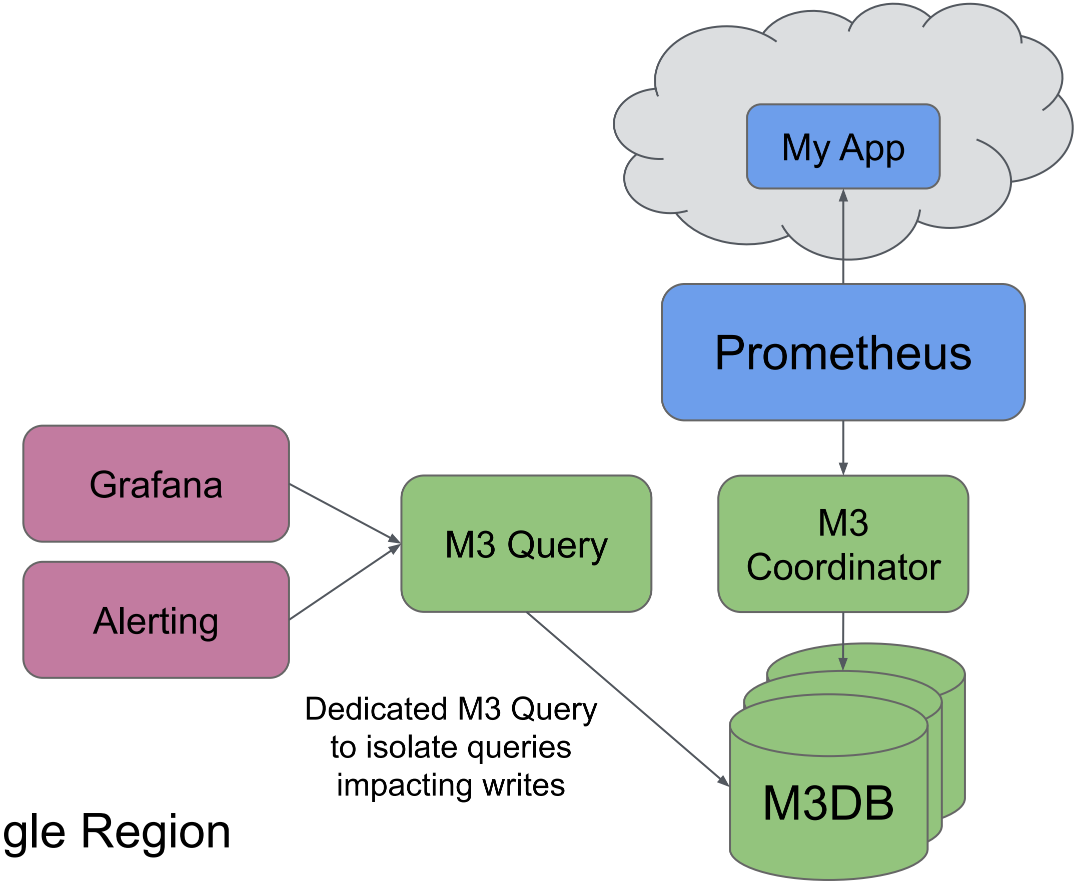
M3 and Prometheus (option 2)



Directly query M3 using coordinator for single Grafana datasource

Single Region

M3 and Prometheus (option 3)



Dedicated M3 Query
to isolate queries
impacting writes

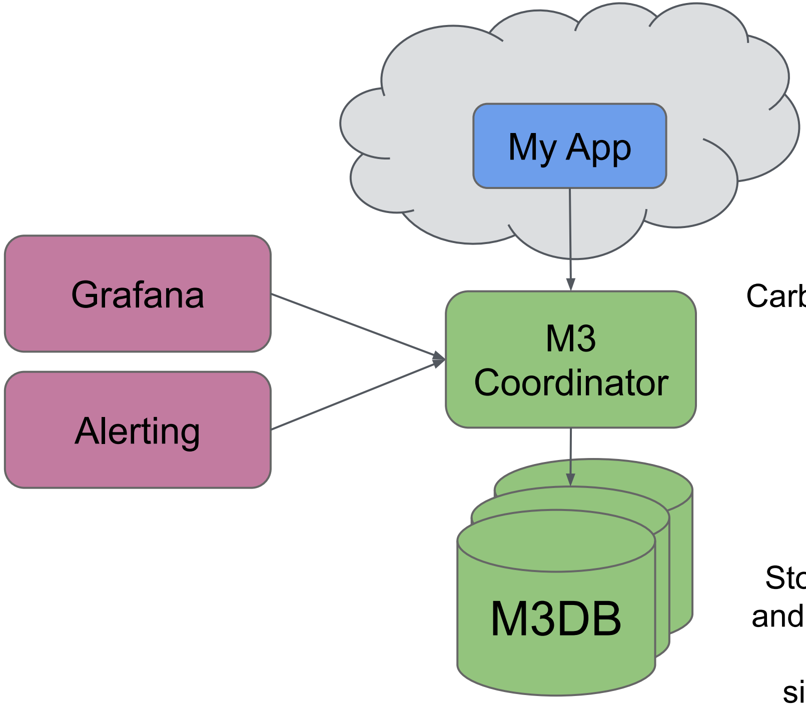
Single Region

1. Runs anywhere

M3 and Graphite

- Ingest: Carbon TCP
- Query: Graphite

M3 and Graphite



Carbon TCP Line Protocol

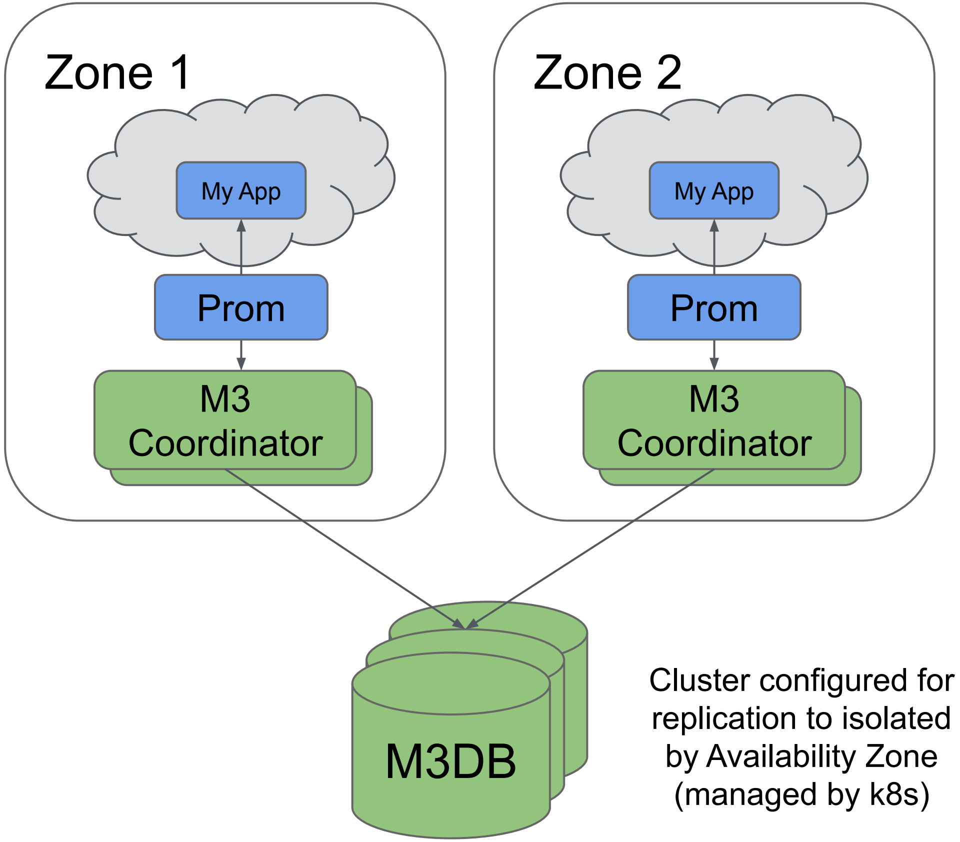
Store Graphite and Prometheus metrics side-by-side

1. Runs anywhere

M3 Multi-Region

- Global metrics collection and query
- Zero cross-region traffic
- Replication across Availability Zones as soon as metric collected

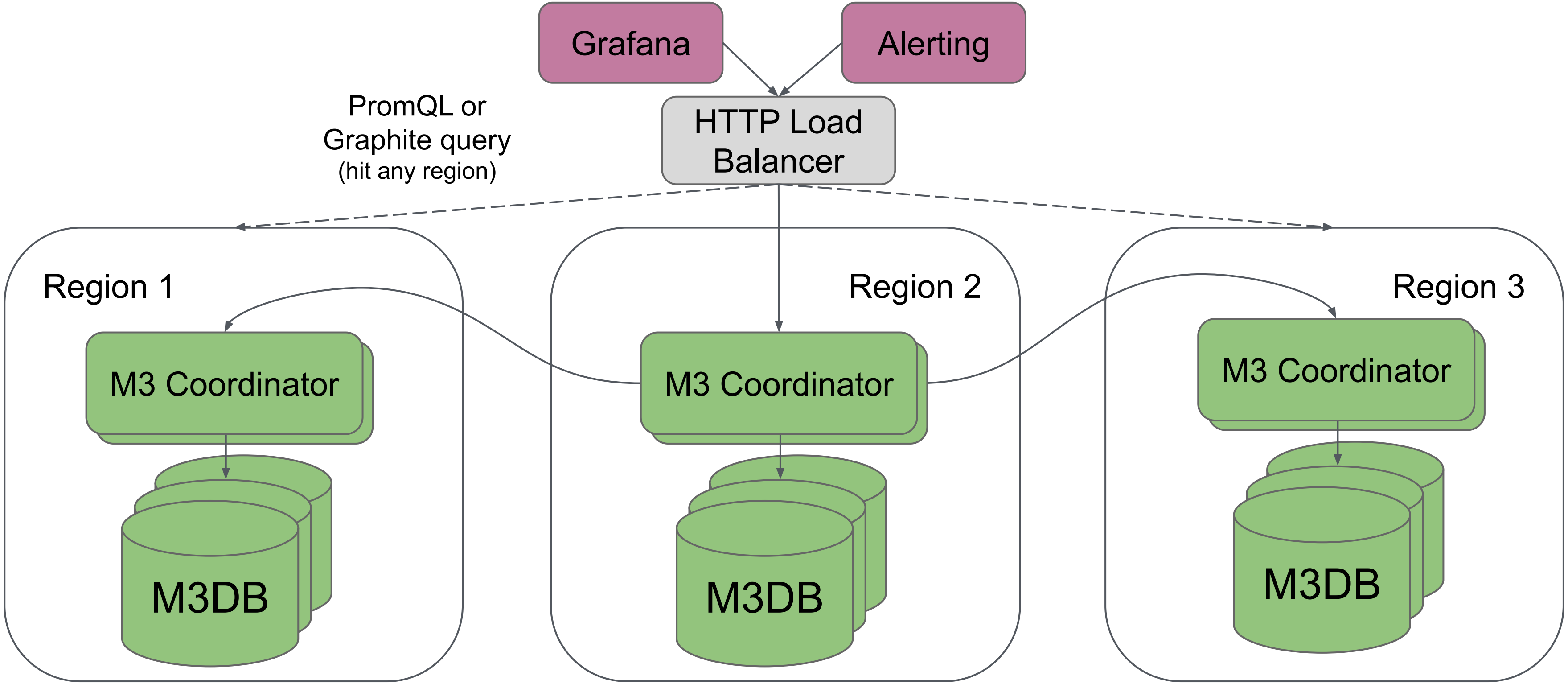
M3 Ingestion (Region Local)



Cluster configured for replication to isolated by Availability Zone (managed by k8s)

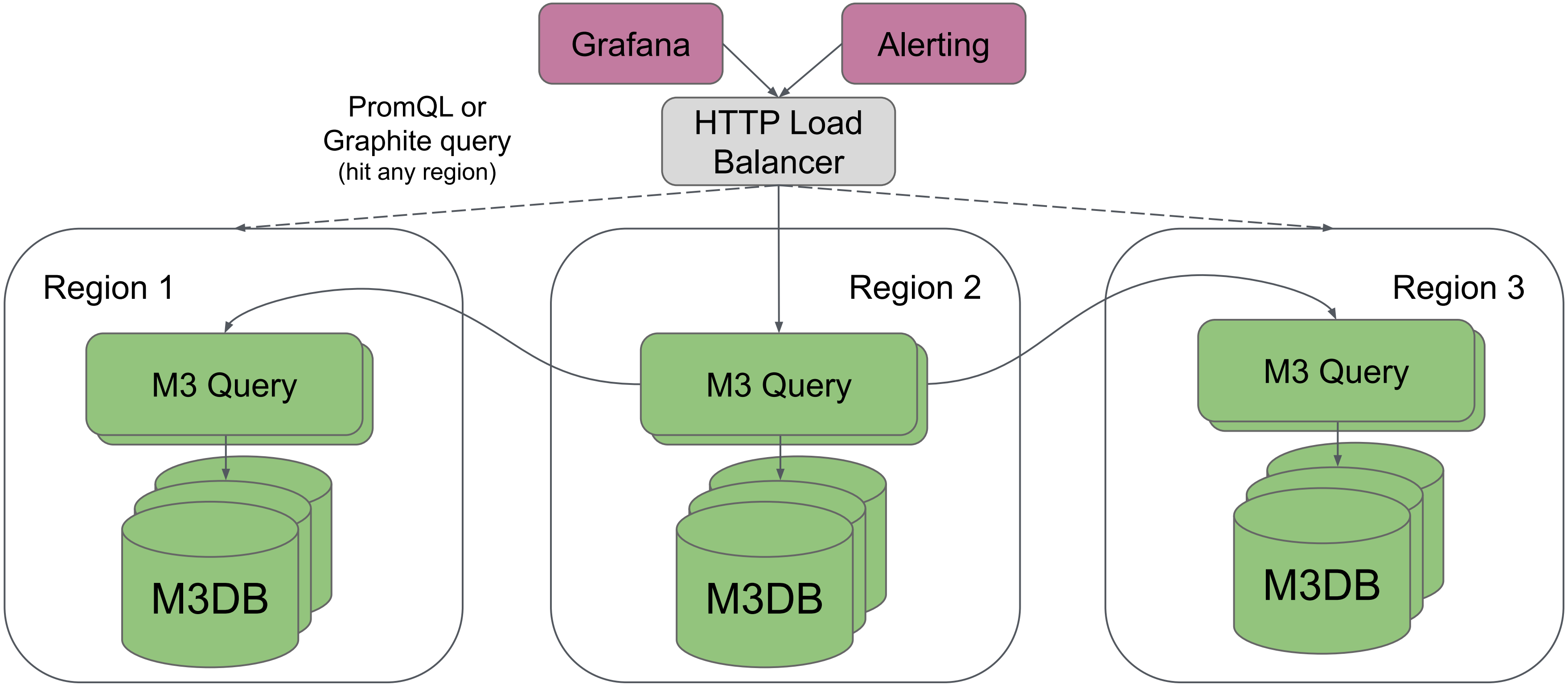
Single Region

M3 Queries (Global)



Multi-Region

M3 Queries (Global)



Multi-Region

2. Scalable to billions of metrics

2. Scalable to billions of metrics

M3 at Uber

- 4,000 plus microservices 🤣
- No onboarding to monitoring or provisioning of servers (just add storage nodes as required)

What’s it used for (and why are there so many metrics)

Used for all manner of things:

- Real-time alerting using application metrics (e.g., p99 response time)
- Tracking business metrics (e.g., number of Uber rides in Berlin)
  - Why? So easy to get started
  - *metrics.Tagged(Tags{“region”: “berlin”}).Counter(“ride_start”).Inc(1)*
- Network fabric bandwidth/latency and datacenter device temperatures
- Capacity planning for compute clusters and storage infrastructure (e.g., container load average, disk space in use, disk failure rate)
- And much more … load balancing Apache Helix based applications, etc

Workload

35M

Metrics stored per second

700M

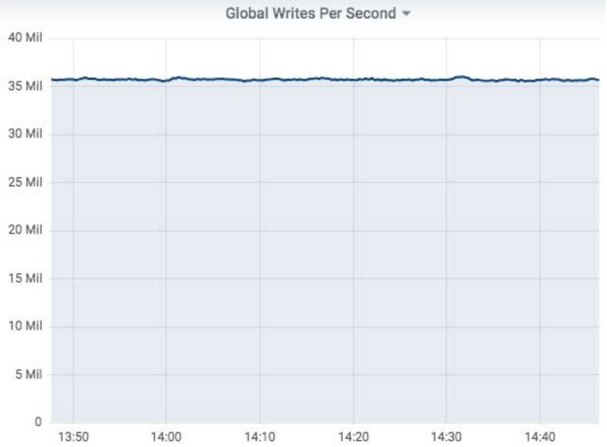
Metrics aggregated per second

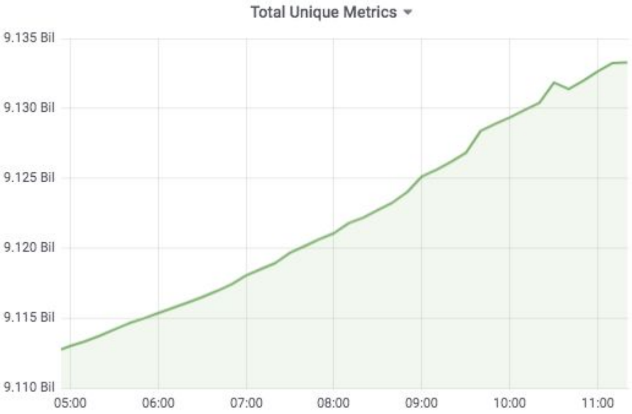
1000+

Instances running M3DB

9B

Unique Metric IDs





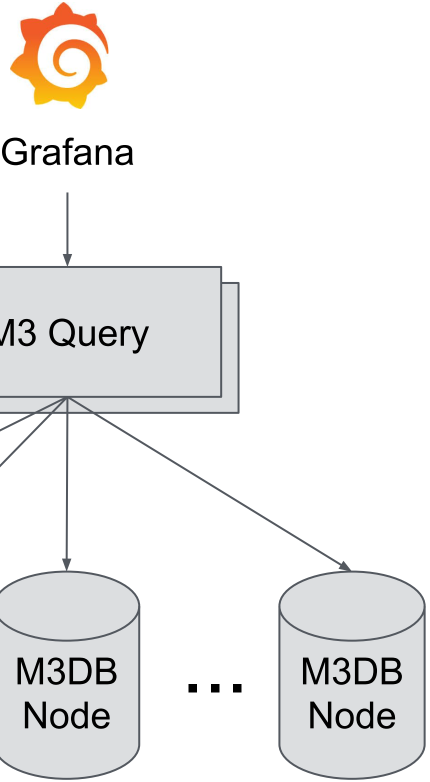
2. Scalable to billions of metrics

Architected for Reliability and Scale

- Each component designed to run across Availability Zones in a Region
- Low inter-region network bandwidth, data always kept in region

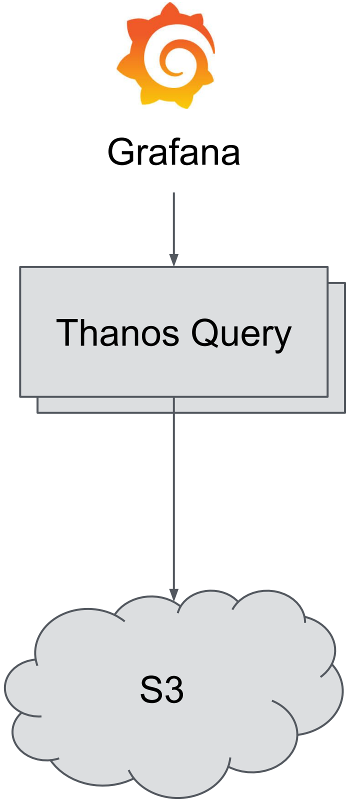
Queries executed in distributed and parallel

Each storage node
Find metrics matching query and return in parallel knowing exactly where to extract series data from local store.



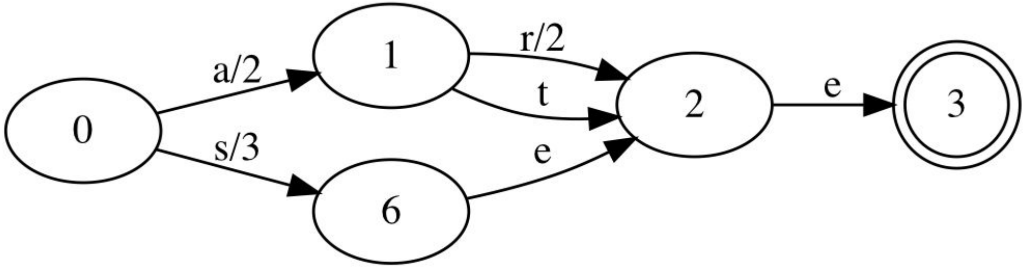
As opposed to fetch archived data to single node

Single query node
Read all index and data chunks for time windows included by query, if too much index data then can't hold it entirely in memory.



Index backed by FST segments

Filter and Regexp queries over billions of metrics
M3DB doesn’t use Go standard Regexp libraries which match each metric through iteration, Finite State Transducer segments (as used by Apache Lucene) are used with upstream changes to the Go Couchbase Vellum library.



3. Focus on simple operability

Powerful with focus on simple operation

- M3 can be deployed on premise without any dependencies.
- M3 also can run on Kubernetes and the M3DB k8s operator can manage your cluster.
    - See more at https://github.com/m3db/m3db-operator
- Clustered version open source and can scale to billions of time series.

Fewer roles, complexity pushed into role

1. Can get started with just two roles, M3 Coordinator and M3DB.
2. No background tasks requiring monitoring (uploads/downsampling/etc).
3. K8s operator handles replacing failed instances & scaling up and down instances as requested.
4. No single node bottleneck on scaling queries.

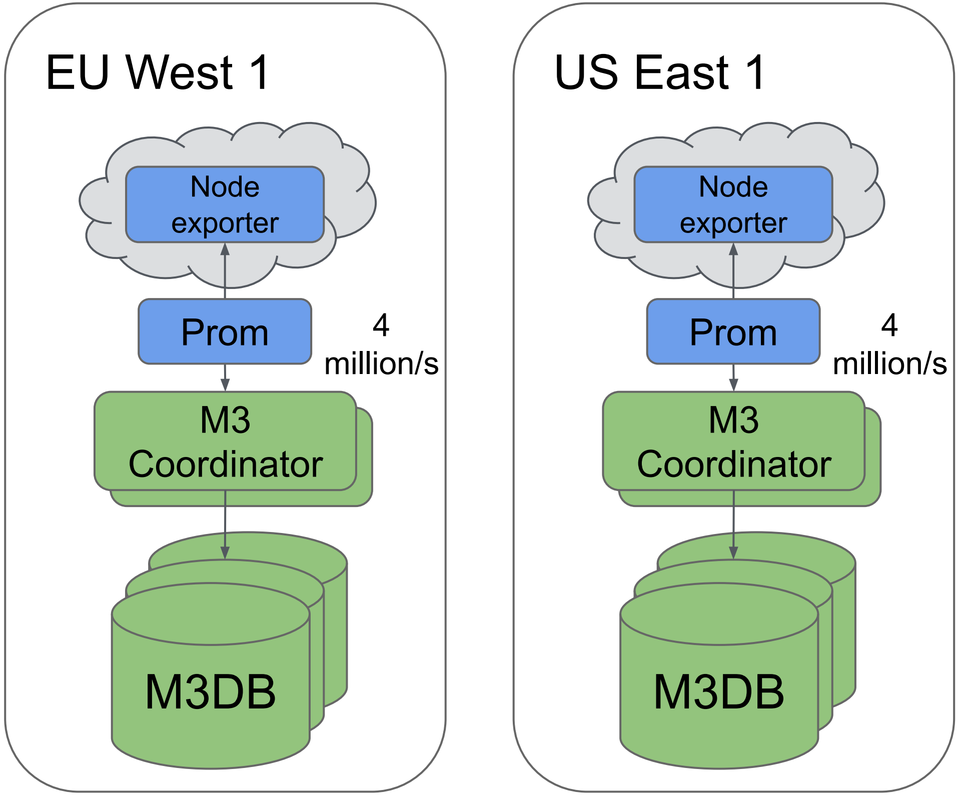
~~1. Runs anywhere~~

~~2. Scalable to billions of metrics~~

~~3. Focus on simple operability~~

Let’s try it out?

Demo https://github.com/m3db/bench_multiregion



Multi-Region

Roadmap

Next

1. Add further lifecycle management to the Kubernetes operator
2. Arbitrary out of order writes for writing data into the past and backfilling
3. Asynchronous cross region replication ($$ but useful in some environments)
4. M3QL query language support
5. Evolving M3DB into a more generic time series database (~event store)
    a. Efficient compression of events in the form of Protobuf messages

Thank you and Q&A

M3 License: Apache 2

Website: https://www.m3db.io

Repo: https://github.com/m3db/m3

Docs: https://docs.m3db.io

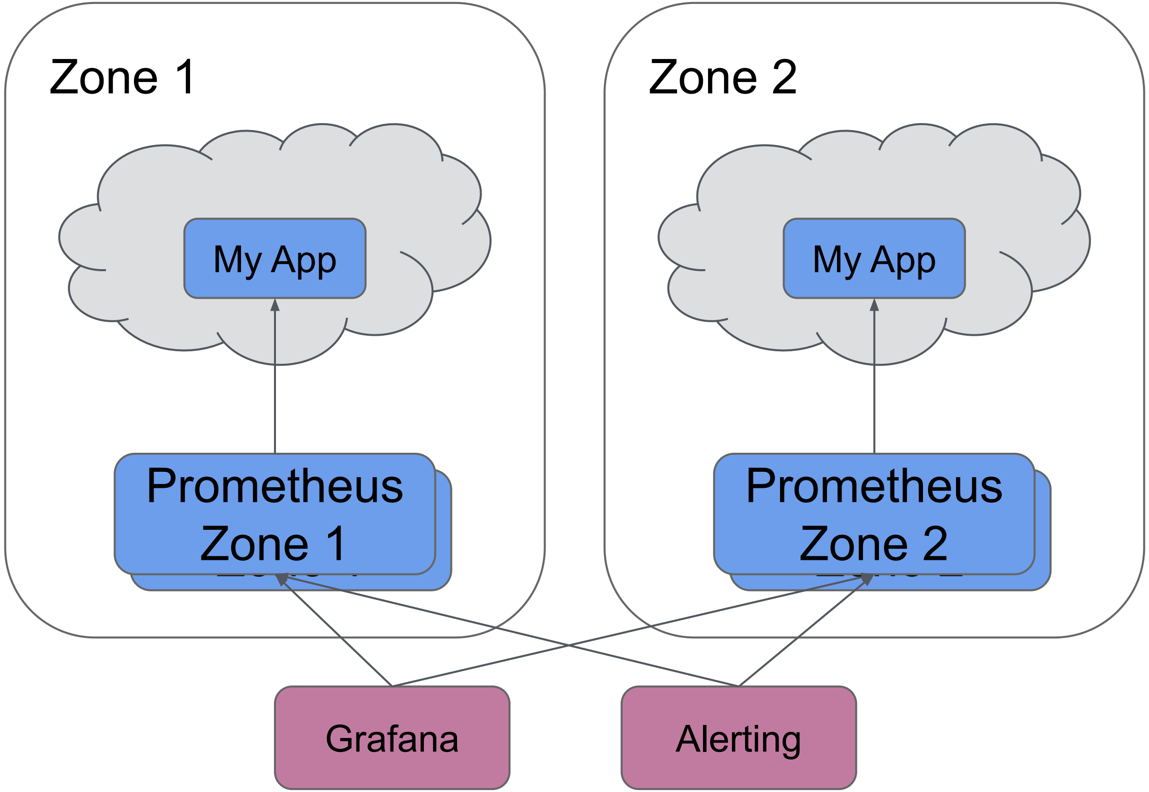
Gitter (chat): https://gitter.im/m3db/Lobby

Mailing list: https://groups.google.com/forum/#!forum/m3db

Blog post: https://eng.uber.com/m3

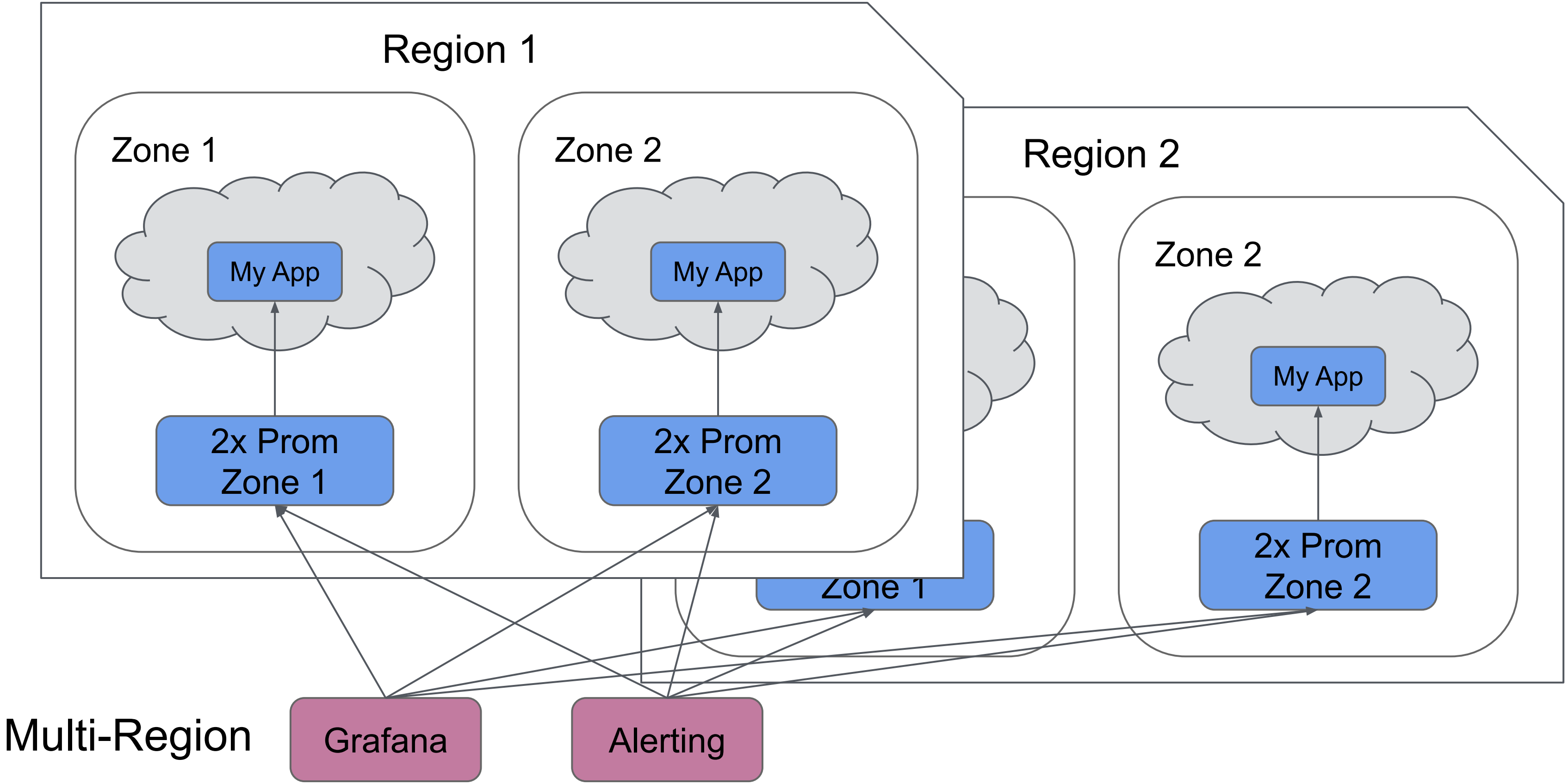
Appendix

Prometheus HA

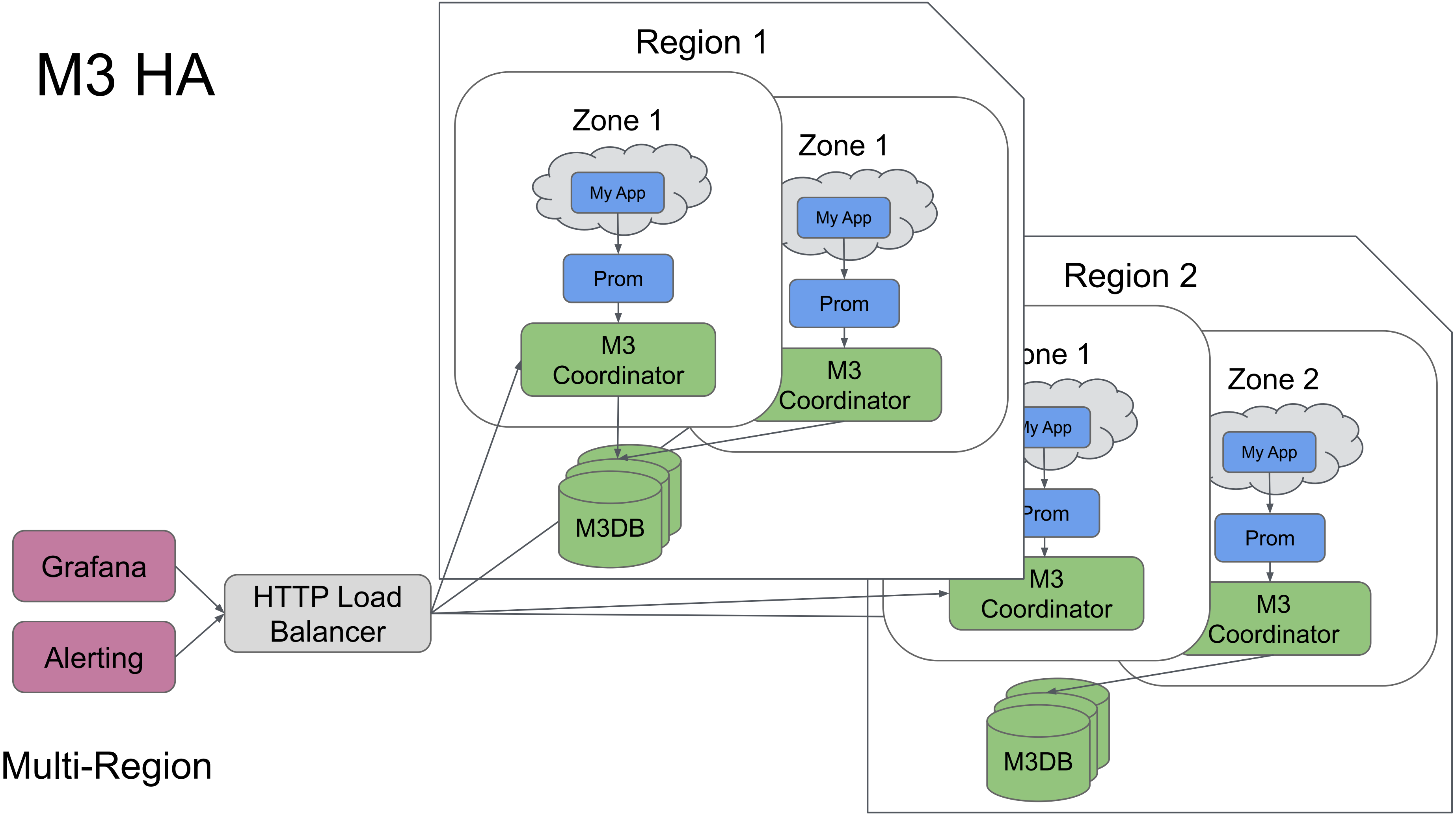


Single Region

Prometheus HA



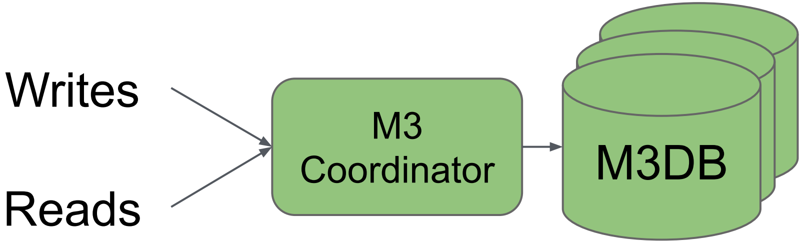
M3 HA



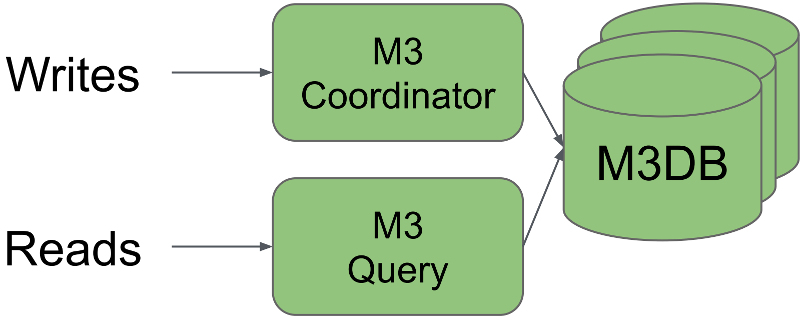
Multi-Region

M3 Coordinator and M3 Query

1. Reduce number of roles, however shared read/write path (reads can impact writes)

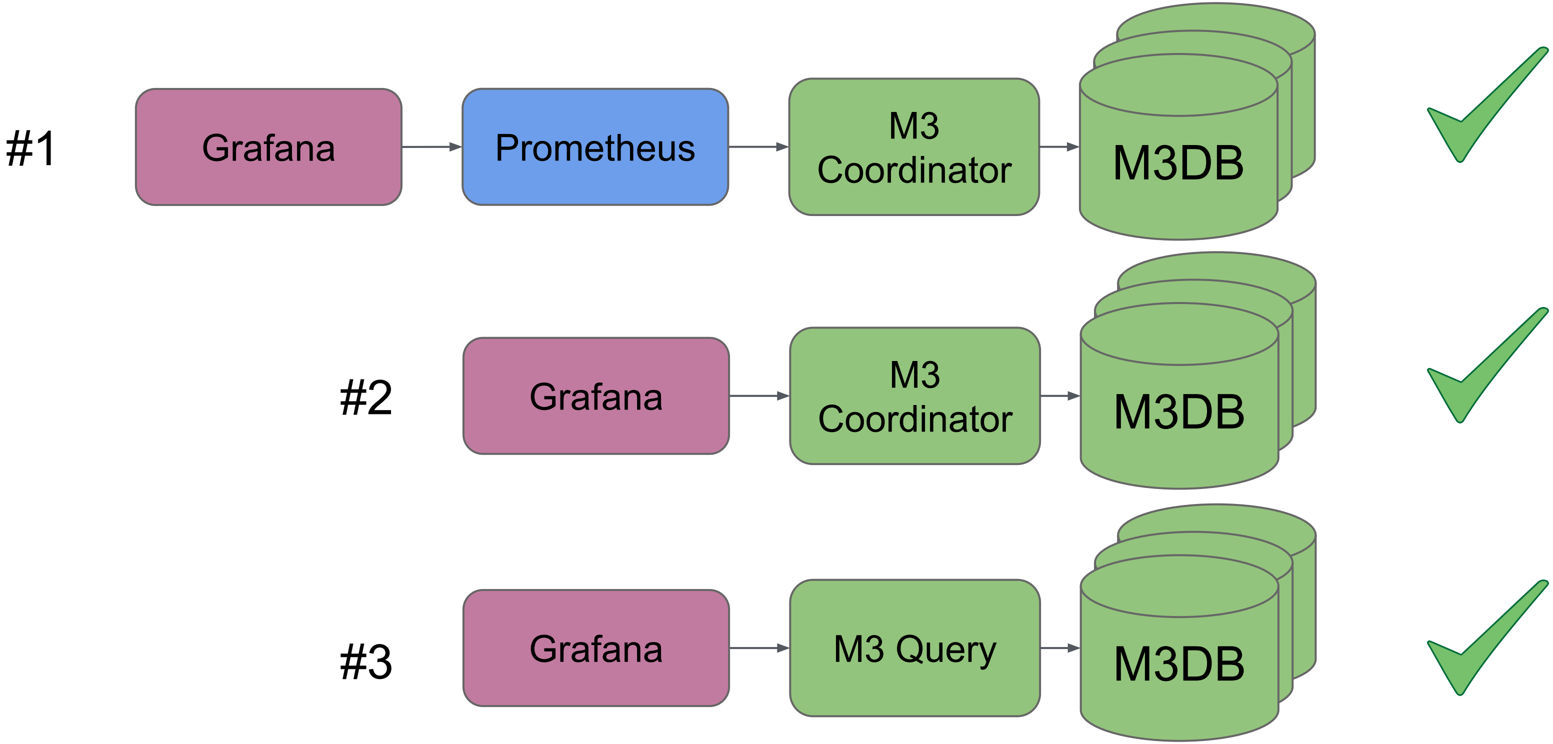


2. Dedicated M3 Query, more roles with isolated read/write path



Directly supports executing PromQL and Graphite

Both M3 Query and M3 Coordinator serving PromQL and Graphite queries directly


#1
Grafana
Prometheus
M3 Coordinator
M3DB
✓
#2
Grafana
M3 Coordinator
M3DB
✓
#3
Grafana
M3 Query
M3DB
✓