Kubernetes 환경에서의 Volume 배포와 데이터 관리의 유연성 확보

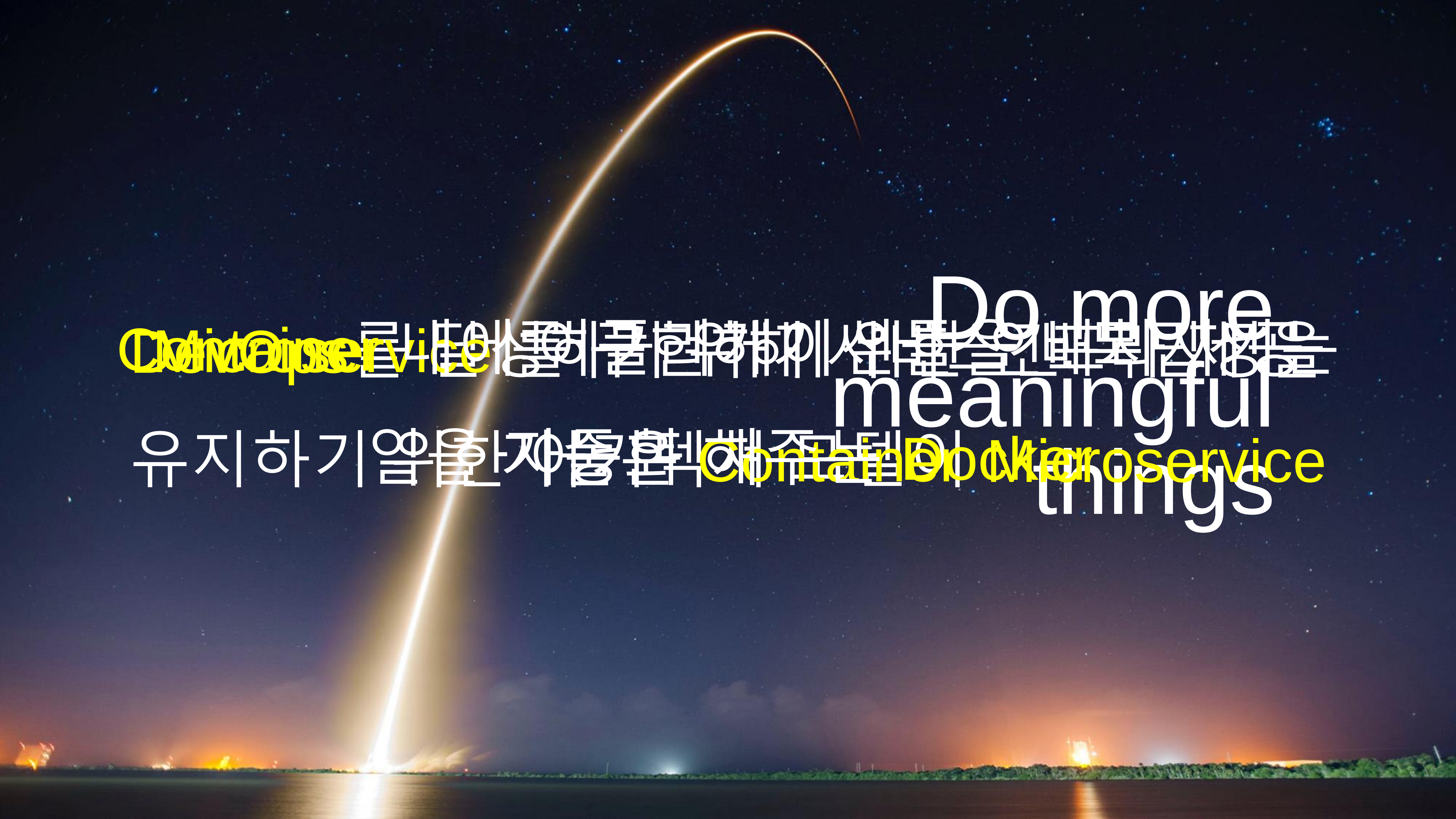
Trident case study

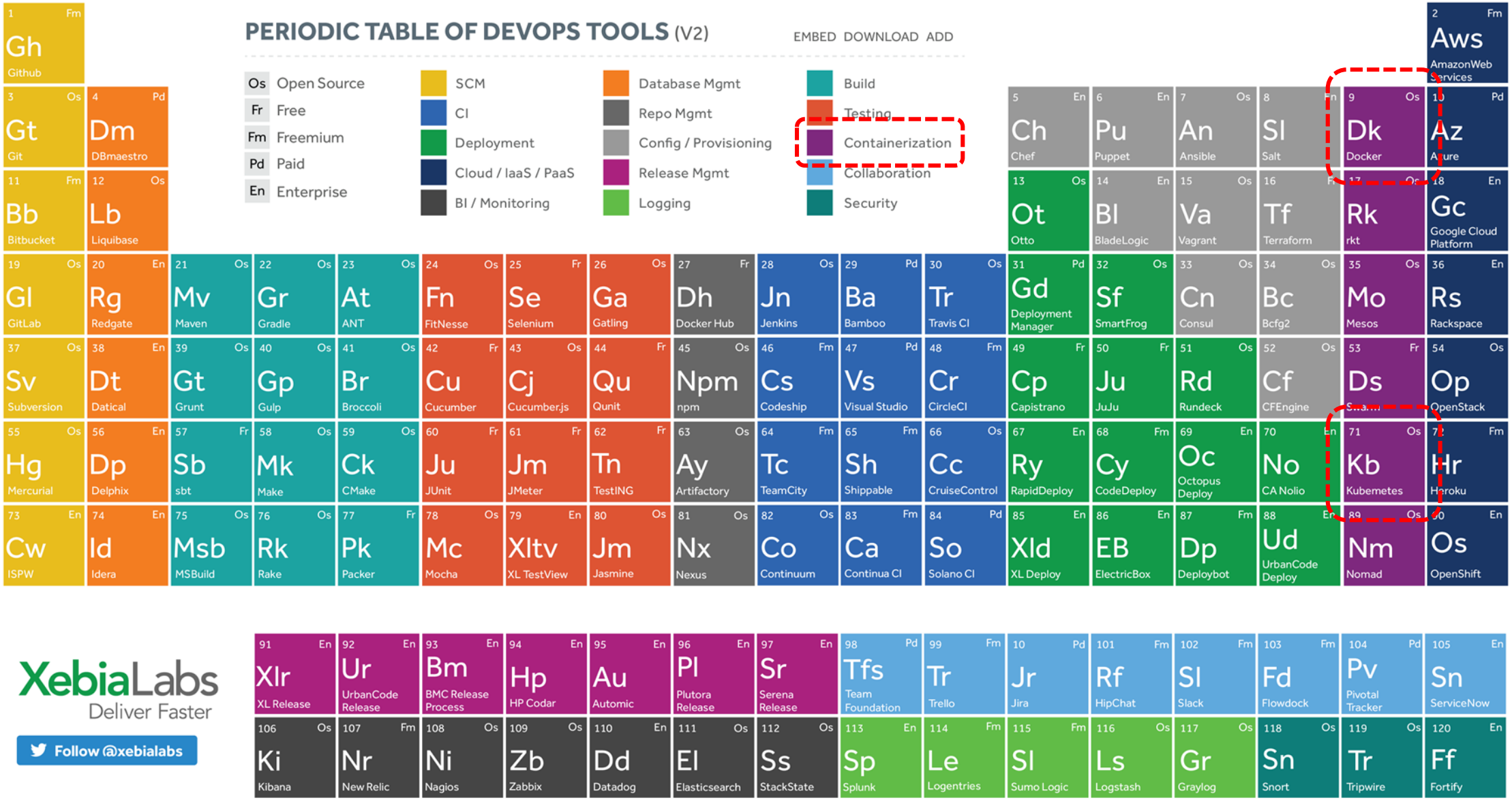
June 2018

NetApp 김진학 부장 / LG CNS 장다성 선임

LG CNS

NetApp


Do more
meaningful
things
유지하기
Microservice

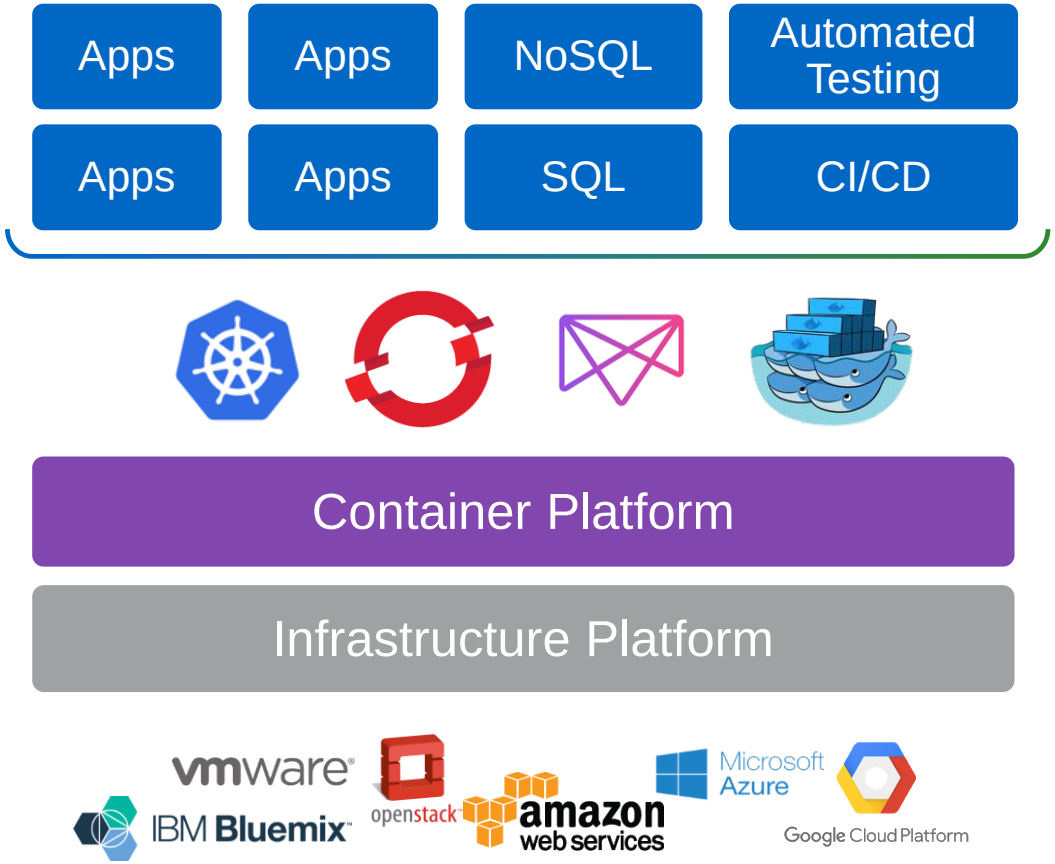


XebiaLabs
Deliver Faster

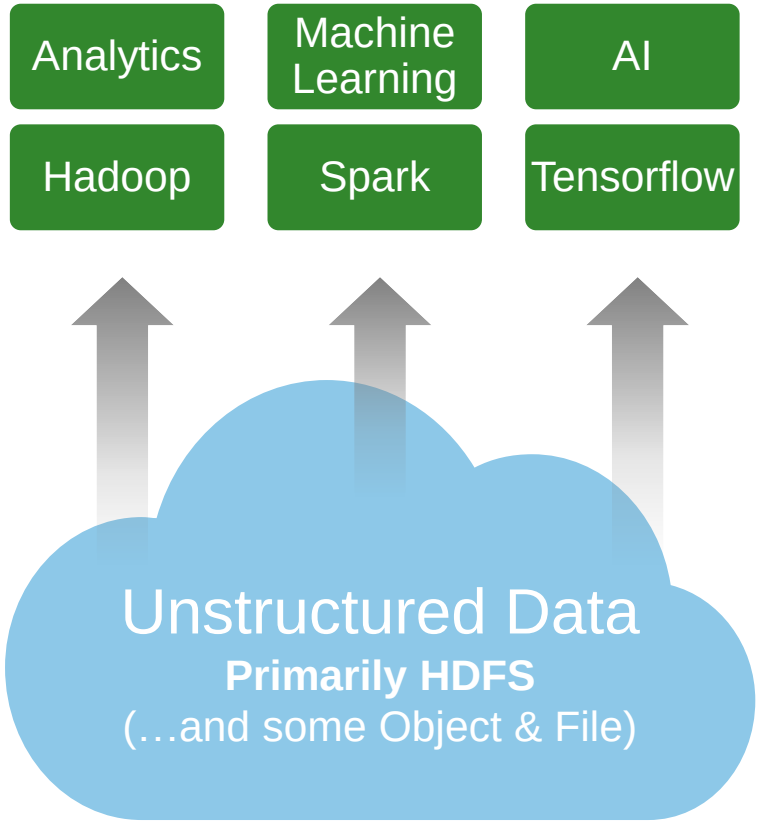
Follow @xebialabs

NetApp

어플리케이션 환경



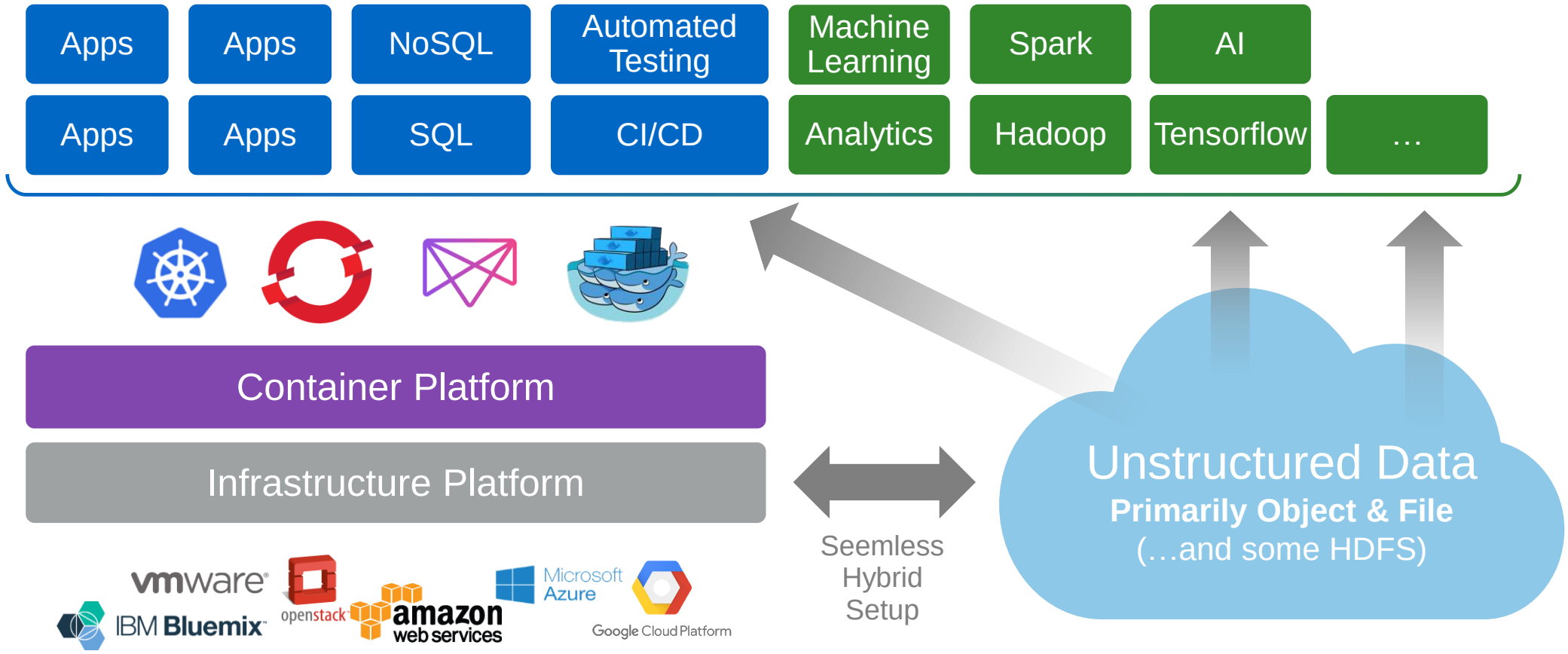
Application Stacks



Large Data Repository

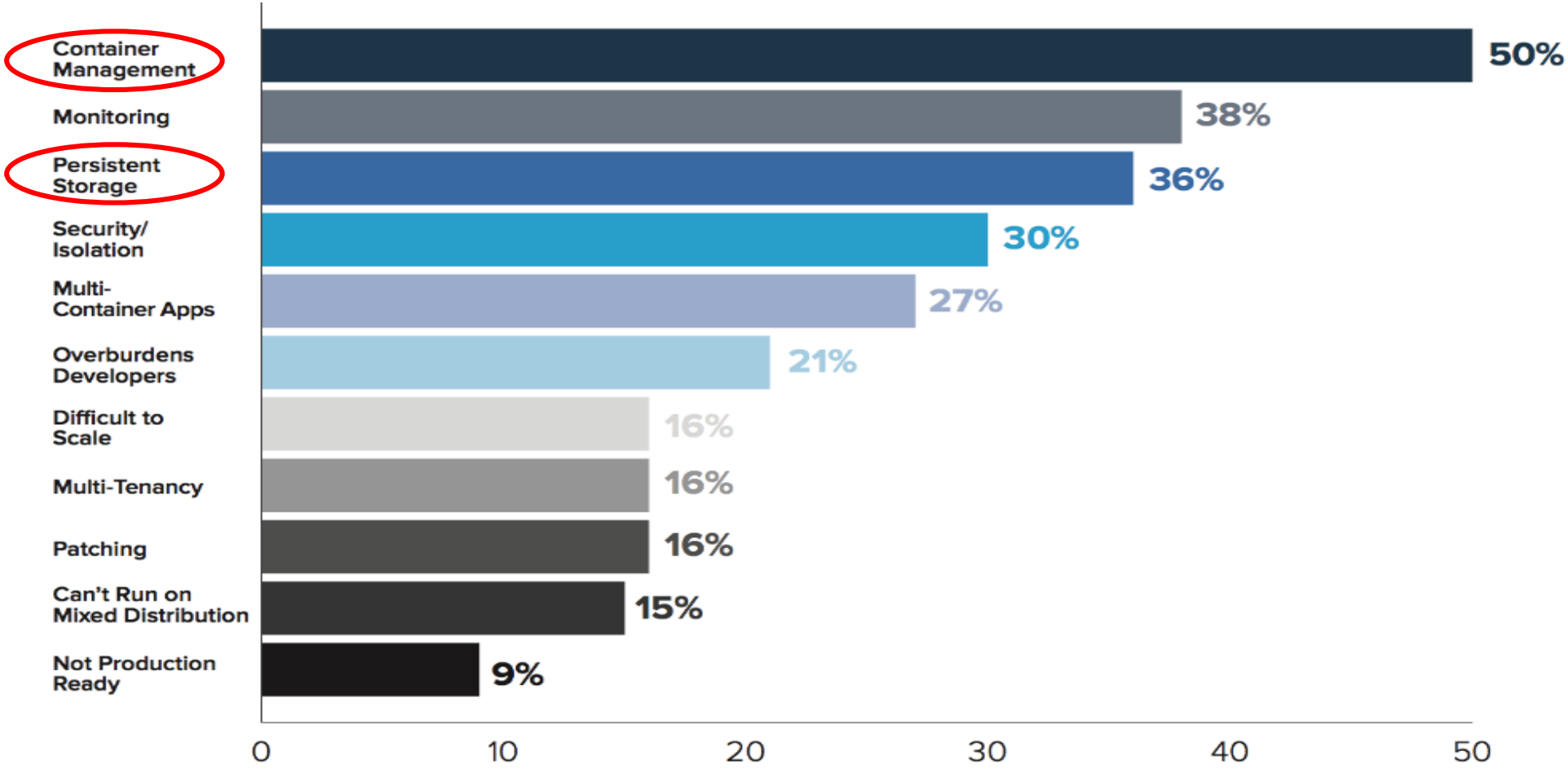
NetApp

어플리케이션 환경



NetApp

Container 도입시 Top challenges

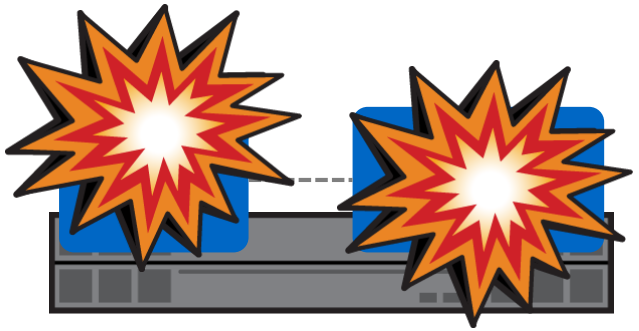


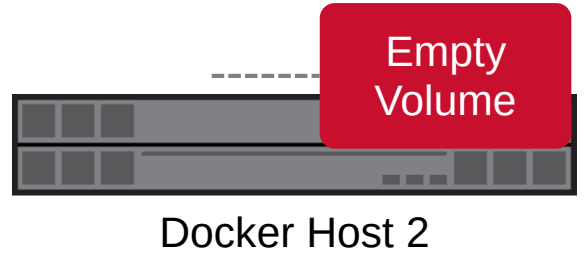
Source: Cloud Foundry Global Perception Study Container Technologies, 2017

NetApp

Containers & Data Persistency

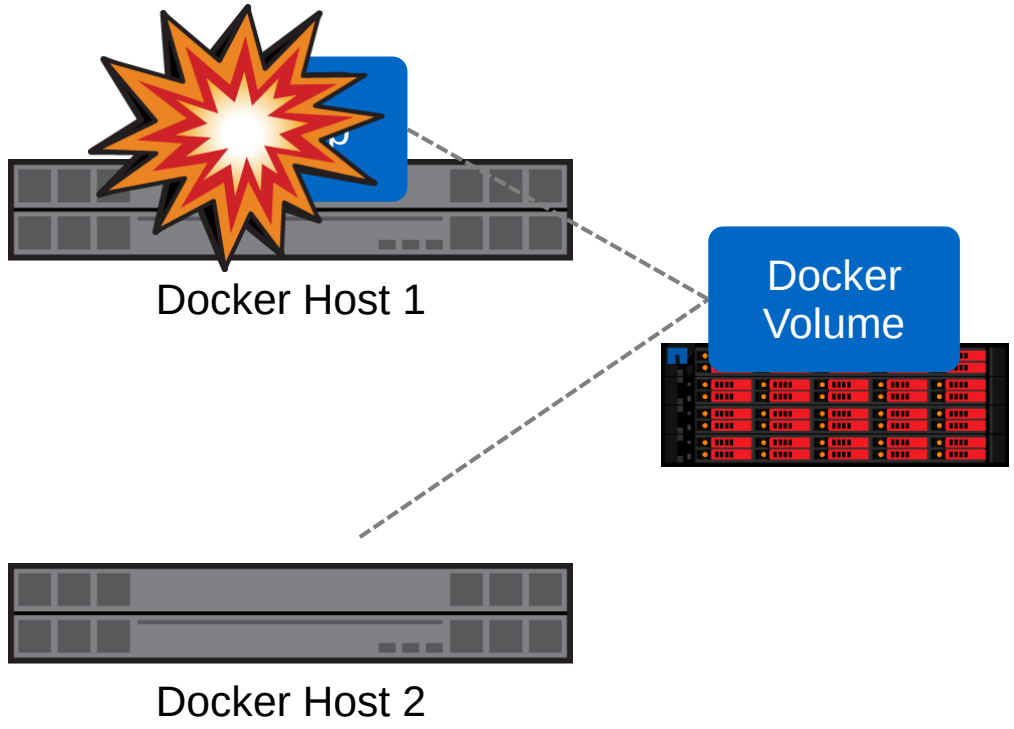
- 디폴트로, Docker Volume은 로컬디스크에 저장됨
- Container는 Dependency 가 없어야 함
  - 실제 환경에서 항상 유효한 것은 아님
- 호스트 장애시 데이터 유실





Containers & Data Persistency

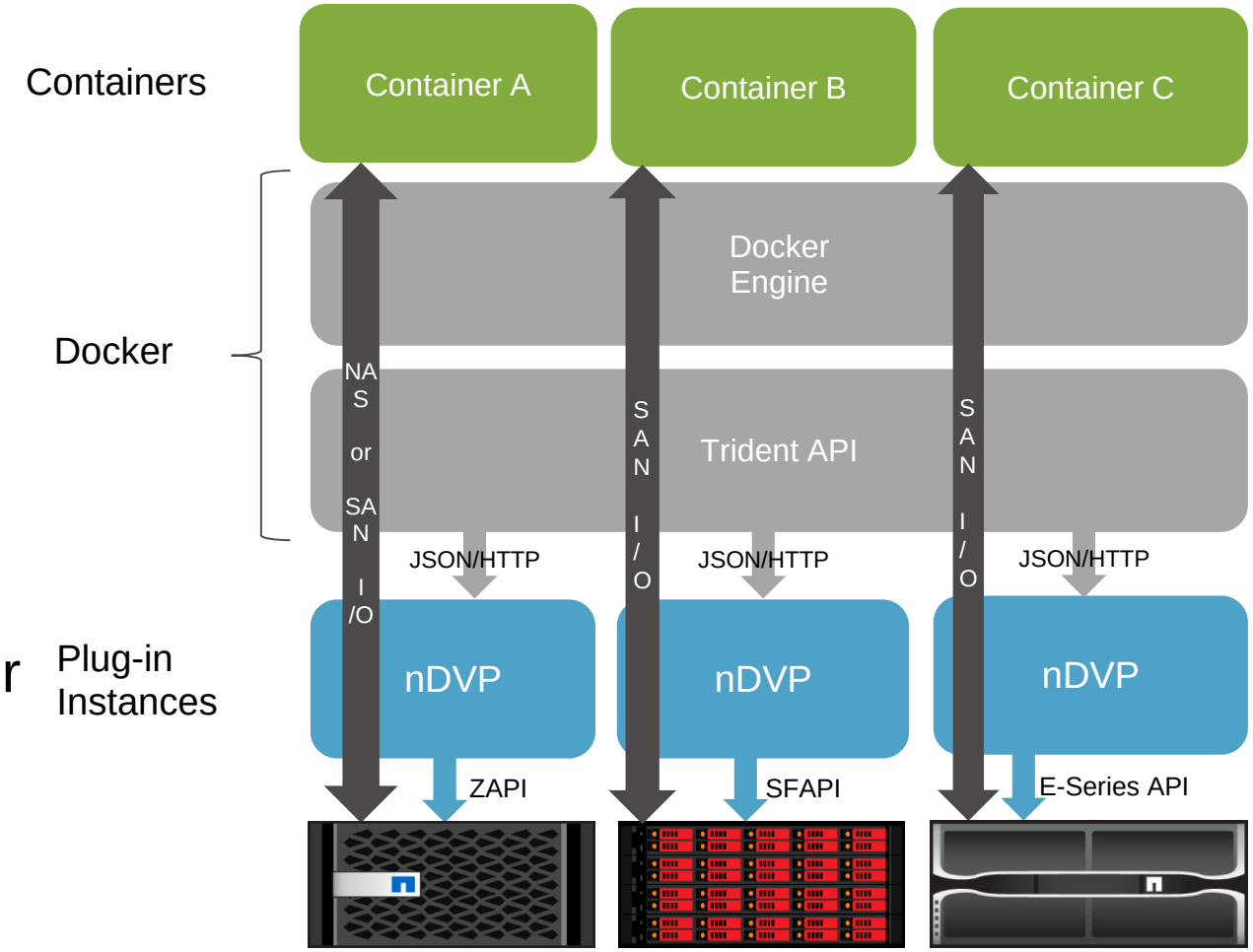
- 외장 스토리지를 이용한 Docker Volume
- 예시: NetApp Docker Volume Plugin (Trident)



NetApp

Docker Volume 플러그인(Trident, formerly nDVP) 특징

- Persistent volume을 생성하거나 복제
- 다중 Backend 구성을 지원
  - 동시에 ONTAP, SolidFire, and/or E-Series iSCSI and NAS 사용
- 사용자 별 애플리케이션에 맞는 스토리지의 볼륨 정의
  - ONTAP: Snapshot policy, space reserve, target aggregate, clone
- 여러 호스트에서 액세스 할 수 있는 Docker volume



NetApp

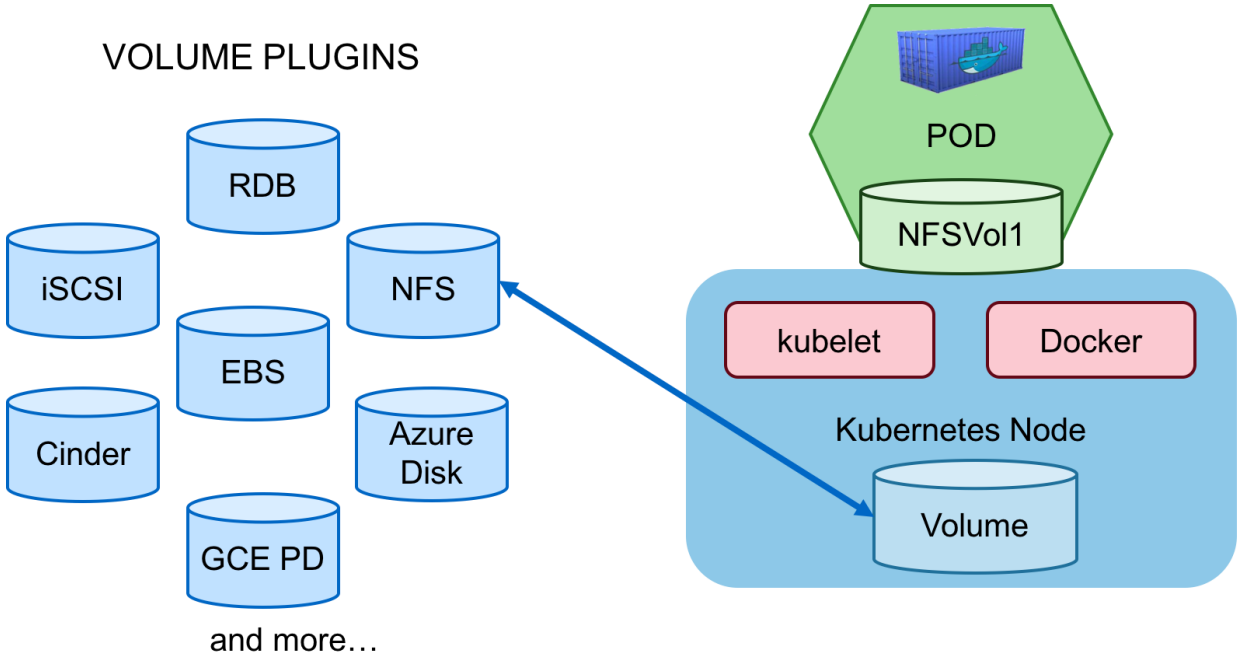
Kubernetes (aka ‘k8s’)

Feature rich, established, and growing

- 컨테이너 오케스트레이션 툴
  - Google이 만든 오픈 소스 Container 관리 시스템
  - 수 많은 호스트에서 운영중인 컨테이너 관리
  - 수평 확장
  - 스토리지 오케스트레이션
  - 셀프 힐링
  - Service 조회와 로드 밸런싱
- Container 사용자의 60% ↑
- OpenShift, Apprenda PaaS
- Hyperscaler : AWS EKS, Azure AKS, Google GKE …

Kubernetes Persistent Volumes

- Kubernetes Volume 지원
  - iSCSI
  - NFS
  - Cinder
  - AWS EBS,GCE PD, Azure disk, …
- 외장 스토리지의 사용
  - 볼륨은 단일 호스트의 수명을 초과하여 유지 될 수 있음
  - 컨테이너에서 대규모 워크로드를 안정적으로 실행 할 수 있다는 점
  - 백업 / 복원, 지역간 복제, 테스트 / Dev (복제) 및 기타 문제 해결
- Static vs Dynamic 프로비저닝



NetApp

Kubernetes Volume 요소

Persistent Volume (PV)

- 관리자가 Kubernetes에 제공한 데이터 저장소
- 백엔더 스토리지로 구성
  - NFS, iSCSI, Cinder, AWS EBS, Azure
- 스토리지 볼륨에 대한 연결 정보를 포함

```
apiVersion: v1
  kind: PersistentVolume
  metadata:
    name: pv0003
  spec:
    capacity:
      storage: 5Gi
    accessModes:
      - ReadWriteMany
  storageClassName: bronze
    nfs:
      path: /tmp
      server: 172.17.0.2
```

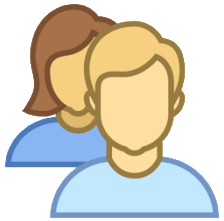
Persistent Volume Claim (PVC)

- 데이터 저장 장치를 요청하기 위해 사용자가 생성
- 사용자의 선택을 돕기 위한 Label과 access mode, 요구 용량등이 표기 됨
- Kubernetes는 PVC에서 요구되는 요구 사항을 충족시키기 위해 PV를 할당

```
kind:
PersistentVolumeClaim
apiVersion: v1
metadata:
  name: thepub
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 5Gi
  storageClassName:
bronze
```

K8s Static Provisioning

Users &
Developers



Ops Team / IT Admins

Kubernetes

K8S PV
Controller

Storage

PVC
5 GB
RWO

PVC
8 GB
RWO

POD

POD

PV
10 GB
RWO
NFS

PV
5 GB
RWO
iSCSI

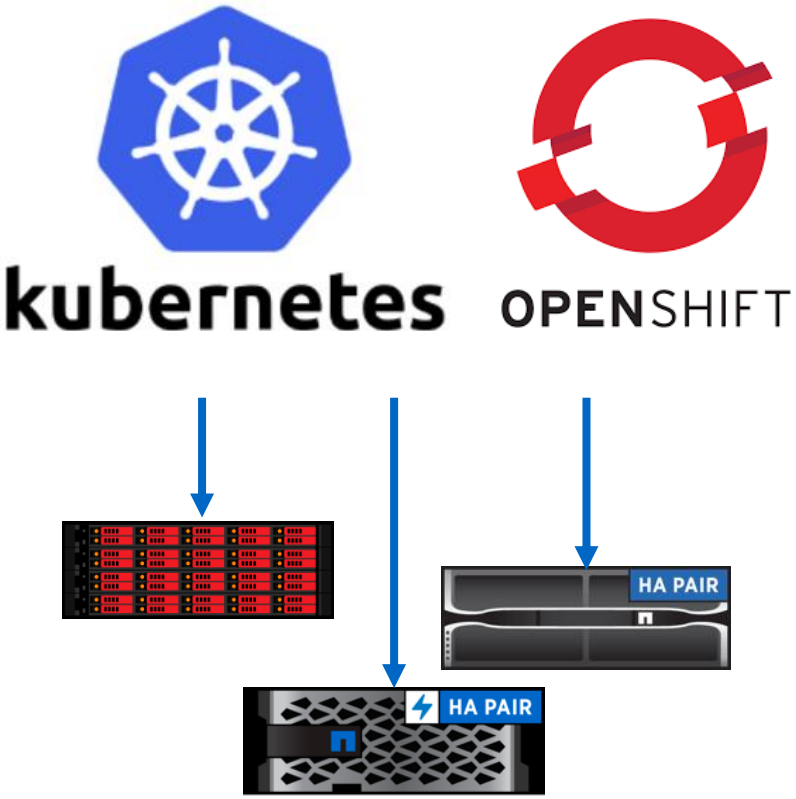
PersistentVolumeClaim

PersistentVolume

NetApp

Trident: Kubernetes 와 NetApp 스토리지 연동

- NetApp의 오픈 소스 Dynamic storage provisioner 지원:
  - NetApp ONTAP
  - NetApp SolidFire
  - E-Series
- 자동화 된 볼륨 생성과 맵핑
- 호환성:
  - OpenShift Origin & Enterprise
  - Kubernetes
- Available on GitHub: https://github.com/NetApp/trident

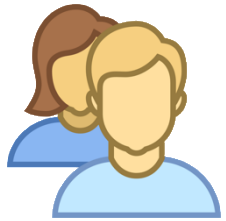


NetApp

Kubernetes Dynamic Provisioning

With NetApp Trident Integration

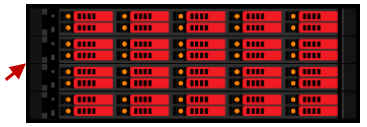
Users & Developers



Ops Team / IT Admins

Kubernetes

K8S PV Controller



NetApp SolidFire

Gold
IOPS:
3000/6000/10000

Silver
IOPS:
2000/4000/8000

Bronze
IOPS:
1000/2000/4000

PVC
8 GB
RWO
GOLD

POD

NetApp Provisioner (Trident)

PV
8 GB
RWO
iSCSI

PersistentVolumeClaim

StorageClass

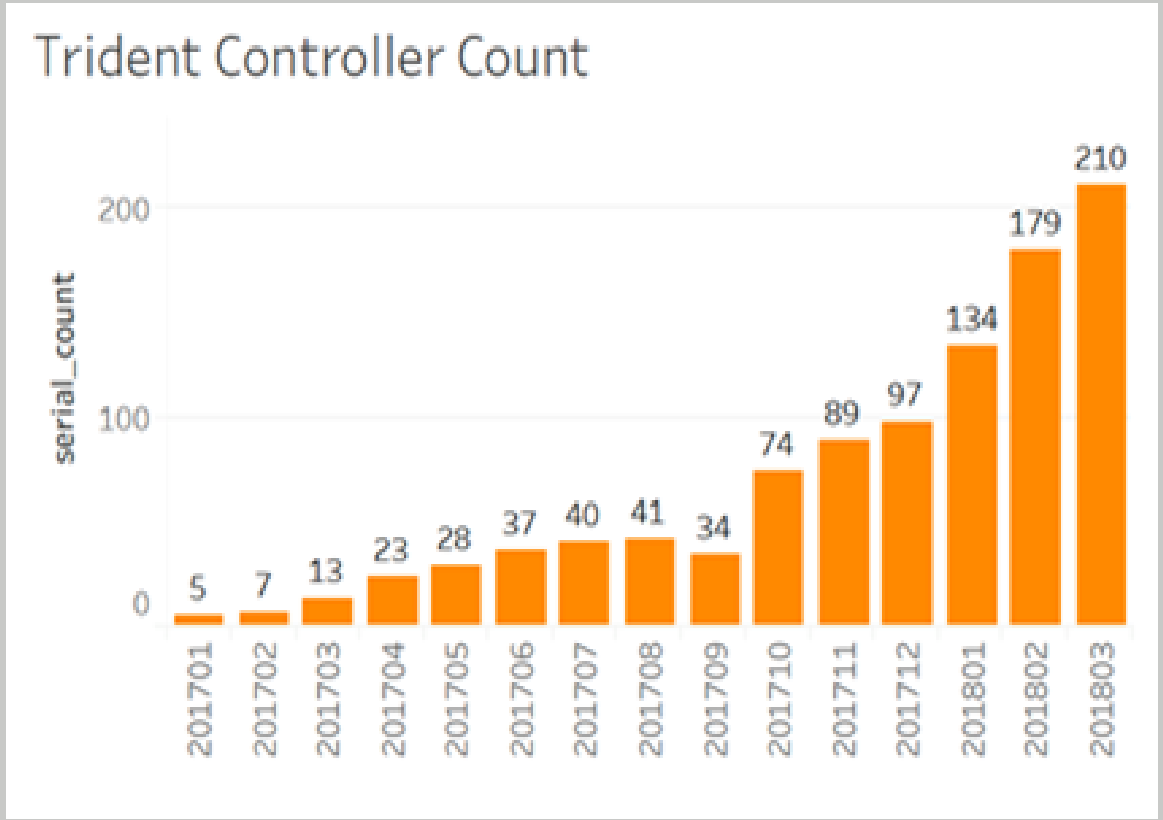
※ StorageClass : Storage provisioner 지정, 카탈로그 정의

NetApp

Trident can do,

- Kubernetes Dynamic provisioning 을 지원
  - Admin의 개입없이 User가 Volume 배포
- Storage 카탈로깅
  - QoS
  - Thin / Thick
  - Snapshot
- PVC 복제 - Rapid Data Cloning
- RWM(ReadWriteMany) access mode 지원
- 검증된 Data protection 기술 활용

Trident 적용 추이



NetApp

NetApp Open Source Contribution

NetApp은 다양한 오픈 소스 커뮤니티에서 스토리지 프로젝트를 리딩

OpenStack

- Diablo부터 커뮤니티 멤버로 활동
- Charter Gold Member
- Elected board representation
- Manila, Cinder 프로젝트 Leader

Docker

- Ecosystem Technology Partner (2016년)
- Docker Certification Program (2017년)
- First Certified Docker Volume Plugin
- First Snapshot & Clone 기능 제공

Kubernetes

- CNCF Gold Member
- First 외장 Dynamic provisioner 개발
- First Clone 기능 제공
- CSI(Container Storage Interface) 지원 예정

NetApp

NetApp Trident PoC

Trident plug-in을 이용한 NFS 가용성 테스트

LG CNS

NFS 가용성 테스트 목적

❑ Persistent Volumes for Container services

- dynamic provisioning
- multi backend

❑ Shared Volumes for Container services

- access data both container and host server

❑ High Availability for trident services

테스트 케이스

❑ trident plug-in 주요 기능 검증

- dynamic provisioning 확인
- shared volume 확인
- access mode 확인

❑ trident 서비스 가용성 검증

- trident pod의 take-over

❑ h/w 장애 유발

- node shutdown
- storage down
- network down

Trident plug-in 주요 기능 검증

```
# cat test_pod.yaml

kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: nfs-test
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 300Gi
  storageClassName: basic
---
apiVersion: v1
kind: ReplicationController
metadata:
  name: nfs-busybox
spec:
  replicas: 2
  selector:
    name: nfs-busybox
  template:
    metadata:
      labels:
        name: nfs-busybox
    spec:
      containers:
      - image: busybox
        imagePullPolicy: IfNotPresent
        name: busybox
        ports:
          - containerPort: 80
            protocol: TCP
        volumeMounts:
          # name must match the volume name below
          - name: nfs-volume
            mountPath: "/mnt"
      volumes:
      - name: nfs-volume
        persistentVolumeClaim:
          claimName: nfs-test
---
```

Trident plug-in 주요 기능 검증

❑ pod 상태

```
root@CPKUBCNTR01:~/test_nfs# k get po -w
NAME                                      READY     STATUS    RESTARTS   AGE
nfs-busybox-69cv5                         1/1       Running   0          58s
nfs-busybox-czmvn                         1/1       Running   0          58s
```

❑ pv 상태

```
root@CPKUBCNTR01:~# k get pv -w
NAME                    CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS     CLAIM              STORAGECLASS   REASON    AGE
default-basic-ccecf     1Gi        RWO            Delete           Bound      default/basic      basic                    11d
trident                 2Gi        RWO            Retain           Bound      trident/trident                             15d
default-nfs-test-9e6cb   300Gi      ROX       Delete    Pending   default/nfs-test   basic2              0s
default-nfs-test-9e6cb   300Gi      ROX       Delete    Bound     default/nfs-test   basic2              0s

^Croot@CPKUBCNTR01:~# k get pvc -w
NAME       STATUS    VOLUME                  CAPACITY   ACCESS MODES   STORAGECLASS   AGE
basic      Bound     default-basic-ccecf     1Gi        RWO            basic          11d
nfs-test   Pending                                      basic2    0s
nfs-test   Pending                                      basic2    0s
nfs-test   Pending   default-nfs-test-9e6cb   0                      basic2    1s
nfs-test   Bound     default-nfs-test-9e6cb   300Gi      ROX         basic2    1s
```

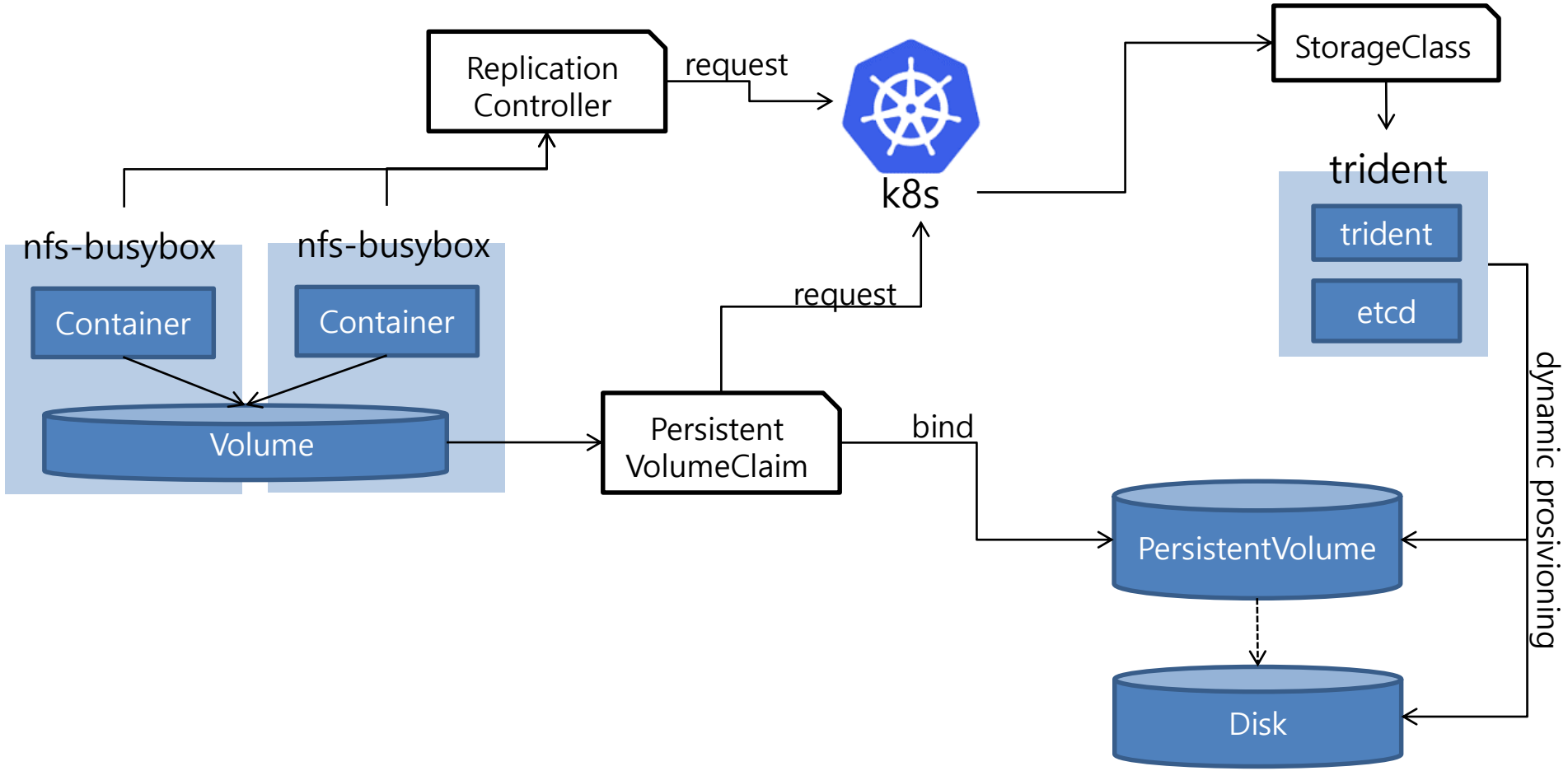
Trident plug-in 주요 기능 검증

❑ mount 상태(pod 내부에서 확인)

```
/ # df -h /mnt
Filesystem                Size      Used Available Use% Mounted on
100.1.33.236:/trident_default_nfs_test_9e6cb
                        285.0G    256.0K    285.0G   0% /mnt

/ # mount | grep mnt
100.1.33.236:/trident_default_nfs_test_9e6cb on /mnt type nfs4 (rw,relatime,vers=4.0,rsize=65536,wsize=65536,
namlen=255,hard,proto=tcp,port=0,timeo=600,retrans=2,sec=sys,clientaddr=100.1.33.234,local_lock=none,addr=100
.1.33.236)
```

Trident plug-in 주요 기능 검증



Trident 서비스 가용성 검증

❑ pod 상태

```
root@CPKUBCNTR01:~/test_nfs# kubectl get pod -o wide -n trident
NAME                       READY     STATUS    RESTARTS   AGE       IP                NODE
trident-cdd5fc7b4-8p5vr    2/2       Running   0          6d        192.168.127.206   cpkubnodep004
```

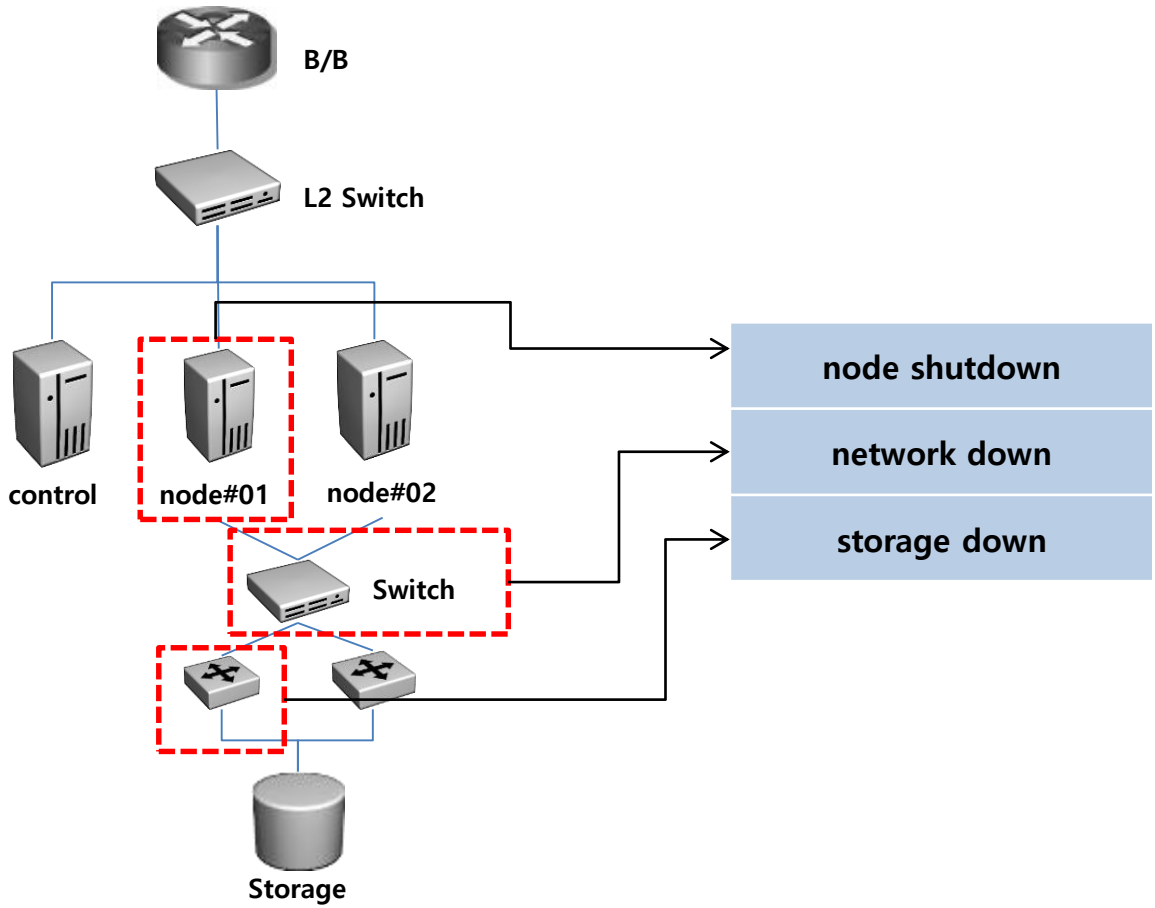
❑ pod 를 다른 node로 take-over

```
^Croot@CPKUBCNTR01:~/test_nfs# kubectl get pod -o wide -n trident -w
NAME                       READY     STATUS              RESTARTS   AGE       IP                NODE
trident-cdd5fc7b4-8p5vr    2/2       Running             0          6d        192.168.127.206   cpkubnodep004
trident-cdd5fc7b4-8p5vr    2/2       Running             0          6d        192.168.127.206   cpkubnodep004
trident-cdd5fc7b4-8p5vr    2/2       NodeLost            0          6d        192.168.127.206   cpkubnodep004
trident-cdd5fc7b4-8p5vr    2/2       Unknown             0          6d        192.168.127.206   cpkubnodep004
trident-cdd5fc7b4-gt7vc    0/2       Pending             0          0s        <none>    <none>
trident-cdd5fc7b4-gt7vc    0/2       Pending             0          0s        <none>    cpkubnodecpu
trident-cdd5fc7b4-gt7vc    0/2       ContainerCreating   0          7s        <none>    cpkubnodecpu
trident-cdd5fc7b4-gt7vc    2/2       Running             0          10s       192.168.14.172    cpkubnodecpu
```

❑ 정상 생성 확인

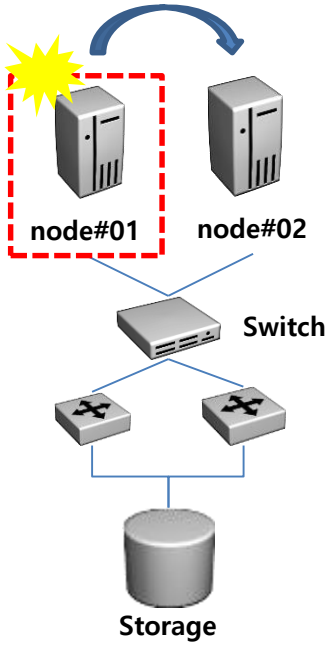
```
root@CPKUBCNTR01:~/test_nfs# kubectl get pvc
NAME       STATUS    VOLUME                     CAPACITY   ACCESS MODES   STORAGECLASS   AGE
basic      Bound     default-basic-ccecf        1Gi        RWO            basic          11d
nfs-test   Bound     default-nfs-test-36d99     300Gi      ROX            basic2         36s
```

H/W 장애 유발



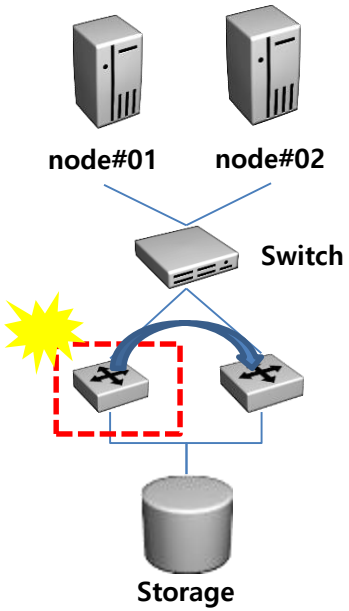
장애 case
cpu fault
memory fault
disk fault
mgmt 네트워크 장애
nfs 네트워크 장애
한쪽 Controller 장애 시
전체 NAS 스토리지 전체 장애 시
disk fault
controller 장애
kubelet service 장애
docker service 장애
trident service 장애
네트워크 장애
disk 부하 발생
cpu 부하 발생

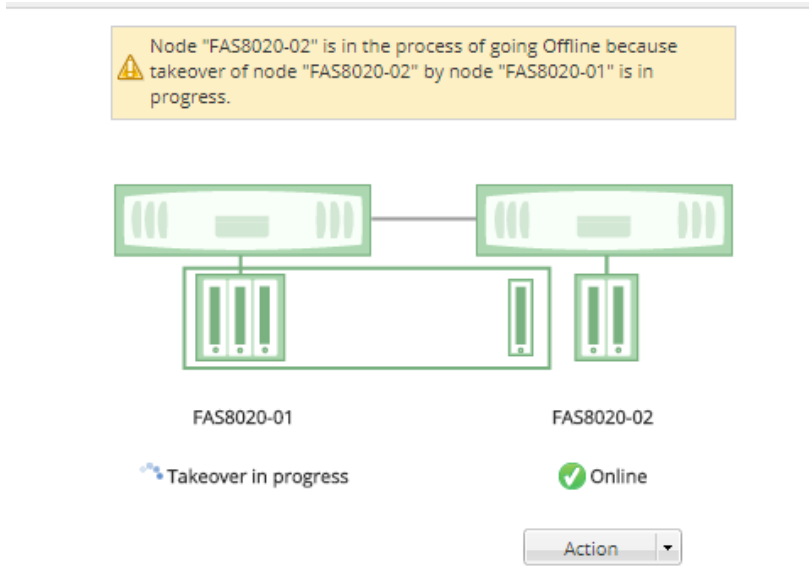
Node shutdown




```
root@CPKUBCNTR01:~/test_nfs# kubectl get pod -o wide -w | grep nfs
nfs-busybox-btd2d                                    1/1       Running   0          16m       192.168.127.220   cpkubnodep004
nfs-busybox-zq48z                                    1/1       Running   0          16m       192.168.14.190    cpkubnodecpu
nfs-busybox-btd2d   1/1       Running   0          17m       192.168.127.220   cpkubnodep004
nfs-busybox-btd2d   1/1       NodeLost   0          22m       192.168.127.220   cpkubnodep004
nfs-busybox-btd2d   1/1       Unknown   0          22m       192.168.127.220   cpkubnodep004
nfs-busybox-zn92b   0/1       Pending   0          0s        <none>    <none>
nfs-busybox-zn92b   0/1       Pending   0          0s        <none>    cpkubnodecpu
nfs-busybox-zn92b   0/1       ContainerCreating   0          0s        <none>    cpkubnodecpu
nfs-busybox-zn92b   1/1       Running   0          2s        192.168.14.191   cpkubnodecpu
```

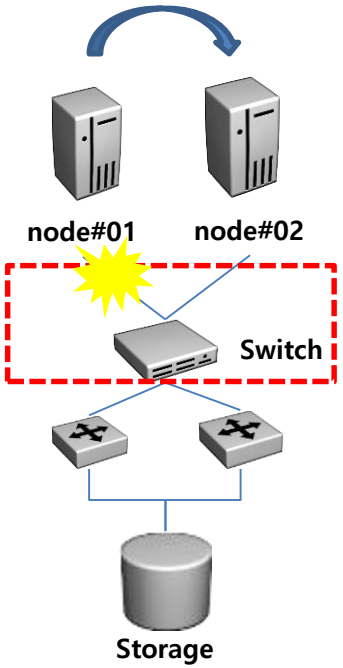
Storage down






```
root@CPKUBCNTR01:~/test_nfs# kubectl get pod -o wide | grep nfs
nfs-busybox-8dvbp                          1/1     Running   0          27s     192.168.127.228   cpkubnodep004
nfs-busybox-99n2r                          1/1     Running   0          27s     192.168.14.138    cpkubnodecpu
```

Network down




```
root@CPKUBCNTR01:~/test_nfs# kubectl exec -it nfs-busybox-8dvbp -- sh
/ # hostname
nfs-busybox-8dvbp
/ # df -h
Filesystem                Size      Used Available Use% Mounted on
none                     85.0G     57.6G     23.0G  71% /
tmpfs                    64.0M         0     64.0M   0% /dev
tmpfs                    94.4G         0     94.4G   0% /sys/fs/cgroup
```

```
/ # mount | grep nfs
100.1.33.236:/trident_default_nfs_test_dba3e on /mnt type nfs4 (rw,relatime,vers=4.0
,rsize=65536,wsize=65536,namlen=255,hard,proto=tcp,port=0,timeo=600,retrans=2,sec=sy
s,clientaddr=100.1.33.235,local_lock=none,addr=100.1.33.236)
```

NFS hang 현상

❑ nfs mountOptions

soft / hard Determines the recovery behavior of the NFS client after an NFS request times out. If neither option is specified (or if the hard option is specified), NFS requests are retried indefinitely. If the soft option is specified, then the NFS client fails an NFS request after retrans retransmissions have been sent, causing the NFS client to return an error to the calling application.

❑ StorageClass

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  creationTimestamp: 2018-06-12T07:50:02Z
  name: basic
  resourceVersion: "10705962"
  selfLink: /apis/storage.k8s.io/v1/storageclasses/basic
  uid: 368921b3-6e15-11e8-9123-f8bc1239f934
mountOptions:
- rw
- nfsvers=3
- proto=tcp
- soft
- timeo=180
parameters:
  backendType: ontap-nas
  provisioningType: thick
provisioner: netapp.io/trident
reclaimPolicy: Delete
```

마무리

❑ nfs 가용성테스트 결과

- nfs network 단절로 인한 hang 현상은 nfs 자체 이슈

(운영에 있어서 이와 관련된 option의 적용 협의 필요)

- plug-in을 통한 nfs backend storage를 지원하는데 이상 없음