

# 2020 IT 21

## Global Conference

Digital New Deal  
Technology Essentials  
디지털 뉴딜 기술 핵심

## Keynote Speech 2

Leveraging Open Source Platforms for  
Elastic & Robust Services in NAVER

김태웅 책임리더 (네이버)



### [요약문]

COVID-19로 인해 사회 모든 분야에서 오프라인보다 비대면 온라인의 비중이 확대되고 있다. 온라인 서비스는 급격하게 증가하는 사용자 동시요청을 수용할 수 있도록 탄력성(Elasticity)을 제공해야 하며, 서비스 연속성을 보장하는 강인성(Robustness)을 제공하는 것이 중요하다. 본 발표에서는 네이버 서비스를 안정적으로 제공하기 위해 백엔드 플랫폼 계층을 오픈 소스 플랫폼들을 활용하여 어떻게 구축하고 있는지, 그 과정에서 해결해야 할 문제들은 어떤 것들이 있는지 살펴본다.

### [발표자 약력]

1993년 서울대학교 컴퓨터공학 학사

1995년 서울대학교 컴퓨터공학 석사

2000년 서울대학교 컴퓨터공학 박사

2000년~2006년 데이터코러스(주) CTO

2006년~현재 네이버 Platform Labs 책임리더 클라우드플랫폼개발랩장, 서비스플랫폼개발센터장, 기술플랫폼위원회 총괄 역임

관심분야 : 분산 파일시스템, 분산 컴퓨팅, 서버 가상화, 플랫폼 소프트웨어 등

# Leveraging Open Source Platforms for Elastic & Robust Services in **NAVER**

IT21 글로벌 컨퍼런스 (2020. 09. 24)

김태웅

**NAVER**



## 네이버에서 하고 있는 일

### Platform Labs 조직에서 플랫폼들의 개발을 리딩

- 컨테이너 기술을 기반으로 컴퓨팅 자원을 제공하는 플랫폼
- 데이터를 안정적이고 비용효율적으로 저장하는 대용량 분산 스토리지 플랫폼
- 언제/어디서/무엇 때문에 문제가 발생했는지 직관적으로 이해하고 분석하는 플랫폼
- 폭발적으로 증가하는 온라인 결제 트래픽을 문제없이 처리하는 차세대 페이 플랫폼
- 서비스의 개발과 운영에 필요한 각종 도구 플랫폼

# Elastic and Robust Service

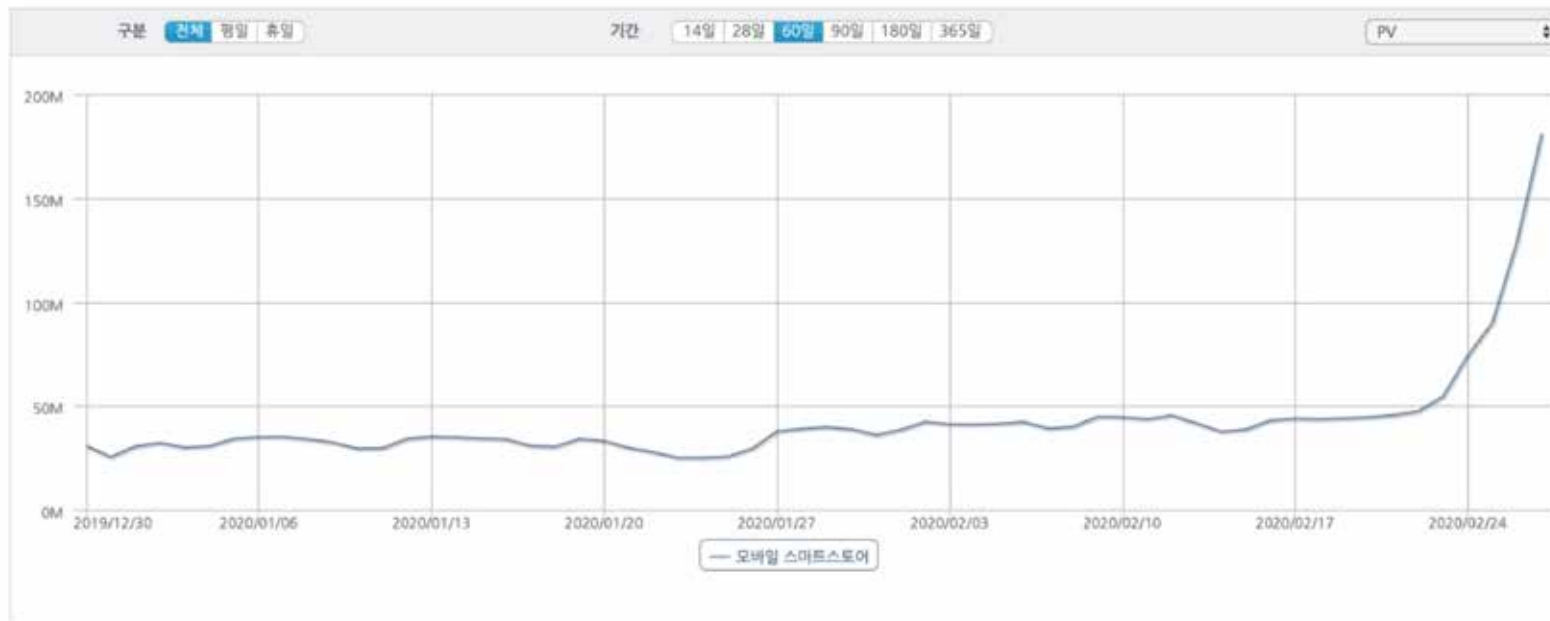
---

## Elastic and Robust Service

Elastic? Not just scalability

Robust? Not just high availability through redundancy

## 사례 1) 2월말경 스마트스토어 트래픽의 급증



전체 평균 PV 4-5 배 수준으로 상승

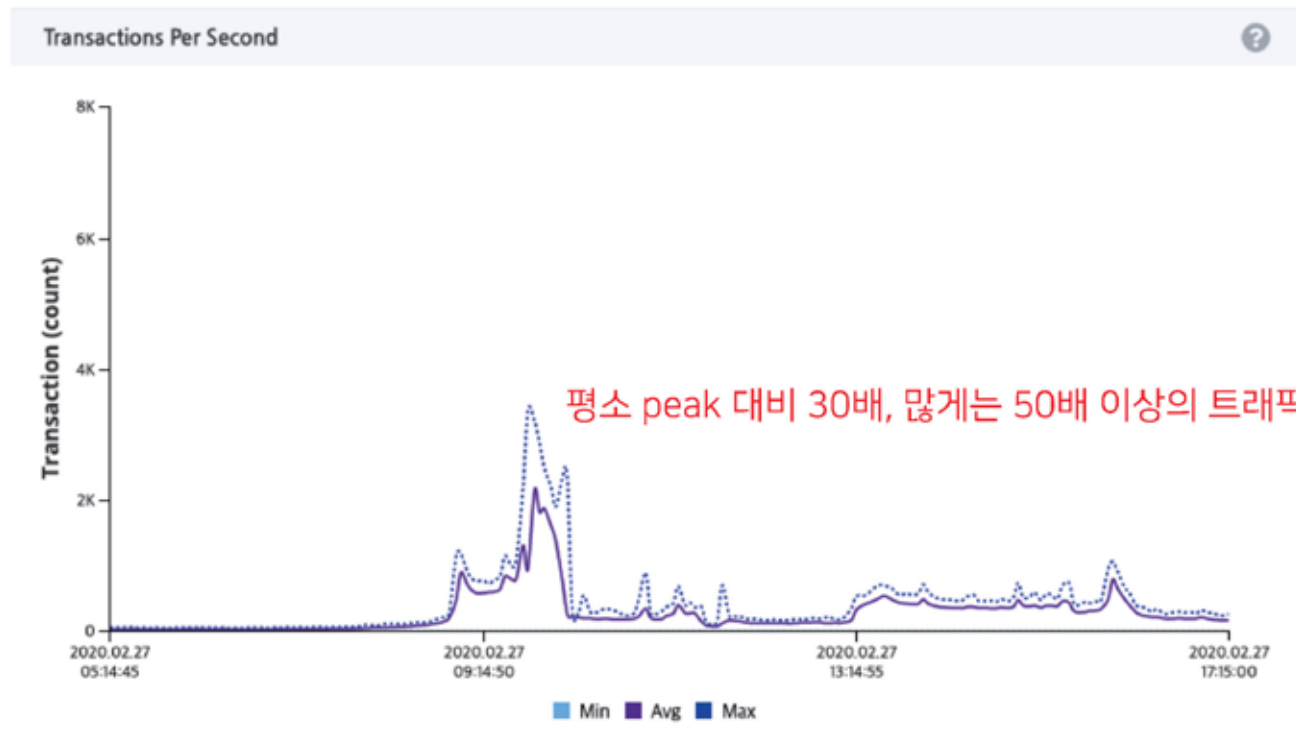
## 사례1) 트래픽 폭증의 원인?

**마스크**  
손 세정제 등 위생용품

마스크를 구매하려는  
소비자들의 필사적 노력의 결과



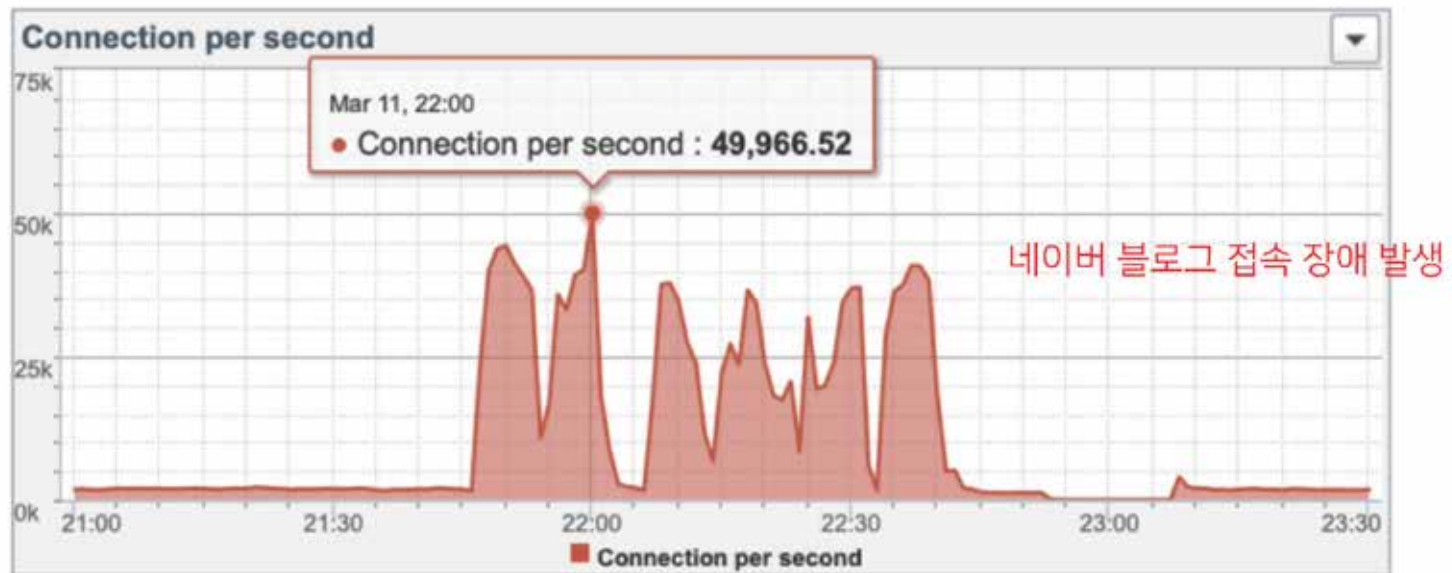
## 사례1) 마스크 상품이 등록되는 시점에 접속장애 발생



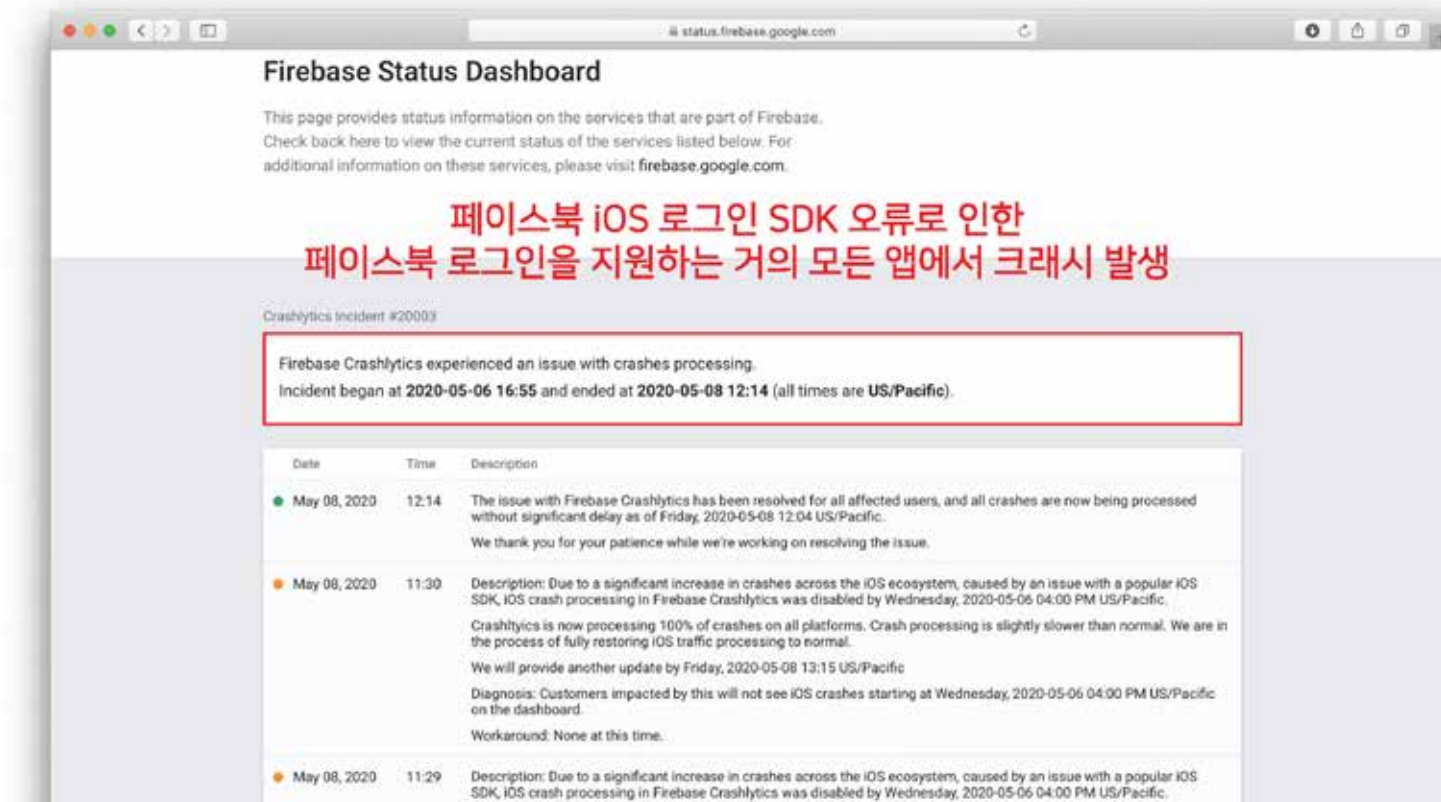


## 사례2) 블로그 접속 요청 폭주

12개 지자체/공공기관 블로그 접속 요청 폭주 (평상시의 28배 이상 증가)



## 사례3) Firebase Crashlytics 장애



The screenshot shows the Firebase Status Dashboard in a web browser. The page title is "Firebase Status Dashboard". Below the title, there is a brief description of the dashboard's purpose. A prominent red banner in the center of the page reads: "페이스북 iOS 로그인 SDK 오류로 인한 페이스북 로그인을 지원하는 거의 모든 앱에서 크래시 발생" (Crashes occurring in almost all apps supporting Facebook login due to Facebook iOS login SDK error). Below this banner, a specific incident is highlighted with a red border. The incident is titled "Crashlytics Incident #20003" and states: "Firebase Crashlytics experienced an issue with crashes processing. Incident began at 2020-05-06 16:55 and ended at 2020-05-08 12:14 (all times are US/Pacific)." Below the incident summary, there is a table with three columns: Date, Time, and Description. The table contains three entries, each with a status icon (green, orange, and orange respectively).

**Crashlytics Incident #20003**

Firebase Crashlytics experienced an issue with crashes processing.  
Incident began at 2020-05-06 16:55 and ended at 2020-05-08 12:14 (all times are US/Pacific).

Date	Time	Description
May 08, 2020	12:14	The issue with Firebase Crashlytics has been resolved for all affected users, and all crashes are now being processed without significant delay as of Friday, 2020-05-08 12:04 US/Pacific. We thank you for your patience while we're working on resolving the issue.
May 08, 2020	11:30	Description: Due to a significant increase in crashes across the iOS ecosystem, caused by an issue with a popular iOS SDK, iOS crash processing in Firebase Crashlytics was disabled by Wednesday, 2020-05-06 04:00 PM US/Pacific. Crashlytics is now processing 100% of crashes on all platforms. Crash processing is slightly slower than normal. We are in the process of fully restoring iOS traffic processing to normal. We will provide another update by Friday, 2020-05-08 13:15 US/Pacific Diagnosis: Customers impacted by this will not see iOS crashes starting at Wednesday, 2020-05-06 04:00 PM US/Pacific on the dashboard. Workaround: None at this time.
May 08, 2020	11:29	Description: Due to a significant increase in crashes across the iOS ecosystem, caused by an issue with a popular iOS SDK, iOS crash processing in Firebase Crashlytics was disabled by Wednesday, 2020-05-06 04:00 PM US/Pacific.

## Elastic and Robust Service

미리 계획된 범위내에서는 늘어나는 요청에 따라 시스템의 처리량을 자동으로 늘려주고

Container 기술을 활용한 시스템 구성요소들의 Autoscaling



CLOUD NATIVE  
COMPUTING FOUNDATION

임계점을 넘어서는 사용자 요청의 유입시 전면 접속 장애에 빠지지 않도록 트래픽을 제어하고 서비스의 응답성을 유지

시스템 상태/자원에 대한 Observability

PINPOINT



비동기식 요청 처리 구조



임계점을 넘어서는 요청에 대한 Admission control

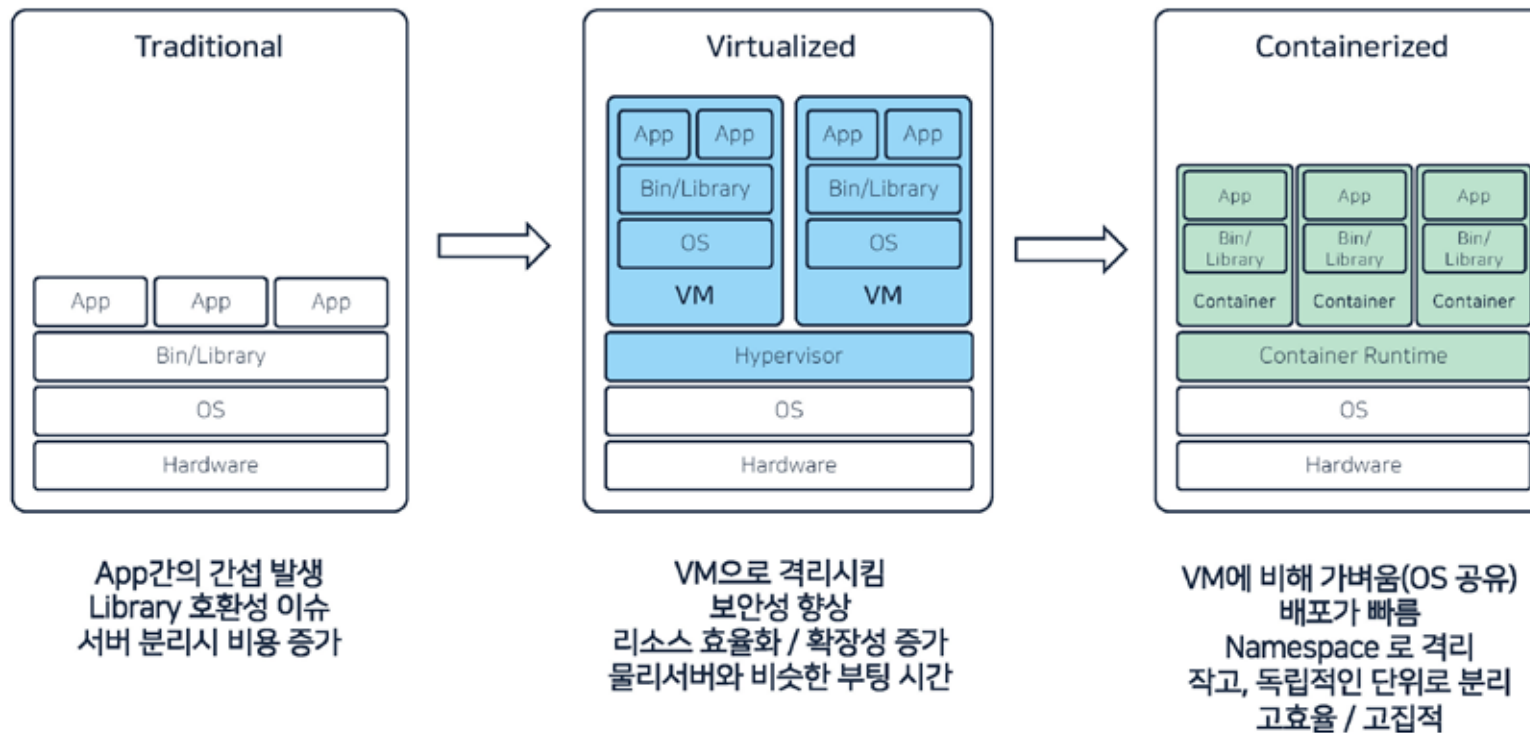
# Elastic computing with containers

---

## Why containers

- Consistent Runtime Environment
- Isolation
- Small Size on Deployment Image
- Low Overhead

# Evolution of computing (deployment)



# Containers at Google

From Gmail to YouTube to Search,  
“Everything at Google runs in containers.”

Google starts over two billion containers per week.



<https://cloud.google.com/containers/>

# Containers at Paypal



## Later Stage Benefits





# Application modernization



## Developer Issues:

- Minor code changes require full re-compile and re-test
- Application becomes single point of failure
- Application is difficult to scale

**Microservices:** Break application into separate operations

**12-Factor Apps:** Make the app independently scalable, stateless, highly available by design

# Kubernetes (K8s)

- [Container orchestration platform](#) for hosting containers in a clustered environment with multiple hosts
- Provides container grouping, load balancing, auto-healing, scaling features

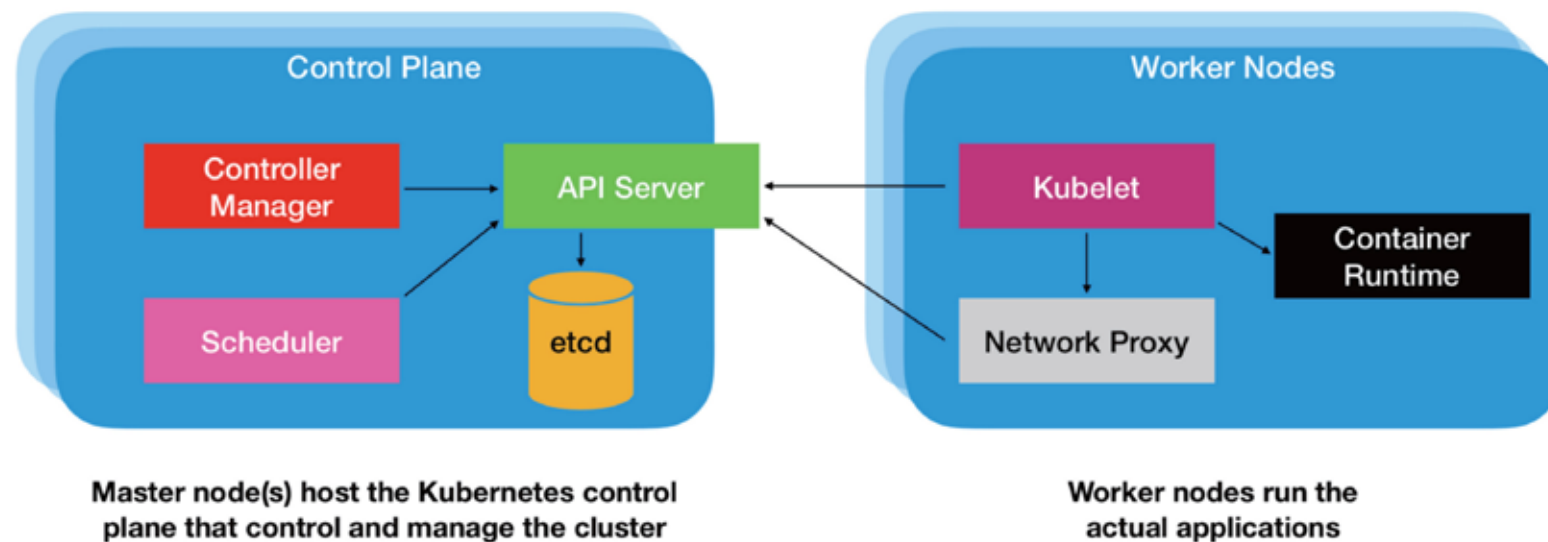


**kubernetes**

## K8s features

- Automated rollouts and roll backs
- Service health monitoring
- Automatic scaling of services
- Declarative management
- Deploy anywhere, including hybrid deployments

# K8s architecture



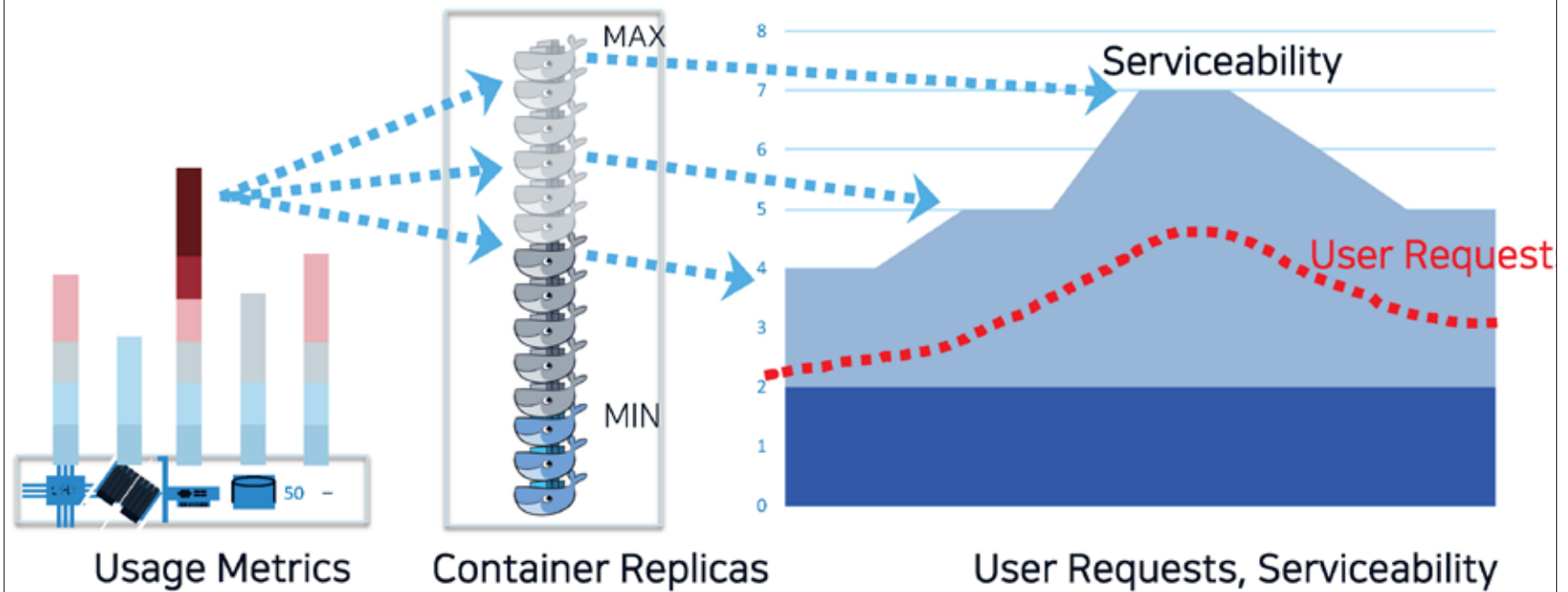
<https://www.slideshare.net/mikaelbarbero/kubernetes-101-a-cluster-operating-system>

# NAVER Container Cluster



- 대규모 서비스 환경과 multi-tenancy를 지원하도록 튜닝 및 플러그인 개발
- 권한관리, 법령 준수에 따른 접근 제어 등 보안 기능 강화
- 기존 네이버 인프라와 연동 및 운영/배포 프로세스 지원

## 부하 변화에 따라 자동 확장과 축소 (auto scaling)



# Things to consider while adopting containers

- Interoperability with the existing infrastructure
  - Authentication & authorization
  - ACL (access control list)
  - Networking (Routing, DNS, IP management)
  - IT management tools
- Logging for trouble shooting
- Monitoring (dashboard)
- Degree of automation
  - Self healing
  - Auto scaling
- Security

# Security

- Container image registry
  - Private image registry
  - Base image must be maintained securely
- Security update & patch
  - Much easier compared to physical and virtual servers
- However, security breach happens it is more complicated to analyze.
  - Logging is important



# Storages for containers

---

# Stateless vs. Stateful Container

## Stateless

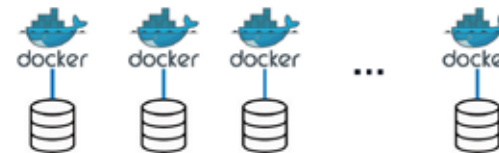
- Nothing to write on disk
- Web (Front-end)
- Easy to scale in/out
- Container is ephemeral
- If delete, will be lost data



*Easy to scale out*

## Stateful

- Needs storage to write
- Database
- Logs
- CI config / repo data
- Secret Keys



*Hard to scale out*

# Ephemeral vs. Persistent Storage

## Ephemeral Storage

- Data Lost
- Local Storage
- Stateless Container

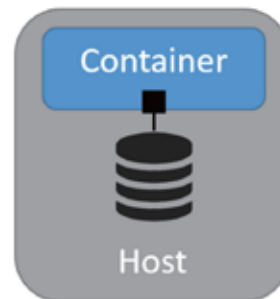
VS

## Persistent Storage

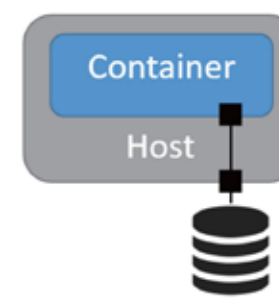
- Data Save
- Network Storage
- Stateful Container



Data in the container  
Lost when the container terminates



Data in a Host Volume  
Lost when the host terminates

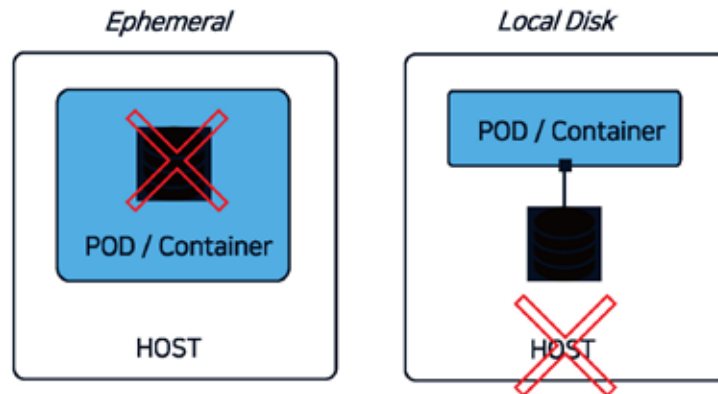


Networked Volume / File System  
Independent of host and container

Reference : <https://www.infoworld.com/article/3106416/cloud-computing/containerizing-stateful-applications.html>

# Storage in K8s

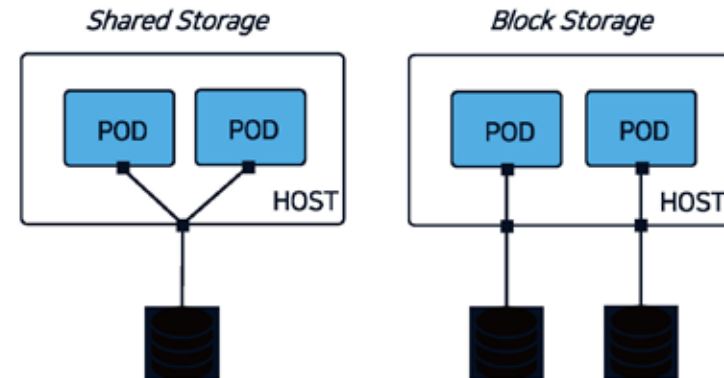
## Local Storage



데이터 저장 Pod(컨테이너) 내부  
 Pod 삭제 시 데이터도 함께 삭제  
 Host 장애 시 데이터 사용 불가

호스트 로컬 디스크  
 삭제되지 않음  
 데이터 사용 불가

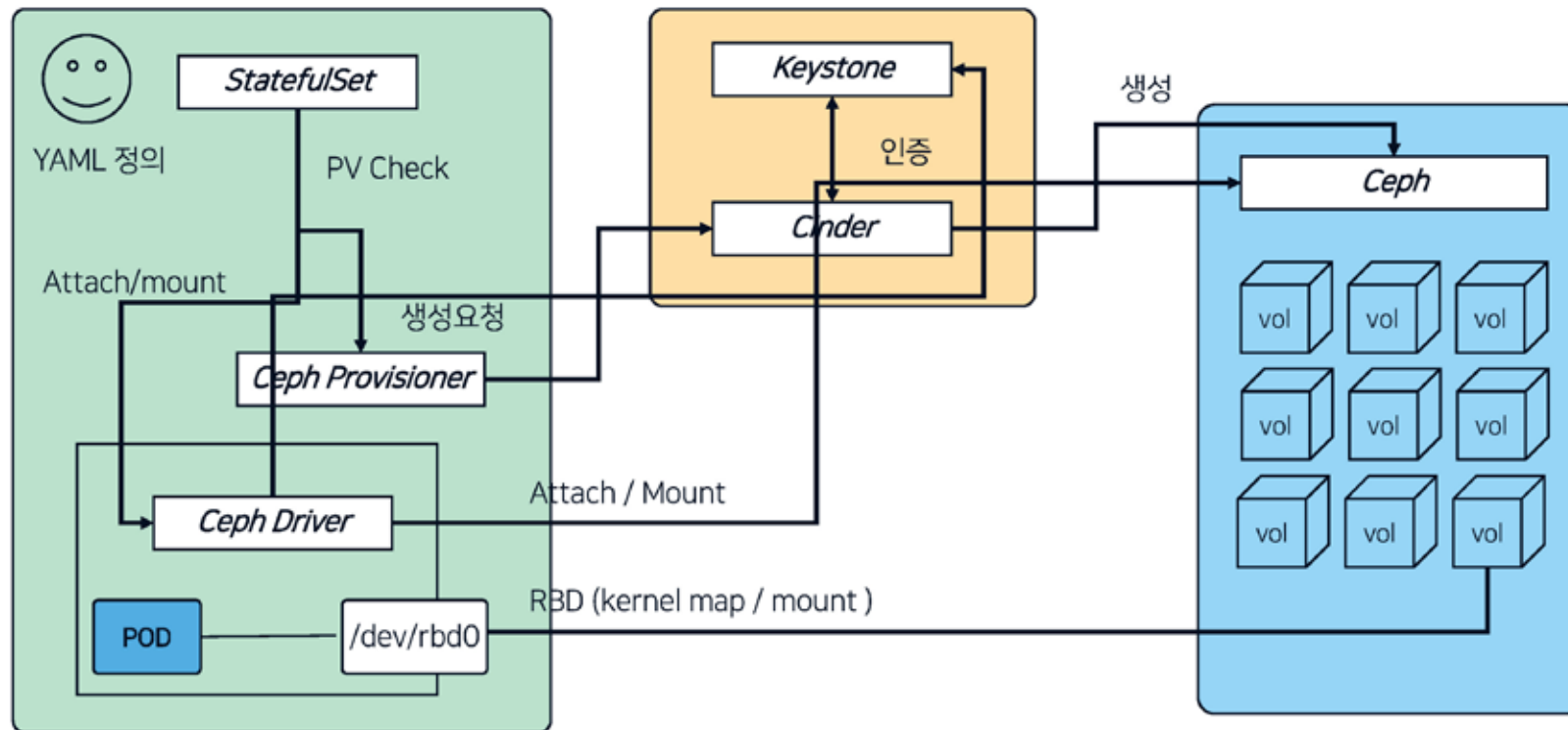
## Remote Storage



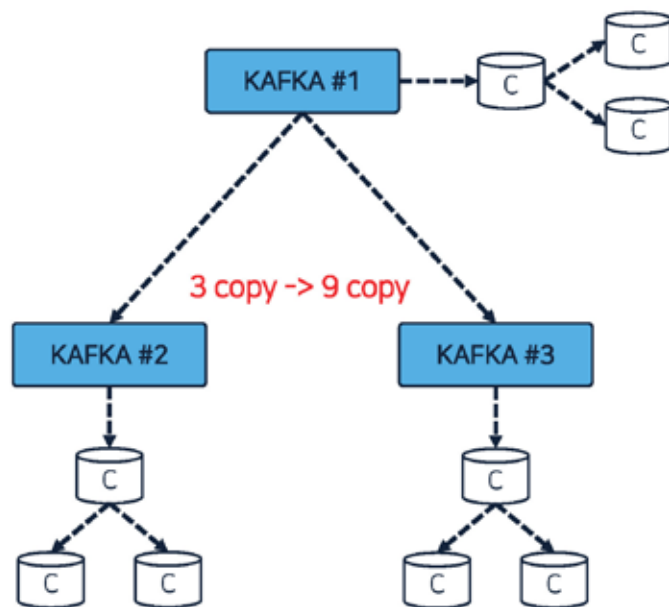
외부 네트워크 스토리지  
 (여러 Pod 가 스토리지 공유)  
 삭제되지 않음  
 서비스 영향 없음

외부 네트워크 스토리지  
 (Pod 별 스토리지 할당)  
 삭제되지 않음  
 서비스 영향 없음

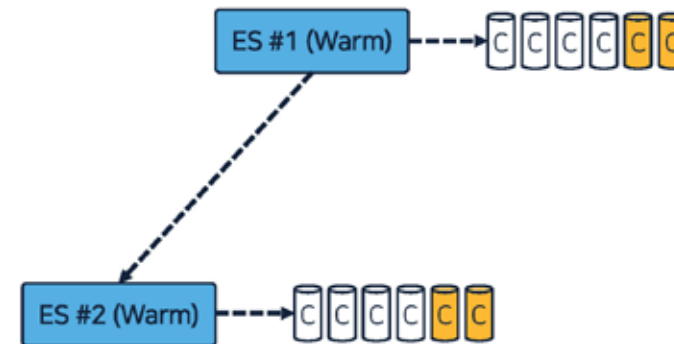
## K8s Volume Plugin in NAVER



## Too many replication problem



Kafka on ceph RBD (3 copy)

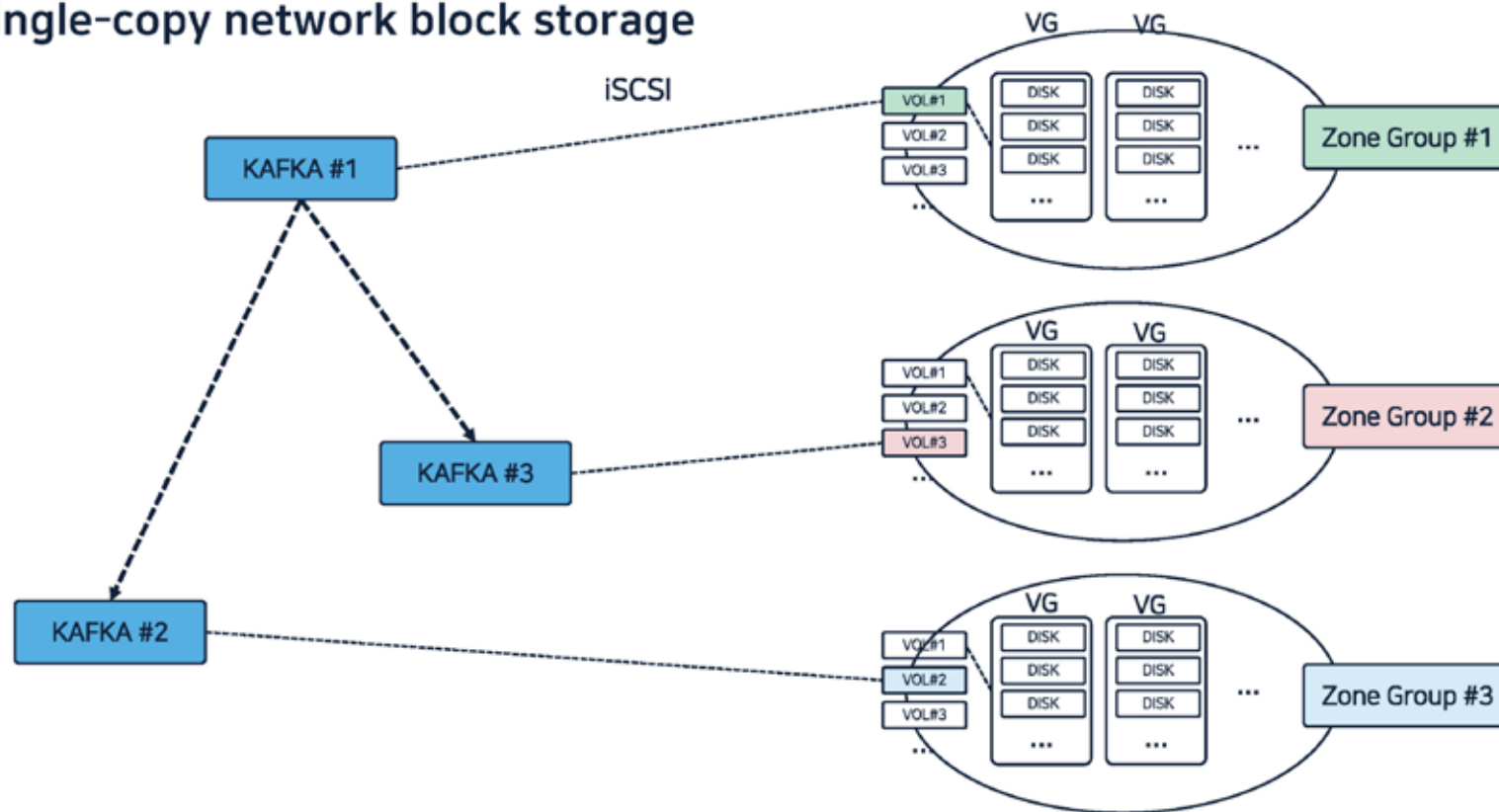


ES : 2 copy  
EC : 1.5 copy  
= 3 copy

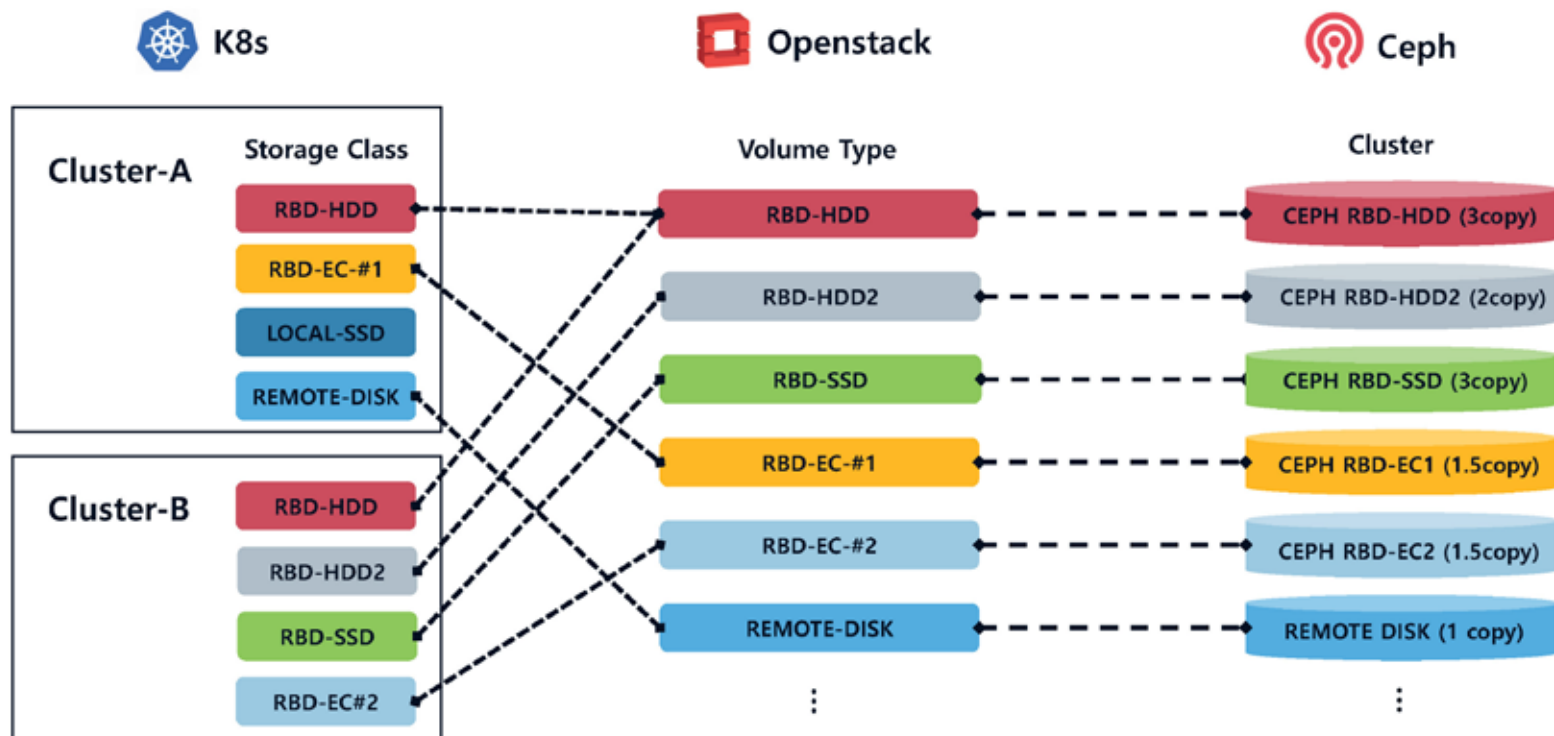
Elastic Search (Warm) on ceph EC (1.5 copy)

# Remote Disk

Single-copy network block storage



# Multi-type Persistent Storage Class

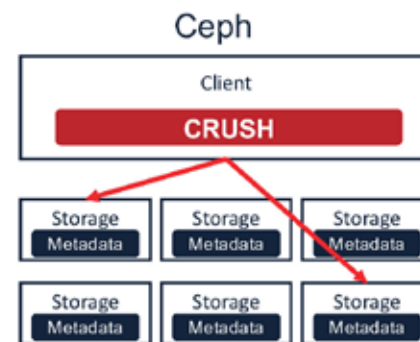
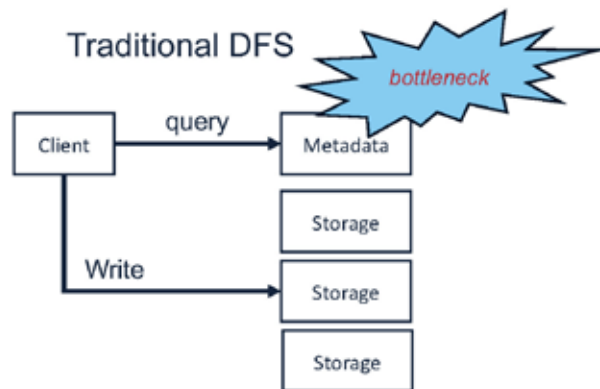




# Redistribution overhead is VERY BIG!

## Ceph Metadata Management

- No metadata bottleneck (no lookup)
- CRUSH algorithm determines data placement



But, there is a big storage redistribution overhead due to capacity expansion & storage server failure/replacement.

# System observability

---

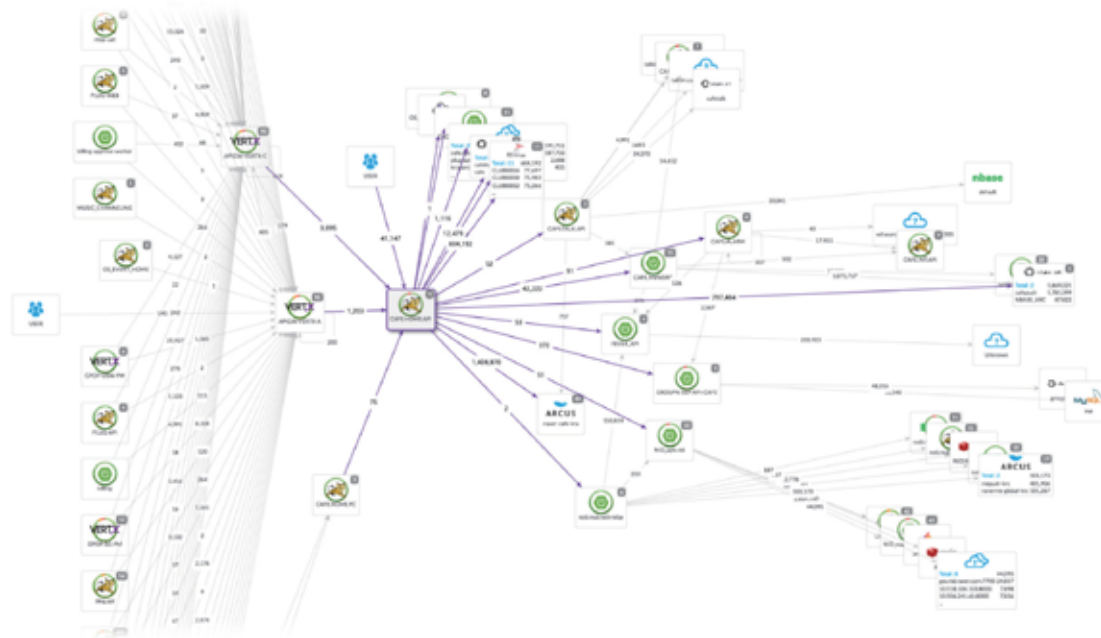
장애나 성능 문제 상황을 빠르게 인지하고 언제, 어디서, 무엇 때문에 발생했는지

직관적으로 이해하고 대응하는데 도움을 주는 플랫폼이 필요

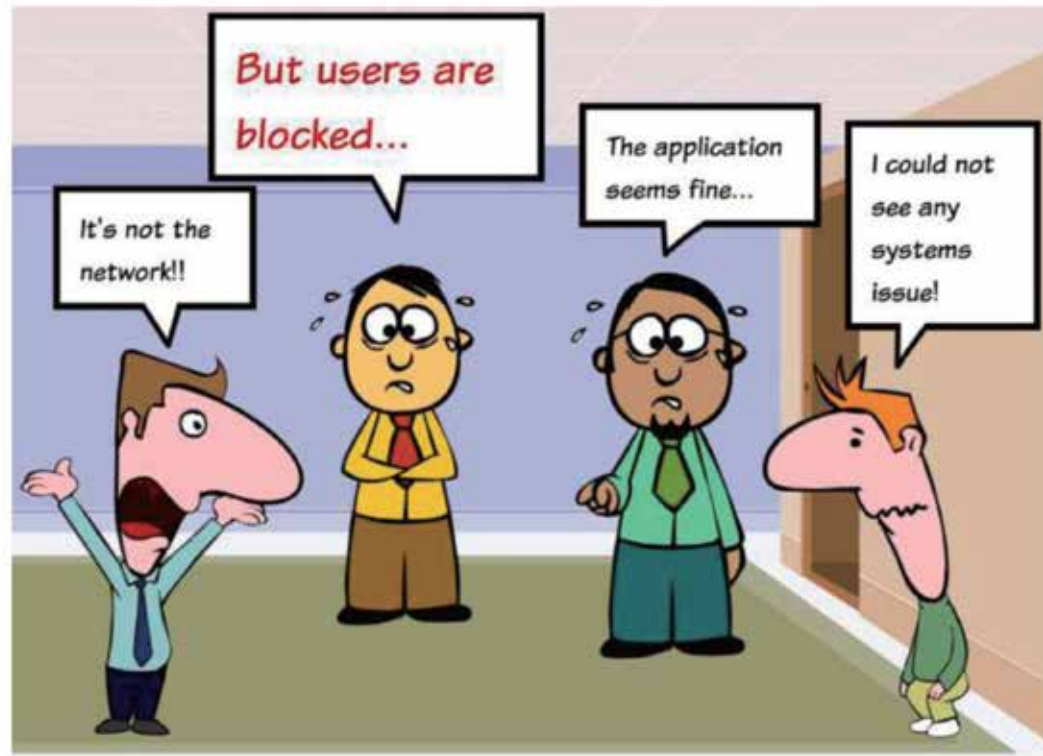
# What is an APM?

**Application Performance Management (APM)** is the monitoring and management of performance and availability of software applications.

- Wikipedia



## Why need an APM?



<https://www.slideshare.net/SecurActive/workshop-apm-in-a-cloud-virtualized-environment>

# APMs around you

## Commercial



## Open Source



# Pinpoint

PINPOINT



- 다양한 기능/ 대규모 분산 서비스 및 시스템의 성능을 분석/진단/추적에 최적화된 기능 보유
- NAVER 에서 만든, NAVER 에서 사용 중인 서비스 (Since 2015) 12,000+ agents
- GitHub 좋아요(★) +10k 오픈소스

# Pinpoint 기능 및 특징

## 코드 수준의 가시성

문제 발생 지점 및 병목 구간을 쉽게 발견할 수 있도록,  
코드 수준의 가시성을 제공

## 수평 확장성

대규모 서버군을 지원할 수 있도록  
수평 확장성을 제공

## 토폴로지 자동 발견

애플리케이션 구성을 쉽게 파악할 수 있도록  
애플리케이션 토폴로지를 자동으로 발견하고,  
Server Map을 제공

## 분산 트랜잭션 추적

분산된 애플리케이션의 메시지를  
추적할 수 있는 기능을 제공



## Bytecode instrumentation

Bytecode instrumentation 기법으로,  
코드 수정 없이 성능 정보를 수집

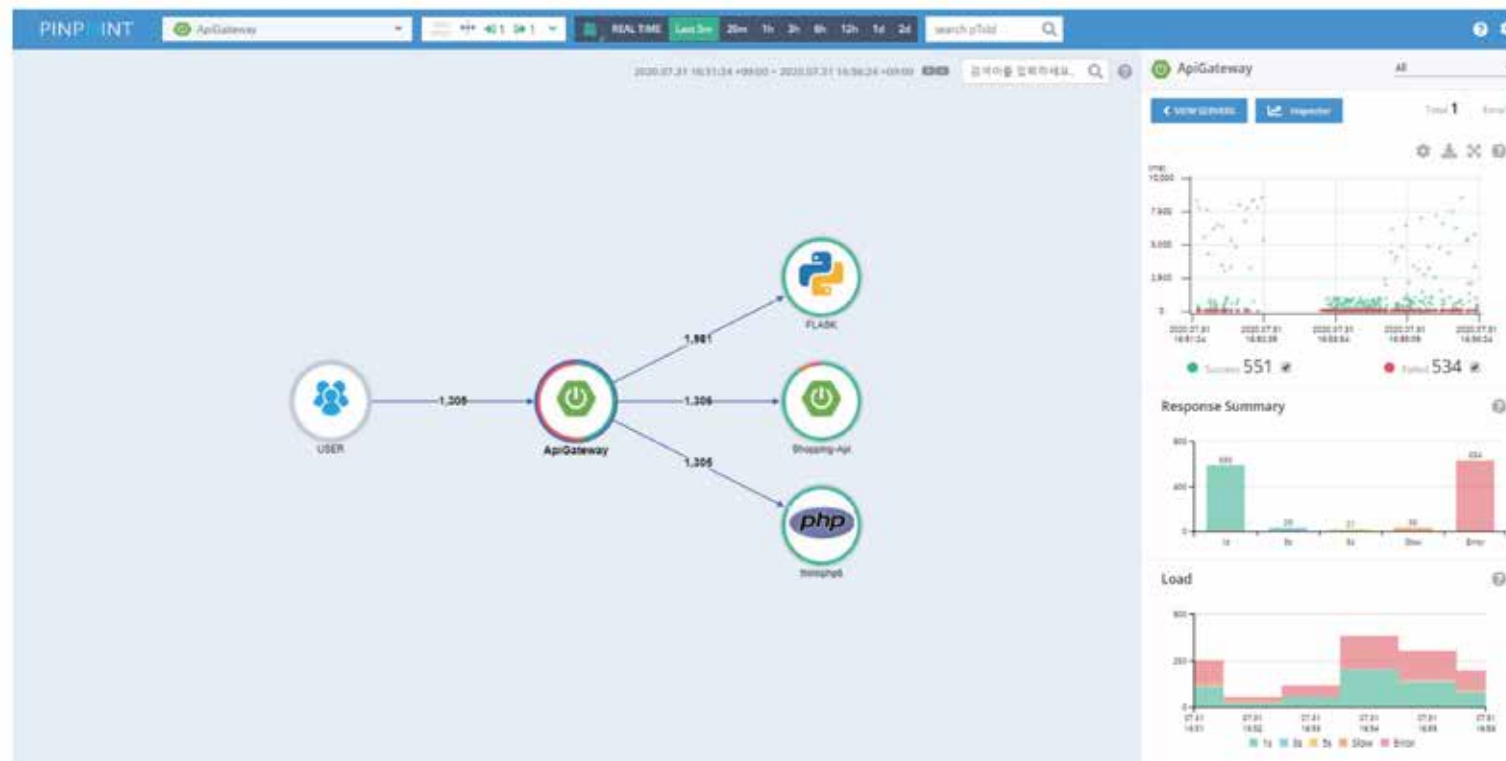
## 알람 기능

모니터링 대상 및 임계치 설정으로 실시간 모니터링과 SMS/Email 알람도 가능

## Plug-in 기능

현재까지 40여종의  
Plug-in 기능을 제공

# Pinpoint 화면: System map





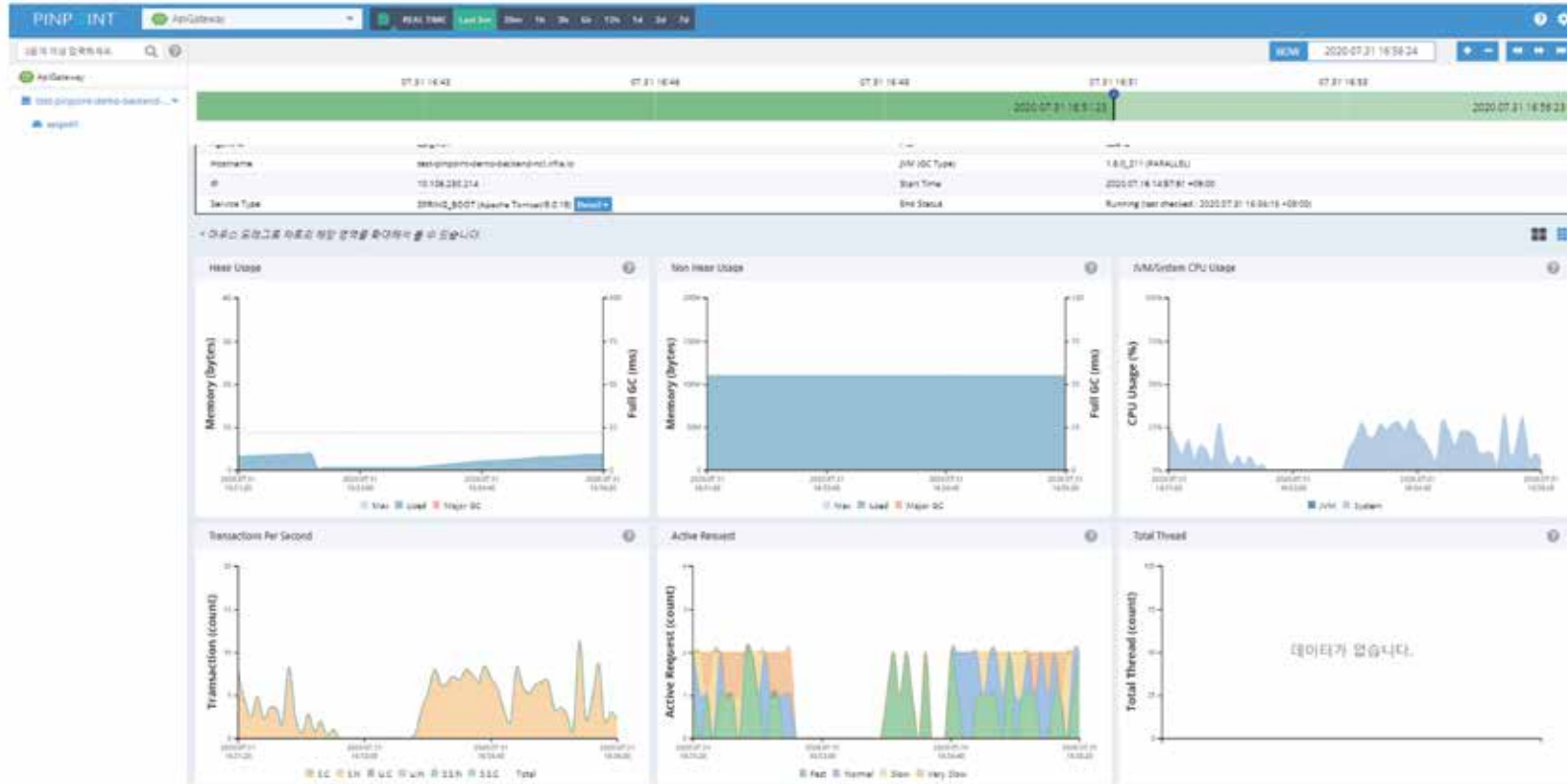
# Pinpoint 화면: Call Stack Tracing

PINPOINT									
10:57 10:58 10:59 10:40 10:41 10:42 More (304/1045)									
#	StartTime	Path	EndPoint	Req...	Res...	Agent	Client IP	Transaction	
31	2020.08.08 10:41:28.799	shopping.orders	zy.backend.com:81...	76	Pass		127.0.0.1	argu-0115948790712091623594	
172	2020.08.08 10:41:28.794	shopping.orders	zy.backend.com:81...	76	Pass		127.0.0.1	argu-01159487907120916235912	
8	2020.08.08 10:41:28.843	shopping.orders	zy.backend.com:81...	76	Pass		127.0.0.1	argu-01159487907120916235944	
16	2020.08.08 10:41:28.887	shopping.orders	zy.backend.com:81...	76	Pass		127.0.0.1	argu-01159487907120916235988	
20	2020.08.08 10:41:28.879	shopping.orders	zy.backend.com:81...	76	Pass		127.0.0.1	argu-0115948790712091623541	
46	2020.08.08 10:41:28.200	shopping.orders	zy.backend.com:81...	76	Pass		127.0.0.1	argu-0115948790712091623530	
79	2020.08.08 10:41:29.043	shopping.orders	zy.backend.com:81...	76	Pass		127.0.0.1	argu-0115948790712091623598	
184	2020.08.08 10:41:06.470	shopping.orders	zy.backend.com:81...	76	Pass		127.0.0.1	argu-01159487907120916235496	
108	2020.08.08 10:41:29.448	shopping.orders	zy.backend.com:81...	76	Pass		127.0.0.1	argu-0115948790712091623574	
78	2020.08.08 10:41:29.821	shopping.orders	zy.backend.com:81...	76	Pass		127.0.0.1	argu-0115948790712091623549	
130	2020.08.08 10:41:18.457	shopping.orders	zy.backend.com:81...	76	Pass		127.0.0.1	argu-0115948790712091623545	
37	2020.08.08 10:41:28.854	shopping.orders	zy.backend.com:81...	76	Pass		127.0.0.1	argu-0115948790712091623535	

ApplicationName: PLASK Agent: Pass TransactionId: argu-0115948790712091623594 Path: /shopping/orders									
Call Tree Server Map Timeline Thread View Console									
Self +									
Method	Argument	StartTime	Response	Response	ExecTls	SelfTime	Class	API	Agent
Root	shopping.orders	10:41:28.799	0	76		0		HTTP/1.1	Pass
RootConnection									
1. PROXY_PUSH_DATA									
2. REMOTE_ADDRESS	127.0.0.1								
3. ENDPOINT	zy.backend.com:8154								
4. URL_ADD_PATH	(url) http://zy.backend.com:8155/shopping/orders/...	10:41:28.799	0	72		72		HTTP/1.1	Pass
5. RETURN_DATA	<response> (200)								
6. HTTP_STATUS_CODE	200								
7. END_REQUEST	shopping.orders	10:41:28.848	90	22		22		HTTP/1.1	Pass
8. HTTP_STATUS_CODE	200								
9. PROXY_PUSH_DATA									
10. REMOTE_ADDRESS	127.0.0.1								
11. ENDPOINT	zy.backend.com:8155								
12. CONNECTION_TIMEOUT, READ_TIMEOUT, WRITE_TIMEOUT, IN...	zy.backend.com:8154	10:41:28.799	-41	0		0	RootConnection	OK_HTTP_CLIENT	Argu-01
13. RETURN		10:41:28.879	1	0	0	0	RootCall	OK_HTTP_CLIENT	Argu-01
14. INTERCEPTOR_INTERCEPTOR	http://zy.backend.com:8155/shopping/orders	10:41:28.879	0	0	0	0	HttpInterceptor	OK_HTTP_CLIENT	Argu-01
15. HTTP_STATUS_CODE	200								
16. CONNECTION_TIMEOUT, READ_TIMEOUT, WRITE_TIMEOUT, IN...	zy.backend.com:8155	10:41:28.879	0	0		0	RootConnection	OK_HTTP_CLIENT	Argu-01

## Pinpoint 화면: 상세지표 조회



# Pinpoint Cloud

## Pinpoint Cloud New



Pinpoint, the performance analysis/diagnosis/tracing platform for large-scale distributed services and systems, is conveniently available through a cloud environment.

[Learn More >](#)



**NAVER  
CLOUD  
PLATFORM**

## Summary

### Toward elastic and robust service

- ✓ Container를 활용한 autoscaling
- ✓ Stateful container를 위한 다양한 저장소
- ✓ Pinpoint를 이용한 System observability

임계점을 넘어서는 트래픽 제어와  
시스템의 여유 처리량 예측 등의 문제는  
서비스마다 특화된 다양한 방식을 시도하고 효과를 검증하고 있는 중

Questions?