

 TECHNICAL WORKSHOP • 40 MINUTES

Grafana Alloy 101

Unified metrics collection: effortless Pull scraping, native Push ingestion, and declarative River pipelines.

wftech

Věroš Kaplan

<https://veroskaplan.cz/>

WORKSHOP RESOURCES

Workshop Repository

Access code examples, configuration files, and hands-on exercises for the session.

GITHUB REPOSITORY

<https://github.com/wftech/grafana-alloy-workshop>

What's included in the repo:

- **River Pipeline Examples:** Complete declarative metrics & logs configurations.
- **Hands-on Exercises:** Step-by-step guides for Pull & Push telemetry setups.
- **Docker Environment:** Local playground with Prometheus, Grafana & Alloy.



Scan QR Code

Quick access on mobile

Pull vs. Push Mechanics in Alloy



Traditional Pull Model



Direct polling: Prometheus queries target endpoints periodically.

Target discovery: Central server coordinates every scrape schedule.

Inbound exposure: Requires open ingress ports to each exporter.



Push Model via Alloy



Local collection: Alloy grabs telemetry directly from app and host info.

Outbound push: Streams samples to Prometheus via remote_write.

Firewall friendly: No open inbound ports required on target machines.

Core Alloy Concepts & Architecture



01 • BUILDING BLOCK

River Components

Declarative units performing a single task: exporting, scraping, transforming, or writing.



02 • TOPOLOGY

DAG Data Flow

Directed Acyclic Graph where data flows automatically from sources to processing sinks.



03 • WIRING

forward_to Binding

Explicit arguments connecting component outputs directly to downstream receivers.

Bundled Exporters, External Targets & Destinations



Bundled Exporters

Embedded exporter + scrape

```
prometheus.exporter.unix "node" {  
  procfs_path = "/host/proc"  
  sysfs_path = "/host/sys"  
}  
  
prometheus.scrape "sys" {  
  targets =  
  prometheus.exporter.unix.node.targets  
  forward_to =  
  [prometheus.remote_write.to_tsdb.receiver]  
}
```

Direct forwarding: Bundled targets scraped and piped straight to remote storage.



External Targets

Third-party & custom apps

```
prometheus.scrape "app_svc" {  
  targets = [  
    {  
      address = "api:8080",  
      env = "prod",  
    }  
  ]  
  metrics_path = "/metrics"  
  forward_to =  
  [prometheus.relabel.tagger.receiver]  
}
```

Scrapes any endpoint: Pulls custom HTTP endpoints and routes to processors.



Pipeline & Destination

Relabeling & TSDB sink

```
prometheus.relabel "tagger" {  
  forward_to =  
  [prometheus.remote_write.to_tsdb.receiver]  
  rule { target_label = "cluster", replacement  
    = "prod" }  
}  
  
prometheus.remote_write "to_tsdb" {  
  endpoint { url =  
    "http://prom:9090/api/v1/write" }  
}
```

Remote destination: Modifies labels and batches samples to Prometheus TSDB.

Pull Pipeline: From Exporter to TSDB



1. Exporter

Generates system metrics on host CPU, RAM, and storage partitions.



2. Scraper

Periodically pulls target metrics and writes to the forwarding stream.



3. Remote Write

Batches and pushes collected samples directly to Prometheus.

Visual Debugging & Instant Feedback

Live Web UI at Port 12345

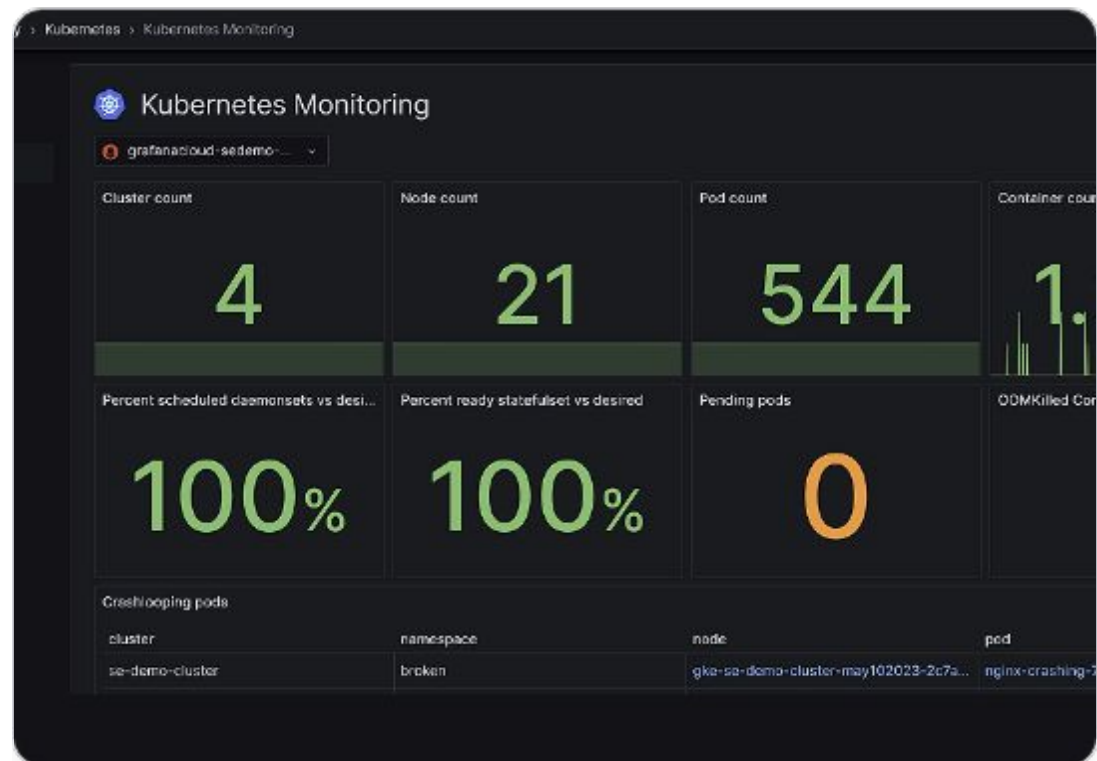
Real-time interactive DAG graph showing component health and state.

Hot Reload Without Restarts

Live reloads via HTTP POST to `/-/reload` keep pipelines running.

Standard Formatting

Built-in alloy fmt formats configuration files automatically.





Let's Start Building

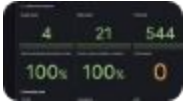
Pull or push, one binary connects your entire telemetry architecture.

 **config.alloy**

 **localhost:12345**

 **localhost:3000**

Image Sources



<https://grafana.com/media/solutions/kubernetes/kubernetes-monitoring-in-grafana-landing.png?w=1128>

Source: grafana.com