

# K8s LMA 'venti-stack' 개발기

김정민 - 삼성SDS

Cloud Native Korea  
Community Day 2024



# 발표자



김정민 / 삼성 SDS / Cloud Engineer

Samsung Kubernetes Engine를 개발하는 업무를 하고 있습니다. K8s와 Go 테스트 관련 주제에 관심이 많습니다.

발표자료는? 질문은?

<https://github.com/jmnote/slides>

# 목차

I . venti-stack 소개

II . 왜 만들었나?

III . 어떻게 만들었나?

IV . 사용 방법

# I venti-stack 소개

# K8s LMA란?

K8s 환경에서 Logging/Monitoring/Alerting(로깅/모니터링/알림)을 통합관리하는 시스템/소프트웨어 스택

- Logging: 클러스터/애플리케이션의 로그 데이터 수집/저장/조회
- Monitoring: 클러스터/애플리케이션의 메트릭 데이터 수집/저장/조회
- Alerting: 수집된 데이터에서 특정 조건 발생시 운영자에게 알림 발송

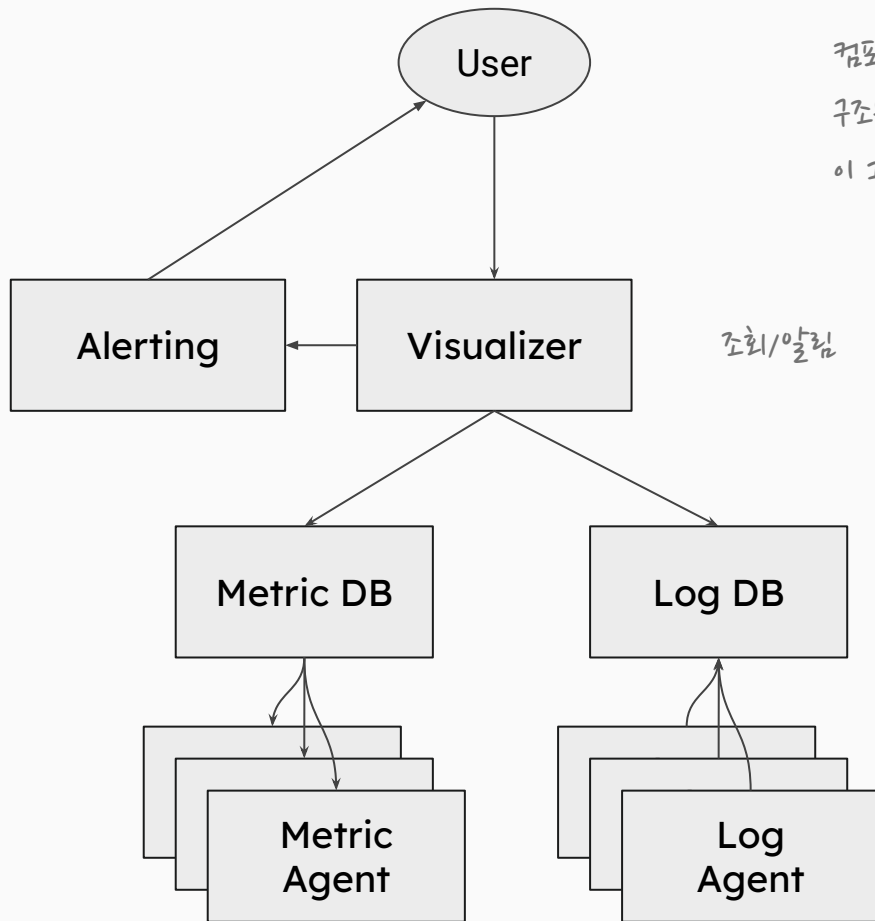
로그 예시

```
$ kubectl logs pod1
[INFO] Starting the application...
[INFO] Successfully connected to the DB.
[ERROR] Failed to connect to the API.
```

메트릭 예시

```
$ kubectl top pod
NAME      CPU(cores)   MEMORY(bytes)
pod1      2m           15Mi
pod2      3500m        16384Mi
```

# k8s LMA 구성도 (개념)



컴포넌트 추가될 수 있으나

구조는 대략 이렇다...

이 그림은 반복해서 나올 예정

조회/알림

저장

수집

# venti-stack이란?

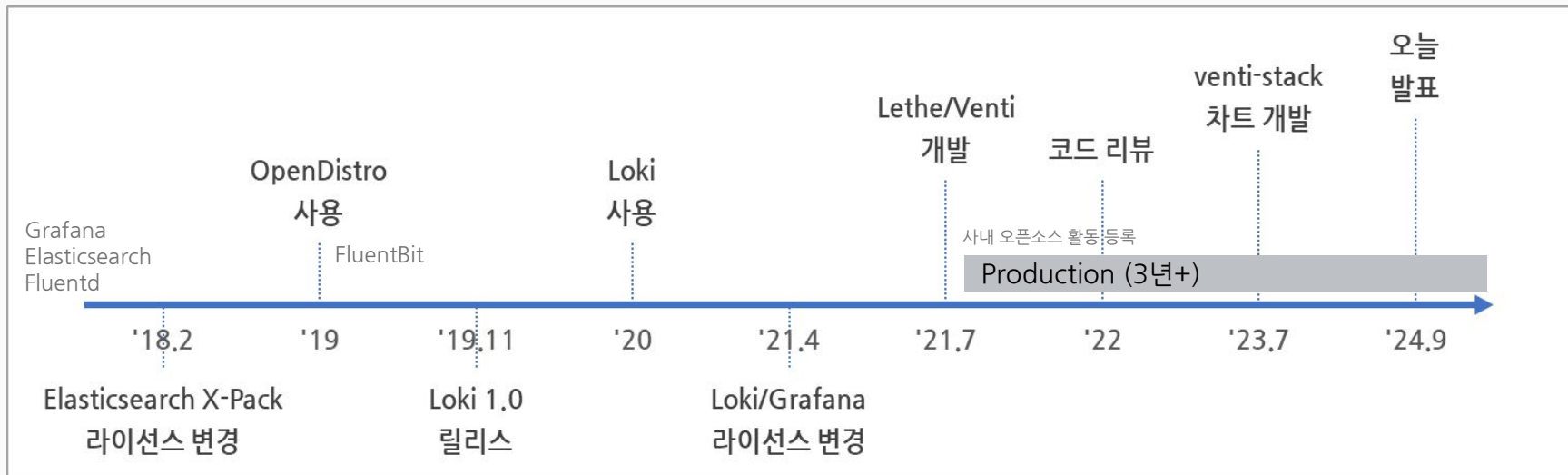
## K8s LMA 스택

- Helm Chart 제공 *LMA 한번에 설치*
- Production 사용 중 *3년+*
- Apache-2.0 라이선스
- 자체 개발 컴포넌트 포함
  - Lethe *LogDB*
  - Venti *visualizer*

 왜 만들었나?

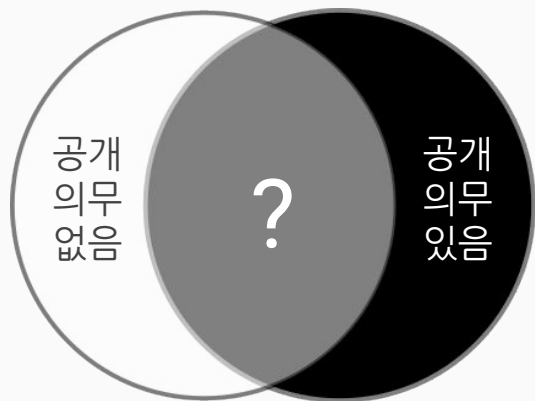


# 타임라인



- 2018-02-28 <https://www.elastic.co/kr/blog/doubling-down-on-open>
- OpenDistro: OpenSearch의 전신
- 2021-04-20 <https://grafana.com/blog/2021/04/20/grafana-loki-tempo-relicensing-to-agplv3/>

# AGPL에 대한 의견들



## 회색 영역

- 바이너리를 그대로 수정 없이 사용하면 괜찮다.<sup>1)</sup>
- 대기업이 AGPL을 피하는 이유 ... 소스코드를 직접 수정하지 않더라도 전체 시스템이 AGPL의 적용을 받을 수 있다.<sup>2)</sup>
- AGPL 소프트웨어를 사용하려면 링크하는 모든 것도 AGPL에 따라 라이선스를 받아야 한다. ... 위험이 이점보다 훨씬 크다.<sup>3)</sup>

결론? 잘 모르겠습니다(회색영역)  
OSS.kr 문의 사례 참고 4) 5) 6) 7)

※ 법적 분쟁 발생시, 본 자료는 법률적 해석이나 논리로 활용될 수 없습니다.

1. <https://medium.com/swlh/understanding-the-agpl-the-most-misunderstood-license-86fd1fe91275>
2. <https://www.signority.com/2024/05/03/is-agpl-a-scam-how-small-companies-can-maximize-benefits-while-remaining-compliant/>
3. <https://opensource.google/documentation/reference/using/agpl-policy/>
4. 2021-09-10 [https://www.oss.kr/oss\\_license\\_qna/show/6c691ee5-bdb3-4f91-ae2a-0e9325f89b8f](https://www.oss.kr/oss_license_qna/show/6c691ee5-bdb3-4f91-ae2a-0e9325f89b8f)
5. 2022-11-17 [https://www.oss.kr/oss\\_license\\_qna/show/cf42621c-2fef-461f-aa45-e28ad3361866](https://www.oss.kr/oss_license_qna/show/cf42621c-2fef-461f-aa45-e28ad3361866)
6. 2023-06-26 [https://www.oss.kr/oss\\_license\\_qna/show/07adc90a-ab99-4a24-9c75-92df081a058e](https://www.oss.kr/oss_license_qna/show/07adc90a-ab99-4a24-9c75-92df081a058e)
7. 2023-07-19 [https://www.oss.kr/oss\\_license\\_qna/show/d6847e91-a21c-4188-8914-7aba5a0a316c](https://www.oss.kr/oss_license_qna/show/d6847e91-a21c-4188-8914-7aba5a0a316c)

# venti-stack의 라이선스

## Apache-2.0

- Permissive 라이선스
- Kubernetes, Prometheus와 같음

## 라이선스 검증 방법

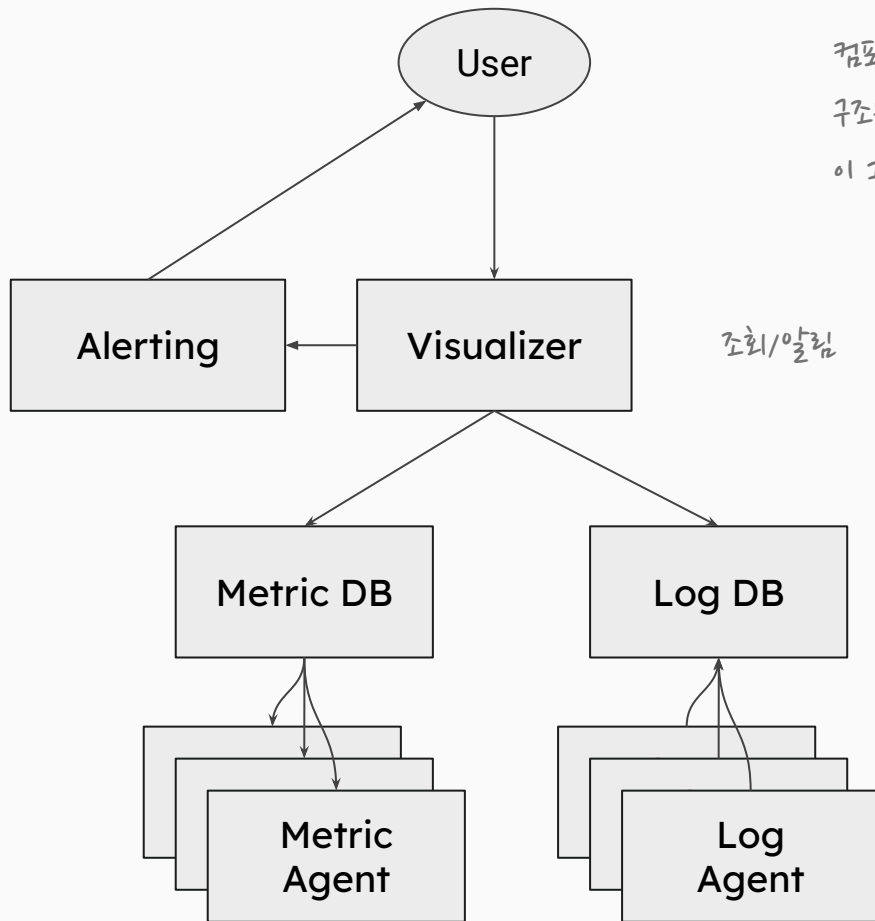
- go-licenses 활용<sup>1)</sup>
  - Google에서 만든 Go용 라이선스 검증 도구
  - 패키지 트리과 라이선스 문서를 분석/보고(report)
  - Google 정책에 따라 금지된 라이선스 확인(check)
- GitHub Actions 라이선스 검증
  - Pull Request 등록시 go-licenses 실행<sup>2) 3)</sup>

1. <https://github.com/google/go-licenses>

2. <https://github.com/kuoss/lethe/blob/v0.2.5/.github/workflows/pull-request.yml>

3. <https://github.com/kuoss/venti/blob/v0.2.20/.github/workflows/pull-request.yml>

# k8s LMA 구성도 (개념)



컴포넌트 추가될 수 있으나

구조는 대략 이렇다...

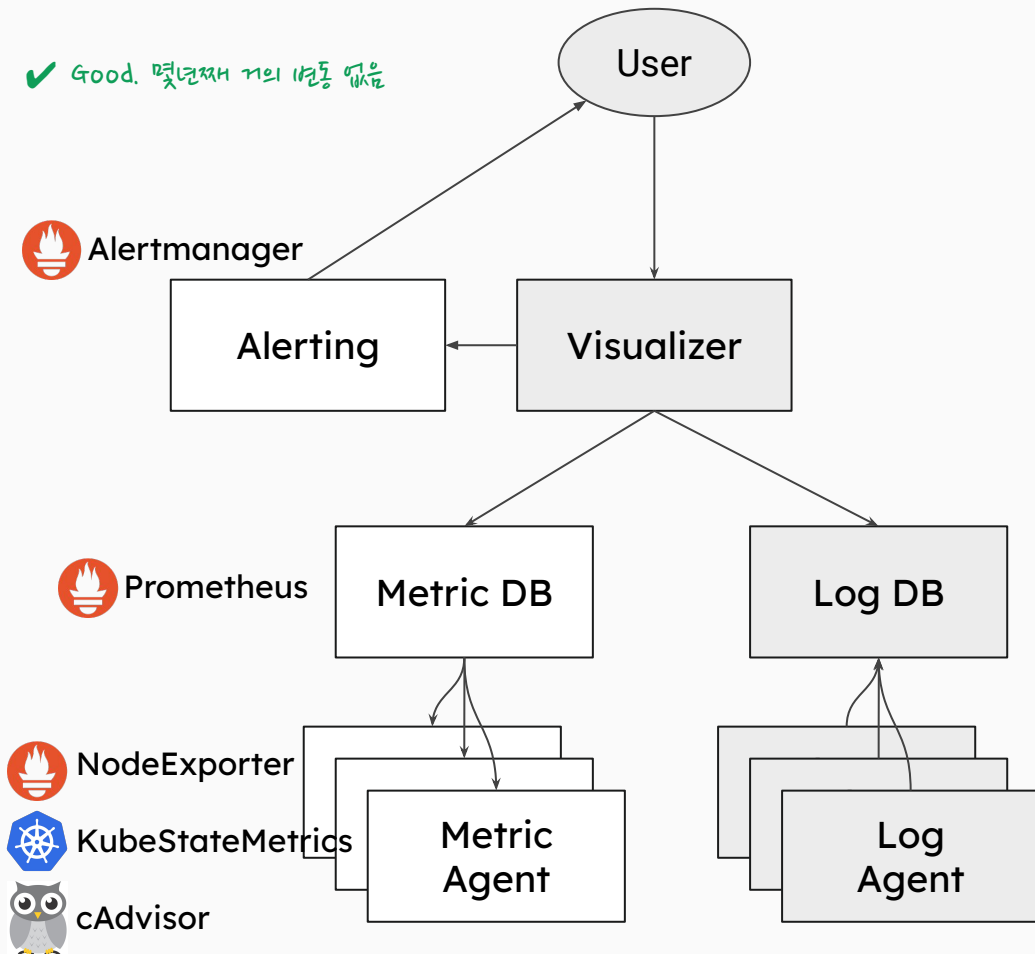
이 그림은 반복해서 나올 예정

조리/알림

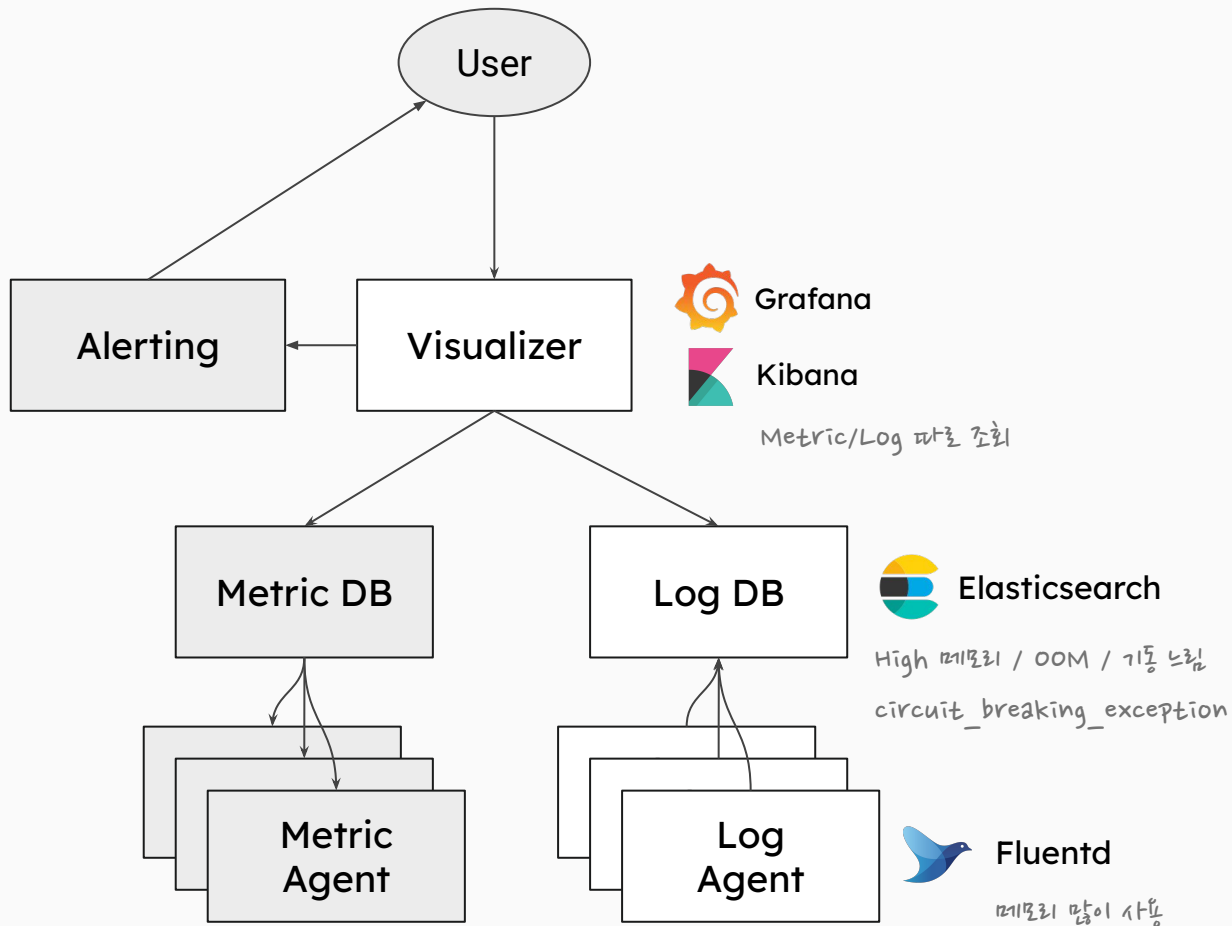
저장

수집

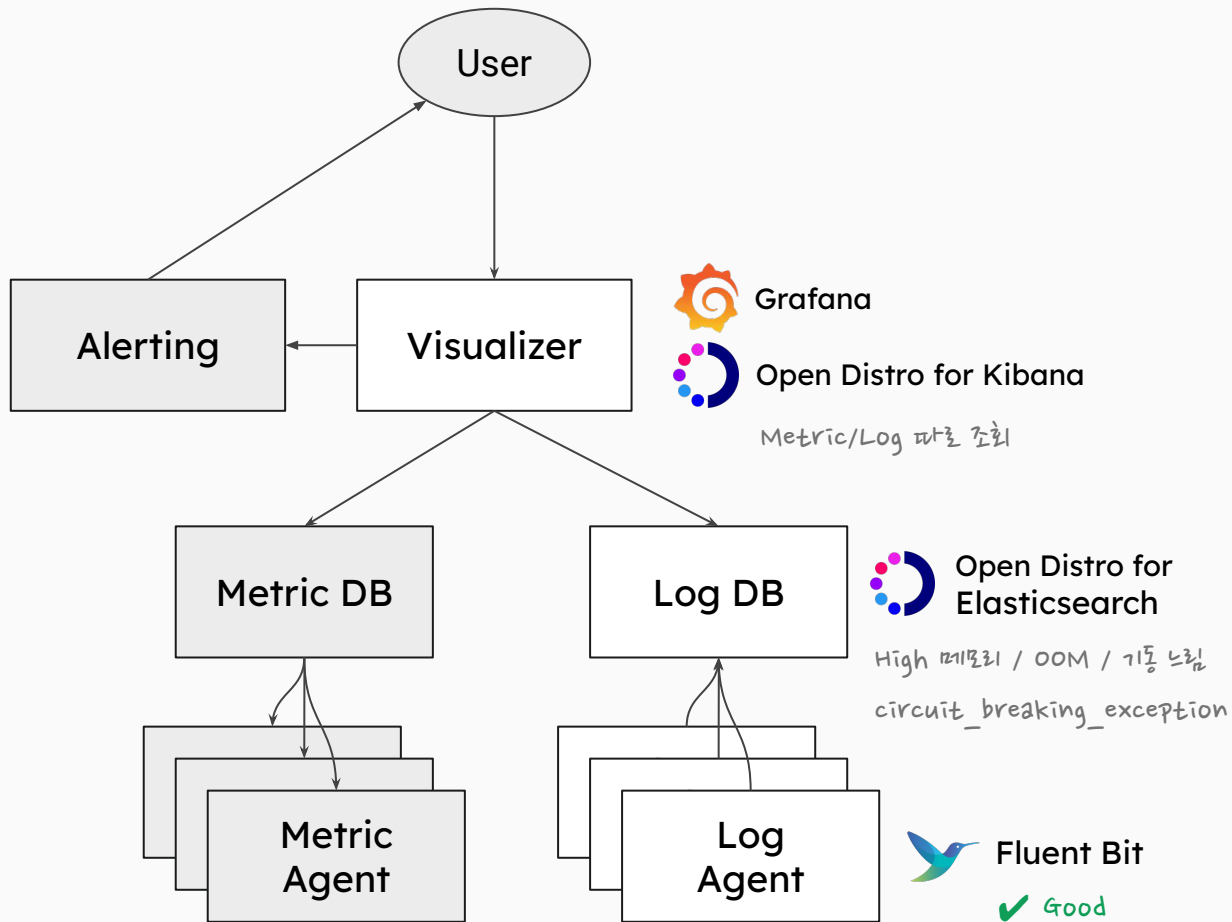
# k8s LMA 구성도 (Metric과 Alerting)



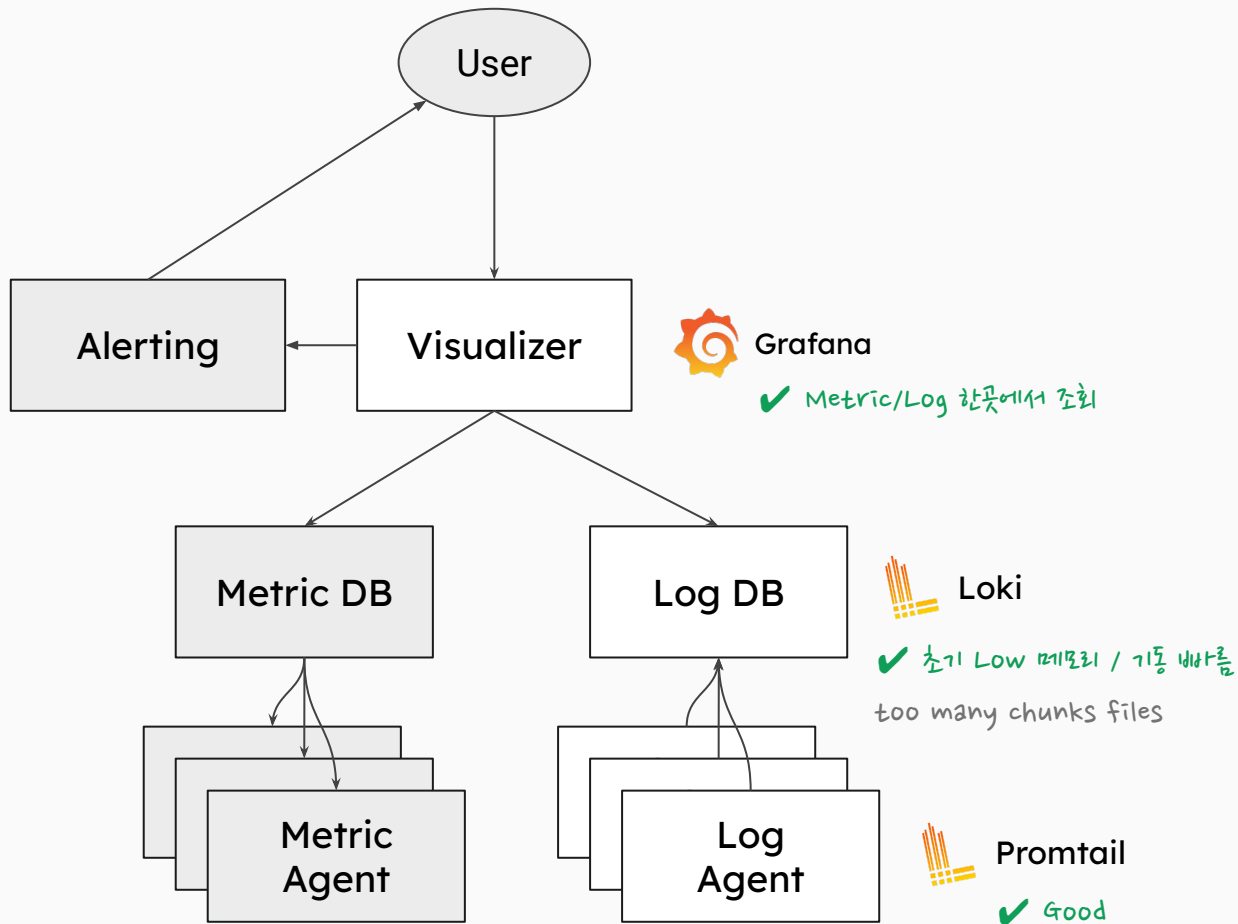
# k8s LMA 구성도 (2018)



# k8s LMA 구성도 (2019)



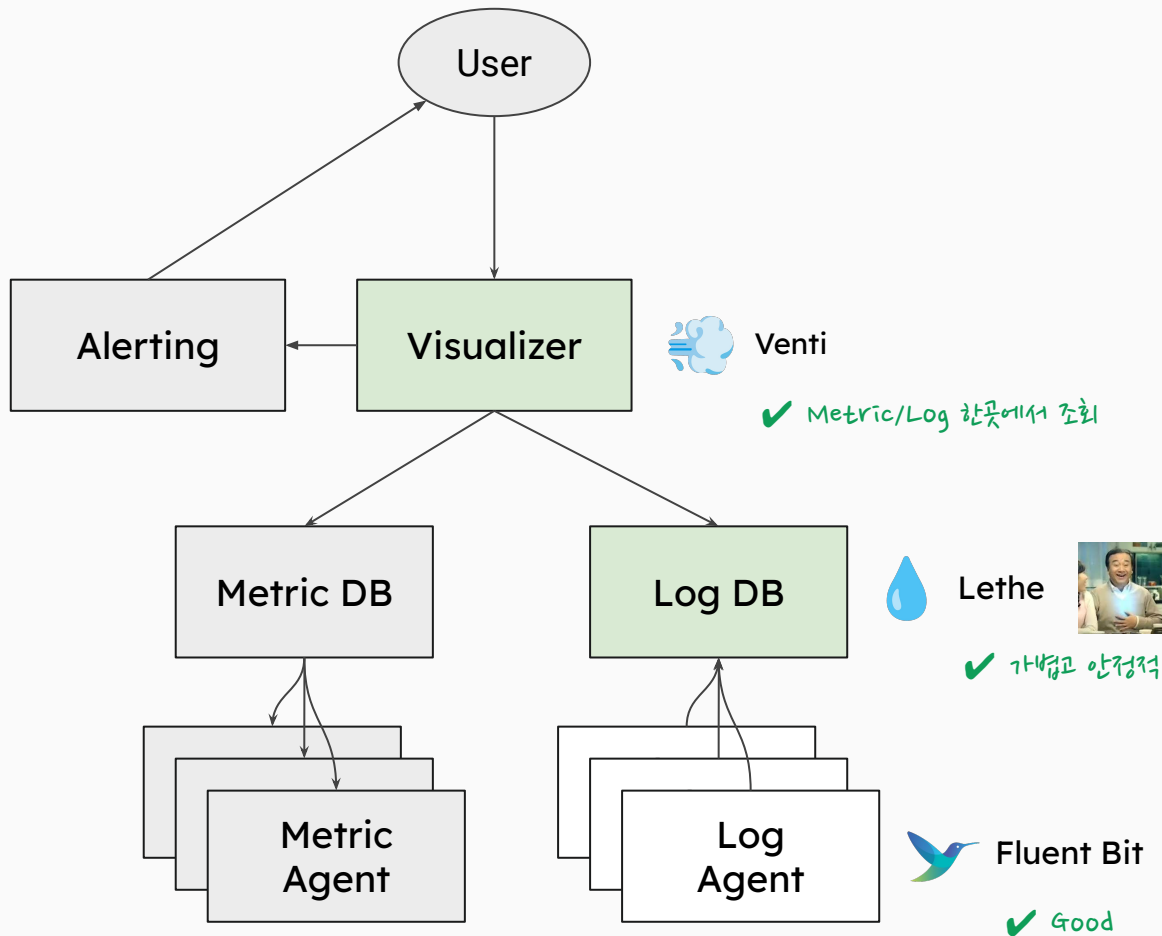
# k8s LMA 구성도 (2020)



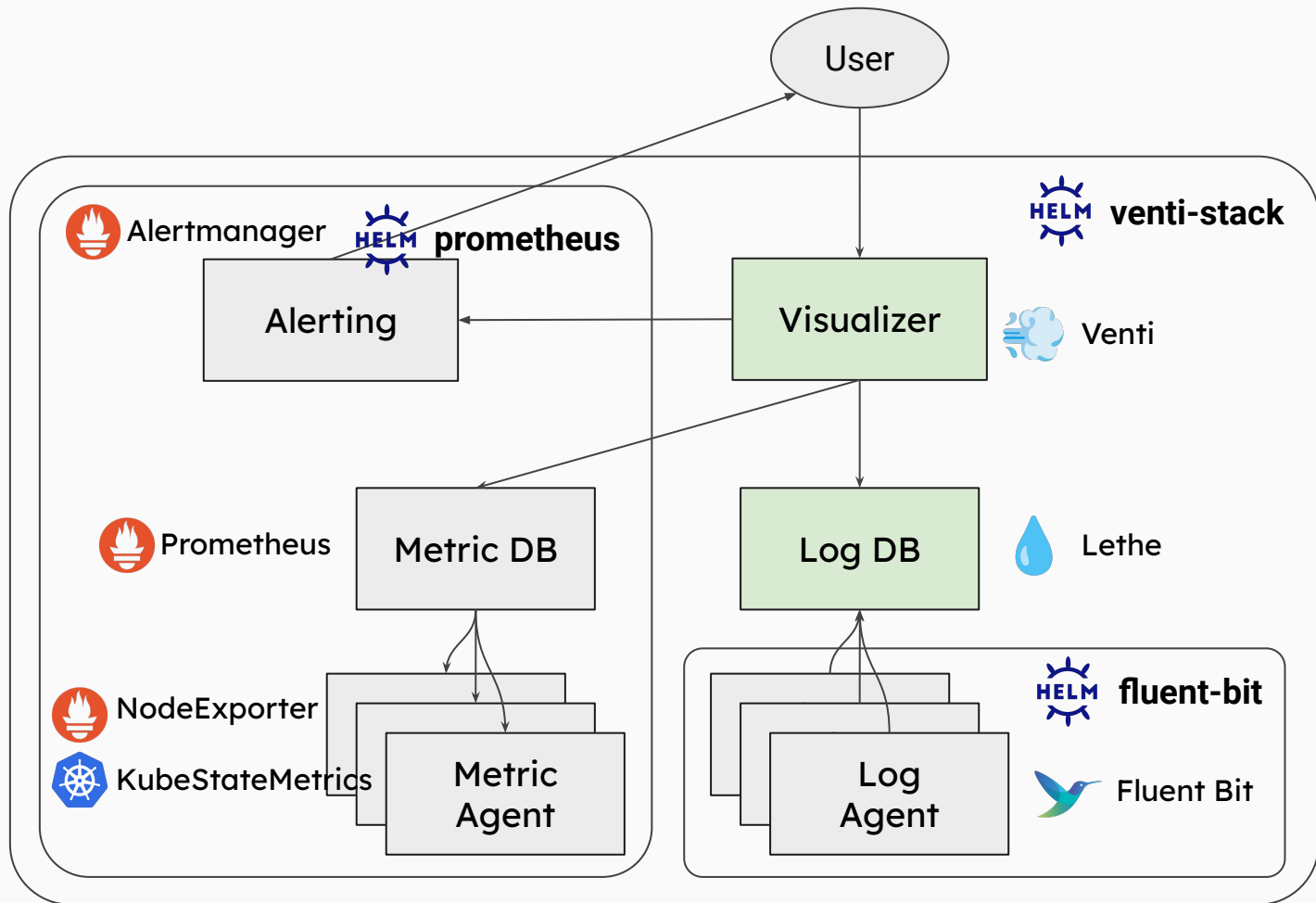


# k8s LMA 구성도 (2021.7~)

✓ Production 3년+



# venti stack 차트 (2023.7~)



# MVP 개발기간은? 이름은 어떻게 지었나?

대략 IMM

'21.6 대안 검토(2주)

'21.7 개발(2주+2주)

'21.8 검증/전환

다음 기준에 따라 아이디어 내고 투표

1) 신화에 나오는 이름

2) Prometheus 상징이 불이니까, LogDB는 물, visualizer는 바람




3) LogDB는 L로, visualizer는 V로 시작

4) 제발 짧은 거. ~~인간적으로 prometheus는 너무 길지~~

그 결과...

```
$ kubectl get VentiStackComponents
```

 이런 명령어 없습니다...

NAME	TYPE	MEANING	SYMBOL
Prometheus	MetricDB	불의 신	 불(햇불)
Lethe	LogDB	망각의 강	 물
Venti	Visualizer	바람의 신	 바람

~~그게만 더 못하면 되겠네~~

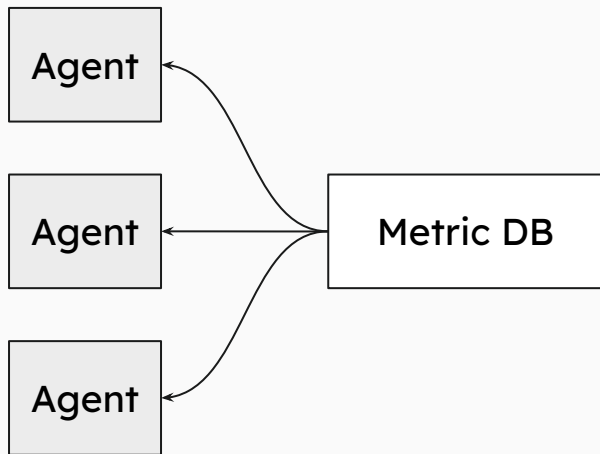


III 어떻게 만들었나?

# 메트릭DB vs 로그DB

메트릭 수집

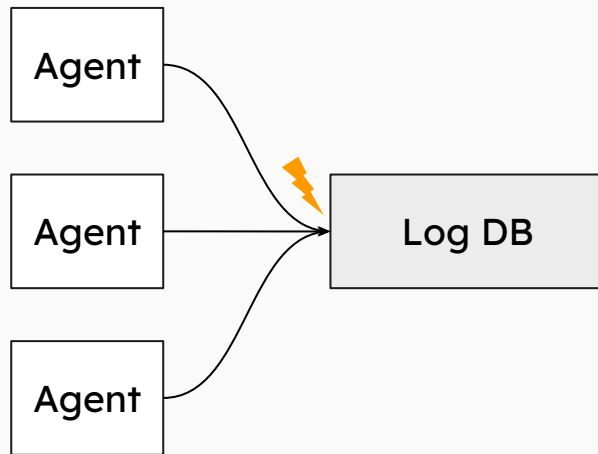
메트릭 저장



MetricDB가 수집주기 조절

로그 수집

로그 저장



과부하 발생 가능성

# Lethe

- 경량 Log DB
- DB이지만 DB가 아니다?
  - 단순히 파일로 저장
  - Fluent Bit 기반
- 기능: 저장, 조회, 로테이션
- LetheQL
  - 필터 연산자
  - 로그 파이프라인



# Fluentd vs Fluent Bit

	Fluentd	Fluent Bit
Scope	Containers / Servers	Embedded Linux / Containers / Servers
Language	C & Ruby	C
Memory	> 60MB	~1MB
Performance	Medium Performance	High Performance
Dependencies	Built as a Ruby Gem, it requires a certain number of gems.	Zero dependencies, unless some special plugin requires them.
Plugins	More than 1000 external plugins are available	More than 100 built-in plugins are available
License	<a href="#">Apache License v2.0</a>	<a href="#">Apache License v2.0</a>

이게 1GB로 로그DB를 경량으로 만든 비결입니다?

Fluent Bit을 로그수집기로 쓰면 로그DB가  
경량화된다는 말인가? (X)

Fluent Bit을 잘 설정해서 로그를  
수집단기에서 경량화해서 로그DB의 부담을  
줄였다는 말인가? (△)

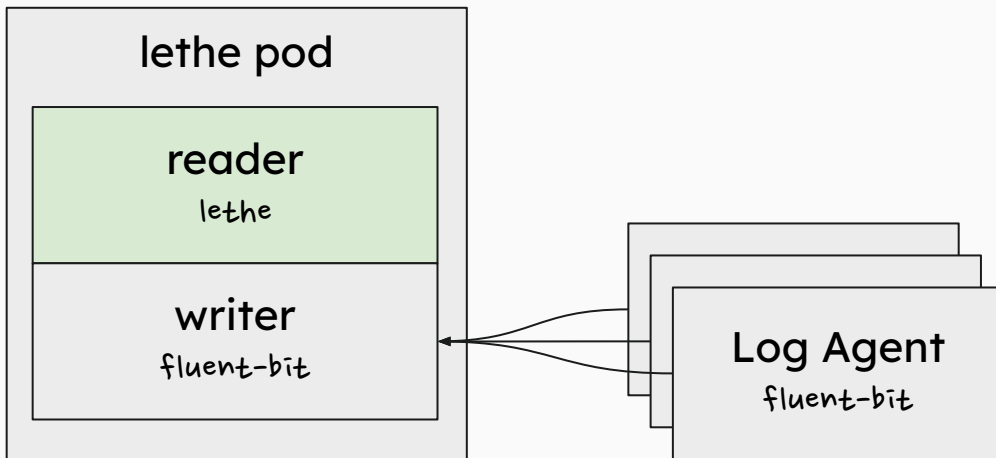
<https://docs.fluentbit.io/manual/about/fluentd-and-fluent-bit>

# Lethe 구성도 (컨테이너 수준)

사이트카 구조

$\text{lethe}(\text{pod}) = \text{lethe} + \text{fluent-bit}$

lethe는 쉬운 일(읽기)만 하고,  
fluent-bit이 힘든 일(쓰기)을 한다.



‘수집-저장’ 과정만 보면

fluent-bit끼리 데이터를 주고 받는 것



# 로그 저장 방식

파일경로 filesystem friendly

node/	node01/	2024-09-24_15.log
	node02/	2024-09-24_15.log
pod/	namespace01/	2024-09-24_15.log
	namespace02/	2024-09-24_15.log

노드/네임스페이스 디렉토리 + 한시간.log

파일내용 human-readable

```
2024-09-24T15:00:01Z[node01|kubelet] hello world
2024-09-24T15:00:02Z[node01|kubelet] lorem ipsum
2024-09-24T15:00:03Z[node01|containerd] hello world
```

날짜시간[노드|서비스] 로그내용

```
2024-09-24T15:00:01Z[namespace01|pod1|container1] hello
2024-09-24T15:00:02Z[namespace01|pod1|container2] lorem
2024-09-24T15:00:03Z[namespace01|pod2|container1] hello
```

날짜시간[네임스페이스|파드|컨테이너] 로그내용

이런 것도 DB라고 할 수 있나요?

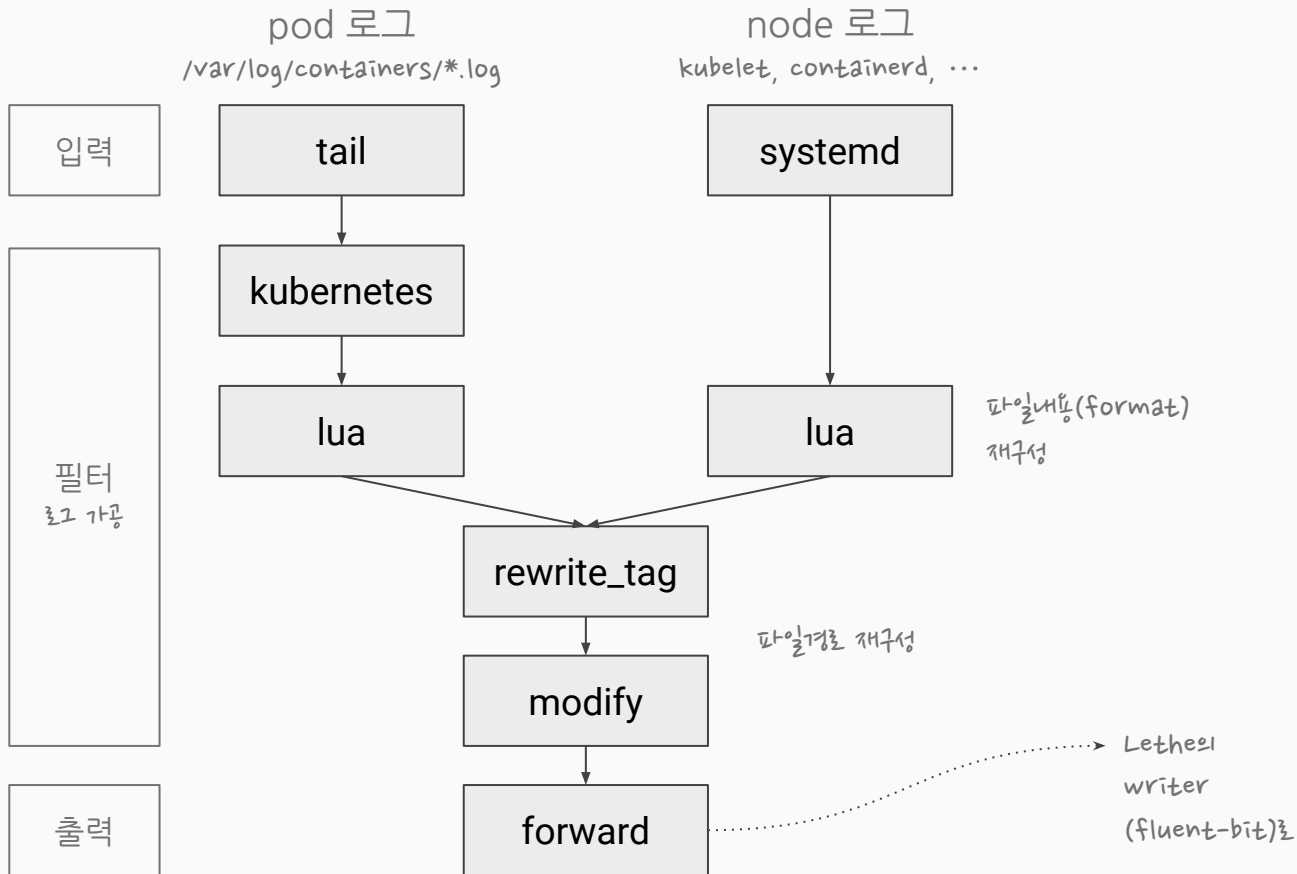
- 1) DB 기능을 하면 DB
- 2) 일반적 DB는 아니고 flat-file DB

다른 특징은?

- label 정보 없음 (pod이름으로 충분)
- 매우 안전 (단순한 로그 파일)

# 로그 수집

로그수집기(fluent-bit)  
내부 파이프라인



<https://github.com/kuoss/helm-charts/blob/venti-stack-0.2.10/charts/venti-stack/values.yaml#L16-L91>

<https://github.com/kuoss/helm-charts/blob/venti-stack-0.2.10/charts/venti-stack/templates/lethe/cm.yaml#L28-L38>

# LetheQL #1 기본 쿼리

Lethe에서 로그를 조회하기 위한 쿼리 언어  
PromQL(Prometheus Query Language)의 확장판 (소스코드 재사용)

```
$ kubectl logs nginx-66b6c48dd5-abcde  
127.0.0.1 - - [08/Sep/2024:12:34:56 +0000] "GET / HTTP/1.1" 200 612 "-" ...
```

→ `pod{namespace="default",pod="nginx-66b6c48dd5-abcde"}`

```
$ kubectl logs deploy/nginx  
127.0.0.1 - - [08/Sep/2024:12:34:56 +0000] "GET / HTTP/1.1" 200 612 "-" ...
```

→ `pod{namespace="default",pod=~"nginx-.*"}`

# LetheQL #2 필터 연산자

!= 문자열 포함  
!= 문자열 미포함  
!~ 정규식 포함  
!~ 정규식 미포함

필터 연산자 4종 지원 (Loki의 LogQL에서 참고)  
파이프라인 연결 가능

```
$ kubectl logs deploy/nginx | grep -v POST | grep favicon
127.0.0.1 - - [08/Sep/2024:12:34:57 +0000] "GET /favicon.ico ...
```

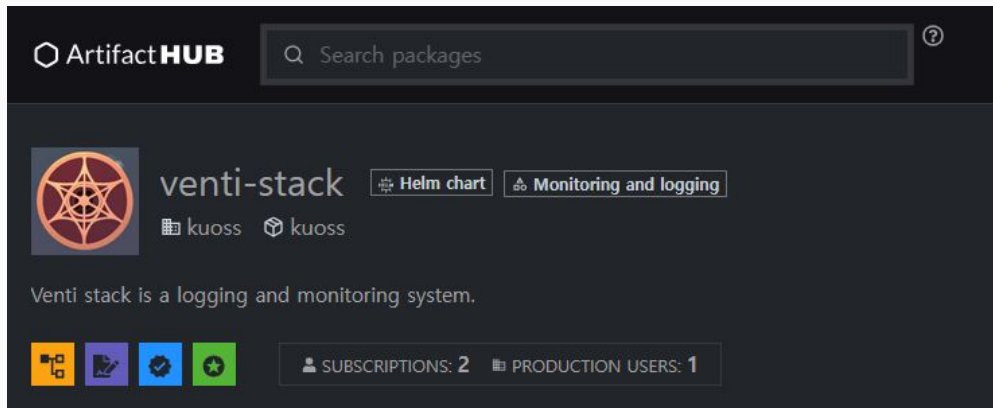
→ `pod{namespace="default",pod=~"nginx-.*"} != "POST" != "favicon"`

```
$ kubectl logs deploy/nginx | grep -v -E '(GET|POST)' | grep -E 'user/[0-9]+'
127.0.0.1 - - [08/Sep/2024:12:34:56 +0000] "DELETE /user/456 ...
```

→ `pod{namespace="default",pod=~"nginx-.*"} != "(GET|POST)" !~ "user/[0-9]+"`

# IV 사용 방법

# 설치 방법



→ 검색엔진(또는 ArtifactHUB)에서 'venti-stack' 검색

```
helm repo add kuoss https://kuoss.github.io/helm-charts
helm repo update
helm install vs --namespace=venti-stack kuoss/venti-stack
```

→ values.yaml에 prometheus.ingress와 venti.ingress 설정하면 좋겠지?

※ kuoss = kubernetes open source software

# Venti



- Visualizer
  - YAML 대시보드 설정
  - 패널: 스탯, 파이차트, 시계열, 로그
  - 마우스오버 데이터테이블
  - 시간 선택기, Auto Refresh
- 데이터 소스
  - 유형: Prometheus, Lethe
  - 멀티 데이터소스 지원
  - 데이터소스 발견  $\asymp$  service discovery
- 메트릭/로그 기반 Alerting



Metrics

Logs

Dashboards (4)

Cluster

Control Plane

Ingress

Node

Alert

Datasource

Status

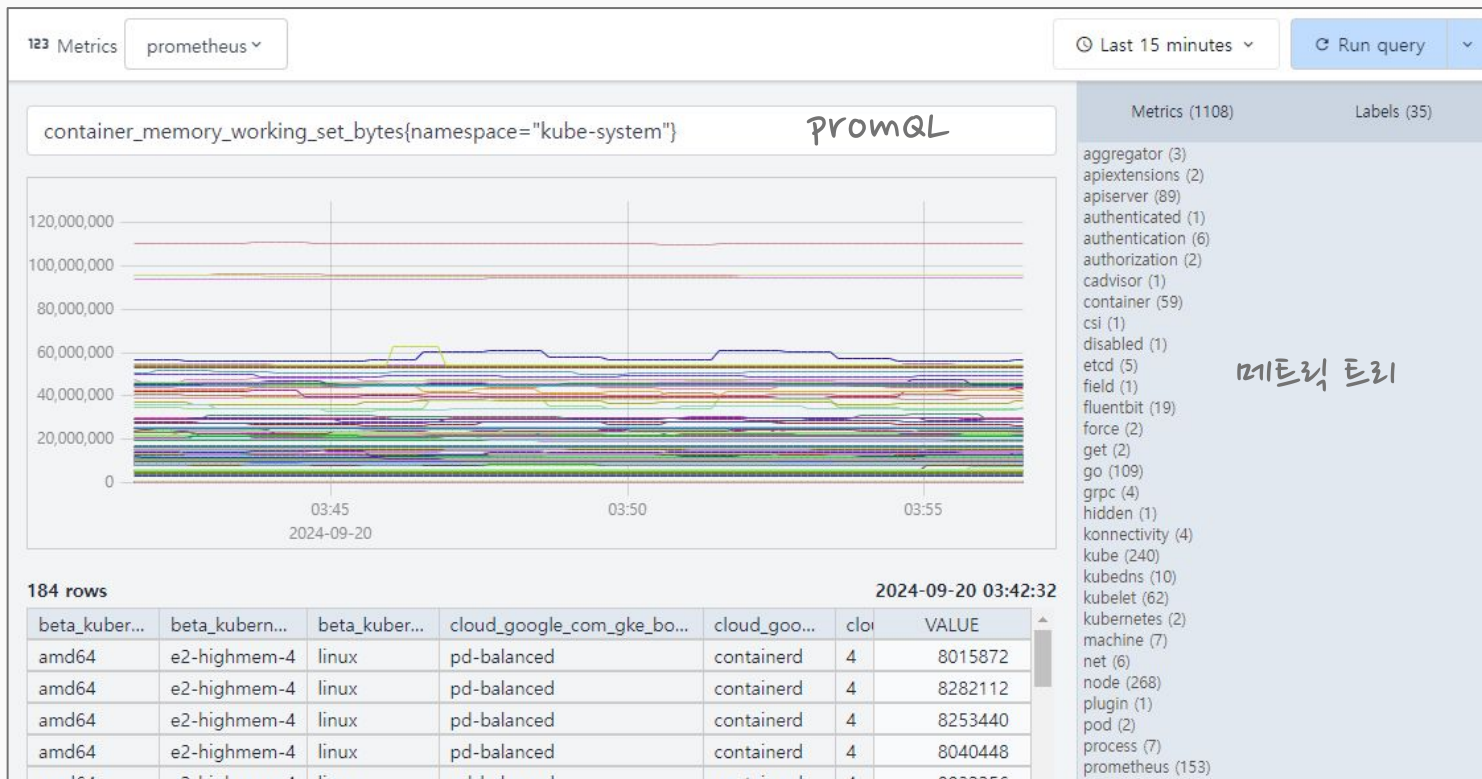
light

dark

Logout

데이터소스 선택기

시간 선택기







Metrics

Logs

Dashboards (4)

Cluster

Control Plane

Ingress

Node

Alert

Datasource

Status

light

dark

Logout

데이터소스 선택기

시간 선택기

Logs

lethe ▾

now-60m to now-30m ▾

Run

pod{namespace="kube-system", pod="calico-node-tfml4"}

LetheQL

2024-09-20 02:34:17 - 03:04:17 pod{namespace="kube-system", pod="calico-node-tfml4"} 209 rows

2024-09-20 02:34:19 kube-system calico-node-tfml4 calico-node 2024-09-19 17:34:19.837 [INFO][62] felix/summary.go 10

0: Summarising 9 dataplane reconciliation loops over 1m7.4s: avg=18ms longest=126ms ()

2024-09-20 02:35:23 kube-system calico-node-tfml4 calico-node 2024-09-19 17:35:23.231 [INFO][62] felix/summary.go 10

0: Summarising 12 dataplane reconciliation loops over 1m3.4s: avg=15ms longest=120ms ()

2024-09-20 02:36:26 kube-system calico-node-tfml4 calico-node 2024-09-19 17:36:26.169 [INFO][62] felix/summary.go 10

0: Summarising 11 dataplane reconciliation loops over 1m2.9s: avg=6ms longest=16ms (resync-filter-v4)

2024-09-20 02:37:26 kube-system calico-node-tfml4 calico-node 2024-09-19 17:37:26.229 [INFO][62] felix/summary.go 10

0: Summarising 8 dataplane reconciliation loops over 1m0.1s: avg=20ms longest=127ms ()

2024-09-20 02:38:31 kube-system calico-node-tfml4 calico-node 2024-09-19 17:38:31.186 [INFO][62] felix/summary.go 10

0: Summarising 11 dataplane reconciliation loops over 1m5s: avg=17ms longest=137ms ()

2024-09-20 02:39:34 kube-system calico-node-tfml4 calico-node 2024-09-19 17:39:34.639 [INFO][62] felix/summary.go 10

0: Summarising 11 dataplane reconciliation loops over 1m3.5s: avg=6ms longest=16ms (resync-filter-v4)

2024-09-20 02:40:00 kube-system calico-node-tfml4 calico-node 2024-09-19 17:40:00.617 [INFO][62] felix/calc\_graph.go

467: Local endpoint updated id=WorkloadEndpoint(node=gke-cluster1-worker5-9fdbf910-ibtd, orchestrator=k8s, workload=cron/analytics-28779460-xg7kn, name=eth0)

2024-09-20 02:40:00 kube-system calico-node-tfml4 calico-node 2024-09-19 17:40:00.617 [INFO][62] felix/int\_dataplan

e.go 1836: Received \*proto.ActiveProfileUpdate update from calculation graph msg=id:<name:"kns.cron" > profile:<inb

ound\_rules:<action:"allow" rule\_id:"W9dpr0z81voikY4p" > outbound\_rules:<action:"allow" rule\_id:"72zes54CurlQHK\_n" >

>

2024-09-20 02:40:00 kube-system calico-node-tfml4 calico-node 2024-09-19 17:40:00.617 [INFO][62] felix/table.go 508:

Provider update of table: kube-system/28779460-xg7kn, felix/summary.go 10

Targets

node

gke-cluster1-pool-1-b04aaa76-2tqc

gke-cluster1-pool-1-b04aaa76-jsp1

gke-cluster1-pool-1-b04aaa76-z59d

gke-cluster1-worker5-9fdbf910-ibtd

pod

> cert-manager

> cron

> default

> dev

> dev2

> devpod

> docker

> gke-managed-system

> ingress-annotator

> ingress-nginx

> kube-node-lease

> kube-public

> kube-system

> deployment (10)

> daemonset (29)

> pod (51)

calico-node-tfml4

calico-node-vertical-autoscaler-774f66

워크로드 트리

# Venti

## 대시보드

대시보드 설정 YAML 파일

<https://github.com/kuoss/helm-charts/blob/v%20ent%20stack-0.2.10/charts/venti-stack/templates/venti-cm-dashboards.yaml>

venti

Metrics

Logs

Dashboards (4)

Cluster

Control Plane

Ingress

Node

Alert

Datasource

Status

light

dark

Logout

All namespaces

All nodes

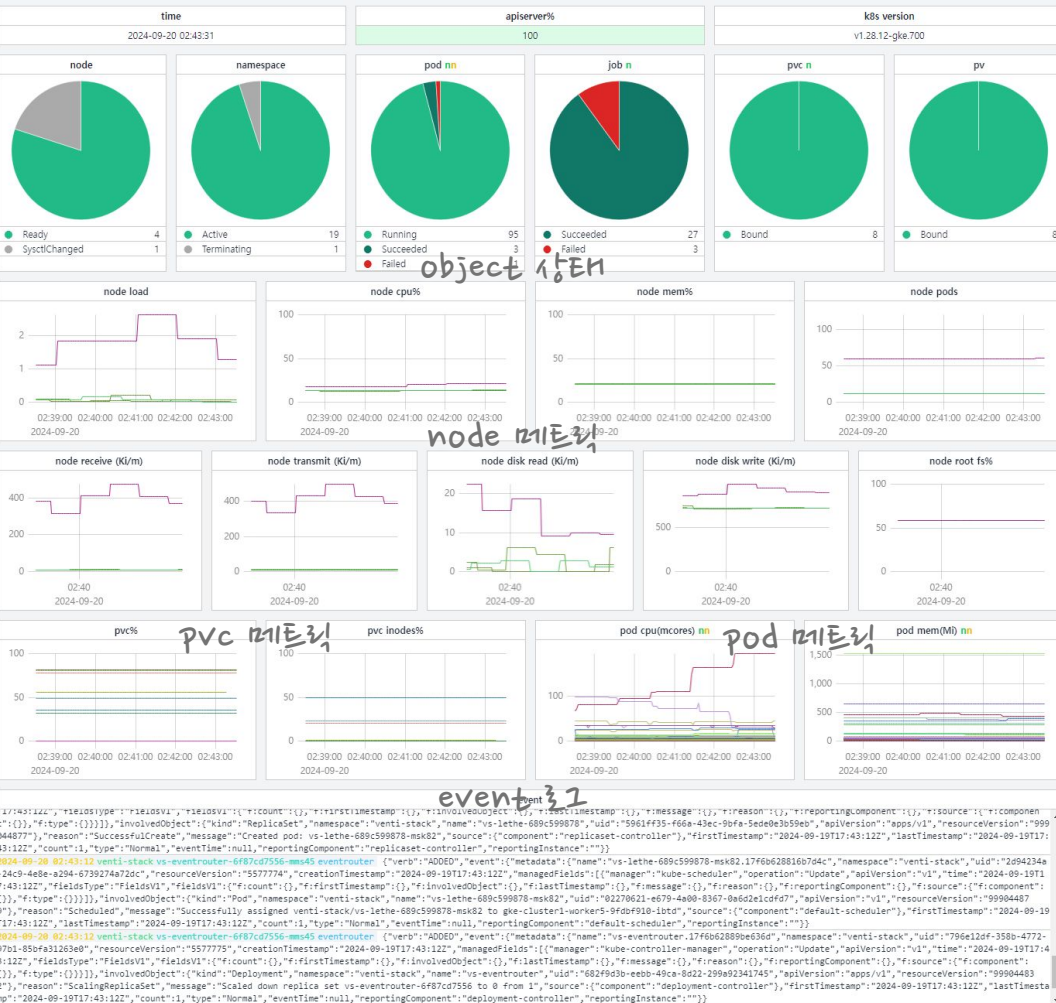
namespace/node 선택기

Cluster

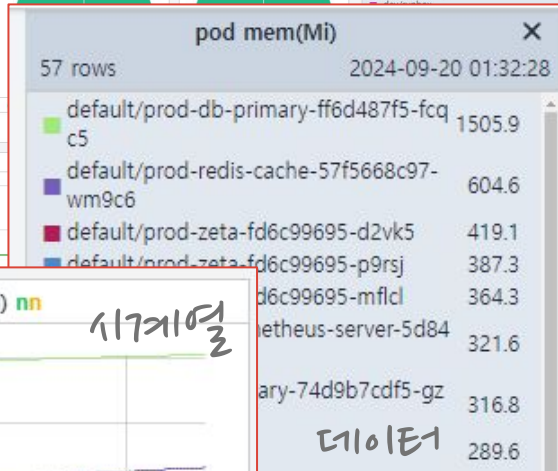
시간 선택기

Last 5 minutes

Refresh



데이터 테이블



디자이너

31

```
117:43:12Z","fieldsType":"FieldsV1","fieldsV1":{"+":count:{},"+":firstTimestamp:{},"+":involve  
t":{},"f:type":{}}}}, "involvedObject":{"kind":"ReplicaSet","namespace":"venti-stack","name"  
044877"},"reason":"SuccessfulCreate","message":"Created pod: vs-lethe-689c599878-msk82","sour  
43:12Z","count":1,"type":"Normal","eventTime":null,"reportingComponent":"replicaset-controlle  
2024-09-20 02:43:12 venti-stack vs-eventrouter-f6f87cd7556-mms45 eventrouter {"verb":"ADDED"}  
-24c9-4e8e-a294-6739274a72dc","resourceVersion":"5577774","creationTimestamp":"2024-09-19T17:
```

# EOF

## venti-stack 요약

- Helm Chart 하나로 LMA 설치
- Production 사용 중 3년+, 가볍고 안정적
- Permissive 라이선스

사용/문의/참여 환영합니다. dozer-jang님 코트리뷰 감사합니다.

## What's next

- venti-stack Go 코드 설명 10/12 Gophercon Korea
- venti-stack Helm Chart 개발기 TBD
- docs 작성, venti TypeScript 전환 진행중
- LogDB 성능 비교 검증 이론적·경험적 경향을 정량적 검증