KubeCon CloudNativeCon Europe 2026

#KubeCon #CloudNativeCon

Discover Cortex: High Scalability Metrics in 2026

Friedrich Gonzalez & Charlie Le, Apple

Agenda

Introduction

Key improvements from the latest 1.20 release

Protocol Improvements
Prometheus Remote Write 2.0 + Native Histograms
OTLP Ingestion

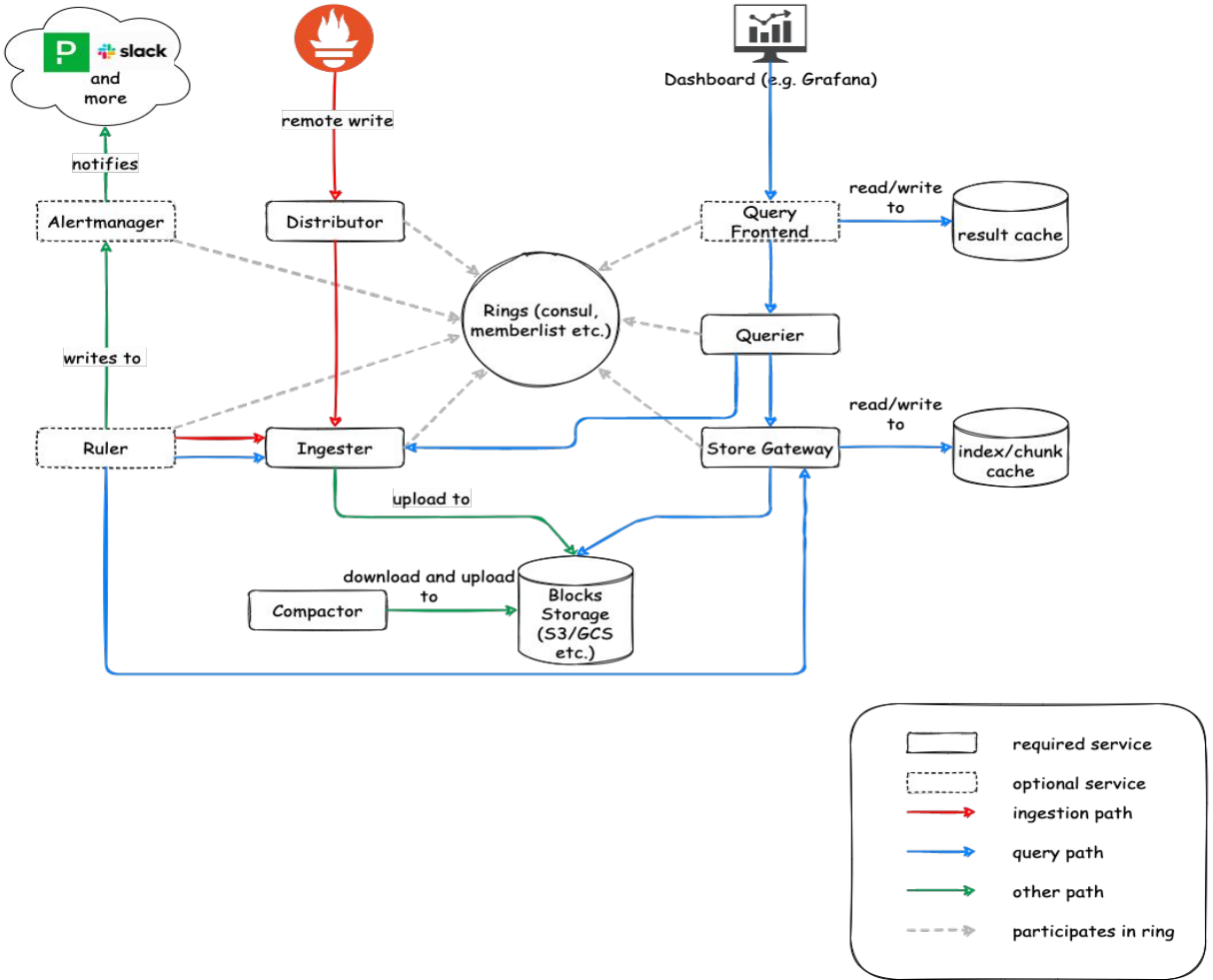
What is coming for 1.21 release
Overrides API
Are my metrics queried at all?

CNCF road to Graduation

Q&A with core maintainers
Get Involved

Introduction

Horizontally scalable, highly available, multi-tenant, long term storage solution for Prometheus and OpenTelemetry Metrics.



Remote Write & Ingestion

Prometheus Remote Write 2.0 - Experimental support

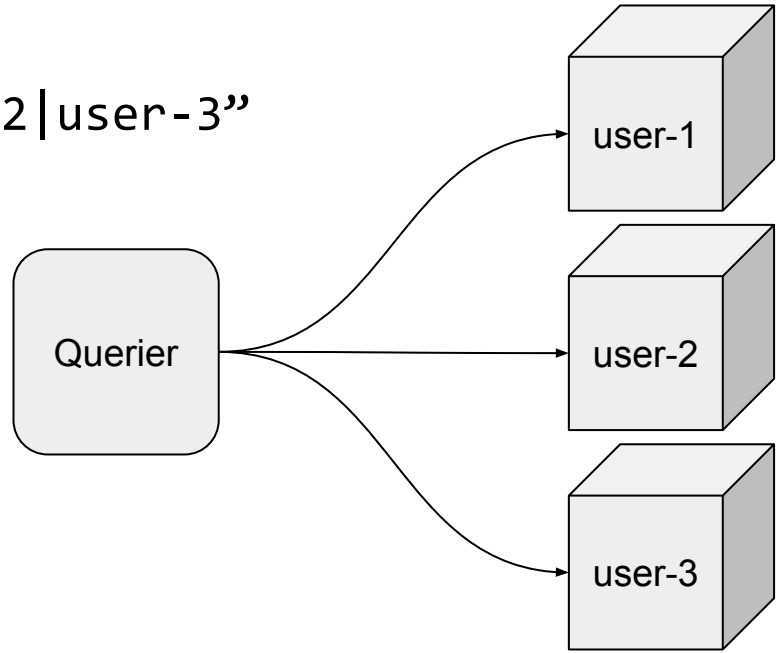
gRPC Stream Push - Experimental stream-based push between distributors and ingesters for improved efficiency

Native Histograms - Out-of-order ingestion support, per-tenant config, active series metrics, and ingestion rate limits

Multi-Tenancy

Regex Tenant Resolver - Pattern matching in X-Scope-OrgID header for flexible multi-tenant federation queries

```
X-Scope-OrgID: “user-1|user-2|user-3”
X-Scope-OrgID: “user-[1-3]”
```



Dynamic Query Splitting

Query frontend auto-adjusts split intervals to maintain shard and duration limits

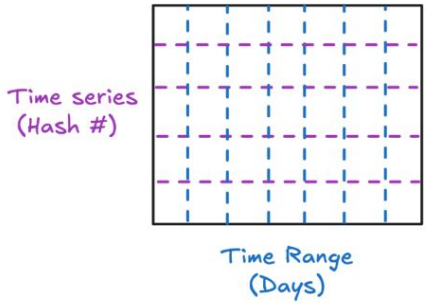
```
query_range:
  split_queries_by_interval: 24h
  dynamic_query_splits:
    max_shards_per_query: 100
    max_fetched_data_duration_per_query: 8760h # 365 day
    enable_dynamic_vertical_sharding: true

limits:
  query_vertical_shard_size: 4
```

Static Splitting

The query time range is split by a fixed interval and
time series are sharded into fixed number of subsets.

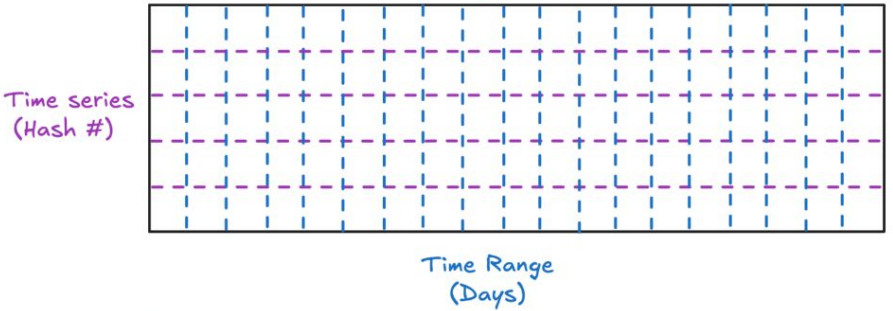
Short Range Query

The query has a healthy amount of sharding



Long Range Query

The query results in too much sharding given a static interval and vertical shard count

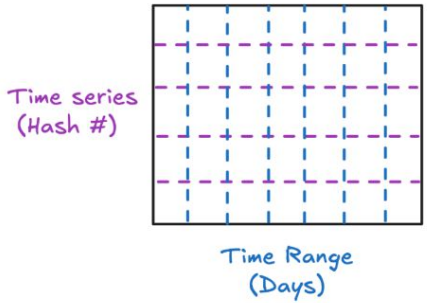


Static Splitting

The query time range is split by a fixed interval and
time series are sharded into fixed number of subsets.

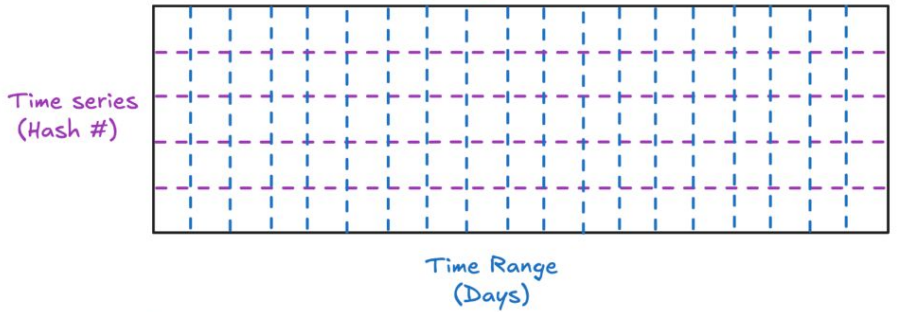
Short Range Query

The query has a healthy amount of sharding



Long Range Query

The query results in too much sharding given a static interval and vertical shard count

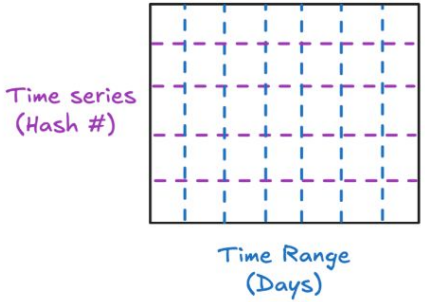


Dynamic Splitting

An optimal combination of split interval and vertical shard count
is used to achieve maximum parallelism within a configured target

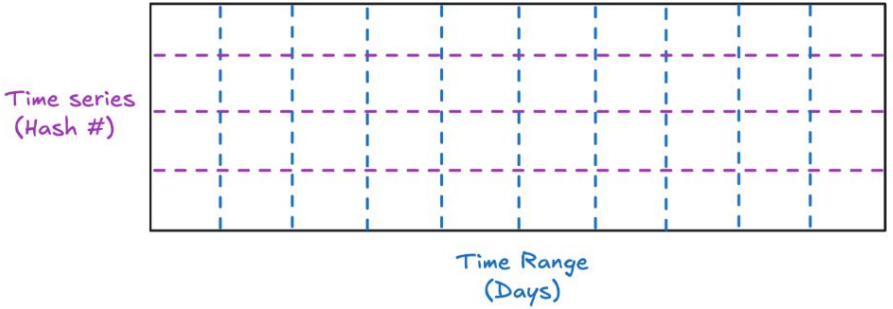
Short Range Query

The query has a healthy amount of sharding



Long Range Query

Split interval and vertical shard count are adjusted to maintain a healthy amount of sharding



Blog

READONLY Ingester Scaling
Query Priority in Cortex
Dynamic Query Splitting
Query Rejection in Cortex
Optimizing PromQL queries
Hello Cortex!

Efficient Query Parallelism in Cortex with Dynamic Query Splitting

RSS

This article explores the motivations behind adapting dynamic query splitting in Cortex, how the dynamic model works, and how to configure it for more efficient scalable PromQL query execution.

By **Ahmed Hassan (@afhassan)** | Tuesday, August 12, 2025

Tags: blog cortex query optimization
Categories: blog

Our Projects
cortex 6

Tag Cloud
blog 6 community 1
cortex 6 ingester 1
optimization 3 query 4
rejection 1 scaling 1
updates 1

Introduction

Cortex traditionally relied on **static query splitting** and **static vertical sharding** to optimize the execution of long-range PromQL queries. Static query splitting divides a query into fixed time intervals, while vertical sharding—when applicable—splits the query across subsets of time series. These techniques offered improved parallelism and reduced query latency but were limited by their one-size-fits-all approach. They did not account for differences in query range, lookback behavior, and cardinality—leading to inefficiencies like over-sharding, redundant data fetches, and storage pressure in large or complex queries.

To address those gaps, Cortex introduced **dynamic query splitting** and **dynamic vertical sharding**—two adaptive mechanisms that intelligently adjust how queries are broken down based on query semantics.

This article explores the motivations behind this evolution, how the dynamic model works, and how to configure it for more efficient scalable PromQL query execution.

Query Splitting

Query splitting breaks a single long-range query into smaller subqueries based on a configured time interval. For example, given this configuration, a 30-day range query will be split into 30 individual 1-day subqueries:

```
query_range:
    split_queries_by_interval: 24h
```

Reliability

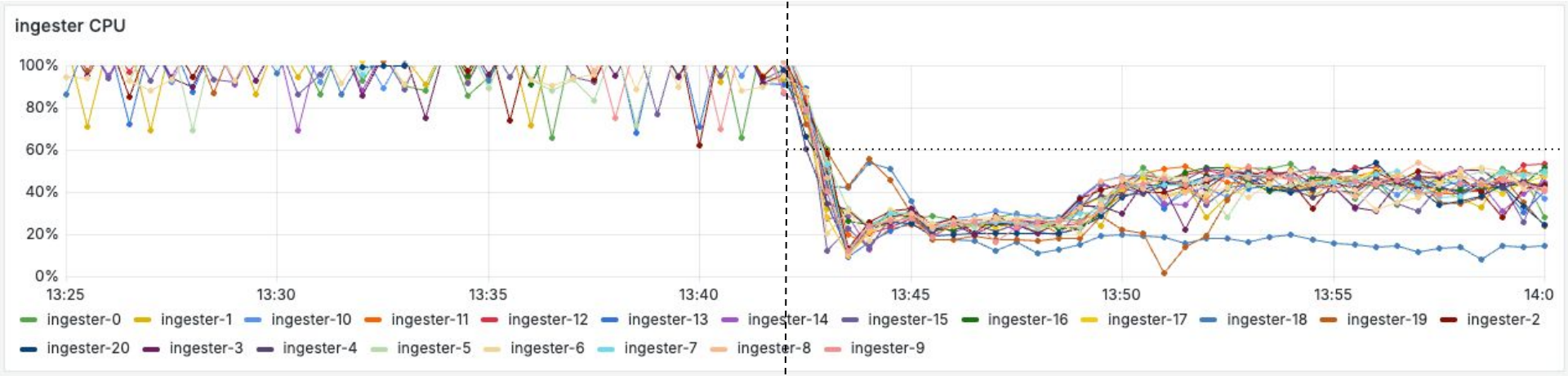
- ResourceMonitor
- ResourceBasedLimiter

New components to prevent service overload under high demand

```
target: ingester

monitored_resources: cpu,heap

ingester:
  query_protection:
    rejection:
      enabled: true
      threshold:
        cpu_utilization: 0.6
        heap_utilization: 0.8
```

No Limit

60% CPU Limit Applied



Resource-based throttling (Experimental)

Although the static limits are able to protect Cortex components from specific query patterns, they are not generic enough to cover different combinations of bad query patterns. For example, what if the query fetches relatively large postings, series and chunks that are slightly below the individual limits? For a more generic solution, you can enable resource-based throttling by setting CPU and heap utilization thresholds.

Currently, it only throttles incoming query requests with error code 429 (too many requests) when the resource usage breaches the configured thresholds.

For example, the following configuration will start throttling query requests if either CPU or heap utilization is above 80%, leaving 20% of room for inflight requests.

```
target: ingester
monitored_resources: cpu,heap
ingester:
  query_protection:
    rejection:
      enabled: true
      threshold:
        cpu_utilization: 0.8
        heap_utilization: 0.8
```

See https://cortexmetrics.io/docs/configuration/configuration-file/:~:text=query_protection for details.

Our Projects

cortex 6

Tag Cloud

blog 6 community 1
cortex 6 ingester 1
optimization 3 query 4
rejection 1 scaling 1
updates 1

Compatibility

UTF-8 Support - -name-validation-scheme flag enables UTF-8 metric name validation

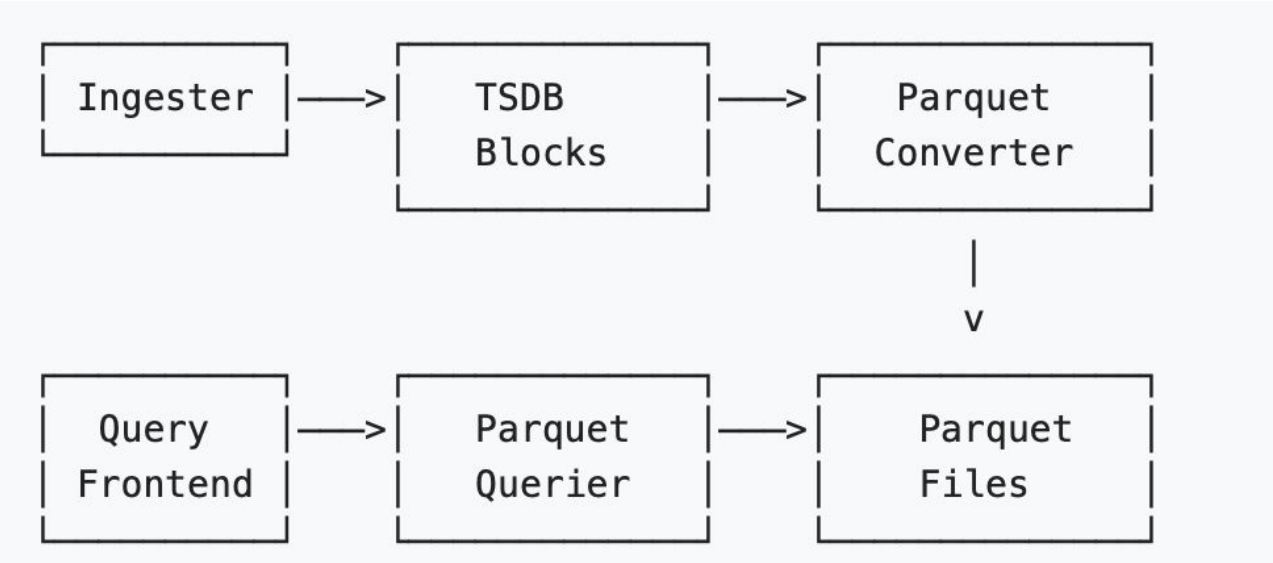
```
metric_name{"my.label.name"="bar"}
```
✅

```
metric.name{"my.label.name"="bar"}
```
❌

```
{"metric.name", "my.label.name"="bar"}
```
✅

Parquet Format

Service to convert TSDB blocks to parquet format, with experimental querier support



Parquet Mode

Overview

Parquet mode in Cortex provides an experimental feature that converts TSDB blocks to Parquet format for improved query performance and storage efficiency on older data. This feature is particularly beneficial for long-term storage scenarios where data is accessed less frequently but needs to be queried efficiently.

The parquet mode consists of two main components:

- **Parquet Converter**: Converts TSDB blocks to Parquet format
- **Parquet Queryable**: Enables querying of Parquet files with fallback to TSDB blocks

Why Parquet Mode?

Traditional TSDB format and Store Gateway architecture face significant challenges when dealing with long-term data storage on object storage:

TSDB Format Limitations

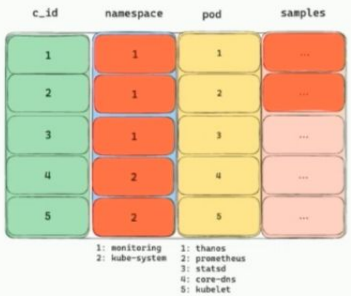
- **Random Read Intensive**: TSDB index relies heavily on random reads, where each read becomes a separate request to object storage
- **Overfetching**: To reduce object storage requests, data that are close together are merged in a single request, leading to higher bandwidth usage and overfetching
- **High Cardinality Bottlenecks**: Index postings can become a major bottleneck for high cardinality data

Store Gateway Operational Challenges

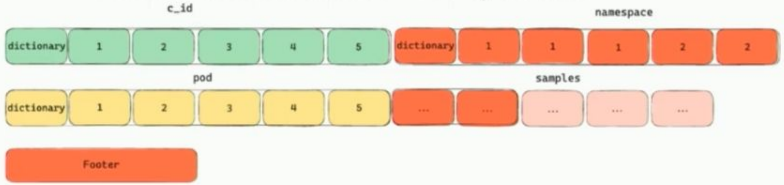
- **Resource Intensive**: Requires significant local disk space for index headers and high memory usage

Data-access patterns

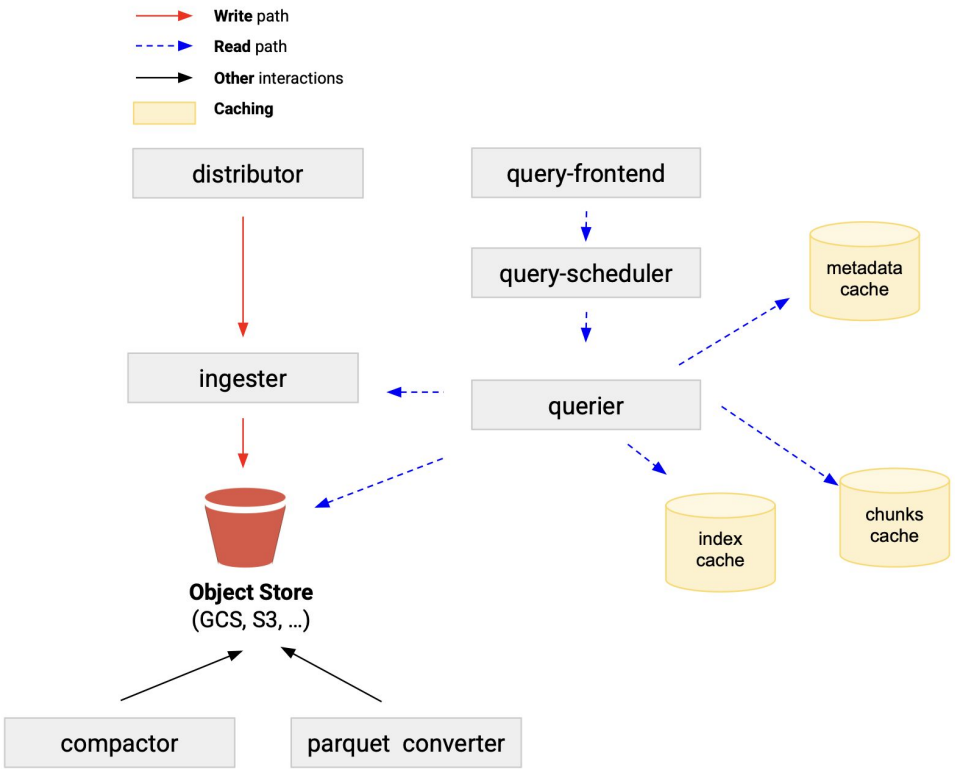
sum(kube_pod_info{**namespace**="monitoring"})



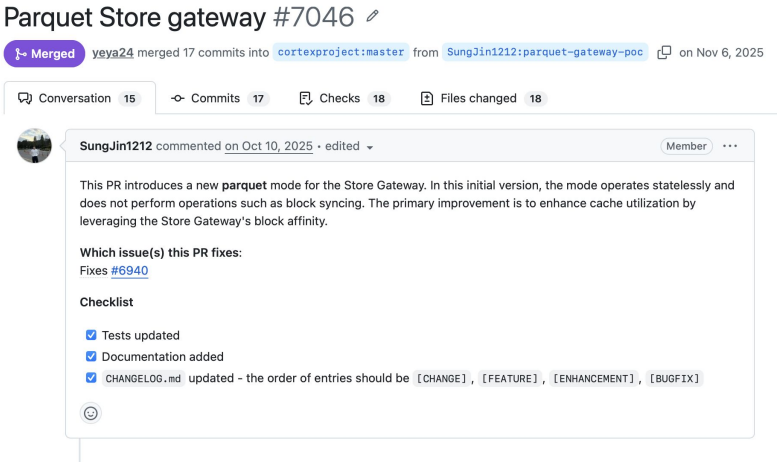
Column-oriented access - 3 IO operations



Parquet mode for Querier (v1.20)



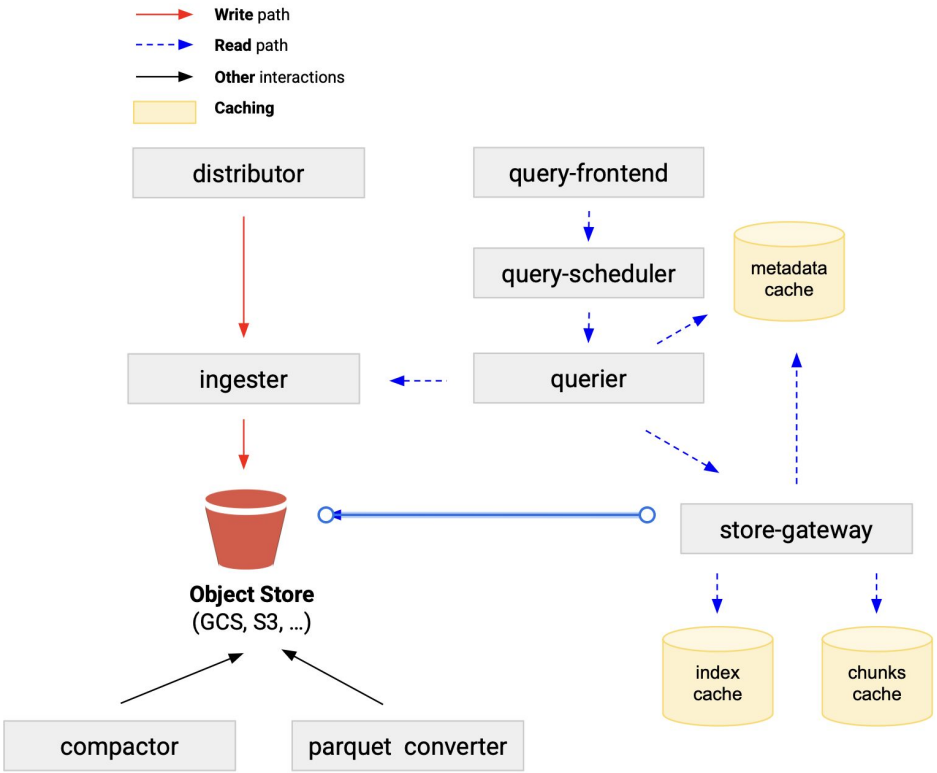
Parquet mode for Store-Gateway (v1.21)



What does it bring?

- Re-use store gateway block affinity to retrieve parquet files
- Improves cache utilization over in memory querier caching

Parquet mode for Store-Gateway (v1.21)



What is in the v1.21 release

This release contains 164 contributions from 29 contributors. We also have 12 new contributors. Thank you all for the contributions!

Release	v1.18.0	v1.19.0	v1.20.0	v1.21.0
Date	2024-08-19	2025-01-22	2025-10-31	2026–03-09
Changes	11	7	4	5
Features	11	12	18	10
Enhancements	23	41	59	68
Bugfixes	8	8	25	25
Total	53	68	106	108

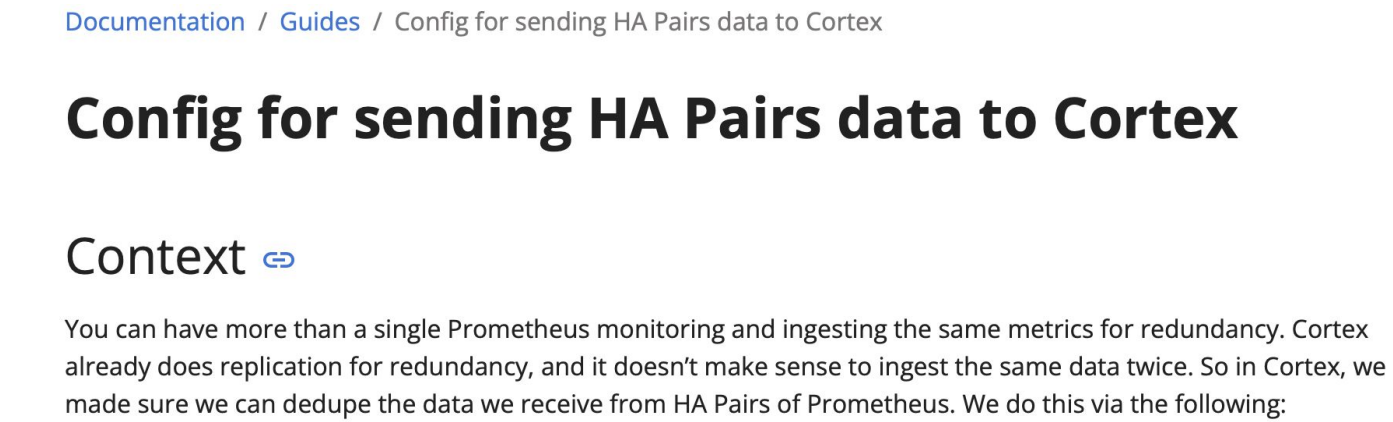
What is in the v1.21 release

Production readiness

- Ruler API
- Alertmanager API/sharding
- Tenant federation
- FIFO/Redis cache
- Instance limits
- Memcached DNS-based service discovery

What is in the v1.21 release

HATracker: Add experimental support for `memberlist` and `multi` as a KV store backend. #7284 (Feature)



Documentation / Guides / Config for sending HA Pairs data to Cortex

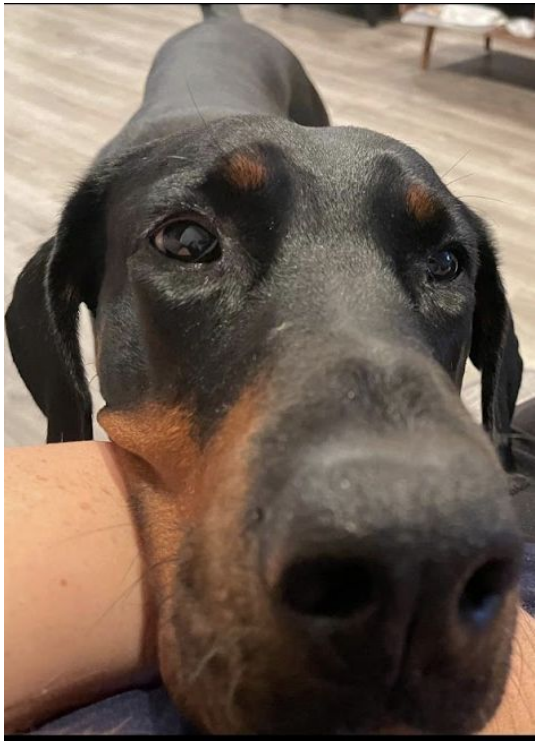
Config for sending HA Pairs data to Cortex

Context

You can have more than a single Prometheus monitoring and ingesting the same metrics for redundancy. Cortex already does replication for redundancy, and it doesn't make sense to ingest the same data twice. So in Cortex, we made sure we can dedupe the data we receive from HA Pairs of Prometheus. We do this via the following:

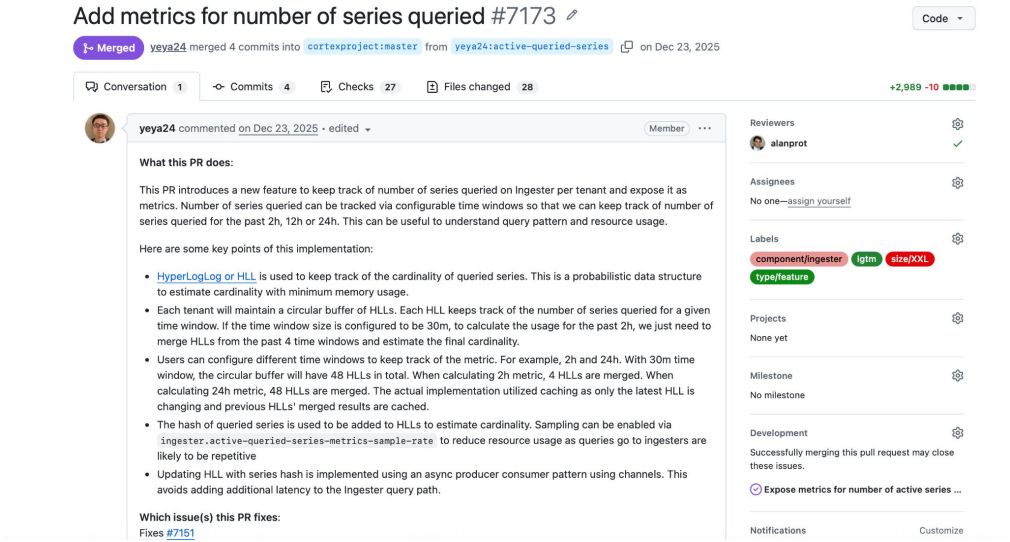
What is in the v1.21 release

Are my metrics queried at all???



What is in the v1.21 release

Now you can have metrics for number of series queried. Useful to understand query pattern and resource usage.



Add metrics for number of series queried #7173

Code

Merged yeya24 merged 4 commits into cortexproject:master from yeya24:active-queried-series on Dec 23, 2025

Conversation 1 Commits 4 Checks 27 Files changed 28 +2,989 -10

yeya24 commented on Dec 23, 2025 · edited Member

What this PR does:

This PR introduces a new feature to keep track of number of series queried on Ingester per tenant and expose it as metrics. Number of series queried can be tracked via configurable time windows so that we can keep track of number of series queried for the past 2h, 12h or 24h. This can be useful to understand query pattern and resource usage.

Here are some key points of this implementation:

- HyperLogLog or HLL is used to keep track of the cardinality of queried series. This is a probabilistic data structure to estimate cardinality with minimum memory usage.
- Each tenant will maintain a circular buffer of HLLs. Each HLL keeps track of the number of series queried for a given time window. If the time window size is configured to be 30m, to calculate the usage for the past 2h, we just need to merge HLLs from the past 4 time windows and estimate the final cardinality.
- Users can configure different time windows to keep track of the metric. For example, 2h and 24h. With 30m time window, the circular buffer will have 48 HLLs in total. When calculating 2h metric, 4 HLLs are merged. When calculating 24h metric, 48 HLLs are merged. The actual implementation utilized caching as only the latest HLL is changing and previous HLLs' merged results are cached.
- The hash of queried series is used to be added to HLLs to estimate cardinality. Sampling can be enabled via `ingester.active-queried-series-metrics-sample-rate` to reduce resource usage as queries go to ingesters are likely to be repetitive
- Updating HLL with series hash is implemented using an async producer consumer pattern using channels. This avoids adding additional latency to the Ingester query path.

Which issue(s) this PR fixes:
Fixes #7151

Reviewers: alanprot

Assignees: No one—assign yourself

Labels: component/ingester, lgtm, size/XXL, type/feature

Projects: None yet

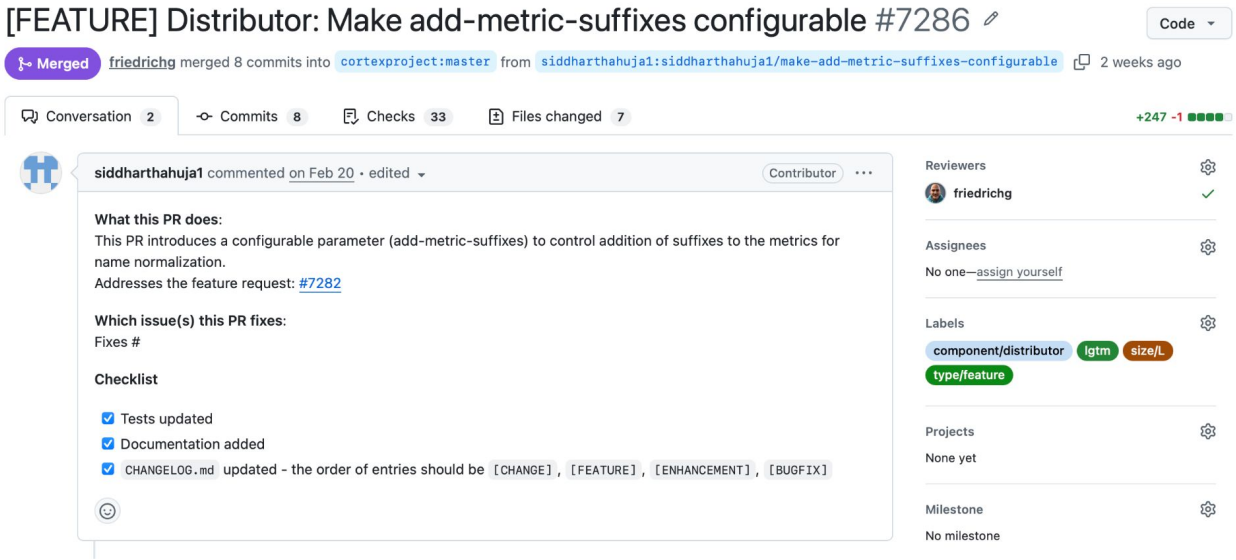
Milestone: No milestone

Development: Successfully merging this pull request may close these issues. Expose metrics for number of active series ...

Notifications Customize

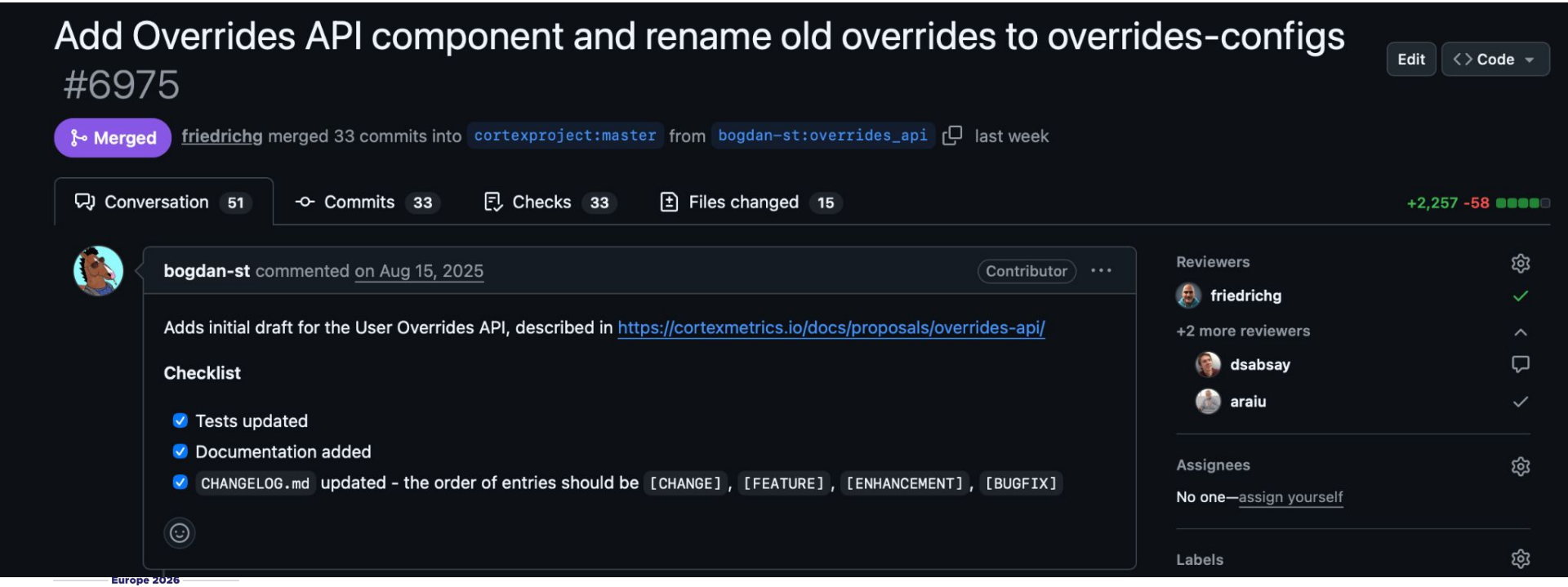
What is in the v1.21 release

Configurable OTLP metric suffixes via `-distributor.otlp.add-metric-suffixes`



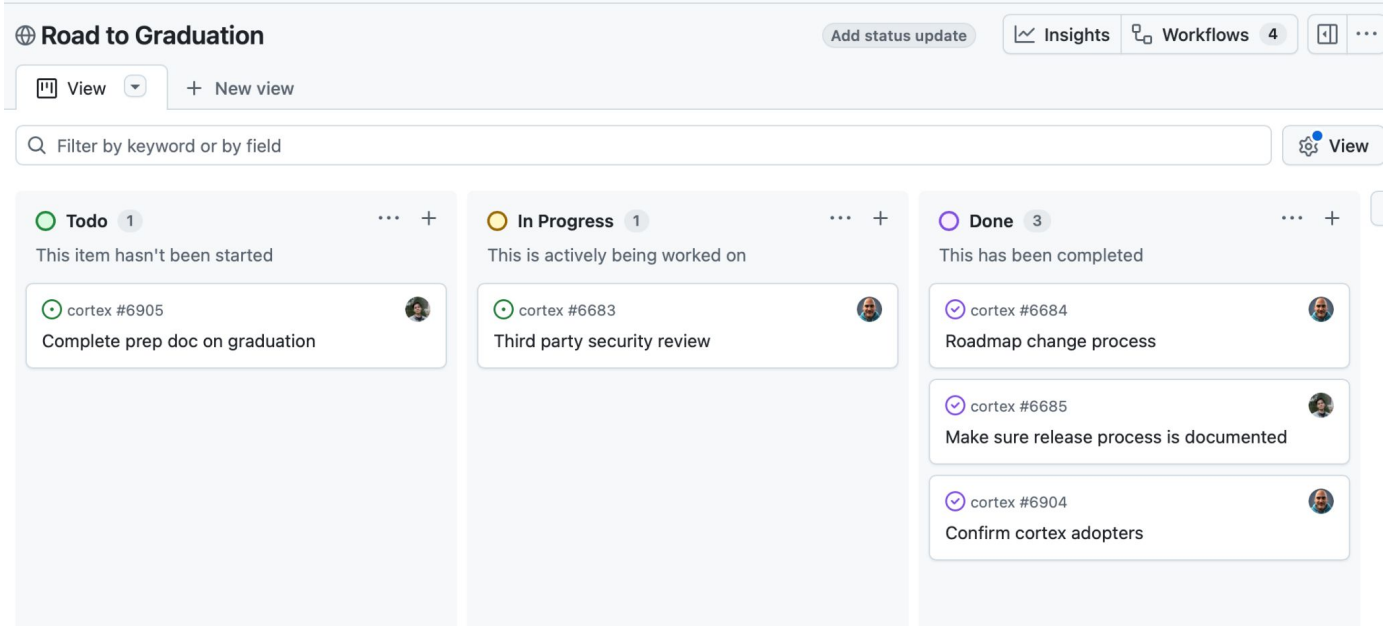
Overrides API

The User Overrides API provides a RESTful interface for managing tenant-specific limit overrides at runtime without requiring manual edits to the runtime configuration file or service restarts.



Road to CNCF Graduation

KubeCon CloudNativeCon Europe 2026



Road to CNCF Graduation

KubeCon CloudNativeCon Europe 2026

Get Involved

Cortex Community

About the community

SIG Meetings: every 4 weeks on Thursdays at 1700 UTC

Slack: https://cloud-native.slack.com/messages/cortex/

Issues: https://github.com/cortexproject/cortex/issues
Good first issues
Special topics
Enhancement Proposal Process

CNCF Code of Conduct

Q&A with core maintainers

Parquet?
Architecture & Scaling?
Multi-Tenancy?
Remote Write?
Community & Governance?
Operations?


KubeCon
CloudNativeCon
Europe 2026


KubeCon
CloudNativeCon
Europe 2026


KubeCon
CloudNativeCon
Europe 2026


KubeCon
CloudNativeCon
Europe 2026

Improved OTLP/HTTP Support

OTLP (OpenTelemetry Protocol) allows applications instrumented with OpenTelemetry SDKs to send traces, metrics, and logs directly to backend platforms or via the OpenTelemetry Collector. This vendor-neutral protocol enables direct-to-backend ingestion or intermediate collection, reducing dependence on specific agents.

Improvements:

- Supports Delta Temporality
- Flag to enable to types and units
- Metadata Ingestion

Connecting Your Laptop to Present

KubeCon CloudNativeCon Europe 2026

If you plan to use speaker notes, please follow the steps below to ensure your laptop is set up correctly. This will allow you to view your notes while the audience sees only your slides.

Before Connecting

- Close all unnecessary apps & tabs
- Turn off notifications
- Disable Night Mode / TrueTone
- Don’t enter full-screen presentation mode yet

At the podium

- Plug in the HDMI cable
- If needed, use your USB-C to HDMI adapter
- On Mac, click “Allow” if prompted

USE “EXTENDED DISPLAY” MODE (NOT MIRRORED)

Using MAC

- Go to System Settings > Displays
- Click the “Arrangement” tab
- Make sure “Mirror Displays” is unchecked

Using PC

- Press ⊞Win + P, then select “Extend”
- Or: Right-click on the Desktop > Display Settings
- Under Multiple Displays, select “Extend these displays”

Connecting Your Laptop to Present

KubeCon CloudNativeCon Europe 2026

When Using PowerPoint

- Start slideshow: ⌘Cmd + Enter (Mac) or F5 (PC)
- Or: Click Slide Show > Play from Start
- If notes show on the wrong screen, click “Swap Displays”
- **https://www.youtube.com/watch?v=gQ3D4m-5pww**

Google Slides

- Use Google Chrome as your browser
- Click the dropdown next to ‘Slideshow’ > select Presentation Display Options
- Check “Presenter View” and “Full Screen”
- If prompted, select "Allow"
- Choose the external display (may have a strange name like PanasonicXYZ)
- Click Start Slideshow
- **https://www.youtube.com/watch?v=-GT7WCvPcys**