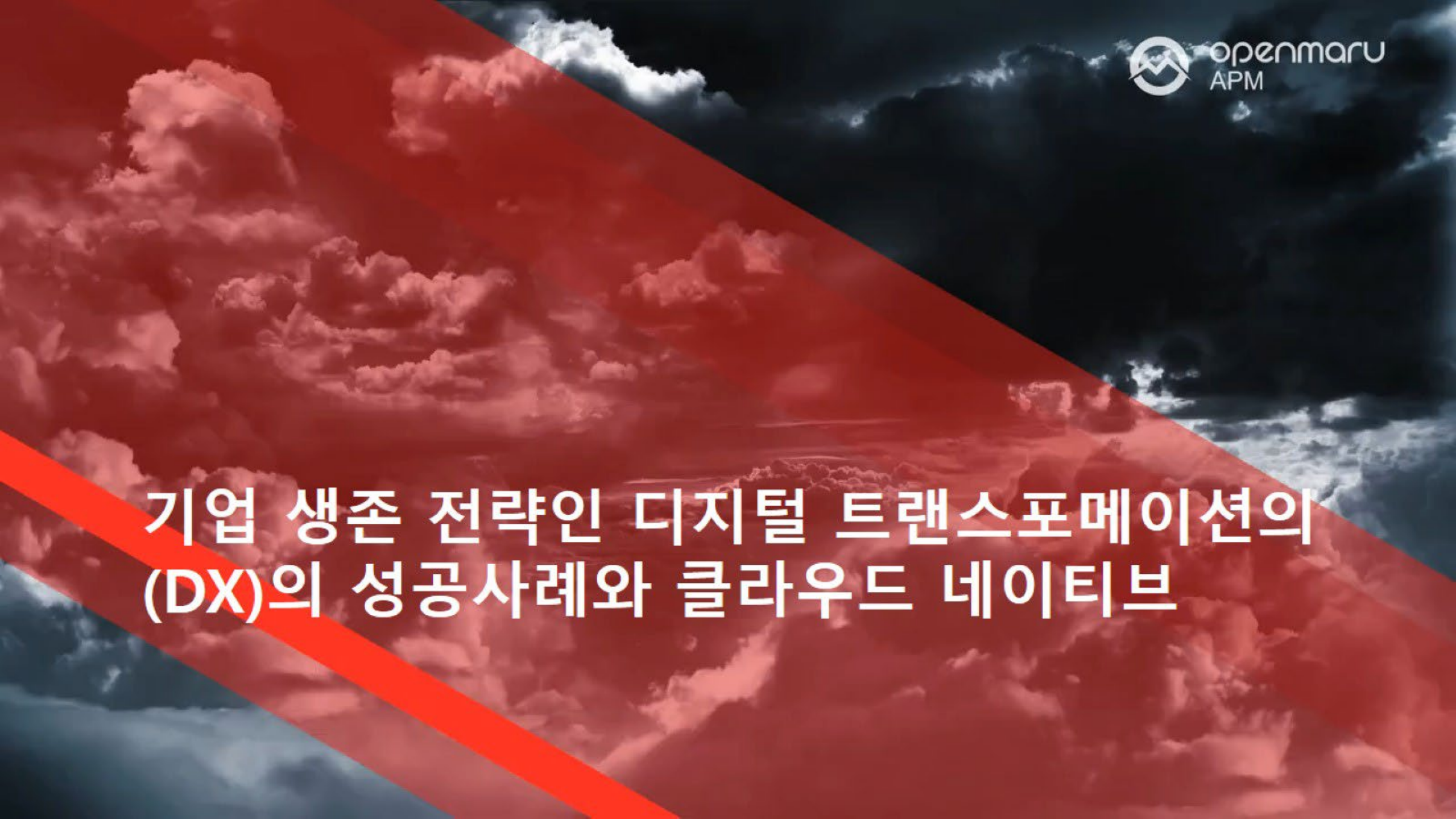

openmaru
APM
기업 생존 전략인 디지털 트랜스포메이션의
(DX)의 성공사례와 클라우드 네이티브

Use of Cloud-Native Technologies Will Be Pervasive, not Just Popular

- **Gartner는 2025년까지 새로운 디지털 워크로드의 95% 이상이 클라우드 네이티브 플랫폼에 배포될 것으로 예상**
- **Gartner 분석가들은 2025년까지 85% 이상의 조직이 클라우드 우선 원칙을 채택할 것**

https://www.gartner.com/en/newsroom/press-releases/2021-11-10-gartner-says-cloud-will-be-the-centerpiece-of-new-digital-experiences

98%

of enterprises surveyed use or plan to use multicloud(1)

>95%

of new digital workloads will be deployed on cloud native platforms by 2025(2)

>90%

of apps will be cloud native by 2025(3)

72%

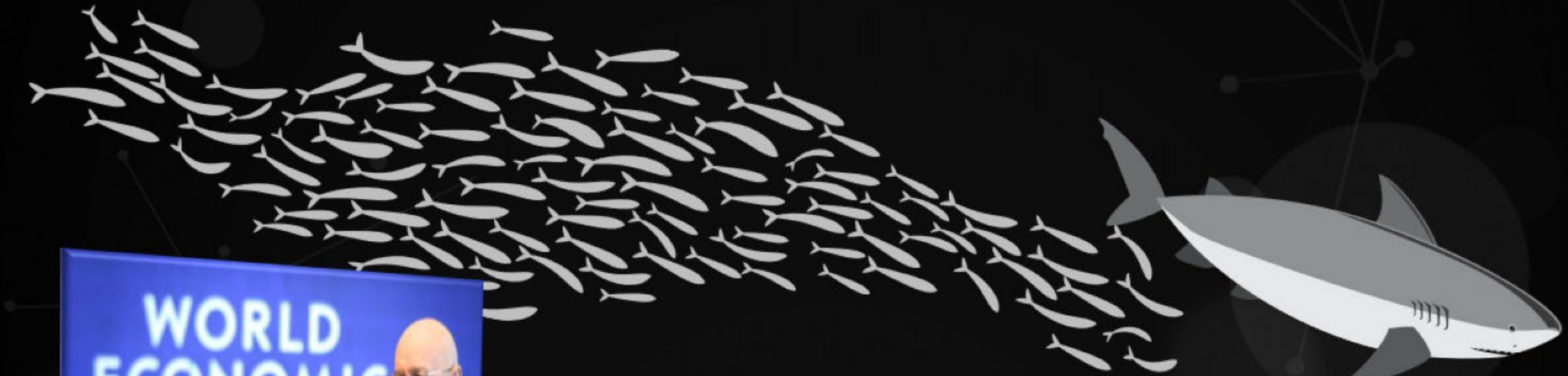
of respondents expect that the majority of their apps will be created using cloud native development by 2023(4)

Source - https://blogs.oracle.com/cloud-infrastructure/post/announcing-graal-cloud-native-gcn

OPENMARU © 2021 | All Rights Reserved.

The fast fish eats the slow fish

"In the new world, it is not the **big fish** which eats the **small fish**,
it's the **fast fish** which eats the **slow fish**."



4 차 산업혁명의 이해 (Mastering the Fourth Industrial Revolution) – 세계 경제 포럼
Klaus Schwab, Founder and Executive Chairman of the World Economic Forum

OPENMARU © 2021 | All Rights Reserved.

openmaru APM

Business Disruptor

airbnb

- **자신의 방이나 집, 별장** 등 사람이 지낼 수 있는 모든 공간을 임대할 수 있으며, 192개국 3만 4800여 개 장소에서 200만여 개의 객실에 대한 숙박을 중개
- Airbnb는 **부동산**을 소유하지 않음

facebook

- 전 세계 14억 9천 만명 이상의 월 활동 사용자가 활동 중인 세계 최대의 **소셜 네트워크 서비스**
- **자체 생산 콘텐츠 없이도** 뉴스, 사진, 비디오 제공

UBER

- 승객이 스마트폰 앱을 이용해 차량을 호출하면 우버와 계약한 기사가 자기 차량을 몰고 와 목적지까지 데려다 주는 '**주문형 개인 기사 서비스**'
- **Uber는 보유한 차량이 없음**

Alibaba Group

- 중국의 전자상거래업체로 '세계에서 가장 큰 쇼핑몰' 서비스
- **Alibaba는** 보유한 **상품재고 없이** 전자상거래

openmaru

"메르세데스 벤츠는 이제 자동차 기업이 아닌 소프트웨어 기업이며 자동차는 궁극의 웨어러블이다."
- 올라 켈레니우스, 메르세데스 벤츠 그룹 대표

"우리는 클라우드 네이티브로 기술을 재구축하면서 조직 운영 방식도 바꿨다."
- 넷플릭스 클라우드 부문 부사장 이즈라일레브스키

"골드만삭스는 정보기술(IT) 회사다."
- 골드만삭스 로이드 블랭크파인 CEO

"인간 중심의 디지털 전략이 미래 리테일 산업의 성공의 열쇠가 될 것입니다."
- Kevin Johnson, CEO of Starbucks

- Confidential -

OPENMARU © 2021 | All Rights Reserved.

Netflix 클라우드 네이티브 성공 사례

Netflix는 클라우드 네이티브 성공사례로,
전세계 확산 서비스를 제공하여 사용자에게 고화질 동영상 서비스를 안정적으로 제공

넷플릭스는 매년 전세계 인터넷 트래픽의
15% 이상을 차지하고 있음

2021년에는 이미 2억 400만명
(3년 만에 2배 증가)의 구독자가 생겼고,
매 4분기 마다 5백만명이
200개 이상의 국가에서 구독 中

NETFLIX
OSS
Netflix
Open SourceSoftware
Center

spring boot
+
NETFLIX

기술팀이 8년간 노력해온 결과

인프라를 자체 센터에서
Public 클라우드 이전

모놀리틱 프로그램을 작게 관
리할 수 있는 마이크로서비
스 아키텍처로 변경

애플리케이션 함수 실행 서비
스하는 서버리스 컴퓨팅 과 및
백엔드 아키텍처를 제공

- Confidential -
OPENMARU © 2021 | All Rights Reserved.

클라우드 네이티브 국내·외 선도 도입 사례

선진 사례

AWS

빠른 배포 구현

수 천개 팀(자율적 DevOps팀) X 마이크로서비스 아키텍처
X 지속적 배포(CD) X 다양한 개발 환경

넷플릭스

가입자 대상서비스 확대

Netflix Open Source Software Center

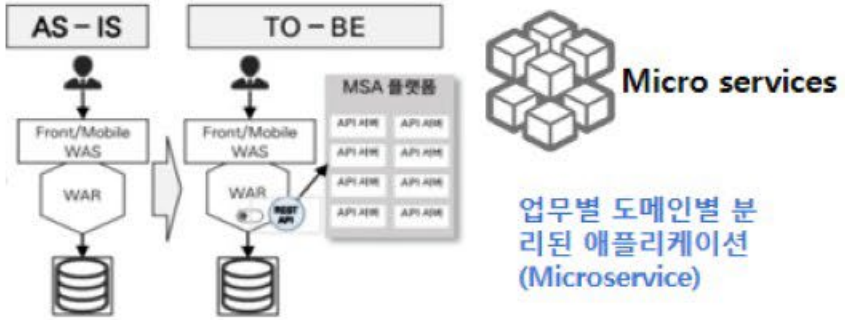
KAKAO

계열사 신규서비스 확대 및 빠른 출시 사례

카카오의 애자일 문화, 일하는 방식 관리를 위한
전담팀 및 개발플랫폼 운영

11번가

서비스 분리를 통한 점진적 MSA전환



업무별 도메인별 분리된 애플리케이션 (Microservice)

Confidential | All Rights Reserved.

기존 IT로부터 새로운 IT로의 과도기

openmaru

기존 IT

- 물리 서버와 PC
- 5년 이상 시스템 사용 기간
- 폭포수형 개발
- 운용의 우선 사항 “안정성”
- 시스템 부문은 시스템에 전념
- 대형 IT 벤더가 기술을 주도

새로운 IT

- 클라우드와 모바일
- 몇 개월 사용 후 폐기
- Agile, DevOps
- 운용의 우선 사항 “민첩성”
- 시스템 부문도 비즈니스에 초점
- 사용자 주도 기술,오픈 소스 주도

- Confidential -

OPENMARU © 2021 | All Rights Reserved.

SoR 에서 SoE 로 전환

openmaru

	Systems of Record (SoR)			Systems of Engagement(SoE)		
	Mainframe	Mini	PC	Internet	Mobile/Cloud	Connected
Time Frame	1960 - 1975	1975-1992	1992-2001	2001-2009	2010-2015	2016-
Data Types	Batch	Dept Process	Documents	Web Pages	User Interactions	IOT / AI
First Movers	IBM, Unisys	DEC, Compaq	Microsoft, Dell, IBM	Google, Microsoft	Facebook, Amazon, Apple	Airbnb, Uber

	역할	
데이터를 기록하는 시스템		연결·관계·상호작용을 만드는 시스템
신뢰성/안전성/정확성/데이터 중심	포커스	즉시성/유연성/사용자 중심
Waterfall	어프로치	DevOps & Agile
Java & . NET	프레임웍	경량/다양한 프로그램 언어
독점 기술 중심/ 물리&가상 환경	기술	오픈 / 클라우드
모노리식	아키텍쳐	마이크로 서비스
웹기반	어플리케이션	Mobile
일괄/실시간 처리	트랜잭션	리얼타임, 탄력적인 인프라

- Confidential -

OPENMARU © 2021 | All Rights Reserved.

openmaru

Development Process

WATERFALL

AGILE

DEVOPS

Application Architecture

MONOLITHIC

N-TIER

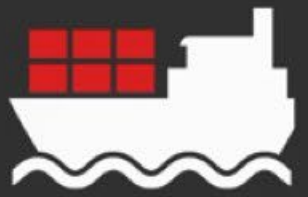
MICROSERVICES

Deployment & Packaging

PHYSICAL SERVERS

VIRTUAL SERVERS

CONTAINERS



Application Infrastructure

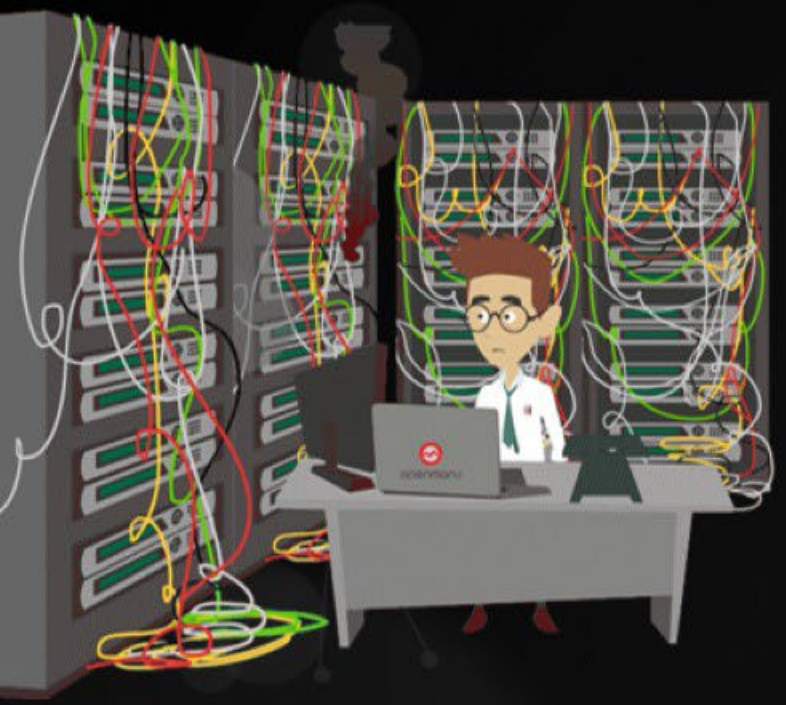
DATA CENTER

HOSTED

CLOUD

Cloud Native
클라우드가 문제가 아니고,
현재에 이슈부터 해결해야 해요.
MAERSK
MAERSK SEALAND
HAMBURG SÜD
Safmarine
K LINE

시스템 비대화로 작업 폭증과 인력 부족 어떻게 할까요?



장애의 65 %는 Human Error이며,시스템 복잡도와 난이도 증가

시스템 운용 업무의 45 %는 정기적으로 수행해야하는 반복 작업

운영 효율화를 통한 비용 절감의 요구

시스템의 대규모 화

높은 수준의 엔지니어 부족

지속적인 시스템 통합 요구

동일한 작업 반복

운영 품질 향상

운영 비용 (TCO) 절감 요구

업무 확대와 관련 데이터양의 비약적인 증가

가상화, 클라우드 등 다양한 운영 환경의 증가와 관리 효율화 요구

운영 품질에 대한 지속적인 향상 요청

OPENMARU © 2021 | All Rights Reserved.

개발팀과 운영팀 누구의 잘못인가? 불행의 시작

- 하드웨어 도입부터, OS, 미들웨어, 빌드/배포, 기타 인프라 환경 등 복잡한 과정
- 관리 서버 증가

기존개발 운영 환경

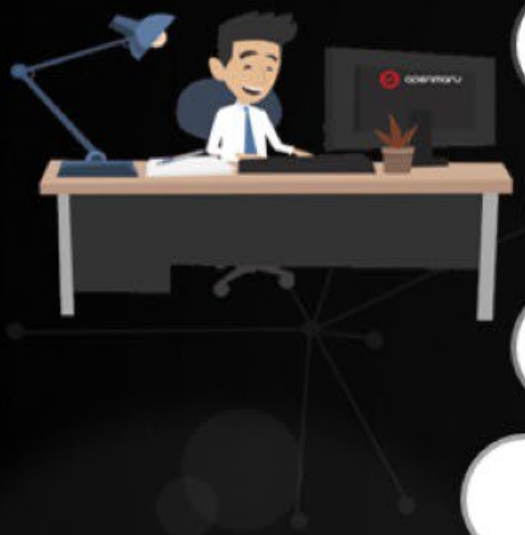
계획 → 예산수립 → H/W 선정 → 입고 및 대기 → H/W 반입 → H/W 설치 → OS 도입 → OS 패치 / 튜닝 → 사용자 생성 → M/W 서버 도입 → 테스트 도구 선정 → 운영 배포 → 서버 구성 및 설정 → 운영서버 이전 → 운영서버 가동 → 서버 추가계획

코드 개발 → 빌드 → 정적 분석 → 시큐어 코딩 → 단위 테스트 → 테스트 배포 → 통합 테스트

- 개발과 운영팀은 분리
- 여러 단계의 개발과 운영 프로세스를 수작업으로 수행함

OPENMARU © 2021 | All Rights Reserved.

개발자의 고민들



- 배포는 왜 야간에만 하나요? 서비스의 중단 없이 언제든지 업무시간 중에 애플리케이션을 배포하고 싶습니다.
- 개발 환경 조차 바로 지원되지 않아요. 시스템 개발 환경 구축에 너무 많은 시간이 걸려요.
- 하드웨어 도입, 네트워크 설정, LB 설정요청, 방화벽 해제 요청 절차가 너무 많아 시간이 너무 오래 걸립니다.
- WAS 운영 배포담당자의 도움 없이도 필요하면 개발팀에서 즉시 배포하고 싶습니다.
- CI/CD 를 구축 해야 하는데 엄두 조차 나질 않아요.
- 바빠서 DevOps 는 아예 생각 할 수도 없어요.
- 다양한 버전의 개발 환경이 필요한데, 지원 받을 수가 없어요.

OPENMARU © 2021 | All Rights Reserved.

운영자의 고민들

- 더 이상 시스템 중단에 대한 공지 없이 365일 , 24 시간 운영 가능한 인프라를 구축하고 싶습니다.
- 솔루션 없이도 모든 시스템에 이중화를 통한 가용성을 확보 하고 싶습니다.
- 시스템 장애 시 운영자의 개입 없이 시스템이 자동으로 복구되었으면 좋겠습니다.
- 시스템이 아니라 서비스의 상태를 확인하여 운영 하였으면 좋겠습니다.
- OS 업그레드, 패치, 보안 패키지 관리 등에 너무 많은 시간과 노력이 필요해요.
- 개발팀에서 오픈소스 S/W 에 대한 요구가 많은데 지원해 줄 수가 없어요.
- 대규모 이벤트로 인한 부하에 대응할 수 있는 시스템 요구

OPENMARU © 2021 | All Rights Reserved.

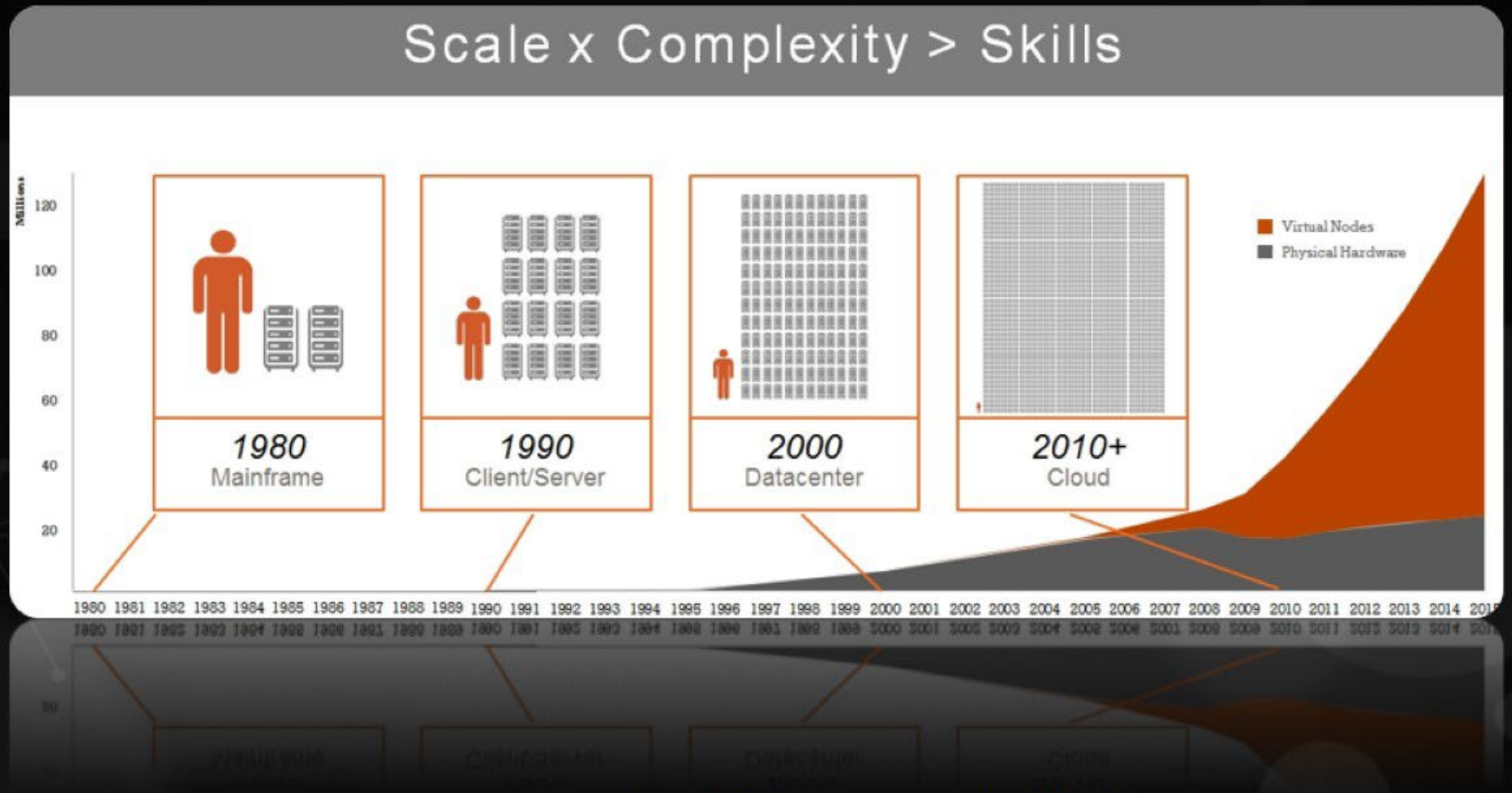
기획자의 고민들

- 서버가 추가되면 OS 와 보안 패키지 때문에 비용 부담이 너무 큽니다.
- VM 환경에서 Guest OS 라이선스나 관리 비용 때문에 고민이신가요?
- 클라우드 네이티브 세상에서 가상화나 HCI 환경을 계속 유지하고 확대 하는 것이 맞는 것일까?
- 단순히 클라우드로 전환하면 안된다고 하는 데, 클라우드 네이티브는 무엇인 가요?
- 시스템 S/W 들을 계속 구매해야 할까요? 공개S/W 를 적극 활용하라고 하는데.. 어떻게 할까요?
- 서비스와 서버는 확대 되고 있지만 IT 개발인력과 운영 인력은 항상 부족해요.
- CentOS 때문에 머리 아파 죽겠어요.

OPENMARU © 2021 | All Rights Reserved.

클라우드 네이티브 시대
다 좋기만 할까?

Increasing scale and complexity means we need admin automation

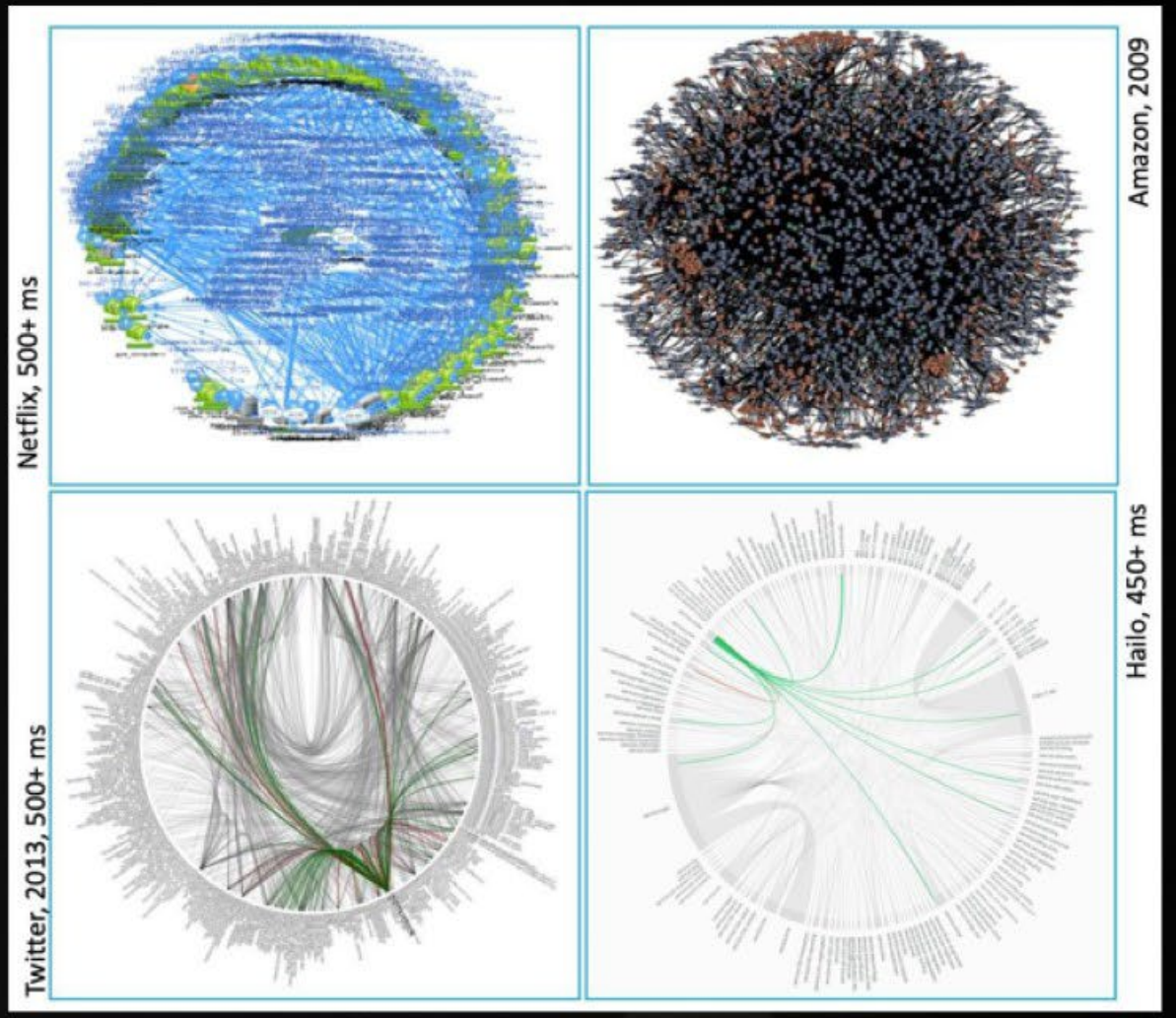


Opscode gets more venture dough for its Chef
From - http://goo.gl/dLcjS

OPENMARU © 2021 | All Rights Reserved.

Observability 배경 -클라우드 네이티브 환경의 애플리케이션 모니터링 어려움

- 애플리케이션당 워크로드 수가 기하급수적으로 증가함에 따라 애플리케이션 간 및 애플리케이션 내 API 트래픽 흐름도 폭발적으로 증가
- 마이크로 서비스 아키텍처일수록
  모니터링 방법이 상당히 복잡
- 마이크로 서비스 아키텍처에
  모니터링과 트러블 슈팅 방법이 고려



Netflix, Amazon, Twitter, Hailo 에서
500개가 넘는 마이크로서비스 간의 연결

「Adoption of Cloud Native Architecture, Part 2: Stabilization Gaps and Anti-Patterns」 https://www.infoq.com/articles/cloud-native-architecture-adoption-part2/

OPENMARU © 2021 | All Rights Reserved.

Cloud Native

디지털 플랫폼 정부와 클라우드 네이티브

디지털플랫폼정부 인프라 구축

4 디지털플랫폼정부 인프라 구축

4-1 공공부문 클라우드 정착

◆ **행정·공공기관의 정보시스템을 클라우드로 본격 전환하여 디지털플랫폼정부 서비스를 신속·유연하게 안정적으로 제공**

○ 각 기관이 **개별적으로 운영 중인 정보시스템을 클라우드로 본격 전환**

※ '23년 공공기관 대상 활용모델 지원 → ~'27년 행정·공공기관 활용모델 확대

- (**다양한 활용모델***) 유사한 영역의 **업무군별 활용모델 확대** 및 **기관의 특성과 여건을 고려한 다양한 형태의 활용모델 적용 지원**

  * 2개 이상의 클라우드 서비스를 동시에 활용하는 멀티클라우드, 하이브리드 클라우드 등

  ※ 지자체 활용모델 시범사업 : 클라우드서비스 제공기업(CSP)과 경남, 세종, 제주 등 지자체가 협력하여 광역지자체 단위로 지자체 정보시스템의 클라우드 전환 추진 중

- (**클라우드 최적화**) **MSA*** **기법의 도입 효과를 극대화하는 업무**를 중심으로 **클라우드 네이티브 구조의 설계**를 적용하여 **단계적 전환**

  * MicroService Architecture : 하나의 큰 애플리케이션을 여러 개의 작은 애플리케이션 단위로 쪼개어 손쉽게 변경·조합이 가능하도록 만든 아키텍처

  ※ 민간클라우드 전환을 위해 심층 컨설팅을 추진 중인 과기정통부 소관 시스템을 대상으로 선도적으로 시범 전환('23년)

행정안전부의 디지털플랫폼정부 추진계획(2023년~2027년)
- https://www.mois.go.kr/frt/sub/a06/b04/egovVision/screen.do

- Confidential -
OPENMARU © 2021 | All Rights Reserved.

행정안전부 보도자료

보도시점 2023. 10. 24.(화) 15:00

정부서비스, 클라우드 혁신 기반으로 더욱 안정되게 빠르게

- 클라우드 네이티브 중심, 공공부문 정보자원 클라우드 전환계획 발표
- 세계적인 기술발전 추세와 국내 클라우드 정책변화를 고려한 새로운 계획
- 다양한 서비스 수요에 신속하게 대응, 안정적인 대국민 서비스 제공 기대

□ 그동안 반복적으로 발생했던 대형 정부시스템의 접속장애가 앞으로는 대폭 개선될 전망이다. 현재는 하나의 대형시스템을 구축하는 방식이지만 앞으로는 디지털플랫폼정부 혁신인프라 위에서 작은 서비스의 묶음으로 시스템을 구축하는 클라우드 네이티브 방식이 적용되기 때문이다.

□ 행정안전부(장관 이상민)와 디지털플랫폼정부위원회(위원장 고진)는 10월24일(화), 이와 같은 내용을 담은 '클라우드 네이티브 중심, 공공부문 정보자원 클라우드 전환계획'을 공동으로 발표했다.

* 클라우드 : IT 자원을 사용자가 필요할 때마다 인터넷을 통해 이용하는 방식
** 클라우드 네이티브 : 정보시스템의 설계 단계부터 클라우드 기술을 적용해 클라우드의 장점을 최대한 활용할 수 있는 방식

□ 전 세계는 서비스의 신속성, 안정성, 비용효율성 등을 제고하기 위해 자체 구축에서 클라우드로 패러다임을 전환 중이며, 국내·외 유수의 기업 들은 클라우드의 효과를 극대화할 수 있도록 클라우드 네이티브 방식을 기반으로 디지털 혁신을 추진 중이다.

○ 정부도 이러한 추세에 빠르게 대응하기 위해 지난 4월 디지털플랫폼 정부 실현계획을 통해 클라우드 네이티브 적용을 디지털플랫폼정부의 핵심 과제로 발표한 바 있다.

□ 내년부터 신규 시스템을 구축하거나 기존 시스템을 고도화할 때, 기관은 불가피한 사유가 없는 한 민간 클라우드와 클라우드 네이티브 우선 적용을 검토하여야 한다.

○ 정보시스템의 노후 서버 교체 시기를 고려하여, 2030년까지 대다수의 시스템이 클라우드로 전환할 예정이다.

○ 정부는 2026년부터는 신규 클라우드 전환물량의 70% 이상(기존 시스템은 50% 이상)을 클라우드 네이티브 방식으로 전환할 예정이다.

□ 정부는 체계적인 클라우드 네이티브 전환을 위해 초기(~'25년)에는 행안부의 전환 지원사업 중심으로 추진하여 기술·제도적 이슈를 지속 발굴·보완하고, 이후는 개별기관 중심으로 전환을 추진할 계획이다.

○ 내년 상반기까지 행안부의 시스템을 대상으로 클라우드 네이티브 시 범 전환을 하는 등 다양한 시범사업을 추진하고, 이 경험을 바탕으로 전환 기준·절차, 가이드라인 등을 마련한다.

○ 정부가 민간의 클라우드서비스를 보다 쉽게 이용할 수 있도록 정부 내부 시스템과의 연계기반 체계를 구축할 계획이며, 공공부문 정보화 담당자를 대상으로 클라우드 네이티브 기술 도입·운영에 관한 교육도 지원할 예정이다.

○ 또한, 내년부터 클라우드 네이티브 적용 효과가 큰 시스템을 중점사업으로 추진하고, 행정·공공기관이 쉽게 클라우드를 적용할 수 있도록 기관을 대상으로 컨설팅을 집중 지원할 계획이다.

□ 고기동 행정안전부 차관은 "이번 계획이 디지털플랫폼 정부의 혁신 인프라 구현을 위한 출발점이 될 것으로 기대한다"며 "국민들께 빠르고 안정적인 공공 서비스를 제공해드릴 수 있도록 민간과 폭넓게 소통하고 협력하겠다"고 밝혔다.

담당 부서	행정안전부 디지털기반정책과	책임자	과 장	한혜남 (044-205-2821)
		담당자	사무관	민선미 (044-205-2828)
담당 부서	디지털플랫폼정부위원회 추진단 플랫폼기획과	책임자	과 장	윤희봉 (02-750-4781)
		담당자	사무관	지주연 (02-750-4793)

대한민국 정책브리핑 OPEN

openmaru

- Confidential -
OPENMARU © 2021 | All Rights Reserved.

클라우드 네이티브 관련 보도 자료 - 2024년 4월 11일 기준

정보통신신문

21개 정보시스템 '클라우드 네이티브' 전환

[정보통신신문=김연균기자] 고용산재보험, 국가대중 교통정보 등 21개 디지털서비스가 '클라우드 네이티브' 방식으로 전환된다. 클라우드는 IT자원을...

15시간 전

네이트

'클라우드 네이티브' 전환으로 행정망 마비·장애 없앤다

한눈에 보는 오늘 : 정치 - 뉴스 : 행정안전부가 올해부터 대국민 디지털서비스의 중단없는 원활한 서비스를 위해 본격적으로 클라우드 네이티브 전환...

1일 전

청년일보

디지털 행정 안정성 높인다...서비스 장애 등 방지 '클라우드 네이티브' 전환

【 청년일보 】 대국민 디지털 서비스 제공 과정에서 사용자 집중 시 서비스 장애 발생 등 클라우드 문제 해결을 위해 정부가 안정성 제고를 위한...

15시간 전

뉴스1

고용산재보험, 내년까지 24시간 가능한 '클라우드 네이티브' 전환한다

(서울=뉴스1) 박우영 기자 | 정부가 24시간 원활한 서비스 제공이 필요한 21개 정보시스템을 내년까지 전환하는 등 본격적인 '클라우드 네이티브'...

1일 전

데이터넷

행안부, 21개 공공 서비스 클라우드 네이티브 전환 시작

[데이터넷] 행정안전부는 고용산재보험 토탈서비스, 국가대중 교통정보 등 21개 행정·공공기관의 서비스를 클라우드 네이티브 환경으로 전환하며,...

4시간 전

시사포커스

정부, 중단없는 디지털 행정...'클라우드 네이티브' 전환 추진

[시사포커스 / 이청원 기자] 정부는 대국민 디지털 서비스를 중단없이 제공하고, 장애 발생 시 즉각 개선을 위한 공공부문 정보 시스템의 '클라우드...

1일 전

- Confidential -

OPENMARU © 2021 | All Rights Reserved.

행정안전부 보도 자료

openmaru

행정안전부 보도자료 다시 대한민국! 새로운 국민의 나라

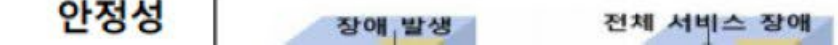
보도시점 (온라인) 2024. 4. 10.(수) 12:00
(지 면) 2024. 4. 11.(목) 조간

클라우드 네이티브 기술로 대국민 서비스 중단 없이 빠르게 제공한다

- 고용산재보험 등 21개 정보시스템 본격적으로 클라우드 네이티브 전환 추진

※ **클라우드란?** IT자원(서버, 저장소 등)을 사용자가 직접 구매해서 사용하지 않고, 필요할 때마다 필요한 만큼 인터넷을 통해 이용하는 방식을 말한다.

※ **클라우드 네이티브란?** 정보시스템 설계 단계부터 클라우드 기술(기능분리, 자동확장, 자동배포 등)을 적용하여 '안정성', '확장성', '신속성' 등을 최대한 활용할 수 있는 방식

구분	클라우드 네이티브 적용 전	클라우드 네이티브 적용 후
안정성	일부 기능 장애가 전체 서비스 장애를 초래	독립된 구조로 인해 개별 서비스 장애가 전체 서비스로 전파되지 않음
		

OPENMARU © 2021 | All Rights Reserved.

OPENMARU NEWSPAPER

클라우드 네이티브가 궁금할땐?
오픈마루의 "클라우드 네이티브 무상 컨설팅"을 신청하세요 클라우드 네이티브의 A부터 Z까지 모두 알려 드립니다.

정부 2030년까지 클라우드로 전환 · 설계부터 클라우드 네이티브 적용 · 클라우드 네이티브 인프라로 전환

접속자 몰려도 먹통 없는 정부 시스템 만든다...
2030년까지 클라우드로 전환

행안부, '클라우드 네이티브 중심, 공공부문 정보자원 클라우드 전환계획' 발표

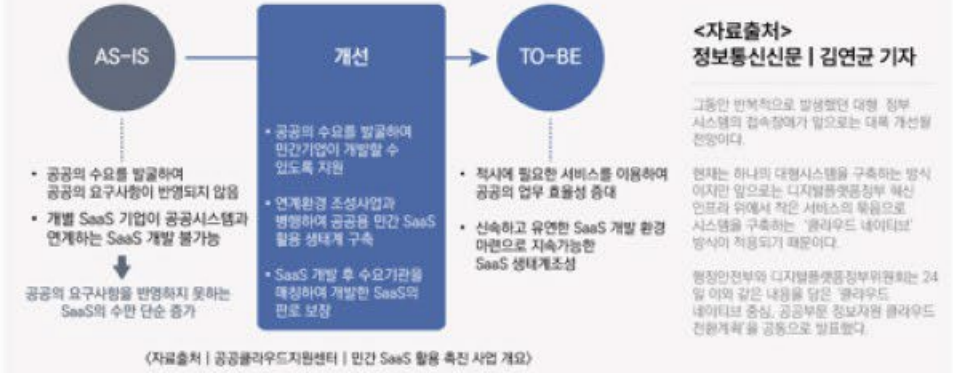
<자료출처 | 정책브리핑 www.korea.kr>

"전환계획에 따르면, 내년부터 신규 시스템을 구축하거나 **기존 시스템**을 **고도화**할 때 기관은 불가피한 사유가 없는 한 **민간 클라우드**와 **클라우드 네이티브 우선 적용**을 검토해야 한다."

"설계부터 클라우드 적용, 정부시스템 대수술"

민간 클라우드 우선 검토
2026년 신규 물량 70%
클라우드 네이티브 전환

대통령직속 디지털플랫폼정부위원회



〈자료출처 | 공공클라우드지원센터 | 민간 SaaS 활용 촉진 사업 개요〉

<자료출처>
정보통신신문 | 김연균 기자

그동안 반복적으로 발생했던 대형 정부 시스템의 접속장애가 앞으로는 대폭 개선될 전망이다.

현재는 하나의 대형시스템을 구축하는 방식이지만 앞으로는 디지털플랫폼정부 핵심 인프라 위에서 작은 서비스의 묶음으로 시스템을 구축하는 '클라우드 네이티브' 방식이 적용되기 때문이다.

행정안전부와 디지털플랫폼정부위원회는 24일 이와 같은 내용을 담은 '클라우드 네이티브 중심, 공공부문 정보자원 클라우드 전환계획'을 공동으로 발표했다.

정부 시스템 클라우드 네이티브 우선 적용 원칙 세운다

단순 클라우드 전환, 체감 효과 더뎌

"디지털플랫폼정부 실현을 위한 주요 과제들이 본격적인 궤도에 올라 속도감 있게 성과를 창출"

<자료출처> 지디넷코리아 | 박수형 기자

정부 시스템을 단순히 클라우드로 전환하는 데 그치지 않고, 클라우드 중심으로 설계하는 방식이 추진된다. 이를 통해 클라우드 효과성 체감을 높이겠다는 목표다.

대통령 직속 디지털플랫폼정부위원회는 24일 오후 정부서울청사에서 제4차 위원회 전체회의를 열고, 행정안전부가 마련한 클라우드 네이티브 중심 공공부문 정보자원 클라우드 전환 계획을 심의했다.

제4차
디지털플랫폼정부위원회 전체회의

2023. 10. 24 (화) 15:00
정부서울청사 별관 3층 국제회의실

대통령직속 디지털플랫폼정부위원회

"앞으로 정부 시스템 '클라우드 네이티브' 우선 적용"... 클라우드 효과 극대화

디지털플랫폼정부위원회, 4차 전체회의
'**클라우드 네이티브**' 적용을 기본원칙으로
응용시스템을 클라우드에 맞게 재설계
단순 이전보다 클라우드 체감 효과 커

<자료출처> 헤럴드경제 | 김현일 기자

고진(가운데) 디지털플랫폼정부위원회 위원장 [연합]

정부서비스 '클라우드 네이티브' 기반 인프라로 전환 공식화

행정안전부

<자료출처> 아이티데일리 | 한정호 기자

[아이티데일리] 행정안전부(장관 이상민)와 디지털플랫폼정부위원회(위원장 고진)가 24일 정부시스템 개선을 위해 공공 시스템을 마이크로서비스 아키텍처(MSA)를 적용한 클라우드 네이티브 방식의 인프라로 전환한다는 내용을 담은 '클라우드 네이티브 중심, 공공부문 정보자원 클라우드 전환계획'을 공동으로 발표했다.

현재 국내 외 유수의 기업들은 클라우드의 효과를 극대화할 수 있도록 클라우드 네이티브 방식을 기반으로 디지털 혁신을 추진 중이다. 정부도 이러한 추세에 빠르게 대응하기 위해 지난 4월 디지털플랫폼정부 실현계획을 통해 클라우드 네이티브 적용을 디지털플랫폼정부의 핵심 과제로 발표한 바 있다.

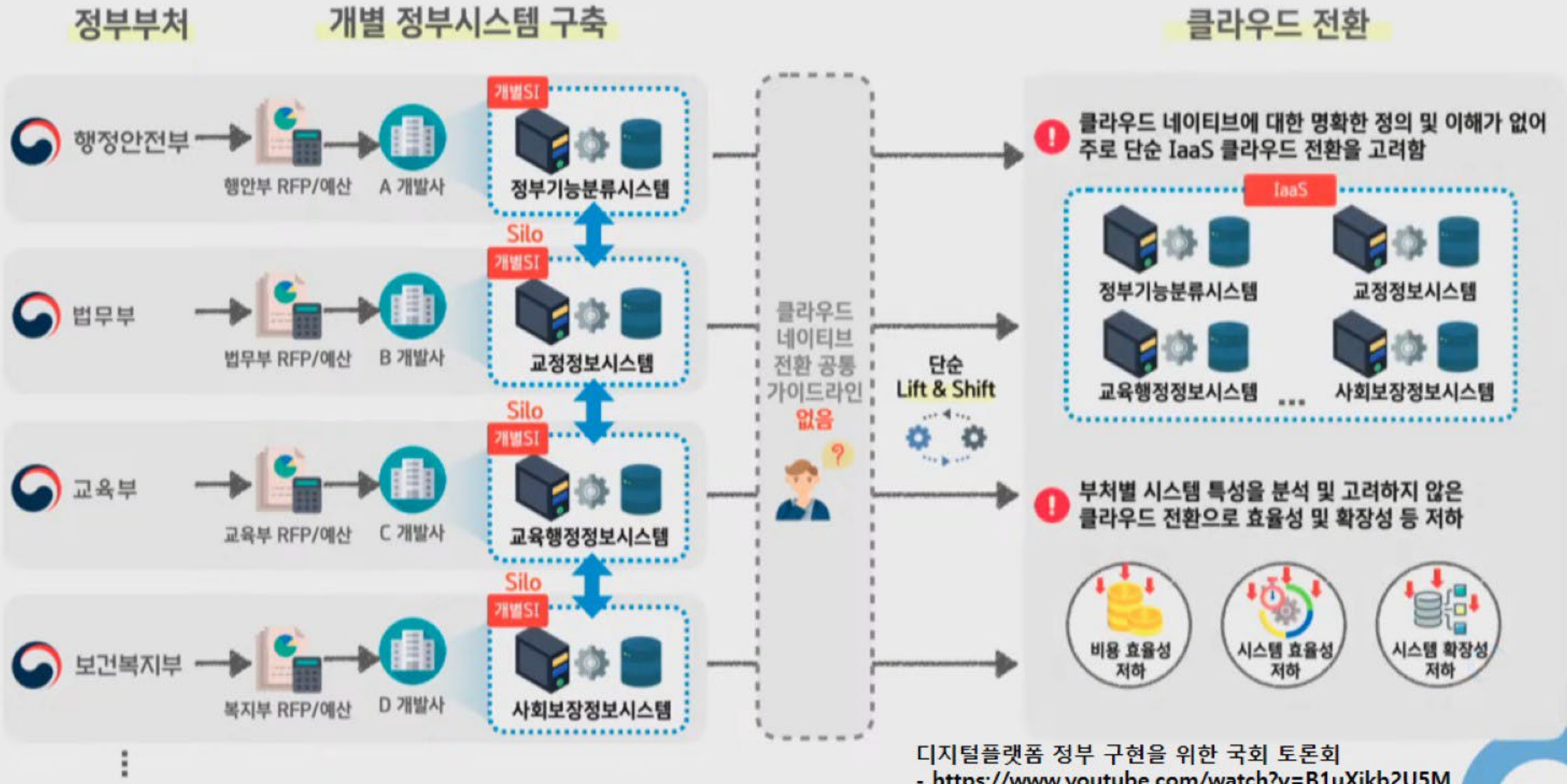
해당 계획에 따르면 내년부터 신규 시스템을 구축하거나 기존 시스템을 고도화할 때, 공공기관은 불가피한 사유가 없는 한 민간 클라우드와 클라우드 네이티브 우선 적용을 검토해야 한다.

행안부는 정보시스템의 노후 서버 교체 시기를 고려해, 2030년까지 대다수 공공 정보시스템의 클라우드 전환을 추진할 계획이다. 또한 2026년부터는 신규 클라우드 전환 물량의 70% 이상(기존 시스템은 50% 이상)을 클라우드 네이티브 방식으로 전환할 예정이다.

openmaru

OPENMARU © 2021 | All Rights Reserved.

과제5 : (현재) 정부시스템과 클라우드 네이티브 전환 방식의 문제점



디지털플랫폼 정부 구현을 위한 국회 토론회
- https://www.youtube.com/watch?v=B1uXikb2U5M

과제5 : (개선) 정부시스템별 최적의 클라우드 네이티브 전환 방안

정부부처 | 개별 정부시스템 구축 | 단계별 전환 프로세스 | 클라우드 네이티브 전환

* 약 1만 7천여개

- 행정안전부 → 행안부 RFP/예산 → A 개발사 → 개별SI → 정부기능분류시스템
- Silo
- 법무부 → 법무부 RFP/예산 → B 개발사 → 개별SI → 교정정보시스템
- Silo
- 교육부 → 교육부 RFP/예산 → C 개발사 → 개별SI → 교육행정정보시스템
- Silo
- 보건복지부 → 복지부 RFP/예산 → D 개발사 → 개별SI → 사회보장정보시스템

클라우드 네이티브 전환 공통 가이드라인 제공

Step1
정부시스템 중요도 및 특성 분석

시스템 지표 항목 및 아키텍처, CSAP 등급, 운영 방안, 비용 등 정부시스템 등급 분석

Step2
클라우드 네이티브 전환 목표 설정

성숙도 모델 기반 클라우드 전환 목표 설정
(1) Cloud Ready, (2) Cloud Optimized
(3) Cloud Native

Step3
세부적인 클라우드 전환방법 결정

클라우드 전환 목표모델에 따라 클라우드 전환방법 결정 및 이행(7R 참고)

참조:microsoft Azure cloud native

- A 시스템 → Retain(유지)
- B 시스템 → Retire(폐기)
- C 시스템 → Cloud-Ready → Rehost
- D 시스템 → Rehost
- E 시스템 → Cloud-Optimized → Replatform
- F 시스템 → Repurchase
- G 시스템 → Cloud-Native → Refactor
- H 시스템 → Refactor

디지털플랫폼 정부 구현을 위한 국회 토론회
- https://www.youtube.com/watch?v=B1uXikb2U5M

클라우드 네이티브로 행정/공공 업무가 이렇게 바뀝니다.

NIA 한국지능정보사회진흥원

클라우드 네이티브 기반
'포스트 코로나 시대의 디지털 정부혁신 발전계획'과
'한국판 뉴딜 종합계획'에 따른

빠른 대국민 서비스가 가능합니다

클라우드 네이티브 기반
공공 클라우드 센터 전면 전환에 따른
센터, 부서, 업무 간의

상호 운용성 제공이 가능합니다.

클라우드 네이티브 기반 마이크로 서비스
품질 강화(즉시 배포 등)에 따른

24 x 365 무중단 행정 서비스

클라우드 네이티브 기반
개발부터 운영까지 전주기 관리를 통한 (DevOps, CI/CD)

서비스 운영 자동화가 가능합니다.

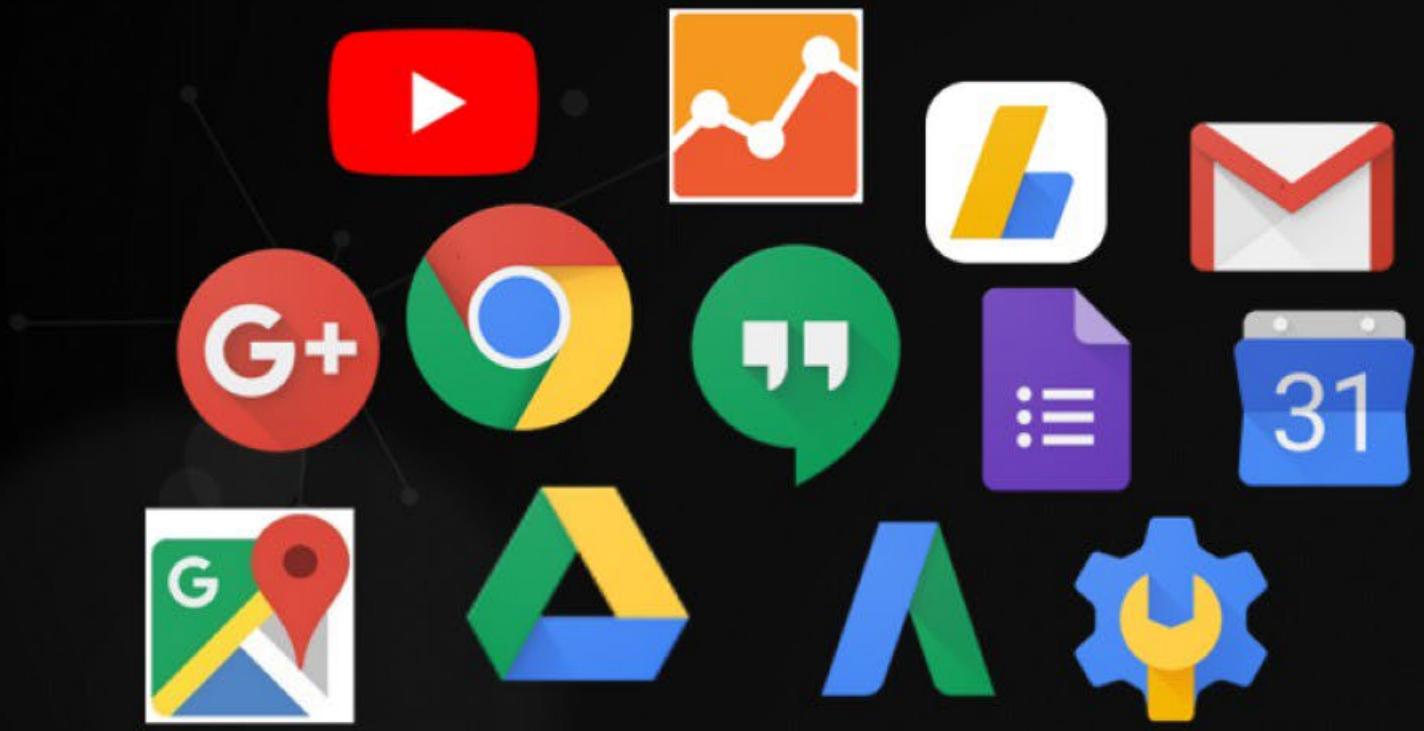
Platform As A Service

누가 왜 클라우드 네이티브를 시작한 건가요?

Google 의 모든 서비스는 컨테이너 에서 실행

openmaru

- Gmail , 검색, 지도 ...
- MapReduce , GFS , Colossus ...
- Google Compute Engine 가상 머신도 컨테이너 에서 실행!
- 매주 20 억개 이상의 컨테이너 를 실행 중



- Confidential -

GOOGLE 과 컨테이너

- **Google의 업무 방식**

Gmail에서 YouTube, 검색에 이르기까지 Google의 모든 제품은 컨테이너에서 실행됩니다.

개발팀은 컨테이너화를 통해 더욱 신속하게 움직이고, 효율적으로 소프트웨어를 배포하며 전례 없는 수준의 확장성을 확보할 수 있게 되었습니다. **Google은 매주 수십억 개가 넘는 컨테이너를 생성**합니다. **지난 10여 년간 프로덕션 환경에서 컨테이너화된 워크로드를 실행하는 방법에 관해 많은 경험**을 쌓으면서 Google은 커뮤니티에 계속 이 지식을 공유해 왔습니다.

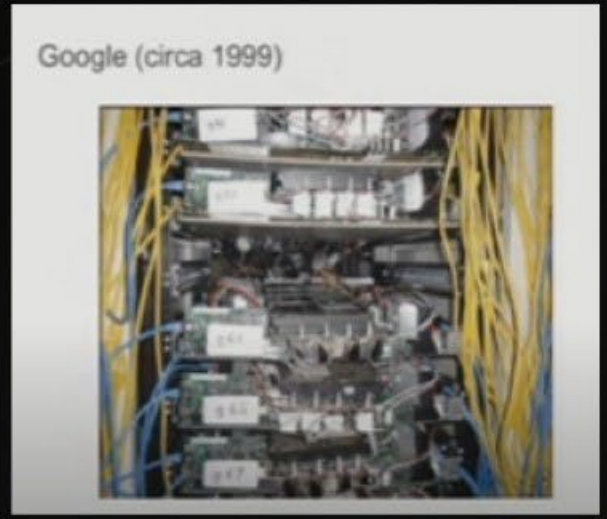
초창기에 cgroup 기능을 Linux 커널에 제공한 것부터 **내부 도구의 설계 소스를 Kubernetes 프로젝트로 공개**한 것까지 공유의 사례는 다양합니다. 그리고 이 전문 지식을 Google Cloud Platform으로 구현하여 개발자와 크고 작은 규모의 회사가 최신의 컨테이너 혁신 기술을 쉽게 활용할 수 있도록 하였습니다.

Source - https://cloud.google.com/containers/

Google 은 하드웨어의 문제를 어떻게 해결 했을까요?

- Google은 2008년 서버 수는20만대 이상
  - 서버 고장이 연간1000건, HDD고장은 수천 건, 전원 분배 장치가 고장 나면 한 번에 500대 이상 정지
  - 2003년 부터 Borg 사용하기 시작했으며, 당시에도 대수는 꽤 되었음

2016년 7월 Gartner 보고서에 따르면 구글은 당시 **약 250만 대**의 서버를 운영했다고 추정

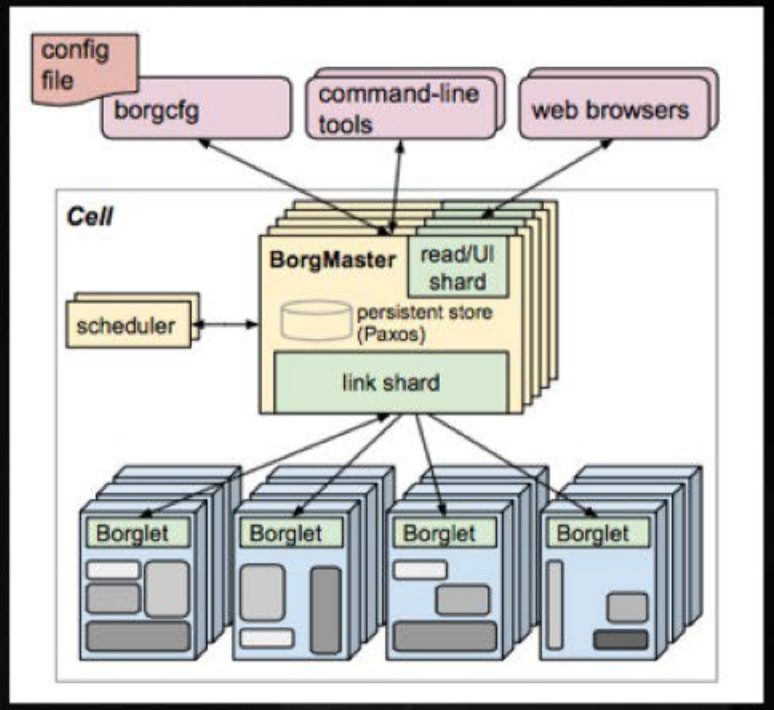


Google I/O 2008 - Underneath the Covers at Google
- https://youtu.be/qsan-GQaeyk?si=ktkKndmqgpn1SXTj

- Confidential -
OPENMARU © 2021 | All Rights Reserved.

Borg, Omega, and Kubernetes

- Borg
  - 2003년부터 Google내에서 사용된 컨테이너 오케스트레이션
- Omega
  - Google 내부에서 사용되는 컨테이너 오케스트레이션
  - Borg 이후, Kubernetes 이전에 만들어 운영
- Kubernetes
  - 2014년부터 Github 에 공개된 컨테이너 오케스트레이션
  - Borg 그리고 Omega 를 참고로 만들어진 오픈소스
  - 클라우드 네이티브의 시작



<Borg 는 Kubernetes 와 매우 유사함>

"Borg is Google's internal container management platform. That project was started back in 2003"

- Kubernetes Podcast "Borg, Omega, and Kubernetes"

Source - Kubernetes Podcast"Borg, Omega, Kubernetes and Beyond, with Brian Grant"
https://kubernetespodcast.com/episode/043-borg-omega-kubernetes-beyond/

OPENMARU © 2021 | All Rights Reserved.

About Kubernetes

- 쿠버네티스(K8s)는 컨테이너화된 애플리케이션을 자동으로 배포, 스케일링 및 관리해주는 오픈소스 소프트웨어
- 쿠버네티스”, “쿠베르네테스”,“K8s”, “쿠베”, “쿠버”, “큐브”라고 부르며
- Go로 작성된 오픈 소스 , 오픈소스 S/W (Apache License 2.0) 라이선스
- 리눅스 재단 (Linux Foundation)산하 Cloud Native Computing Foundation (CNCF) 에서 관리
- 구글에서 개발하고 설계한 플랫폼으로서 사내에서 이용하던 컨테이너 클러스터 관리 도구인 “Borg”의 아이디어를 바탕으로 개발

> "Kubernetes is open source-a contrast to Borg and Omega, which were developed as purely Google-internal systems. "
>
> *- Borg, Omega, and Kubernetes*

컨테이너는 클라우드 에서 Java 와 같이 벤더 종속성 해제

openmaru

2000 년 – Java 를 통한 Vendor Lock-In 해제

OS	hp UX	AIX	solaris	Linux	Windows
CPU	Intanium	Power	Sparc	X86	X86
Vendor	hp	IBM	ORACLE	THE LINUX FOUNDATION	Microsoft

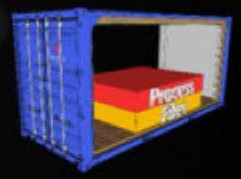
→ Java

2020 년 – 컨테이너와 Kubernetes 를 통한 Vendor Lock-In 해제

Write Container, run anywhere!

Hypervisor	Xen Project	KVM	Hyper-V	Xen Project	Xen Project
Public Cloud	aws	Google Cloud	Microsoft Azure	ORACLE	IBM Cloud

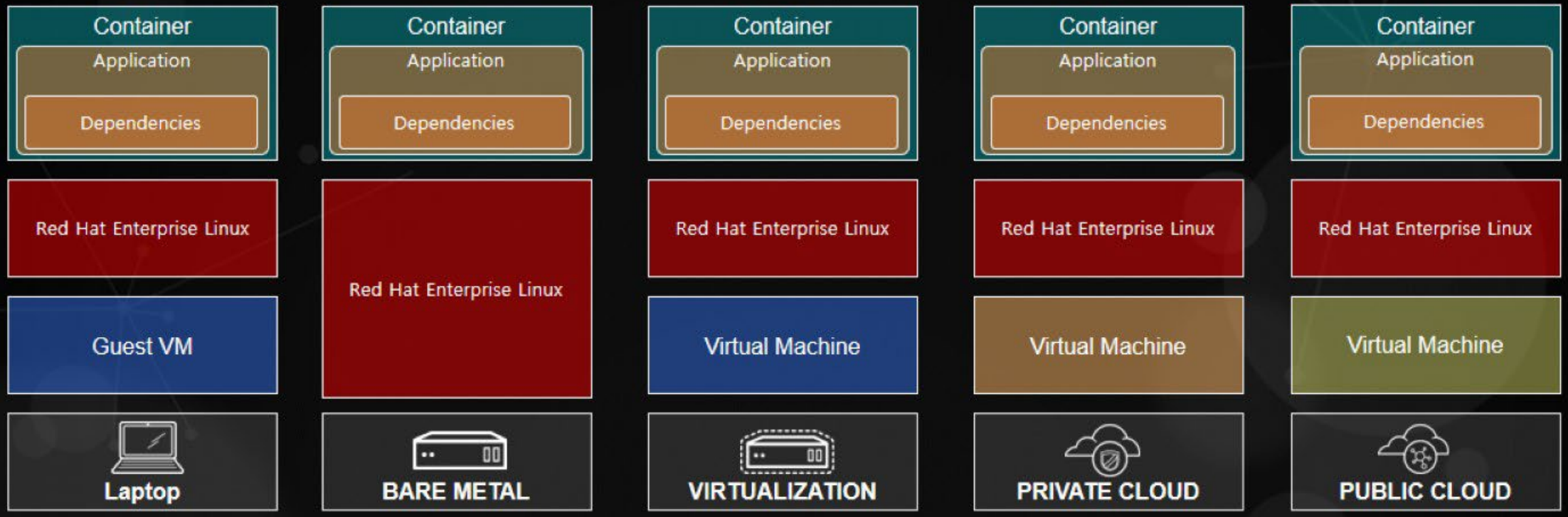
→ kubernetes



OPENMARU © 2021 | All Rights Reserved.

컨테이너의 이동성

- Linux 커널 구조를 이용하고있다 = Linux 움직이는 모든 환경에서 이동성



OPENMARU © 2021 | All Rights Reserved.

클라우드 네이티브 개념과 기술요소들
Cloud Native

Cloud Native ERA

- 애플리케이션을 실행하기 위한 최적의 인프라 최적 솔루션 중 하나
  - 컨테이너 기술
    - 애플리케이션이 동작하기 위한 운영 환경을 함께 패키징
    - 개발자를 위한 이미지 빌드/배포 용이성
    - 빠른 애플리케이션 실행과 낮은 오버헤드

Develper Experience장점

- Kubernetes(컨테이너 오케스트레이션)
  - 애플리케이션 실행을 먼저 생각할 때
  - 어떤 인프라를 만들지 주축으로 설계된 인프라 기반

Reconcilliation model의 정교함

CNCF Cloud Native Definition v1.0

openmaru

클라우드 네이티브 기술을 사용하는 조직은 현대적인 퍼블릭, 프라이빗, 그리고 하이브리드 클라우드와 같이 동적인 환경에서 확장성 있는 애플리케이션을 만들고 운영할 수 있다.

컨테이너, 서비스 메시, 마이크로서비스, 불변의 인프라스트럭처, 그리고 선언적 API가 전형적인 접근 방식에 해당한다.

이 기술은 회복성이 있고, 관리 편의성을 제공하며, 가시성을 갖는 느슨하게 결합된 시스템을 가능하게 한다.

견고한 자동화와 함께 사용하면, 엔지니어는 영향이 큰 변경을 최소한의 노력으로 자주, 예측 가능하게 수행할 수 있다.

Cloud Native Computing Foundation은 **벤더 중립적인 오픈소스 프로젝트 생태계**를 육성하고 유지함으로써 해당 패러다임 채택을 촉진한다.

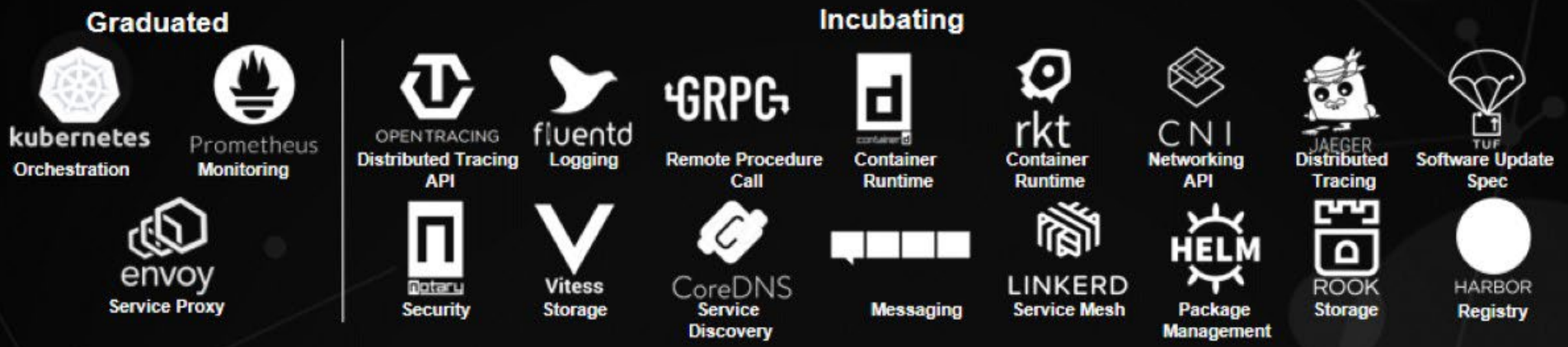
우리 재단은 최신 기술 수준의 패턴을 대중화하여 이런 혁신을 누구나 접근 가능하도록 한다.

Source:https://github.com/cncf/toc/blob/master/DEFINITION.md#%ED%95%9C%EA%B5%AD%EC%96%B4

- Confidential -
OPENMARU © 2021 | All Rights Reserved.

Cloud Native Computing Foundation

- **Non-profit, part of the Linux Foundation; founded Dec 2015**



- **Platinum members:**

Alibaba Cloud, aws, cisco, Dell Technologies, docker, FUJITSU, Google Cloud, HUAWEI, IBM Cloud, intel, JD.COM, MESOSPHERE, Microsoft Azure, ORACLE, Pivotal, redhat, SAMSUNG SDS, SAP, vmware

OPENMARU © 2021 | All Rights Reserved.

클라우드 전환 클라우드 이민과 클라우드 네이티브

Cloud Immigrant
(클라우드 이민 단계)

Cloud Native
(클라우드 네이티브 단계)

기존 어플리케이션	AS-IS 시스템	클라우드 네이티브 어플리케이션
• 기존 애플리케이션 그대로 • 기존 배포 방법	어플리케이션	어플리케이션실행영역 마이크로서비스
• 기존 운영 환경과 방법 그대로 • 시스템 S/W 설치와 라이선스 • 비용과 빌링 방법 변경 • 가상화 이미지로 구성	미들웨어/ OS/DB HW	컨테이너 PaaS (플랫폼)
미들웨어/ OS/DB 가상화 IaaS (인프라)		

- 기존 애플리케이션 또는 MSA 로 전환
- 컨테이너 이미지 배포
- 시스템 S/W 컨테이너 이미지제공 (번들, OS, Java, WAS)
- 하이퍼바이저와 Guest OS 불필요

OPENMARU © 2021 | All Rights Reserved.

클라우드 네이티브를 저해하는 요인

- **생각과 시스템을 클라우드 네이티브 하게 전환하지 못하면, 클라우드라도 개선이 없음**

클라우드를 임대하여 사용하는 것일 뿐

- 가상머신과 스토리지를 임대하여 사용하고 있을 뿐, 기존의 인프라와 다르지 않음

클라우드 특징에 맞게 설계하고 운영 하지 않음

- 클라우드 특성을 이해하지 못하고 기존 인프라를 단순히 대체하여 설계
- 비용 부분에서만 정액제 클라우드로 전환하였으나, 벤더 종속성과 비용만 높아짐

인프라만 클라우드 일 뿐 조직은 그대로

- 기존의 개발팀과 운영팀이 수행하던 역할과 프로세스 그대로 운영

클라우드로 전환했으나 구인난과 고비용 구조로 더 큰 문제

- 클라우드를 이용하고 있음에도 불구하고 수작업 프로세스에서 벗어나지 못함
- 운영 인력 부족과 업무 효율성을 개선하지 못함

OPENMARU © 2021 | All Rights Reserved.

Cloud Immigrant vs. Cloud Native

구분	Cloud Immigrant	Cloud Native
서비스 모델	가상화 기반 IaaS (Infrastructure As A Service)	컨테이너 기반 PaaS (Platform As A Service)
디자인	On Premise 에 구축된 시스템을 클라우드로 이전하여 운영	시작 단계부터 클라우드의 장점인 민첩성, 확장성 그리고 이동성을 최대한 활용할 수 있도록 설계
구현	특정 클라우드 벤더에 의존적인 설정이 있어 구축에 시간이 걸림	어떤 클라우드 환경에서도 빠르고 효율적으로 전환 (Portability)
확장성	애플리케이션 업데이트가 수작업이기 때문에 장시간의 다운타임일 필요하고 Scale In/Out 이 어려움	컨테이너와 MSA 기반으로 서비스에 영향을 주지 않고, 업데이트가 필요한 서비스만 변경할 수 있으며, 서비스 단위의 Scale In/out 지원
비용	애플리케이션이 커질 수록 인프라 비용이 상승	인프라 부분의 종속성이 없어 비용이 저렴
유지보수	버전관리, 설치 그리고 구성관리가 수작업이며 복잡함	CI (Continuous Integration) / CD (Continuous Delivery)

OPENMARU © 2021 | All Rights Reserved.

Open Source Software

"누군가가 나의 등잔의
심지에서 불을 붙여가도
내 등잔의 불은 여전히
빛나고 있습니다."

미국의 정치가 토머스 제퍼슨

openmaru
APM

Cloud Native

컨테이너 기술

IT Infrastructure 에 대한 고민들

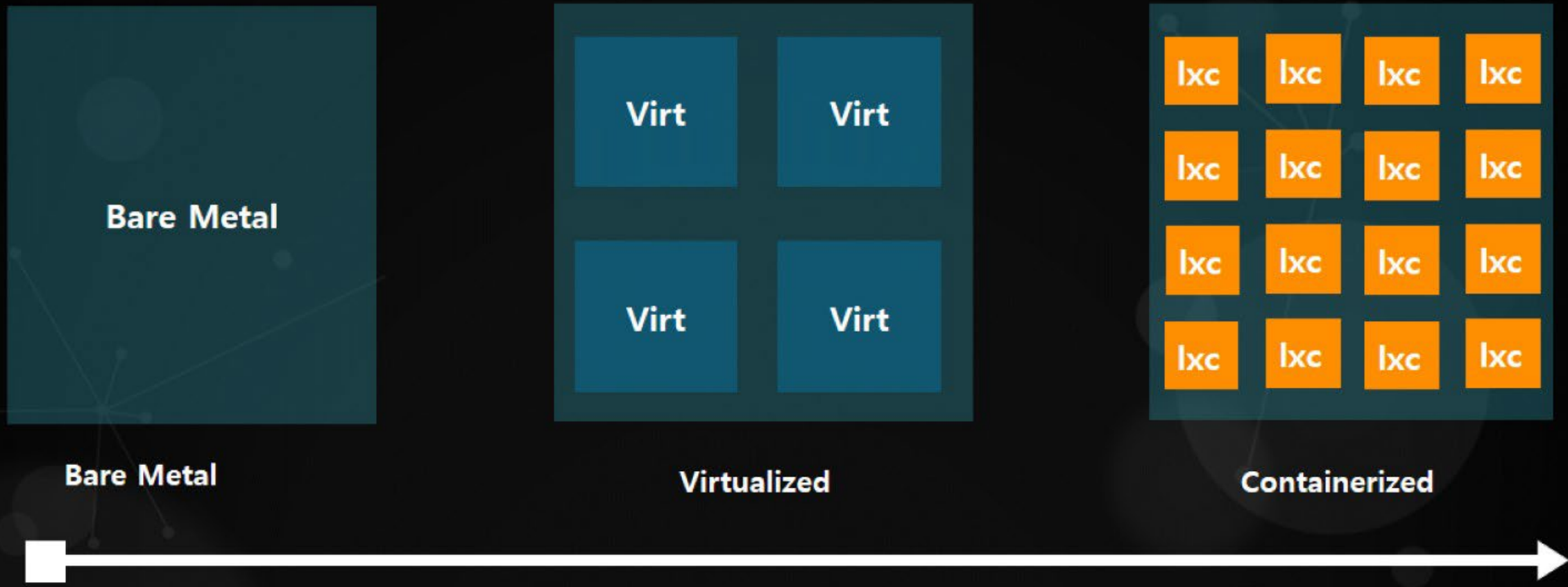
- IT 운영 부담을 줄이려고 하는 개발팀,

  시스템 운영팀, 네트워크 운영팀, DEVOPS 팀

  - 애플리케이션 변경 요건이 많고,하루에도 여러 번 배포
  - 대규모 이벤트로 인한 부하에 대응할 수 있는 시스템 요구
  - 빈번한 설정 변경과 시스템 작업으로 인한 이력 관리
  - 복잡한 작업 절차에 대한 자동화
  - 비즈니스 요구에 적합한 배포 정책 요구와 롤백 방안

OPENMARU © 2021 | All Rights Reserved.

Evolution of Infrastructure Architectures



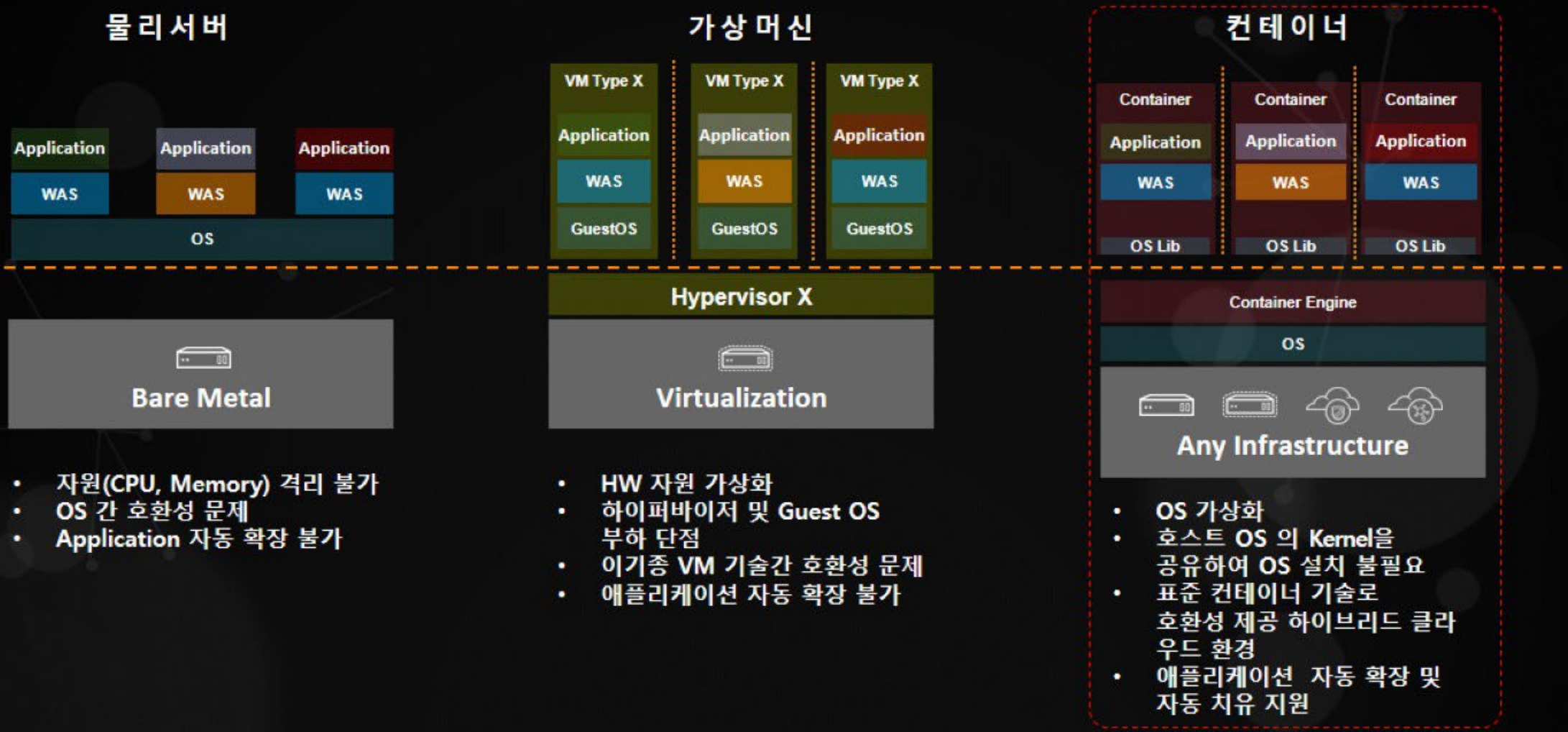
OPENMARU © 2021 | All Rights Reserved.

before containerization

After Containerization
컨테이너에 화물을 싣고
openmaru

WHY CONTAINER ?

- 자원 효율성, 자원 격리, 호환성, Auto Scaling, DevOps, MSA, 관리 편의성



- 자원(CPU, Memory) 격리 불가
- OS 간 호환성 문제
- Application 자동 확장 불가

- HW 자원 가상화
- 하이퍼바이저 및 Guest OS 부하 단점
- 이기종 VM 기술간 호환성 문제
- 애플리케이션 자동 확장 불가

- OS 가상화
- 호스트 OS 의 Kernel을 공유하여 OS 설치 불필요
- 표준 컨테이너 기술로 호환성 제공 하이브리드 클라우드 환경
- 애플리케이션 자동 확장 및 자동 치유 지원

- Confidential -
OPENMARU © 2021 | All Rights Reserved.

오버헤드 - Containers vs. VMs

- OS에서 응용 프로그램을 작동하는 경우, 하드웨어 가상화에서는 가상화 된 하드웨어 및 하이퍼바이저를 통해 처리하기 때문에 물리적 시스템보다 처리에 부가적인 시간 (오버 헤드)가 필요
- 컨테이너 형 가상화 커널을 공유하고 개별 프로세스가 작업을 하는 것과 같은 정도의 시간 밖에 걸리지 않기 때문에 대부분 오버 헤드가 없음

OPENMARU © 2021 | All Rights Reserved.

가상머신(VM) 부팅과 컨테이너 시작 시간 비교

openmaru

가상머신(VM) 부팅 절차 및 시간

50초

하드웨어 초기화(BIOS/UEFI)

20초

BIOS or UEFI를 실행하여 하드웨어를 초기화

```
Rocky Linux (4.18.0-477.10.1.el8_8.x86_64) 8.8 (Green Obsidian)
Rocky Linux (0-rescue-edd5fc58d9194044b0479268299b667b) 8.8 (Gre
```

부트로더 실행

5초

RAM Disk에 커널을 로딩하여 실행

```
Nov  6 05:20:25 localhost kernel: Command line: BOOT_IMAGE=(hd0,msdos1)/vmlinuz-4.18.0-477.10.1.el8_8.x86_64 root=UUID
=ac0f7df8-bbec-4dfa-a668-25871accd0b4 ro console=ttyS0 selinux=0
```

커널 실행

23초

시스템 하드웨어, 메모리 관리, 프로세스 관리, 파일시스템, 네트워크 실행

```
Nov  6 05:20:26 localhost systemd[1]: Detected virtualization qemu.
Nov  6 05:20:26 localhost systemd[1]: Detected architecture x86-64.
Nov  6 05:20:26 localhost systemd[1]: Running in initial RAM disk.

Nov  6 05:20:33 localhost systemd[1]: Mounted /sysroot.
Nov  6 05:20:38 localhost systemd[1]: Started Remount Root and Kernel File Systems.
```

systemd 프로세스 실행

1초

시스템 서비스 실행(예제는 Apache HTTPD 실행)

```
[Mon Nov 06 05:20:46.795029 2023] [core:notice] [pid 608:tid 140261225593152] AH00094: Command line: '/usr/sbin/httpd
-D FOREGROUND'
```

컨테이너 시작 시간

1초

X

X

X

컨테이너는 OS가 없기 때문에
OS의 부팅 절차가 없음

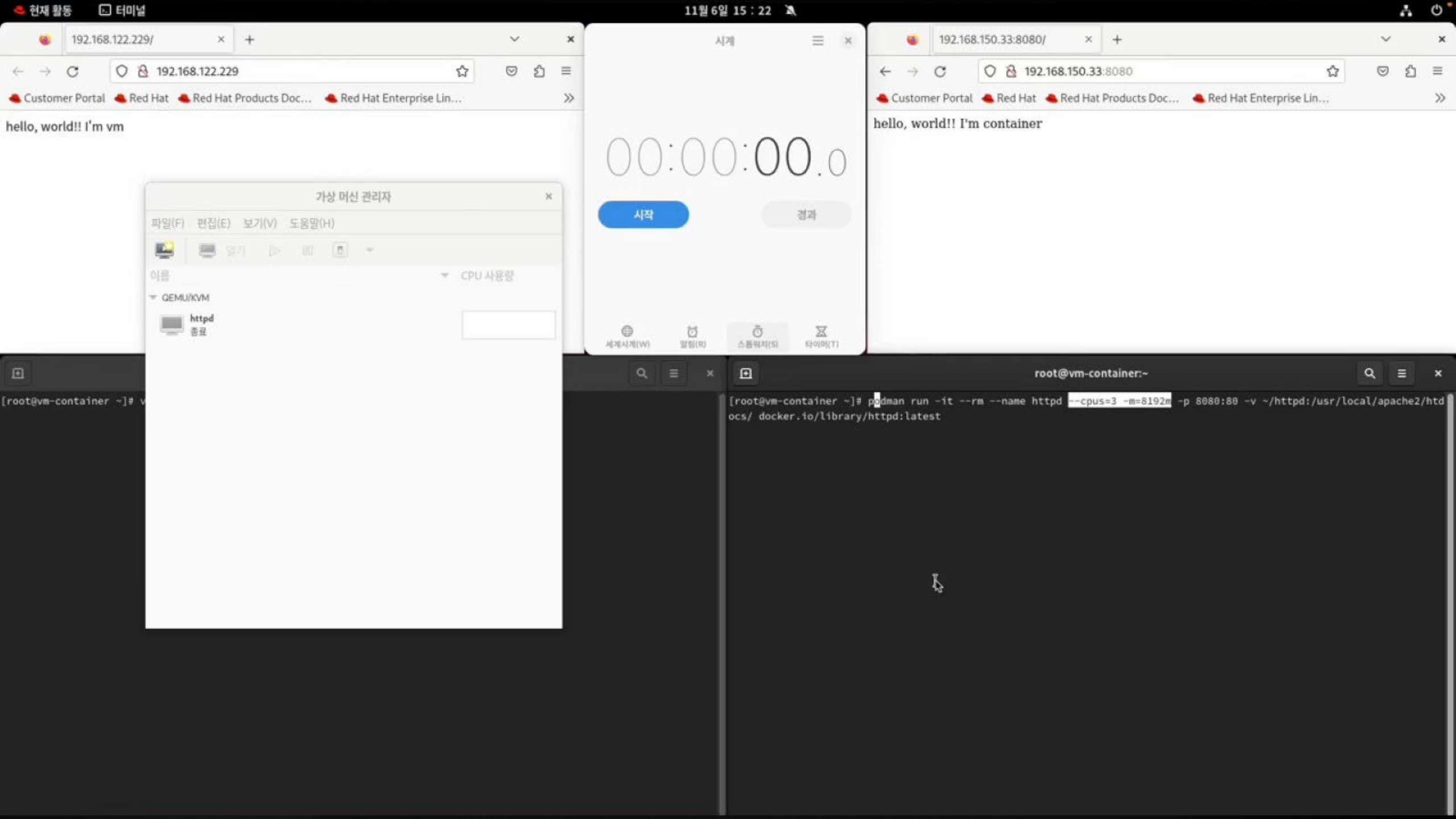
Apache HTTPD 프로세스 실행

1초

초기 프로세스 실행(예제는 Apache HTTPD 실행)

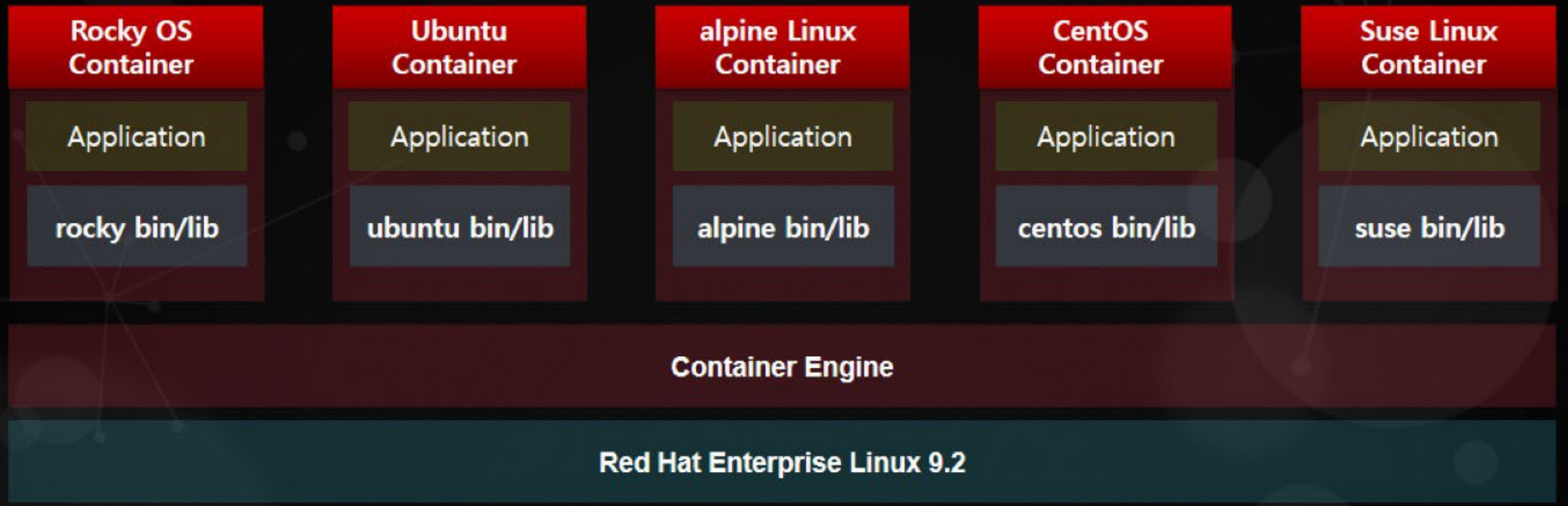
```
[Mon Nov 06 11:03:31.775057 2023] [core:notice] [pid 1:tid 140538360010624] AH00094: Command line: 'httpd -D FOREGROUN
D'
```


OPENMARU © 2021 | All Rights Reserved.

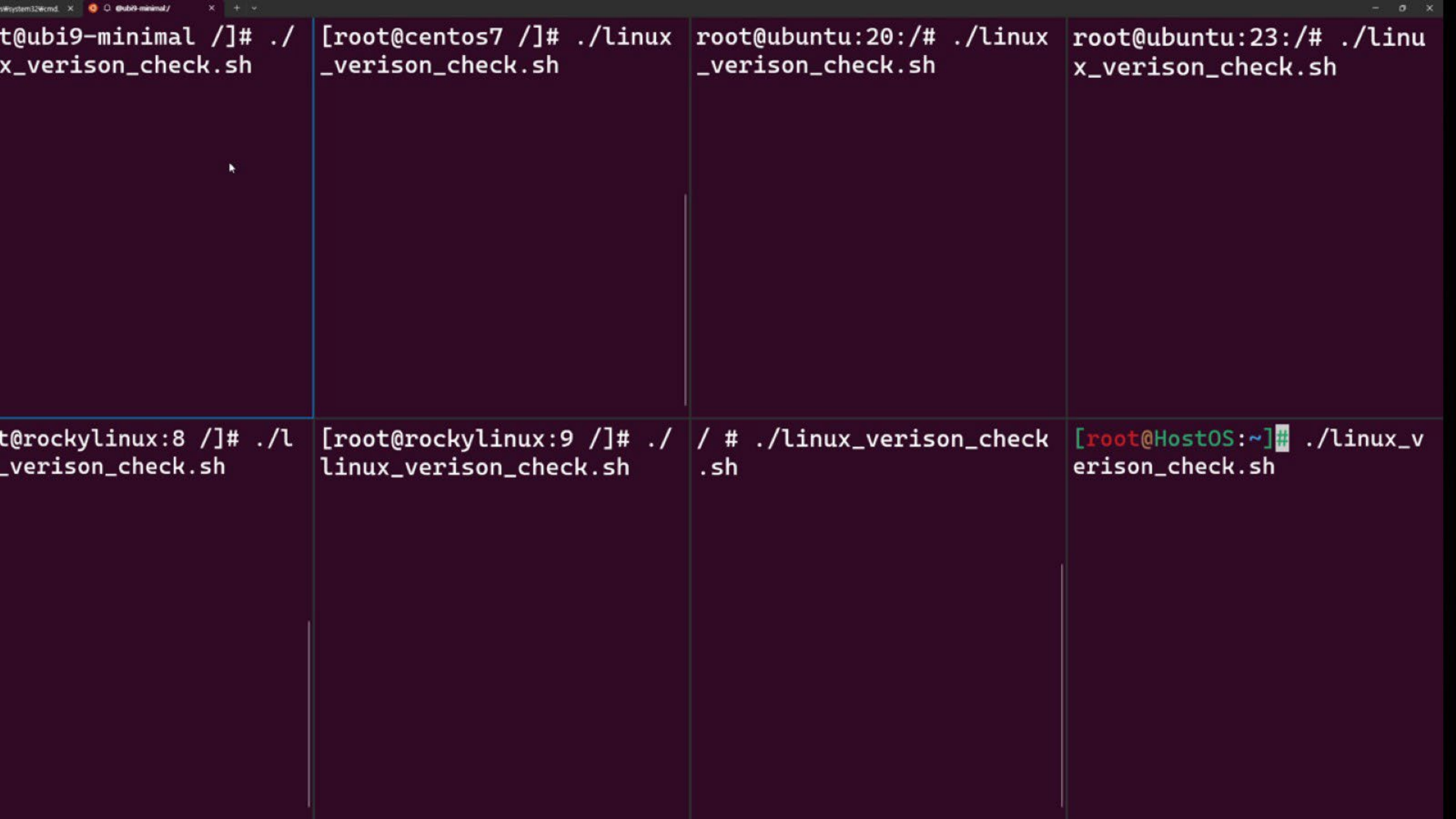

현재 활동
터미널
11월 6일 15 : 22
192.168.122.229/
192.168.122.229
Customer Portal
Red Hat
Red Hat Products Doc...
Red Hat Enterprise Lin...
hello, world!! I'm vm
시계
00:00:00.0
시작
경과
세계시계(W)
알림(R)
스톱워치(S)
타이머(T)
192.168.150.33:8080/
192.168.150.33:8080
Customer Portal
Red Hat
Red Hat Products Doc...
Red Hat Enterprise Lin...
hello, world!! I'm container
가상 머신 관리자
파일(F)
편집(E)
보기(V)
도움말(H)
열기
이름
CPU 사용량
QEMU/KVM
httpd
종료
[root@vm-container ~]# v
root@vm-container:~
[root@vm-container ~]# podman run -it --rm --name httpd --cpus=3 -m=8192m -p 8080:80 -v ~/httpd:/usr/local/apache2/htdocs/ docker.io/library/httpd:latest

Container 에는 Guest OS 가 없다.

- 컨테이너를 이용한 다양한 리눅스 버전에서 애플리케이션 운영
- Container는 Host의 kernel을 공유하여 사용



OPENMARU © 2021 | All Rights Reserved.


t@ubi9-minimal /]# ./
x_verison_check.sh
[root@centos7 /]# ./linux
_verison_check.sh
root@ubuntu:20:/# ./linux
_verison_check.sh
root@ubuntu:23:/# ./linu
x_verison_check.sh
t@rockylinux:8 /]# ./l
_verison_check.sh
[root@rockylinux:9 /]# ./
linux_verison_check.sh
/ # ./linux_verison_check
.sh
[root@HostOS:~]# ./linux_v
erison_check.sh

컨테이너 환경 온-나라 애플리케이션의 Base OS 변경 방법

openmaru

기존의 CentOS 7.9를 OS로 사용하고 있는 온-나라 애플리케이션을 Rocky, RHEL OS로 변경

- Containerfile(Dockerfile)의 FROM 절을 수정하여 Base OS 변경
- CentOS와 Rocky는 모두 RHEL의 Copy 버전으로
- RHEL 계열이 아닌 다른 OS도 손쉽게 수정 가능

```
FROM docker.io/library/centos:centos7
#FROM docker.io/library/rockylinux:8.9

WORKDIR /opt

# Install JAVA
RUN mkdir jdk-8
ADD contrib/jdk-8u241-linux-x64.tar.gz jdk-8/
ENV JAVA_HOME=/opt/jdk-8/jdk1.8.0_241

# Install JBoss EAP
ADD contrib/jboss-eap-7.4.0.zip ./
RUN yum install -y unzip && \
    unzip jboss-eap-7.4.0.zip && \
    rm -rf jboss-eap-7.4.0.zip

# Application
COPY contrib/standalone.xml /opt/jboss-eap-7.4/standalone/configuration/st
COPY contrib/eGovCloudOnnara.war /opt/jboss-eap-7.4/standalone/deployments
COPY contrib/JDBC-10.2-latest-cubrid.jar /opt/jboss-eap-7.4/standalone/dep

ENTRYPOINT ["/opt/jboss-eap-7.4/bin/standalone.sh", "--server-config", "st
```

< Containerfile-CentOS >

CentOS → Rocky

FROM의 Base Image 변경

```
#FROM docker.io/library/centos:centos7
FROM docker.io/library/rockylinux:8.9

WORKDIR /opt

# Install JAVA
RUN mkdir jdk-8
ADD contrib/jdk-8u241-linux-x64.tar.gz jdk-8/
ENV JAVA_HOME=/opt/jdk-8/jdk1.8.0_241

# Install JBoss EAP
ADD contrib/jboss-eap-7.4.0.zip ./
RUN yum install -y unzip && \
    unzip jboss-eap-7.4.0.zip && \
    rm -rf jboss-eap-7.4.0.zip

# Application
COPY contrib/standalone.xml /opt/jboss-eap-7.4/standalone/configuration/
COPY contrib/eGovCloudOnnara.war /opt/jboss-eap-7.4/standalone/deploymer
COPY contrib/JDBC-10.2-latest-cubrid.jar /opt/jboss-eap-7.4/standalone/c

ENTRYPOINT ["/opt/jboss-eap-7.4/bin/standalone.sh", "--server-config", "
```

< Containerfile-Rockylinux >

OPENMARU © 2021 | All Rights Reserved.

컨테이너 환경에서 더이상 OS 의미가 없음.

openmaru

CentOS 7.9
Containerfile

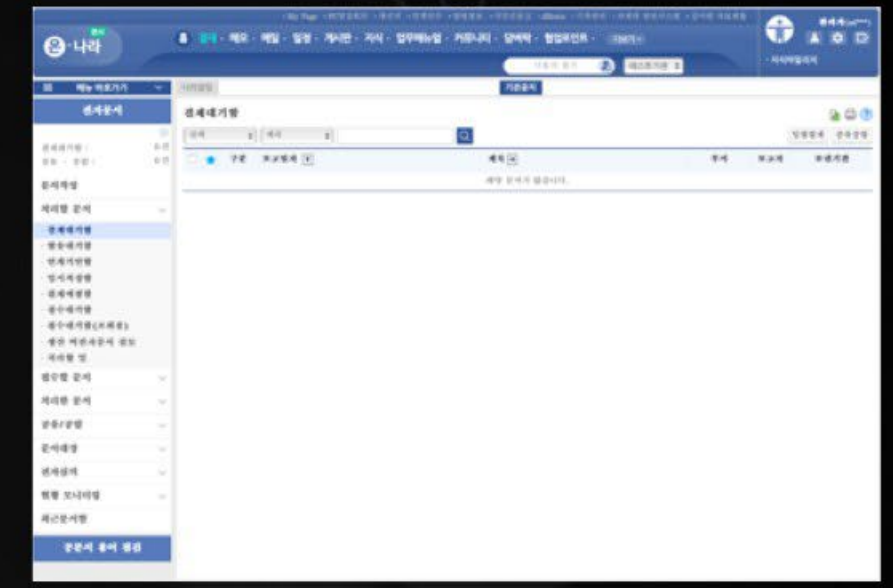
```
FROM centos:centos7
#FROM rockylinux:8.9
```

Build

CentOS 7.9
Container Image

```
STEP 1/11: FROM docker.io/library/centos:centos7
STEP 2/11: WORKDIR /opt
STEP 3/11: RUN mkdir jdk-8
STEP 4/11: ADD contrib/jdk-8u241-linux-x64.tar.gz jdk-8/
STEP 10/11: COPY contrib/JDBC-10.2-latest-cubrid.jar /opt/jboss-eap-7.4/bin/standalone.sh", "--server-config", "standalone.xml"]
COMMIT onnara:centos-7.9
```

Run



! OS 변경을 위한 Containerfile 손쉽게 작성

! OS 및 Version 변경에도 정상 Build

Rocky 8.9
Containerfile

```
#FROM centos:centos7
FROM rockylinux:8.9
```

Build

Rocky 8.9
Container Image

```
STEP 1/11: FROM docker.io/library/rockylinux:8.9
STEP 2/11: WORKDIR /opt
STEP 3/11: RUN mkdir jdk-8
STEP 4/11: ADD contrib/jdk-8u241-linux-x64.tar.gz jdk-8/
STEP 10/11: COPY contrib/JDBC-10.2-latest-cubrid.jar /opt/jboss-eap-
STEP 11/11: ENTRYPOINT ["/opt/jboss-eap-7.4/bin/standalone.sh", "--s
COMMIT onnara:rocky-8.9
```

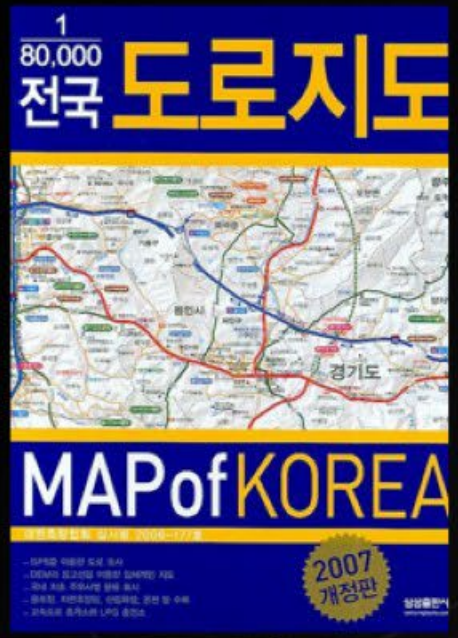
Run

! 컨테이너 환경에서는 Base OS에 영향을 받지 않고
일관적인 애플리케이션 배포

OPENMARU © 2021 | All Rights Reserved.

Application Performance Management
그땐 꼭 필요했지만 사라진 것들

우리 일상생활에서 한때는 꼭 필요했지만 지금은 사라진 것들



- Confidential -
OPENMARU © 2021 | All Rights Reserved.

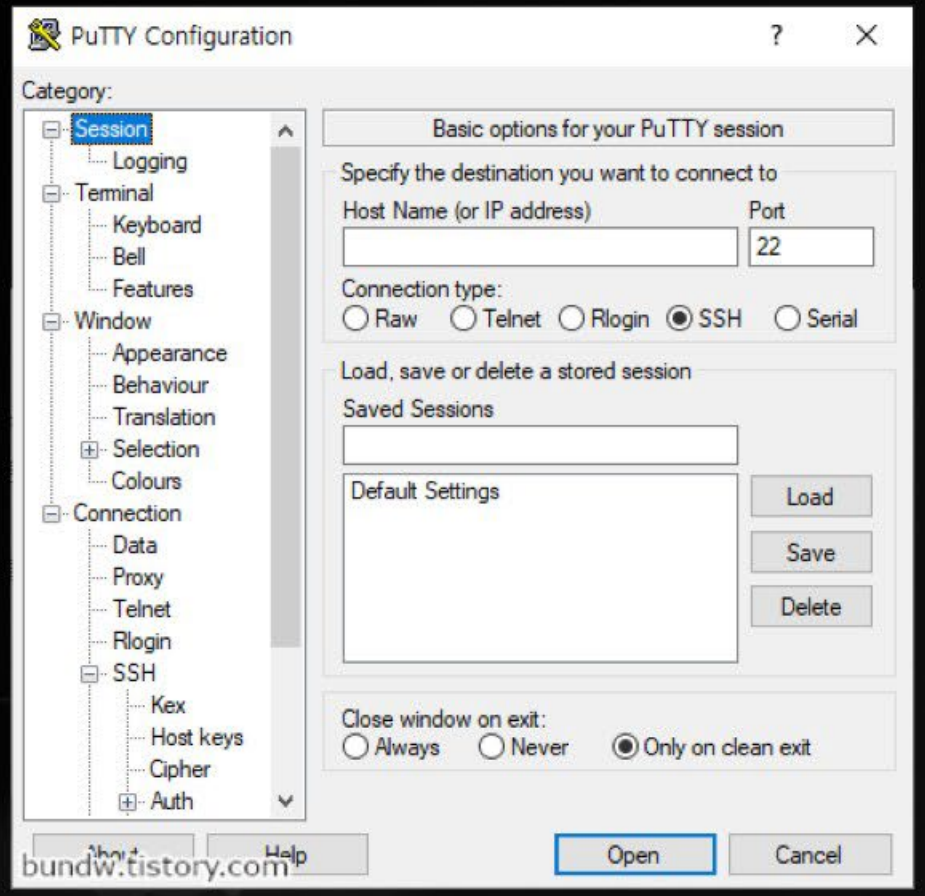
Paradigm Shift

- 혁신 이후 이전으로 돌아 갈 수는 없다.
- 너무 많은 것이 변하기 때문에 개념을 이해하기 어렵다. 직접 사용해 보면 안다.
- 클라우드 네이티브는 스마트폰 혁명 수준

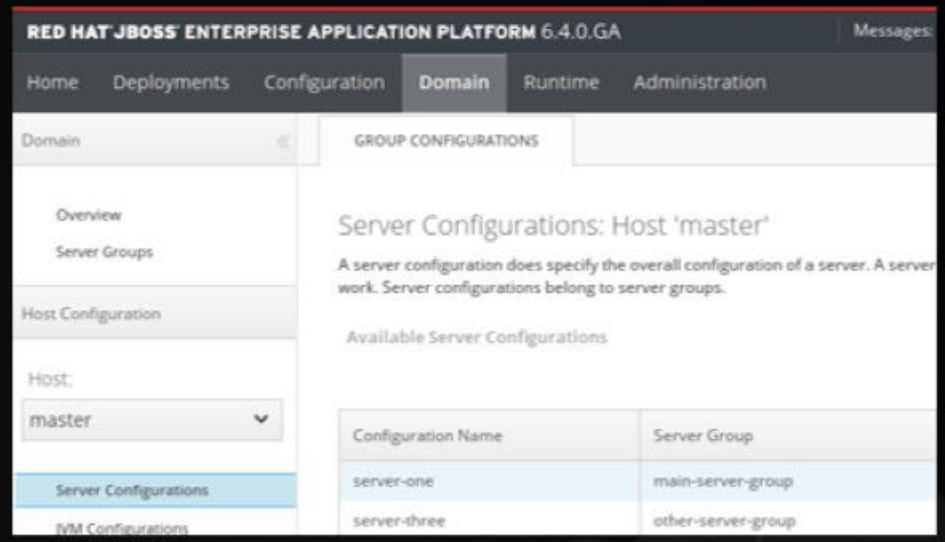
BEFORE

AFTER

클라우드 네이티브 시대 없어 질 것들 (1)



Terminal 접속

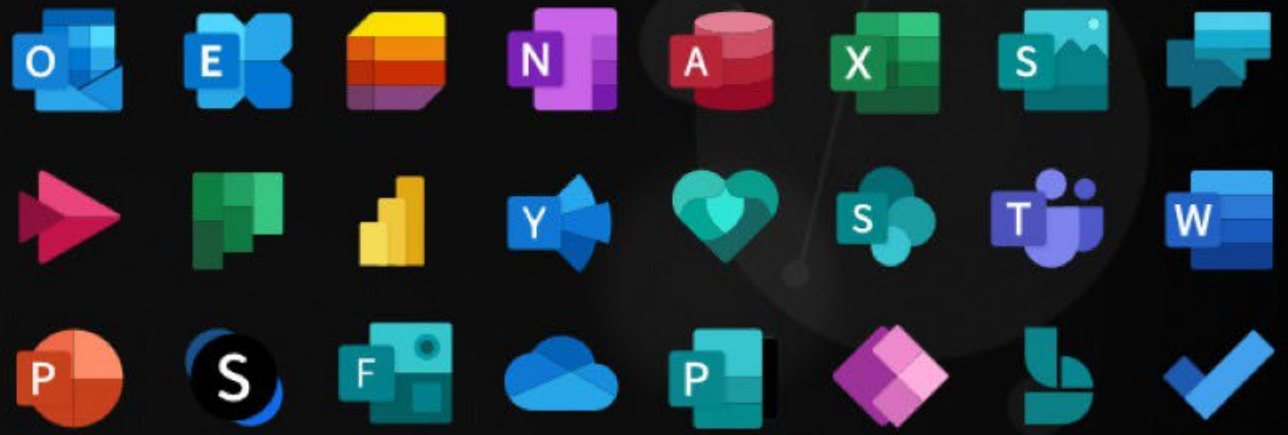


WAS Admin Console

- Confidential
OPENMARU © 2021 | All Rights Reserved.

OS 가 정말 필요한가요?

- 내가 필요한 애플리케이션만 실행



- Confidential -

OPENMARU © 2021 | All Rights Reserved.

Cloud 에서 OS 가 필요할까요?

- 클라우드 네이티브는 OS 마저도 가상화하여 OS 없이 운영 가능한 환경 제공
- 컨네이너에서는 OS 없이 애플리케이션 실행환경 구성이 가능
- 컨테이너로 담기면 프라이빗 클라우드 와 퍼블릭 클라우드 어디서든 운영 가능

Red Hat
Ubuntu
AlmaLinux
fedora
Rocky Linux
ORACLE Linux
CentOS
slackware linux
SUSE
archlinux

- Confidential -

IT 도 자율 운영 할 수 있다.

openmaru

Tesla Model Y

BMW 5 Series

Genesis concept car

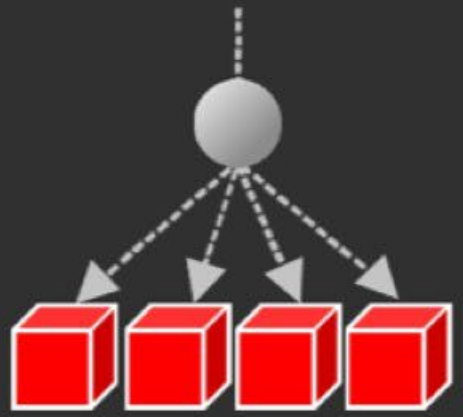
- Confidential -

OPENMARU © 2021 | All Rights Reserved.

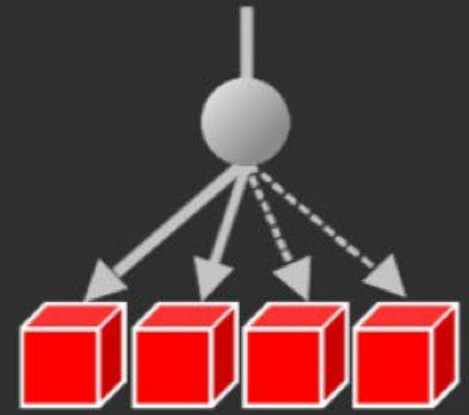
openmaru

Auto Scale Out / In

Load Balancer

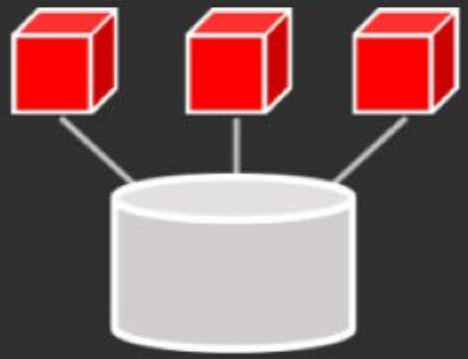


Auto Rolling Update

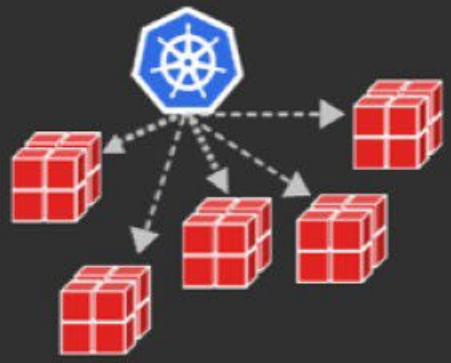


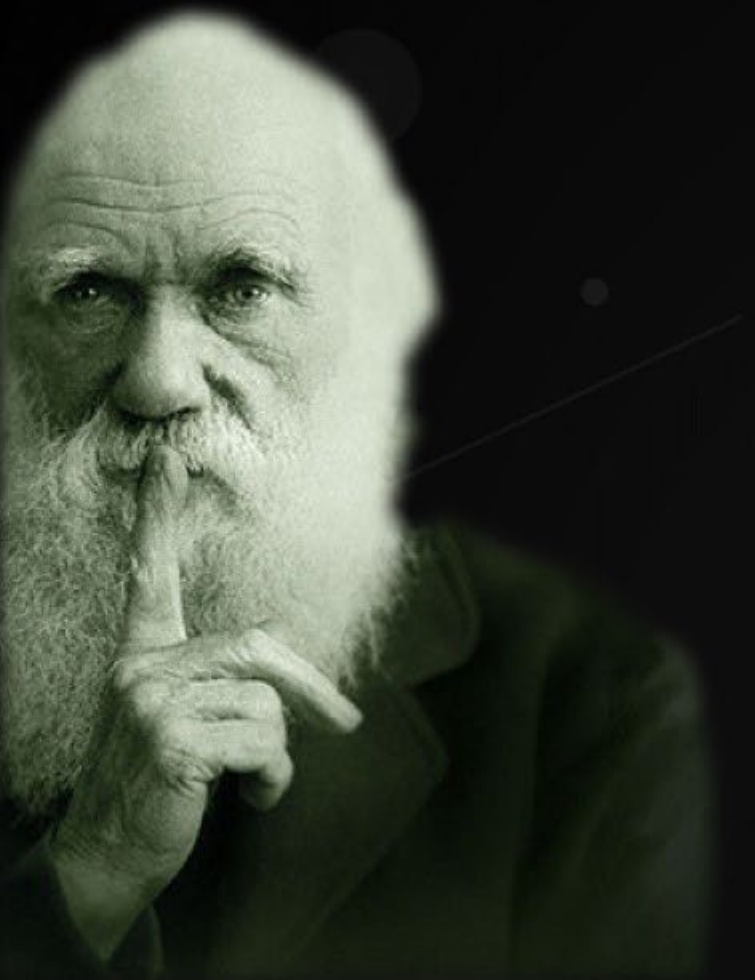
Auto Healing

Persistence Volume



Container Orchestration





openmaru

"살아 남는 종(種)은 강한 종이 아니고,
또 우수한 종도 아니다.
변화에 적응하는 종이다. "

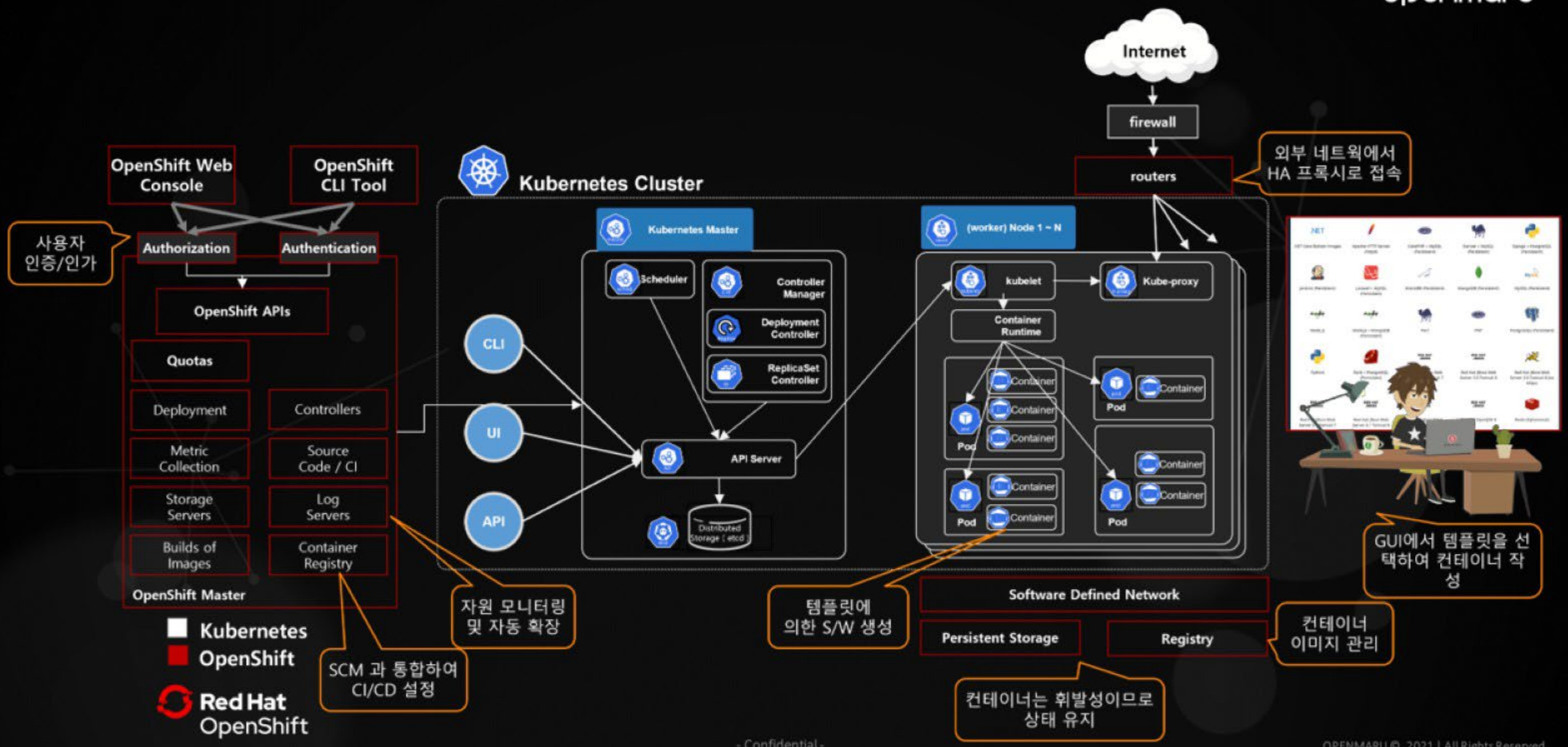
- Charles Darwin, 1809

- Confidential -

OPENMARU © 2021 | All Rights Reserved.

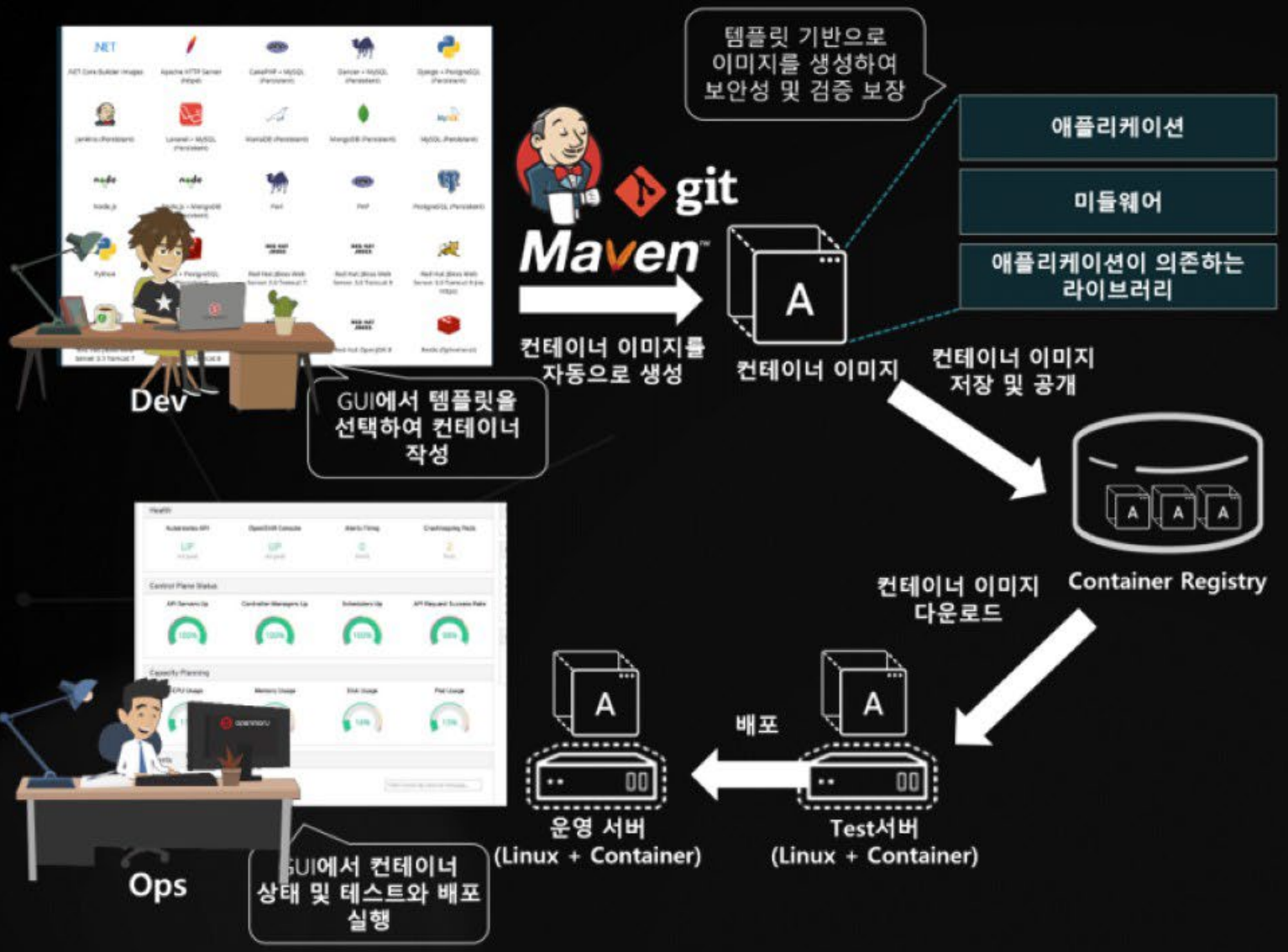
Cloud Native 기술을 통한 Open Hybrid Cloud 구현
클라우드 네이티브 기대효과
openmaru
APM

Kubernetes 에 추가된 PaaS 기능 들



OPENMARU © 2021 | All Rights Reserved.

PaaS 와 Kubernetes 기능 측면 비교



카테고리	Kubernetes	OpenShift
기업용 리눅스	구매	포함
기업용 미들웨어	구매	포함
이미지 레지스트리	별도	포함
스토리지 인터페이스	별도	포함
SDN	별도	포함
통합로그 관리	별도	포함
멀티 테넌시	별도	포함
미터링	별도	포함
CI/CD	별도	포함
인증된 DB,M/W등	없음	포함
무정지 업그레이드	불가	지원
24X7 기술지원	자체	지원
보안	자체	지원

OPENMARU © 2021 | All Rights Reserved.

단독 업무용 시스템 구조 vs. 복합 업무용 시스템 구조

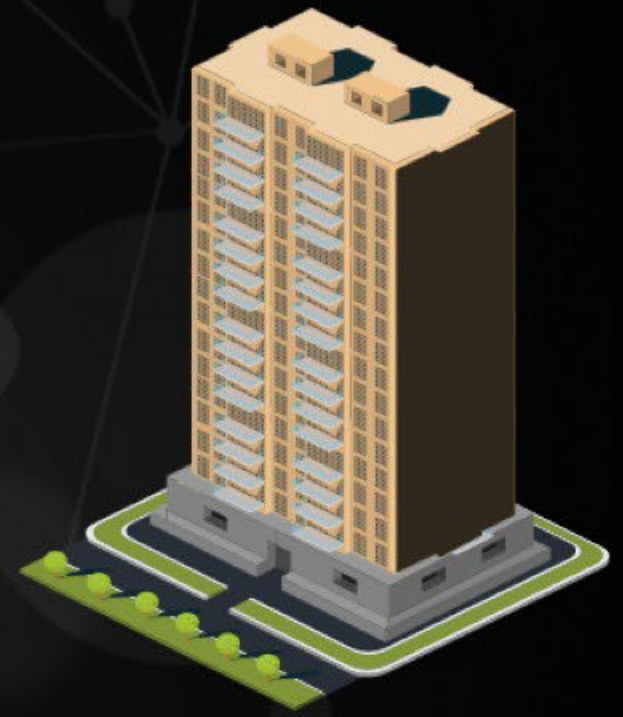
- Cloud Immigrant 위치와 형태만 변경되었을 뿐 기존 방식 그대로 월세로만 변경되었을 뿐
- Cloud Native 로 가면 계단이 아닌 엘리베이터 타는 방법은 알아야 한다.
  - 계단이 없어서 어떻게 올라 가냐고 많이 물어봄
  - 자동화와 관리 최소화로 독립된 공간 확보

Single Tenancy

Cloud Immigrant

Single Tenancy

Cloud Native



Multi Tenancy

OPENMARU © 2021 | All Rights Reserved.

Private Cloud 는 복합 업무용 플랫폼을 구축하는 것

- 클라우드로 전환할 수록 관리와 비용은 낮추어야 하고, 집적도는 높아져야 함.
- 클라우드 네이티브에서는 OS 없이 시스템을 컨테이너로 쌓아서 구성

Single Tenancy

물리/가상 서버	구분	클라우드 네이티브
직접 구축 관리	인프라	공동 관리
낮음	집적도	매우 높음
모든 걸 직접 관리	관리	자동화된 관리
단독 업무용 서버 구조	토지 (OS)	복합 업무용 서버 구조
IaaS	서비스모델	PaaS

Multi Tenancy

OPENMARU © 2021 | All Rights Reserved.

수작업으로 관리하지 않는 IT 자율 운영으로 전환

- 클라우드 네이티브에서는 관리 대상이 **WAS에서 컨테이너로 변경**
  - 컨테이너 단위로 WAS에 대한 확장/축소, 장애 복구 , 업그레이드, 패치 작업 수행
  - 더 이상 WAS 에 대한 관리 기능이 필요 없음, 컨테이너는 **휘발성으로 상태를 가지고 있지 않음**

기존 WAS는 단독 주택

새롭게 추가된 WAS 요건

- WAS는 물론 컨테이너 모니터링
- 자동장애복구/자동확장축소 모니터링 정보 제공
- 폐기된 컨테이너 정보 제공
- OpenShift 관리 정보와의 통합

컨테이너용 WAS 는 아파트

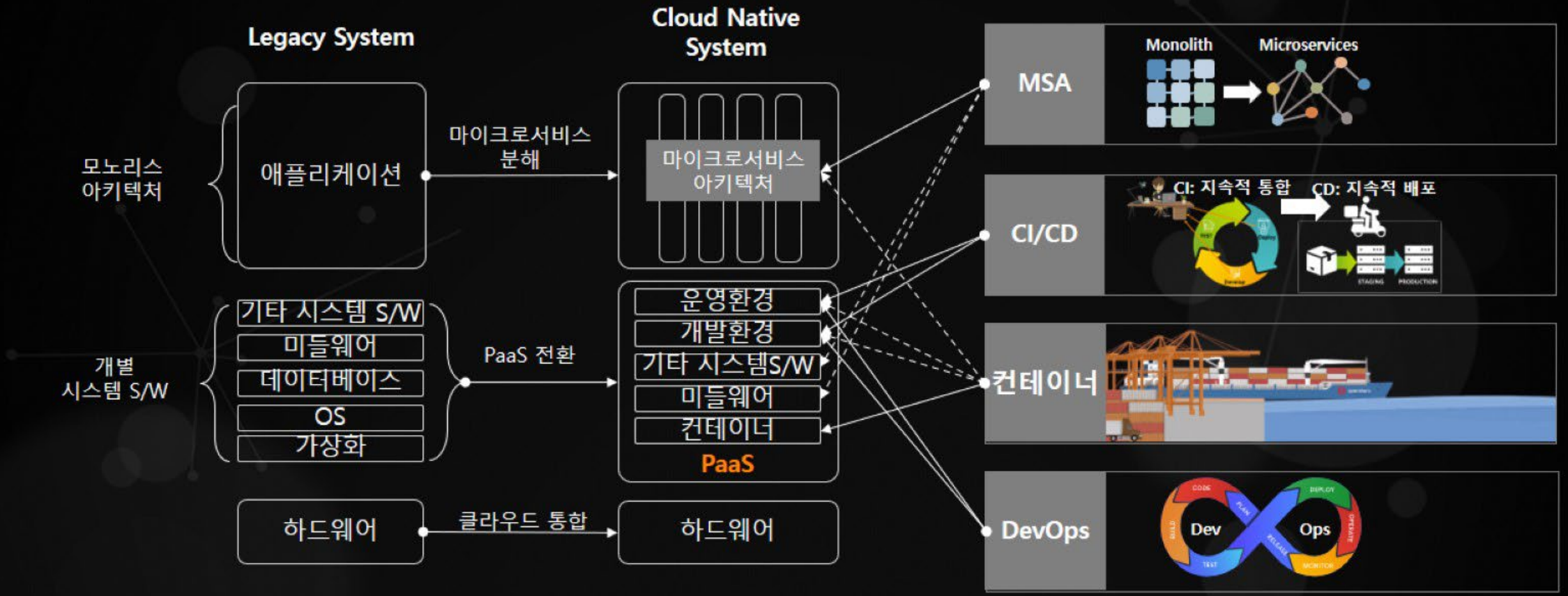
운영 요구 사항	사람의 수작업에 의한 운영		
WAS증설과 축소	운영자	수작업	직접확인
WAS 장애	운영자	수작업	직접확인
애플케이션 배포	운영자	수작업	직접확인
WAS 운영관리	운영자	수작업	직접확인
WAS 업그레이드	운영자	수작업	직접확인

클라우드에서 규칙의 의한 운영 자동화		
클라우드	부하에 따른	자동 확장/축소
클라우드	자동감지	자동 장애복구
클라우드	무정지 서비스	자동 배포
클라우드	컨테이너로	WAS 관리
클라우드	이미지에 의한	WAS 업그레이드

- Confidential -
OPENMARU © 2021 | All Rights Reserved.

IT 운영환경도 클라우드 네이티브 기반 PaaS로 전환

- 클라우드 네이티브는 "클라우드가 '클라우드 다울 수 있도록' 애플리케이션을 구축, 실행하는 방식"



OPENMARU © 2021 | All Rights Reserved.

컨테이너 자동화 도구를 이용한 프로세스

- 파이프라인 기반의 자동화 환경을 이용하여 신속한 개발 및 운영 환경 구축
- 필요에 따라 구축과 삭제가 편리

DevOps 환경

개발 환경 구축
(Service Catalog)

자동 확장/ 자동 복구
(Auto Scaling/ Healing)

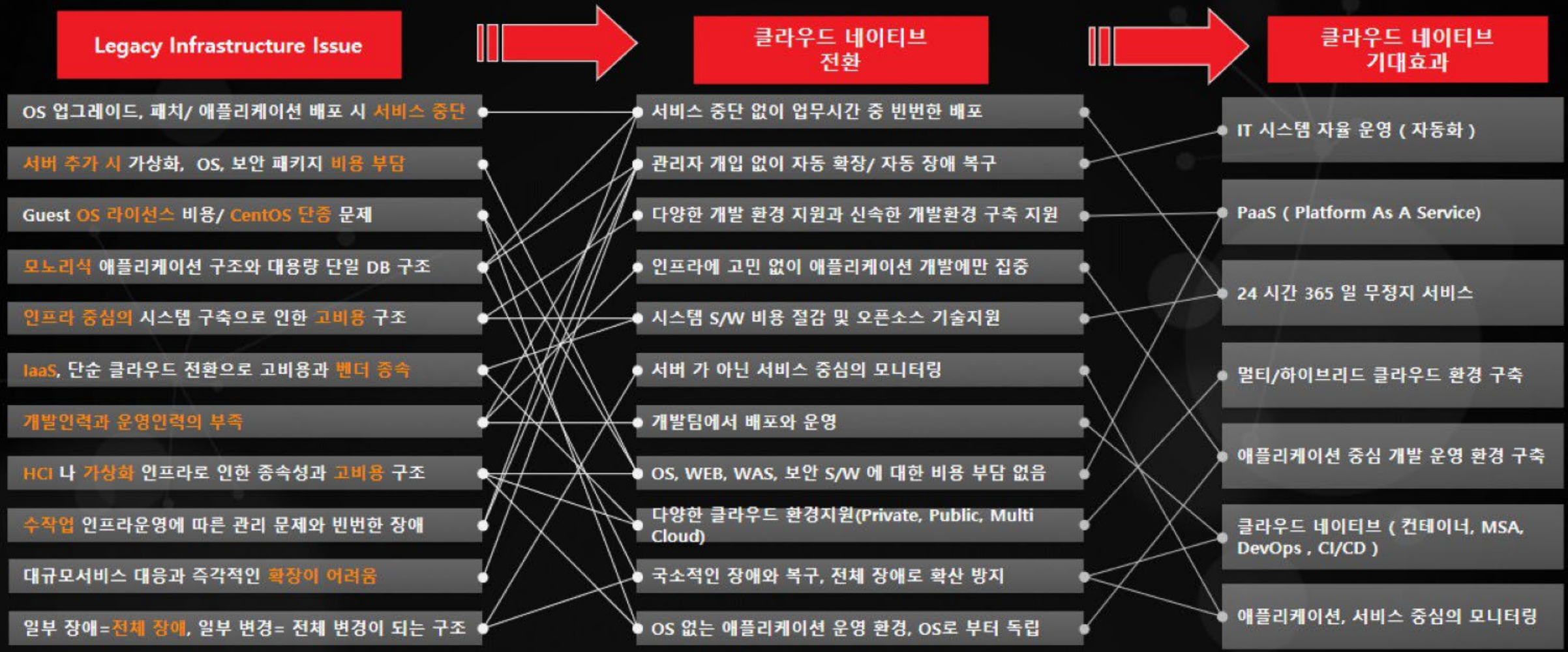
CI: 지속적인 통합
(Continuous Integration)

CD: 지속적인 배포
(Continuous Delivery)

OPENMARU © 2021 | All Rights Reserved.

클라우드 네이티브 플랫폼 구축 기대 효과

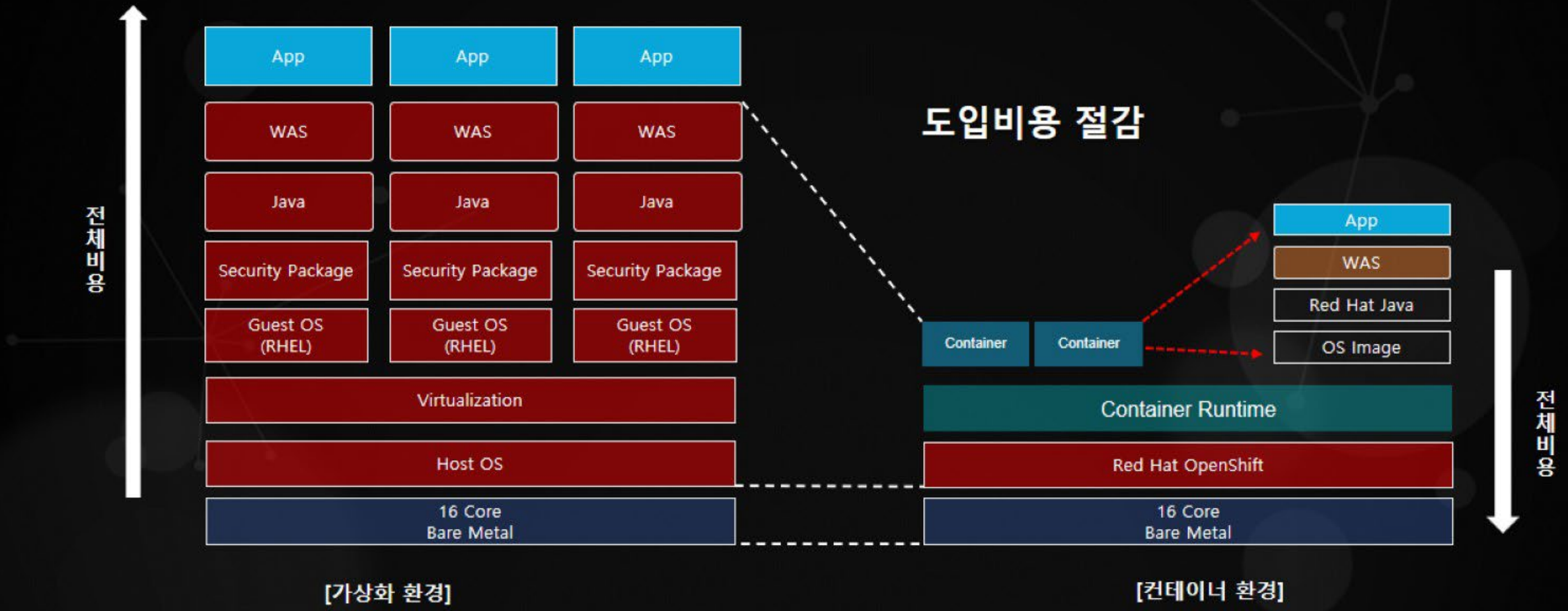
- 클라우드 네이티브(PaaS)를 통한 서비스 무정지 , MSA, DevOps , CI/CD 등 경쟁력 있는 개발/운영 체계 확립



- Confidential -
OPENMARU © 2021 | All Rights Reserved.

가상화 VS 컨테이너 비교 – 비용적인 측면

- 가상화 대비 Guest OS 유지보수, 라이선스, 관리비용 제거
- 서버 접근제어를 비롯한 보안 솔루션 제거



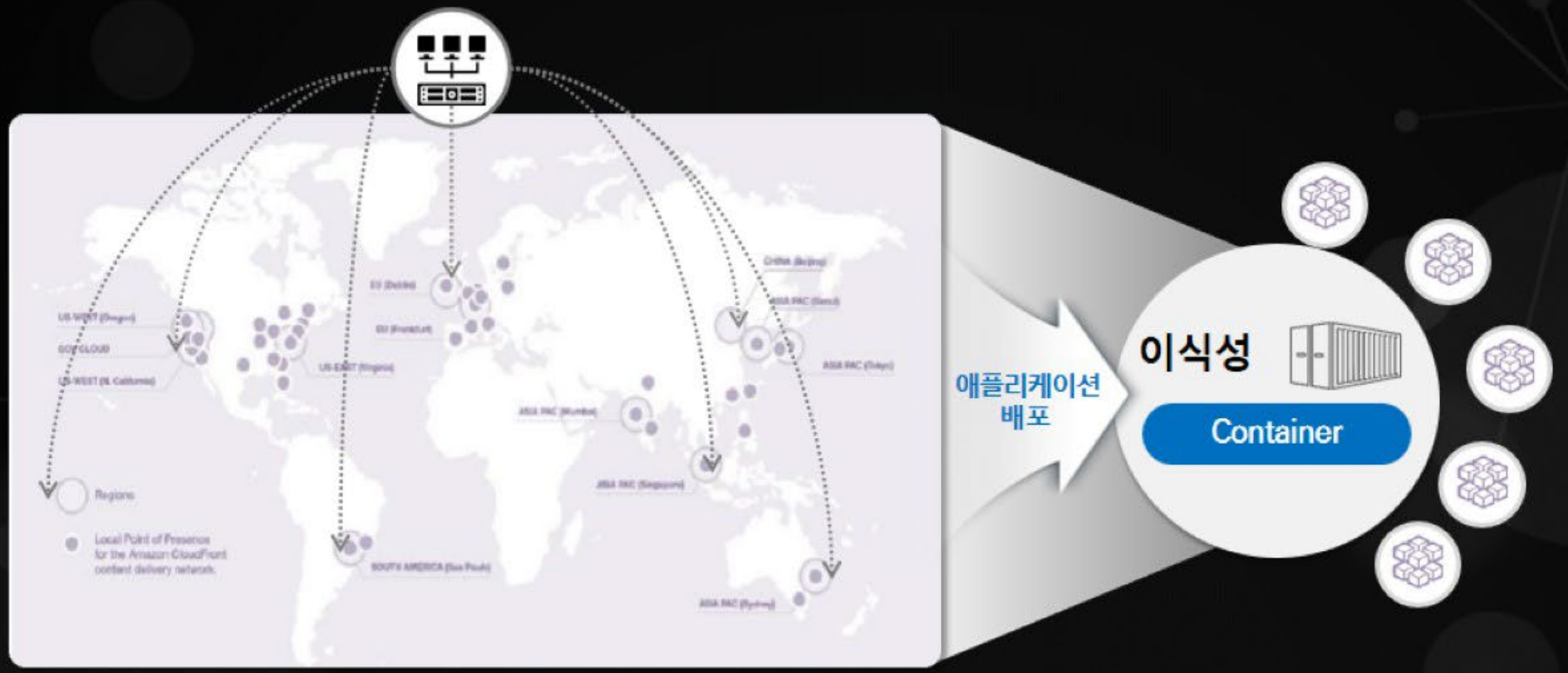
OPENMARU © 2021 | All Rights Reserved.

Cloud Native 기술을 통한 Open Hybrid Cloud 구현

데모로 이해하는 클라우드 네이티브
- 하이브리드 클라우드 데모

클라우드 네이티브 특징

- 클라우드 네이티브는 작고, 가볍고 손쉽게 배포
- 클라우드 환경에서의 →서비스 배포는 전세계에 한번의 클릭으로 애플리케이션 배포 →전세계를 상대로 서비스 가능

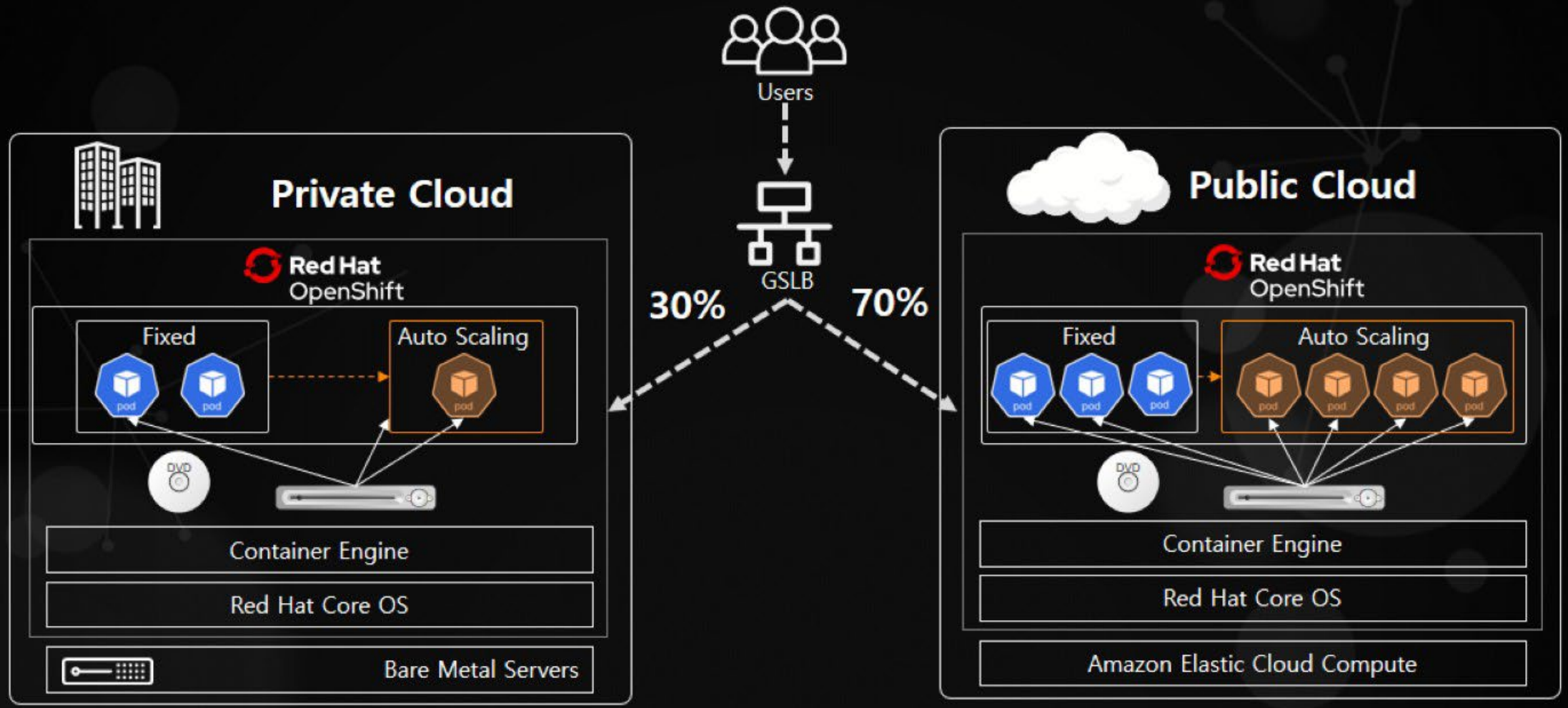


Source : 클라우드 네이티브 추진 시 고려사항 (교육 교재)
클라우드 네이티브 기반 행정·공공 서비스 확산 지원 - 한국지능정보사회진흥원

OPENMARU © 2021 | All Rights Reserved.

하이브리드 클라우드 데모 – 사용자 증가 자동 확장

- 하이브리드 클라우드로 운영되는 포탈을 접속하는데 자동확장이 필요한 사용자 폭주상태를 가정
- 내부 클라우드과 외부 클라우드에 모두 동일한 홈페이지 서비스를 하고 프라이빗 30% vs. 퍼블릿 70% 로 운영 중



OPENMARU © 2021 | All Rights Reserved.

하이브리드 클라우드 데모 – 전자정부 F/W 포탈에 대한 글로벌 서버 부하 분산

- **GSLB (Global Server Load Balancing)을 통한 업무 부하 분산 데모**
- **전자정부 F/W 포탈 서비스를 내부와 외부 클라우드에서 동시 운영**

https://portal.egov.openmaru.io

30%

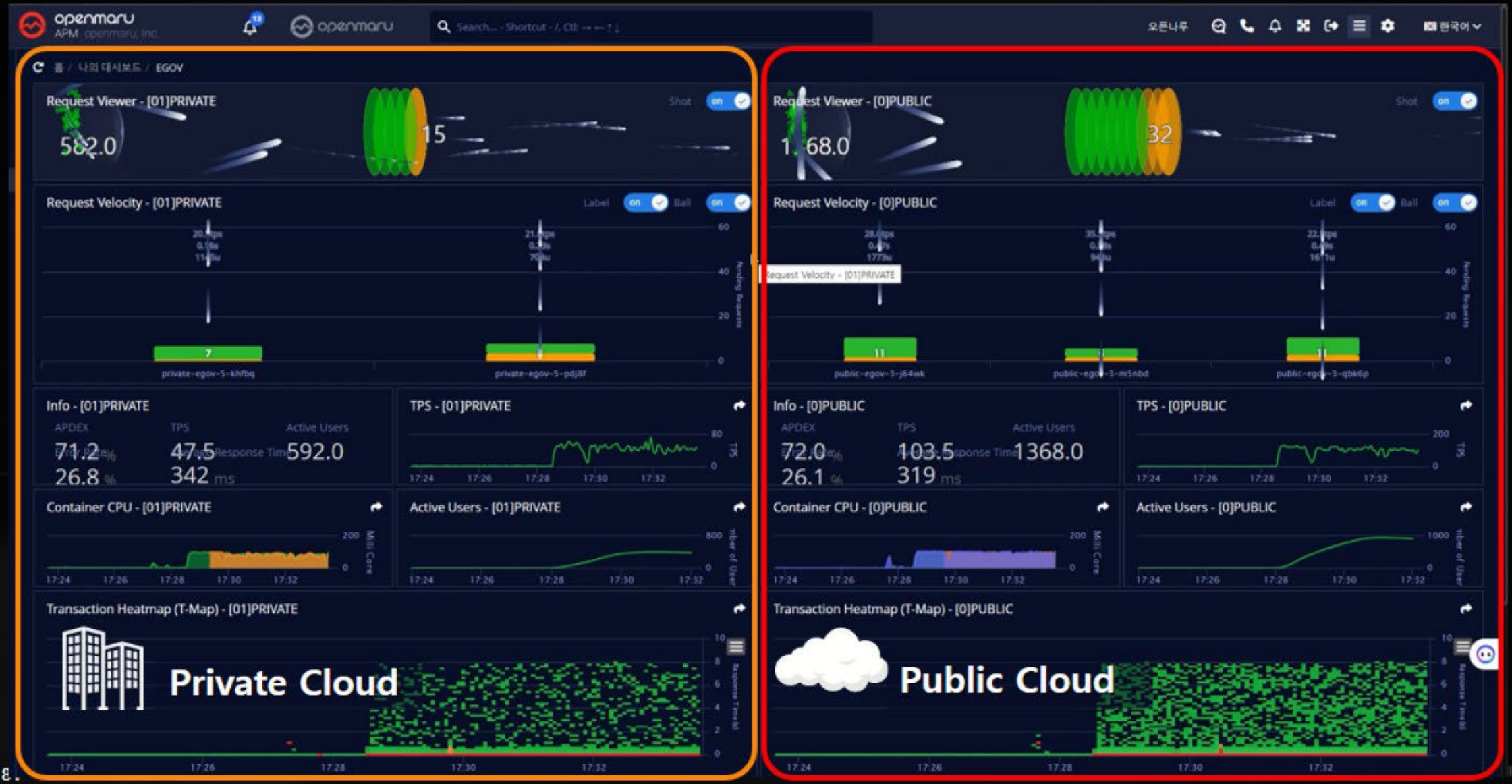
70%

https://**private**.egov.openmaru.io

https://**public**.egov.openmaru.io

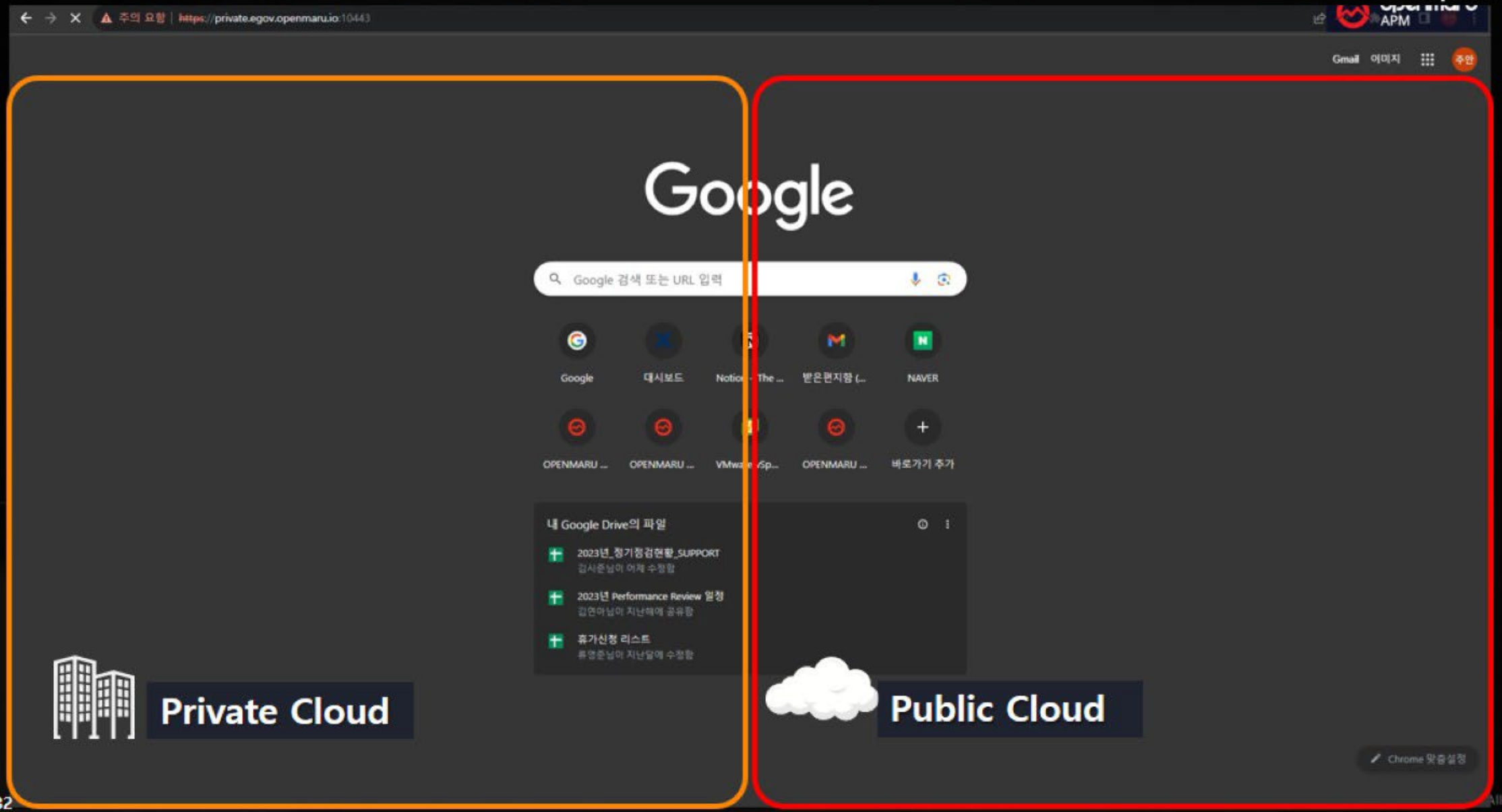
OPENMARU © 2021 | All Rights Reserved.

하이브리드 클라우드 데모 – 자동 확장 이전



Rights Reserved.

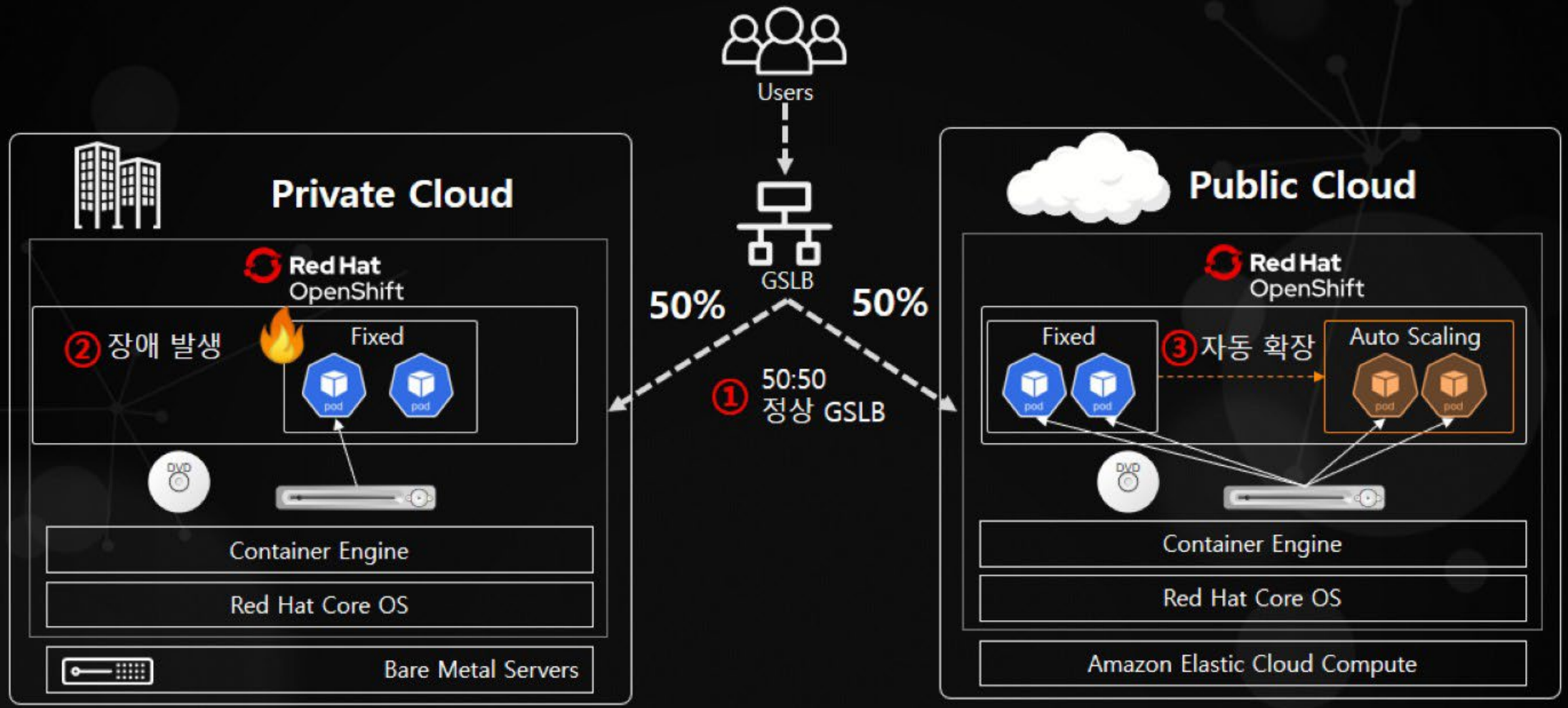
하이브리드 클라우드 데모 – 자동 확장 이후



All Rights Reserved.

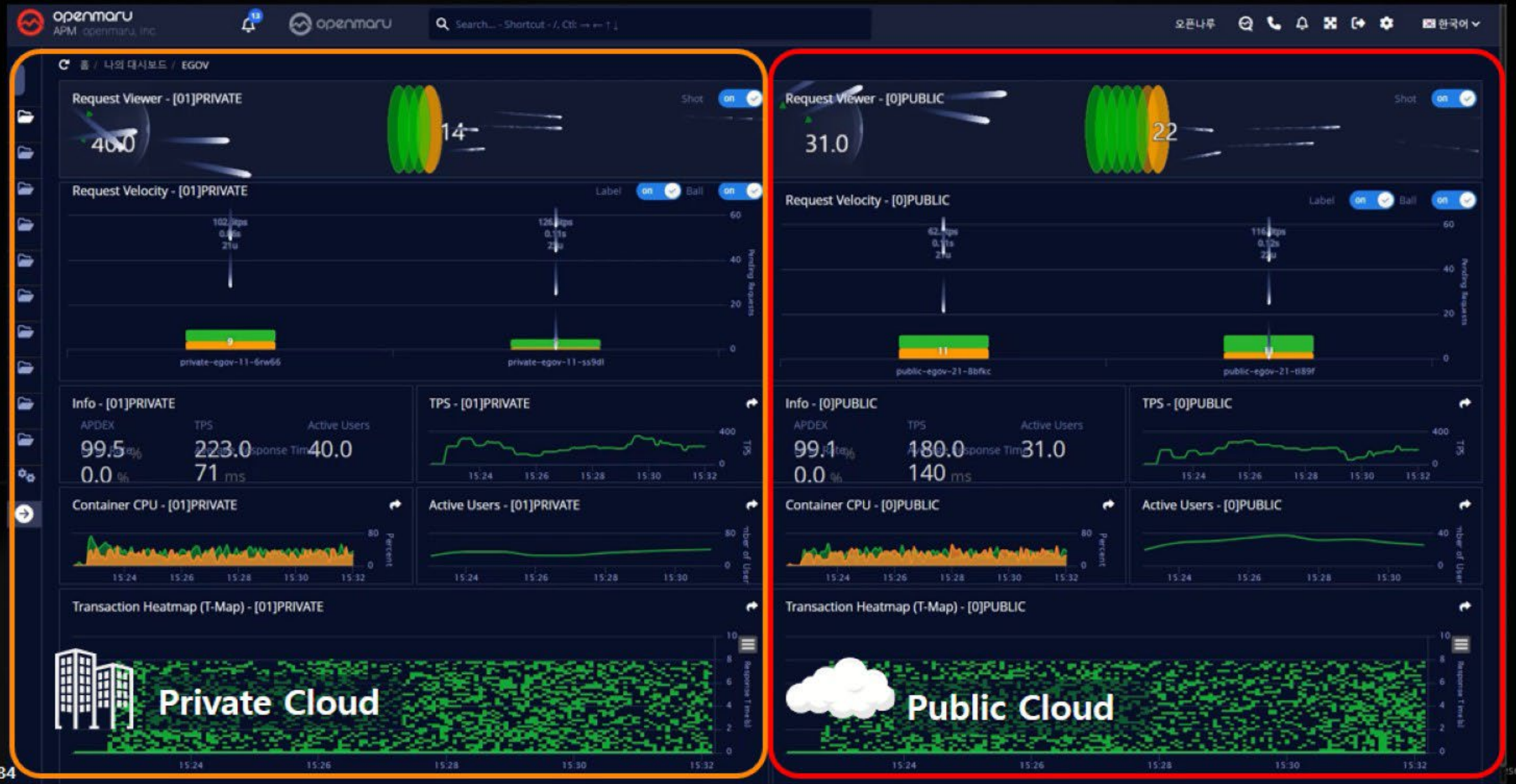
하이브리드 클라우드 데모 – Active Active DR

- 하이브리드 클라우드로 운영되는 포탈을 접속하는데 자동확장이 필요한 사용자 폭주상태를 가정
- 내부 클라우드과 외부 클라우드에 모두 동일한 홈페이지 서비스를 하고 프라이빗 30% vs. 퍼블릿 70% 로 운영 중



OPENMARU © 2021 | All Rights Reserved.

하이브리드 클라우드 데모 – 장애 발생 이전 Active-Active GSLB



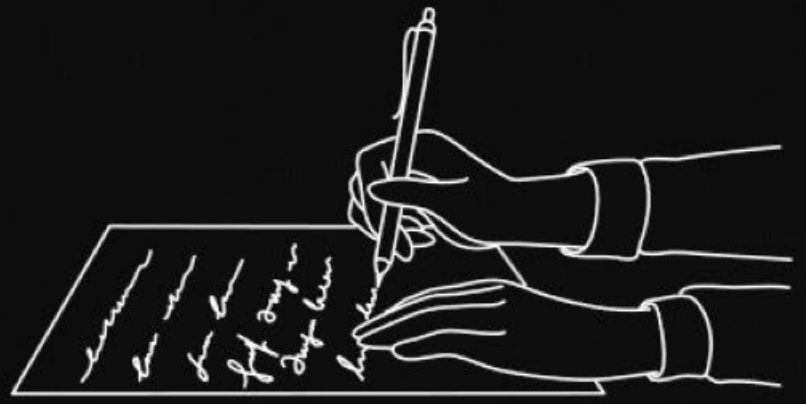
하이브리드 클라우드 데모 – Public Cloud 에서 부하에 따른 자동확장

Cocktail Cloud
APM
Application Centric Infrastructure

Immutable Infrastructure (불변의 인프라스럭처)

- 서버에 수정 또는 변경이 필요하다면 새로운 서버를 기존 이미지를 기반으로 생성하여 이전 서버를 대체
- 클라우드 네이티브 핵심 프로젝트인 Kubernetes 가 이러한 방식의 Immutable Infrastructure 개념으로 개발

아래의 그림의 공통점은 무엇일까요?

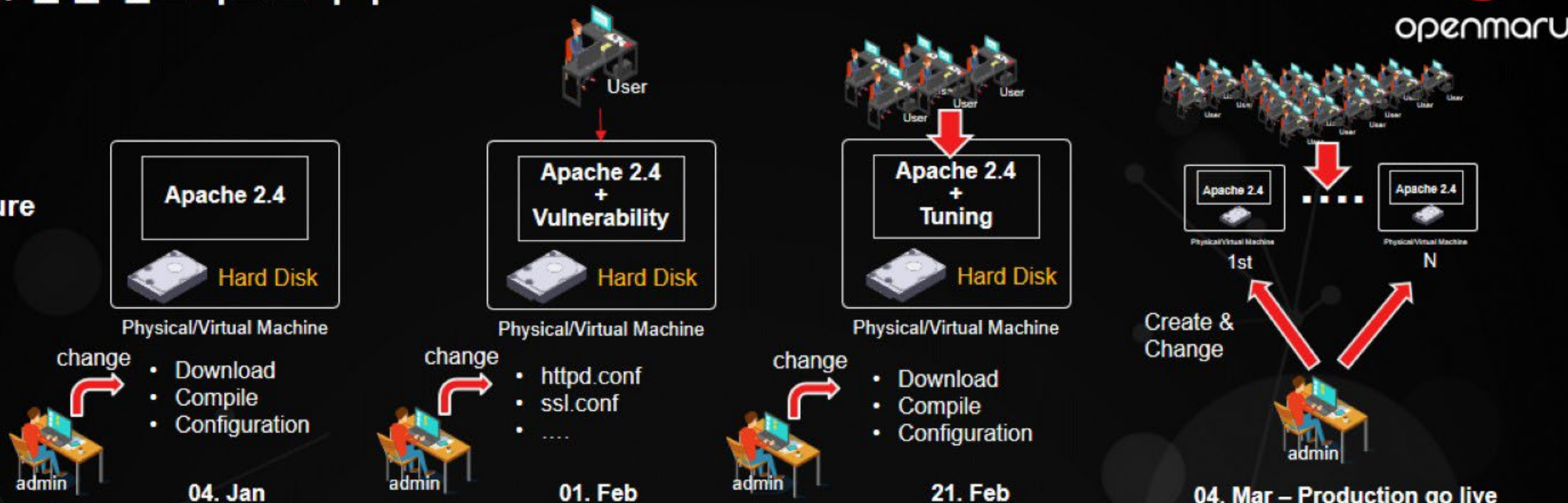


OPENMARU © 2021 | All Rights Reserved.

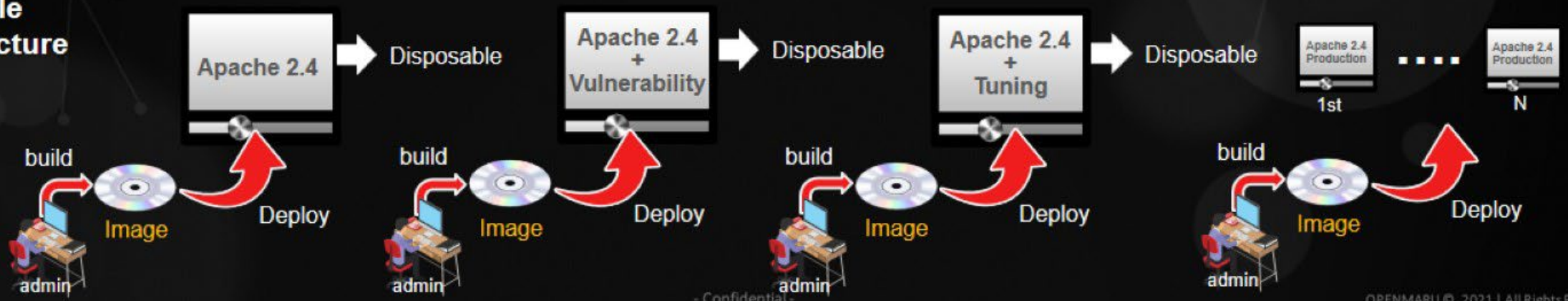
가변 vs. 불변 인프라스트럭처

openmaru

Mutable Infrastructure (In-Place)



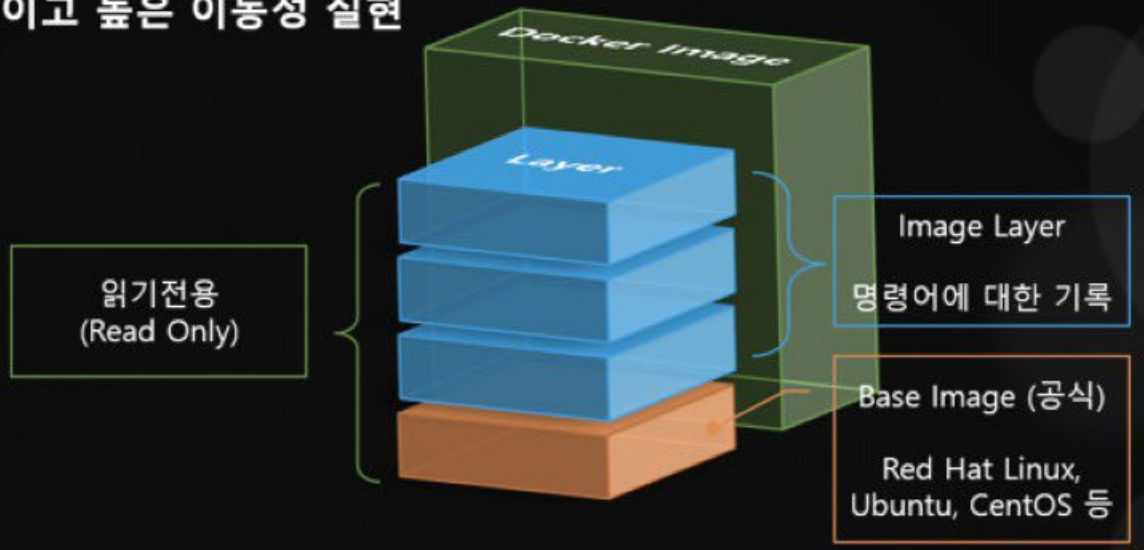
Immutable Infrastructure (Replace)



- Confidential -
OPENMARU © 2021 | All Rights Reserved.

Container Image

- 컨테이너를 실행할 때 필요한 파일시스템
  - 이미지 레이어의 집합체
    - 파일 내용과 메타 정보를 포함
  - 레이어는 부모와 자식 관계
  - 변경분만 기록
  - Read Only (읽기 전용) 으로 쓸 수 없음
- 공통 레이어를 이미지 간에 공유
  - 디스크 용량을 줄이고 높은 이동성 실현



OPENMARU © 2021 | All Rights Reserved.

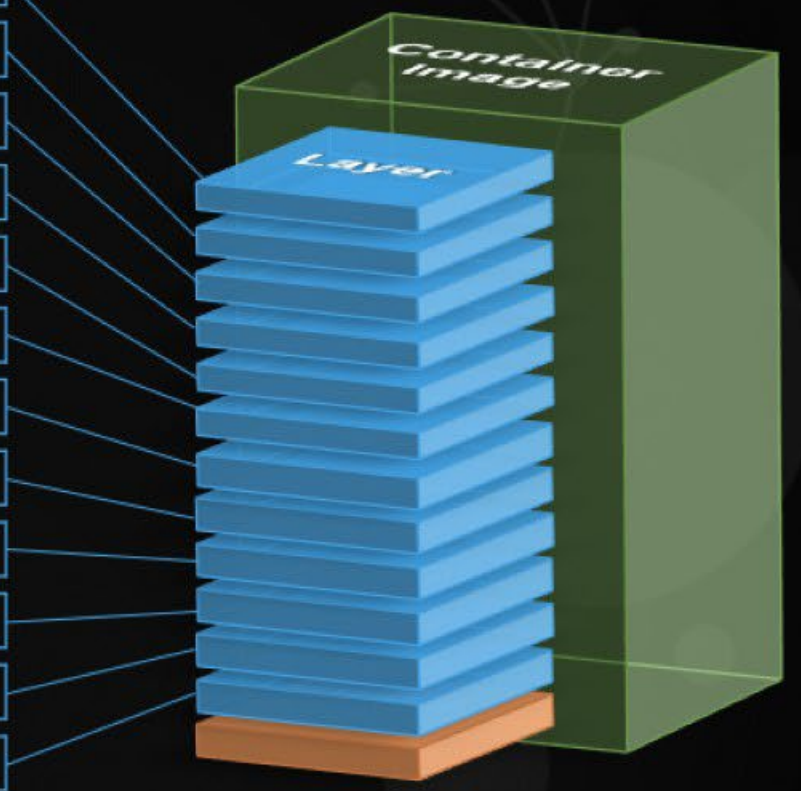
Container 이미지 구조 예시

OPENMARU © 2021 | All Rights Reserved.

Container Image 구조

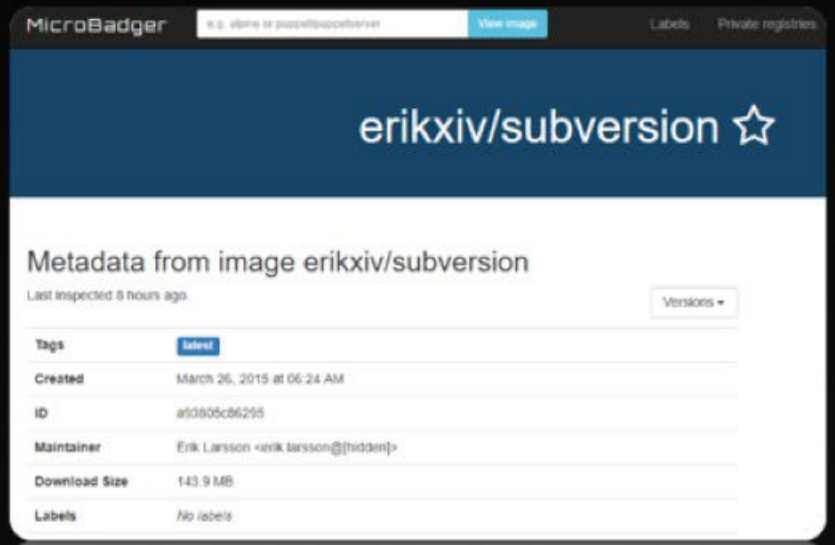
- **docker history <image id / name>**

	CREATED	CREATED BY	SIZE
a486da044a3f	10 weeks ago	/bin/sh -c # (nop) CMD ["nginx" "-g" "daemon o	0 B
3cb7f49c6bc4	10 weeks ago	/bin/sh -c # (nop) EXPOSE 443 /tcp 80 /tcp	0 B
42d2189f6cbe	10 weeks ago	/bin/sh -c # (nop) VOLUME [/var/cache/nginx]	0 B
6dda3f3a8c05	10 weeks ago	/bin/sh -c ln -sf /dev/stderr /var/log/nginx/	11 B
9108e25be489	10 weeks ago	/bin/sh -c ln -sf /dev/stdout /var/log/nginx/	11 B
72b67c8ad0ca	10 weeks ago	/bin/sh -c apt-get update &&	7.695 MB
e7e7a55e9264	10 weeks ago	/bin/sh -c # (nop) ENV NGINX_VERSION=1.9.4-1 ~ j	0 B
97df1ddba09e	3 months ago	/bin/sh -c echo "deb http://nginx.org/package	221 B
5dd2638d10a1	3 months ago	/bin/sh -c apt-key adv --keyserver hkp : // pgp.	1.997 kB
aface2a79f55	3 months ago	/bin/sh -c # (nop) MAINTAINER NGINX Container Mai	0 B
9a61b6b1315e	3 months ago	/bin/sh -c # (nop) CMD ["/bin/bash"]	0 B
902b87aaaec9	3 months ago	/bin/sh -c # (nop) ADD file:e1dd18493a216ecd0c	125.2 MB



OPENMARU © 2021 | All Rights Reserved.

Container Image에 대한 Layer 정보



Source: https://microbadger.com/images/erikxiv/subversion

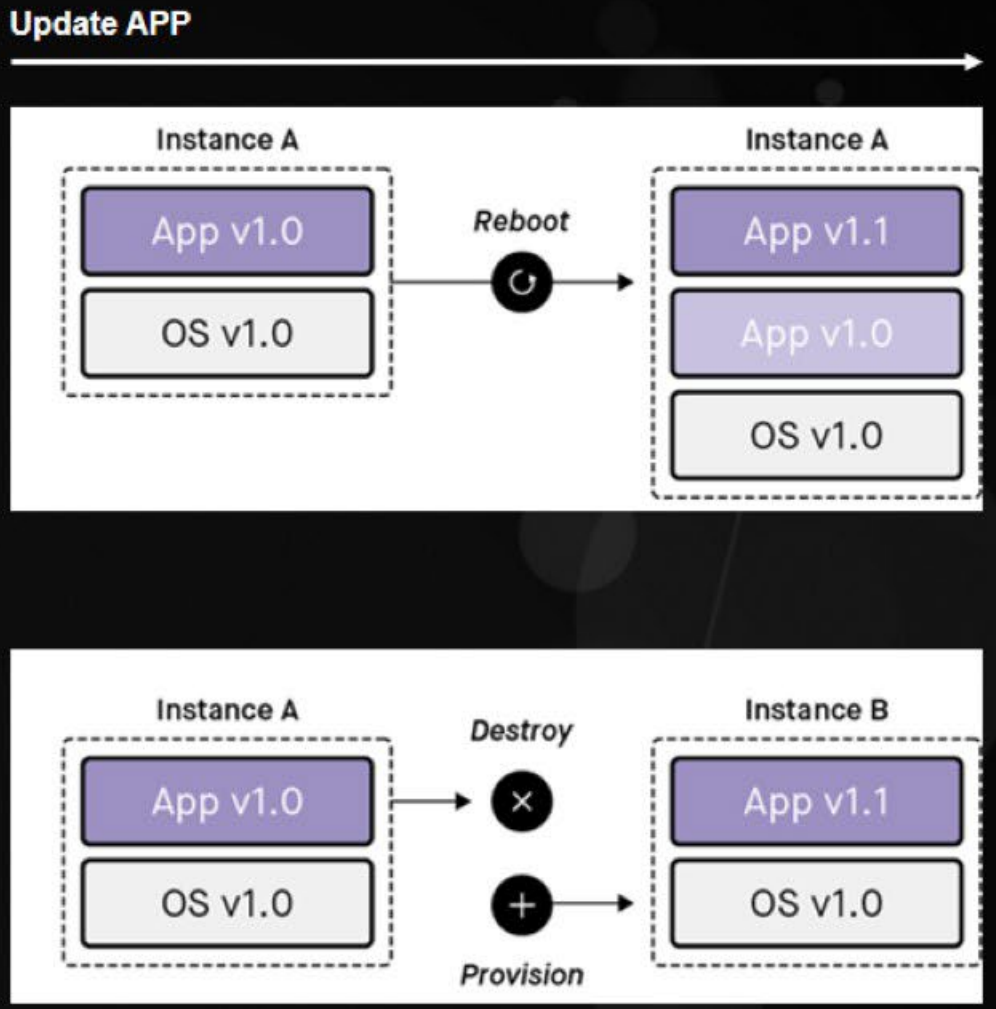
OPENMARU © 2021 | All Rights Reserved.

Mutable vs. Immutable Infrastructure (가변 vs. 불변)

openmaru

Mutable Infrastructure (물리서버, 가상화)

Immutable Infrastructure (컨테이너)



- Confidential -

OPENMARU © 2021 | All Rights Reserved.

머신 중심에서 애플리케이션 중심 인프라로 변화

- 컨테이너화는 데이터 센터를 머신 중심에서 애플리케이션 중심으로 전환
  - 개발자와 운영팀에게 서버와 운영 환경에 대한 세부 사항을 추상화
  - 운영 중인 애플리케이션과 개발자에 미치는 영향을 최소화하면서 새로운 하드웨어 또는 운영 환경을 업그레이드하여 인프라팀에게 유연성을 제공
  - 서버의 CPU와 메모리 정보 뿐만 아니라 애플리케이션과 관련한 매트릭을 연결하여 오토 스케일링

Machine Centric Infrastructure

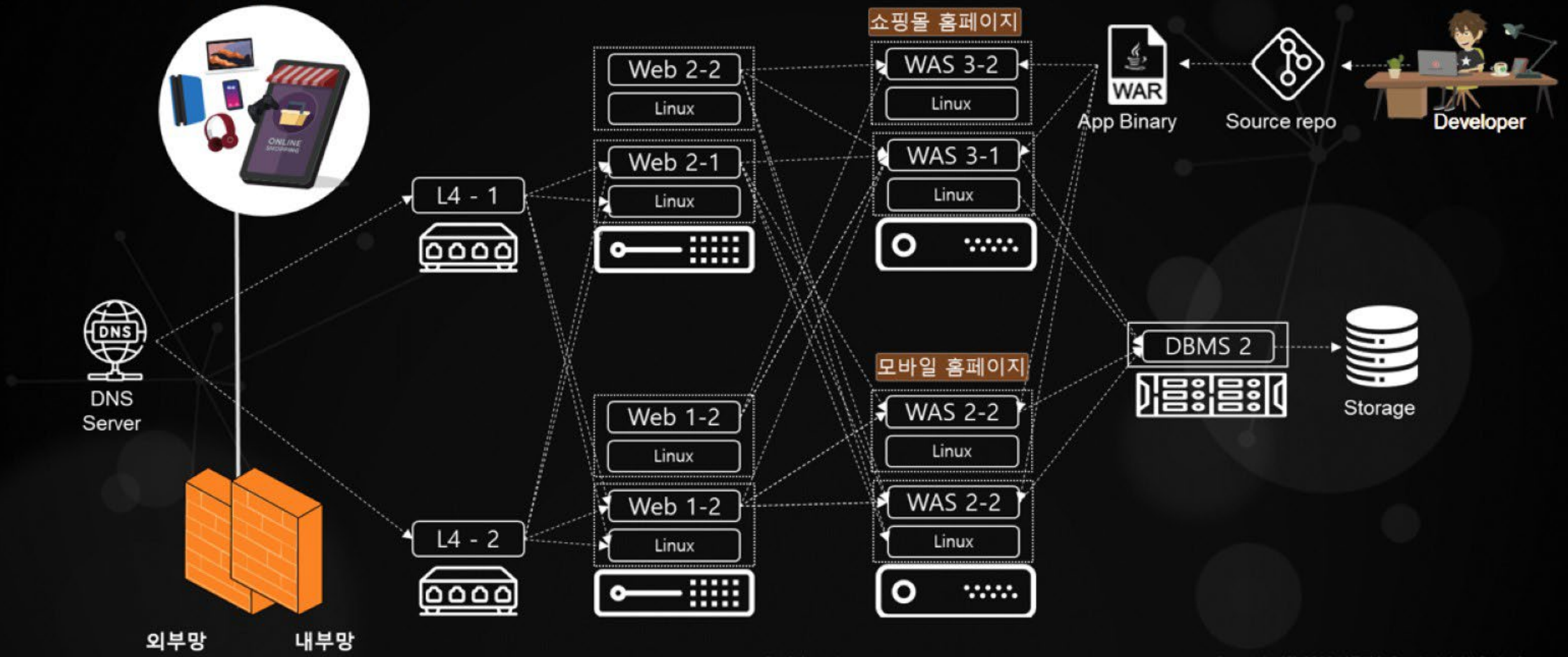
Application Centric Infrastructure

OPENMARU © 2021 | All Rights Reserved.

AS IS – 인프라 구성

Cocktail Cloud APM

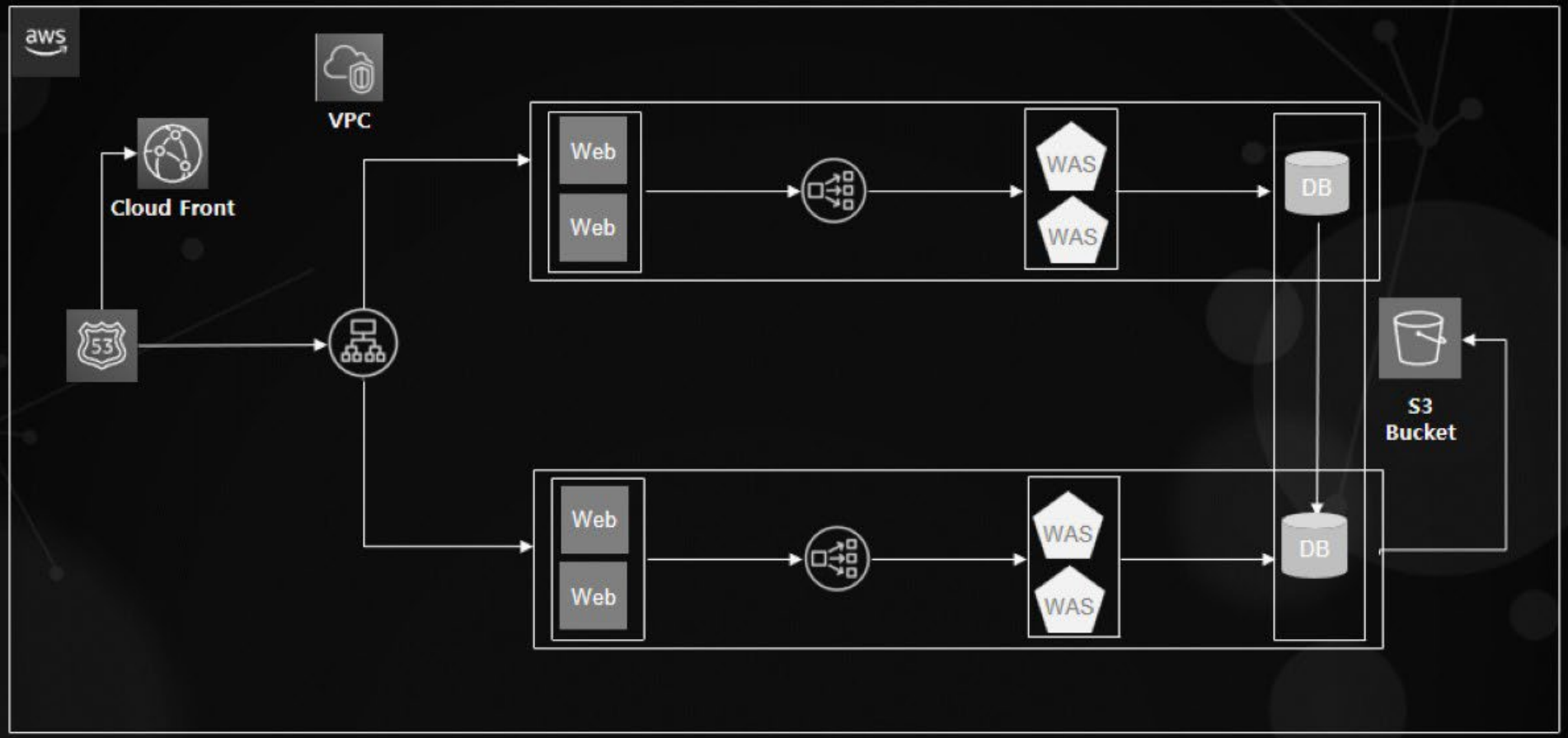
- 웹서버, WAS 서버, 데이터베이스 의 역할 별로 티어를 나눈 3 티어 구조
  - 각 티어 별로 확장과 관리



Confidential
Copyright (C) 2023 나무기술 Co., Ltd. | All Right Reserved.

AWS 클라우드 기반 구성

- Public 클라우드로 전환하지만 가상머신을 사용하는 것은 같기 때문에 AS-IS 시스템만큼 관리가 필요
  - EC2 = 가상머신, ELB = L4, VPC = VLAN, RDS = RAC

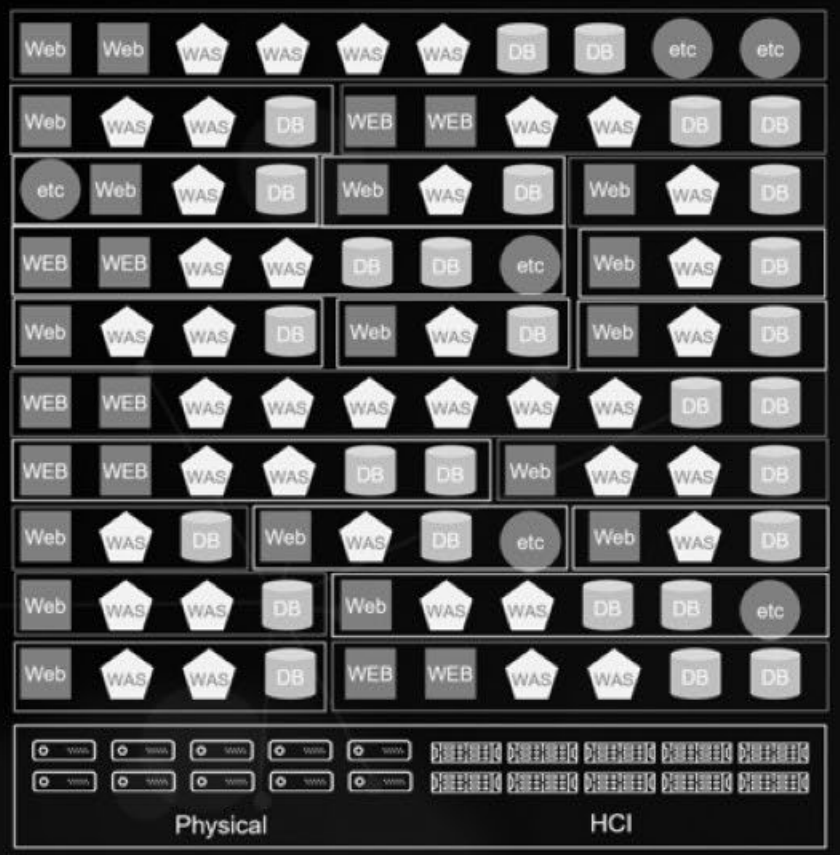


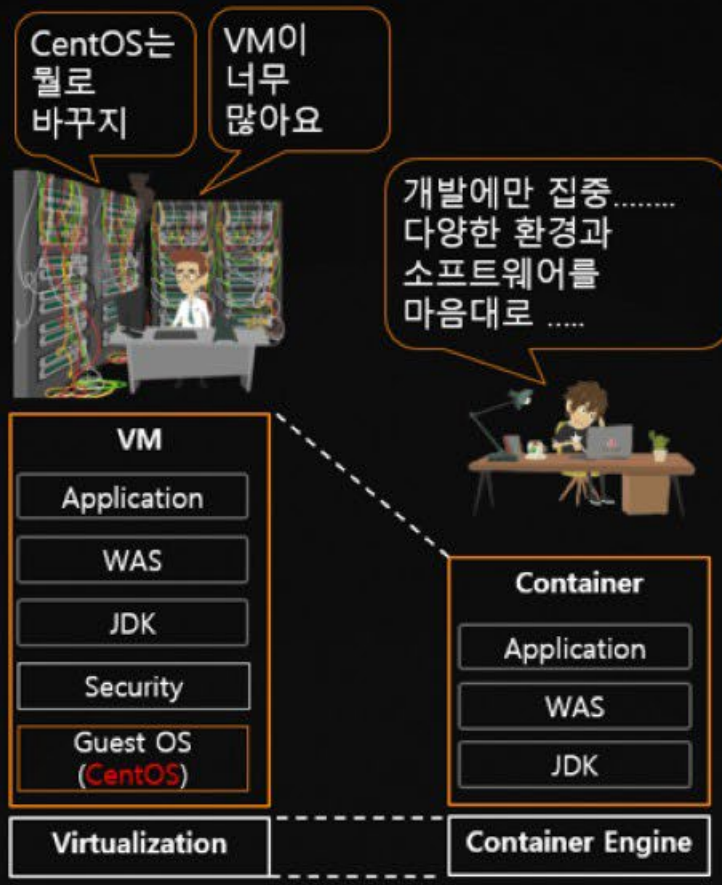
OPENMARU © 2021 | All Rights Reserved.

Legacy 시스템 구축 vs. 클라우드 네이티브 구축

openmaru

시스템 마다 Web/WAS/DB 형태로 구축





- 물리서버나 가상화 형태로 WEB/WAS/DB 3 티어로 구축
  - Guest OS 에 따른 OS 라이선스 이슈와 보안 솔루션 구매 비용 발생

S/W 를 플랫폼에서 제공 – 개발에만 집중

- 애플리케이션에 필요한 S/W 만 컨테이너로 배포
  - 클라우드 네이티브로 전환, DevOps, CD/CD 기반

Confidential
© OPENMARU © 2021 | All rights reserved.

Pets vs Cattle

openmaru

Pets

Cattle

- Confidential -

OPENMARU © 2021 | All Rights Reserved.

Executive Summary

- PaaS 기술 중 하나인 "OpenShift Container Platform"은 개발환경 및 자원을 클라우드(공동화)하여 빠르게 개발환경 및 인프라를 사용할 수 있는 기술입니다.
- 클라우드 서비스 중 PaaS (Platform as a Service)의 도입은 Digital Transformation의 가장 핵심이 되는 기술요소로 진화하였습니다. 그 이유로는,
- IT기획 및 비즈니스 측면에서 개발단계가 획기적으로 감소하고, 개발자와 운영담당자 간의 R&R Gray zone이 제거되어, 더욱 빠르게 비즈니스 변화에 대처할 수 있고 장애대응 속도 역시 증가됩니다.
- IT개발적인 측면에서 불필요한 업무 (IT자원할당, 개발 후 테스트 및 배포에 따른 업무)가 혁신적으로 감소, 개발업무에 더 창의성과 리소스를 투자할 수 있습니다.
- IT운영적인 측면에서는 다양한 운영환경을 통합적으로 관리할 수 있게 되어 비즈니스 확장에 추가적인 리소스없이도 빠르고 안정적으로 대응할 수 있고, 업무 연속성이 획기적으로 증가되어 더이상 IT운영담당자가 신경쓰지 않아도 될 정도로 빠르게 대응할 수 있습니다.
- OpenShift Container Platform은 이제 가상화 시대가 끝나고 클라우드 시대로 넘어가는 IT트렌드의 첨병에 있으며 이후 IT자원효율화 시대를 이끌어 나갈 가장 핵심적인 기술입니다.

- Confidential -
OPENMARU © 2021 | All Rights Reserved.

Application Performance Management
감사합니다.
openmaru
APM

openmaru

Cloud Native 기술을 통한 Open Hybrid Cloud 구현

어떤 공공업무에 클라우드 네이티브의 적용이 가능할까요?

클라우드 네이티브 대상 업무선정 방향 (전문가 의견)

클라우드 네이티브 업무는 학계, 업체, 정부정책을 반영하여 대상을 선정할 수 있습니다.

서비스 복잡도가 높은 시스템

- 마틴파울러(2015, 최초 용어정의) : "마이크로서비스는 복잡한 시스템에서 유용할 때 MSA전환"

+ 정부정책

명확한 경계가 가능한 시스템

- 샘뉴먼(2019, 저서) : "해당분야를 제대로 이해하지 못해 적절한 경계를 찾기 어렵다면 MSA전환 불리"

더 이상 확장할 수 없는 한계지점에 도달한 시스템

- 수잔파울러(2019, 저서) : "확장성 한계로 인해 심각한 안정 문제 발생하여, 개발생산성·효율성 저하 시 MSA 전환"

Source : 클라우드 네이티브 추진 시 고려사항 (교육 교재)
클라우드 네이티브 기반 행정·공공 서비스 확산 지원 - 한국지능정보사회진흥원

- Confidential -
OPENMARU © 2021 | All Rights Reserved.

클라우드 네이티브 대상 업무선정 방향 (전문가 의견)

- 클라우드 네이티브 업무는 학계, 업체, 정부정책을 반영하여 대상을 선정할 수 있습니다.

정부내 전환가능 업무 식별

✔ SRM(서비스참조모델)의 전환가능 업무

정부내지원 서비스 →	업무변화 少 (적용 불리)
공통기술 서비스 →	시스템 연계 多 (적용 유리)
대국민 서비스 →	업무변화 多/ 폭주성 高 (적용 유리)

기관·시스템특성을 반영한 업무선정

✔ 시스템 특성 반영 (시스템복잡성)

규모 大 × 부서 多 × 시스템 메뉴 多

※국가및기초자치 단체226, 공공기관 338개기관업무대상

✔ 정부정책 특성 반영 (제도개선이많은업무)

A기관	B기관	C기관	D기관

Source : 클라우드 네이티브 추진 시 고려사항 (교육 교재)
클라우드 네이티브 기반 행정·공공 서비스 확산 지원 - 한국지능정보사회진흥원

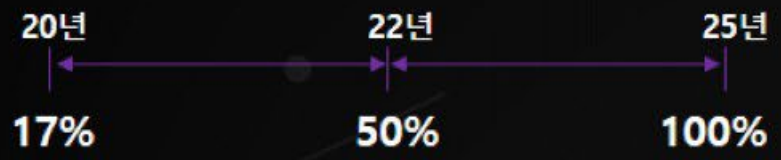
- Confidential -
OPENMARU © 2021 | All Rights Reserved.

클라우드 네이티브 적용 검토

- 공공 클라우드 전면 전환에 따른 클라우드 네이티브 상호 운용성 확보하여 서비스 간 연결

공공클라우드 전면 전환 사업진행

✔ 공공기관 전면전환 비율 (목표)

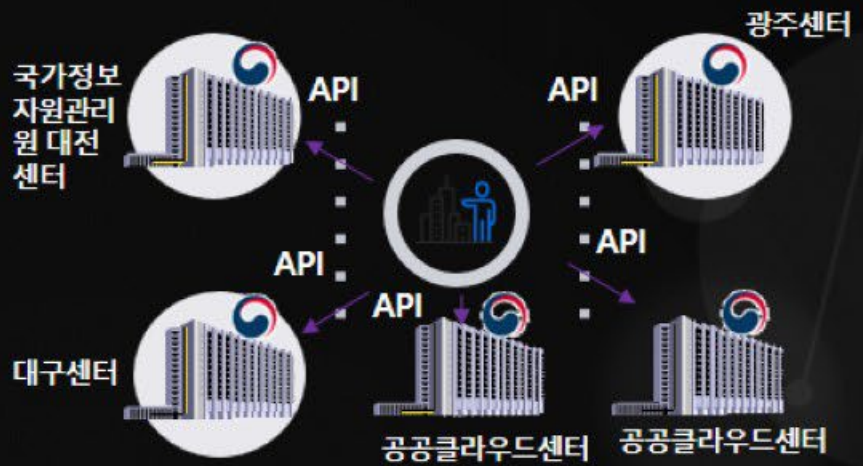


✔ 공공 (G-클라우드, 자체), 민간클라우드 센터

공공 클라우드센터	민간 클라우드센터
54%	46%

공공클라우드 센터 상호운용성 확보

✔ 디지털정부 서비스 개발환경인 클라우드 표준 플랫폼으로 고도화 필요



※ 마이크로서비스아키텍처에서는API로멀티센터의서비스를통합제공

Source : 클라우드 네이티브 추진 시 고려사항 (교육 교재)
클라우드 네이티브 기반 행정·공공 서비스 확산 지원 - 한국지능정보사회진흥원

- Confidential -
OPENMARU © 2021 | All Rights Reserved.

정보시스템 자가 진단 체크리스트

현행 정보시스템에 대한 체크리스트를 도출하였으며, 6개 이상 "Y"응답시, 클라우드 네이티브 도입이 필요한 것으로 판단할 수 있습니다.

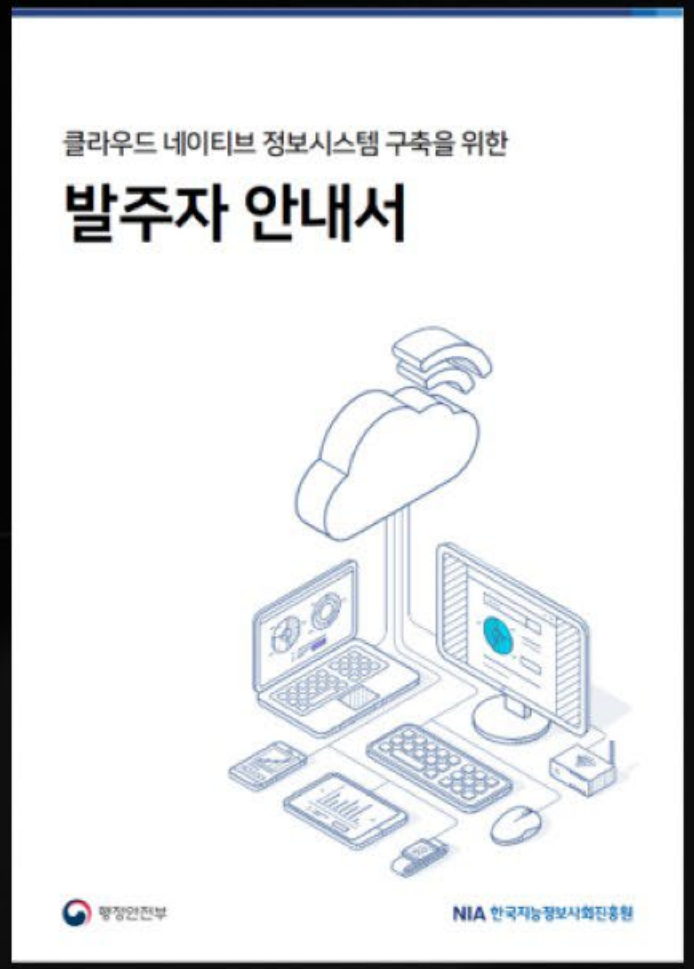
정보시스템 자가진단 체크리스트 예시

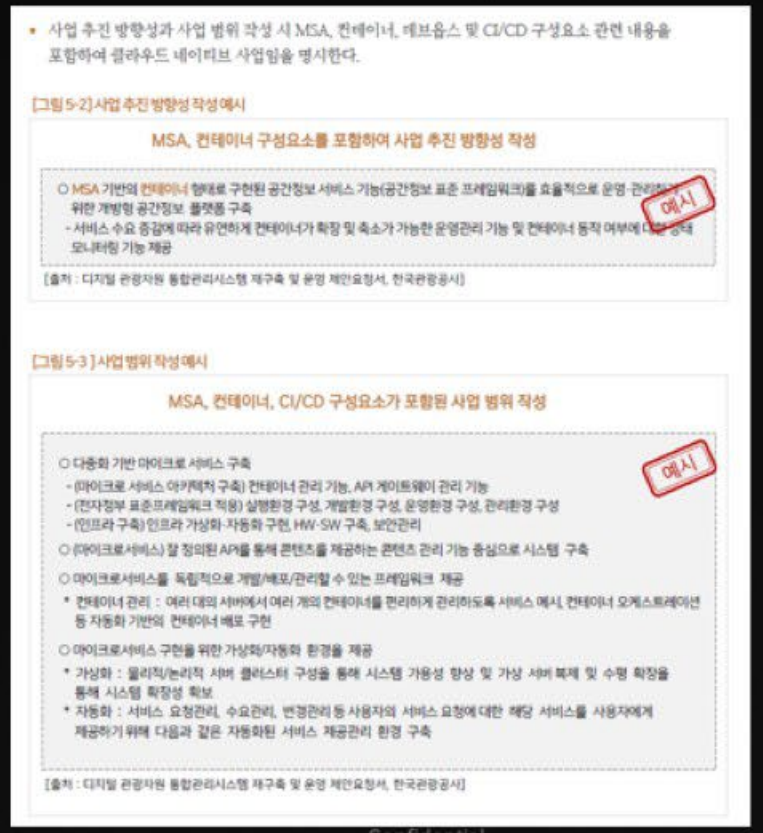
구분(목표)		자가진단항목	답변	
안정적 서비스 운영	1	• 초기개발비의 약 15%이상을 매년 추가개발 및 유지보수 비용으로 사용하고 있습니까?	✓	6개 이상 "YES" 응답 시 도입 검토
	2	• 다양한 원인에 의한 장애 발생 시 장애복구(예.시스템증설, 업그레이드 등)를 위해 서비스를 중단한 적이 있습니까?	✓	
	3	• 특정시점(년, 월, 주, 시)에 트래픽이 증가로 접속지연으로 불만이 제기된 적이 있습니까?		
업무 및 기술 변화 대응	4	• 수시로 정책, 업무 요건 등의 변화에 따른 요구사항에 대해 신속한 대응이 필요합니까?		
	5	• 디지털 신기술(빅데이터, AI, 블록체인, IoT등) 적용 및 다양한 언어 및 다양한 오픈소스에 대한 요구사항 반영이 필요합니까?	✓	
	6	• 소규모 서비스 단위로 기능과 DB의 명확한 분리가 가능하고, 독립적 단위로 실행이 가능합니까? (공통 기능 및 데이터 사용, 타 시스템과의 연계성, 서비스 의존관계 등 확인)	✓	
개발 품질 향상	7	• 시스템 개발 및 운영시 개발 및 운영 조직의 분리에 따라 의사소통, 개발 및 배포 지연 등의 문제가 존재합니까?	✓	
	8	• 소스코드의 복잡성으로 서비스 확장이 곤란하여 서비스 분리 및 소스코드 개선이 필요합니까?	✓	
개발기간	9	• 개발된 SW를 형상관리 시스템에 커밋 후 개발계, 검증계, 운영계 서버에서 빌드, 테스트, 배포하는 과정에 빌드·테 스트·배포 도구를 사용하지 않거나 부분적으로 사용하고 있습니까?		
	10	• 현행 시스템의 배포주기를 단축하고 싶습니까?		

NMARU © 2021 | All Rights Reserved.

발주자 안내서 - 클라우드 네이티브 정보시스템 구축

- 발주기관에서 클라우드 기반 정보화 사업을 기획하고 발주하기 전에 발주자 안내서를 통해 클라우드 네이티브의 개념 , 주요 기술 등을 이해하고 도입 적합성을 검토





• 사업 추진 방향성과 사업 범위 작성 시 MSA, 컨테이너, 데브옵스 및 CI/CD 구성요소 관련 내용을 포함하여 클라우드 네이티브 사업임을 명시한다.

[그림 5-2] 사업 추진 방향성 작성 예시

MSA, 컨테이너 구성요소를 포함하여 사업 추진 방향성 작성

○ MSA 기반의 컨테이너 형태로 구현된 공간정보 서비스 기능(공간정보 표준 프레임워크)을 효율적으로 운영·관리하기 위한 개방형 공간정보 플랫폼 구축
- 서비스 수요 증감에 따라 유연하게 컨테이너가 확장 및 축소가 가능한 운영관리 기능 및 컨테이너 동작 여부에 대한 상태 모니터링 기능 제공

예시

[출처 : 디지털 관광자원 통합관리시스템 재구축 및 운영 제안요청서, 한국관광공사]

[그림 5-3] 사업 범위 작성 예시

MSA, 컨테이너, CI/CD 구성요소가 포함된 사업 범위 작성

○ 다중화 기반 마이크로 서비스 구축
- (마이크로 서비스 아키텍처 구축) 컨테이너 관리 기능, API 게이트웨이 관리 기능
- (전자정부 표준프레임워크 적용) 실행환경 구성, 개발환경 구성, 운영환경 구성, 관리환경 구성
- (인프라 구축) 인프라 가상화-자동화 구현, HW·SW 구축, 보안관리

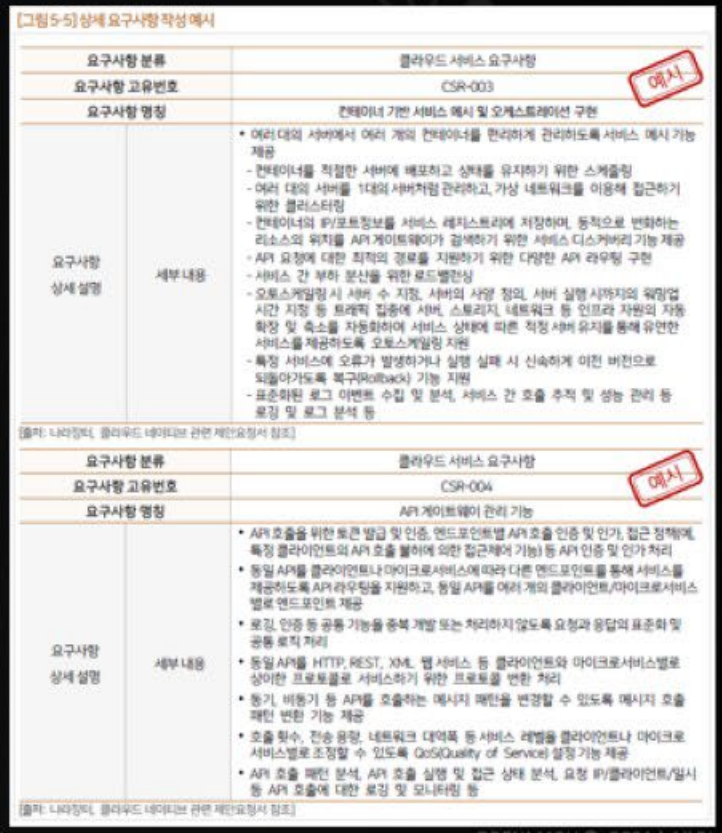
○ (마이크로서비스) 잘 정의된 API를 통해 콘텐츠를 제공하는 콘텐츠 관리 기능 중심으로 시스템 구축

○ 마이크로서비스를 독립적으로 개발/배포/관리할 수 있는 프레임워크 제공
* 컨테이너 관리 : 여러 대의 서버에서 여러 개의 컨테이너를 편리하게 관리하도록 서비스 메시, 컨테이너 오케스트레이션 등 자동화 기반의 컨테이너 배포 구현

○ 마이크로서비스 구현을 위한 가상화/자동화 환경을 제공
* 가상화 : 물리적/논리적 서버 클러스터 구성을 통해 시스템 가용성 향상 및 가상 서버 복제 및 수평 확장을 통해 시스템 확장성 확보
* 자동화 : 서비스 요청관리, 수요관리, 변경관리 등 사용자의 서비스 요청에 대한 해당 서비스를 사용자에게 제공하기 위해 다음과 같은 자동화된 서비스 제공관리 환경 구축

예시

[출처 : 디지털 관광자원 통합관리시스템 재구축 및 운영 제안요청서, 한국관광공사]



[그림 5-5] 상세 요구사항 작성 예시

요구사항 분류		클라우드 서비스 요구사항
요구사항 고유번호		CSR-003
요구사항 명칭		컨테이너 기반 서비스 메시 및 오케스트레이션 구현
요구사항 상세 설명	세부 내용	• 여러 대의 서버에서 여러 개의 컨테이너를 편리하게 관리하도록 서비스 메시 기능 제공 - 컨테이너를 적절한 서버에 배포하고 상태를 유지하기 위한 스케줄링 - 여러 대의 서버를 1대의 서버처럼 관리하고, 가상 네트워크를 이용해 접근하기 위한 클러스터링 - 컨테이너의 IP/포트정보를 서비스 레지스트리에 저장하여, 동적으로 변화하는 리소스의 위치를 API 게이트웨이가 검색하기 위한 서비스 디스커버리 기능 제공 - API 요청에 대한 최적의 경로를 지원하기 위한 다양한 API 라우팅 구현 - 서비스 간 부하 분산을 위한 로드밸런싱 - 오토스케일링 시 서버 수 지정, 서버의 사양 정의, 서버 실행 시까지의 워밍업 시간 지정 등 트래픽 집중에 서버, 스토리지, 네트워크 등 인프라 자원의 자동 확장 및 축소를 자동화하여 서비스 상태에 따른 적정 서버 유지를 통해 유연한 서비스를 제공하도록 오토스케일링 지원 - 특정 서비스에 오류가 발생하거나 실행 실패 시 신속하게 이전 버전으로 되돌아가도록 복구(Rollback) 기능 지원 - 표준화된 로그 이벤트 수집 및 분석, 서비스 간 호출 추적 및 성능 관리 등 로깅 및 로그 분석 등

예시

[출처: 나라장터, 클라우드 네이티브 관련 제안요청서 참조]

요구사항 분류		클라우드 서비스 요구사항
요구사항 고유번호		CSR-004
요구사항 명칭		API 게이트웨이 관리 기능
요구사항 상세 설명	세부 내용	• API 호출을 위한 토큰 발급 및 인증, 엔드포인트별 API 호출 인증 및 인가, 접근 정책에 따른 특정 클라이언트의 API 호출 불허에 의한 접근제어 기능) 등 API 인증 및 인가 처리 • 동일 API를 클라이언트나 마이크로서비스에 따라 다른 엔드포인트를 통해 서비스를 제공하도록 API 라우팅을 지원하고, 동일 API를 여러 개의 클라이언트/마이크로서비스별로 엔드포인트 제공 • 로깅, 인증 등 공통 기능을 중복 개발 또는 처리하지 않도록 요청과 응답의 표준화 및 공통 로직 처리 • 동일 API를 HTTP, REST, XML, 웹 서비스 등 클라이언트와 마이크로서비스별로 상이한 프로토콜로 서비스하기 위한 프로토콜 변환 처리 • 동기, 비동기 등 API를 호출하는 메시지 패턴을 변경할 수 있도록 메시지 호출 패턴 변환 기능 제공 • 호출 횟수, 전송 용량, 네트워크 대역폭 등 서비스 레벨을 클라이언트나 마이크로서비스별로 조정할 수 있도록 QoS(Quality of Service) 설정 기능 제공 • API 호출 패턴 분석, API 호출 실행 및 접근 상태 분석, 요청 IP/클라이언트/일시 등 API 호출에 대한 로깅 및 모니터링 등

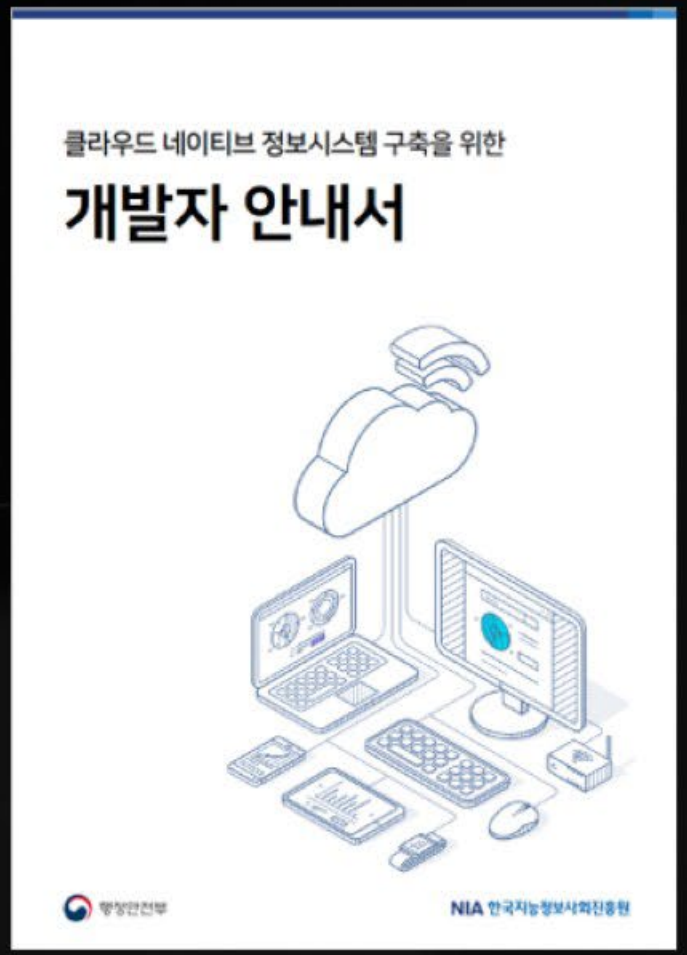
예시

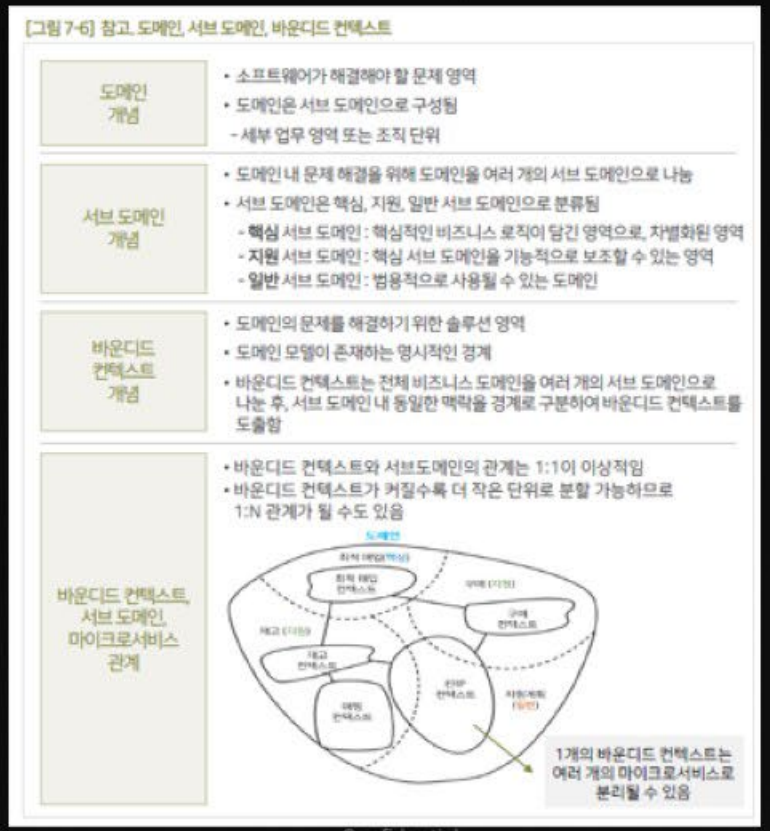
[출처: 나라장터, 클라우드 네이티브 관련 제안요청서 참조]

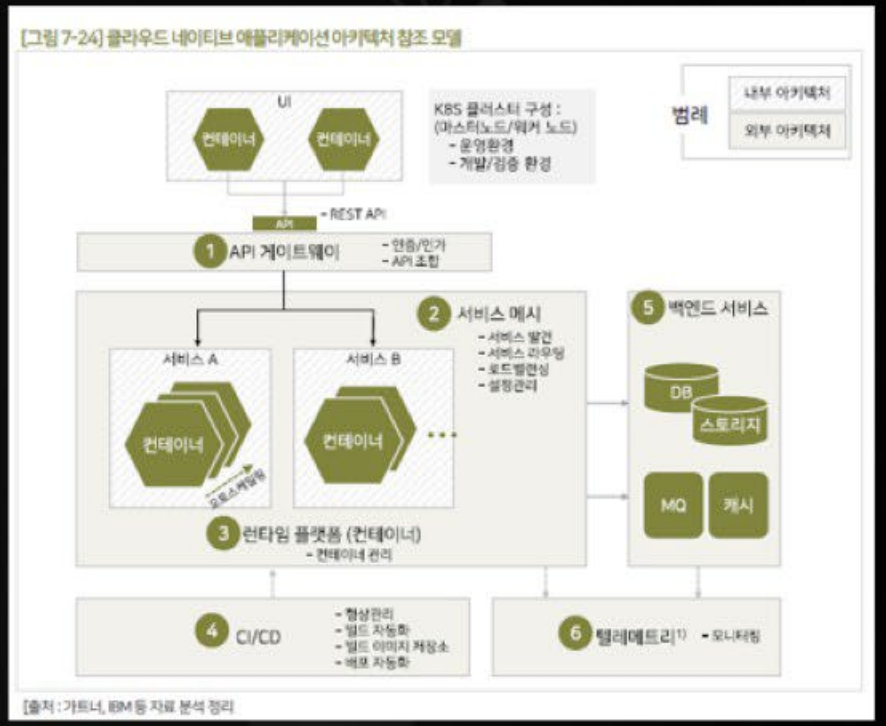
OPENMARU © 2021 | All Rights Reserved.

개발자 안내서 - 클라우드 네이티브 정보시스템 구축

- 클라우드 관련 정보화 사업을 준비하는 중앙행정기관 , 지방자치단체 , 공공기관 등 발주자와 클라우드 네이티브 정보시스템 구축 및 운영 사업에 참여하거나 관심이 있는 개발자

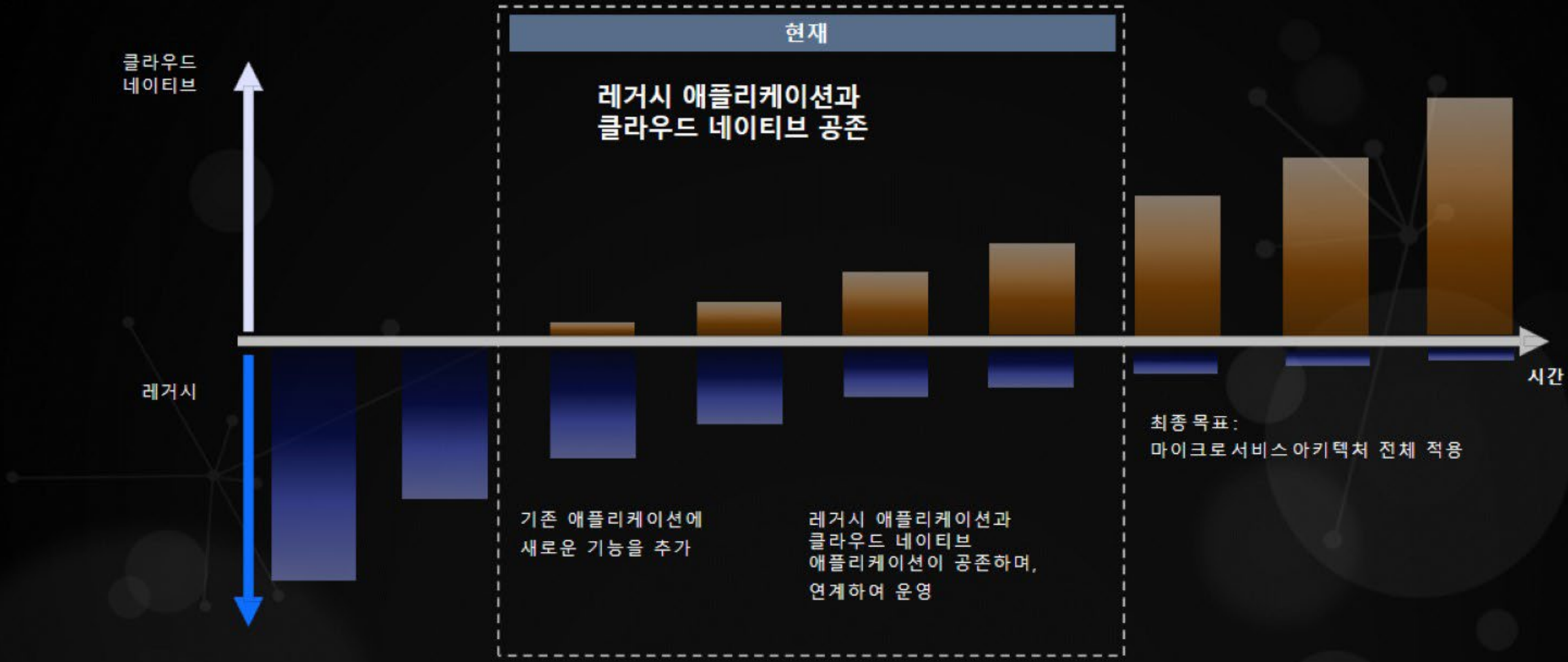






OPENMARU © 2021 | All Rights Reserved.

레거시와 클라우드 네이티브 애플리케이션 전환

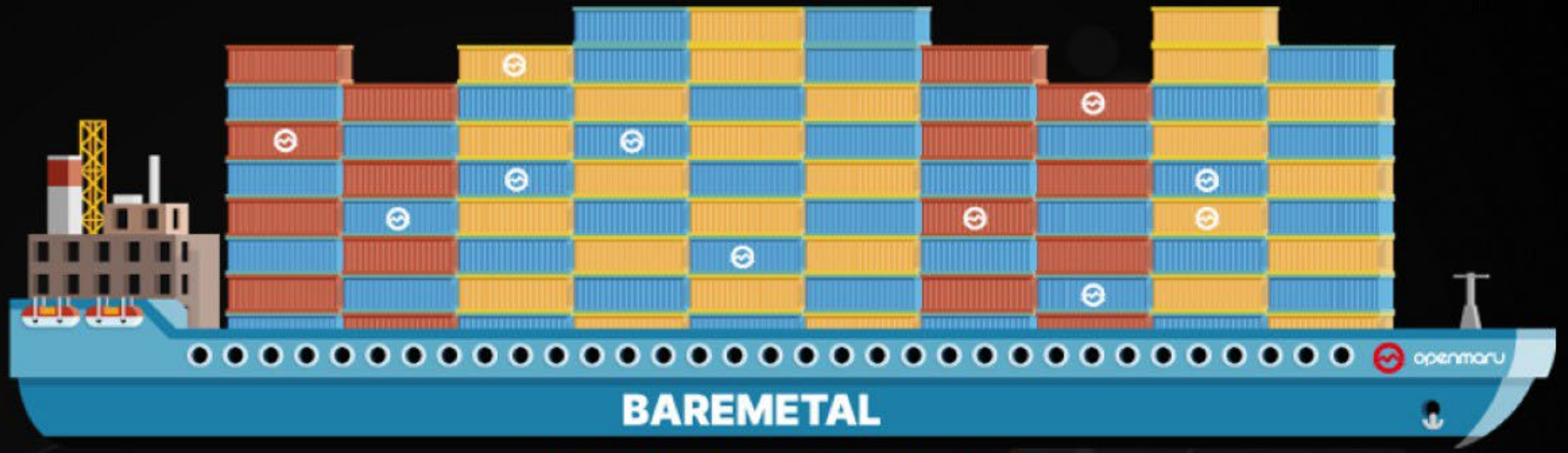


Source : IBM

OPENMARU © 2021 | All Rights Reserved.

Cloud Native Summary

Cloud Native vs. Cloud Immigrant



배위에 규격화된 컨테이너를 바로 올려요

배위에 또 배를 싣고 그 위에 짐을 올려요

- Confidential -

OPENMARU © 2021 | All Rights Reserved.

Maker (OpenShift) vs. Taker (... ?)

openmaru

Maker

오픈소스 리더, 커미터

Red Hat

okd

Vs

Taker

오픈소스 기술지원

안정성,성능, 보안 등 품질 관리는?

구축은 했지만, 운영은 누가?

유지보수 (Patch)

OPENMARU © 2021 | All Rights Reserved.

오픈 소스 S/W 의 Maker 와 Taker

openmaru

구분	Maker	Taker
제품 개발 방법	• 오픈소스로 만들고 책임도 지겠습니다.	• 만들지는 않았지만 오픈소스라 동일한 제품입니다.
비즈니스 형태	• 오픈소스 커뮤니티를 리딩하며, Full Time 엔지니어를 채용하여 오픈소스 S/W 제품을 개발	• 오픈소스 S/W 를 가져와 단순 설치나 기술지원
판매 방식	• 오픈소스는 소유가 불가능 -> 서브스크립션	• 인력 중심의 인건비
주요 제품	• Linux, JBoss, OpenShift, OpenStack 등	• CentOS, Apache , Tomcat , Redis , Kubernetes
조직	• SW 제품 개발과 유지를 위한 전체 조직 체계 운영 • Product PM, Committer, QA, Engineer, Support 등	• 설치 중심의 기술지원 인력
기술지원	• 미션크리티컬 부분 - 성능, 보안 , 안정성 (엔진 중심) • 장기 제품 라이프사이클 지원	• 설치/구성 과 UI 커스터마이징 중심 • 성능, 보안, 안정성 그리고 제품에 대한 책임은 한계
라이선스 책임	• 도입 제품에 대한 오픈소스 라이선스 책임	• 없음
로드맵	• 제품의 로드맵 및 기능 개선/ 검토	• 제품 로드맵에 참여하기 어려움
개발자	• 컨설팅, 교육, 파트너 체계	• 설치/구성 엔지니어
Vendor	• Red Hat , Enterprise DB	• 단순 오픈소스 기술 지원

OPENMARU © 2021 | All Rights Reserved.