클라우드 네이티브 무상 컨설팅
찾아가는 클라우드 네이티브 세미나

고객분들이 ISP/ISMP 단계에서 요청하는 클라우드 네이티브를 위한 자료

클라우드 핵심 개념 : Architecture & Model

openmaru
오픈마루㈜

Architecture

클라우드 네이티브 (Container)

vs.

클라우드 이민 (Virtualization)

Service Model

SaaS : Software As A Service

PaaS : Platform As A Service

IaaS : Infrastructure As A Service

Delivery Model

Hybrid Cloud

Private Cloud

Public Cloud

클라우드 서비스 모델

openmaru 오픈마루(주)

On-Premise

- Application
- Data
- Runtime
- Middleware
- O/S
- Virtualization
- Servers
- Storage
- Network

- 고객은 인프라 제공, 유지 및 애플리케이션 호스팅 모두 책임

Infrastructure-as-a-Service (IaaS)

- Application
- Data
- Runtime
- Middleware
- O/S
- Virtualization
- Servers
- Storage
- Network

- 공급 업체는 인터넷을 통해 컴퓨팅 인프라를 제공
- 예 : AWS EC2, MSFT Azure

Platform-as-a-Service (PaaS)

- Application
- Data
- Runtime
- Middleware
- O/S
- Container
- Servers
- Storage
- Network

- 애플리케이션 개발을 위한 플랫폼을 제공
- 공급 업체는 서버, 스토리지, 네트워크를 관리
- 고객은 애플리케이션 관리
- 예 : OpenShift , Heroku

Software-as-a-Service (SaaS)

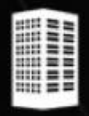
- Application
- Data
- Runtime
- Middleware
- O/S
- Container
- Servers
- Storage
- Network

- 인터넷을 통해 제공되는 소프트웨어
- 공급 업체가 소프트웨어 구축, 유지, 운영
- 예 : G-suite, Microsoft 365

범례 : 기업 고객 관리 | 클라우드 공급자 관리

Cloud Delivery Model

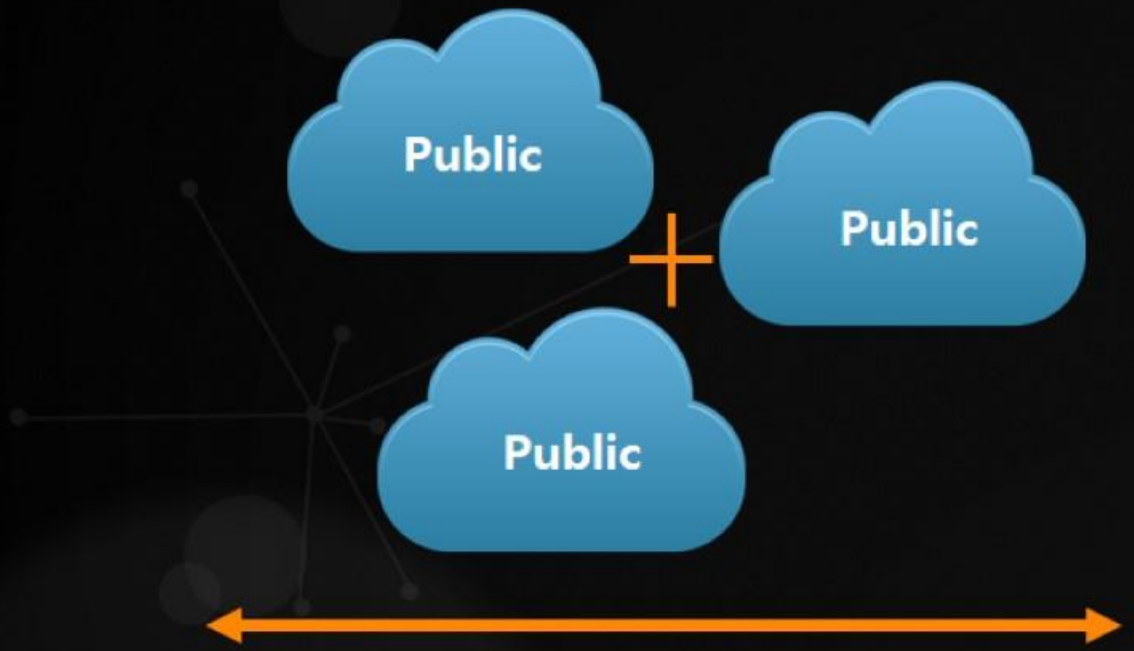
openmaru
오픈마루(주)

구분	클라우드 네이티브	구조	특징	벤더
	**Public Cloud**	클라우드 서비스 업체가 인터넷을 통해 컴퓨팅 리소스를 제공하고, 서버의 유지 관리	• 여러 기업이 하나의 클라우드 인프라를 이용 (Multi-Tenant) • 더 적은 구축 비용 • 더 적은 유지 보수	• AWS • Azure • Google Cloud • Oracle • Alibaba
	**Private Cloud**	기업이 클라우드 서버를 독점 아키텍처를 이용하여 자사의 데이터 센터 운영	• 하나의 기업에 하나의 인프라스트럭처 (싱글 테넌트) • On Premise 하드웨어 • 고객이 인프라 관리	• HPE • VMWare • Dell EMC • IBM • Red Hat
	**Hybrid Cloud**	On Premise 인프라, 프라이빗 클라우드 과 퍼블릭 클라우드의 혼합 컴퓨팅 환경	• 데이터 처리를 더 • 개인 및 제어 가능 • 버스트 기능 • 기존 시스템을 함유 할 수있다	• Red Hat • AWS • Azure • Google Cloud

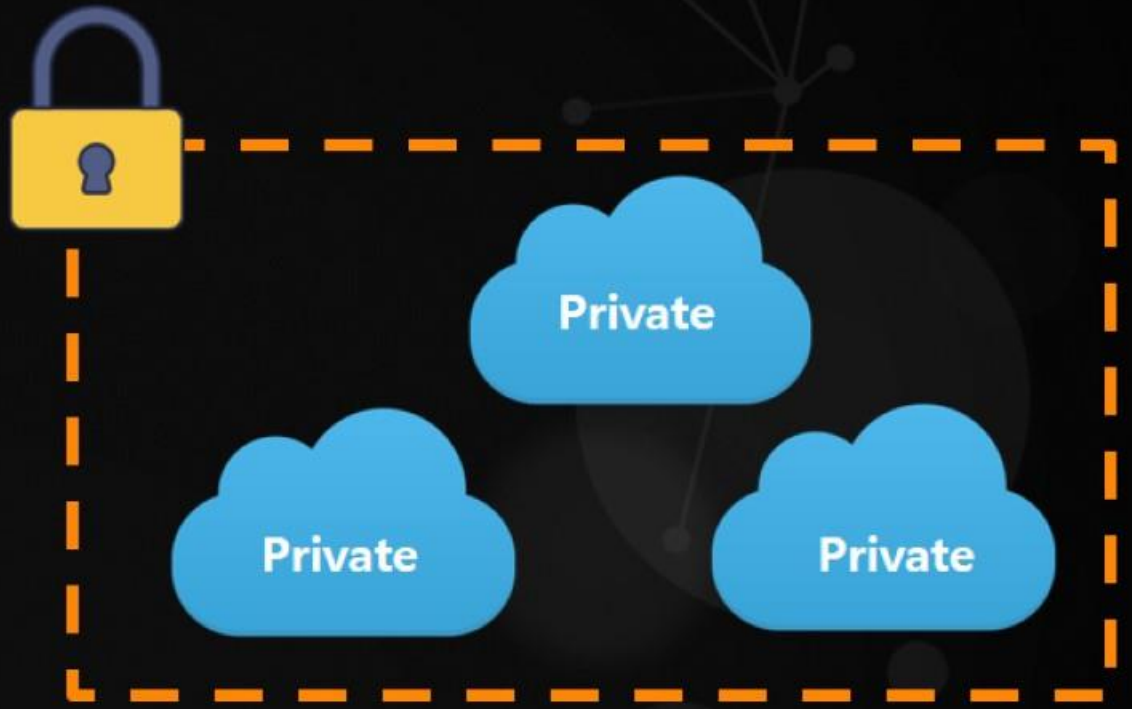
퍼블릭/프라이빗 클라우드의 장점

openmaru
오픈마루㈜

퍼블릭 클라우드의 빠르고 편리한 확장성

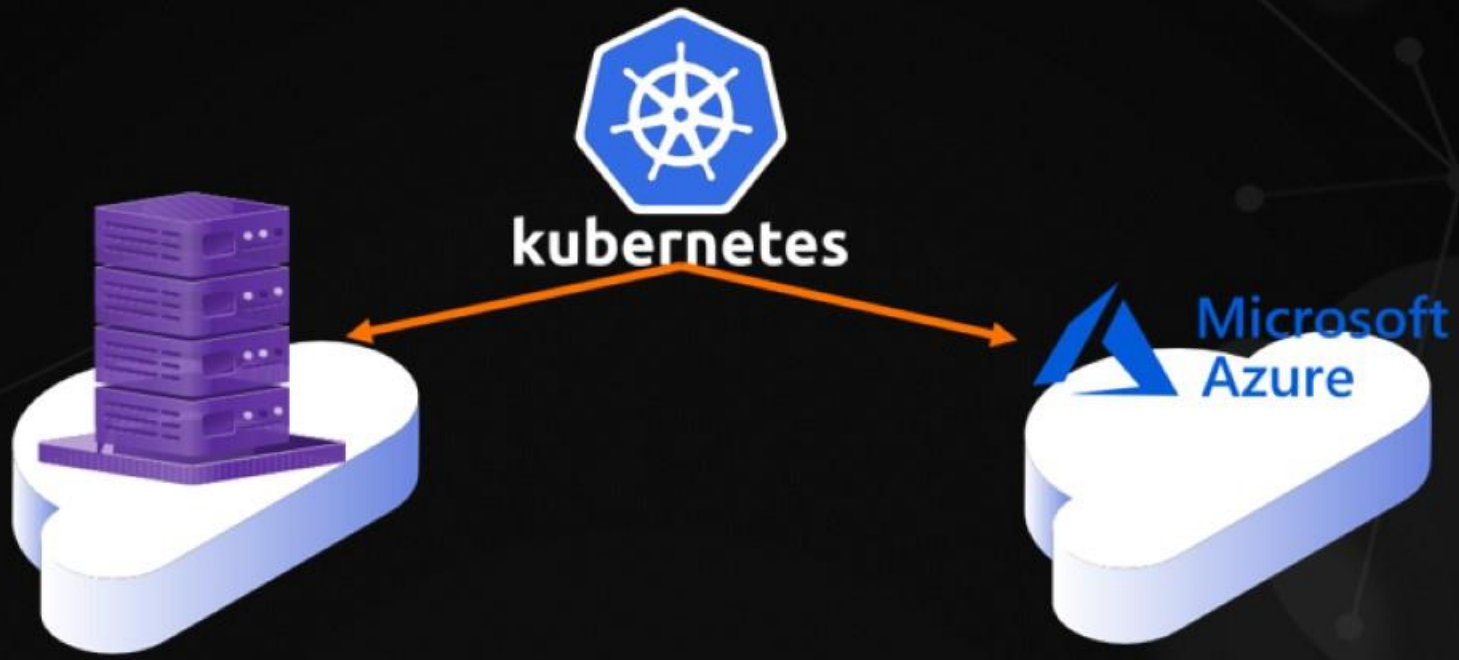


프라이빗 클라우드의 데이터, 보안 중요성



하이브리드 클라우드는 ?

- 하이브리드 클라우드란? 퍼블릭 클라우드와 프라이빗 클라우드를 결합
- 다양한 퍼블릭, 프라이빗 환경에서도 개발과 실행이 가능한 컨테이너 기반의 클라우드 네이티브 도입이 필요.



Private Cloud

- 기업 내에서 사용하기 위해 구축한 클라우드 컴퓨팅 환경
- 기업 내 환경/ 데이터 센터 등

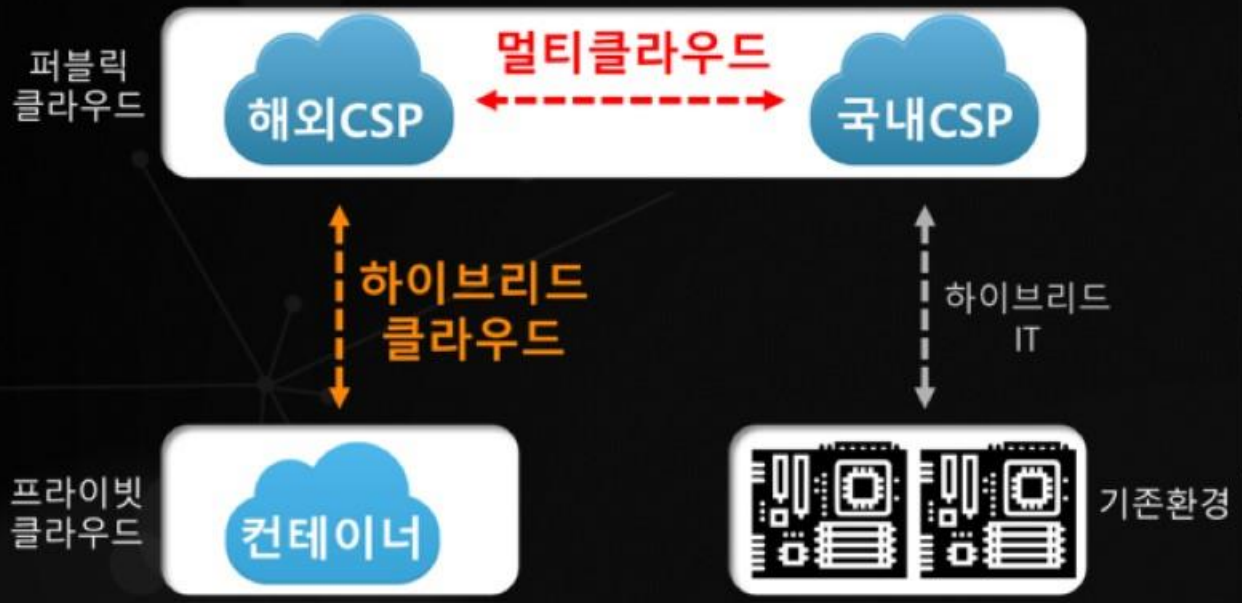
Public Cloud

- 일반에게 제공하는 클라우드 컴퓨팅 환경
- Amazon AWS, MS Azure , Google Cloud 등 클라우드 제공자

클라우드 이용 유형별 특징

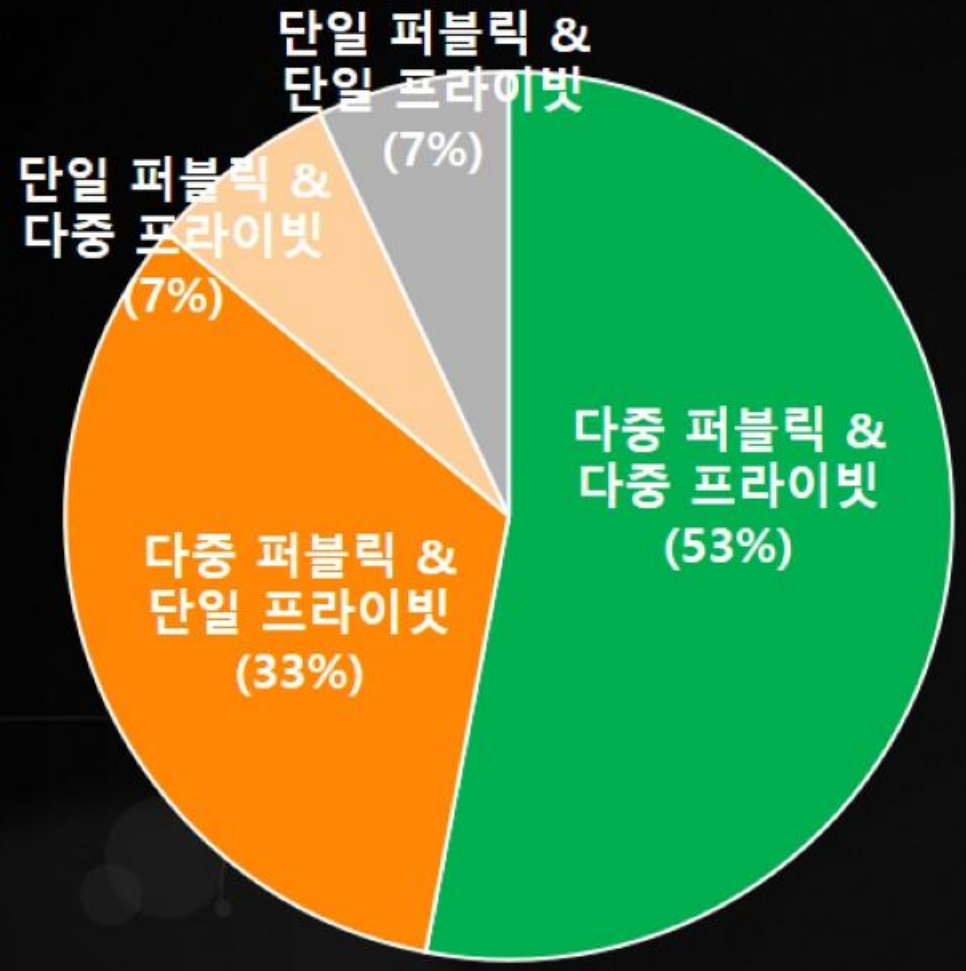
openmaru
오픈마루㈜

하이브리드 클라우드는 온프레미스 컴퓨팅, 프라이빗 클라우드,
퍼블릭 클라우드 **모두를 효과적으로 혼용하는 컴퓨팅 플랫폼**



종류	특징
프라이빗 클라우드	• 단일 조직에서 독점적으로 사용되는 컴퓨팅 리소스 제공 • 대규모, 보안이 중요한 조직에 적합
퍼블릭 클라우드	• 클라우드 서비스 공급자가 서버 및 클라우드 리소스 제공 • 소규모, 빠른 서비스가 필요한 조직에 적합
멀티 클라우드	• **여러 퍼블릭 클라우드를 함께 쓰는 방식** • 안정성 확보를 위한 클라우드 분산 운영이 필요한 조직에 적합
하이브리드 클라우드	• **퍼블릭 클라우드와 프라이빗 클라우드 장점을 모두 필요로 하는 조직에 적합** • 컨테이너 기반 클라우드 관리 • 가장 진화된 방식의 클라우드 이용 모델

클라우드 환경은 높은 이식성이 요구됨



출처: 플렉세라 보고서, 750개 기업 대상 조사 결과, 2020

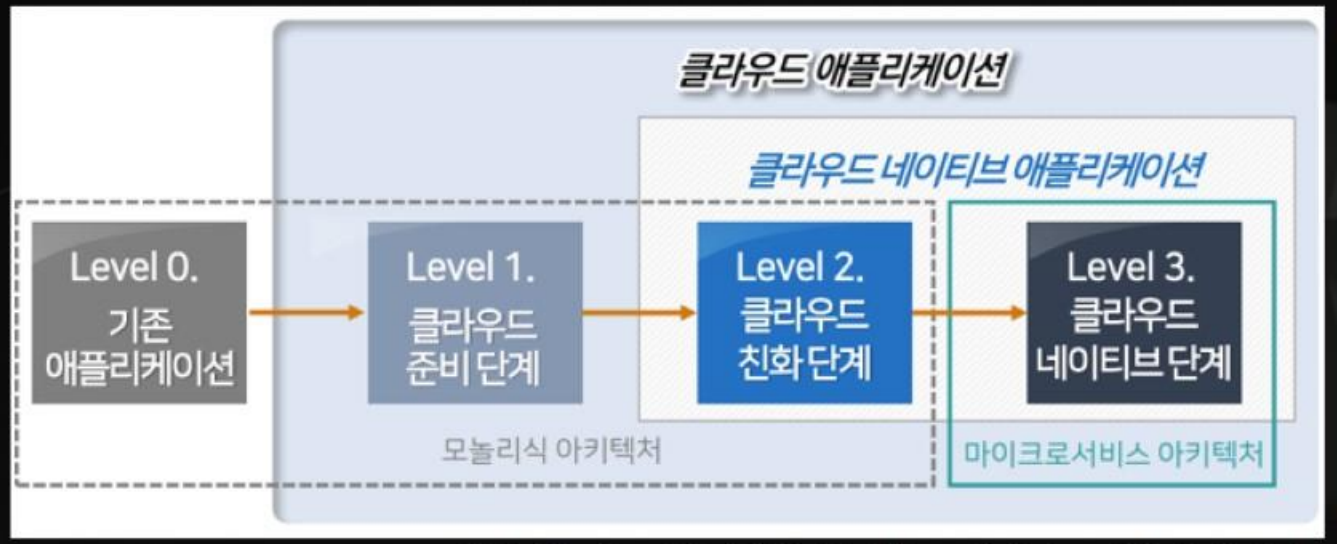
- 2020년 플렉세라 보고서에 따르면 클라우드 사용 기업의 7%만이 단일 퍼블릭 또는 단일 프라이빗 클라우드를 사용하는 것으로 응답함
- 93%의 기관은 2개 이상의 클라우드를 이용하고, 86%는 하이브리드 클라우드를 적용함
  - 클라우드 간 워크로드 이식 경험이 있는 조직은 36% 수준으로 상당히 높음

클라우드간 높은 이식성이 요구됨

클라우드 애플리케이션 성숙도 단계

openmaru
오픈마루(주)

- 클라우드 네이티브는 컨테이너 기반의 PaaS로 시작하여, CI/CD, MSA 모두 포함된 단계
- Lv1. 클라우드 준비 단계 → Lv2. 클라우드 친화 단계 → Lv3. 클라우드 네이티브 단계
  - PaaS와 컨테이너를 도입하는 클라우드 친화 단계
  - 데브옵스, CI/CD, MSA를 적용하는 클라우드 네이티브 단계



한국지능정보사회진흥원 클라우드 네이티브 발주자 안내서

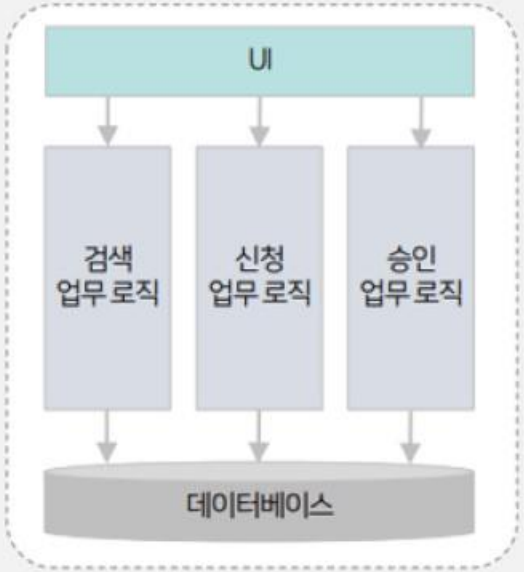
마이크로서비스 아키텍처로의 전환 예시

openmaru 오픈마루 (주)

한국지능정보사회진흥원 클라우드 네이티브 발주자 안내서

모놀리식 아키텍처

총무관리 시스템 - 회의실 관리 서비스



- 논리적 또는 물리적으로 웹[2], WAS[3], DB로 구성되어 수직적 인터페이스가 주로 발생

리팩터링 아키텍처

총무관리 시스템 - 회의실 관리 서비스

- 일부 서비스를 분리하여 별도 DB를 가지는 구조로 전환
- 서비스는 업무 로직 외에 서비스 연결을 위한 API가 필요함

마이크로서비스 아키텍처

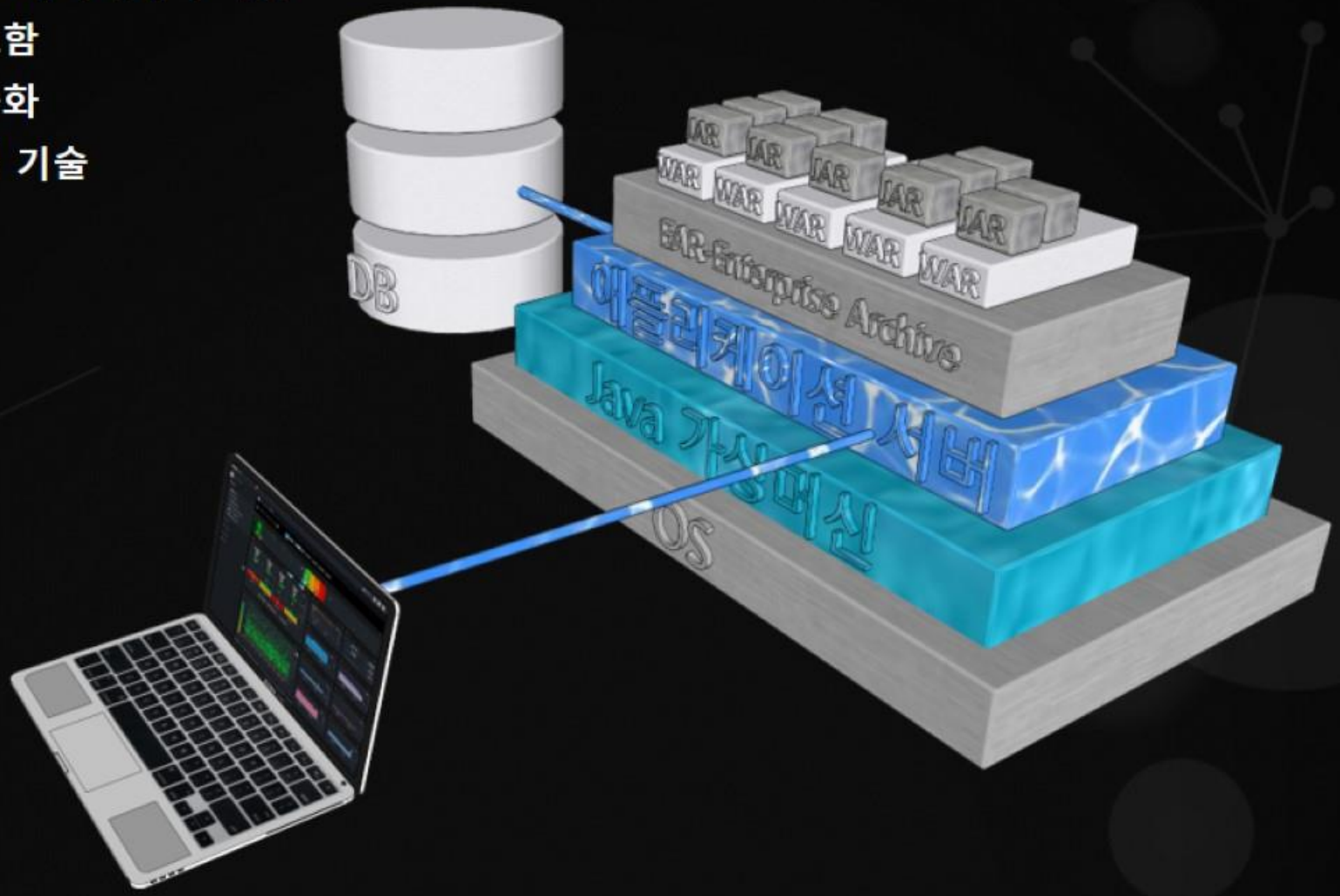
총무관리 시스템 - 회의실 관리 서비스

- 서비스별로 UI, API, 업무 로직, DB를 별도로 구성
- 해당 서비스별로 빌드-배포가 이뤄짐

Monolith Architecture

openmaru
오픈마루㈜

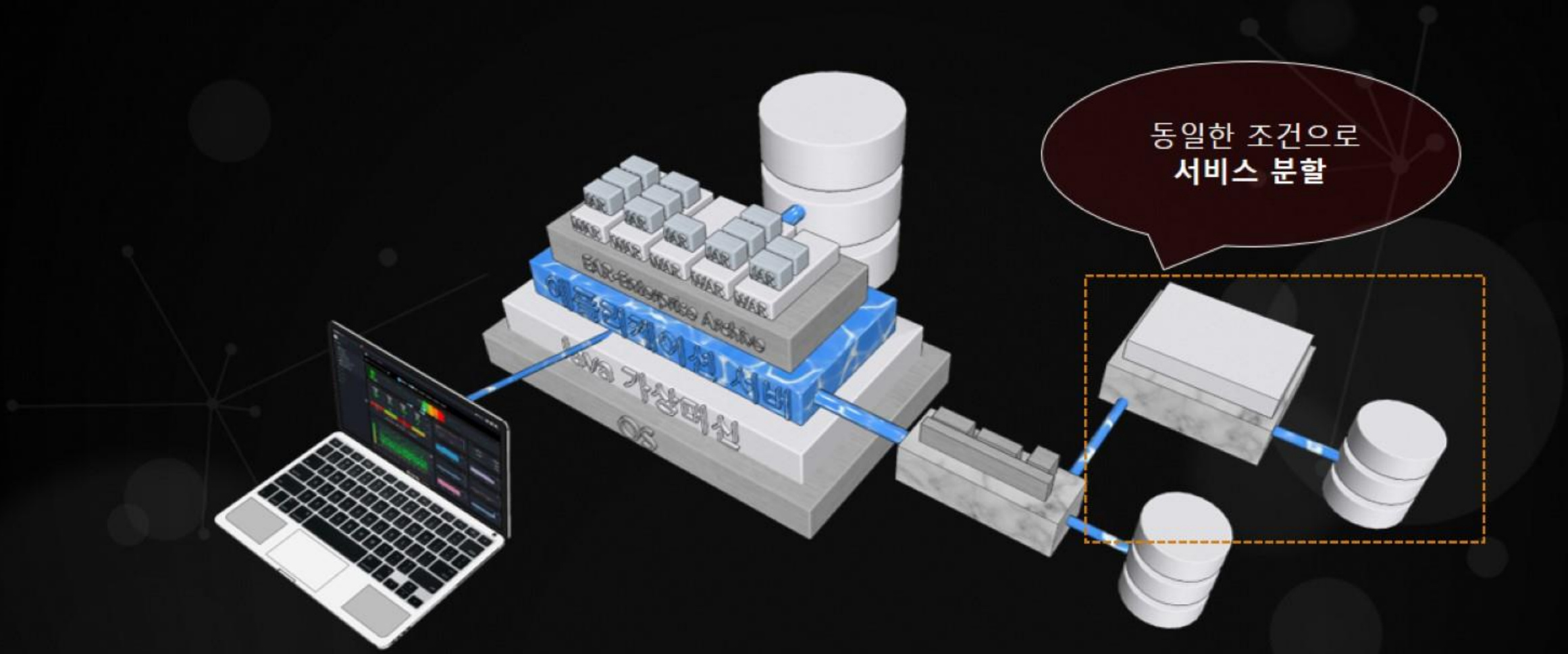
- 모노리스 아키텍처의 특징
  - 견고함
  - 표준화
  - 단일 기술



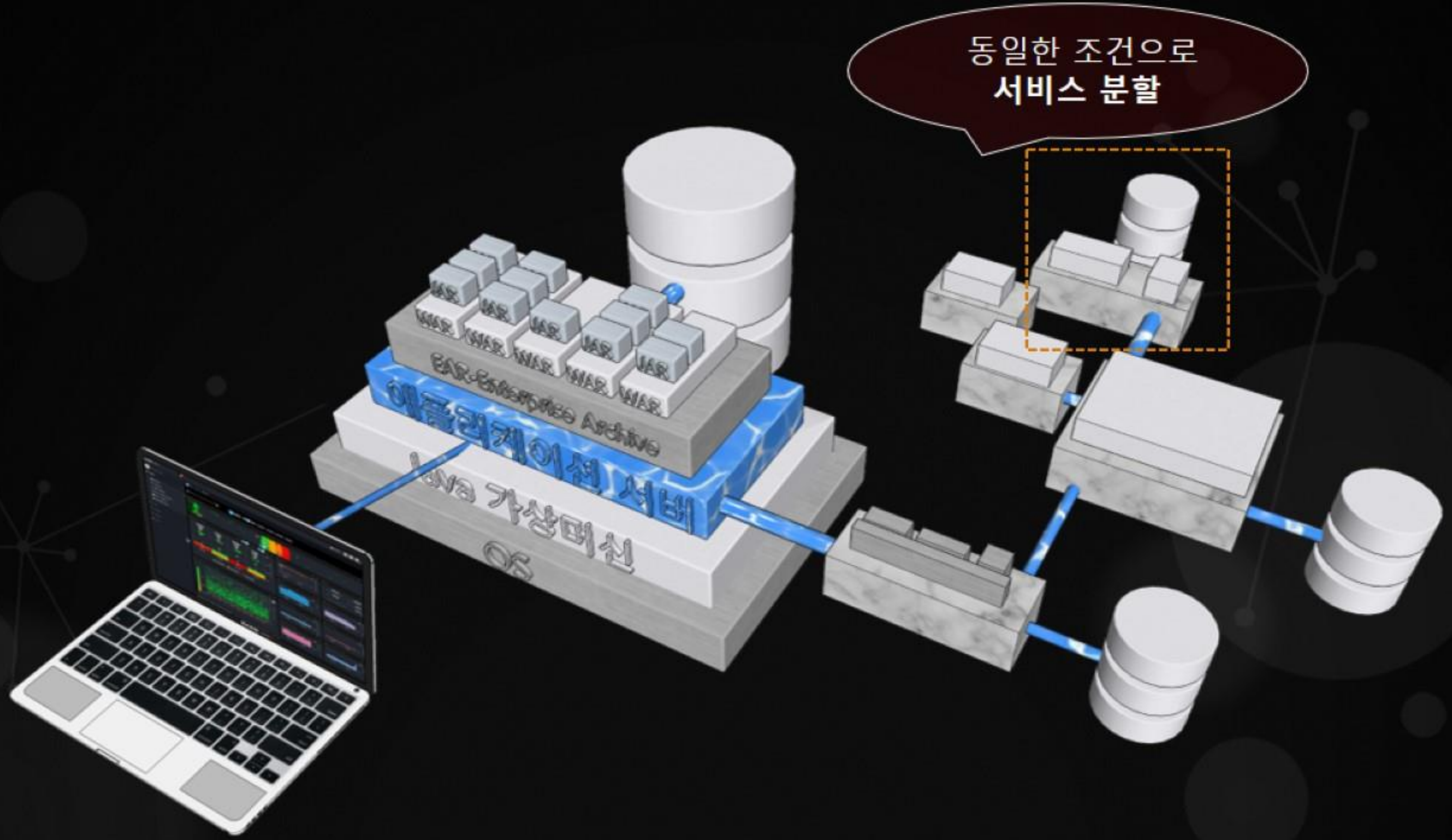
Monolith → Microservices – Phase 1

모노리스에서 마이크로서비스로 전환 – Phase 2

openmaru
오픈마루㈜



모노리스에서 마이크로서비스로 전환 – Phase 3

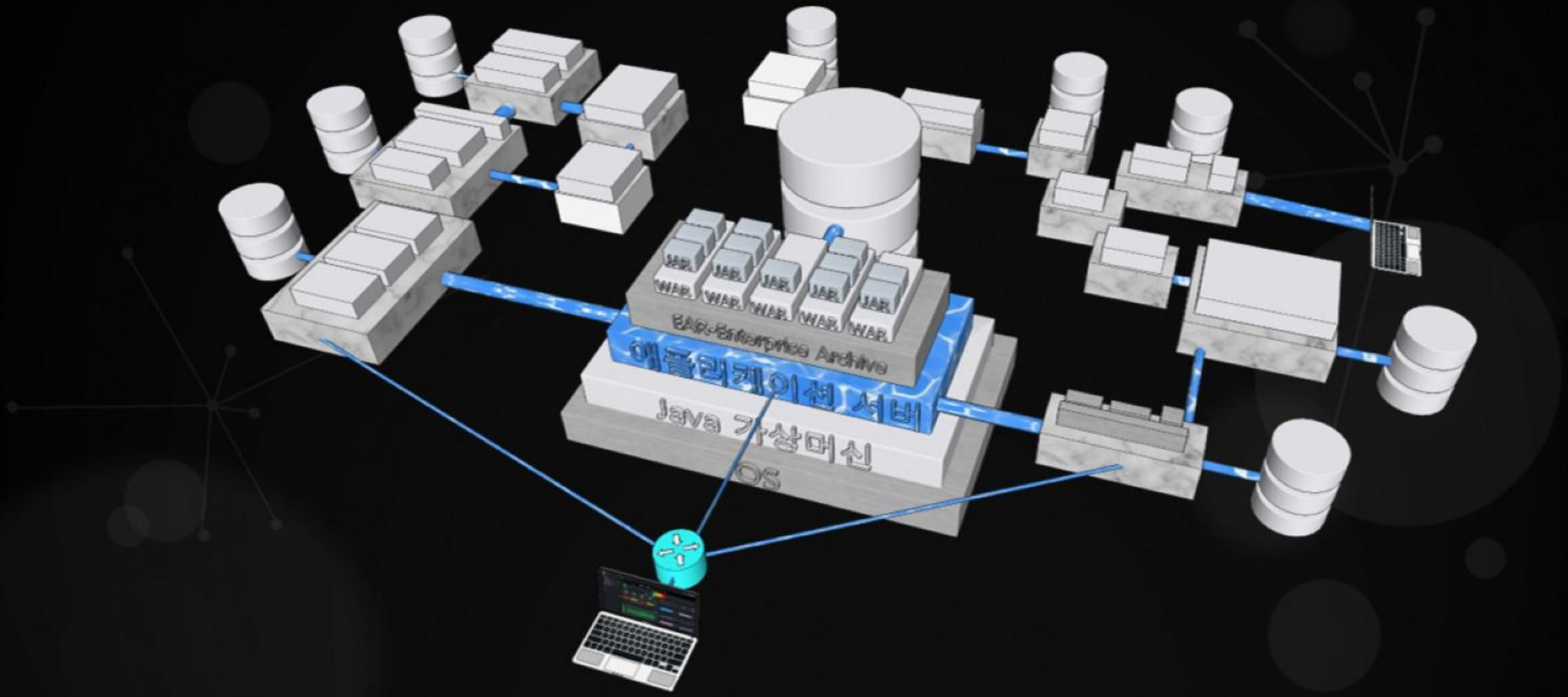


모노리스에서 마이크로서비스로 전환 – Phase 4

모노리스에서 마이크로서비스로 전환 – Phase 5

openmaru
오픈마루㈜

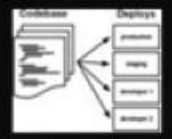
- 개선을 반복하여 최적의 마이크로 서비스를 제공



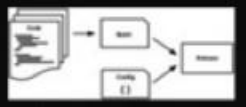
Cloud Native Application(애플리케이션 현대화) 도전!

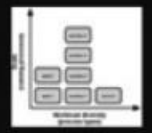
openmaru 오픈마루㈜

- 클라우드의 이점을 최대로 활용할 수 있도록 애플리케이션을 구축하고 실행하는 방식
- 신속한 개발과 클라우드 확장성 확보를 위한 클라우드 네이티브 애플리케이션 개발은 필수
  - SaaS 12-Factor, Cloud, MSA, Container, CI/CD 등 DevOps 중요
  - https://landscape.cncf.io/



12-Factor applications





Independent, autonomous, stateless, processes

Cloud Environment

NETFLIX

Configuration sources, service references

OpenJDK kafka

Microservices redis

CouchDB mongoDB.

Application as composition of polyglot services

CD.FOUNDATION Spinnaker

DevOps

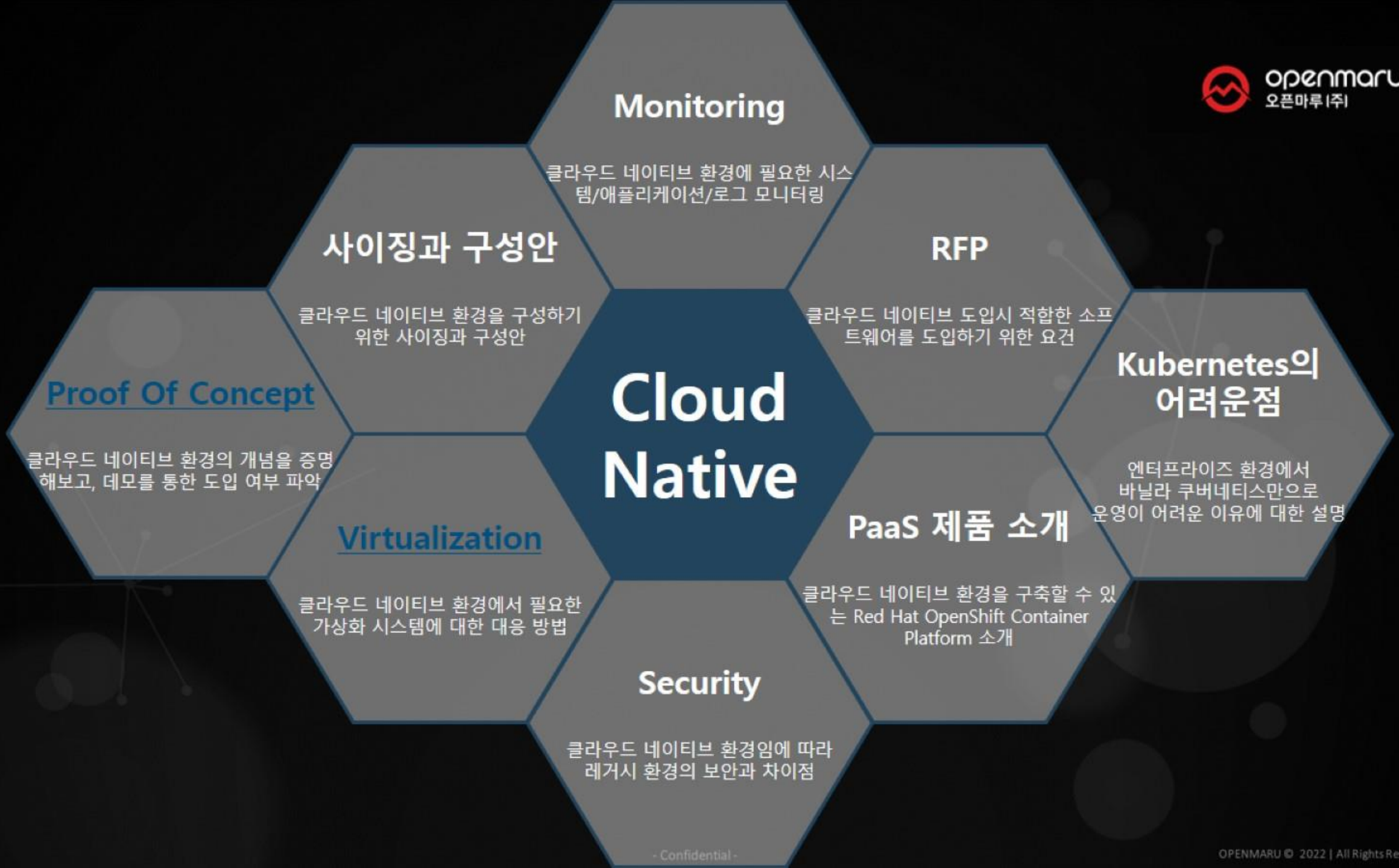
JENKINSX

Build, package, deploy, observe

기존 애플리케이션과 Cloud Native 애플리케이션 비교

openmaru
오픈마루(주)

분류	기존 애플리케이션	Cloud Native 애플리케이션
실행 환경	물리 서버 중심	컨테이너 중심
확장	Scale Up (수직 확장)	Scale Out (수평 확장)
결합	크고 조밀 결합	느슨하게 & 서비스 기반
인프라 의존성	인프라 의존	인프라 독립적으로 이동성 보장
Delivery 방법	폭포수형으로 장기간 개발	Agile & Continuous Delivery
개발 도구	로컬 IDE 개발 도구	클라우드 기반의 지능형 개발 도구
조직구조	사일로화 된 개발, 운영, 보안 팀	DevSecOps, NoOps 또는 협업



쿠버네티스, 정확한 이해 없이 성공적인 도입 어려워...

openmaru
오픈마루(주)

VM웨어가 매년 실시하고 있는 쿠버네티스 실태 조사 보고서에 의하면 지난해에 이어 올해도 많은 기업이 쿠버네티스를 채택하고 있다는 것을 증명했지만, 그 과정에서 여러 과제들도 안고 있는 것으로 나타났다.

https://www.datanet.co.kr/news/articleView.html?idxno=174878

- 검증된 기술지원 역량 고려
  - 핸들링할 역량이 충분한지 PoC를 통해서 검증
  - 대부분의 업체들이 고객 요구사항에 대응이 가능하다고 하지만, 정작 그렇지 못한 경우들도 많으니 주의

- 엔터프라이즈용 쿠버네티스 지원
  - 순수 커뮤니티를 이용해서 운영하는 것은 자체 고급인력 확보, 보안, 버그, 짧은 라이프사이클로 인한 업그레이드 이슈 등을 안고 같이 가야 하는 것과 마찬가지
  - 기술 자체의 이슈에 대한 지원을 받을 수 있어야만 그 기반의 문제에 대한 빠른 대응과 극복이 가능
  - 순수 쿠버네티스로는 도입 후 성공적으로 운영하기는 많은 어려움

KUBERNETES 를 제대로 운영하기 어려운 이유는?

openmaru
오픈마루㈜

- Kubernetes 훌륭한 기초 기술이지만, 스토리지, 네트워킹, 보안, 애플리케이션 프레임 워크 등을 통합하고 이를 분기별로 갱신하는 것은 큰 부담입니다

- Ashesh Badani, Red Hat

Why running your own Kubernetes deployment could be a terrible idea

Kubernetes is hard, but becomes doubly so when you take on the burden of supporting this fast-moving project.

By Matt Asay | June 21, 2018, 11:23 AM PST

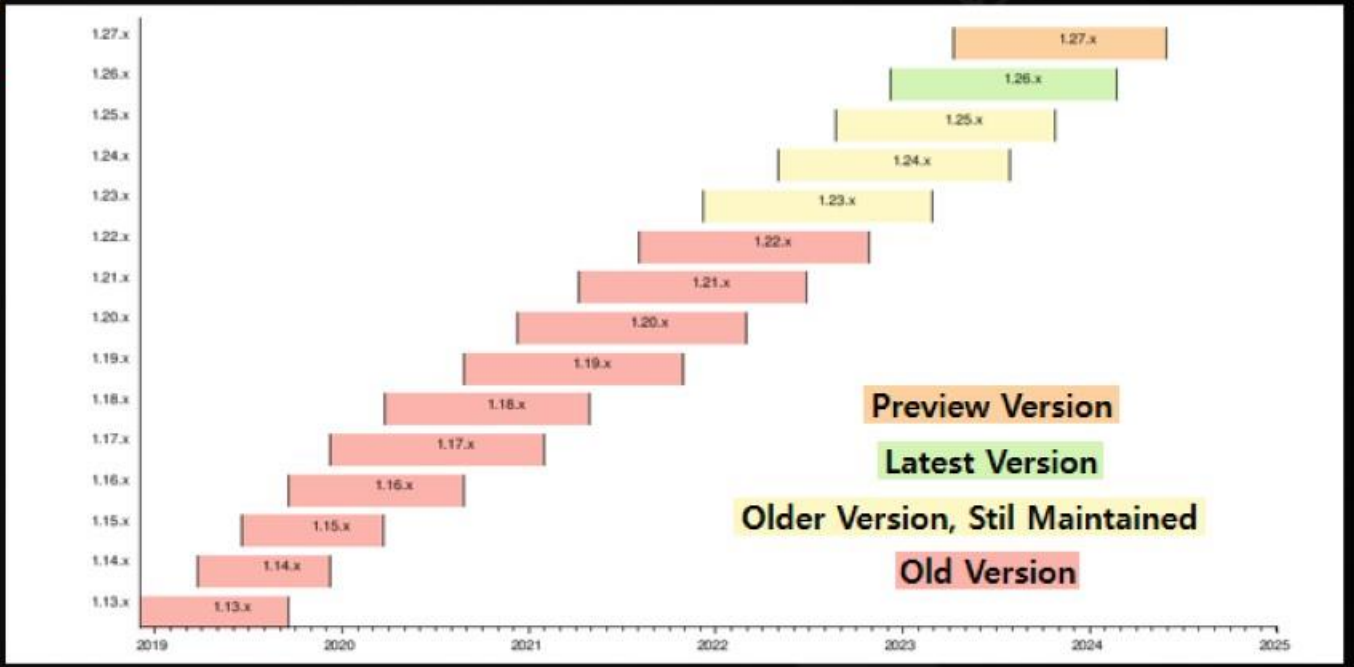
https://www.techrepublic.com/article/why-running-your-own-kubernetes-deployment-could-be-a-terrible-idea/

K8S의 업그레이드 주기

- Kubernetes의 경우 1년에 3~4 버전의 Major Release
- 2023년시점 21년 중반에 release한 1.22.x 버전 역시 maintain 종료
- 업그레이드 버전간 Dependency에 대한 명확한 설명이 부족함.

https://kubernetes.io/docs/tasks/administer-cluster/kubeadm/kubeadm-upgrade/

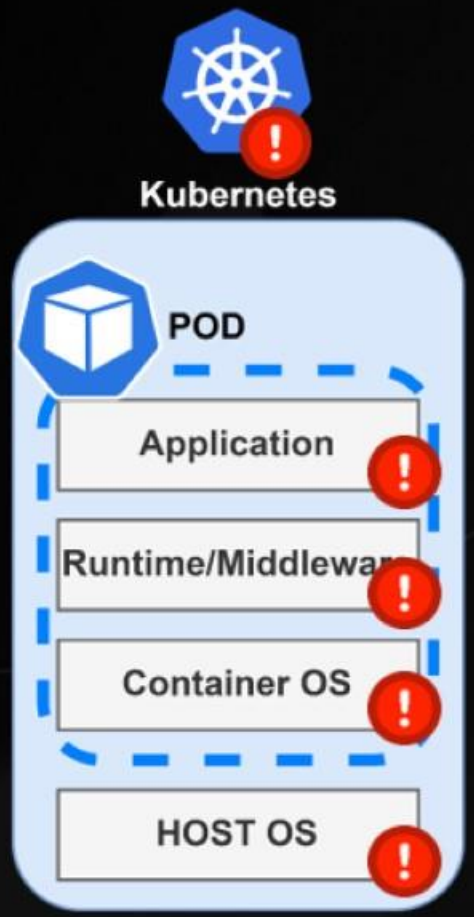
  - Make sure you read the release notes carefully.
  - The cluster should use...
  - Make sure to back up...
  - Swap must be disabled...



https://en.wikipedia.org/wiki/Kubernetes

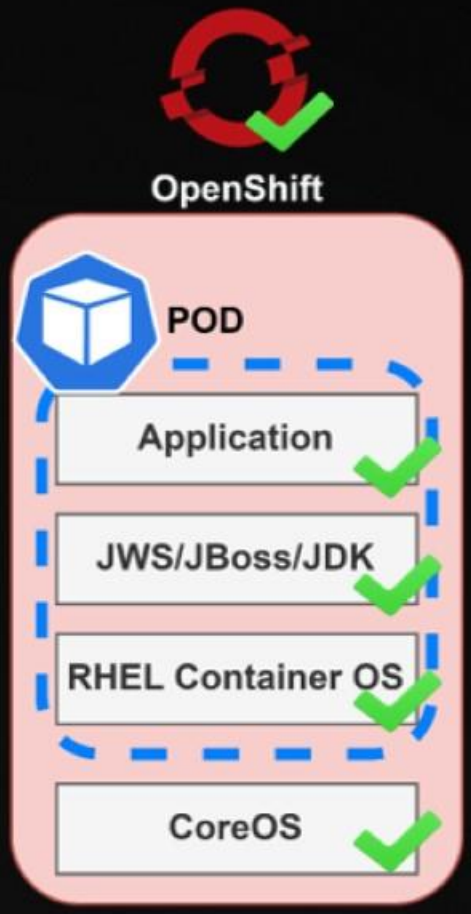
신뢰할 수 있는 Software 스택 제공

openmaru
오픈마루㈜



Kubernetes + DIY Stack

- 신뢰할 수 없는 컨테이너 이미지
  - 컨테이너 보안 업데이트 X
- QA팀을 통한 안정화 테스트 X
- HOST OS 유지보수
  - 업그레이드
  - 패치
  - 트러블슈팅
  - 구매비용 발생
- Runtime/Middleware 유지보수
  - 업그레이드
  - 패치
  - 트러블슈팅
  - 구매비용 발생



OpenShift Container Platform

- 신뢰할 수 있는 컨테이너 이미지
  - Quay, red hat registry
- QA팀을 통한 안정화 테스트
- Red Hat Core OS
  - 업그레이드 지원
  - 버그 패치 지원
  - 트러블슈팅 지원
  - 컨테이너 특화 OS
  - 구매비용 발생 X
- OpenJDK / JBoss Web Server, EAP
  - 업그레이드 지원
  - 패치 지원
  - 트러블슈팅 지원
  - 구매비용 발생 X
    (EAP의 경우 구매비용 발생)

(참고)Elasticsearch Log4j 보안취약점 이슈

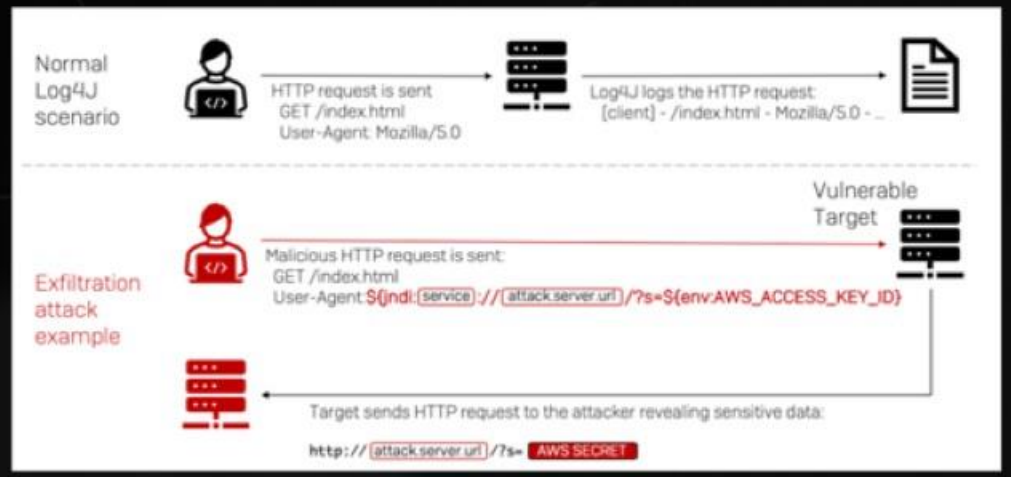
openmaru 오픈마루(주)

- 2021년 12월 9일 식별된 CVE-2021-44228 취약점

  RCE(Remote Code Execution)라고 일컬어지는 이 취약점은 공격자가 원격 서버에서 코드를 실행할 수 있게 합니다.

- Kubernetes의 자체적인 컴포넌트들은 대부분 Go언어이나 그 위에 동작하는 운영을 위한 시스템들은 JAVA와 같은 여러가지 언어로 작성된 프로그램

- 로깅 스택의 Elastic Search의 경우 CVE-2021-44228 취약점을 조치한 버전을 release 했으나, 그것을 운영중인 Kuberentes 클러스터에 적용시키는 것은 Kubernetes 클러스터 관리자의 몫



AWS환경에서 CVE-2021-44228 취약점을 활용한 공격 구성 (출처 : SOPHOS Labs)

Introducing 7.16.2 and 6.8.22 releases of Elasticsearch and Logstash to upgrade Apache Log4j2

By Tom Callahan, Quin Hoxie, Rajiv Raghunarayan

2021년 12월 19일

한국어

https://www.elastic.co/kr/blog/new-elasticsearch-and-logstash-releases-upgrade-apache-log4j2

Kubernetes 구축과 운영의 복잡성

openmaru
오픈마루㈜

Kubernetes 사용자 중 **75 %** 는

구축과 운영의 복잡성으로 도입하기 어렵다고 함

INSTALL	DEPLOY	HARDEN	OPERATE
• Templating	• Identity & Security Access	• Platform Monitoring & Alerts	• OS Upgrade 및 Patch
• Validation	• App Monitoring & Alerts	• Metering & Chargeback	• Platform Upgrade 및 Patch
• OS Setup	• Storage & Persistence	• Platform Security Hardening	• Image Upgrade 및 Patch
	• Egress, Ingress & Integration	• Image Hardening	• App Upgrade 및 Patch
	• Host Container Images	• Security Certifications	• Security Patches
	• Build / Deploy Methodology	• Network Policy	• Continuous Security Scanning
		• Disaster Recovery	• Multi-environment Rollout
		• Resource Segmentation	• Enterprise Container Registry
			• Cluster & App Elasticity
			• Monitor, Alert, Remediate
			• Log Aggregation

Source : The New Stack, The State of the Kubernetes Ecosystem, August 2017

기업에서 필요한 Kubernetes 기술지원

openmaru
오픈마루㈜

장기 수명주기

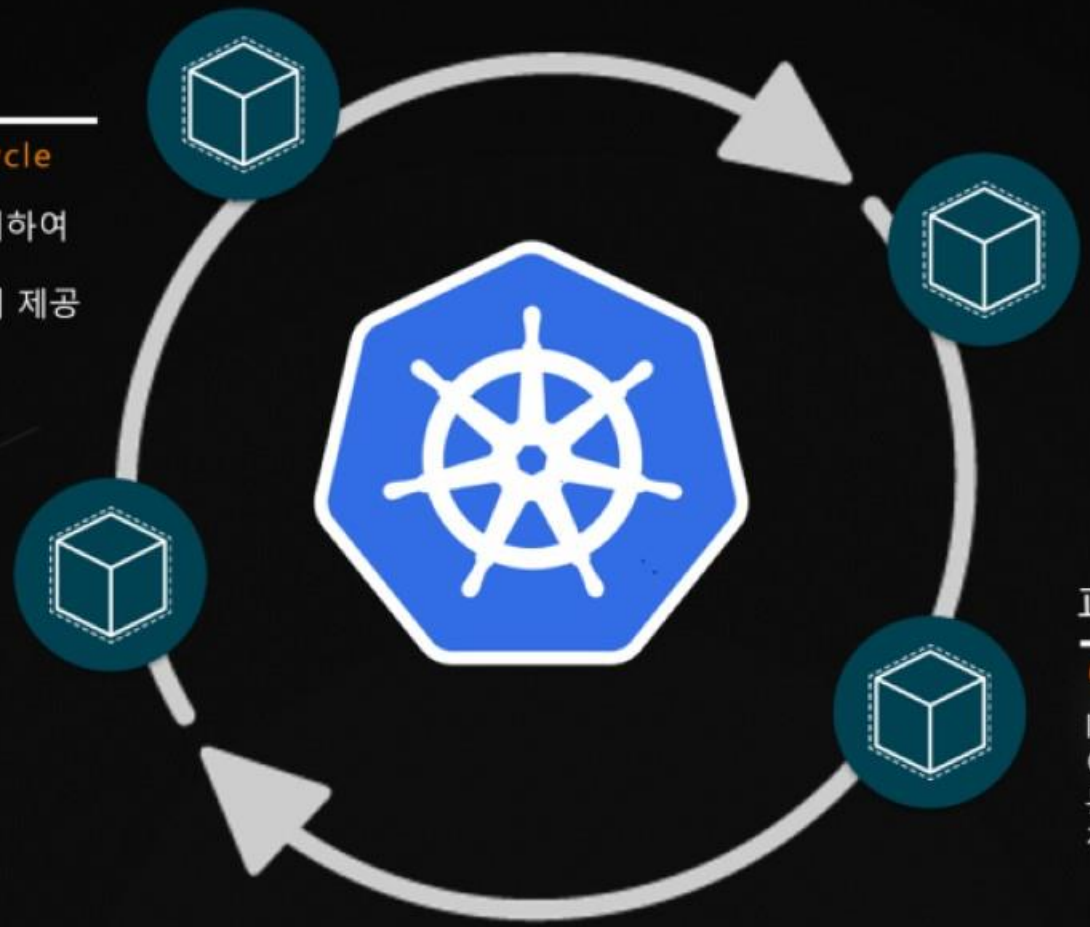
Predictable and long lifecycle

예측 가능한 라이프 사이클을 제시하여 운영하는 애플리케이션과 비즈니스에 필요한 장기 지원 체계 제공

SLA 와 기술 지원

SLA and support

장애에 대한 응답 및 복구 할 수 있는 기한에 따라 SLA 을 지원하며, 벤더를 통한 명확한 지원 체계 제공



교육

Training

장기적인 지원 뿐만 아니라 제품 교육과 자격증 제도를 제공하여 기술 내재화를 통한 서비스의 지속적인 운영을 지원

파트너 인증

Certification

ISV 가 제공하는 3rd party 소프트웨어를 포함하여 쿠버네티스 상에서
동작을 확인하고 비즈니스에 중요한 워크로드에 적합한 지 보장

Kubernetes를 구축할 때 고려해야할 것들

openmaru
오픈마루㈜

1. 보안

- 인증서 관리
- 컨테이너 이미지 신뢰성
- Runtime 신뢰성
- 플랫폼 계정관리
- Host OS 보안
- 컨테이너 보안

2. 운영/관리

- Web Console
- Cloud Native 3rd Party
- 서비스 라우팅
- 기술지원
- Cluster Upgrade
- Software Defined Network
- 모니터링

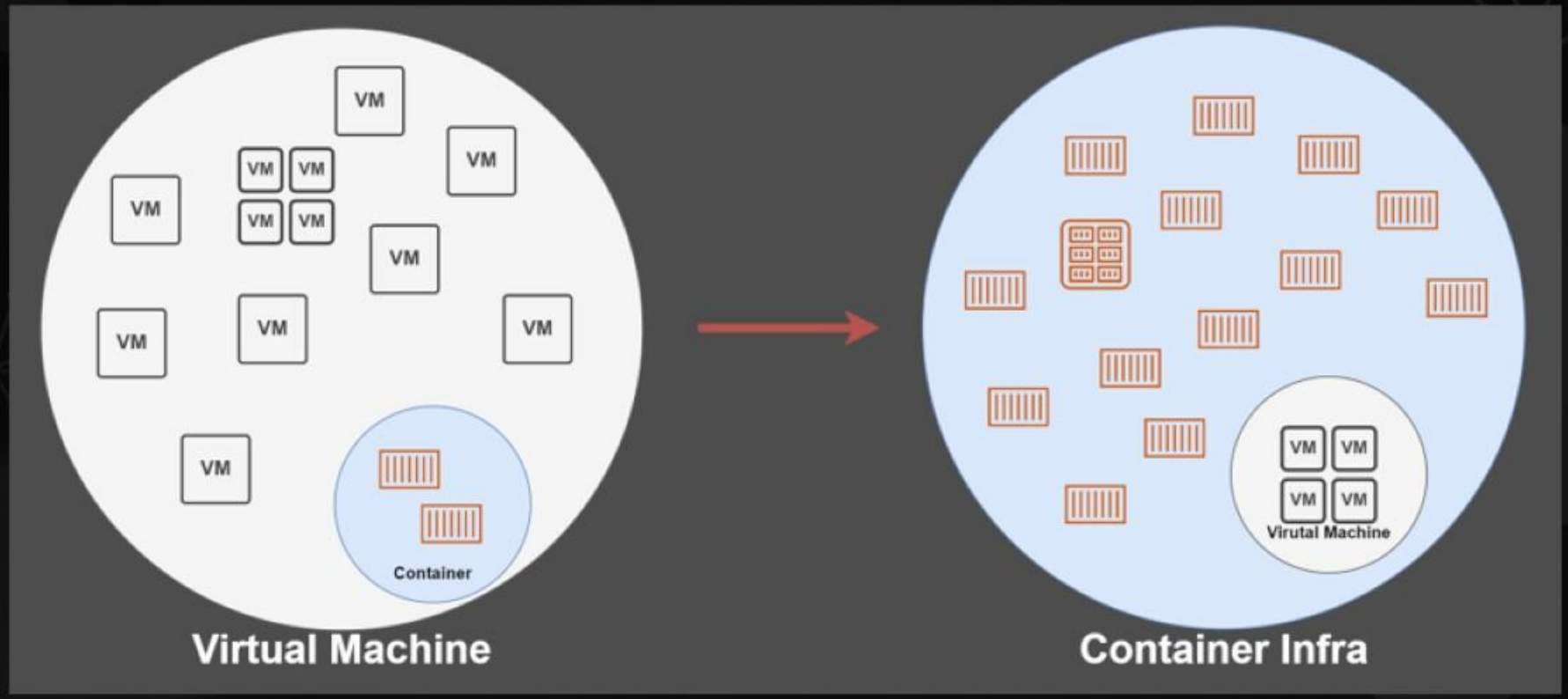
3. 애플리케이션

- 애플리케이션 컨테이너화
- 애플리케이션 로깅
- 애플리케이션 빌드배포
- 서비스메시
- 애플리케이션 모니터링

앞으로의 VM과 Container의 비중 변화

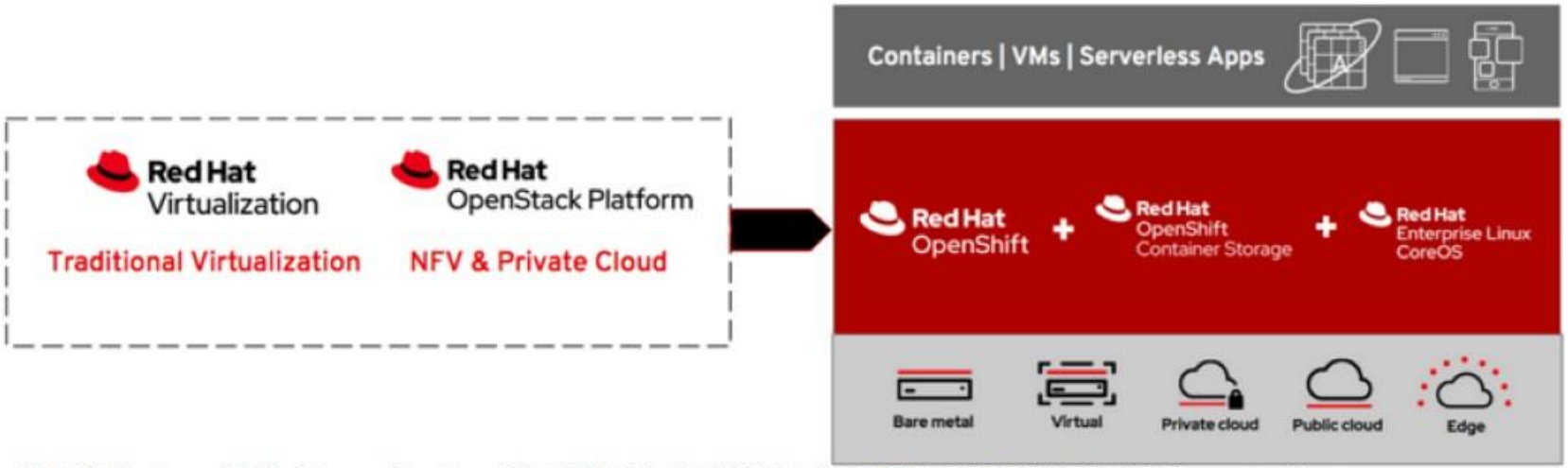
openmaru
오픈마루(주)

- 기존 환경에서 VM의 대다수는 애플리케이션을 기동 시키기 위한 서버
- Container Infra에서는 서비스 애플리케이션은 모두 Container로 기동!
- VM 비중이 축소됨에 따라 Container Infra에서 VM을 기동한다면? 기동하기 위한 솔루션은?



OpenShift Virtualization

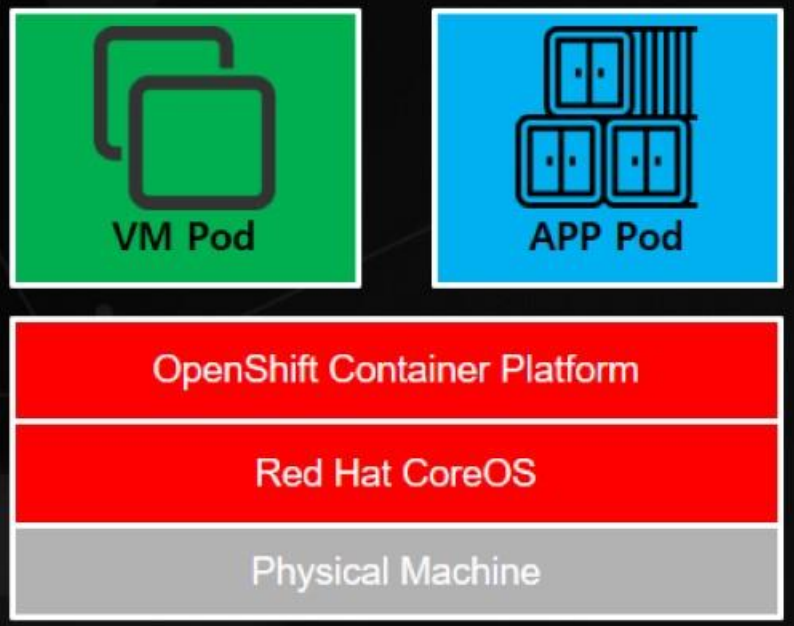
- OpenShift Virtualization을 통해 컨테이너 네이티브 아키텍처에서 가상 머신, 컨테이너를 통합 관리
- OpenShift에서 가상 머신과 컨테이너를 모두 사용함에 따라 애플리케이션을 컨테이너 네이티브에 적합하게 개발
- 컨테이너 기반 애플리케이션과 동일한 플랫폼에서 VM 기반 워크로드를 운영
- Windows 혹은 별도 솔루션 같은 Containerize가 어려운 자원들에 대한 지속 운영 가능



레드햇이 OpenShift Virtualization으로 제공할 가상화의 미래, 현재 전통적인 가상화 integration
https://cloud.redhat.com/blog/blog-openshift-virtualization-whats-new-with-virtualization-from-red-hat

OpenShift Virtualization

- Kubernetes 네이티브방식을 사용하여 가상 머신을 실행하고관리하기위해 KubeVirt를 기본으로 OpenShift에 구축된 가상화 API 및 런타임
- KubeVirt 프로젝트는 Red Hat의 주도하에 개발



- Kubernetes 시스템으로직접 컨테이너화할 수 없는 애플리케이션구성 요소를 전환하는방법을 제공
- VM 리소스를컨테이너네이티브로연결 및 사용
- 가상 머신은 기존 애플리케이션 컨테이너와 동일한 Red Hat OpenShift 노드에서 병렬로 실행
- VMware vSphere및 Red Hat Virtualization 가상 머신을 포함한 기존 가상 머신 가져 오기 및 복제 지원
- k8s 오케스트레이션, 관리를 통한 성숙하고 안정적인 KVM 기반 가상화

OpenShift Virtualization의 구매비용?

- OpenShift Virtualization은 별도의 제품이 아닌 기능
- OpenShift Container Platform과 OpenShift Container Engine에 번들된 기능
- OpenShift 운영시 별도의 서브스크립션, 제품 구매비용 발생하지 않음
- OCP에 Red Hat Enterprise Linux (RHEL) Virtual Datacenter Subscription이 포함되어 있기 때문에 RHEL Guest OS 역시 추가 금액이 발생하지 않음.

Question: Is OpenShift Virtualization a product?	**Answer:** OpenShift Virtualization is a feature, not a product. It is based on the upstream open source KubeVirt project and is available to download as a Red Hat OpenShift operator. More information on how to get and install the OpenShift Virtualization operator can be found in the OpenShift Virtualization documentation.
Question: How will OpenShift Virtualization be made available?	**Answer:** OpenShift Virtualization is a feature of Red Hat OpenShift Container Platform and Red Hat OpenShift Kubernetes Engine. It is not an add-on or a separate product. The OpenShift Virtualization operator must be installed to access the feature. All current and future subscribers receive OpenShift Virtualization as part of their Red Hat OpenShift subscription. OpenShift Virtualization is a feature of Red Hat OpenShift Container Platform and Red Hat OpenShift Kubernetes Engine. It is not an add-on or a separate product. The OpenShift Virtualization operator must be installed to access the feature. All current and future subscribers receive OpenShift Virtualization as part of their Red Hat OpenShift subscription.

https://www.redhat.com/en/resources/openshift-virtualization-faq

Virtual Machines · Red Hat Ope

주의 요함 | https://console-openshift-console.apps.ocp4.ov.com/k8s/ns/comodo/kubevirt.io~v1~VirtualMachine

openmaru APM

Red Hat OpenShift Container Platform

31 admin

Administrator

Home

Operators

Workloads

Virtualization

Overview

VirtualMachines

Templates

Networking

Storage

Builds

Observe

Compute

User Management

Administration

Project: comodo

Virtual Machines

Launch Migration Tool | Create

With Wizard

With YAML

No virtual machines found

See the templates tab to quickly create a virtual machine from the available templates.

Create virtual machine

Learn how to use virtual machines

OpenShift내에 가상머신 생성

console-openshift-console.apps.ocp4.ov.com/k8s/ns/comodo/.../wizard?namespace=como...

클라우드 네이티브 정보시스템 구축 사업 추진 방향성 작성 예시

openmaru
오픈마루㈜

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

→ 정부지식공유활동기반 고도화 사업의 아키텍처 모델은 G-클라우드 가상화 기반으로 구축되며, G-클라우드 서비스와 호환되는 **오픈 소스 소프트웨어로 구성된 PaaS 솔루션을 활용**하여 서비스의 특성에 최적화된 클라우드 가상인프라 플랫폼을 설계, 구축

정부 지식 공유 활동 기반 고도화 사업 아키텍처 설계서 일부 발췌

→ **MSA 기반**의 **컨테이너 형태**로 구현된 공간정보서비스 기능(공간정보표준 프레임워크)을 효율적으로 운영·관리하기 위한 개방형공간정보 플랫폼 구축 - 서비스 수요 증감에 따라유 연하게 컨테이너가 확장 및 축소가 가능한 운영관리 기능 및 컨테이너 동작 여부에 대한 상태 모니터링 기능 제공

디지털 광자원 통합관리시스템 재구축 및 운영 제안요청서, 한국관광공사

클라우드 네이티브 기반 PaaS의 상세 RFP

openmaru
오픈마루㈜

- 제안 제품에 대한 소스코드 전체가 공개되고 다운로드 가능한 PaaS
- 특정 벤더에 종속되지 않은 오픈소스 기반의 소프트웨어 제품으로, 기술지원이 가능한 엔터프라이즈용 PaaS 기술 적용
- 표준 컨테이너(Containerd, CRI-O) 기반의 컨테이너 관리 기술 적용
- CNCF(Cloud Native Compute Foundation)의 CK(Certified Kubernetes) 인증을 획득 유지중인 제품
- 다양한 S/W 벤더들과 Platform간 상호 호환성이 확보된 자체의 표준 이미지 저장소 제공
- 다양한 S/W, H/W 벤더들과 Platform 간 상호 호환성이 확보된 Operator 저장소 제공
- 안정적인 컨테이너 서비스 제공을 위해 기술지원 가능한 엔터프라이즈용 Linux OS 기본 제공 (커뮤니티 운영체제 배제-CentOS, Fedora, Ubuntu 등)
- 엔터프라이즈에서 주로 사용하는 Java(Open JDK)에 대한 기술지원을 포함하여 제공
- 개별 소스 반영시 CI/CD를 위한 자동 컨테이너 빌드 메커니즘 제공
- 플랫폼과 통합된 다양한 CI/CD 파이프라인 도구 제공 (Jenkins, Tekton, ArgoCD)
- MSA(마이크로서비스)를 위한 Istio 기반의 기능 및 기술지원 제공
- VM 서비스를 위한 KubeVirt 기반의 가상화 기능 및 기술지원 제공
- PaaS가 설치되는 Host OS는 PaaS의 S/W Appliance 형태로 설계 및 구성되어있으며 변경 불가능한(immutable) 방식으로 관리되어야 함.

아직도 WAS BMT 나 POC를 하시나요?

openmaru 오픈마루㈜

- WAS 의 가용성/확장성/성능/신뢰성 등의 기능은 플랫폼으로 이전
- 더 가볍고 더 빠르고, 자동화에 친화적인 WAS 로 전환

WAS 제품 BMT

ORACLE WEBLOGIC

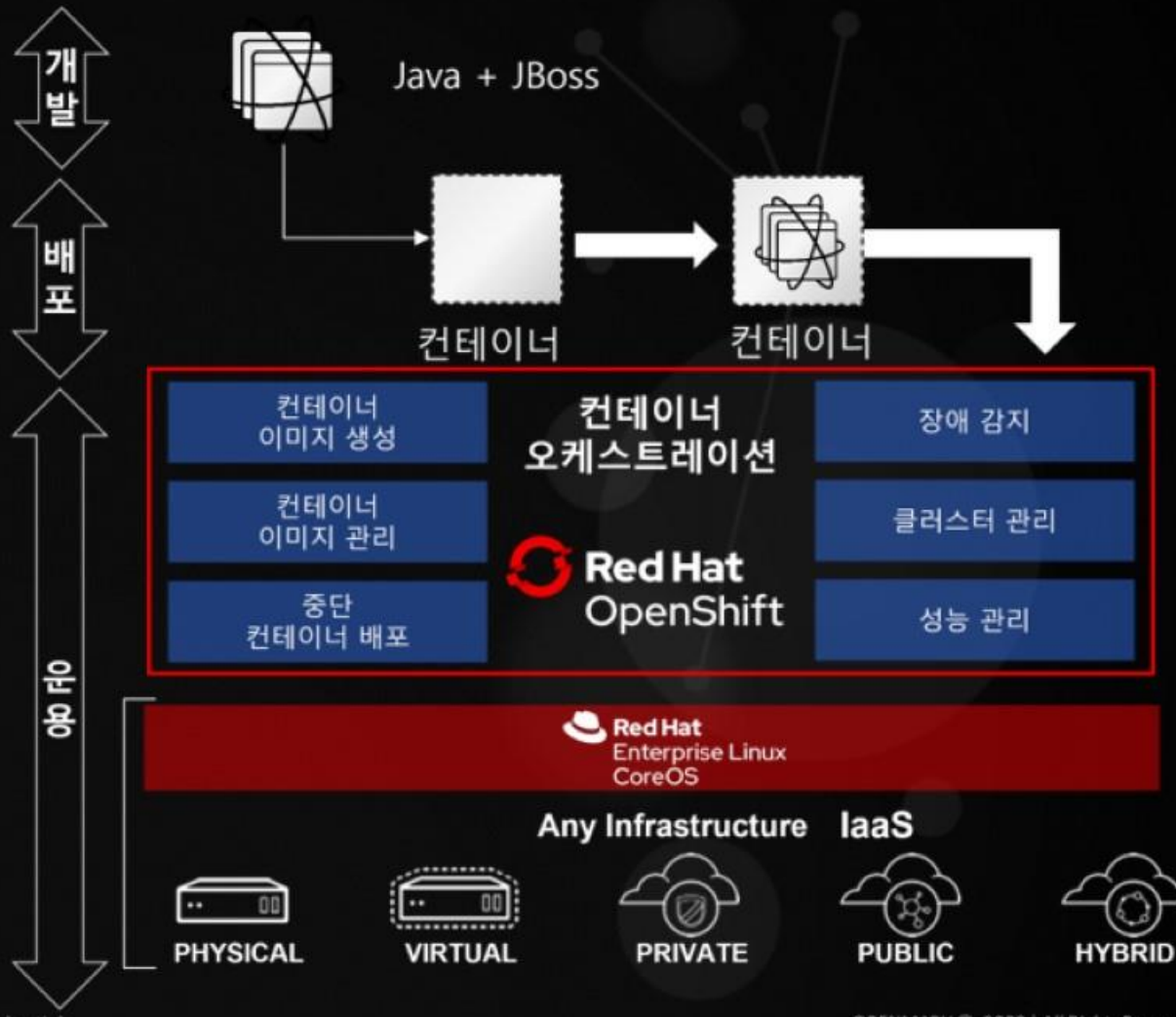
JBoss by Red Hat

WebSphere

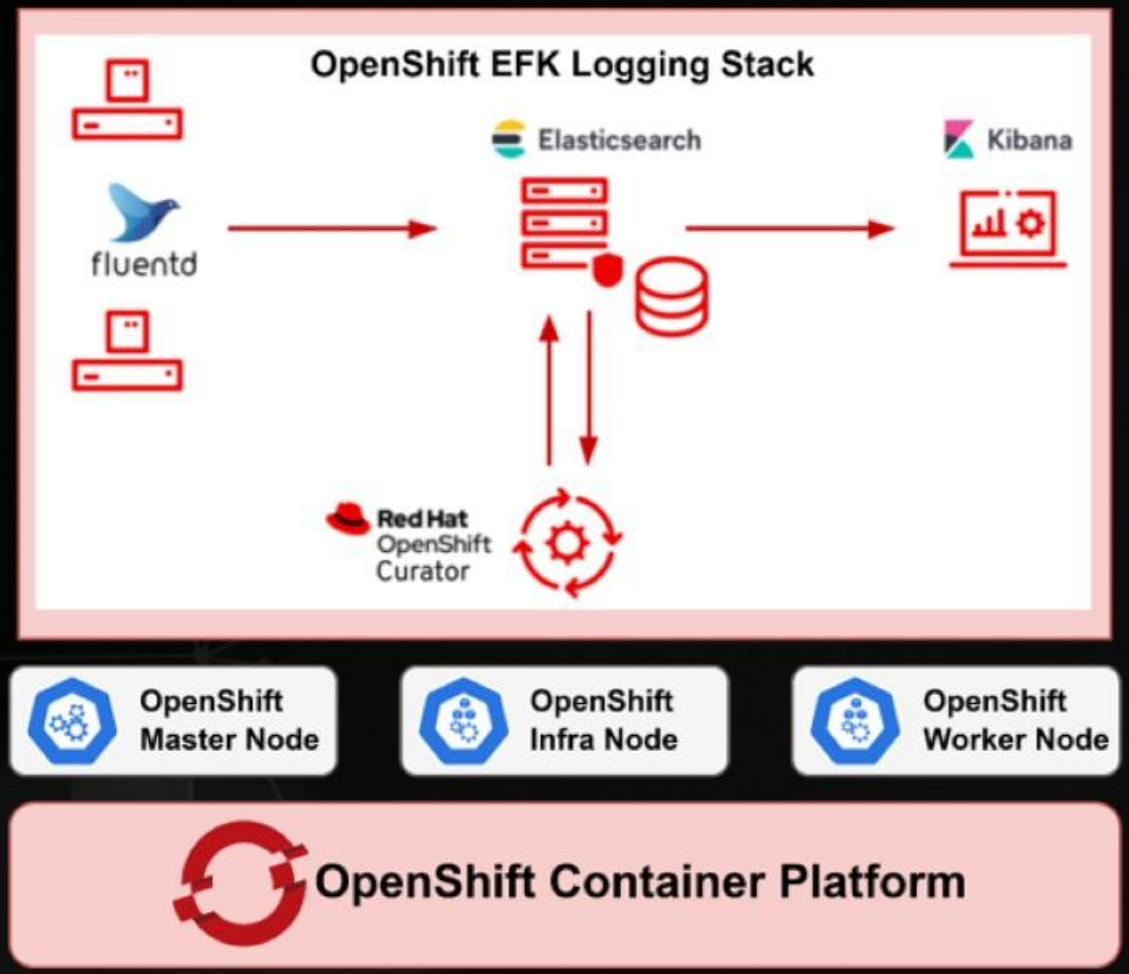
Tmax JEUS

RASP 에 의한 WAS평가

- 신뢰성 (Reliability):
  - 과부하 테스트
- 가용성 (Availability):
  - 일부 장애 발생시 전체 시스템 영향 최소화
- 확장성 (Scalability):
  - 자원 추가에 따른 선형적인 성능 개선
- 성능 (Performance):
  - TPS, 응답시간 등 평가
- 보안 기능 (Security)
  - (RBAC 등)
- WAS 도구 평가 (Manageability)
  - Admin Console



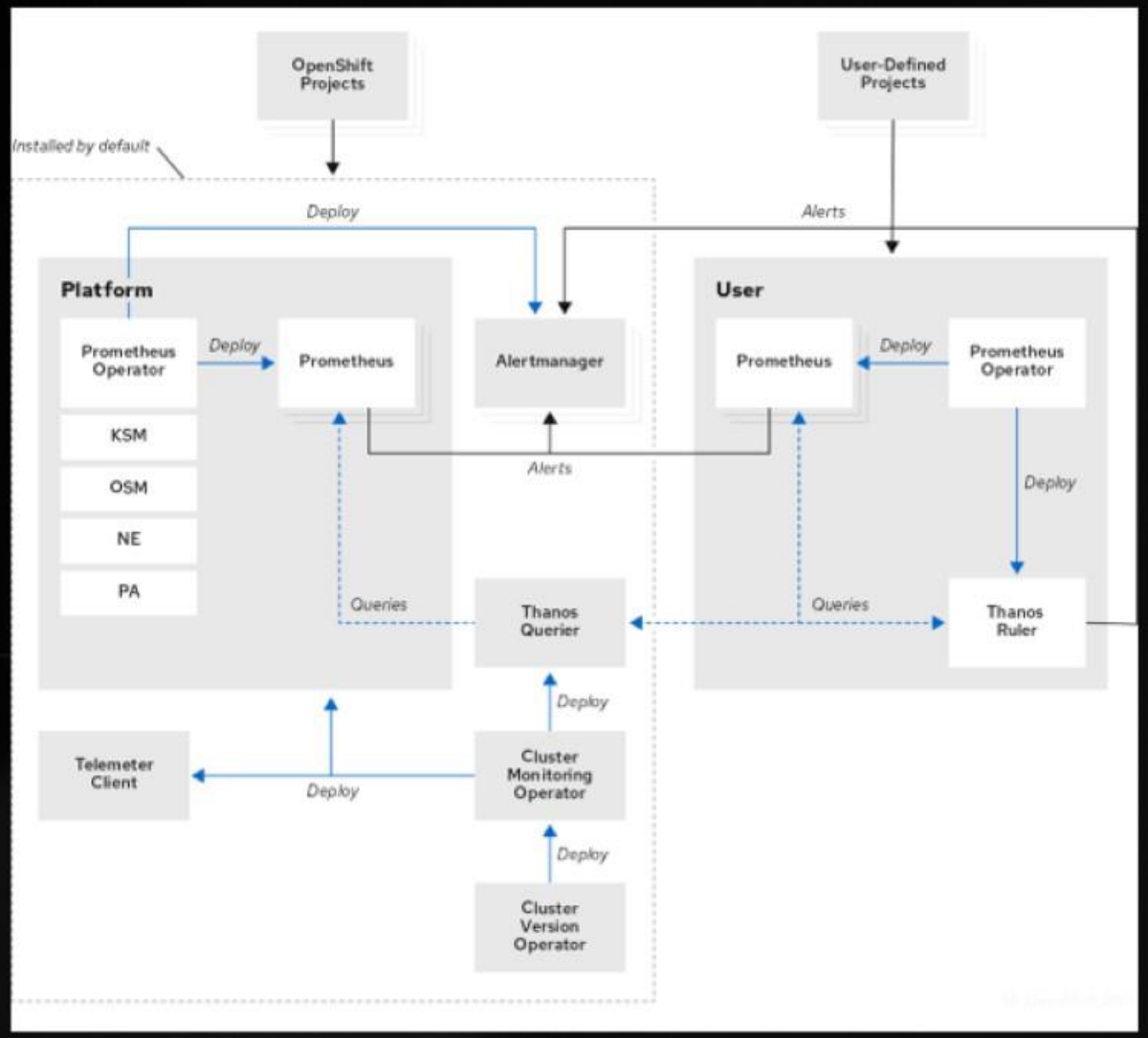
Cloud Native Logging Stack, EFK



Elasticsearch + Fluentd + Kibana Logging Stack

- 컨테이너 인프라 환경, Cloud Native 환경 같은 경우 로그 집중화 스택이 필요함
  - 수 많은 Container 기동
  - Container 삭제시 로그데이터 삭제
- Cloud Native에 적합한 로그 스택, EFK
  - Elasticsearch : 로그 데이터 저장소
  - Fluentd : 로그 데이터 전송
    (logstash 대체하는 경우도 있음)
  - Kibana : 로그 데이터 시각화
  - Curator : 로그 메타데이터 rotate
- K8S 운영시 자체적으로 클러스터에 통합해야함.

시스템 메트릭 모니터링



- **OpenShift에는 Default로 시스템 메트릭 정보들을 수집하는 Prometheus가 포함**
  - 필요에 따라 사용자가 정의한 프로젝트에 대해서 별도의 시스템 모니터링도 가능
- **수집된 메트릭 정보를 OpenShift Web Console로 확인**
  - **Administrator perspective**
    - API performance
    - etcd
    - Kubernetes compute resources
    - Kubernetes network resources
    - Prometheus
    - USE method dashboards relating to cluster and node performance
  - **Developer perspective**
    - CPU usage
    - Memory usage
    - Bandwidth information
    - Packet rate information

클라우드 네이티브 환경의 애플리케이션 모니터링 어려움

openmaru
오픈마루㈜

6. Monitor, log and troubleshoot from the start

The construction of applications from a set of microservice Legos considerably complicates how to monitor and troubleshoot systems and their performance. Various microservices often trigger a cascade of events that leads to an application failure. To minimize failures -- which aren't a *maybe*, but a reality -- incorporate monitoring and troubleshooting into microservices design.

https://www.techtarget.com/searchitoperations/tip/Follow-these-6-steps-to-deploy-microservices-in-production

- 마이크로 서비스 아키텍처일수록 **모니터링 방법이 상당히 복잡**해 집니다.
- 마이크로 서비스 아키텍처에 **모니터링과 트러블 슈팅 방법이 고려**되어야 합니다.

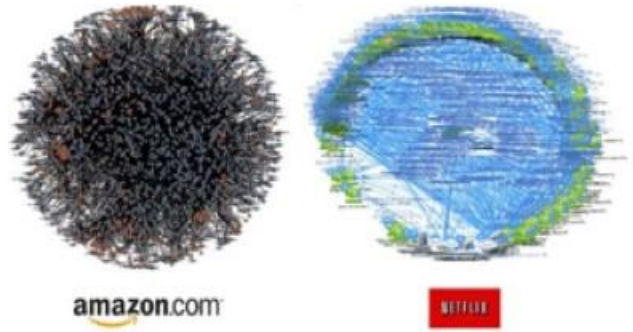
- Uber는 2014년 말 에 4,000개가 넘는 독점 마이크로 서비스와 점점 더 많은 수의 오픈 소스 시스템이 모니터링 시스템에 문제를 제기했다고 보고
- 컨테이너 인프라는 모니터링 시스템이 필수적인 환경
- **마이크로 서비스에 특화된 모니터링 시스템이 필요**

2. Microservices instead of a monolith

Following a microservice architecture, a typical monolith application would be broken down into a dozen or more microservices, each one potentially running its own programming language and database, each one independently deployed, scaled and upgraded.

Uber for example reported in late 2014 over 4,000 proprietary microservices and a growing number of open source systems which posed a challenge for their monitoring system.

The Challenge: A surge in the number of discrete components you need to monitor.

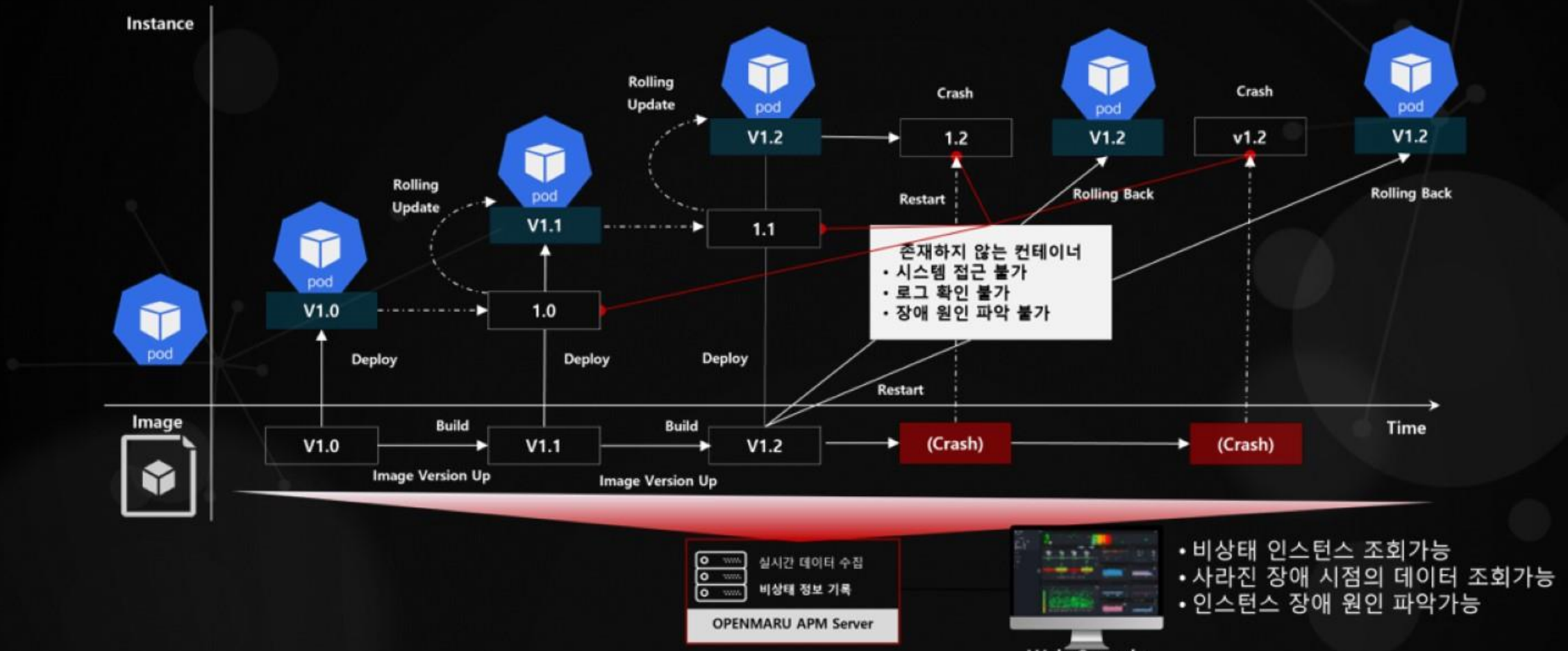


https://www.infoq.com/articles/microservice-monitoring-right-way

컨테이너는 무상태 (Immutable) 인프라스트럭처로 상태를 저장하지 않음

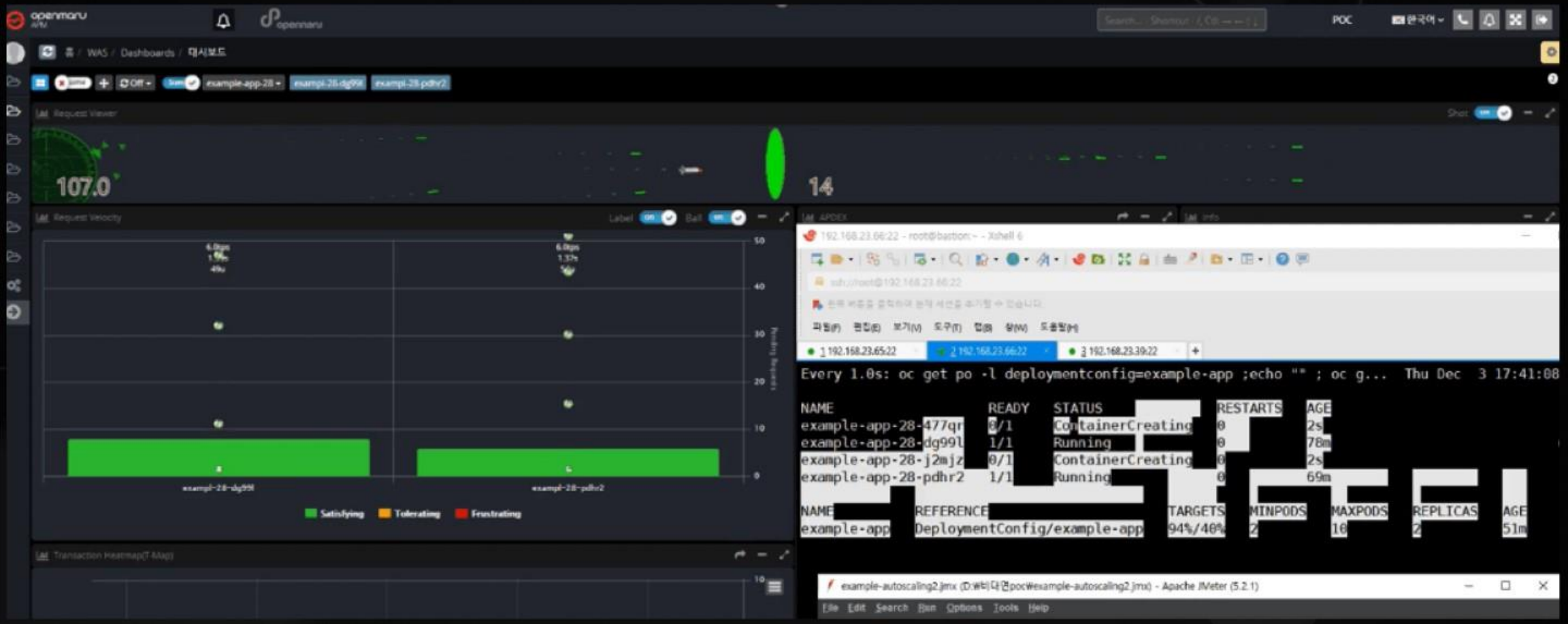
openmaru 오픈마루(주)

- PaaS 환경에서 컨테이너가 중지된 후 장애원인 파악을 위한 방법을 제공
- APM 에서 사라진 컨테이너에 대한 정보를 보관하여 장애원인을 파악할 수 있음



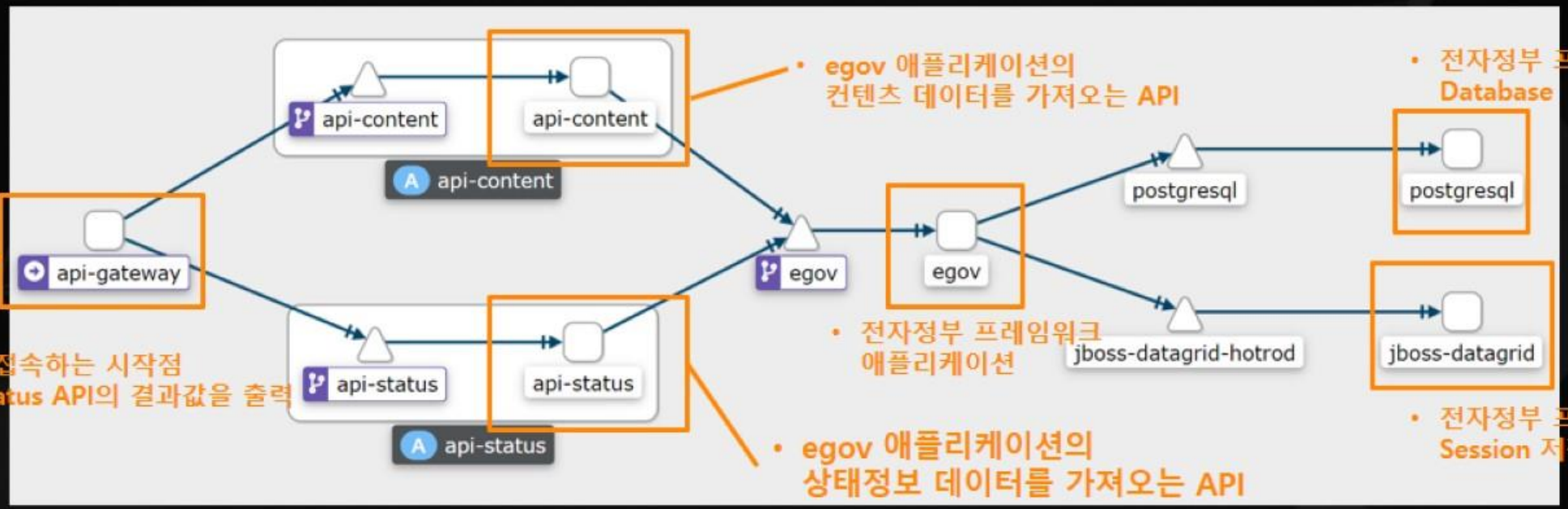
컨테이너 환경에서의 애플리케이션 모니터링

- **Auto Scaling으로 Container(Pod)가 유연하게 Scale Out/In이 발생하는 환경**
- **클라우드 네이티브 애플리케이션을 트러블 슈팅할 수 있는 기능이 있는 모니터링 시스템이 필요함**



MSA 모니터링 애플리케이션 구성도

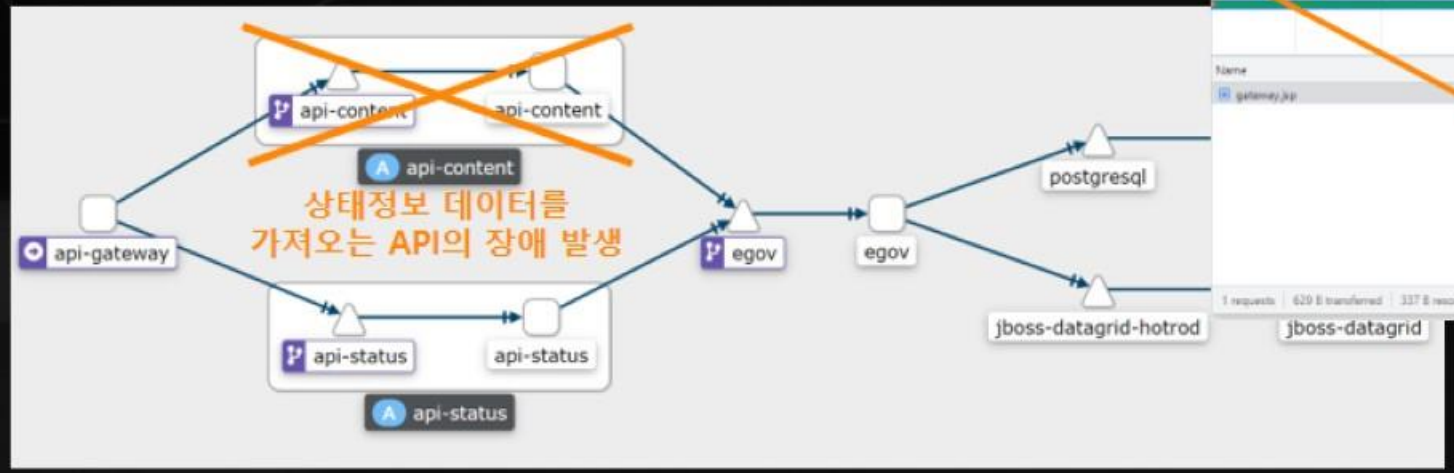
- 구성에 포함된 애플리케이션 총 4개
  - gateway, content, status, egov
- egov 애플리케이션을 기동시키기 위한 기타 소프트웨어
  - Postgresql : Database
  - jboss-datagrid : 세션 저장을 위한 데이터 그리드, OPENMARU CLUSTER 포함

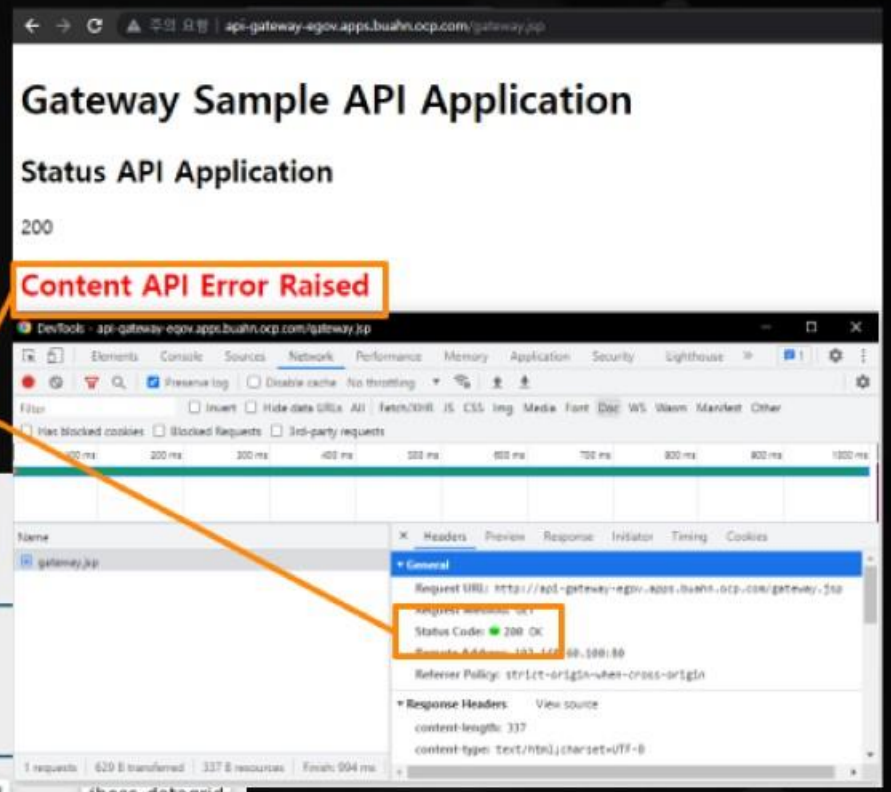


일부 애플리케이션 장애 발생

- MSA 환경의 특성상(느슨한 결합) 일부 서비스의 장애가 발생했을 때 식별이 어려움
  1. gateway를 접속하더라도 정상 페이지(status 200) 출력
  2. Content API로 받아오는 데이터는 에러 발생
- MSA에서 장애가 발생한 애플리케이션을 식별 및 분석에 어려움

Content API에서는 장애가 발생했지만
Gateway 애플리케이션은 200 상태코드 반환

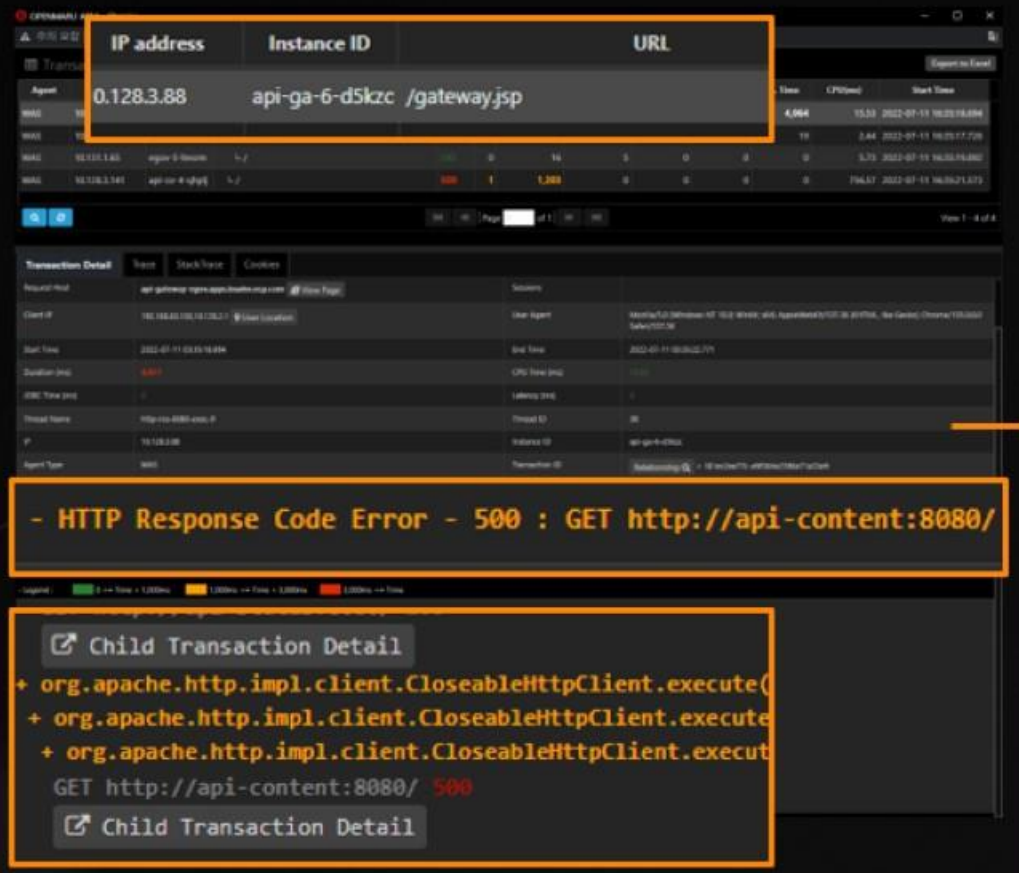


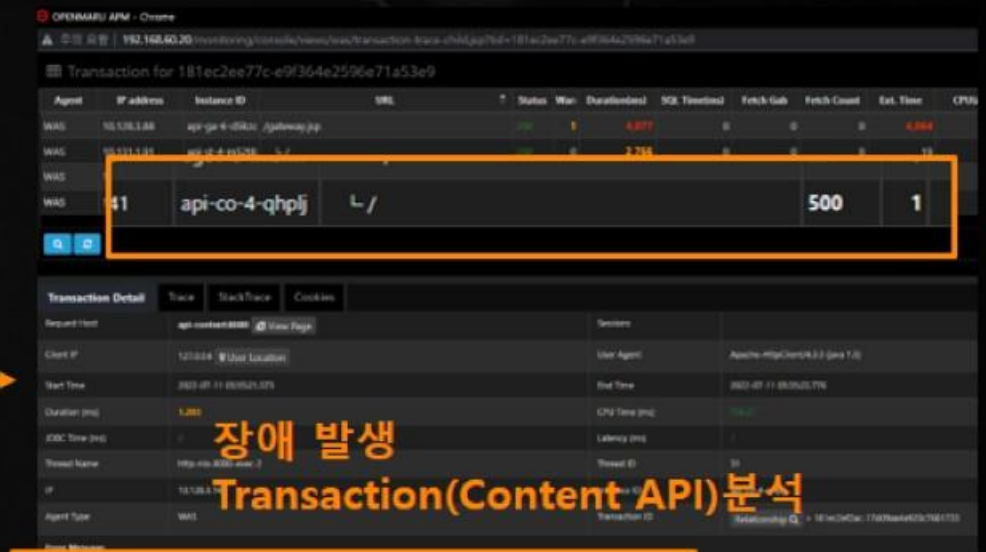


장애 발생 애플리케이션의 분석 절차

openmaru
오픈마루㈜

장애 발생
트랜잭션을 드래그





장애 발생
Transaction(Content API)분석

```
    Unable to compile class for JSP:

An error occurred at line: [21] in the jsp file: [/index.jsp]
testtesttest cannot be resolved to a type
18:     } catch ( Exception e ) {
19:         e.printStackTrace();
20:     }
21:     testtesttest
22:     %>
23:     < h2>Content API Application< /h2>
24:     < %
```

레거시 환경에서 필요한 서버 보안

- OS 보안, 서버 접근제어 등 보안 5종 S/W를 OS 설치
  - OS 마다 설치하기 때문에 물리서버는 1Copy이고, 가상화는 VM 개수 만큼 설치
- 서비스에 따라 부가적인 보안 소프트웨어 필요(개인정보 검출, 비정형 암호화 등)
- 법적인 근거로 인해 서버보안 – 전자금융법, ISMS 등의 요건

컨테이너를 기동시키기 위한 CoreOS

Immutable Infra Structure, Red Hat CoreOS

기존 OS에 필요한 Host OS 보안 프로그램

- 취약점 스캐너
- 서버접근제어
- OS 보안
- 백업 Agent
- 로그 수집
- VM 혹은 Baremetal 대수 만큼 필요

Red Hat CoreOS는 Immutable Infrastructure

- RHCOS는 플랫폼에 포함된 Compliance 시스템
  - 1금융권 사례
- Immutable Infrastructure 특성으로 Read-Only OS

컨테이너 환경에서 필요한 보안

openmaru
오픈마루(주)

왜 AS-IS 환경의 보안들이 필요하지 않을까?

기존의 보안 소프트웨어가
필요하지 않은 환경

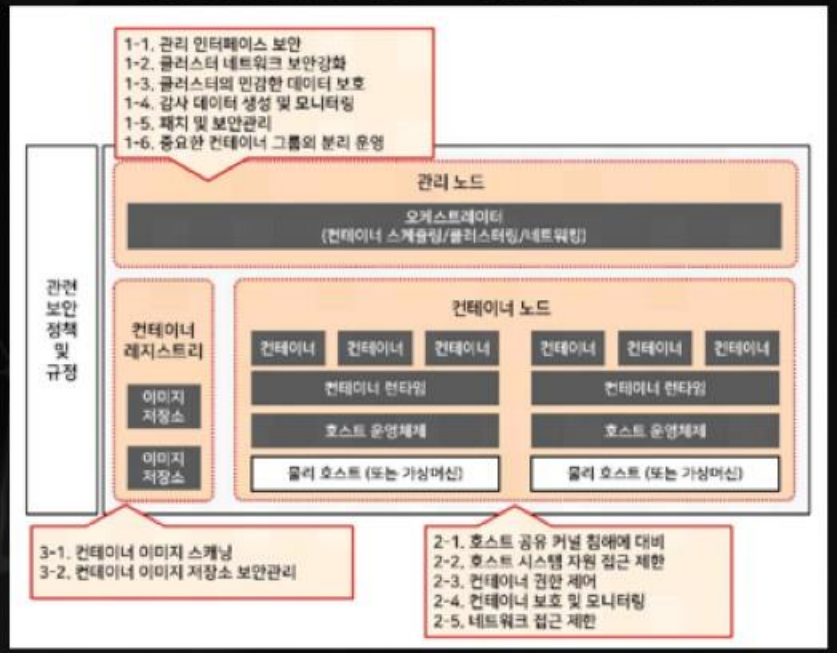
실질적으로 컨테이너환경에서 필요한 보안들

- 컨테이너 실행 권한에 대한 보안 : SCC
  - Container breakout
- 취약한 Container Image 사용
  - 악성코드, 채굴 프로그램이 포함된 이미지
- Role Base Access Controle : RBAC
  - 사용자별 필요 권한 부여
- 컨테이너 플랫폼의 Audit 로깅
  - EFK
- 신뢰할 수 있는 컨테이너 런타임 보안
  - Capabilities, SELinux, Seccomp & Namespace
- 컨테이너간의 네트워크 격리
  - Network Policy

국정원의 컨테이너 기술 보안 기준

- 컨테이너 기반 정부 업무 인프라의 안정성과 신뢰성을 제고하기 위한 보안기준
- 보안 기준은 관리노드, 컨테이너 노드 및 컨테이너 레지스트리로 분류
- 각급 기관은 컨테이너 기반의 업무 서비스 도입 및 운영 시 해당 보안 기준 내용 준수

국정원 컨테이너 기술 스택에 대한 보안 기준



국정원 컨테이너 기술 점검항목

5. 컨테이너 가상화 기술 보안기준 및 점검항목

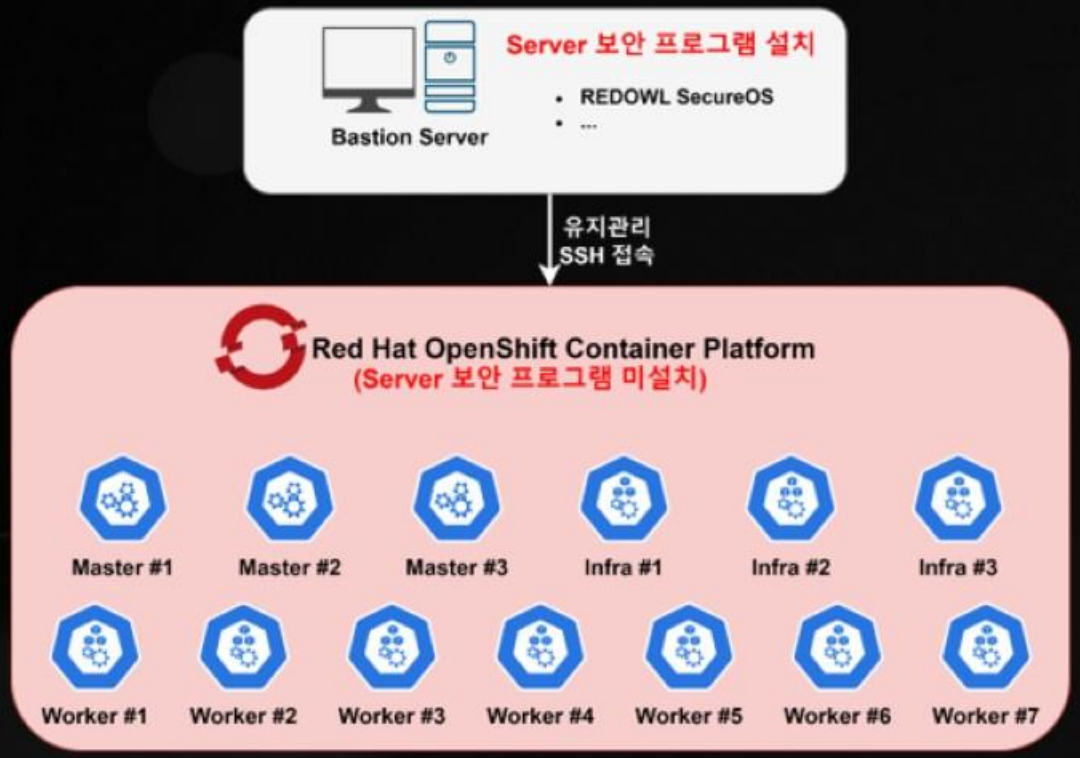
○ 관리 노드

점검항목	중요도	점검결과
1-1. 관리 인터페이스에 대한 인증과 접근 제어를 강화해야 한다.		
(1) 관리자 계정 접근에 대해 다중요소(Multi-Factor) 인증을 사용하는가?	필수	
(2) 네트워크를 통한 관리 인터페이스 접속 시에는 암호화 프로토콜을 사용하고 있는가?	필수	
(3) RBAC(역할 기반 접근 제어, Role Based Access Control) 정책을 적용하고 있는가?	권고	
1-2. 클러스터의 네트워크 보안을 강화해야 한다.		
(1) 클러스터 관리 구성요소에 대한 불필요한 네트워크 접근을 차단하고 있는가?	필수	
(2) 클러스터 네트워크의 통신 채널을 암호화하고 있는가?	필수	
1-3. 클러스터의 민감한 데이터를 보호해야 한다.		
(1) 클러스터의 민감한 정보를 안전한 위치에 보관하고 인가되지 않은 사용자가 접근하지 못하도록 설정하고 있는가?	필수	
(2) 민감한 정보를 암호화하여 저장하고 있는가?	필수	
1-4. 감사 데이터를 생성하고 모니터링 해야 한다.		
(1) 컨테이너 운영 환경의 감사 데이터를 생성하고 모니터링하고 있는가?	필수	
(2) 감사 데이터의 임계치를 설정하고 모니터링 하고 있는가?	권고	
1-5. 주기적으로 보안 패치와 설정 점검을 수행해야 한다.		
(1) 오케스트레이터 취약점에 대한 보안 패치를 수행하고 있는가?	필수	
(2) 주기적으로 취약한 설정을 점검하고 관리하고 있는가?	필수	
1-6. 중요한 컨테이너 그룹을 식별하고, 분리된 컨테이너 자원을 할당하고 있는가?		
(1) 중요도가 높은 서비스를 식별하고, 분리된 컨테이너 자원을 할당하고 있는가?	필수	

○ 컨테이너 노드

점검항목	중요도	점검결과
2-1. 호스트 운영체제 커널의 침해에 대비해야 한다.		
(1) 가상머신 위에 호스트 운영체제와 컨테이너 엔진(런타임)을 설치하고 컨테이너를 운영하고 있는가?	필수	
(2) 컨테이너 탈출 등에 영향을 줄 수 있는 취약점을 식별하고 보안 패치를 수행하고 있는가?	필수	
2-2. 호스트 시스템 자원에 대한 접근을 제한해야 한다.		
(1) 컴퓨팅 자원이 고갈되지 않도록 컨테이너에 대한 자원 할당량을 제한하고 있는가?	필수	
(2) I/O 장치 자원이 고갈되지 않도록 컨테이너별 대역폭을 제한하고 있는가?	필수	
(3) 호스트 운영체제에 대한 컨테이너의 접근을 제한하고 있는가?	필수	
(4) 컨테이너의 호스트 네트워크 자원 사용을 제한하고 있는가?	필수	
2-3. 컨테이너를 최소 권한으로 실행해야 한다.		
(1) 특권(Privileged) 모드 컨테이너의 실행을 금지하고 있는가?	필수	
(2) 호스트와 네임스페이스를 공유하는 것을 금지하고 있는가?	필수	
(3) 오케스트레이터의 컨테이너 권한 제어 기능을 활용하고 있는가?	필수	
2-4. 관리 인터페이스에 대한 인증과 접근 제어를 강화해야 한다.		
(1) SELinux, AppArmor와 같은 강제 접근제어 기술과 Seccomp와 같은 시스템 기능 제한 메커니즘을 활용하고 있는가?	필수	
(2) 컨테이너의 비정상적인 활동을 탐지할 수 있는 보안 솔루션을 활용하고 있는가?	필수	
(3) 컨테이너의 상태에 대한 모니터링을 수행해야 하고 있는가?	필수	
2-5. 컨테이너 및 컨테이너 런타임에 대한 네트워크 접근을 제한해야 한다.		
(1) 컨테이너 런타임의 관리 인터페이스 원격 접근을 금지하고 있는가?	필수	
(2) 컨테이너 런타임의 관리 인터페이스 접근을 위한 전용 물리 NIC을 할당하거나 불가능한 경우 전용 가상 네트워크를 할당 하고 있는가?	필수	
(3) 네트워크를 통한 컨테이너 런타임의 관리 인터페이스 접속 시에는 암호화 프로토콜을 사용하고 있는가?	필수	
(4) 컨테이너 내부 접속을 위한 SSH 서버를 제거했는가?	필수	

OpenShift Node OS 보안솔루션 설치



- OpenShift Node들은 CoreOS 사용
  Master Node 3대
  Infra Node 3대
  Worker Node 7대
- 필수 서버 보안 프로그램 설치 이슈
- CoreOS는 OpenShift의 Appliance로 설치 예외 조치
- CoreOS를 SSH 접속할 수 있는 유일한 서버인
  Bastion Server에만 서버 보안 프로그램 설치로 협의.

CoreOS SSH Port 변경 요청

openmaru
오픈마루(주)

*리눅스 서버 원격 접속 서비스 **SSH**를 통한 무작위 대입 공격 여전히 위험*
8월 한 달간 총 4,436개의 IP로부터 약 65만번의 SSH 무작위 대입 시도 발생
SSH의 서비스 포트 변경, root 계정의 원격 로그인 금지, 강한 패스워드 정책 등 시행해야

https://www.boannews.com/media/view.asp?idx=109910

The 8 Most Vulnerable Ports to Check When Pentesting

https://www.makeuseof.com/vulnerable-ports-check-when-pentesting/

This kind of modification is highly discouraged and doesn't fall under tested/documented methods, keeping in mind that the Red Hat Core OS was already designed with all the possible security measures in hand.

- Red Hat Core OS는 이미 모든 가능한 보안 조치를 고려하여 설계

One such security measure is the accessibility of the RHCOS machines that is possible only through the SSH access keys generated prior to the installation, which is compatible only with the

- 설치 중에 생성한 SSH 액세스 키를 통해서만 가능한 Red Hat Core OS 시스템에 액세스 가능

https://access.redhat.com/solutions/5495101

Red Hat OpenShift OS Layer 보안

- **RHEL CoreOS로 컨테이너 호스트를 안전하게 보호**
- **Container를 기동하는 Host OS에 적합한 개념을 구현한 RHEL CoreOS**

An Ideal Container Host would be	RHEL CoreOS
Minimal	Only what's needed to run containers
Secure	Read-only & locked down
Immutable	Immutable image-based deployments & updates
Always up-to-date	OS updates are automated and transparent
Updates never break my apps	Isolates all applications as containers
Updates never break my cluster	OS components are compatible with the cluster
Supported on my infra of choice	Inherits majority of the RHEL ecosystem
Simple to configure	Installer generated configuration
Effortless to manage	Managed by Kubernetes Operators

New Paradigm for Operating System

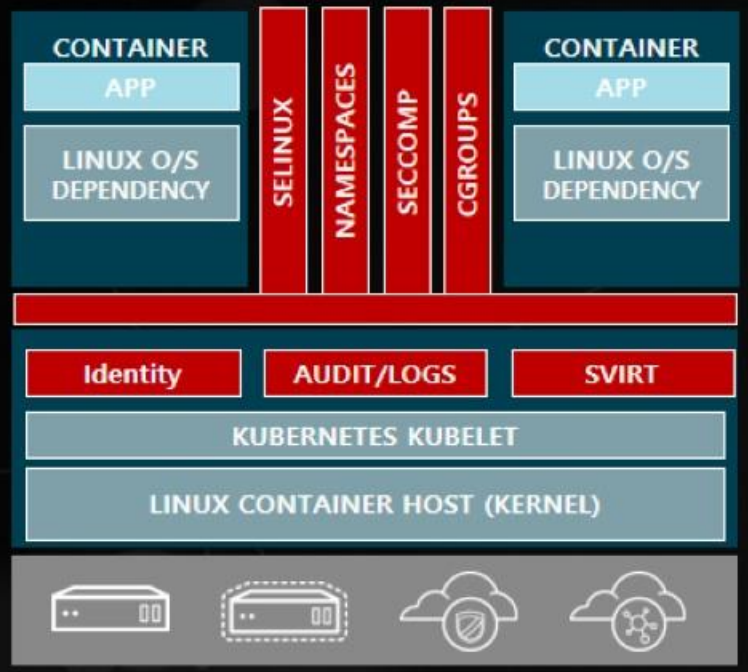
openmaru
오픈마루㈜

- You must see operating system as one of many components of OpenShift that it installs and manages
- System administrator would not have to ssh into machines/vms to tune, config, patch or upgrade RH CoreOS. In fact RH recommends disabling ssh into OpenShift's machines/vms in production
- RHEL CoreOS is the minimum set of libraries and kernel needed to run containers
- RHEL CoreOS is immutable
- admins cannot install new libraries
- admins cannot alter certain filesystems (only /etc and /var are writable)

Red Hat OpenShift OS Layer 보안 기술

openmaru
오픈마루㈜

SELINUX, NAMESPACES, SECCOMP, CGROUPS의
LINUX SECURITY 기술을 이용한 HOST OS 보호



- Security in the RHEL host applies to the container
- SELINUX and Kernel Namespaces are the one-two punch no one can beat
- Protects not only the host, but containers from each other
- RHEL CoreOS provides minimized attack surface

Red Hat OpenShift CoreOS 보안

openmaru
오픈마루㈜

RHEL CoreOS로 컨테이너 호스트를 안전하게 보호

Red Hat Enterprise Linux CoreOS is versioned with OpenShift

CoreOS is tested and shipped in conjunction with the platform.

Red Hat runs thousands of tests against these configurations.

Red Hat Enterprise Linux CoreOS is managed by the cluster

The Operating system is operated as part of the cluster,
with the config for components managed by Machine Config Operator:

- CRI-O config
- Kubelet config
- Authorized registries
- SSH config

OPENSHIFT 4

Machine Config Operator (MCO)

openmaru 오픈마루㈜

Provides cluster-level configuration, enables rolling upgrades, and prevents drift between new and existing nodes.

The MCO is the heart of what makes RHCOS a kube-native operating system.

Configure Kernel Arguments for the Cluster

- oc create -f 50-kargs.yaml
- oc edit mc/50-kargs

MCO can be paused to suspend operations

- Provides control for when changes can be deployed

Custom MachinePools can have inheritance

- Enables MachineConfigs to scale

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worke
r
  name: 50-kargs
spec:
  KernelArguments:
      audit=1
      audit_backlog_limit=8192
      net.ifnames.prefix=net
```

Product Manager: Ben Breard

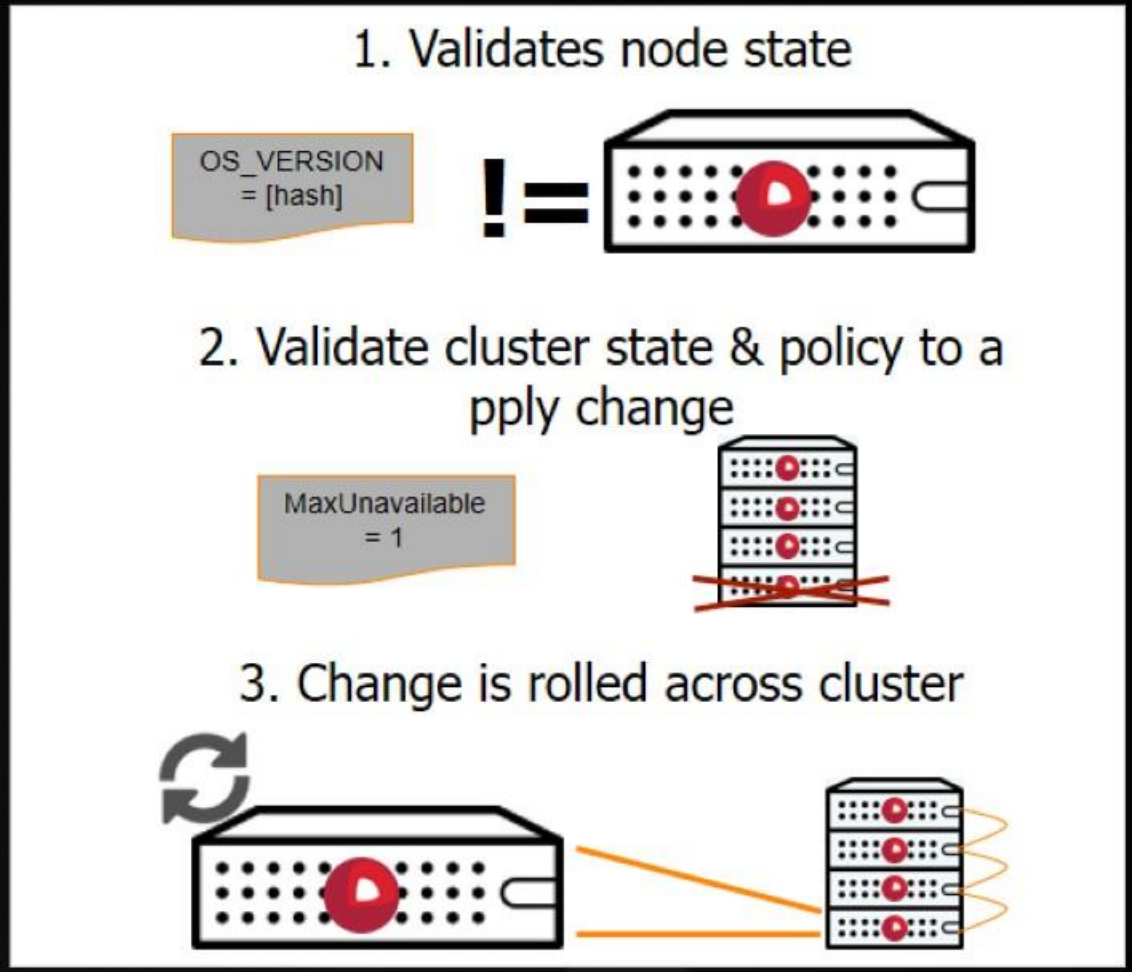
Generally Available

Machine Config Operator (MCO) 동작 과정

openmaru
오픈마루㈜

Provides the connective tissue between the controllers and operating system to handle the following:

1. **Cordin / uncordins nodes**
2. **Drain pods**
3. **Apply OS updates**
4. **Apply config changes**
5. **Apply changes for systemd units**
6. **Reboot**



Red Hat OpenShift 플랫폼 SCC 보안

openmaru
오픈마루㈜

SCC(Security Context Constraints)로 플랫폼 보안 및 접근제어

```
$ oc describe scc restricted
Name:                                   restricted
Priority:                               <none>
Access:
  Users:                                <none>
  Groups:                               system:authenticated
Settings:
  Allow Privileged:                     false
  Default Add Capabilities:             <none>
  Required Drop Capabilities:           KILL,MKNOD,SYS_CHROOT,SETUID,SETGID
  Allowed Capabilities:                 <none>
  Allowed Seccomp Profiles:             <none>
  Allowed Volume Types:                 configMap,downwardAPI,emptyDir,persistentVolumeClaim,projected,
  Allow Host Network:                   false
  Allow Host Ports:                     false
  Allow Host PID:                       false
  Allow Host IPC:                       false
  Read Only Root Filesystem:            false
  Run As User Strategy: MustRunAsRange
```

- Allow administrators to control permissions for pods
- Restricted SCC is granted to all users
- By default, no containers can run as root
- Admin can grant access to privileged SCC
- Custom SCCs can be created

Red Hat OpenShift 플랫폼 Oauth 보안

openmaru
오픈마루㈜

LDAP 및 SSO 연결로 사용자 계정 및 그룹 관리

OpenShift includes an OAuth server

- Identifies the person requesting a token, using a configured identity provider
- Determines a mapping from that identity to an OpenShift user
- Issues an OAuth access token which authenticates that user to the API

Supported Identity Providers include

- Keystone
- LDAP
- GitHub
- GitLab
- GitHub Enterprise (new with 3.11)
- Google
- OpenID Connect
- Security Support Provider Interface (SSPI) to support SSO flows on Windows (Kerberos)

Red Hat OpenShift 플랫폼 권한 관리

openmaru
오픈마루㈜

RBAC으로 전체 플랫폼의 제어 권한 관리

Role based authorization

- **Project scope & cluster scope available**
- **Matches request attributes (verb,object,etc)**
- **If no roles match, request is denied (deny by default)**
- **Operator- and user-level roles are defined by default**
- **Custom roles are supported**

For more information see: Managing RBAC in OpenShift

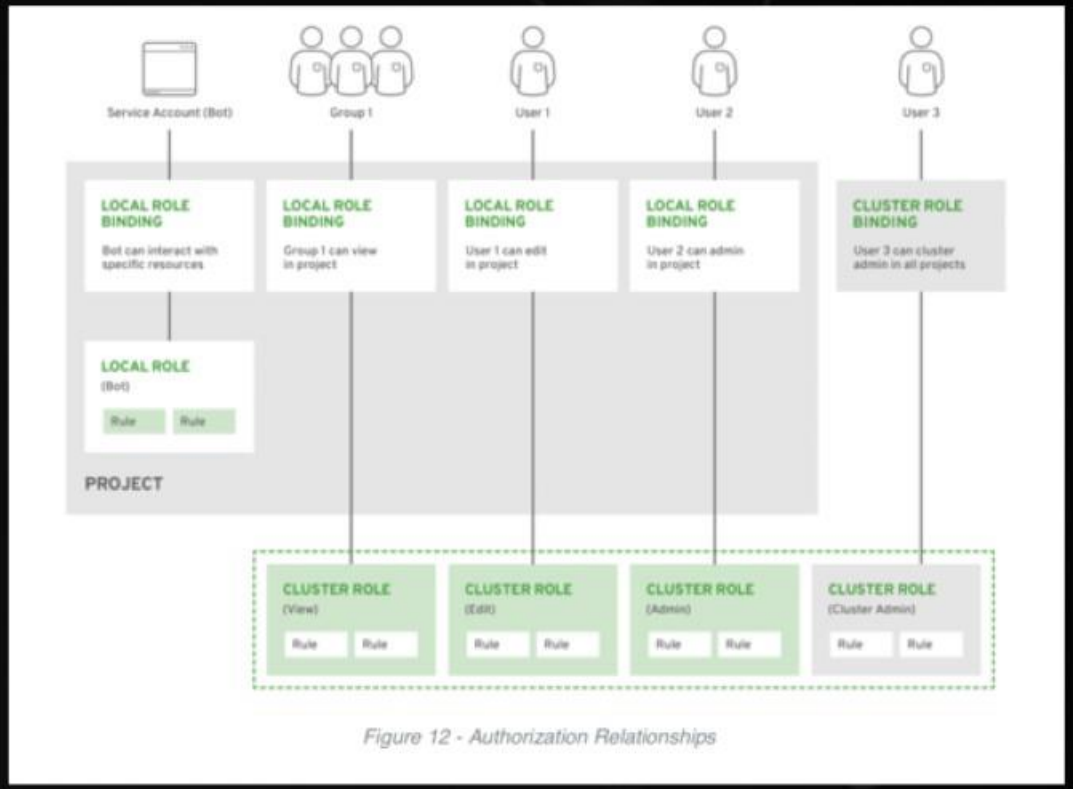
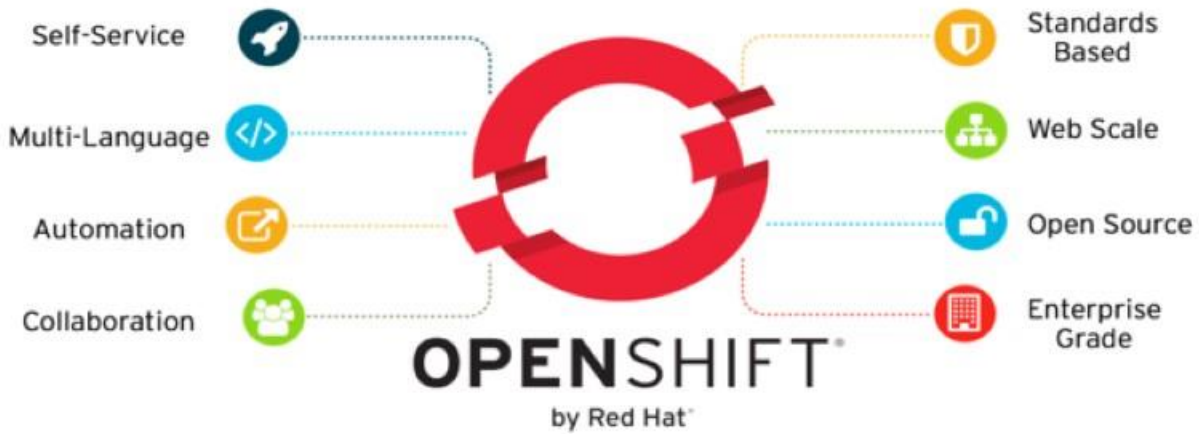


Figure 12 - Authorization Relationships

1. 제품 개요

Red Hat OpenShift Container Platform은 표준 오픈소스 기술을 사용한 가장 신뢰성 있는 PaaS 클라우드 플랫폼입니다. 또한 Red Hat이 제공하는 엔터프라이즈 솔루션 기술력을 기반으로 보안에 강화된 안정적인 PaaS 클라우드 플랫폼을 구축합니다.



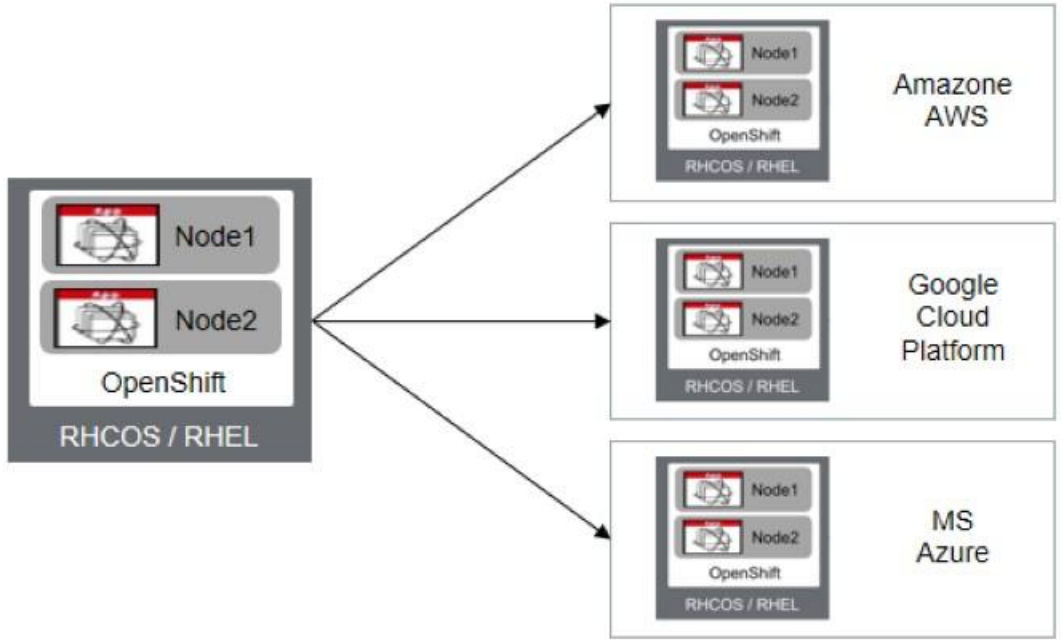
구분	세부 내용
개요	· 오픈소스 기반의 PaaS(Platform-as-a-Service) 플랫폼 · 다중 언어, DB, Application Framework 등 지원 · 컨테이너 통합 관리를 위한 Enterprise Level Kubernetes 포함 · 엔터프라이즈를 위한 사설 환경 플랫폼 지원 · 부하량에 따른 자동 확장 (Auto-Scale) 기능 · OpenShift의 소스 코드는 범용성을 가진 다양한 언어로 제작 및 지원 (Java, Go, C, C++, Python 등)
다중 언어 지원	· Java, .NET, Go, Node.js, Ruby, Python, PHP 및 Perl 등이 포함되어 개발자가 작업의 특성을 기준으로 가장 적절한 프로그래밍 언어를 선택 가능
클라우드 인프라 선택	· Red Hat Enterprise Linux를 기반으로 사용자가 인프라를 선택할 수 있도록 설계되어 Bare-Metal, KVM, VMware, HyperV, OpenStack 등 에서 실행 가능

표준 오픈소스 기술	• CRI-O : 업계 표준 오픈소스 컨테이너 기술 • Kubernetes : 가장 강력한 오케스트레이션 기술 • Open vSwitch : 표준 오픈소스 SDN 기술
자동화	• 셀프서비스 기반 프로비저닝 • 자동화된 Auto Scaling • 서비스 무중단을 위한 다양한 배포 전략 • 소프트웨어 인프라 기반의 MSA 전략 제시
플랫폼 다양성	• 다양한 개발언어 런타임 제공 • 다양한 DB 플랫폼 제공 • Source-To-Image 빌드 / Docker 빌드 등 다양한 빌드 전략 수립 • CI/CD 파이프라인 구축 방안 제시 • 협업 중심의 DevOps 전략 제시
엔터프라이즈 기술	• Red Hat Enterprise Linux CoreOS (RHCOS) 및 Red Hat Enterprise Linux (RHEL)의 강력한 보안 제공 • RHCOS, RHEL 기술지원 포함 • 기업 환경에 바로 적용 가능한 xPaaS 플랫폼 제공 • 다양한 3rd Party S/W와 호환성 및 안정성 제공

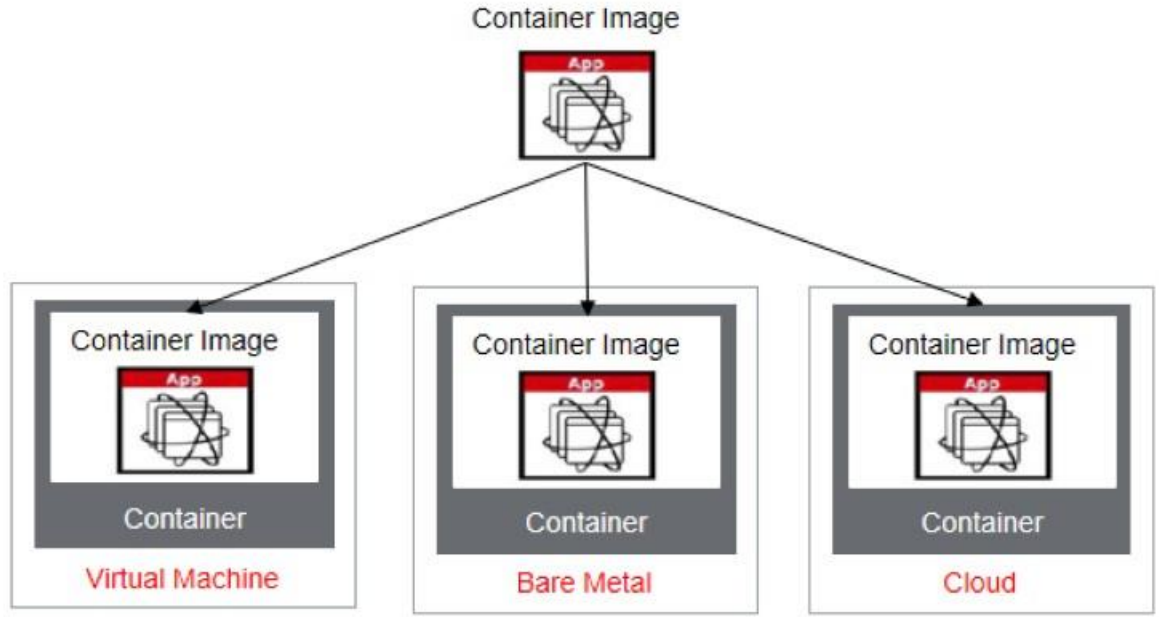
2. 제품 특장점 > 개방성

Red Hat Enterprise Linux CoreOS와 Red Hat Enterprise Linux로 구성된 모든 환경에서 완벽한 PaaS 플랫폼을 구축할 수 있으며, Container를 실행 할 수 있는 모든 환경에서는 현재 구축된 모든 Application에 대해 실행할 수 있습니다.

- **Private/Public 클라우드(Amazon, Azure, GCP) 및 VM, Bare Metal 등의 Red Hat Linux OS를 지원하는 모든 인프라 환경 구축 가능**



- **표준 Container를 지원하는 모든 환경에서 생성된 Image를 그대로 사용**
- **VM, Bare Metal, Cloud 등의 다양한 환경의 표준 Container 지원**

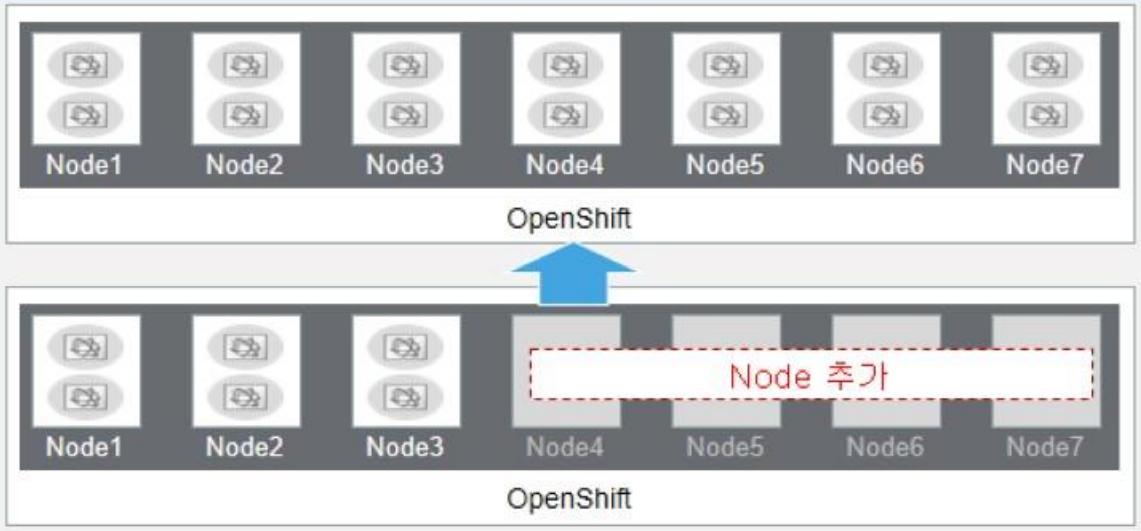


Red Hat

2. 제품 특장점 > 확장성

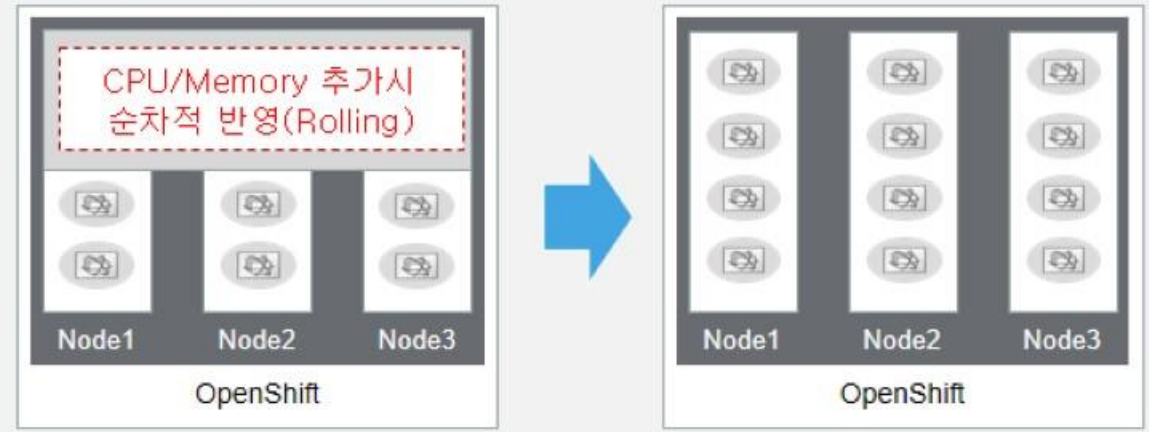
운영 시 발생 할 수 있는 부하에 따른 서버 증설 및 서버 Spec 조정시에 유연하게 확장할 수 있는 기능을 제공합니다.

노드 Scale-Out



- 운영 서비스의 중단 없는 새로운 **Node** 추가 기능 제공.
- 컨테이너는 **Scheduler**에 의해 자동으로 배치.

노드 Scale-up



- 기존 **Node**의 **CPU/Memory** 증설을 통한 확장으로, 노드당 컨테이너 수용량을 증가.
- 노드의 재기동 필요.

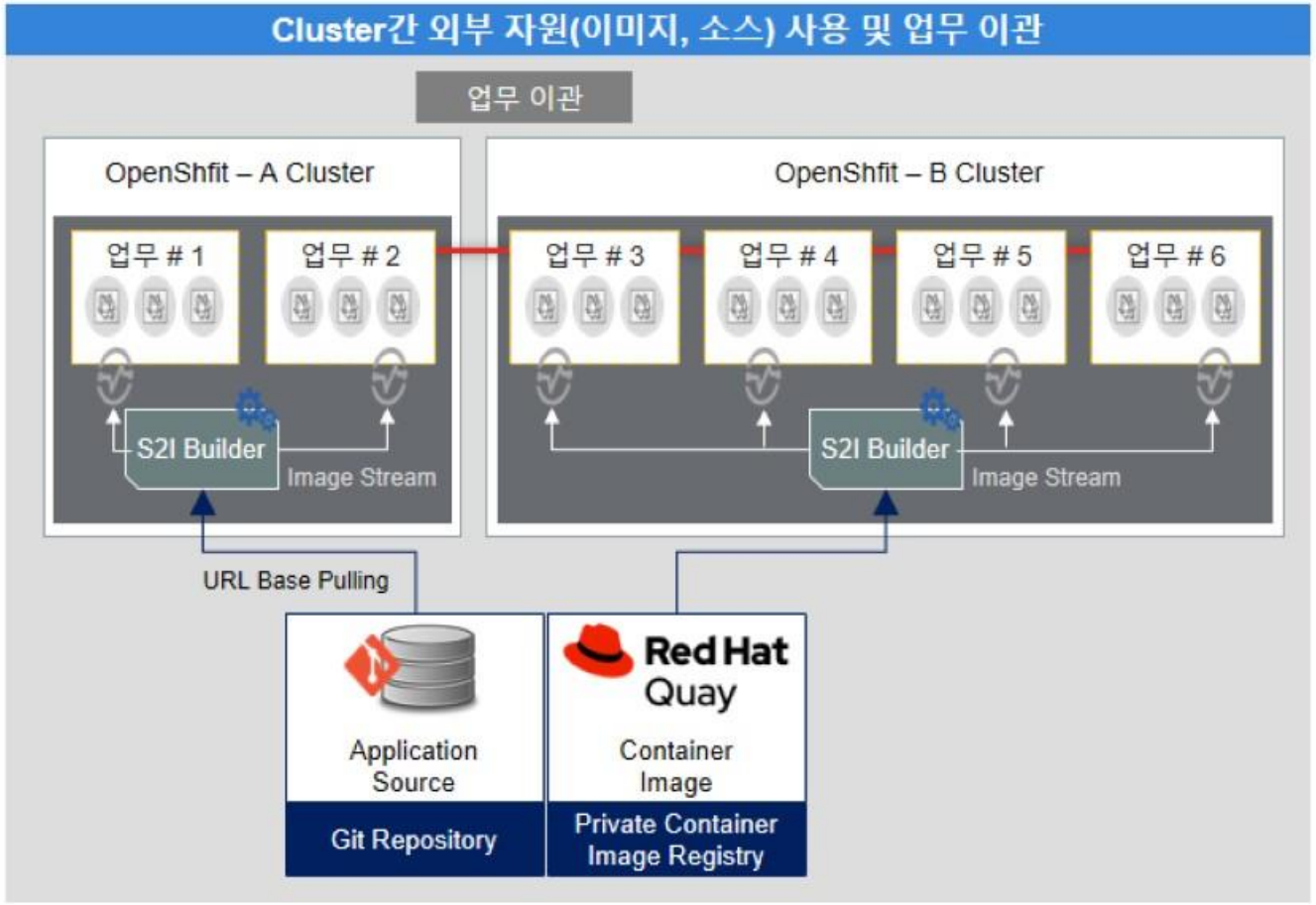
노드 Scale-Out : 서비스 중단 없음.

노드 Scale-Up : CPU/Memory 증설 후 서버 재시작시 서비스 중단 없음.

Red Hat

2. 제품 특장점 > 유연성

인터넷이 차단된 환경에서 Private Container Registry와 Git Repository를 구성하여 Cluster 내부와 Cluster 간의 Application Source, Container Image의 공유 기능 및 시스템 전체를 타 Cluster로 이관할 수 있는 기능을 제공합니다.

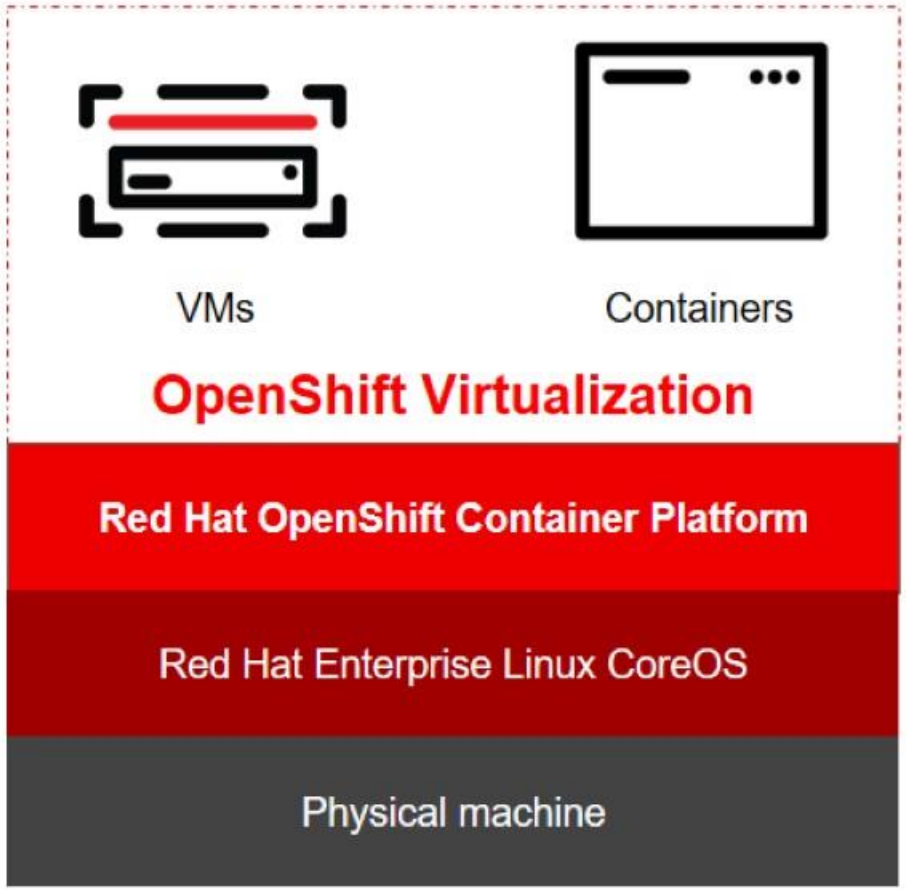


구분			세부 내용
From	To	가능여부	
OpenShift Container Platform	OpenShift Container Platform	◎	
	OpenShift Origin (Community)	◎	오픈소스 버전
	OpenShift Dedicated	◎	Red Hat에서 제공하는 기업용 Public 클라우드
	OpenShift Online	◎	Public 클라우드 서비스
	Kubernetes + Docker	○	EFK Logging Aggregation, Hawkular Metrics 등 OpenShift의 컴포넌트 형태의 일부 애플리케이션은 마이그레이션 불가.

Red Hat

2. 제품 특장점 > 가상 머신 운영 및 관리

Red Hat OpenShift Container Platform은 기본 내장된 OpenShift Virtualization을 통해 컨테이너 플랫폼 위에 가상 머신(VM)을 운영, 관리하는 기능을 제공합니다.



Core

- 10년이상 안정성을 제공하던 KVM 기술 기반 (RHV, RHOSP)
- 강력한 VM 기준치 성능 제공
- RHV, VMware에서 가져오기 기능 제공(콜드, 웜 마이그레이션)
- 윈도우 서버 2019, 2016, 2012, Win 10 등 Certified
- Live Migration, SnapShot 기능 제공
- K8s가 Container처럼 VM을 오케스트레이션
  고가용성(Health체크), 배포, 스케쥴, 스케일 인/아웃/업

Network

- 온라인 네트워크 인터페이스 추가 및 제거 기능 제공
- VM 용 CNI 연결 제공인증 테스트 스위트 추가

Storage

- 기존 OS 이미지 및 템플릿으로 쉬운 VM 생성 기능 제공
- CSI 스냅샷을 통한 빠른 DataVolume CDI 복제
- ContainerDisks를 영구 스토리지로 보다 효율적으로 가져옴

2. 제품 특장점 > 개발 지원

Red Hat OpenShift Container Platform은 Java, Node.js, .NET, Perl, Ruby, PHP, Python 등 다양한 Runtime 환경을 위한 Template Catalog를 지원하며 이외에도 Docker Build 및 Image Stream 생성을 통해 다양한 Container를 서비스 할 수 있습니다.

구분	Support Image
개발언어	▪ .NET Core 3.1 ▪ Perl 5.26, Perl 5.30 ▪ Node.js 12, Node.js 14, Node.js 16 ▪ PHP 7.3, PHP 7.4 ▪ Python 2.7, Python 3.6, Python 3.8 ▪ Ruby 2.5, Ruby 2.6, Ruby 2.7, Ruby 3.0 ▪ Go 1.16.12 ▪ Java 8, 11, 17
미들웨어	▪ Red Hat JBoss EAP 7.3, 7.4 ▪ Red Hat JBoss Web Server(Tomcat) 3.1(7.0,8.0), 5.6(9.0) ▪ Red Hat JBoss Data Grid 7.3, 8 ▪ Red Hat JBoss A-MQ 6.3 ▪ Red Hat Single Sign-On 7.5 ▪ Red Hat JBoss Fuse Integration Services 7.10 ▪ Red Hat JBoss Data Virtualization 6.4 ▪ Apache httpd 2.4 ▪ Nginx 1.16, 1.18, 1.20

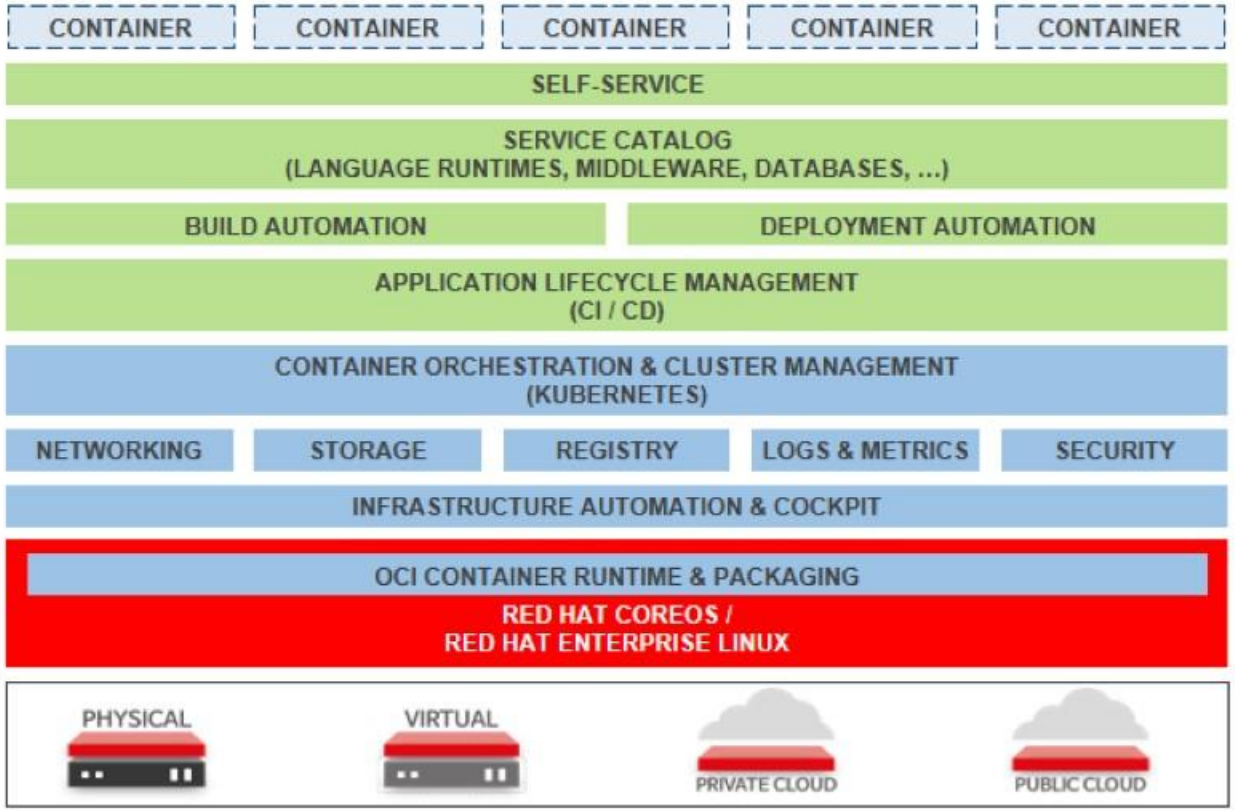
구분	Support Image
DataBase	▪ MariaDB 10.3, 10.5 ▪ MySQL 8.0 ▪ PostgreSQL 10, 12, 13 ▪ Redis 5, 6
Framework	▪ Spring Boot 2.3, 2.4 ▪ Quarkus 2.2 ▪ Eclipse Vert.x 4.x ▪ Jenkins2
기타	▪ catalog.redhat.com/software/containers/search 에 없는 이미지의 경우 커스텀 이미지로 제작

Red Hat

3. 제품 아키텍처

Red Hat OpenShift Container Platform은 Linux 기반의 경량 컨테이너 이미지 패키징 및 생성을 위한 추상화를 제공하며 Kubernetes 기반의 클러스터 관리를 제공하여 여러 호스트에서 컨테이너를 제어합니다.

- 모듈 구성도



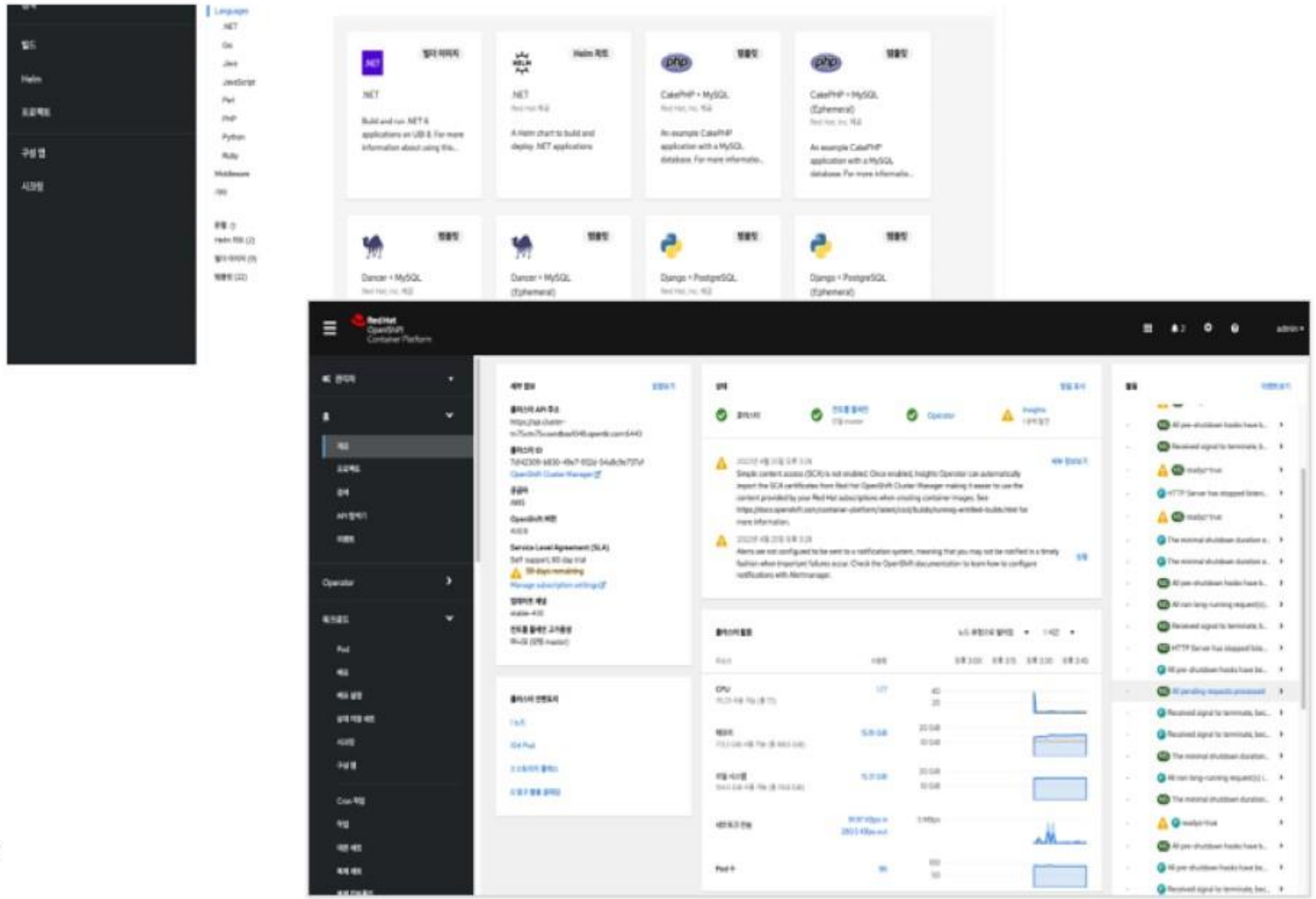
- 모듈별 주요 역할

구분	모듈	주요 역할
Kubernetes	Container Orchestration	컨테이너 생성 및 삭제 기능
	Cluster Management	클러스터 내 프로젝트 생성, 환경 설정 기능
Network	Open vSwitch	POD간의 통신 기능 담당
Storage	Persistent Volume	PV를 통해 스토리지와 연계하여 영구적인 데이터 저장 기능 제공
Registry	Integrated Registry	RHOCP 내 도커 이미지 저장소
Logs & Metric	EFK	컨테이너에 대한 로그 관리, 분석, 모니터링
	Metircs / Prometheus	컨테이너 자원(CPU, Network, Mem) 사용률 모니터링
Security	Identity Provider	인증된 사용자만 접근 허용
Container	**CRI-O**	**표준 컨테이너 기술**

Red Hat

4. 주요 기능 > Master Portal

Web Console의 메뉴는 크게 관리자용 화면과 개발자용 화면으로 구분되며 하위 메뉴에서 Pod의 Container Health check, Performance Monitoring, Deployment 및 워크로드 제어, Build 제어, Scale 변경 관리, Log 이력 관리 및 플랫폼 이벤트 모니터링의 기능 등을 제공합니다.

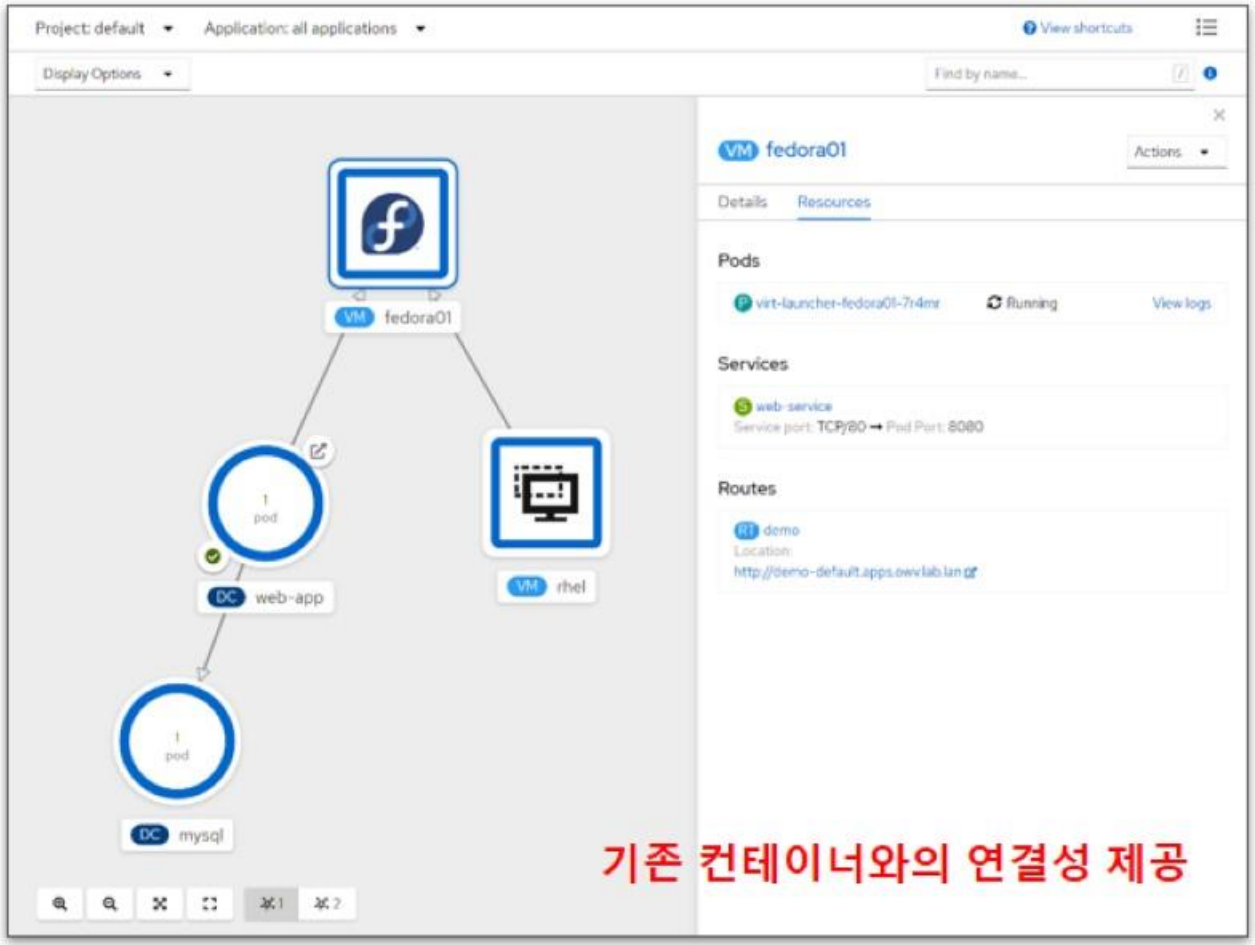


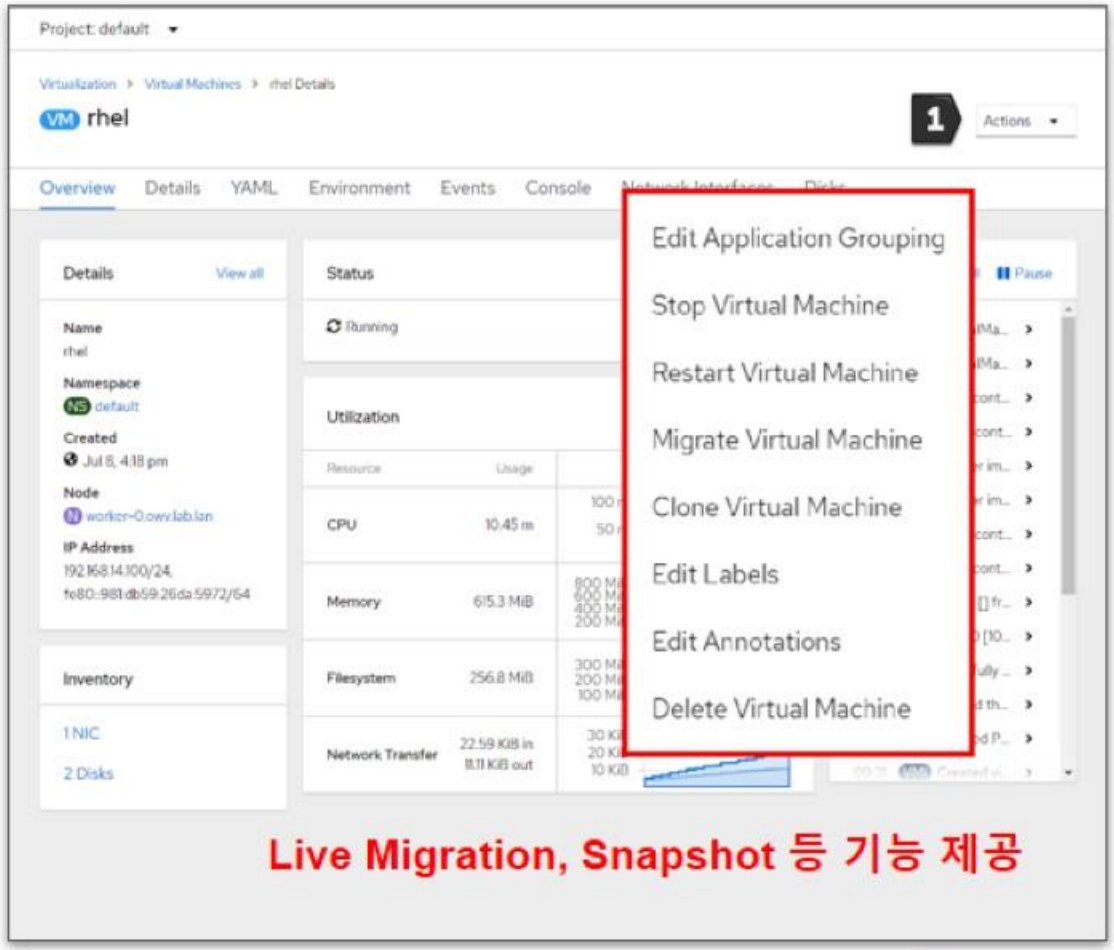
- Project(namespace) 관리
- 사용자의 Role 관리
- Image Catalog 검색 및 애플리케이션 생성
- Pod 상태 확인
- 애플리케이션 및 리소스 설정 제어
- Node(Host) 상태와 자원, 정보 확인
- 사용중인 CPU, Memory, Disk 자원 확인
- Event 확인
- 추가 모니터링 대시보드 제공
  : Prometheus, Grafana, Event Dashboard

Red Hat

4. 주요 기능 > 가상화 기능

Red Hat OpenShift Container Platform은 OpenShift Virtualization 기능을 통해 기존 가상 머신(VM)도 컨테이너처럼 구동 가능하며, 컨테이너와 VM에 대해 동일한 관리 편의성 및 확장성을 제공합니다.



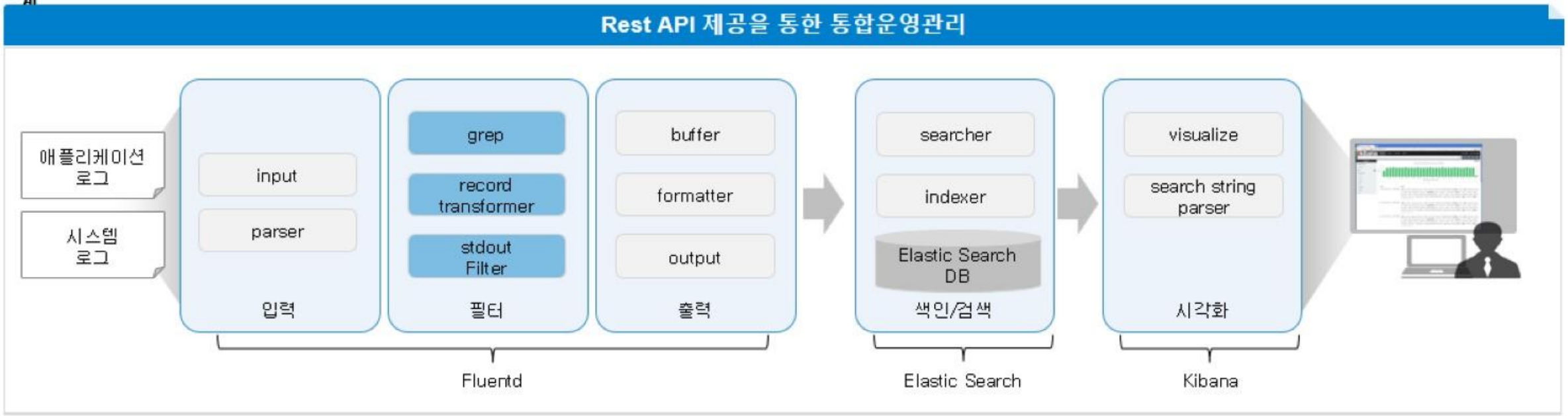


Red Hat

4. 주요 기능 > 로깅

Red Hat OpenShift Container Platform은 PaaS 클라우드에서 서비스되는 컨테이너의 로그를 관리하기 위해 EFK (ElasticSearch-Fluentd-Kibana) 스택을 사용합니다. EFK는 로그 수집 단계에서 로그를 필터링하거나 필드 추가/삭제 또는 마스킹하고, 시각화 단계에서 수집된 로그를 기반으로 필터링하여 필요한 로그를 빠르게

PaaS 플랫폼 통합 성능 및 이벤트 모니터링 체계

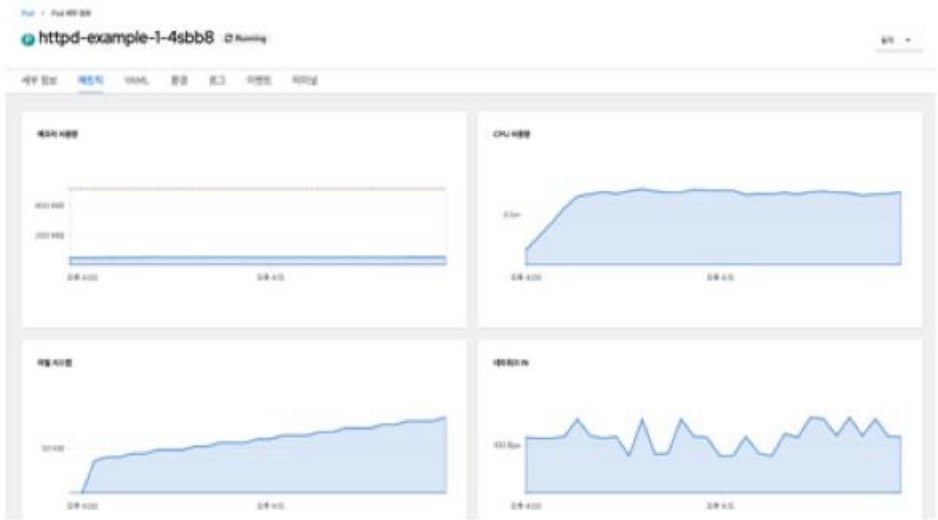


입력 필터	비고
grep	로그에 특정 문자열, 정규표현식, 또는 포함하지 않을 경우 이벤트에 따라 로그 수집
record transformer	로그의 값을 치환, 추가 등 변환 작업 수행
stdout Filter plugin	특정 패턴의 로그는 stdout 또는 다른 output 채널을 써서 출력하고 json, ltsv 등 포맷 지정

4. 주요 기능 > 메트릭

Red Hat OpenShift Container Platform의 Node와 Pod의 리소스 메트릭은 Prometheus를 통해 수집, 모니터링 됩니다. 기본 Web Console의 워크로드별 메트릭 화면으로 개별 워크로드의 CPU, Memory, Network 사용량을 확인할 수 있으며, 대시보드 메뉴를 통해 클러스터, Namespace, Node, Pod 에 대한 전체적인 리소스 사용량을 모니터링 합니다.

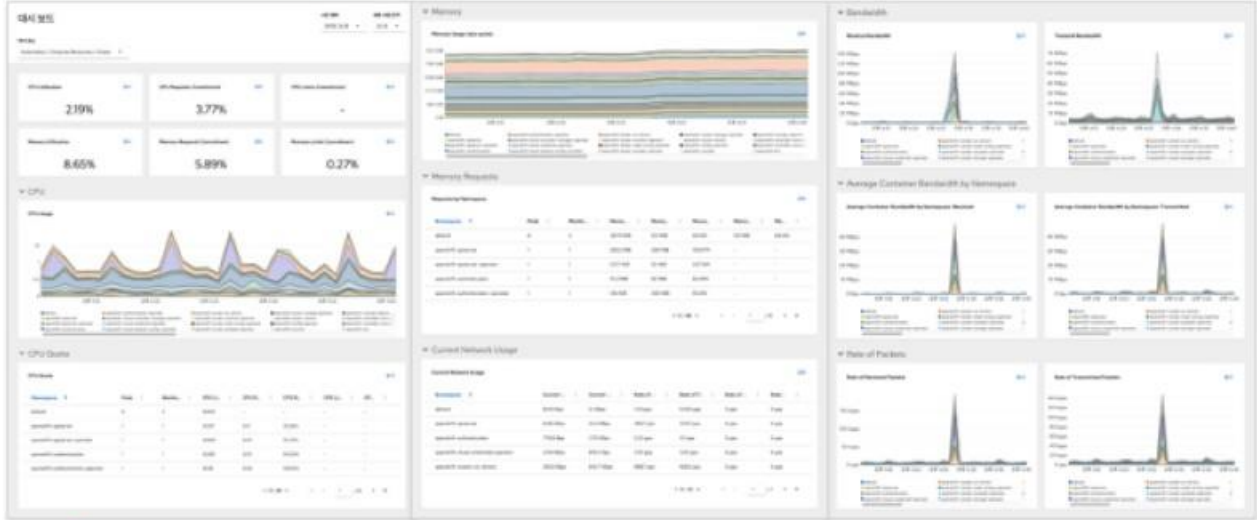
워크로드 메뉴의 메트릭 서브 메뉴 화면



기능

- POD 별 실시간 CPU/Memory / Network 모니터링

대시보드 메뉴 화면



기능

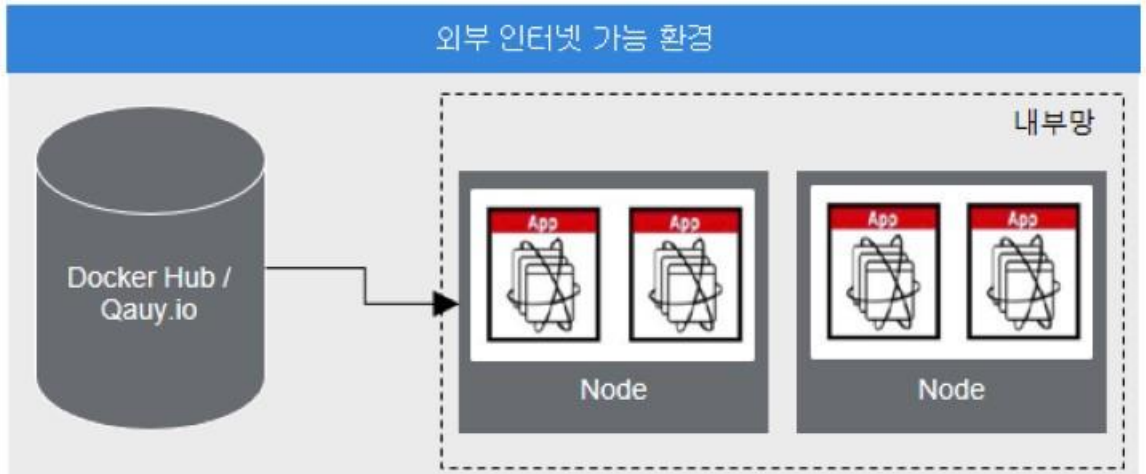
- 노드별 모니터링(CPU, Memory, Disk I/O, Network)
- POD/Project/Cluster 별 컨테이너 리소스 사용현황

Red Hat

4. 주요 기능 > 이미지 레지스트리

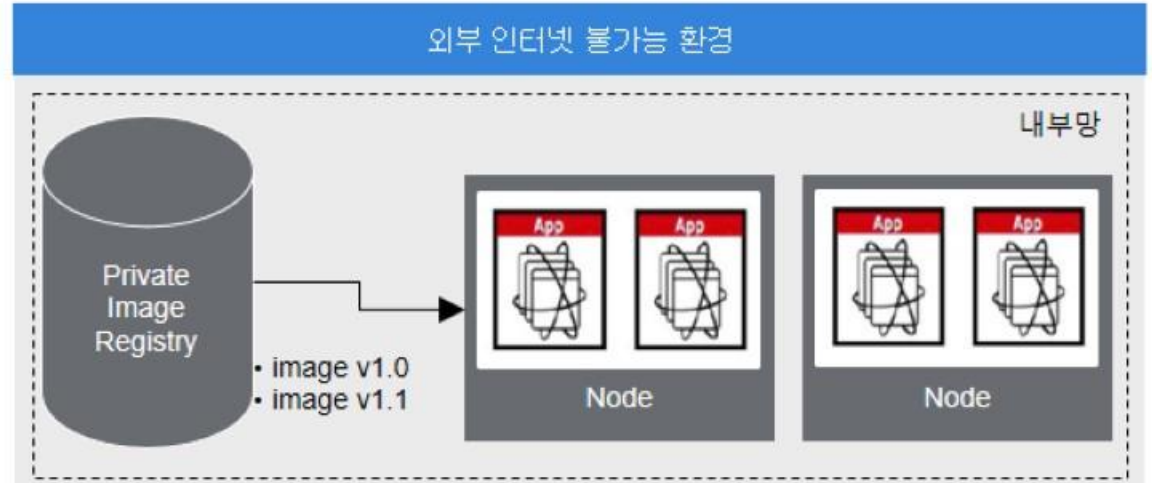
외부 인터넷 연결이 불가능한 환경인 경우 내부망에 Private Image Registry를 구축하여 컨테이너 이미지를 관리할 수 있습니다.

●외부망 Container Image Registry 연동



- 외부 인터넷이 연결 가능한 환경에서는 별도 저장소 구축 없이 간편하게 컨테이너 이미지 및 소스를 가져올 수 있다.

●내부망 Private Image Registry 구축 및 연동



- 인터넷 연결이 안 되는 환경은 내부에 컨테이너 저장소를 별도 구축하여 사용해야 하며 이미지 버전 관리를 할 수 있다.

Red Hat

4. 주요 기능 > Persistent Volume

Red Hat OpenShift Container Platform은 NFS, SAN, iSCSI, FiberChannel 등 일반적인 스토리지 뿐만 아니라 OpenStack Cinder, OpenStack Manila, Red Hat OpenShift Data Foundation 등 다양한 스토리지 가상화 솔루션을 연계하는 방안을 제공합니다. 컨테이너는 필요한 공간을 요청(Persistent Volume Claim)하고 RHOCP는 요구에 맞는 저장공간(Persistent Volume)을 할당합니다.

지원 스토리지 및 제공 기능

종 류	Access Mode			동적 프로비저닝 plugin
	ReadWriteOnce	ReadOnlyMany	ReadWriteMany	
AWS EBS	✓			✓
Azure File	✓	✓	✓	✓
Azure Disk	✓			✓
Cinder	✓			✓
Fiber Channel	✓	✓		
GCE Persistent Disk	✓			✓
HostPath	✓			
iSCSI	✓	✓		
Local Volume	✓			
NFS	✓	✓	✓	
OpenStack Manila			✓	✓
Red Hat OpenShift Data Foundation	✓		✓	
VMware vSphere	✓			✓

엑세스 모드 설명

Access Mode	CLI 약어	비고
ReadWriteOnce	RWO	단일 노드에서 read-write 모드로 마운트
ReadOnlyMany	ROX	복수 노드에서 read-only 모드로 마운트
ReadWriteMany	RWX	복수 노드에서 read-write 모드로 마운트

Red Hat

4. 주요 기능 > Persistent Volume

Red Hat OpenShift Container Platform은 NFS, SAN, iSCSI, FiberChannel 등 일반적인 스토리지 뿐만 아니라 OpenStack Cinder, OpenStack Manila, Red Hat OpenShift Data Foundation 등 다양한 스토리지 가상화 솔루션을 연계하는 방안을 제공합니다. 컨테이너는 필요한 공간을 요청(Persistent Volume Claim)하고 RHOCP는 요구에 맞는 저장공간(Persistent Volume)을 할당합니다.

오픈시프트를 위한 권장 구성 가능한 스토리지 기술

스토리지 유형	ROX[1]	RWX[2]	Registry	확장 레지스트리	Metrics[3]	Logging	Apps
Block	예[4]	아니오	구성 가능	구성 불가능	권장	권장	권장
File	예[4]	예	구성 가능	구성 가능	구성 가능[5]	구성 가능[6]	권장
Object	예	예	권장	권장	구성 불가능	구성 불가능	구성 불가능[7]

1 ReadOnlyMany

2 ReadWriteMany

3 metric에 사용되는 기본 기술은 Prometheus입니다.

4 물리적 디스크, VM 물리적 디스크, VMDK, NFS를 통한 루프백, AWS EBS 및 Azure Disk에는 적용되지 않습니 다.

5 metric의 경우 RWX(ReadWriteMany) 액세스 모드로 파일 스토리지를 사용하는 것은 안정적이지 않습니다. 파일 스토리지를 사용하는 경우 metric과 함께 사용하도록 구성된 PVC(영구 볼륨 클레임)에서 RWX 액세스 모드를 구성하지 마십시오.

6 로깅의 경우 공유 스토리지 사용은 안티 패턴입니다. Elasticsearch당당 하나의 볼륨이 필요합니다.

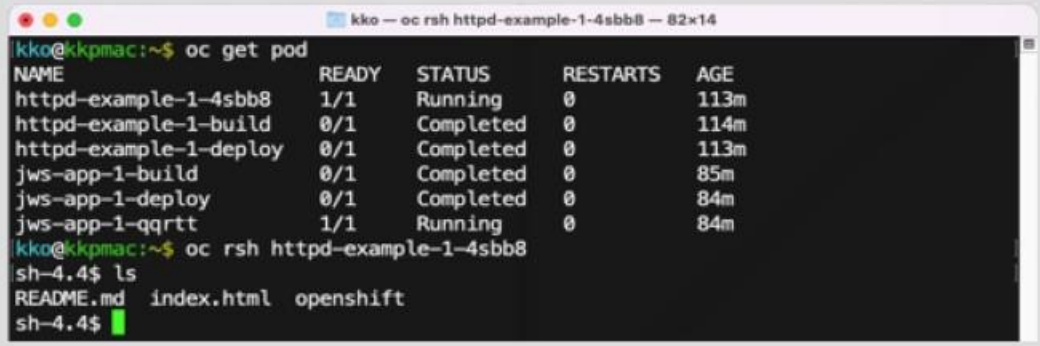
7 OpenShift Container Platform의 PV 또는 PVC를 통해서는 오브젝트 스토리지가 사용되지 않습니다. 앱은 오브젝트 스토리지 REST API와 통합해야 합니다.

4. 주요 기능 > 컨테이너 접근

관리자는 Web Console 또는 CLI로 서비스 중인 Pod에 접속할 수 있으며, configuration이나 로그를 확인할 수 있습니다. 또한 rsync를 통해 Pod 내부의 파일을 로컬 파일시스템에 upload/download 하는 기능을 제공합니다.

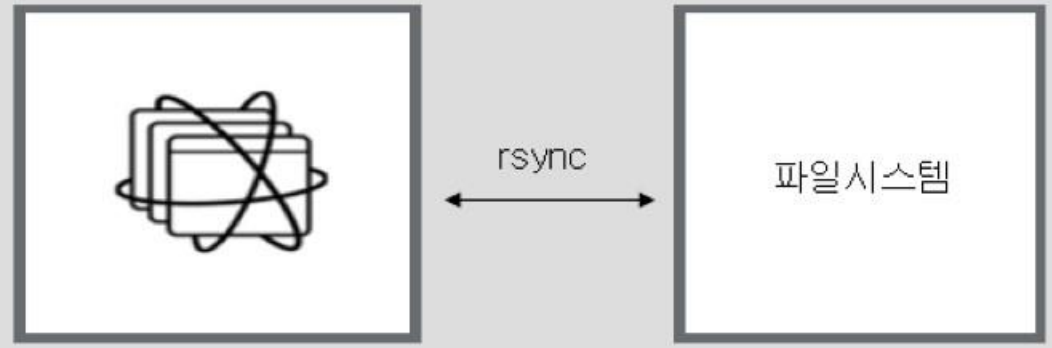
SSH 터미널 접근 방안

[Web Console]



[CLI]

Container File up/download 방안



1. 파일 업로드 (로컬 -> 컨테이너)

$ oc rsync <로컬파일> <pod이름>:<pod디렉토리>

2. 파일 다운로드 (컨테이너 -> 로컬)

$ oc rsync <pod이름>:<pod파일> <로컬디렉토리>

Red Hat

4. 주요 기능 > 이미지 자동 설치

Red Hat OpenShift Container Platform은 사용자가 쉽게 서비스를 생성 할수 있도록 Web UI를 통한 다양한 Catalogue를 제공 하고 있으며 서비스 유형에 따른 설정 정보 변경의 편의성을 위해 Template을 제공하고 이에 대한 재활용하여 추가/변경이 용이합니다.

웹 UI를 통한 Self-Service제공

1. 프로젝트 생성

2. 서비스 선택

3. Configuration

4. 빌드(자동)

5. 서비스 모니터링

6. 서비스(POD) 생성

7. 서비스&경로 생성

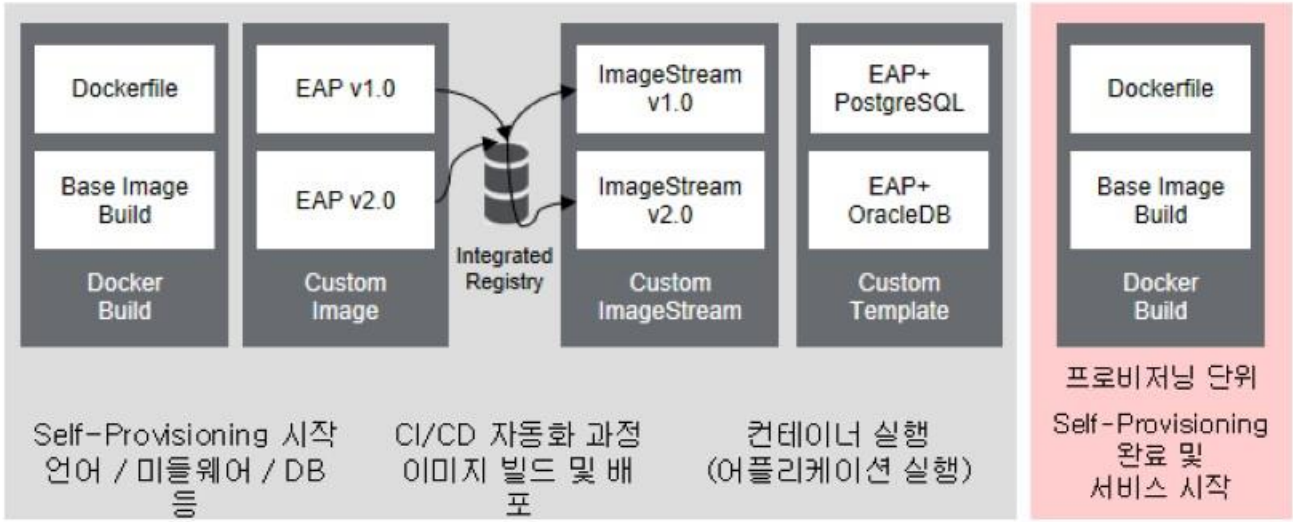
8. 빌드 이미지 생성

Red Hat

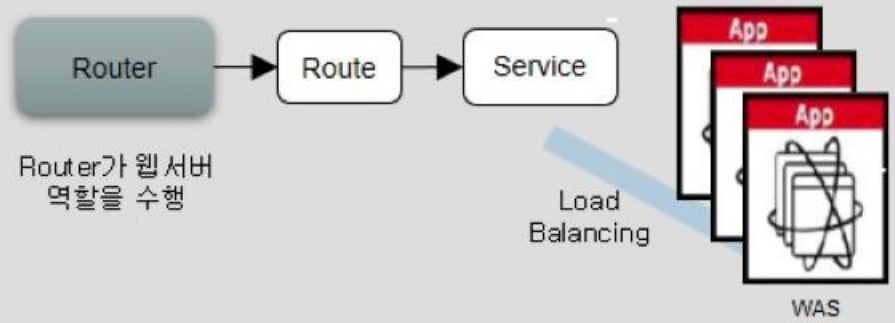
4. 주요 기능 > 템플릿/이미지/Yaml 배포

컨테이너를 사용자가 직접 생성 관리할 수 있습니다. 사용자는 미들웨어 자원 (WEB/WAS/DB) 및 Application을 선택 후 Build/Deploy 에 대한 자동화 단계를 수행합니다. 서비스를 위해 생성된 N개의 컨테이너는 서비스와 관리의 단위가 됩니다.

단계별 Self-Service 프로비저닝



WEB/WAS 통합 구성

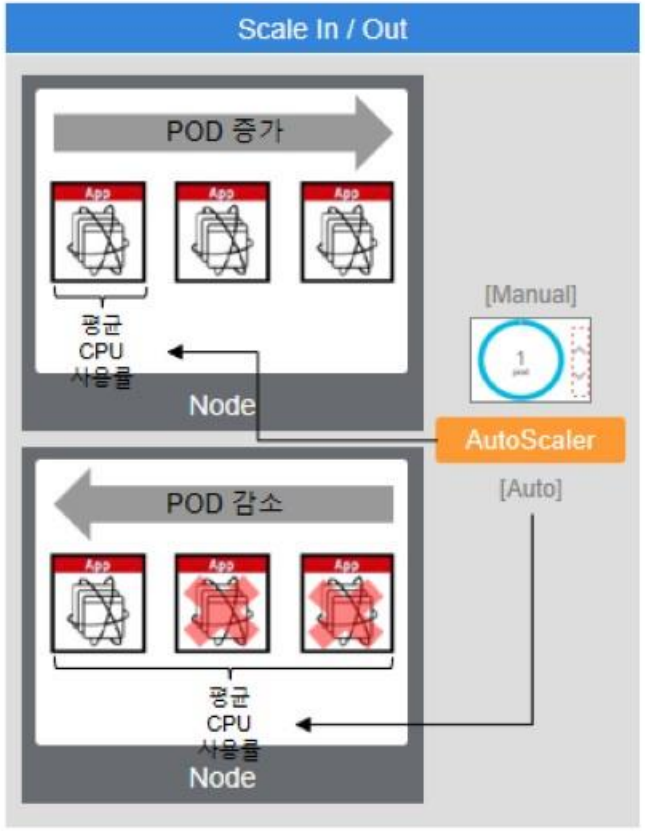


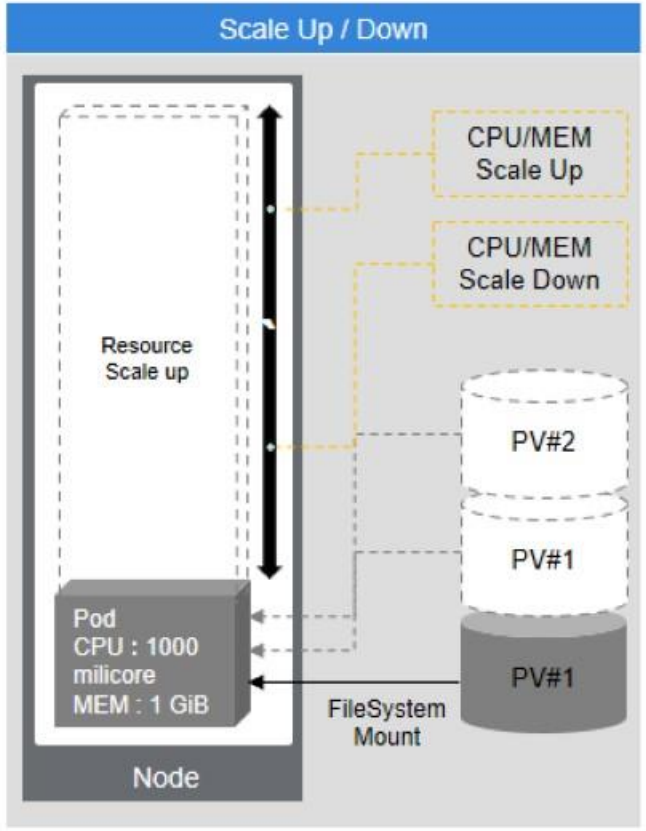
- 이미지/HTML 등 웹 서버에서 서비스하는 정적 리소스를 애플리케이션과 함께 배포해야 함
- 웹서버에 적용된 기능을 WAS에서 제공하는 기능으로 변경해야 함

Red Hat

4. 주요 기능 > 컨테이너 스케일링

Red Hat OpenShift Container Platform은 컨테이너의 Scale In/Out과 Scale Up/Down에 대해 Manual 방식과 Auto 방식을 제공합니다. Auto Scale Out의 경우 분 단위 CPU 평균사용 지표를 기준으로 HPA에 의해 Scale Out 됩니다.

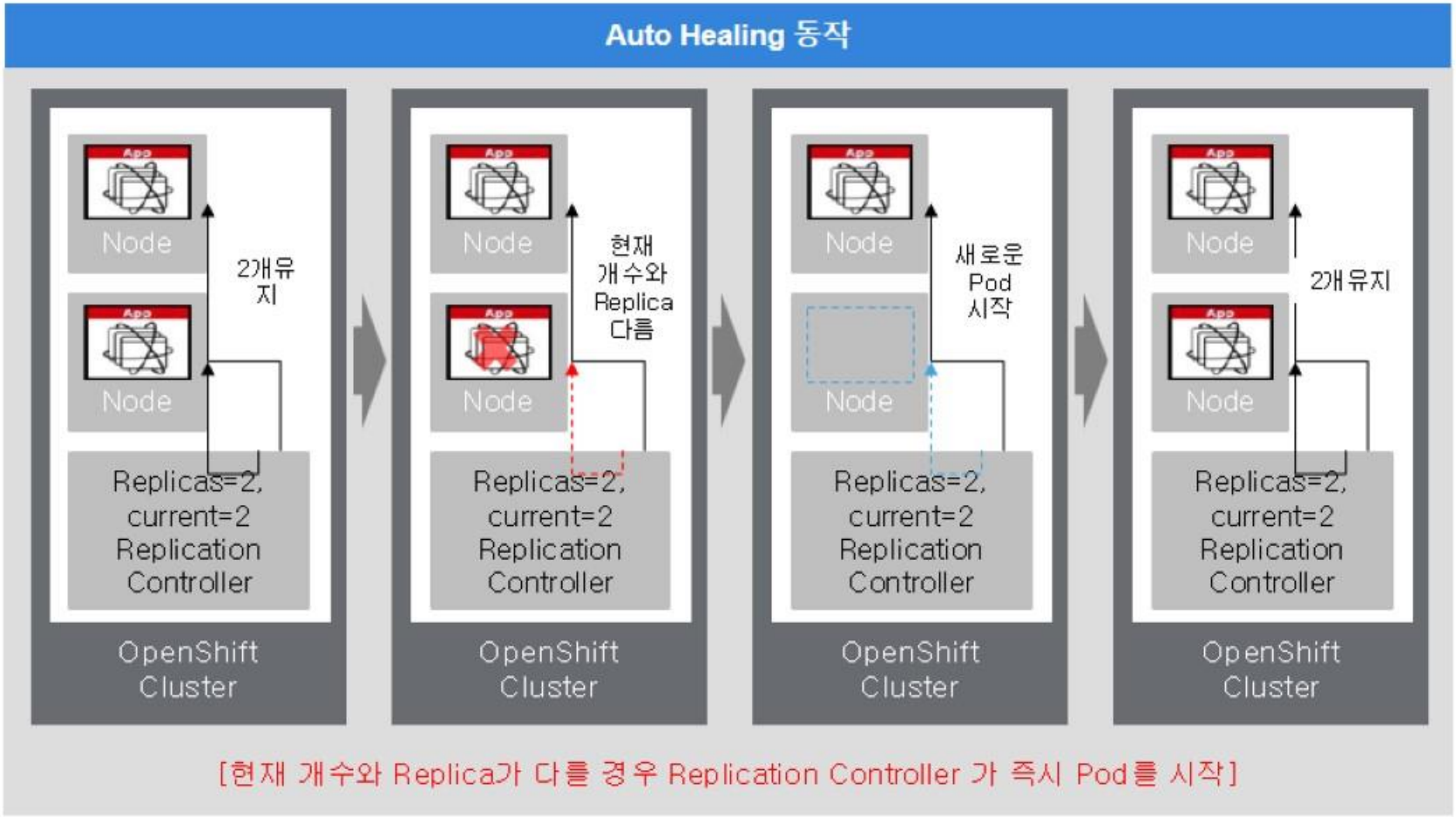




컨테이너 관리	자동화 여부	변경대상	담당	비고
Scale In / Out	Auto	Pod 개수	AutoScaler	Pod 평균 CPU 사용률 기준
	Manual	Pod 개수	운영자	운영자가 모니터링 및 판단
Scale Up / Down	Auto	CPU / MEM	AutoScaler	Pod 평균 CPU/MEM 사용률 기준
	Manual	CPU / MEM / DISK	운영자	CPU/MEM 증설시 Pod 재시작 필요

4. 주요 기능 > 컨테이너 고가용성

Red Hat OpenShift Container Platform은 가용한 Pod 개수를 유지하는 Auto Healing 특성을 통해 일부 Pod 장애시 서비스를 유지하는 고가용성을 제공합니다.



구성 설정

- 적정 replicas 개수를 Deployment Config 또는 Replication Controller 에 미리 설정

ex) Replication Controller 에 설정 예시

```
...
spec:
  replicas : 4
...
```

웹콘솔 설정

- 운영자가 웹콘솔에서 수동으로 조정

1. Manual Scale Out 시
   → Replicas 개수를 증가
2. Manual Scale In 시
   → Replicas 개수를 감소

CLI 설정

- 운영자가 CLI 명령을 사용하여 수동 조정

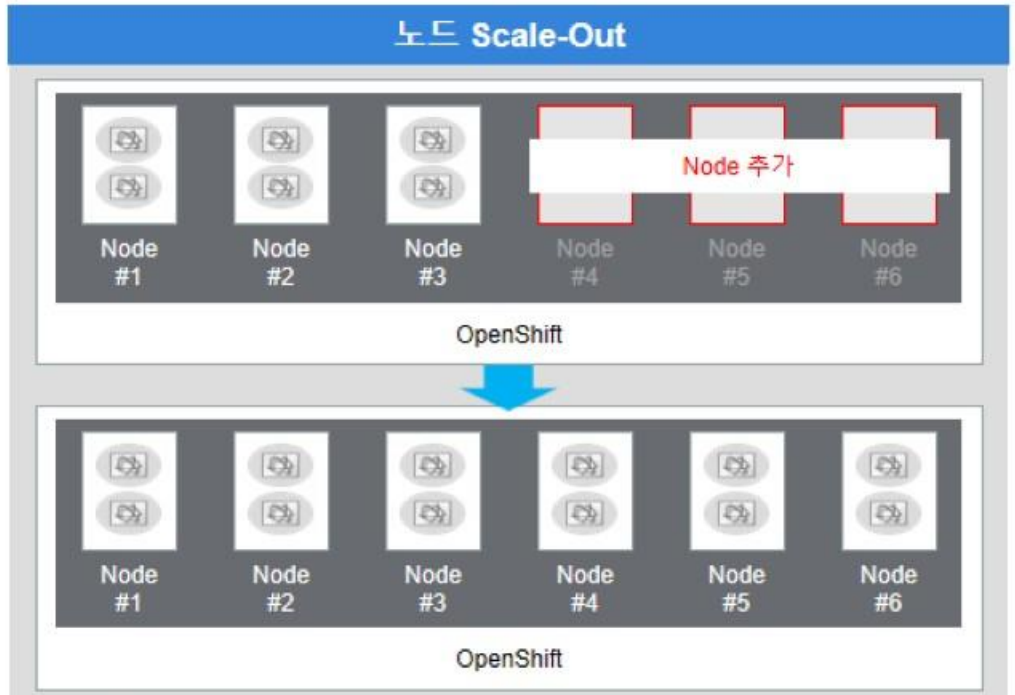
```
$ oc scale
dc/dcex --
replicas=3
```

1. oc scale 명령을 통해 replicas 개수를 지정
2. 지정 대상:
   deployments,
   replicaset,
   deployments config,
   replication controller

Red Hat

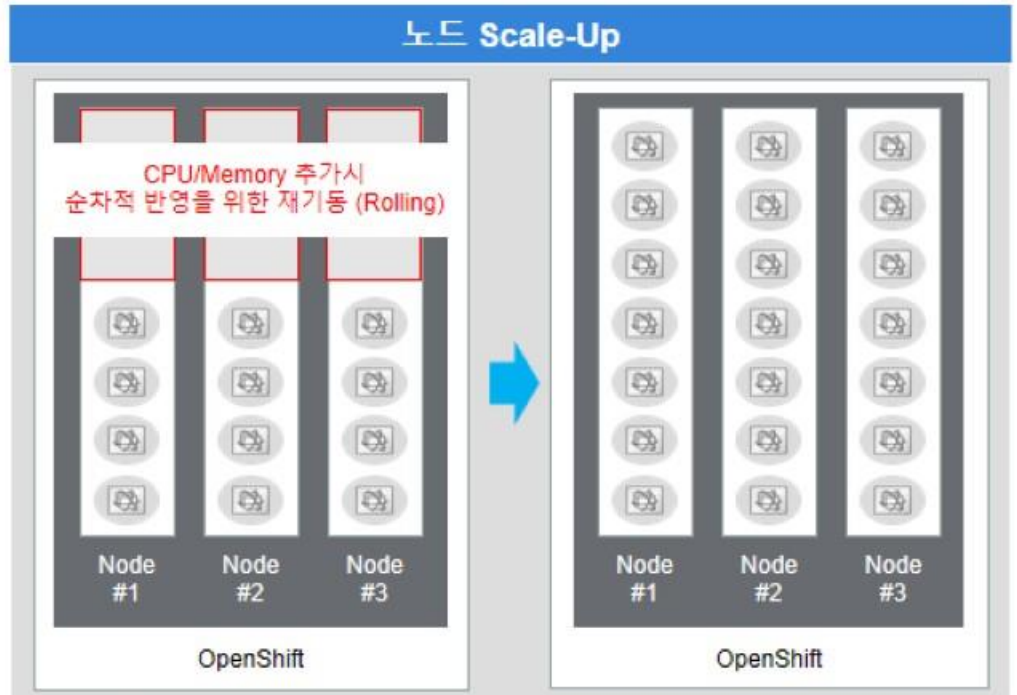
4. 주요 기능 > 클러스터 스케일링

Red Hat OpenShift Container Platform은 운영 시 발생 할 수 있는 부하에 따른 서버 증설 및 서버 Spec 조정시에 유연하게 확장할 수 있는 기능을 제공합니다.



Scale-Out

- 운영 서비스의 중단 없는 새로운 Node 추가 기능 제공.
- 컨테이너는 Scheduler에 의해 자동으로 배치.



Scale-Up

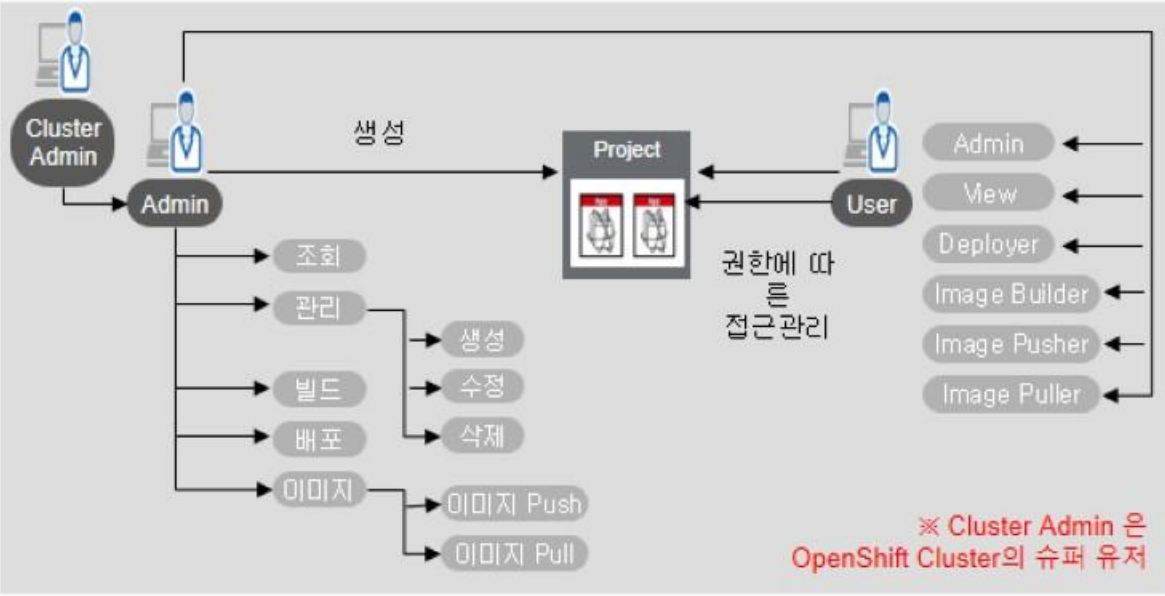
- 기존 Node의 CPU/Memory 증설을 통한 확장 으로, 서비스 노드의 컨테이너 수용량을 증가.
- 노드의 재기동 필요.

Red Hat

4. 주요 기능 > 권한 관리

Red Hat OpenShift Container Platform은 namespace(프로젝트) 기반으로 사용자에게 role을 부여합니다. 기본적으로 프로젝트 생성자는 해당 프로젝트의 admin 권한을 가지며 프로젝트 내 모든 작업을 수행합니다. admin은 다른 사용자에게 role을 세분화하여 부여하여 role 기반 접근관리를 수행합니다.

- **계정 생성 및 서비스 포탈 접속 및 계정 권한 부여**



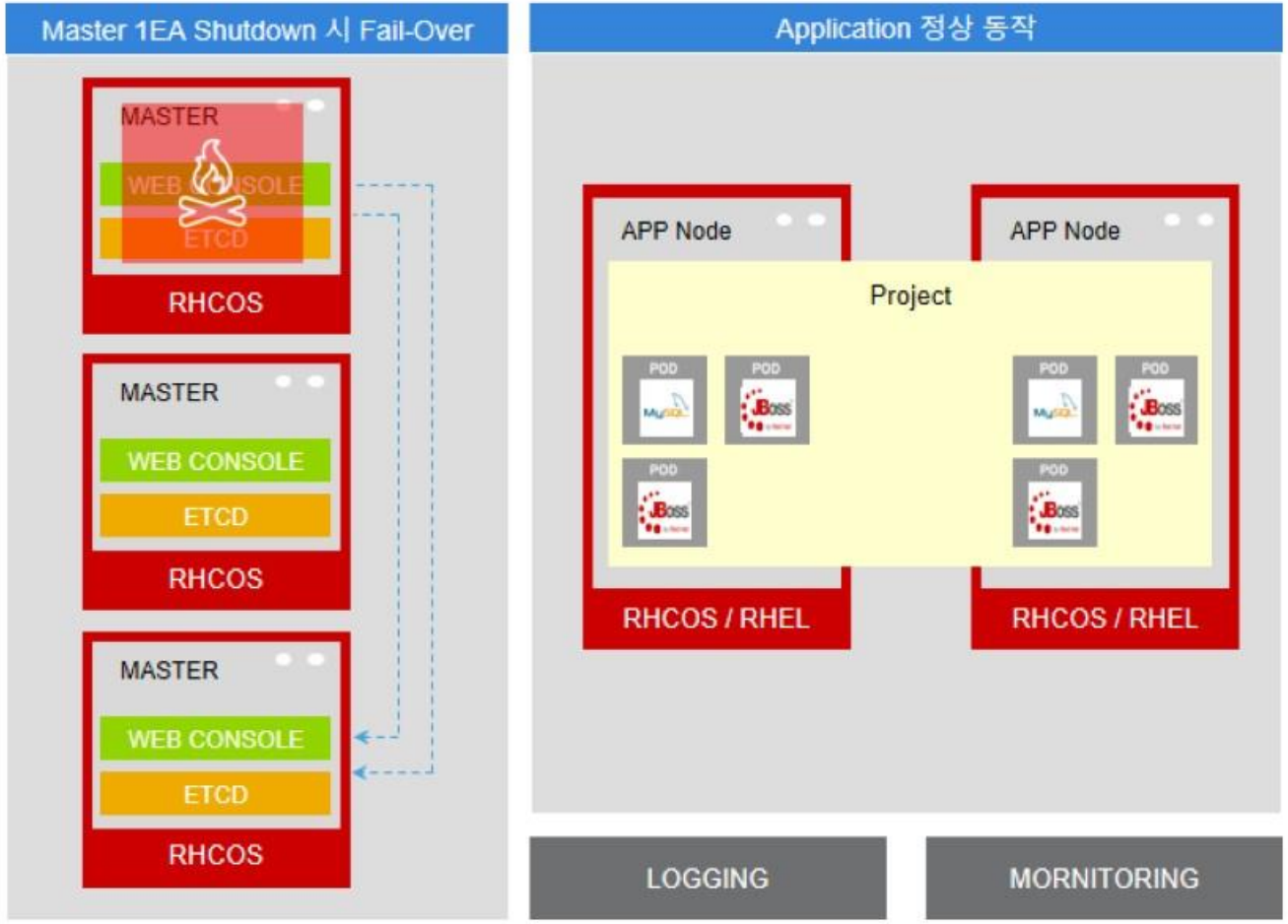
- **지원되는 ID 공급자**

ID 공급자	설명
HTPasswd	htpasswd를 사용하여 생성된 플랫 파일에 대해 사용자 이름 및 암호의 유효성을 확인하도록 htpasswd ID 공급자를 구성합니다.
Keystone	내부 데이터베이스에 사용자를 저장하는 OpenStack Keystone v3 서버와의 공유 인증을 지원하기 위해 OpenShift Container Platform 클러스터를 Keystone과 통합하도록 keystone ID 공급자를 구성합니다.
LDAP	단순 바인드 인증을 사용하여 LDAPv3 서버에 대해 사용자 이름 및 암호의 유효성을 확인하도록 ldap ID 공급자를 구성합니다.
기본 인증	사용자가 원격 ID 공급자에 대해 검증된 자격 증명을 사용하여 OpenShift Container Platform에 로그인할 수 있도록 기본 인증 ID 공급자를 구성합니다. 기본 인증은 일반적인 백엔드 통합 메커니즘입니다.
요청 헤더	X-Remote-User와 같은 요청 헤더 값에서 사용자를 확인하도록 요청 헤더 ID 공급자를 구성합니다. 일반적으로 요청 헤더 값을 설정하는 인증 프록시와 함께 사용됩니다.
GitHub 또는 GitHub Enterprise	GitHub 또는 GitHub Enterprise의 OAuth 인증 서버에 대해 사용자 이름 및 암호의 유효성을 확인하도록 github ID 공급자를 구성합니다.
GitLab	GitLab.com 또는 기타 GitLab 인스턴스를 ID 공급자로 사용하도록 gitlab ID 공급자를 구성합니다.
Google	Google의 OpenID Connect 통합을 사용하여 google ID 공급자를 구성합니다.
OpenID Connect	인증 코드 Flow를 사용하여 OpenID Connect ID 공급자와 통합하도록 oidc ID 공급자를 구성합니다.

Red Hat

4. 주요 기능 > 마스터 노드 장애시 가용성 제공

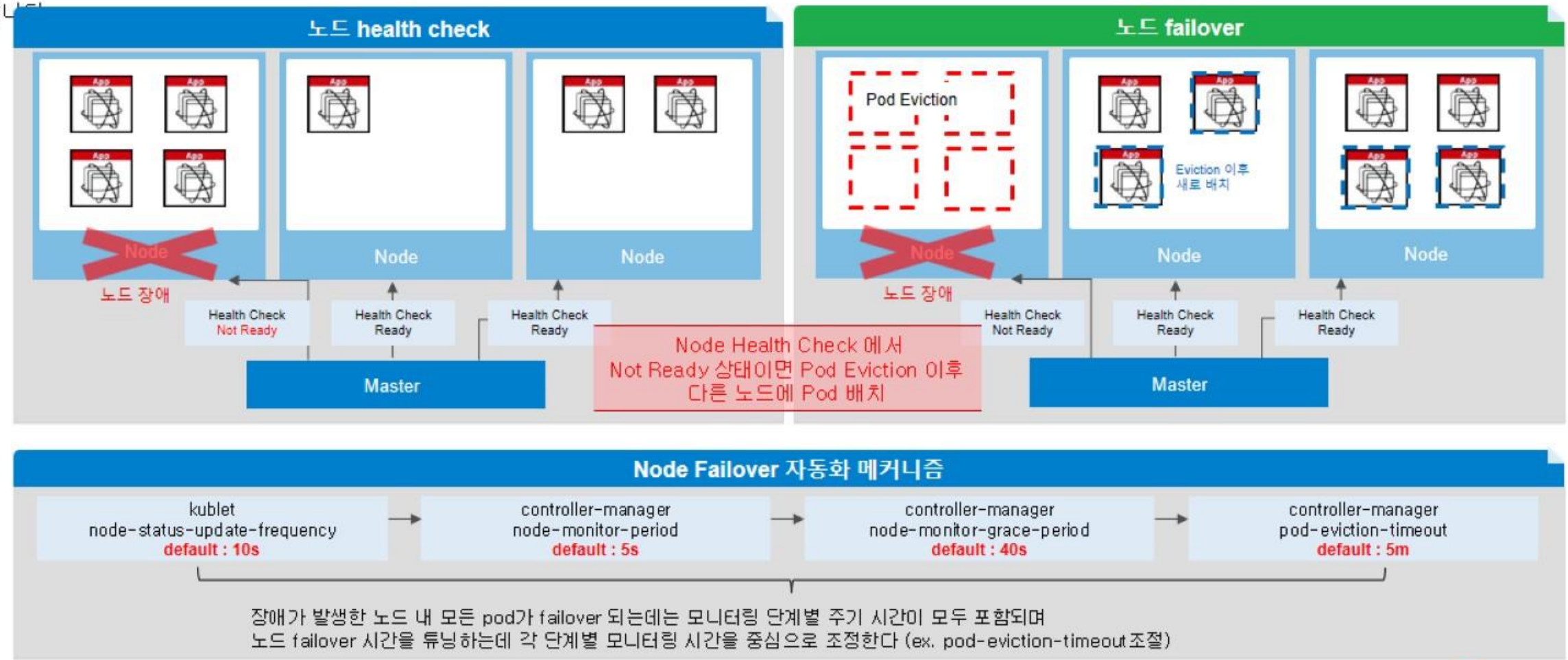
마스터 노드와 마스터 노드에 있는 Etcd 서비스는 3중화되어 장애 시에도 가용성을 제공합니다. 이미 노드에서 운영중인 애플리케이션은 마스터와 etcd 서비스 장애 여부와 상관없이 동작합니다.



장애 상황	복구 방안 및 특징
마스터 노드 1대 Down	● 마스터 노드 다중화로 한 노드에서 장애가 발생하더라도 다른 마스터 노드에서 처리 가능 ● 마스터 노드의 서비스 영향과 상관없이 현재 서비스 중인 **워커노드 상의 컨테이너는 정상적으로 서비스 됨**
마스터 노드 2대 Down	● 마스터 노드 2대 다운 시에도 **워커 노드의 서비스에는 영향이 없음** ● 마스터 노드 2대 재 시작 ● 재시작으로 복구 불가한 경우 정상인 마스터 노드 1대로 **etcd 복구 가능**
마스터 노드 3대 Down	● 마스터 노드 3대 다운 시에도 **워커노드의 서비스에는 영향이 없음** ● 기존 etcd 백업을 사용하여 이전 클러스터 상태로 복원 ● etcd 백업을 사용하여 단일 컨트롤 플레인 호스트 복원 ● etcd 클러스터 Operator는 나머지 컨트롤 플레인 호스트에 대한 스케줄링 처리

4. 주요 기능 > 워커 노드 장애시 가용성 제공

Red Hat OpenShift Container Platform은 워커 노드 장애 시, Node Health Check와 Node Failover 자동화 메커니즘을 통해 노드 내 운영 중인 컨테이너를 다른 가용한 노드로 이동시켜 서비스 장애를 방지합니다. 레지스트리, 메트릭 서비스 등도 컨테이너로 동작하기 때문에 Pod / Node Auto Healing 기능으로 가용성을 제공합니다.



Red Hat

4. 주요 기능 > CI/CD

Red Hat OpenShift Container Platform은 CI/CD 파이프라인 도구인 Jenkins와 Openshift Pipelines를 모두 제공합니다. Tekton 기반의 Openshift Pipeline은 클라우드 네이티브 개발 플렛폼인 OpenShift 에 최적화된 파이프라인 도구입니다. CI/CD 구현 방안에 알맞는 도구를 선택하여 활용하실 수 있습니다.

Traditional CI/CD
가상 시스템에 맞게 설계
유지보수를 위한 IT 운영 필요
엔진에서 플러그인을 공유
업데이트 주기가 정의되지 않은 플러그인 종속성
Kubernetes 리소스와의 상호 운용성 없음
지속적인 관리자의 관리가 필요
엔진 이미지 빌드 구성

Cloud-Native CI/CD
컨테이너 및 쿠버네티스용으로 설계
운영 오버헤드 없이 서비스형 파이프라인
파이프라인이 서로 완전히 격리됨
컨테이너 이미지로 라이프사이클 관리
기본 Kubernetes 리소스
플랫폼에서 지속적으로 관리
ConfigMaps를 통해 구성됨

Jenkins

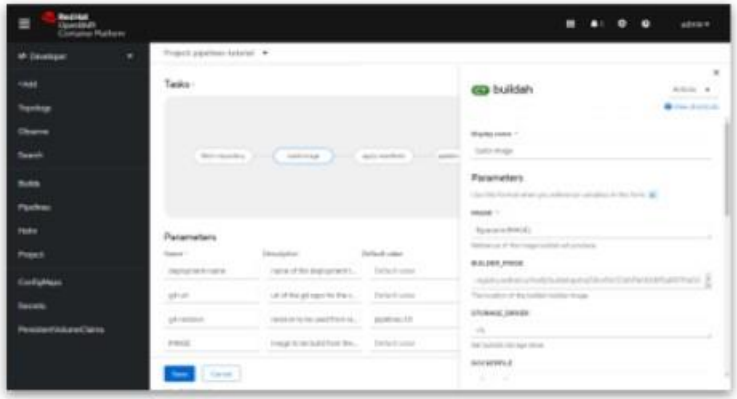
Red Hat OpenShift Pipelines

Red Hat

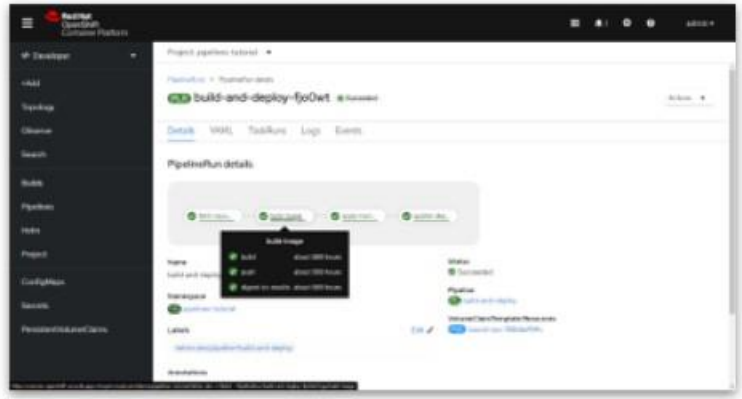
4. 주요 기능 > CI/CD

Red Hat OpenShift Pipelines는 OpenShift 웹 콘솔과 CLI 명령어를 통해 파이프라인의 생성, 관리, 상태 모니터링할 수 있는 기능을 제공합니다. 또한 Visual Studio Code, IntelliJ 와 같은 다양한 IDE 개발 도구와 연계할 수 있는 플러그인을 제공하여 OpenShift를 활용한 컨테이너 개발 생산성을 향상시켜 줍니다.

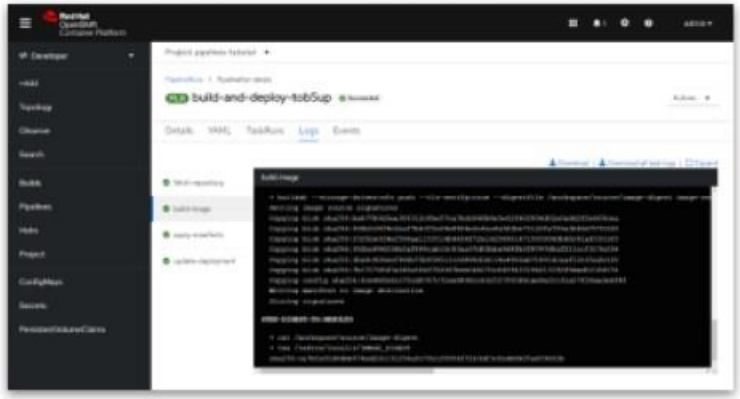
- **Web Console을 통한 파이프라인 생성**



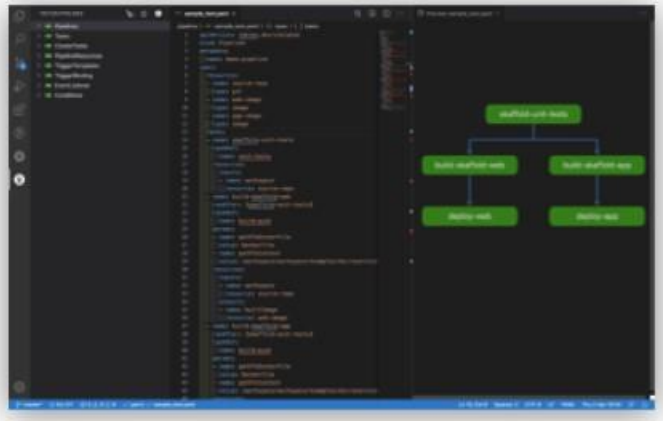
- **파이프라인 현황 확인**



- **파이프라인 로그 모니터링**



- **VS Code 플러그인**



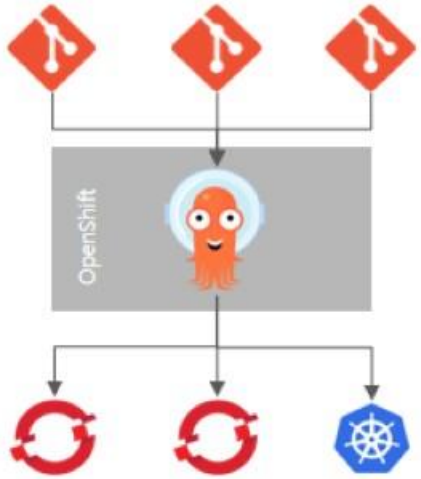
- **IntelliJ 플러그인**

Red Hat

4. 주요 기능 > CI/CD

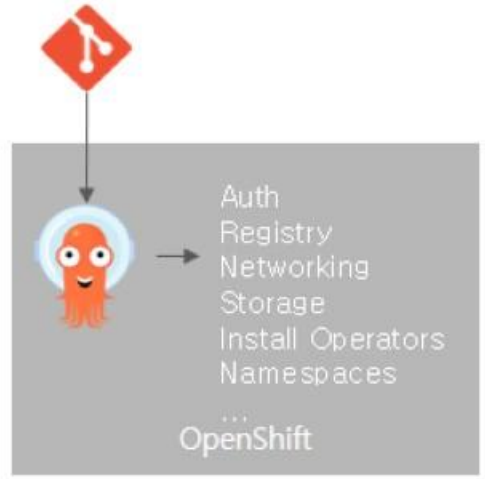
Red Hat OpenShift GitOps는 개발, 스테이징, 프로덕션과 같은 다양한 환경의 다양한 클러스터에 애플리케이션을 배포할 때 애플리케이션의 일관성을 보장합니다. 이를 통해 다중 클러스터, 단일 클러스터 및 어플리케이션 단위로 관리하기 위한 반복 가능한 배포 전략을 구현할 수 있습니다.

Red Hat OpenShift GitOps를 통한 유연한 배포 전략



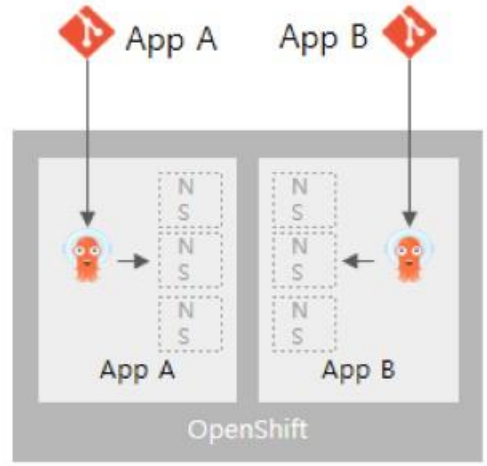
Central Hub (Push)

중앙 Argo CD는 Git repository 콘텐츠를 원격 오픈시프트 및 k8s 클러스터로 푸시합니다.



Cluster Scoped (Pull)

클러스터 범위 Argo CD는 클러스터 서비스 구성을 오픈시프트 클러스터로 가져옵니다.



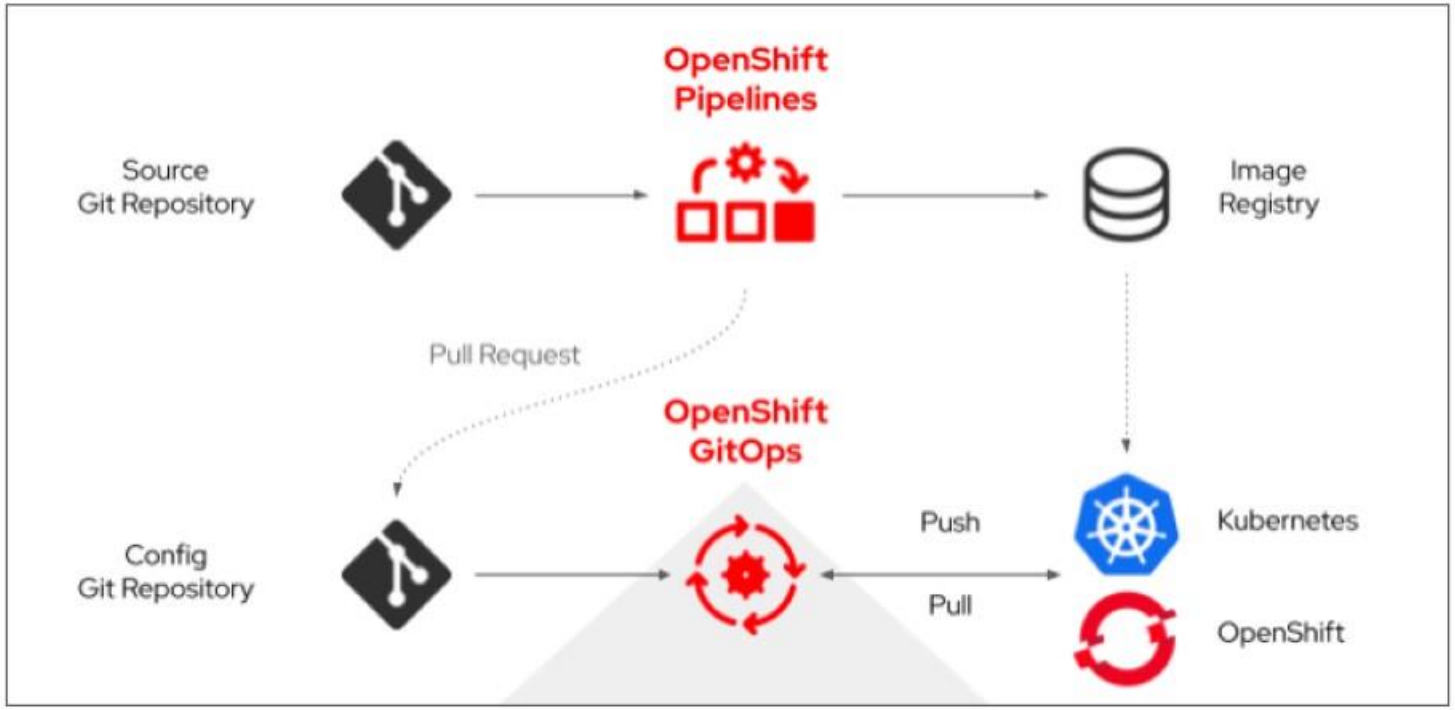
Application Scoped (Pull)

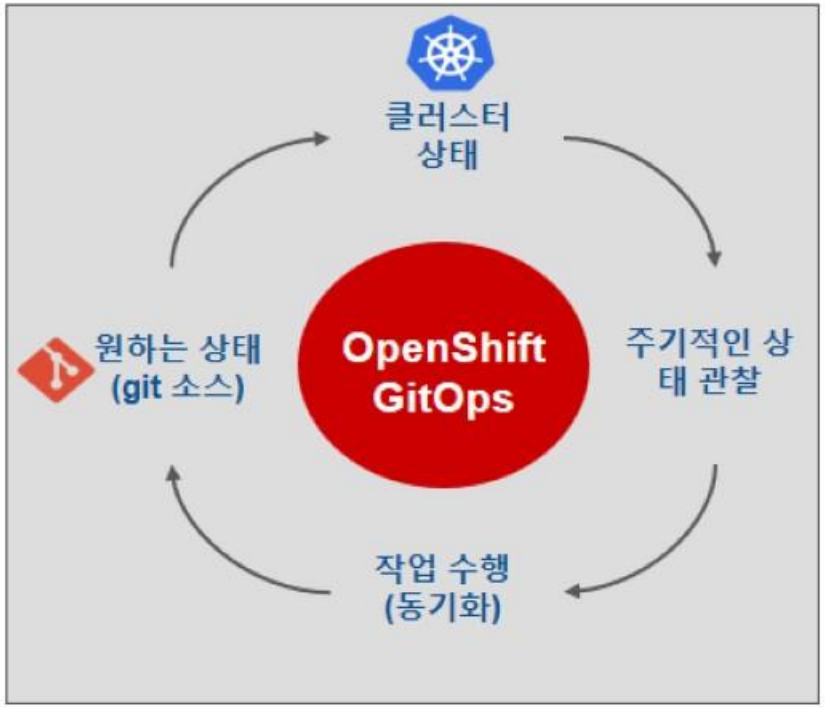
애플리케이션 범위의 Argo CD는 애플리케이션 배포 및 구성을 앱 네임스페이스로 가져옵니다.

Red Hat

4. 주요 기능 > CI/CD

Red Hat OpenShift Container Platform은 OpenShift GitOps와 OpenShift Pipelines를 통해 소스 컴파일, 컨테이너 이미지 빌드, 서비스 배포 등 전반적인 CICD 빌드 배포 파이프라인의 각 단계들을 지원하고 자동화합니다.

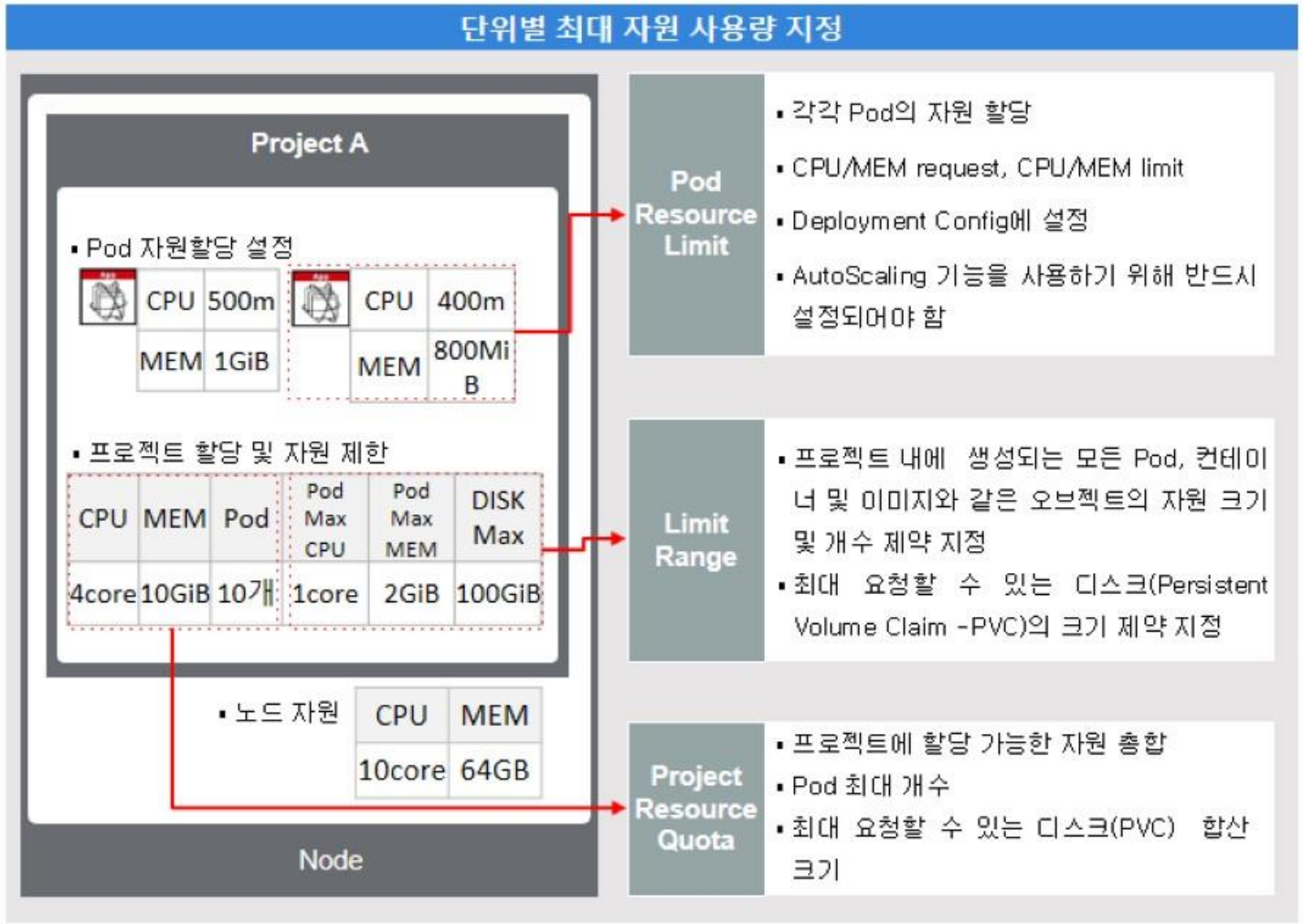




Red Hat

4. 주요 기능 > 자원 할당 관리

Red Hat OpenShift Container Platform은 프로젝트 혹은 Pod 단위로 자원을 할당하고 관리합니다. 하나의 프로젝트에서 가용할 수 있는 컨테이너 개수를 제한하고 물리적인 자원 한계를 상회하지 않도록 하여 서비스의 품질을 보장할 수 있는 다양한 자원 설정 방법을 제공합니다.




```
kind: LimitRange
apiVersion: v1
metadata:
  name: limits
spec:
  limits:
  - type: Pod
                max:
                cpu:
100m
                memor
y: 750Mi
                min:
                cpu:
10m
                memor
y: 5Mi
  - type: Container
                max:
                cpu:
100m
                memor
y: 750Mi
                min:
                cpu:
10m
                memor
y: 5Mi
                default
                :
                cpu:
100m
                memor
y: 5Mi
```

※참고사항

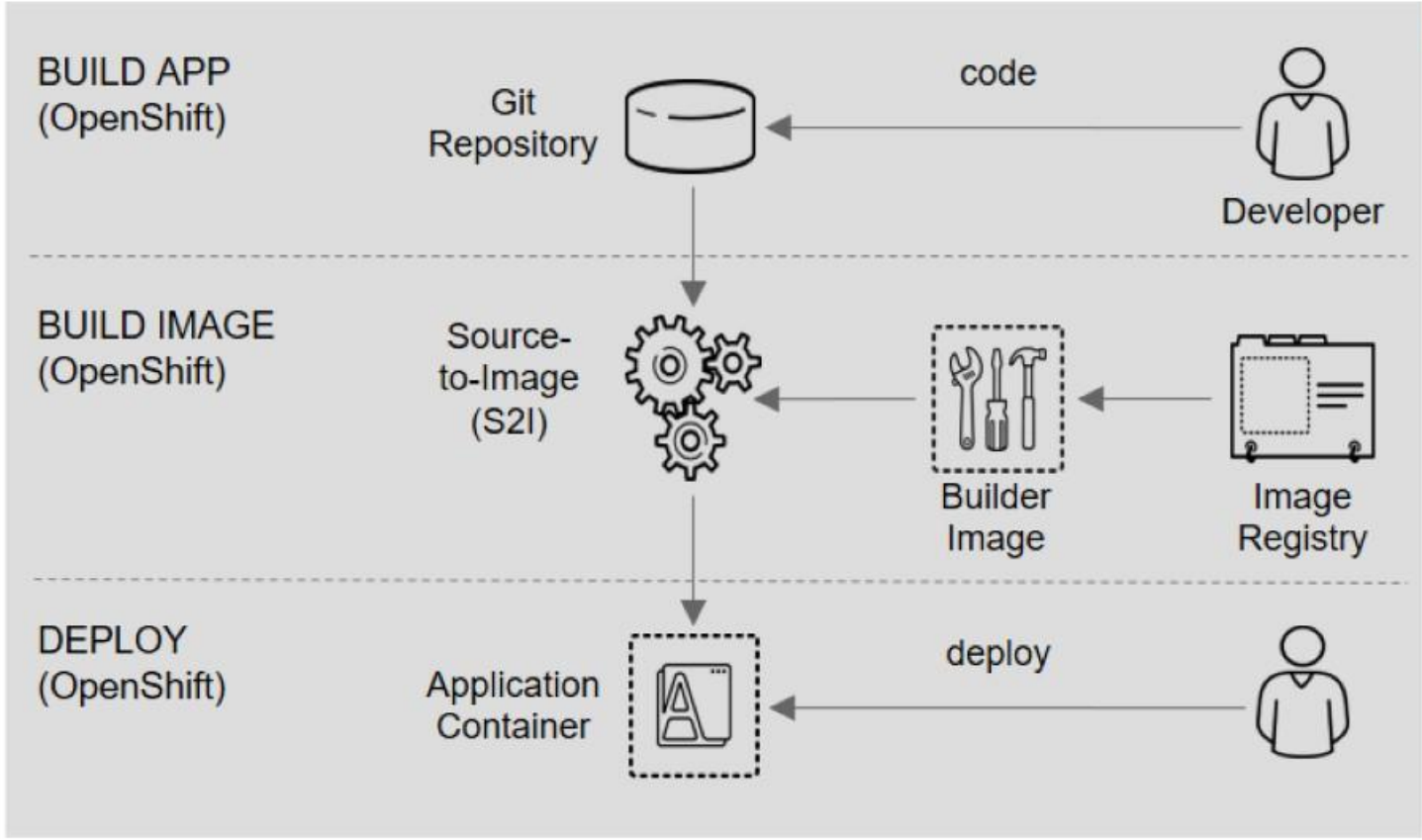
- 자원 사용량 제약을 운영 중단 없이 반영 가능함
  - 단, Pod의 Resource Limit은 pod를 재시작 해야 반영됨
- Pod는 1 core 당 최대 10개 이내 생성을 권장함
  - Pod 개당 100 milicore 이상 할당 권장
- WAS (JBoss EAP)의 경우 힙메모리는 pod 할당 메모리의 50%(default, 변경 가능)로 자동 산정되며, WAS가 필요로 하는 힙 메모리를 분석하여 pod에 할당할 메모리 크기를 산정해야함

Red Hat

4. 주요 기능 > S2I 빌드

Red Hat OpenShift Container Platform은 다양한 방법으로 컨테이너 이미지를 생성할 수 있습니다. 특히 S2I (Source to Image) 빌드를 통해 컨테이너 이미지를 만들 수 있는 기능을 제공합니다.

소스코드를 통해 컨테이너 이미지를 생성



S2I 베이스 이미지 정보 (일부)

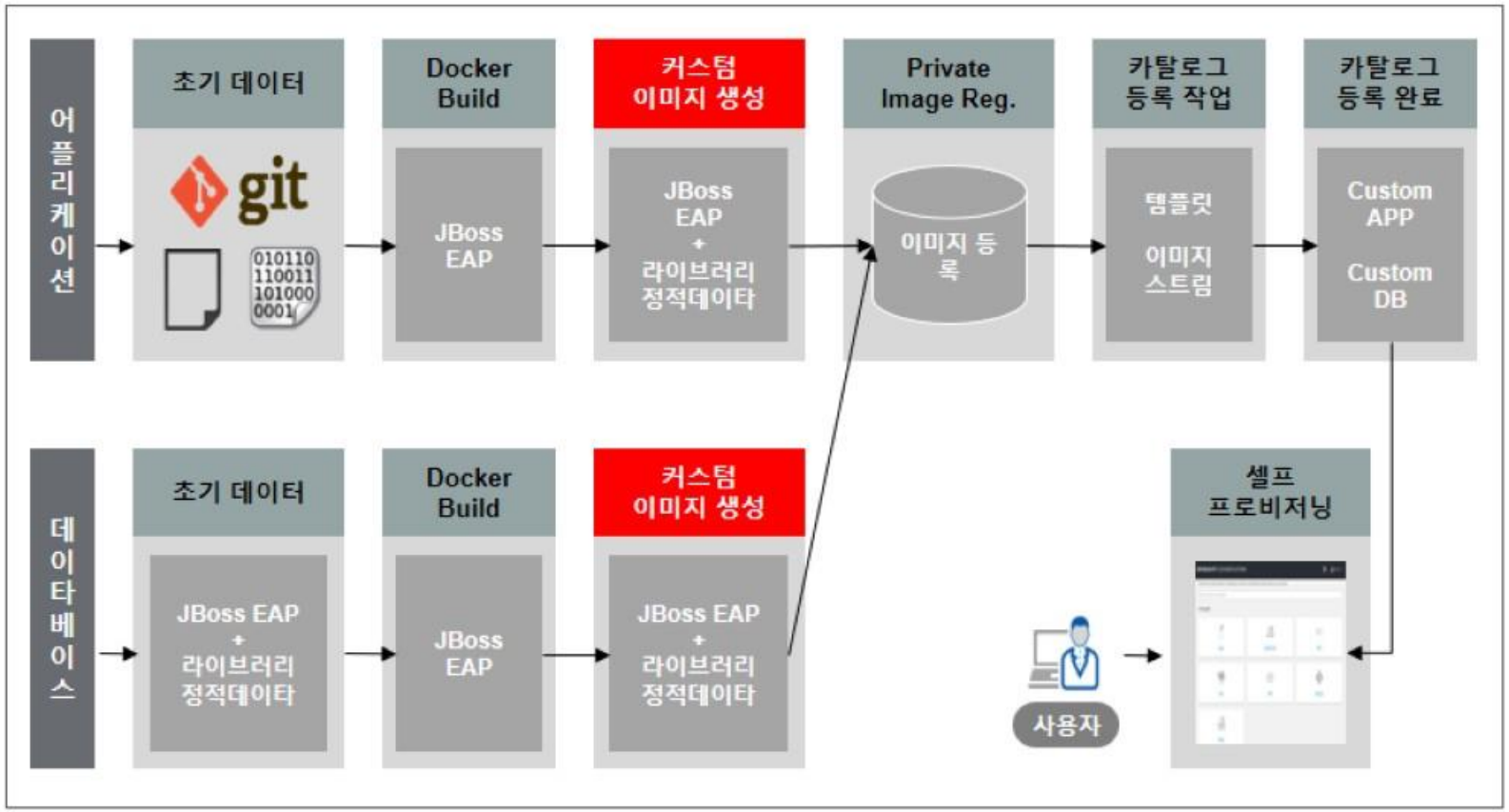
이미지 이름	TAG
dotnet	3.1,3.1-el7,3.1-ubi8,5.0,5.0-ubi8,6.0
dotnet-runtime	3.1,3.1-el7,3.1-ubi8,5.0,5.0-ubi8,6.0
driver-toolkit	410.84.202204050541-0,latest
fis-java-openshift	1.0,2.0
fis-karaf-openshift	1.0,2.0
golang	1.16.7-ubi7,1.16.7-ubi8,latest
httpd	2.4,2.4-el7,2.4-el8,latest
java	11,8,latest,openjdk-11-el7,openjdk-11-ubi8
jboss-eap74-openjdk11-openshift	7.4.0,latest
jboss-webserver56-openjdk11-tomcat9-openshift-ubi8	5.6.0,latest
jboss-webserver56-openjdk8-tomcat9-openshift-ubi8	5.6.0,latest
nginx	1.18-ubi7,1.18-ubi8,1.20-ubi7,1.20-ubi8
nodejs	14-ubi7,14-ubi8,14-ubi8-minimal,16-ubi8
perl	5.26-ubi8,5.30,5.30-el7,5.30-ubi8
php	7.3,7.3-ubi7,7.4-ubi8,latest
python	2.7,2.7-ubi7,2.7-ubi8,3.6-ubi8,3.8
ruby	2.5-ubi8,2.6,2.6-ubi7,2.6-ubi8,2.7

Red Hat

4. 주요 기능 > Docker Build

Provisioning시에 초기 Data를 포함한 Container를 배포하기 위해서는 Base Docker Image를 기준으로 Image 작업을 선행하거나 Docker Build 시에 동적으로 Data가 생성될수 있도록 Dockerfile 또는 Containerfile에 초기 Data 생성에 필요한 Layer 작업이 필요 합니다. Container 시작 시에 외부 Data로 부터 Import하는 방법도 가능합니다.

커스텀 이미지 Self-Service 프로비저닝



※추가 필요 작업

- 초기 데이터를 포함한 이미지를 생성하기 위해서는 아래 2가지 작업은 반드시 필요하다.

  ex) dockerfile 컨테이너 기동시 스크립트 실행 명령어 추가

  CMD ["/app/work/init.sh"]

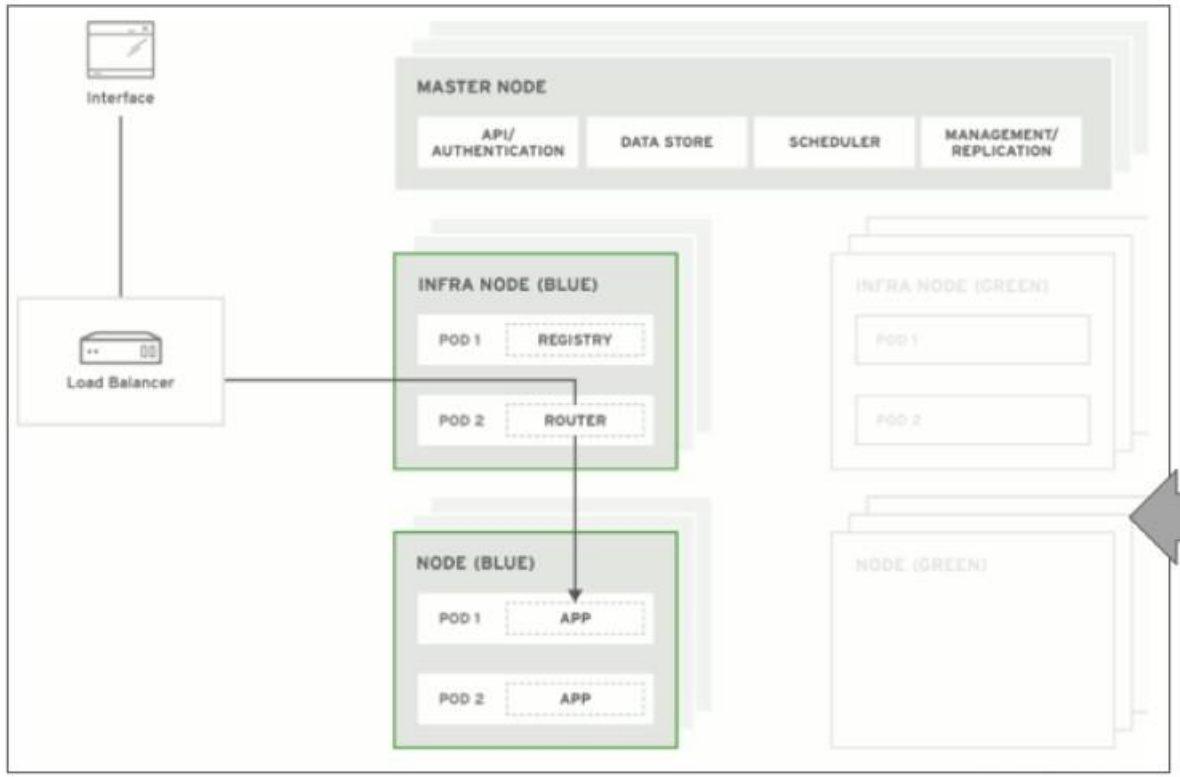
  또는

  ENTRYPOINT ["/app/work/init.sh"]

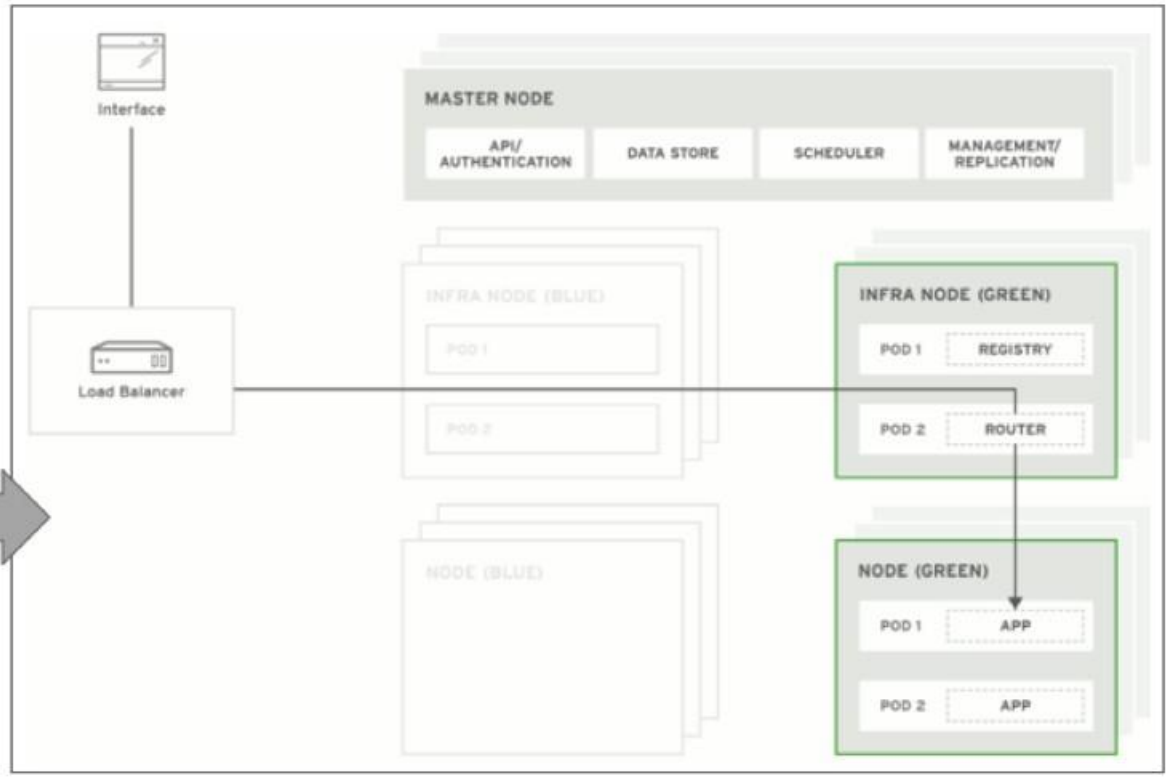
Red Hat

4. 주요 기능 > 고급 배포 전략

Red Hat OpenShift Container Platform은 고급 배포 전략으로 Blue-Green 배포를 지원합니다. Blue-Green 배포는 두 가지 동일한 환경을 갖는 것을 말하며, 트래픽을 전달할 수있는 OpenShift의 라우터로 해당 기능을 제공합니다.



Blue 서비스 상태

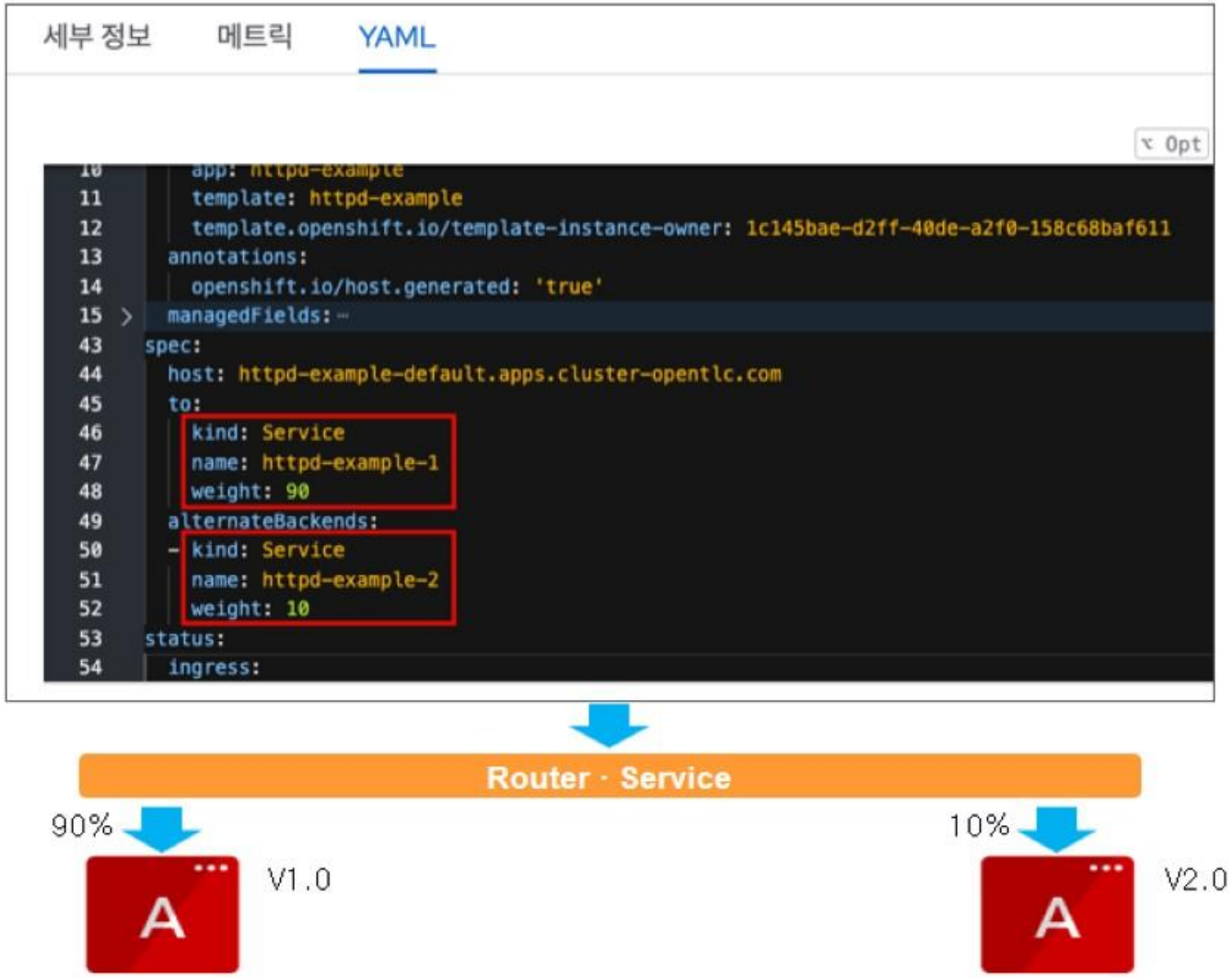


Green 서비스 상태

Red Hat

4. 주요 기능 > 고급 배포 전략

Red Hat OpenShift Container Platform은 고급 배포 전략으로 Canary Deployment를 지원합니다. 웹 콘솔을 통해 서비스들의 트래픽 비율을 조정할 수 있으며, 신규 버전에 소규모의 트래픽을 보내서 테스트 할 수 있습니다.

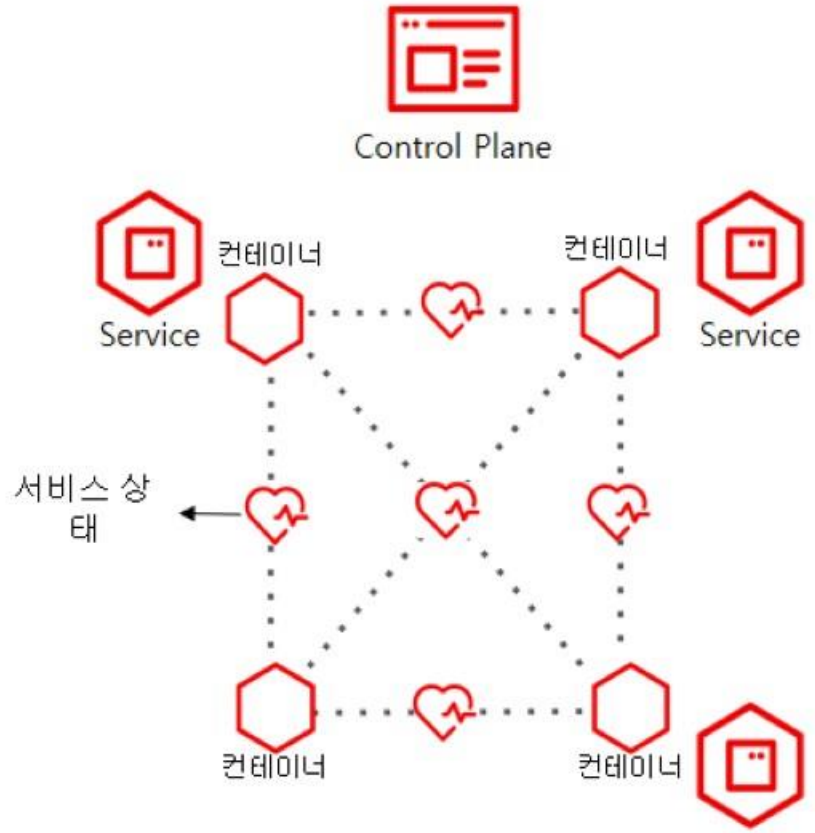


Red Hat

4. 주요 기능 > 서비스 메시

Red Hat OpenShift Container Platform은 Istio 기반의 OpenShift ServiceMesh를 통해 Envoy Sidecar proxy를 삽입하여 Circuit breaker, 비율 기반 Traffic 분배, 우회 처리 등의 고급 라우팅 룰을 처리할 수 있습니다.

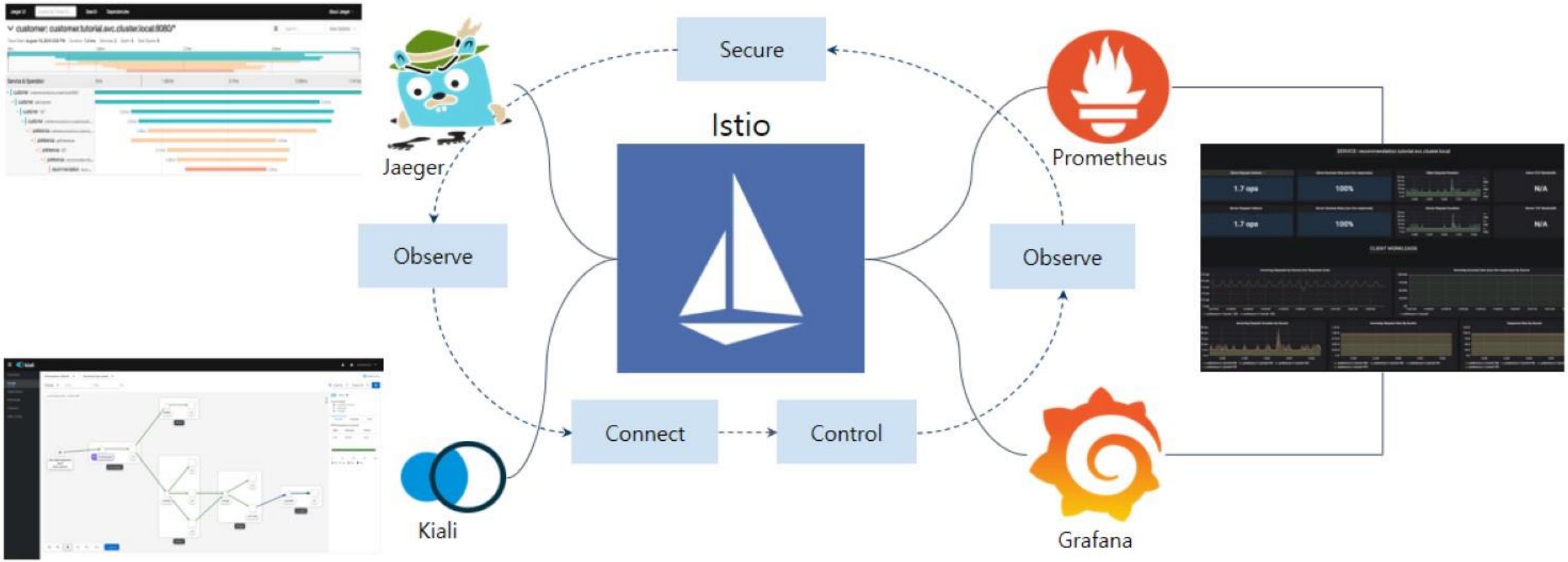
- 통신은 프록시를 통해 이루어지므로 소스 코드 변경없이 시스템에 회복력 제공 가능
- 서킷브레이커, 타임아웃
  - 함께 작동하여 서비스 간의 계단식 오류 방지
- 재시도(Retry) 설정
  - 요청 실패 선언 전 자동 재시도 횟수를 설정하여 네트워크 및 서비스의 일시적 실패를 극복 할 수있는 기능을 제공
- 장애 주입(Fault injection)
  - 불가피하게 장애가 발생할 때 서비스가 어떻게 수행되는지 검증하는 기능을 제공
  - chaos engineering을 가능하게 함



Red Hat

4. 주요 기능 > 서비스 메시

품질 모니터링(Quality Monitoring)과 분석 기능을 위해 Red Hat OpenShift ServiceMesh는 MSA에 필요한 서비스 메시 기능과 더불어 서비스 Tracing을 위한 Jaeger와 어드민 기능 관리를 위한 Kiali, 모니터링을 위한 서비스매시 전용 Prometheus와 Grafana 대시보드를 함께 제공합니다.

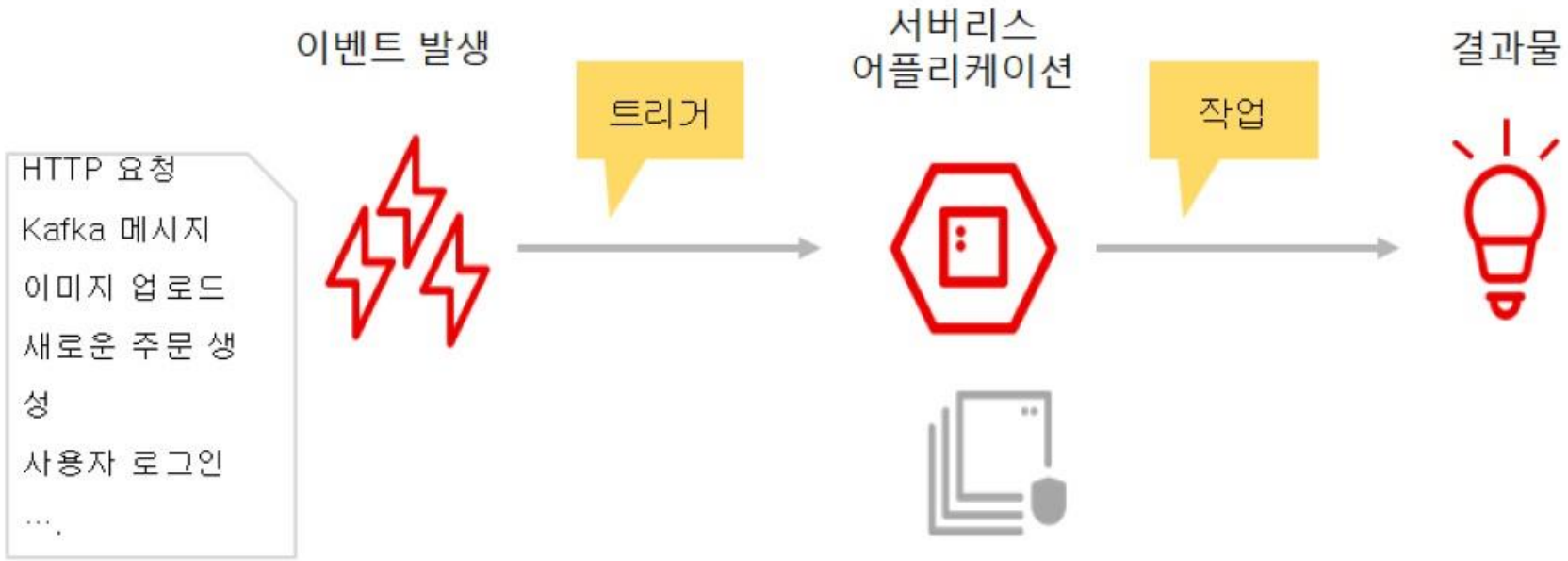


Red Hat

4. 주요 기능 > 서버리스

Red Hat OpenShift Container Platform은 Knative 기반의 OpenShift Serverless를 통해 어플리케이션 현대화 및 관리 최소화, 리소스 사용 최적화 등의 이점을 제공합니다.

간단하면서 강력한 이벤트 기반의 어플리케이션



서버리스 패턴의 이점

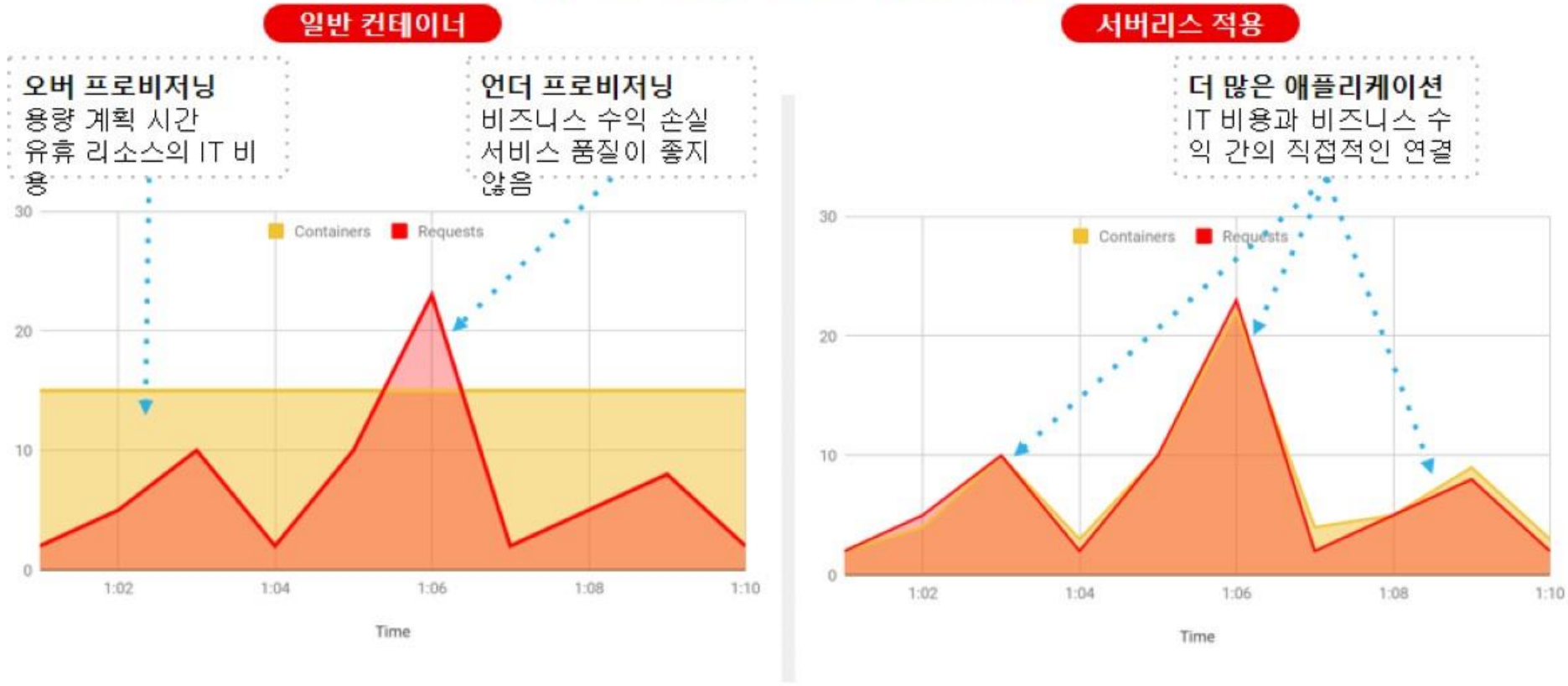
- Auto-scaling / Load balancers 설정 불필요
- 애플리케이션의 빠른 배포 (Deploy)
- Event Driven Architectures (EDA) 패턴의 활용가능
- 조직내 IT와 비용과 연계
- 서버리스 컨테이너로 기존 애플리케이션의로 전환

Red Hat

4. 주요 기능 > 서버리스

Red Hat OpenShift Container Platform은 Knative 기반의 OpenShift Serverless를 통해 어플리케이션 현대화 및 관리 최소화, 리소스 사용 최적화 등의 이점을 제공합니다.

서버리스 적용시 리소스 사용량 최적화



Red Hat


openmaru