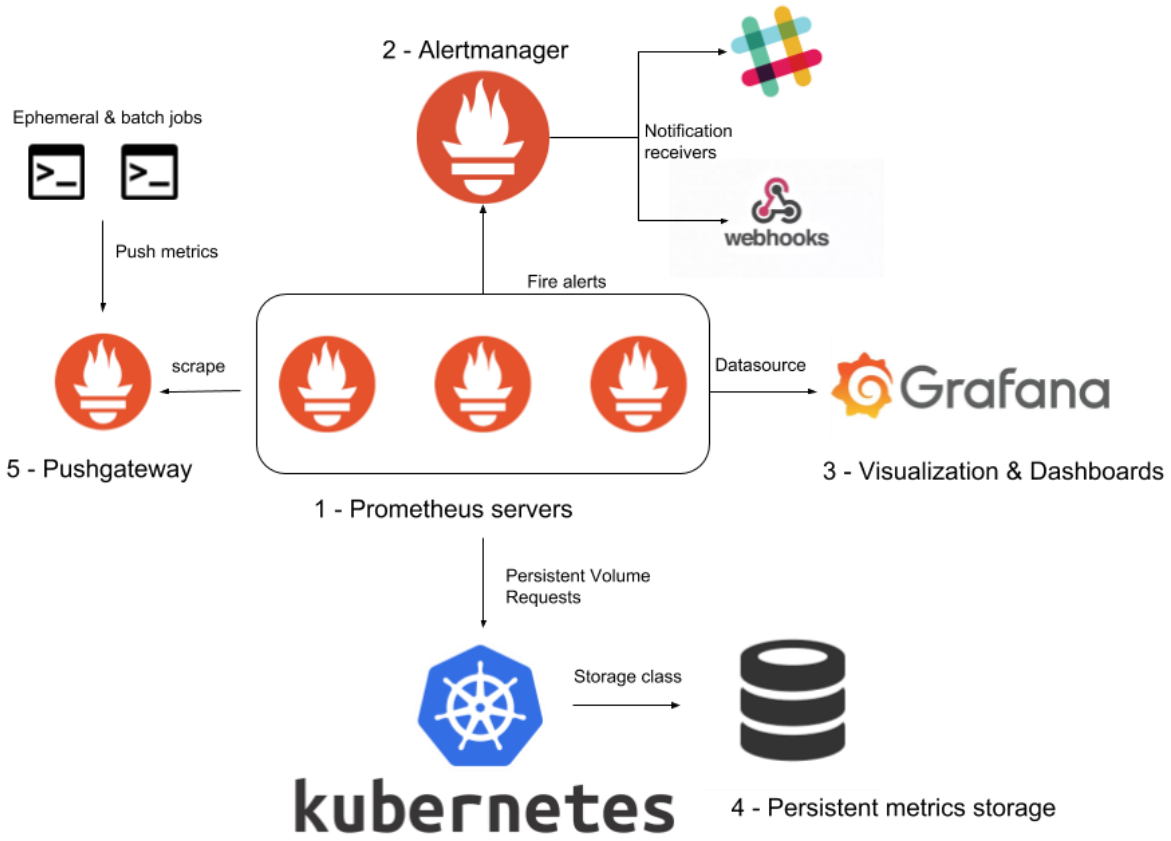
Prometheus

1. 아키텍쳐와 장단점

1) 아키텍쳐



오픈소스 시스템 모니터링 및 경고 툴 킷 으로 시스템을 모니터링하고 알람을 발생시켜 운영을 지원하는 것을 주요한 기능으로 갖는다.

Prometheus Server

- **exporter**에 수집된 데이터를 주기적으로 가져와**(Retrieval)** 시계열 저장소에 넣고**(TSDB)** 웹서버**(HTTP server)**를 통해 조회 및 활용 할 수 있도록 한다.
- exporters의 HTTP endpoint에서 수집된 데이터를 주기적으로 가져오는 역할을 수행하는 Pool 방식으로 동작한다.

Jobs/exporter

-메트릭을 수집하는 프로세스로 간단하게 모니터링 데이터를 수집하는 Agent 역할을 한다.

- Exporters는 수집한 메트릭을 HTTP를 통해 가져갈 수 있도록 /metrics 라는 HTTP endpoint를 제공한다.

Service Discovery
- K8S 와 같은 컨테이너 환경에서는 대상의 IP가 동적으로 변경되는 경우가 많아 일일이 설정 파일에 넣는데 한계가 있다.
- DNS나 Consul, etcd와 같은 다양한 서비스와 연동을 통해 자동으로 모니터링 대상의 목록을 가지고 올 수 있다.

Grafana
-PromQL을 사용하여 데이터를 시각화 하거나 재가공 할 수 있다.

중앙서버(Prometheus 서버)에서 에이전트에게 메트릭 정보를 polling 하는 구조

2) 프로메테우스의 장단점

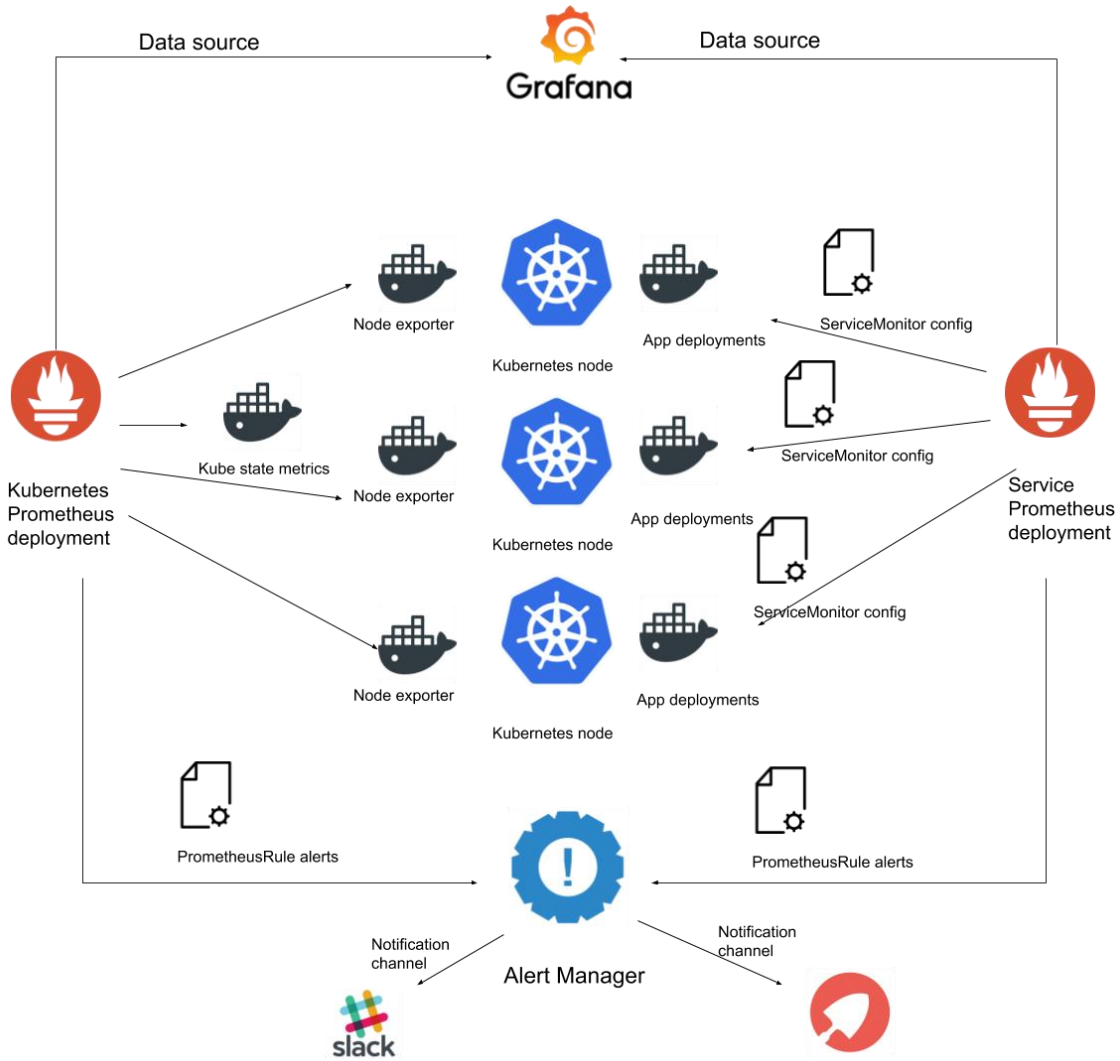
(1) 장점

프로메테우스는 모든 데이터를 수집하지 않고 일정 주기(default 15s)로 발생하는 메트릭을 수집하며, 에이전트 Pull 방식의 구조로 중앙 서버의 부하가 적음.

데이터 저장소가 시계열 데이터 저장소로 구성되어 있어, 많은 양의 정보를 빠르게 검색 가능.

(2) 단점

Sale-out이 불가하여 일반적인 Clustering(Host를 여러대 두고 gossip을 통해 서로를 discovery 하고, 데이터를 sharding 및 replication하여 특정 노드가 죽어도 HA를 보장하는 것) 이 불가능.
만약 Prometheus를 여러 대에 구성하여 사용하려면 Prometheus에 Prometheus를 연결하여 Hierarchy 구조로 만들어야함..
(레퍼런스 : https://prometheus.io/docs/prometheus/latest/federation/)

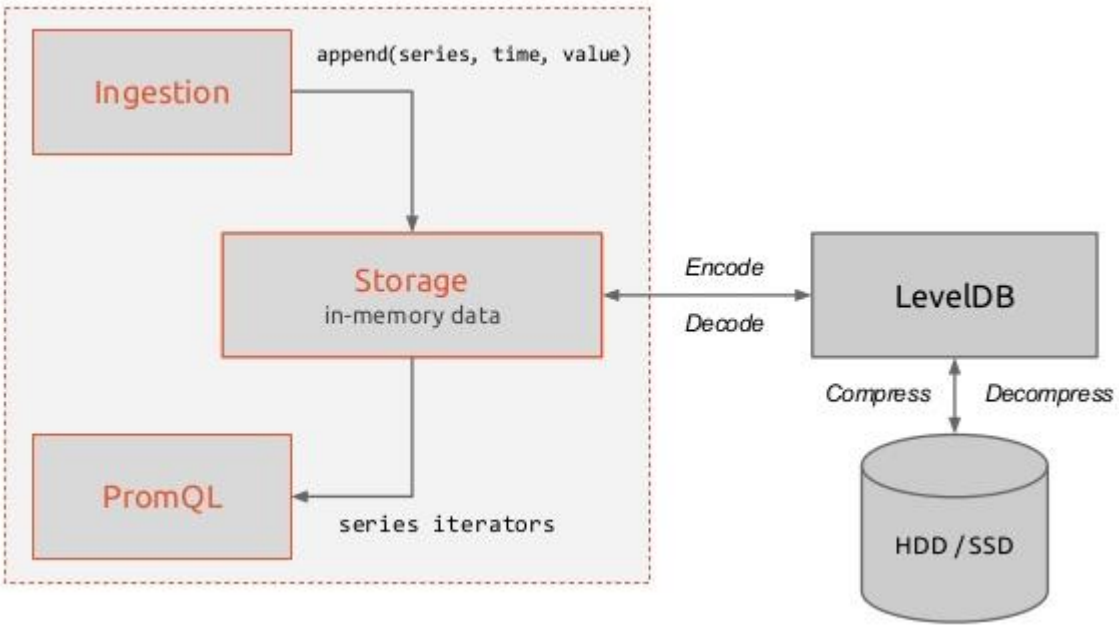


4. 프로메테우스 TSDB 및 Metrics

1) 프로메테우스의 TSDB는 Google Level DB를 활용

Google Level DB : light-weight의 Key-value storage

(https://github.com/google/leveldb)


Ingestion
append(series, time, value)
Storage
in-memory data
PromQL
series iterators
Encode
Decode
LevelDB
Compress
Decompress
HDD / SSD