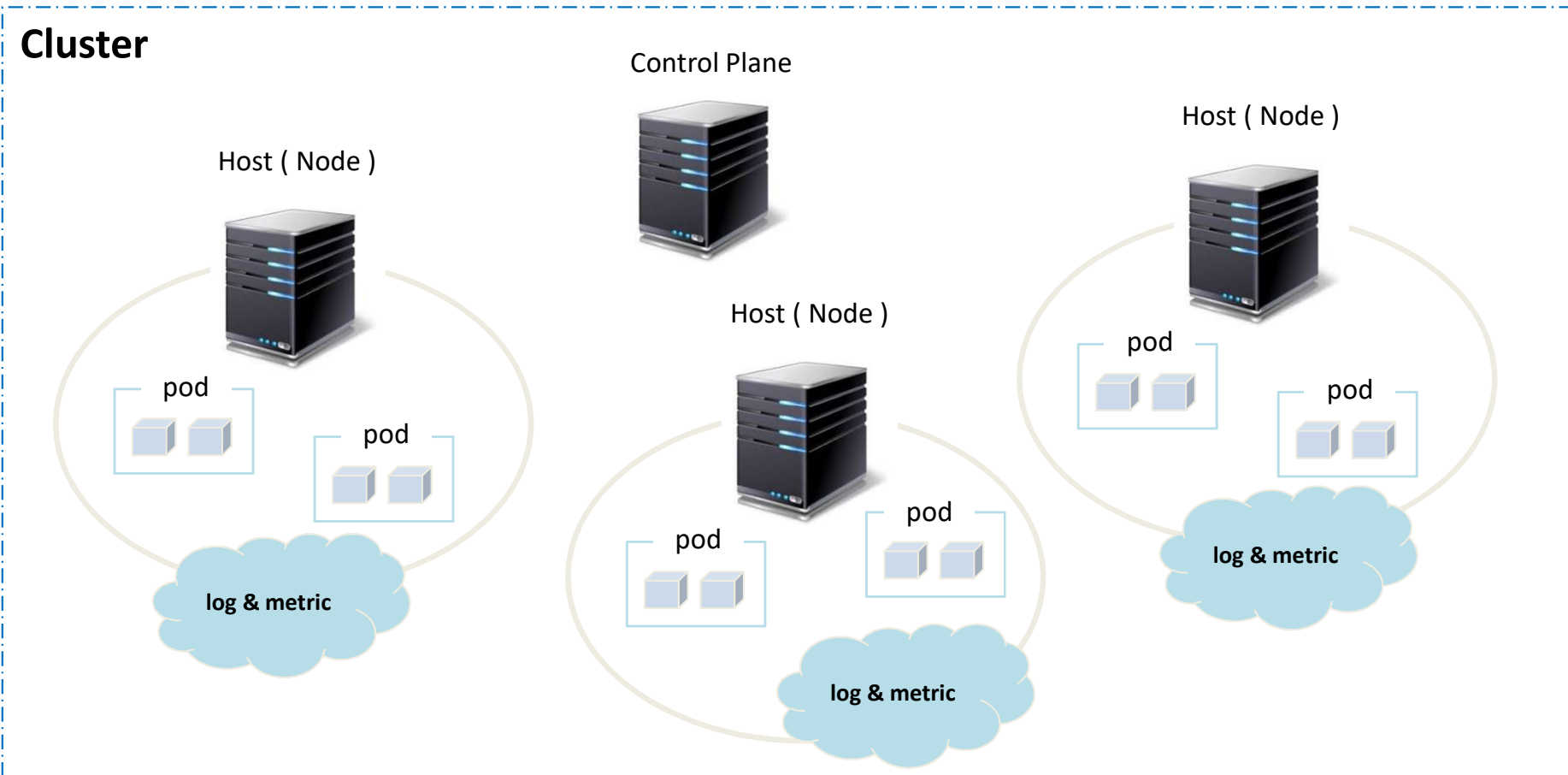


# Monitoring System 개요

## ◆ Why use monitoring system?

### Cluster



⇒ Current host PC status (CPU, memory, storage, network traffic .....)?










⇒ Current pod & app status ( running, using memory .....)?

⇒ Cluster & PV & PVC ( storage ) .....

⇒ log & metric management ?

## ◆ Monitoring System

- logging : 시스템에서 발생하는 log를 수집 & 정리 하여 관리하는 monitoring system
- metrics : 시스템의 각종 지표를 수집하여 저장 및 검색 할 수 있는 시스템

	Data collect	Data store & search	Data visualize
logging	 <b>fluentd</b>  <b>logstash</b>  <b>beats</b>	 <b>elasticsearch</b>	 <b>kibana</b>
Metric	<div>Exporter (Prometheus)</div>	  <b>influxdb</b>	 <b>Grafana</b>  <b>graphite</b>

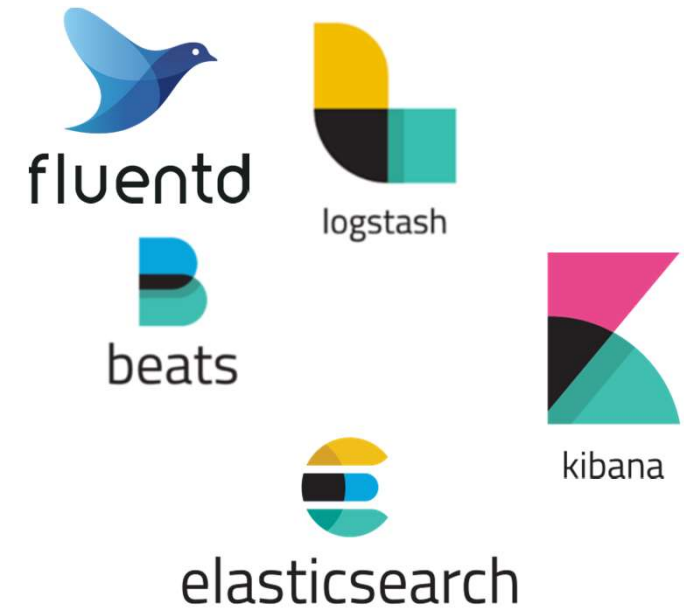
## ◆ Metric monitoring system

- System 및 App의 특정 시점에서의 측정 값을 저장 및 감시 하는 시스템
- 일반적으로 Metric name + label + data(정수) 값이 수집됨
- 고정 된 시간 간격으로 data를 수집
- Time series (시계열) Database 사용하여 metric data를 용도에 맞게 빠르게 정리



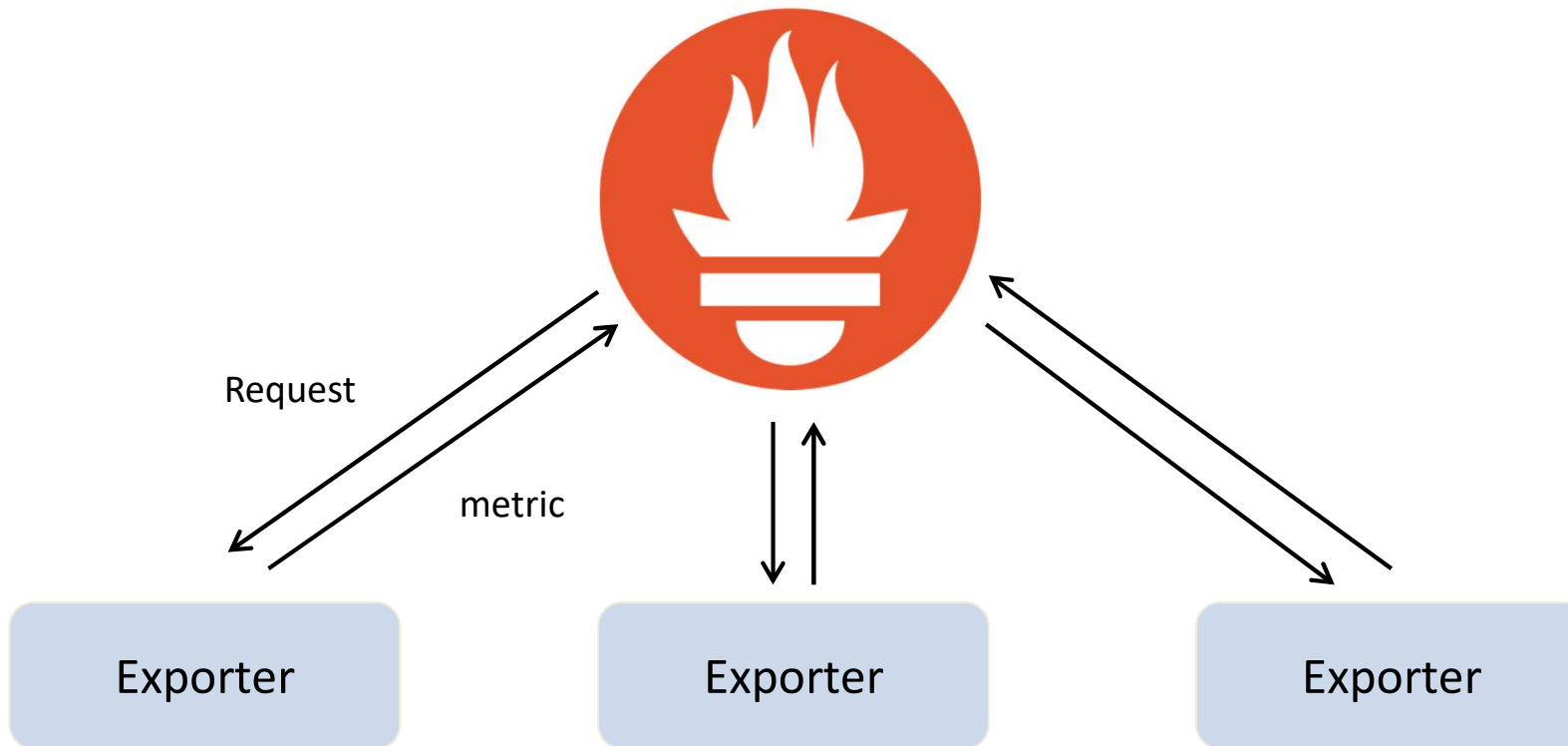
## ◆ Logging monitoring system

- 여러 이벤트를 통해 발생하는 log를 수집 및 관리 하는 모니터링 시스템
- 특정 상황에 대해 발생하는 log를 저장
- 비 주기적인 data 수집 ( 이벤트 발생시 전달 )

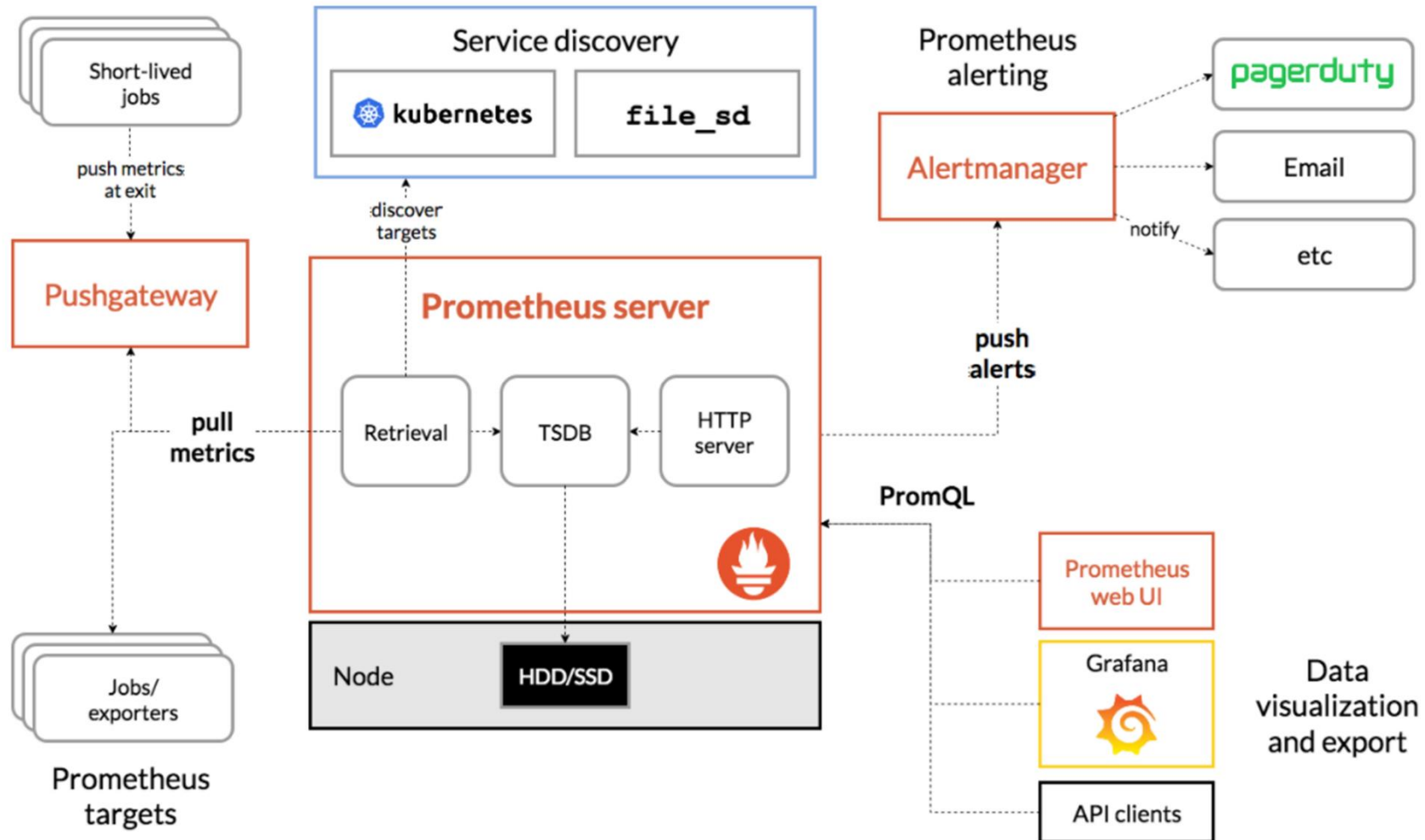


## ◆ Prometheus

- ⇒ SoundCloud에서 개발한 open-source **time series** system monitoring & alerting toolkit
- ⇒ 간단한 텍스트 형식으로 metric을 쉽게 노출 가능하며, 데이터 모델은 key-value 형태로 레이블을 집계한 후, Grafana같은 dashboard시스템에서 그래프로 쉽고 간단하게 dashboard를 만들 수 있다.
- ⇒ 일반 모니터링 시스템이 각 host의 agent가 서버로 metric을 보내는 push 방식을 사용하는 반면 Prometheus는 주기적으로 agent(exporter)로부터 metric을 pull하는 방식
- ⇒ Metric을 분류 및 활용하기 위해 PromQL이용



## ◆ Prometheus architecture



## ◆ Metric & PromQL (Prometheus Query)

⇒ 각 Export는 Prometheus가 필요한 metric을 생성하고 Prometheus는 주기적으로 metric을 가져와 Database에 저장

⇒ Prometheus & Grafana는 저장된 metric을 PromQL을 사용하여 사용자가 보기 편하도록 분류 하며 시각적으로 표현함

### ■ Metric Format

<metric name>{<label key>=<label value>, <label key>=<label value> ...} <metric value> [<timestamp>]

ex)

node\_network\_receive\_bytes\_total{device="eth0"} 5.5904233e+07

node\_network\_receive\_bytes\_total{device="eth1"} 3.74191033e+08

### ■ PromQL Format

func(<metric name>{<label key>=<label value>, <label key>=<label value> ...} [range vector])

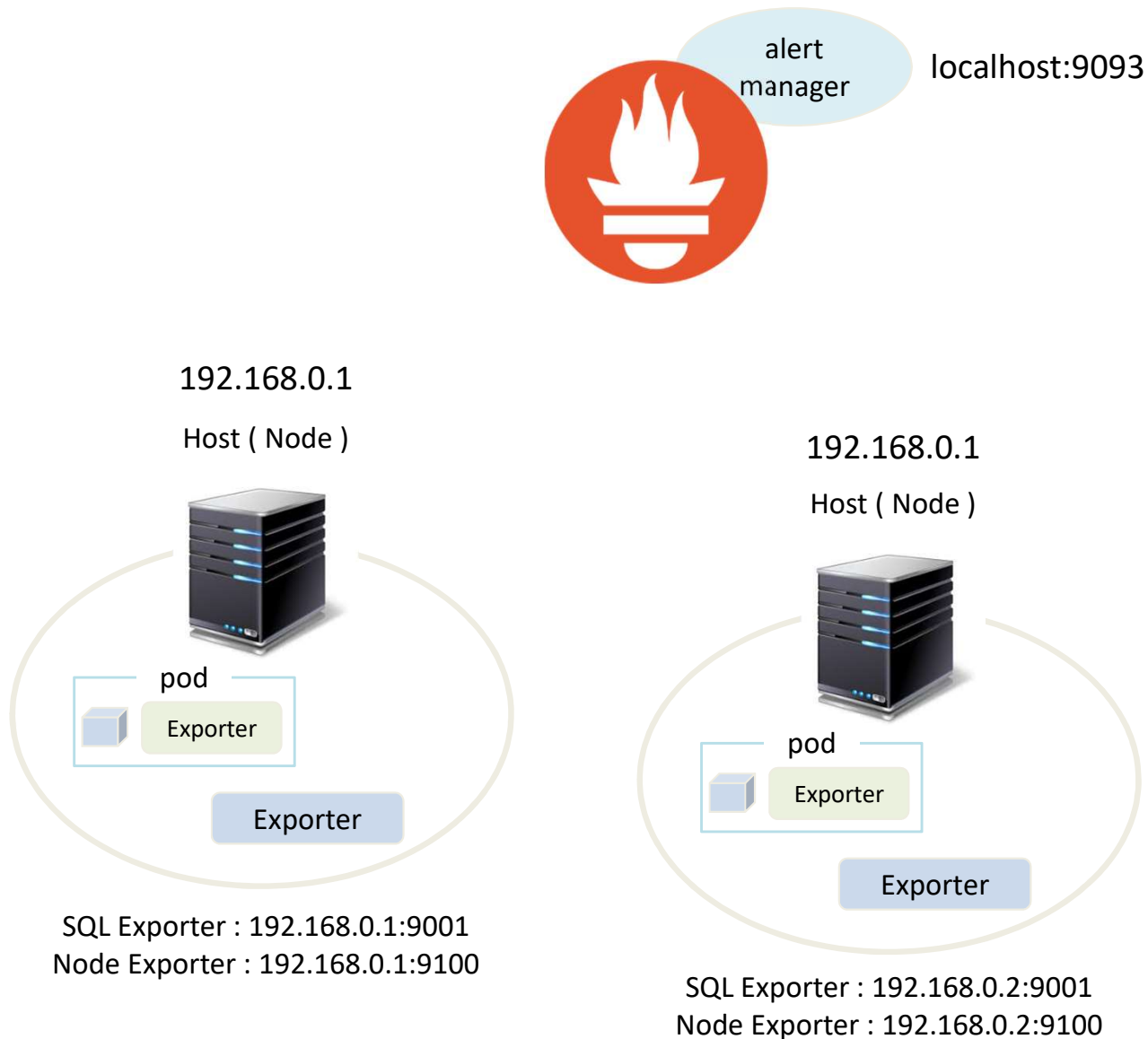
ex)

rate(node\_network\_receive\_bytes\_total{device="eth0"}[3m])

rate(node\_network\_receive\_bytes\_total{device!="eth0"}[10m])



## ◆ Prometheus config



## prometheus.yml

```
global:
  scrape_interval: 15s
  evaluation_interval: 15s

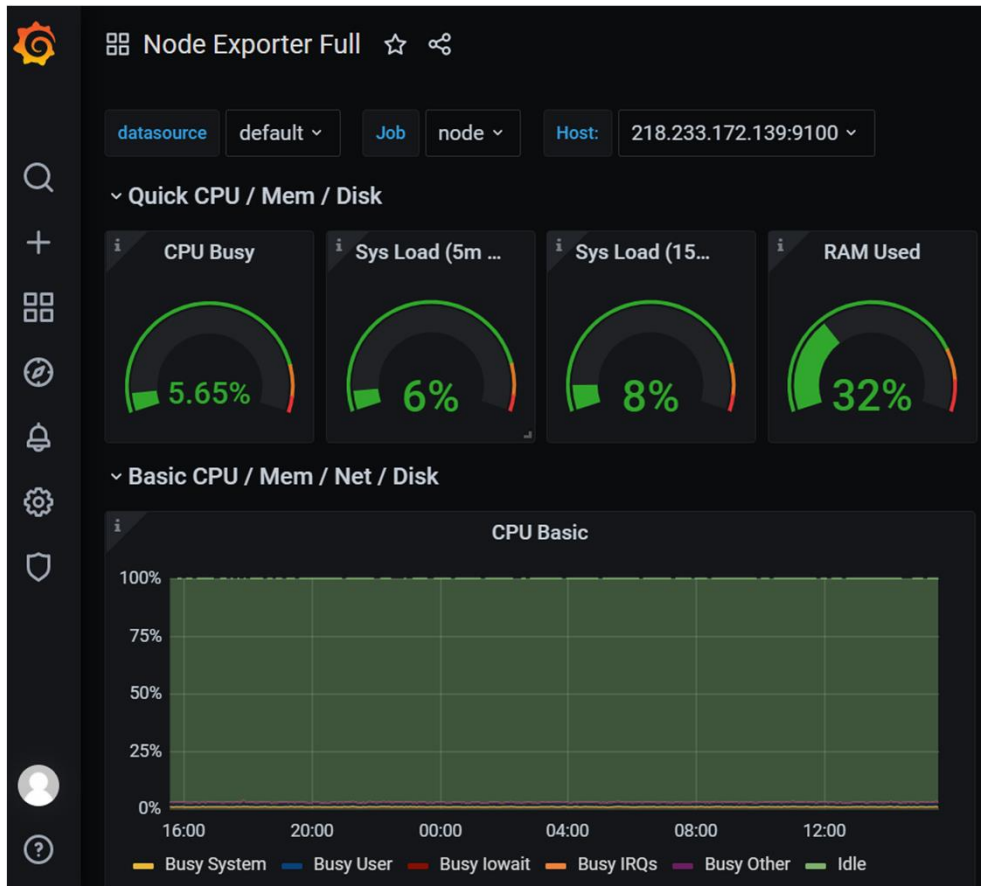
rule_files:
  - alert.rules.yml
alerting:
  alertmanagers:
    - static_config:
        - target: ['localhost:9093']

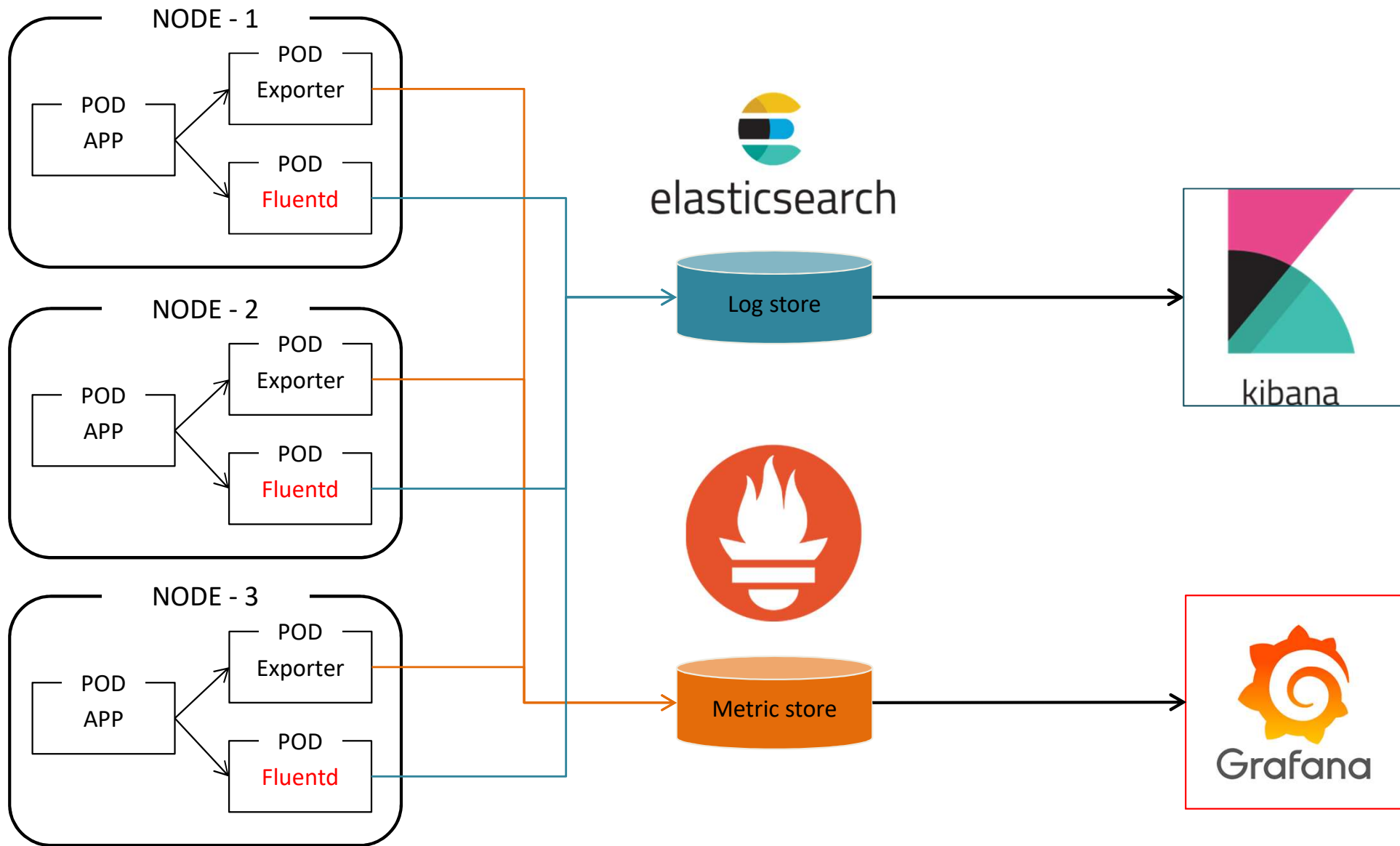
scrape_configs:
  - job_name: 'node'
    static_configs:
      - targets: ['192.168.0.1:9100']
      - targets: ['192.168.0.2:9100']
  - job_name: 'SQL Server'
    static_configs:
      - targets: ['192.168.0.1:9001']
      - targets: ['192.168.0.2:9001']
```



## ◆ Grafana

- ⇒ metric & log 시각화 dash board
- ⇒ database 혹은 cloud service의 metric & log를 가져와 data를 시각화
- ⇒ 사용자가 원하는 형식의 dash board 제작 및 Grafana lab을 통해 유저들이 제작 해 놓은 dash board 사용 가능
- ⇒ Prometheus 뿐만 아니라 다양한 프로그램을 지원 ( Elasticsearch, InfluxDB .. )





## **CONFIDENTIALITY**

Our industry is extremely competitive. The confidentiality of companies' plans and data is obviously critical. Will protect the confidentiality of all such company information. We view our approaches and insights as proprietary and therefore look to our future to protect 's interests in our presentations, methodologies and analytical techniques. Under no circumstances should this material be shared with any third party without the written consent of HFR.