

Managing Kubernetes

2021-06-04

written by A.K.A whatwant

Agenda

Chapter1. Kubernetes Overview

1주차: **Docker and Kubernetes**

Chapter2. Kubernetes Core

2주차: **Environment & POD**

3주차: **Replication and other controllers**

4주차: **Services**

5주차: **Volumes (Service 보충 수업 포함)**

6주차: **ConfigMaps, Secrets & Kubernetes REST API**

7주차: **Deployment & StatefulSet**

8주차: **Summary**

Chapter3. Kubernetes Managing

9주차: **Authentication and User Management & Authorization & Admission Control**

10주차: **Networking & Monitoring**

11주차: **Disaster Recovery**

※ 참고 : <https://home.modulabs.co.kr/product/managing-kubernetes/>

**7 week
Deployment &
StatefulSet**

[Tip] Master Node에서 Docker build 하기 (네트워크 이슈)

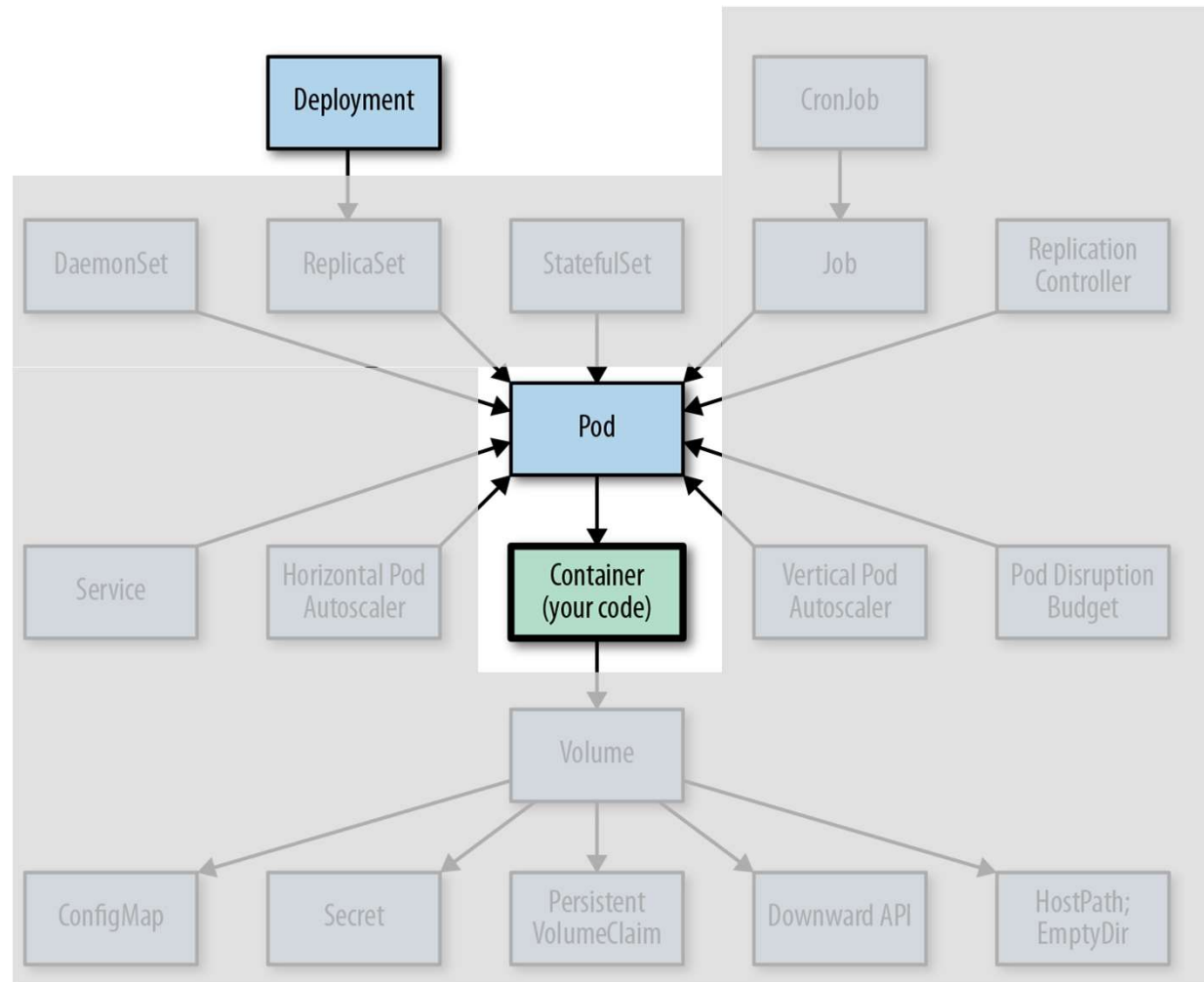
- Master Node에 있는 Docker는 Kubernetes 환경에 맞춘 network 설정이 되어있음
- . 일반적인 docker build 수행을 위해서는 network 설정을 명시적으로 선언해 주어야 함

```
> docker build --network host ...
```

Deployment

Today ...

Deployment



Application Update

- 기존 Pod를 버리고, 새로운 Pod로 변경하고 싶은 경우

#1. Deleting old pods and replacing them with new ones

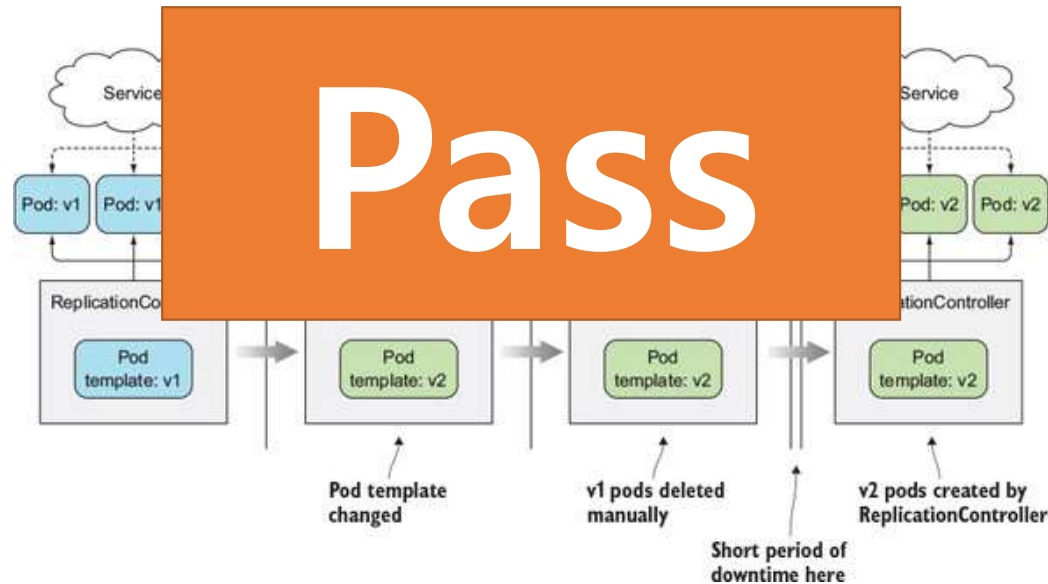
#2. Switching from the old to (new Deployment)

#3. Rolling update

Pass

#1. Deleting old pods and replacing them with new ones

- Application의 변경(like version-up)이 필요한 경우 ReplicationController/ReplicationSet으로 구성되었다면 손쉽게 적용 가능
 - ① Template에서 새로운 version으로 변경 작성
 - ② Pod 삭제
 - ③ 변경된 Template 기준으로 새로운 Pod 자동 생성
- 짧은 시간의 다운타임을 허용할 수 있다면, 가장 간단한 방법

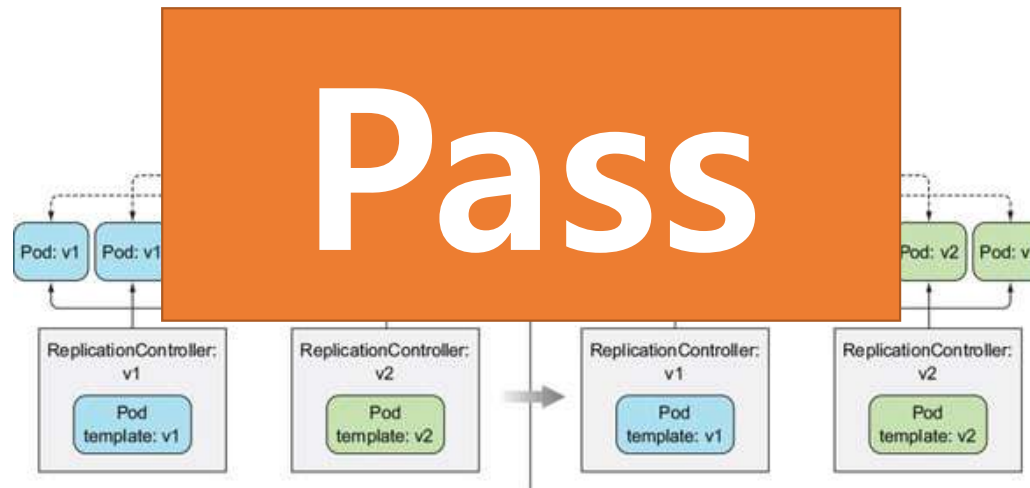


※ 참고 : <https://livebook.manning.com/book/kubernetes-in-action/chapter-9/22>

#2. Switching from the old to the new version at once

- 다운타임이 발생하지 않고 한 번에 여러 version의 application이 실행하는 것을 지원하는 경우
 - ① 새로운 version의 Template으로 신규 Pod 생성, 기존 version은 지속 서비스 中
 - ② 한 번에 Service를 신규 Pod를 바라보도록 전환 *kubectl set selector* 기능을 사용하면 간단하게 가능!
 - ③ 전환 완료되면, 기존 Pod 삭제

= Blue-Green Deployment

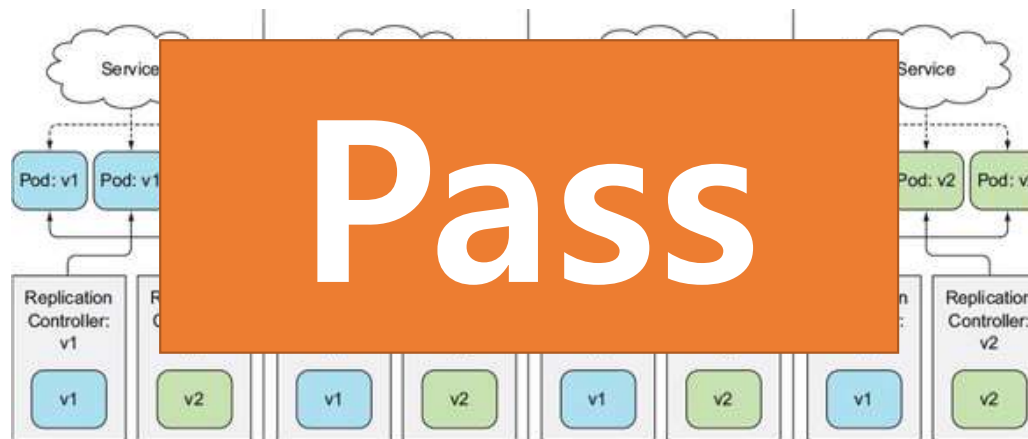


※ 참고 : <https://livebook.manning.com/book/kubernetes-in-action/chapter-9/29>

#3. Rolling update - overview

- Pod를 단계별로 교체

. 수작업으로 진행하기에는 상당히 번거롭고, 실수할 여지가 많음 → kubernetes에서 제공해주는 여러 방법 존재



※ 참고 : <https://livebook.manning.com/book/kubernetes-in-action/chapter-9/32>

#3. Rolling update – 실습 : v1.0 Pod 배포

- Container 이미지를 v1.0 → v2.0 으로 Rolling update 하는 과정을 실습해보자.

01-node-web-rs-v1.yaml

```
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: node-web-v1

spec:
  replicas: 3
  selector:
    matchLabels:
      app: node-web
  template:
    metadata:
      name: node-web
    labels:
      app: node-web
    spec:
      containers:
        - image: whatwant/node-web:1.0
          name: node-web
          ports:
            - containerPort: 8080
              protocol: TCP
```

01-node-web-svc.yaml

```
apiVersion: v1
kind: Service
metadata:
  name: node-web

spec:
  selector:
    app: node-web
```

```
> kubectl apply -f 01-node-web-rs-v1.yaml
```

```
> kubectl apply -f 01-node-web-svc.yaml
```

```
> kubectl get pods
```

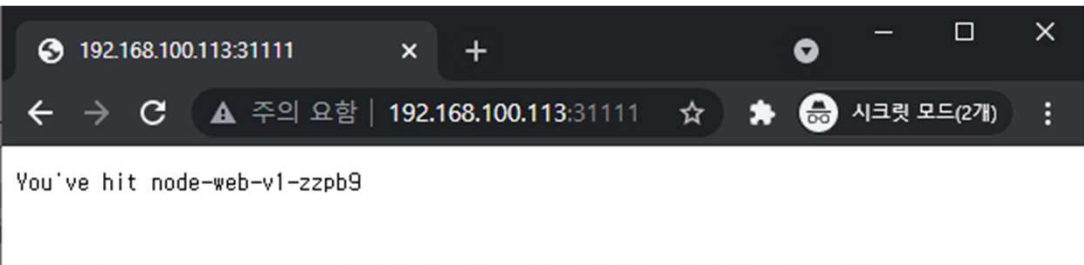
NAME	READY	STATUS	RESTARTS	AGE
node-web-1	1/1	Running	0	22s
node-web-2	1/1	Running	0	22s
node-web-3	1/1	Running	0	22s

Pass

NAME	EXTERNAL-IP	PORT(S)	AGE
node-web	10.233.52.35	443/TCP	24d
node-web	10.233.52.35	80:31111/TCP	5m54s

[node-web:1.0](#)
[container 상세 설명](#)

<https://hub.docker.com/repository/docker/whatwant/node-web>



#3. Rolling update – 실습 : v2.0 이미지 준비

교체할 이미지가 이미 있다면 생략 가능

- v2.0 Container 이미지를 만들어서 DockerHub에 업로드 하자.

02-app.js

```
const http = require('http');
const os = require('os');

console.log("node-web v2.0 server starting...");

var handler = function(request, response) {
  console.log("Received request from " + request.remoteAddress);
  response.writeHead(200);
  response.end("You've hit " + os.hostname());
};

var www = http.createServer(handler);
www.listen(8080);
```

02-Dockerfile

```
node:latest
02-app.js /app.js
POINT ["node", "app.js"]
```

Pass

```
> docker build -t <user-id>/node-web:2.0 -f 02-Dockerfile .
> docker push <user-id>/node-web:2.0
```

#3. Rolling update – 실습 : cli 방식

- `kubectl rolling-update` 명령어는 deprecated 되었다. → `kubectl rollout` 명령어로 대체되었다.
- 그런데, ReplicationController/ReplicaSet 에서는 사용할 수 없다.



Pass

※ 참고 : <https://github.com/kubernetes/kubernetes/issues/88051>

Deployment – 실습 : v1.0 Pod 배포

- Container 이미지를 v1.0 → v2.0 으로 Rolling update 하는 과정을 실습해보자.

03-node-web-deployment.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: node-web

spec:
  replicas: 3
  selector:
    matchLabels:
      app: node-web
  template:
    metadata:
      name: node-web
    labels:
      app: node-web
    spec:
      containers:
        - image: whatwant/node-web:1.0
          name: node-web
          ports:
            - containerPort: 8080
              protocol: TCP
```

03-node-web-svc.yaml

```
apiVersion: v1
kind: Service
metadata:
  name: node-web

spec:
  type: NodePort
  ports:
    - port: 80
      targetPort: 8080
      nodePort: 31111
  selector:
    app: node-web
```

node-web:1.0
container 상세 설명

<https://hub.docker.com/repository/docker/whatwant/node-web>

```
> kubectl create -f 03-node-web-rc-v1.yaml --record
```

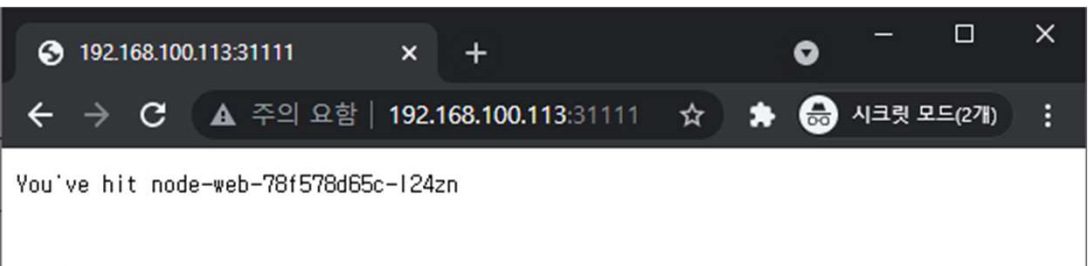
```
> kubectl create -f 03-node-web-svc.yaml
```

```
> kubectl get pods
```

NAME	READY	STATUS	RESTARTS	AGE
node-web-78f578d65c-8skvx	1/1	Running	0	25s
node-web-78f578d65c-l24zn	1/1	Running	0	25s
node-web-78f578d65c-xm2fx	1/1	Running	0	25s

```
> kubectl get services
```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
kubernetes	ClusterIP	10.233.0.1	<none>	443/TCP	24d
node-web	NodePort	10.233.52.35	<none>	80:31111/TCP	5m54s



Deployment – 실습 : status

```
> kubectl get deployments -o wide
```

NAME	READY	UP-TO-DATE	AVAILABLE	AGE	CONTAINERS	IMAGES	SELECTOR
node-web	3/3	3	3	3m36s	node-web	whatwant/node-web:1.0	app=node-web

```
> kubectl get replicaset -o wide
```

NAME	DESIRED	CURRENT	READY	AGE	CONTAINERS	IMAGES	SELECTOR
node-web-78f578d65c	3	3	3	77s	node-web	whatwant/node-web:1.0	app=node-web,pod-template-hash=78f578d65c

```
> kubectl get services -o wide
```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE	SELECTOR
kubernetes	ClusterIP	10.233.0.1	<none>	443/TCP	24d	<none>
node-web	NodePort	10.233.52.35	<none>	80:31111/TCP	83m	app=node-web

```
> kubectl get pods -o wide
```

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE	NOMINATED NODE	READINESS GATES
node-web-78f578d65c-8skvx	1/1	Running	0	91s	10.233.103.119	worker2	<none>	<none>
node-web-78f578d65c-l24zn	1/1	Running	0	91s	10.233.103.121	worker2	<none>	<none>
node-web-78f578d65c-xm2fx	1/1	Running	0	91s	10.233.103.120	worker2	<none>	<none>

```
> kubectl rollout status deployment node-web
```

```
deployment "node-web" successfully rolled out
```

Deployment – 실습 : v2.0 이미지 준비

교체할 이미지가 이미 있다면 생략 가능

- v2.0 Container 이미지를 만들어서 DockerHub에 업로드 하자.

02-app.js

```
const http = require('http');
const os = require('os');

console.log("node-web v2.0 server starting...");

var handler = function(request, response) {
  console.log("Received request from " + request.connection.remoteAddress);
  response.writeHead(200);
  response.end("You've hit " + os.hostname() + " for v2.0\n");
};

var www = http.createServer(handler);
www.listen(8080);
```

02-Dockerfile

```
FROM node:latest
ADD 02-app.js /app.js
ENTRYPOINT ["node", "app.js"]
```

```
> docker build -t <user-id>/node-web:2.0 -f 02-Dockerfile .
> docker push <user-id>/node-web:2.0
```


Kubernetes의 기존 리소스 수정하기 → Deployment 수정 방법

명령어	설명	예시
kubectl edit	기본 편집기로 오브젝트의 manifest를 엽니다. 변경 후 파일을 저장하고 편집기를 종료하면 오브젝트가 업데이트 된다.	kubectl edit deployment node-web
kubectl patch	오브젝트의 개별 속성을 수정한다.	kubectl patch deployment web -p '{"spec": {"minReadySeconds": 10}}'
kubectl apply	전체 YAML/JSON 파일의 속성 값을 적용해 오브젝트를 수정한다. 파일에는 리소스의 전체 정의를 포함하여야 한다.	kubectl apply -f node-web-v2.yaml
kubectl replace	YAML/JSON 파일로 오브젝트를 새 것으로 교체한다. 오브젝트가 없을 때 실행하면 오류를 출력한다.	kubectl replace -f node-web-v2.yaml
kubectl set image	Pod, Deployment, ReplicaSet, DaemonSet, Job에 정의된 컨테이너 이미지를 변경한다.	kubectl set image deployment node-web nodejs=ww/node-web:v2.0

Deployment – 실습 : kubectl set image 1/2

```
> sh -c 'while true; do curl http://192.168.100.113:31111; sleep 2; done'
```

You've hit node-web-78f578d65c-l24zn

You've hit node-web-78f578d65c-xm2fx

You've hit node-web-78f578d65c-8skvx

You've hit node-web-64f47c76b8-ggpfr for v2

You've hit node-web-78f578d65c-8skvx

You've hit node-web-64f47c76b8-hlps7 for v2

You've hit node-web-64f47c76b8-ggpfr for v2

You've hit node-web-64f47c76b8-j6jd2 for v2

You've hit node-web-64f47c76b8-hlps7 for v2

You've hit node-web-64f47c76b8-ggpfr for v2

```
> kubectl set image deployment node-web node-web=whatwant/node-web:2.0  
deployment.apps/node-web image updated
```

```
> kubectl rollout status deployment node-web
```

Waiting for deployment "node-web" rollout to finish: 2 out of 3 new replicas have been updated...

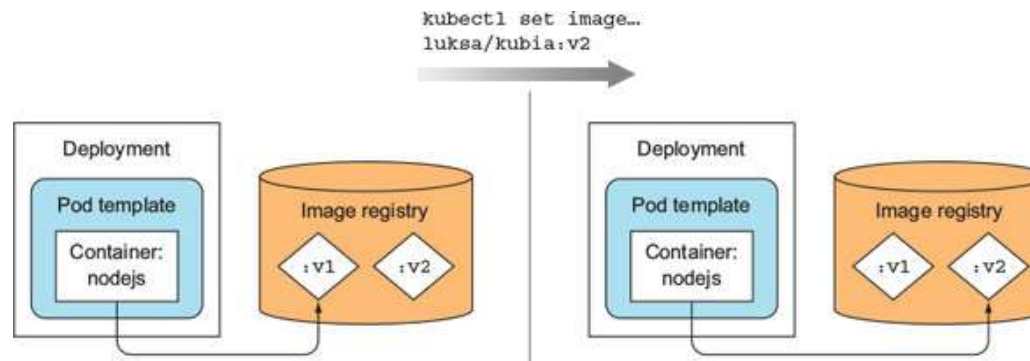
Waiting for deployment "node-web" rollout to finish: 2 out of 3 new replicas have been updated...

Waiting for deployment "node-web" rollout to finish: 2 out of 3 new replicas have been updated...

Waiting for deployment "node-web" rollout to finish: 1 old replicas are pending termination...

Waiting for deployment "node-web" rollout to finish: 1 old replicas are pending termination...

deployment "node-web" successfully rolled out

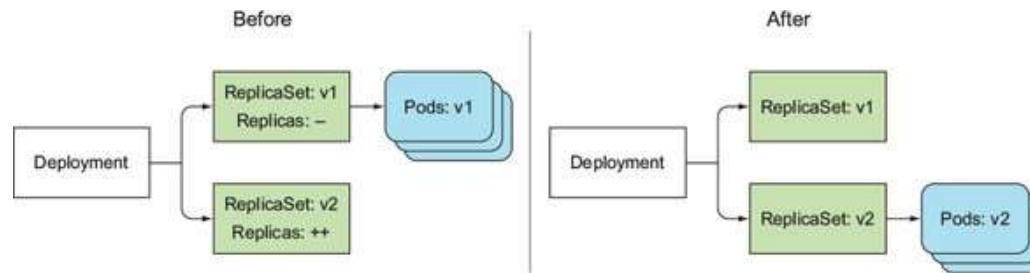


※ 참고 : <https://livebook.manning.com/book/kubernetes-in-action/chapter-9/154>

Deployment – 실습 : kubectl set image 2/2

```
> kubectl get replicaset -o wide
```

NAME	DESIRED	CURRENT	READY	AGE	CONTAINERS	IMAGES	SELECTOR
node-web-64f47c76b8	3	3	3	5m44s	node-web	whatwant/node-web:2.0	app=node-web,pod-template-hash=64f47c76b8
node-web-78f578d65c	0	0	0	8m25s	node-web	whatwant/node-web:1.0	app=node-web,pod-template-hash=78f578d65c



※ 참고 : <https://livebook.manning.com/book/kubernetes-in-action/chapter-9/166>

Deployment – 실습 : kubectl rollout undo

```
> sh -c 'while true; do curl http://192.168.100.113:31111; sleep 2; done'
```

```
You've hit node-web-64f47c76b8-j6jd2 for v2  
You've hit node-web-64f47c76b8-hlps7 for v2  
You've hit node-web-64f47c76b8-ggpfr for v2  
You've hit node-web-78f578d65c-r75ct  
You've hit node-web-64f47c76b8-hlps7 for v2  
You've hit node-web-78f578d65c-nftp6  
You've hit node-web-78f578d65c-r75ct  
You've hit node-web-78f578d65c-g6js6  
You've hit node-web-78f578d65c-nftp6
```

```
> kubectl rollout undo deployment node-web
```

deployment.apps/node-web rolled back

```
> kubectl get replicaset
```

NAME	DESIRED	CURRENT	READY	AGE
node-web-64f47c76b8	0	0	0	15m
node-web-78f578d65c	3	3	3	18m

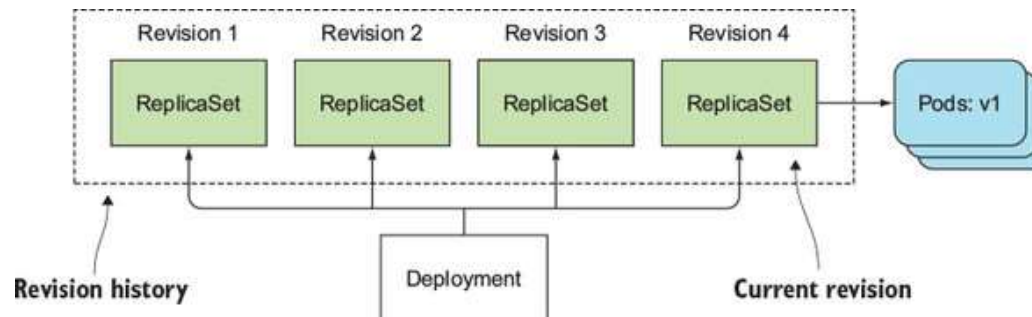
```
> kubectl rollout history deployment node-web
```

deployment.apps/node-web

REVISION	CHANGE-CAUSE
----------	--------------

2	kubectl create --filename=03-node-web-deployment.yaml --record=true
---	---

3	kubectl create --filename=03-node-web-deployment.yaml --record=true
---	---



※ 참고 : <https://livebook.manning.com/book/kubernetes-in-action/chapter-9/199>

'*editionHistoryLimit*'으로 이력 목록 길이 관리 가능

spec.strategy.rollingUpdate : 롤아웃 속도 제어 – 1/2

04-node-web-deployment-strategy.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: node-web
```

```
spec:
  replicas: 3
  selector:
    matchLabels:
      app: node-web
```

```
template:
  metadata:
    name: node-web
  labels:
    app: node-web
```

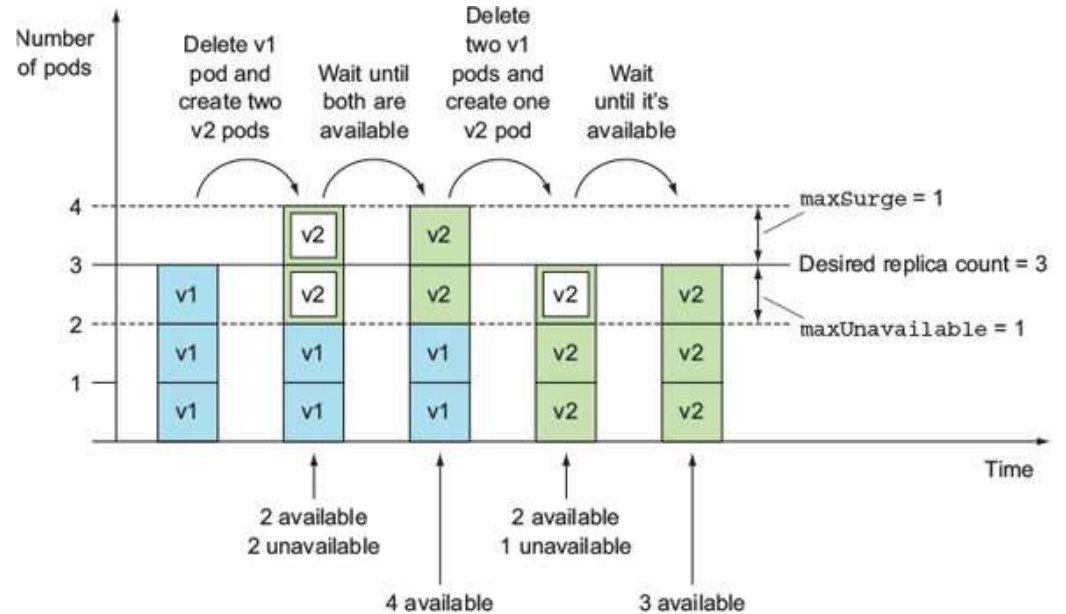
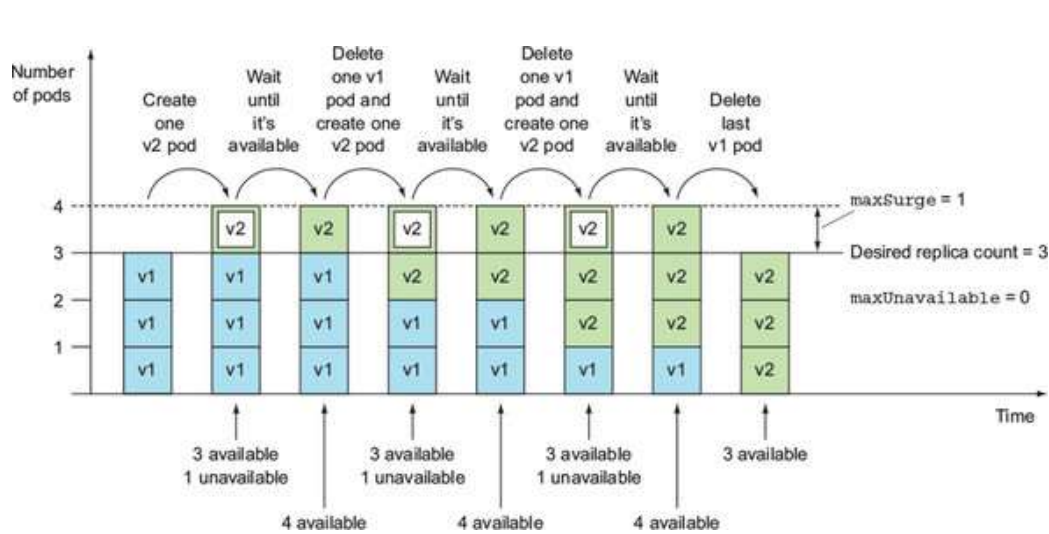
```
spec:
  containers:
    - image: whatwant/node-web:1.0
      name: node-web
      ports:
        - containerPort: 8080
          protocol: TCP
```

```
spec:
  strategy:
    rollingUpdate:
      maxSurge: 1
      maxUnavailable: 1
    type: RollingUpdate
```

속성	설명
maxSurge	디플로이먼트가 의도하는 레플리카 수보다 얼마나 많은 파드 인스턴스 수를 허용할 수 있는지 결정한다. 기본적으로 25%로 설정되고 의도한 개수보다 최대 25% 더 많은 파드 인스턴스가 있을 수 있다. 의도하는 레플리카 수가 4로 설정된 경우 업데이트 중에 동시에 5개 이상의 파드 인스턴스가 실행되지 않는다. 백분율을 절대 숫자로 변환하면 숫자가 반올림된다. 백분율 대신 값이 절댓값일 수도 있다(예: 하나 또는 두 개의 추가 파드가 허용될 수 있음).
maxUnavailable	업데이트 중에 의도하는 레플리카 수를 기준으로 사용할 수 없는 파드 인스턴스 수를 결정한다. 또한 기본적으로 25%로 설정되고 사용 가능한 파드 인스턴스 수는 의도하는 레플리카 수의 75% 이하로 떨어지지 않아야 한다. 여기서 백분율을 절대 숫자로 변환하면 숫자가 내림된다. 의도하는 레플리카 수가 4로 설정되고 백분율이 25%이면 하나의 파드만 사용할 수 없다. 전체 롤아웃 중에 요청을 처리할 수 있는 파드 인스턴스 세 개가 항상 있어야 한다. maxSurge와 마찬가지로 백분율 대신 절댓값을 지정할 수도 있다.

마르코 룩사, 『Kubernetes IN ACTION』, 강인호/황주필/이원기/임찬식 옮김-에이콘출판사/MANNING(2020), 419p

spec.strategy.rollingUpdate : 롤아웃 속도 제어 – 2/2



※ 참고 : <https://livebook.manning.com/book/kubernetes-in-action/chapter-9/215>

rollout – pause/resume

```
> kubectl set image deployment node-web node-web=whatwant/node-web:2.0
```


```
> kubectl rollout pause deployment node-web
```

```
> kubectl rollout resume deployment node-web
```

실수 방지 장치 = minReadySeconds & readinessProbe

- minReadySeconds : Pod를 사용 가능한 것으로 취급하기 전에 새로 만든 Pod를 준비할 시간 지정 = rollout 속도 조정

. Pod가 사용 가능할 때까지 rollout 프로세스 대기 : maxUnavailable 속성

. Pod 준비 여부 판단 : readinessProbe 

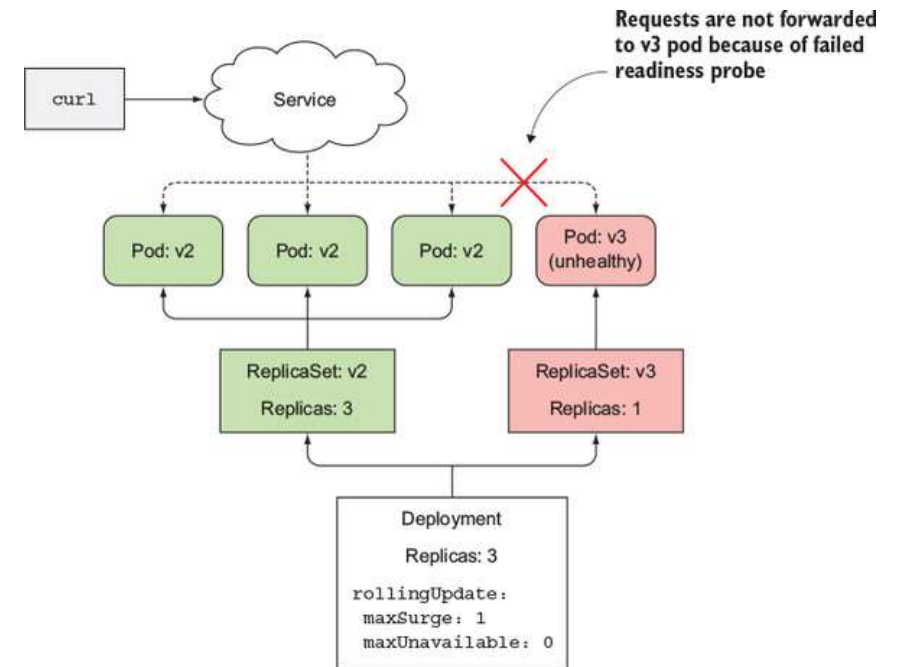
→ 새로 시작한 Pod가 준비 상태를 계속 보고할 수 있도록 minReadySeconds를 큰 값으로 설정

= 버그가 있는 버전이 프로덕션 환경에 배포가 되더라도 큰 혼란을 일으키지 않도록 하는 에어백 같은 역할

- 명시적으로 readinessProbe 설정 해주지 않으면,

Application이 오류를 일으켜도 Container/Pod 준비된 것으로 간주됨

→ 안전한 rollout을 위해 readinessProbe 설정 필요!



※ 참고 : <https://livebook.manning.com/book/kubernetes-in-action/chapter-9/262>

rollout Deadline

- default 10분이 지나면 rollout 실패로 간주
- `progressDeadlineSeconds` 속성으로 설정 가능

Listing 9.12. Seeing the conditions of a Deployment with `kubectl describe`

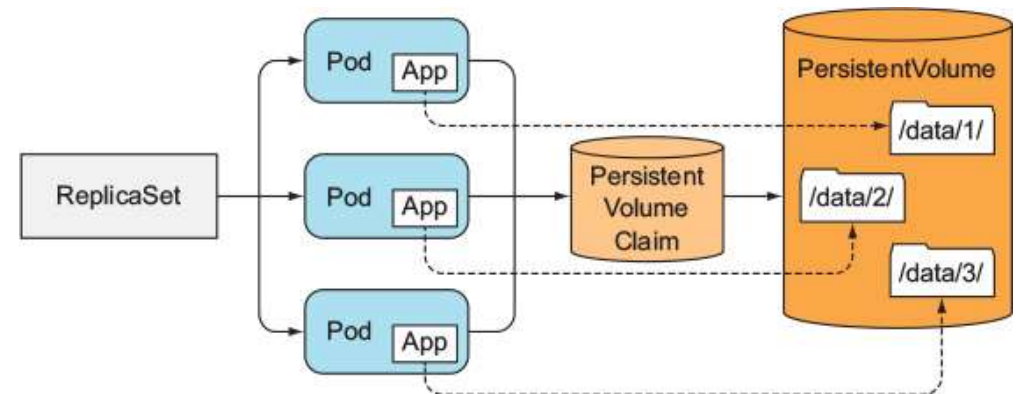
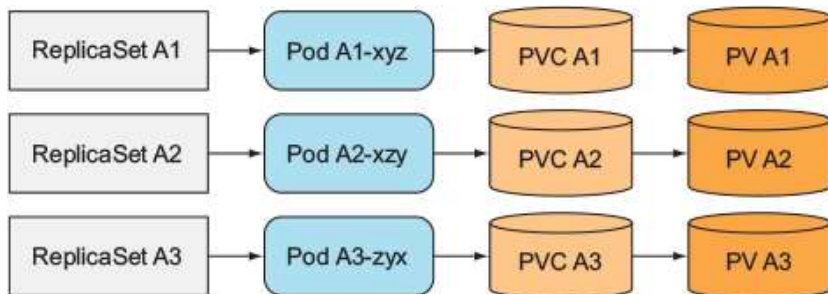
```
1 $ kubectl describe deploy kubia
2 Name: kubia
3 ...
4 Conditions:
5   Type           Status Reason
6   ----           -
7   Available       True  MinimumReplicasAvailable
8   Progressing     False ProgressDeadlineExceeded
```

※ 참고 : <https://livebook.manning.com/book/kubernetes-in-action/chapter-9/269>

StatefulSet

Why StatefulSet ... ? – 1/2

- Pod 인스턴스 별로 독립적인 저장공간을 갖도록 하려면,
 - . 수동 Pod 생성 / 1개의 Pod를 갖는 ReplicaSet 다수 생성 / 동일 Volume을 directory로 구분 사용
- 어렵고 귀찮음

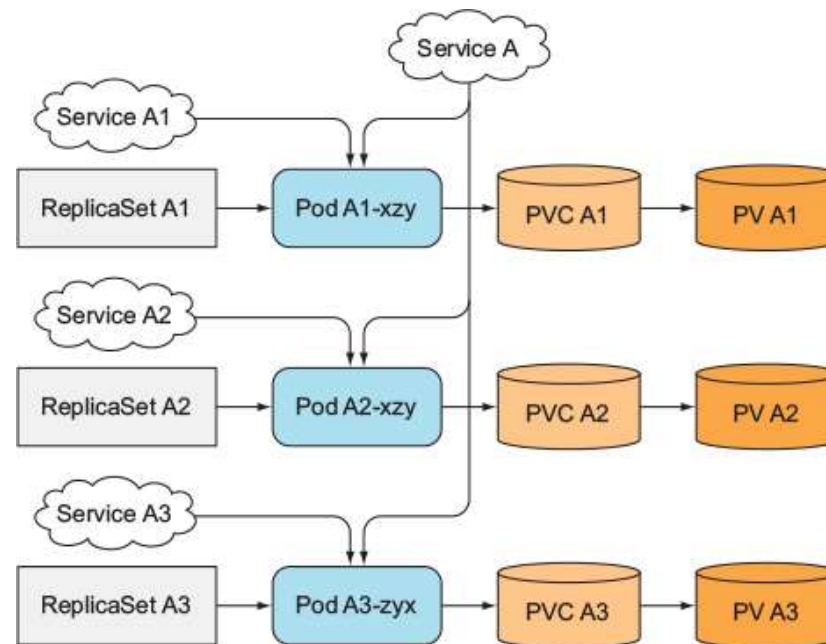


Why StatefulSet ... ? – 2/2

- stable identity를 요구하는 Application 존재 (Pod가 재시작해도 기존 identity 유지 필요)

. identity : hostname, IP

→ 어렵고 귀찮음



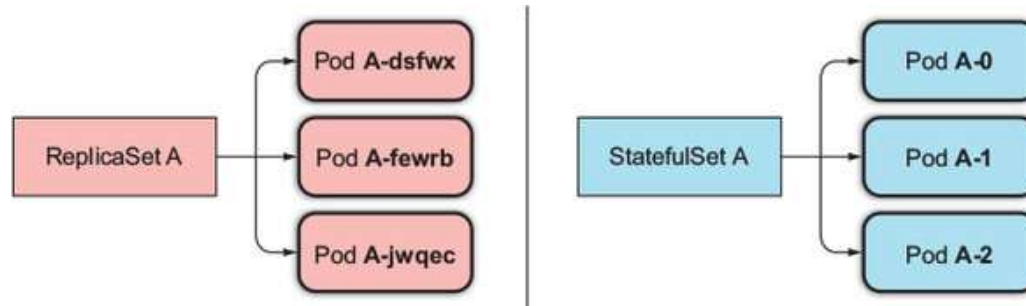
※ 참고 : <https://livebook.manning.com/book/kubernetes-in-action/chapter-10/30>

StatefulSet vs ReplicaSet – 1/4

- 애완동물(Pet) vs 가축(Cattle)

- StatefulSet

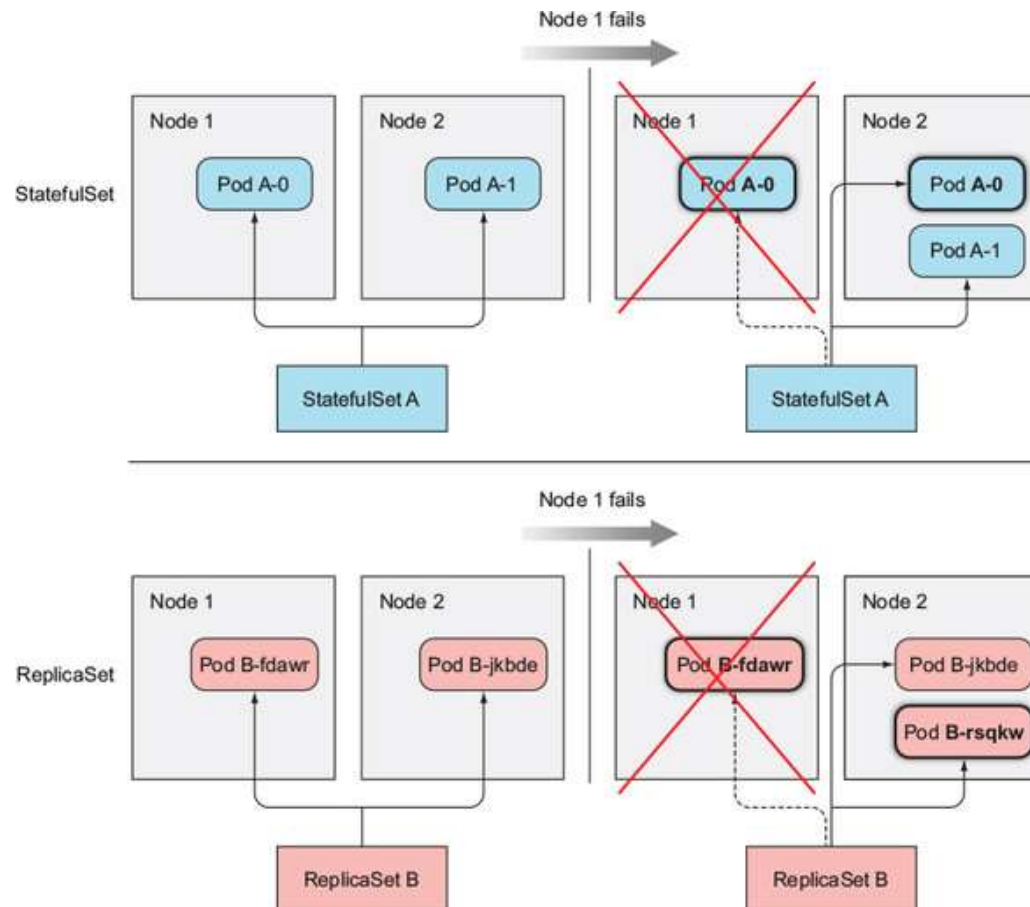
- . 새로운(교체되는/재시작 하는) Pod 인스턴스는 교체되는 Pod와 hostname/IP 동일하게 실행됨
- . 각 Pod는 다른 Pod와 다른 자체 Volume 소유
- . 새로운 Pod 인스턴스의 identity는 예측 가능
- . governing headless service : a-0.foo.default.svc.cluster.local



※ 참고 : <https://livebook.manning.com/book/kubernetes-in-action/chapter-10/49>

StatefulSet vs ReplicaSet – 2/4

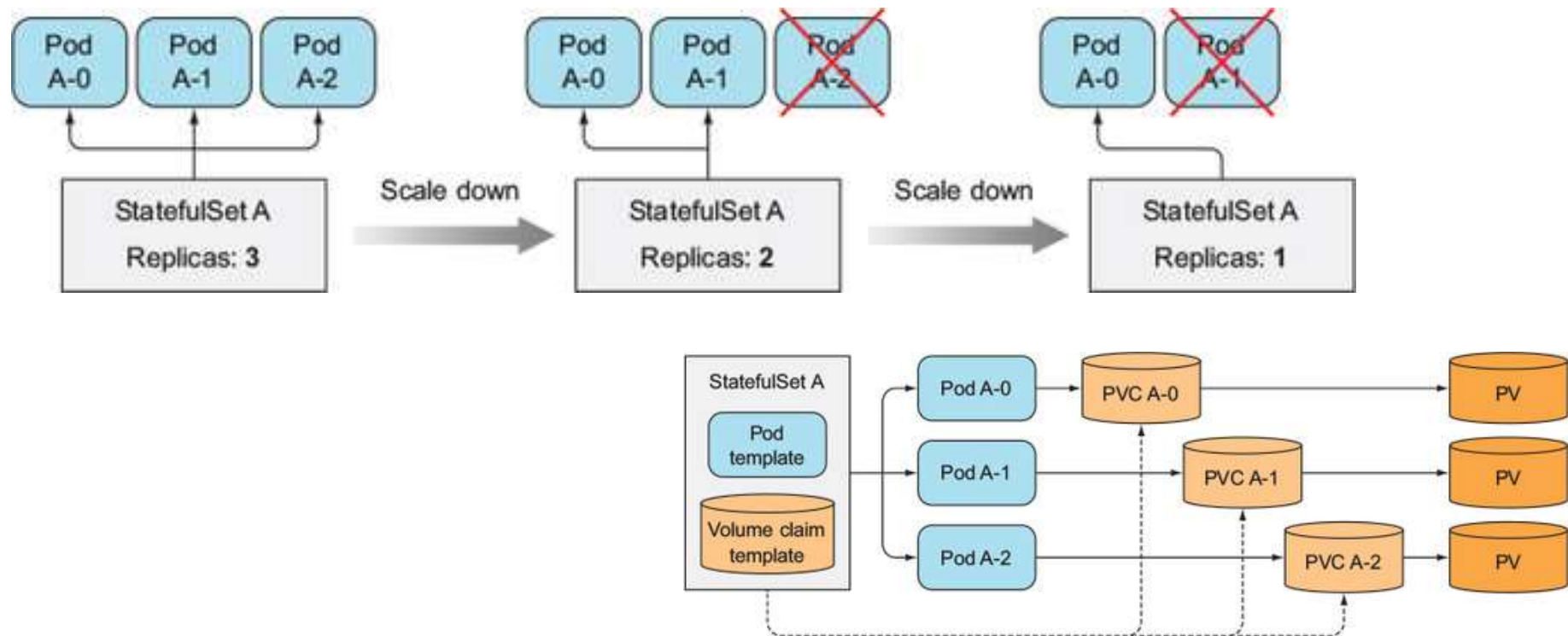
- Restart(Replace)



※ 참고 : <https://livebook.manning.com/book/kubernetes-in-action/chapter-10/61>

StatefulSet vs ReplicaSet – 3/4

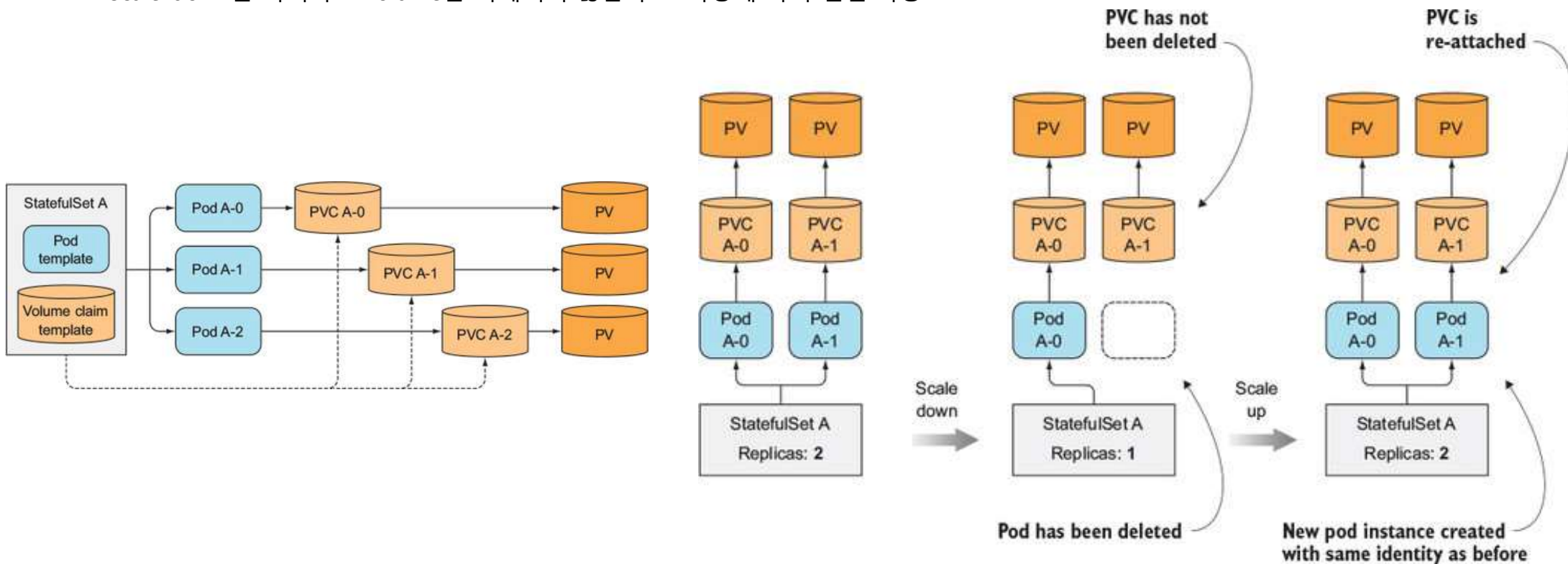
- Scaling



※ 참고 : <https://livebook.manning.com/book/kubernetes-in-action/chapter-10/61>

StatefulSet vs ReplicaSet – 4/4

- Volume claim template
- scale-down을 하더라도 volume은 삭제되지 않는다 → 나중에 다시 연결 가능



실습 준비 : Container

05-app.js

```
const http = require('http');
const os = require('os');
const fs = require('fs');

const dataFile = "/var/data/kubia.txt";

function fileExists(file) {
  try {
    fs.statSync(file);
    return true;
  } catch (e) {
    return false;
  }
}

var handler = function(request, response) {
  if (request.method == 'POST') {
    var file = fs.createWriteStream(dataFile);
    file.on('open', function (fd) {
      request.pipe(file);
      console.log("New data has been received and stored.");
      response.writeHead(200);
      response.end("Data stored on pod " + os.hostname() + "\n");
    });
  }
```

```
} else {
  var data = fileExists(dataFile) ? fs.readFileSync(dataFile, 'utf8') : "No data posted yet";
  response.writeHead(200);
  response.write("You've hit " + os.hostname() + "\n");
  response.end("Data stored on this pod: " + data + "\n");
}
};

var www = http.createServer(handler);
www.listen(8080);
```

05-Dockerfile

```
FROM node:latest

ADD 05-app.js /app.js

ENTRYPOINT ["node", "app.js"]
```

```
> docker build -t whatwant/node-web:3.0 -f 05-Dockerfile .
> docker push whatwant/node-web:3.0
```

```
> docker run -it -d -p 8080:8080 --name web whatwant/node-web:3.0
> curl http://localhost:8080
```

잘 동작하는지 테스트 (하지 않아도 된다)

실습 준비 : PersistentVolume



05-PersistentVolume-list.yaml

```
kind: List
apiVersion: v1
items:
- apiVersion: v1
  kind: PersistentVolume
  metadata:
    name: pv-a
  spec:
    capacity:
      storage: 1Mi
    accessModes:
      - ReadWriteOnce
    persistentVolumeReclaimPolicy: Recycle
    hostPath:
      path: /tmp/pv-a
      type: DirectoryOrCreate

- apiVersion: v1
  kind: PersistentVolume
  metadata:
    name: pv-b
  spec:
    capacity:
      storage: 1Mi
```

```
    accessModes:
      - ReadWriteOnce
    persistentVolumeReclaimPolicy: Recycle
    hostPath:
      path: /tmp/pv-b
      type: DirectoryOrCreate

- apiVersion: v1
  kind: PersistentVolume
  metadata:
    name: pv-c
  spec:
    capacity:
      storage: 1Mi
    accessModes:
      - ReadWriteOnce
    persistentVolumeReclaimPolicy: Recycle
    hostPath:
      path: /tmp/pv-c
      type: DirectoryOrCreate
```

```
> kubectl create -f 05-PersistentVolume-list.yaml
```

실습 준비 : Service

05-headless-service.yaml

```
apiVersion: v1
kind: Service
metadata:
  name: node-web

spec:
  clusterIP: None
  selector:
    app: node-web
  ports:
    - name: http
      port: 80
```

```
> kubectl create -f 05-headless-service.yaml
```

실습 : StatefulSet - 생성

05-statefulset.yaml

```
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: node-web

spec:
  serviceName: node-web
  replicas: 2
  selector:
    matchLabels:
      app: node-web

  template:
    metadata:
      labels:
        app: node-web
    spec:
      containers:
        - name: node-web
          image: whatwant/node-web:3.0
          ports:
            - name: http
              containerPort: 8080
```

```
volumeMounts:
  - name: data
    mountPath: /var/data
```

```
volumeClaimTemplates:
  - metadata:
      name: data
    spec:
      resources:
        requests:
          storage: 1Mi
      accessModes:
        - ReadWriteOnce
```

```
> kubectl create -f 05-statefulset.yaml
```

```
> kubectl get statefulsets -o wide
```

NAME	READY	AGE	CONTAINERS	IMAGES
node-web	2/2	3m5s	node-web	whatwant/node-web:3.0

```
> kubectl get pods
```

NAME	READY	STATUS	RESTARTS	AGE
node-web-0	1/1	Running	0	4m7s
node-web-1	1/1	Running	0	4m2s

하나씩 순차적으로 생성된다.

실습 : StatefulSet - 살펴보기

```
> kubectl get pods node-web-0 -o yaml
```

```
apiVersion: v1
kind: Pod
metadata:
  annotations:
    cni.projectcalico.org/podIP: 10.233.103.126/32
    cni.projectcalico.org/podIPs: 10.233.103.126/32
    creationTimestamp: "2021-05-29T01:07:09Z"
    generateName: node-web-
  labels:
    app: node-web
    controller-revision-hash: node-web-786dc8946
    statefulset.kubernetes.io/pod-name: node-web-0
  ...
spec:
  containers:
  - image: whatwant/node-web:3.0
    imagePullPolicy: IfNotPresent
    name: node-web
    ports:
    - containerPort: 8080
      name: http
      protocol: TCP
```

```
resources: {}
terminationMessagePath: /dev/termination-log
terminationMessagePolicy: File
  volumeMounts:
  - mountPath: /var/data
    name: data
  - mountPath: /var/run/secrets/kubernetes.io/serviceaccount
    name: default-token-4b9h2
    readOnly: true
  ...
  volumes:
  - name: data
    persistentVolumeClaim:
      claimName: data-node-web-0
  - name: default-token-4b9h2
    secret:
      defaultMode: 420
      secretName: default-token-4b9h2
  ...
```

```
> kubectl get persistentvolumeclaims
```

NAME	STATUS	VOLUME	CAPACITY	ACCESS MODES	STORAGECLASS	AGE
data-node-web-0	Bound	pv-c	1Mi	RWO		20m
data-node-web-1	Bound	pv-a	1Mi	RWO		20m

실습 : StatefulSet – 직접 접근하기 with proxy & API

- API Server를 통해 개별 Pod에 직접 Proxy 연결 가능 (StatefulSet에서만 적용되는 것이 아니라 본래 가능)

```
<apiServerHost>:<port>/api/v1/namespaces/default/pods/kubia-0/proxy/<path>
```

```
> kubectl proxy
```

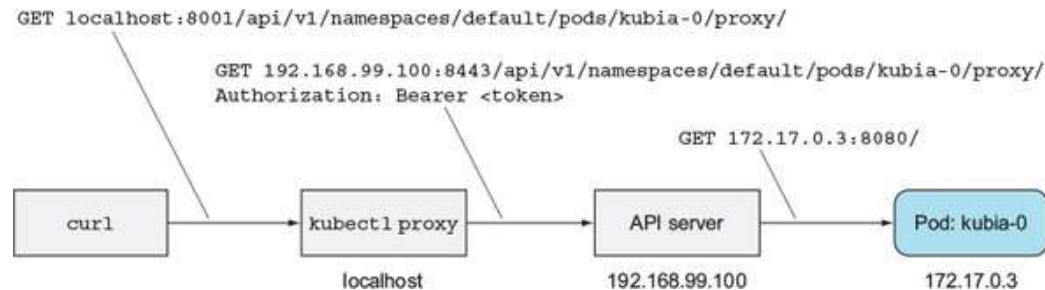
Starting to serve on 127.0.0.1:8001

API Server에 접근은 되어야 하기에...

```
> curl http://localhost:8001/api/v1/namespaces/default/pods/node-web-0/proxy/
```

You've hit node-web-0

Data stored on this pod: No data posted yet



※ 참고 : <https://livebook.manning.com/book/kubernetes-in-action/chapter-10/153>

실습 : StatefulSet – 다른 Pod와의 관련성

- POST를 통해 데이터를 전달해서 저장 시켜보자

```
> curl -X POST -d "Wow" http://localhost:8001/api/v1/namespaces/default/pods/node-web-0/proxy/
```

```
Data stored on pod node-web-0
```

```
> curl http://localhost:8001/api/v1/namespaces/default/pods/node-web-0/proxy/
```

```
You've hit node-web-0
```

```
Data stored on this pod: Wow
```

- `node-web-1` ? → `node-web-0`과 관련이 없다는 것을 알 수 있음 !!!

```
> curl http://localhost:8001/api/v1/namespaces/default/pods/node-web-1/proxy/
```

```
You've hit node-web-1
```

```
Data stored on this pod: No data posted yet
```

실습 : StatefulSet – Volume 재연결

- Pod를 삭제해서, 재시작 된 Pod가 기존 Volume을 다시 mount하는지 확인

```
> kubectl get pods -o wide
```

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE	NOMINATED NODE	READINESS GATES
node-web-0	1/1	Running	0	98m	10.233.103.126	worker2	<none>	<none>
node-web-1	1/1	Running	0	98m	10.233.103.127	worker2	<none>	<none>

```
> kubectl delete pods node-web-0
```

pod "node-web-0" deleted

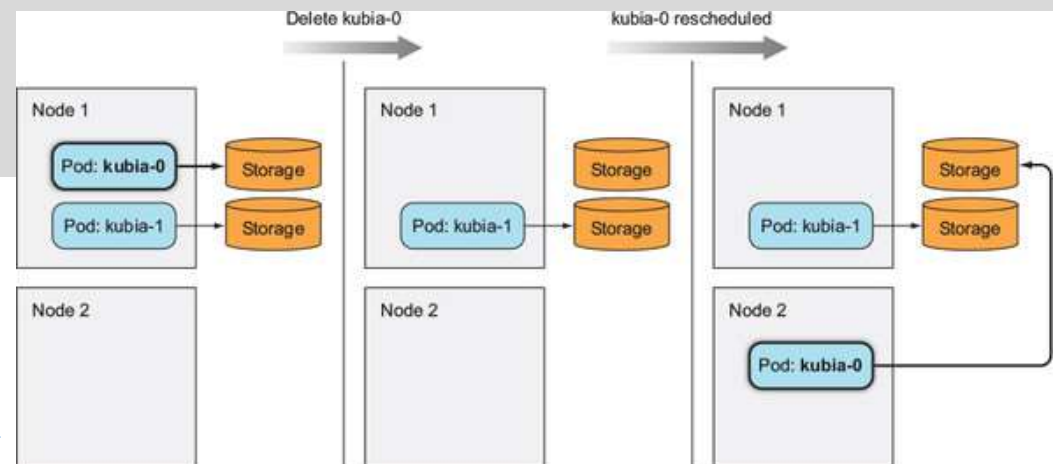
```
> kubectl get pods -o wide
```

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE	NOMINATED NODE	READINESS GATES
node-web-0	1/1	Running	0	3s	10.233.103.128	worker2	<none>	<none>
node-web-1	1/1	Running	0	100m	10.233.103.127	worker2	<none>	<none>

```
> curl http://localhost:8001/api/v1/namespaces/default/pods/node-web-0/proxy/
```

You've hit node-web-0

Data stored on this pod: Wow



※ 참고 : <https://livebook.manning.com/book/kubernetes-in-action/chapter-10/170>

실습 : StatefulSet – 서비스로 접근하기 with proxy & API

- 일반적인 Service를 생성하여, 이를 통해 Pod 접근

05-general-service.yaml

```
apiVersion: v1
kind: Service
metadata:
  name: node-web-general
spec:
  selector:
    app: node-web
  ports:
    - port: 80
      targetPort: 8080
```

```
> kubectl create -f 05-general-service.yaml
```

```
service/node-web-general created
```

```
> kubectl get pods -o wide
```

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE	NOMINATED	NODE	READINESS	GATES
node-web-0	1/1	Running	1	19h	10.233.103.130	worker2	<none>		<none>	
node-web-1	1/1	Running	1	20h	10.233.103.129	worker2	<none>		<none>	

```
> kubectl get service -o wide
```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE	SELECTOR
kubernetes	ClusterIP	10.233.0.1	<none>	443/TCP	25d	<none>
node-web	ClusterIP	None	<none>	80/TCP	20h	app=node-web
node-web-general	ClusterIP	10.233.48.97	<none>	80/TCP	11s	app=node-web

```
> curl http://localhost:8001/api/v1/namespaces/default/services/node-web-general/proxy/
```

```
You've hit node-web-1
```

```
Data stored on this pod: No data posted yet
```

```
> curl http://localhost:8001/api/v1/namespaces/default/services/node-web-general/proxy/
```

```
You've hit node-web-0
```

```
Data stored on this pod: Wow
```

실습 : StatefulSet – Discovering peers (다른 Pod 찾기) 1/2

- DNS 서버의 SRV 레코드 조회 → dig 명령어로 확인 가능

Record	설명
A	도메인의 IP 주소를 갖고 있는 레코드
CNAME	하나의 도메인이나 하위 도메인을 다른 도메인으로 전달하며, IP 주소를 제공하지는 않습니다.
MX	이메일을 이메일 서버로 전송합니다.
TXT	관리자가 텍스트 메모를 레코드에 저장할 수 있습니다.
NS	DNS 항목의 이름 서버를 저장합니다.
SOA	도메인에 대한 관리자 정보를 저장합니다.
SRV	특정 서비스에 대한 포트를 지정합니다.
PTR	리버스 조회에서 도메인 이름을 제공합니다.

Dig (Domain Information Groper) is a powerful command-line tool for querying DNS name servers.

※ 참고 : <https://www.cloudflare.com/ko-kr/learning/dns/dns-records/>

※ 참고 : <https://linuxize.com/post/how-to-use-dig-command-to-query-dns-in-linux/>

실습 : StatefulSet – Discovering peers (다른 Pod 찾기) 2/2

```
> kubectl run -it srvlookup --image=gcr.io/kubernetes-e2e-test-images/dnsutils:1.3 --rm --restart=Never -- dig SRV node-web.default.svc.cluster.local
```

```
; <<>> DiG 9.11.6-P1 <<>> SRV node-web.default.svc.cluster.local
```

```
. . .
```

```
;; OPT PSEUDOSECTION:
```

```
; EDNS: version: 0, flags:; udp: 4096
```

```
; COOKIE: a912679e001ea8fc (echoed)
```

```
;; QUESTION SECTION:
```

```
;node-web.default.svc.cluster.local. IN SRV
```

```
;; ANSWER SECTION:
```

```
node-web.default.svc.cluster.local. 5 IN SRV      0 50 80 node-web-1.node-web.default.svc.cluster.local.
```

```
node-web.default.svc.cluster.local. 5 IN SRV      0 50 80 node-web-0.node-web.default.svc.cluster.local.
```

```
;; ADDITIONAL SECTION:
```

```
node-web-1.node-web.default.svc.cluster.local. 5 IN A 10.233.103.129
```

```
node-web-0.node-web.default.svc.cluster.local. 5 IN A 10.233.103.130
```

```
;; Query time: 1 msec
```

```
;; SERVER: 169.254.25.10#53(169.254.25.10)
```

```
;; WHEN: Sun May 30 15:27:02 UTC 2021
```

```
;; MSG SIZE rcvd: 395
```

```
pod "srvlookup" deleted
```

실습 : StatefulSet Update – container 준비 1/2

06-app.js

```
const http = require('http');
const os = require('os');
const fs = require('fs');
const dns = require('dns');

const dataFile = "/var/data/kubia.txt";
const serviceName = "node-web.default.svc.cluster.local";
const port = 8080;

function fileExists(file) {
  ... 파일 존재 여부 확인 ...
}

function httpGet(reqOptions, callback) {
  ... 웹 접근하여 본문 얻어 오기 ...
}

var handler = function(request, response) {
  if (request.method == 'POST') {
    var file = fs.createWriteStream(dataFile);
    file.on('open', function (fd) {
      request.pipe(file);
      response.writeHead(200);
      response.end("Data stored on pod " + os.hostname() + "\n");
    });
  }
```

```
} else {
  response.writeHead(200);
  if (request.url == '/data') {
    ... 파일 내용 보여주기 (없으면 없다고) ...
  } else {
    response.write("You've hit " + os.hostname() + "\n");
    response.write("Data stored in the cluster:\n");
    dns.resolveSrv(serviceName, function (err, addresses) {
      ...
      addresses.forEach(function (item) {
        var requestOptions = {
          host: item.name,
          port: port,
          path: '/data'
        };
        httpGet(requestOptions, function (returnedData) {
          ... 서버에 접속해서 데이터 읽어오기 ...
        });
      });
    });
  }

  var www = http.createServer(handler);
  www.listen(port);
```

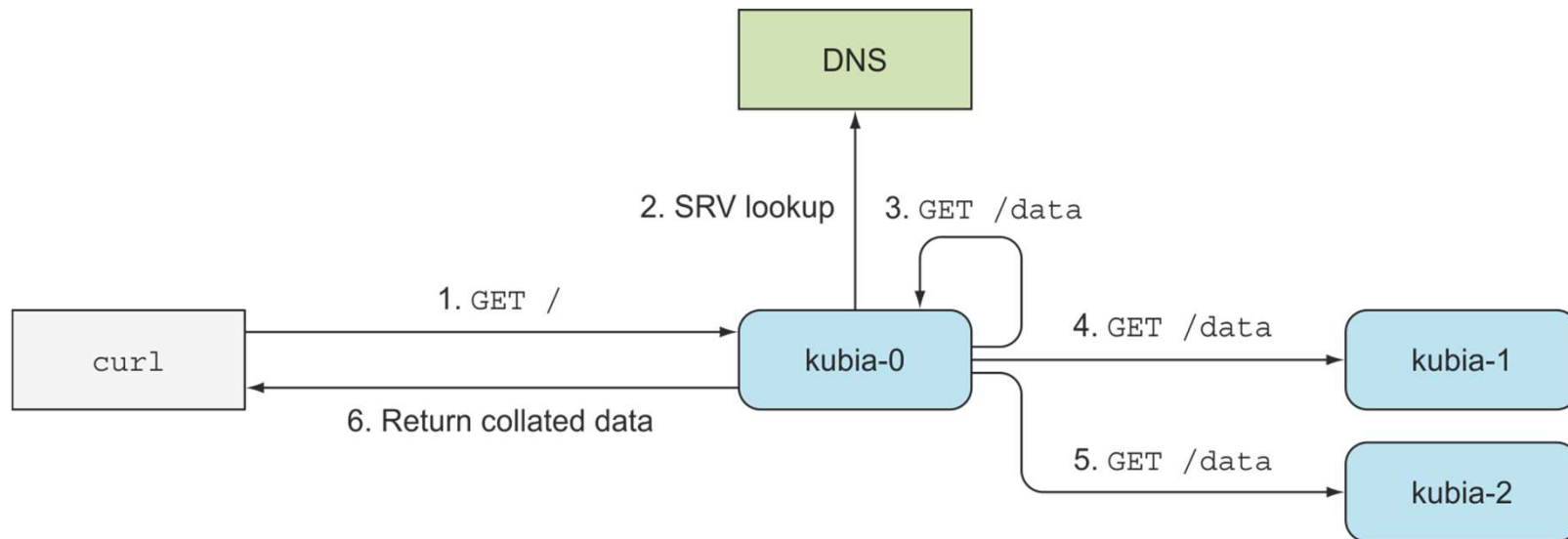
실습 : StatefulSet Update – container 준비 2/2

06-Dockerfile

```
FROM node:latest  
  
ADD 06-app.js /app.js  
  
ENTRYPOINT ["node", "app.js"]
```

```
> docker build -t whatwant/node-web:4.0 -f 06-Dockerfile .
```

```
> docker push whatwant/node-web:4.0
```



※ 참고 : <https://livebook.manning.com/book/kubernetes-in-action/chapter-10/215>

실습 : StatefulSet Update – edit template & delete

```
> kubectl get pods -o wide
```

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE	NOMINATED	NODE	READINESS	GATES
node-web-0	1/1	Running	1	38h	10.233.103.130	worker2	<none>		<none>	
node-web-1	1/1	Running	1	39h	10.233.103.129	worker2	<none>		<none>	

```
> export KUBE_EDITOR=nano
```

```
> kubectl edit statefulsets node-web
```

statefulset.apps/node-web edited

```
> kubectl get pods -o wide
```

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE	NOMINATED	NODE	READINESS	GATES
node-web-0	1/1	Running	1	38h	10.233.103.130	worker2	<none>		<none>	
node-web-1	1/1	Running	1	39h	10.233.103.129	worker2	<none>		<none>	
node-web-2	1/1	Running	0	4s	10.233.103.135	worker2	<none>		<none>	

```
> kubectl delete pods node-web-0 node-web-1
```

pod "node-web-0" deleted

pod "node-web-1" deleted

```
> kubectl get pods -o wide
```

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE	NOMINATED	NODE	READINESS	GATES
node-web-0	1/1	Running	0	21s	10.233.103.136	worker2	<none>		<none>	
node-web-1	1/1	Running	0	18s	10.233.103.137	worker2	<none>		<none>	
node-web-2	1/1	Running	0	2m20s	10.233.103.135	worker2	<none>		<none>	

...

spec:

podManagementPolicy: OrderedReady

replicas: 3

revisionHistoryLimit: 10

selector:

matchLabels:

app: node-web

serviceName: node-web

template:

metadata:

creationTimestamp: null

labels:

app: node-web

spec:

containers:

- image: whatwant/node-web:4.0

imagePullPolicy: IfNotPresent

name: node-web

...

실습 : StatefulSet Update – 결과 확인

```
> curl http://localhost:8001/api/v1/namespaces/default/services/node-web-general/proxy/
```

You've hit node-web-1

Data stored in the cluster:

- node-web-1.node-web.default.svc.cluster.local: No data posted yet
- node-web-2.node-web.default.svc.cluster.local: No data posted yet
- node-web-0.node-web.default.svc.cluster.local: Wow

chani ▶ /srv/workspace/managing-kubernetes/07-week ▶ ↵ main ±▶

```
> curl -X POST -d "Red Sun" http://localhost:8001/api/v1/namespaces/default/services/node-web-general/proxy/
```

Data stored on pod node-web-0

```
> curl http://localhost:8001/api/v1/namespaces/default/services/node-web-general/proxy/
```

You've hit node-web-2

Data stored in the cluster:

- node-web-1.node-web.default.svc.cluster.local: No data posted yet
- node-web-2.node-web.default.svc.cluster.local: No data posted yet
- node-web-0.node-web.default.svc.cluster.local: Red Sun

```
> curl -X POST -d "Blue Moon" http://localhost:8001/api/v1/namespaces/default/services/node-web-general/proxy/
```

Data stored on pod node-web-1

```
> curl http://localhost:8001/api/v1/namespaces/default/services/node-web-general/proxy/
```

You've hit node-web-1

Data stored in the cluster:

- node-web-1.node-web.default.svc.cluster.local: Blue Moon
- node-web-0.node-web.default.svc.cluster.local: Red Sun
- node-web-2.node-web.default.svc.cluster.local: No data posted yet

실습 : StatefulSet Rolling Update – container 준비

- Kubernetes v1.7 부터는 StatefulSet에서도 Rolling Update를 지원해준다고 해서...

07-app.js

```
...  
    response.write("You've hit " + os.hostname() + " for v5 \n");  
...
```

07-Dockerfile

```
FROM node:latest  
  
ADD 07-app.js /app.js  
  
ENTRYPOINT ["node", "app.js"]
```

```
> docker build -t whatwant/node-web:5.0 -f 07-Dockerfile .
```

```
> docker push whatwant/node-web:5.0
```


실습 : StatefulSet Rolling Update – 실행

```
> kubectl set image statefulset node-web node-web=whatwant/node-web:5.0
```

```
statefulset.apps/node-web image updated
```

```
> sh -c 'while true; do curl http://localhost:8001/api/v1/namespaces/default/services/node-web-general/proxy/; sleep 2; done'
```

You've hit node-web-2

Data stored in the cluster:

- node-web-2.node-web.default.svc.cluster.local: No data posted yet
- node-web-0.node-web.default.svc.cluster.local: Red Sun
- node-web-1.node-web.default.svc.cluster.local: Blue Moon

You've hit node-web-1

Data stored in the cluster:

- node-web-1.node-web.default.svc.cluster.local: Blue Moon
- node-web-0.node-web.default.svc.cluster.local: Red Sun

You've hit node-web-2 for v5

Data stored in the cluster:

- node-web-2.node-web.default.svc.cluster.local: No data posted yet
- node-web-0.node-web.default.svc.cluster.local: Red Sun

You've hit node-web-0

Data stored in the cluster:

- node-web-0.node-web.default.svc.cluster.local: Red Sun
- node-web-2.node-web.default.svc.cluster.local: No data posted yet

You've hit node-web-1 for v5

Data stored in the cluster:

- node-web-2.node-web.default.svc.cluster.local: No data posted yet
- node-web-1.node-web.default.svc.cluster.local: Blue Moon

You've hit node-web-2 for v5

Data stored in the cluster:

- node-web-2.node-web.default.svc.cluster.local: No data posted yet
- node-web-1.node-web.default.svc.cluster.local: Blue Moon

You've hit node-web-0 for v5

Data stored in the cluster:

- node-web-0.node-web.default.svc.cluster.local: Red Sun
- node-web-1.node-web.default.svc.cluster.local: Blue Moon
- node-web-2.node-web.default.svc.cluster.local: No data posted yet

You've hit node-web-1 for v5

Data stored in the cluster:

- node-web-1.node-web.default.svc.cluster.local: Blue Moon
- node-web-0.node-web.default.svc.cluster.local: Red Sun
- node-web-2.node-web.default.svc.cluster.local: No data posted yet

실습 : StatefulSet 장애 – Worker Node 오류 1/2

- Stateful Pod가 실행 중인 Worker Node에 장애가 발생하면 ?

> kubectl get nodes -o wide

NAME	STATUS	ROLES	AGE	VERSION	INTERNAL-IP	EXTERNAL-IP	OS-IMAGE	KERNEL-VERSION	CONTAINER-RUNTIME
master-stg	Ready	control-plane,master	26d	v1.20.6	192.168.100.111	<none>	Ubuntu 20.04.2 LTS	5.4.0-73-generic	docker://19.3.15
worker1	Ready	<none>	26d	v1.20.6	192.168.100.112	<none>	Ubuntu 20.04.2 LTS	5.4.0-73-generic	docker://19.3.15
worker2	Ready	<none>	26d	v1.20.6	192.168.100.113	<none>	Ubuntu 20.04.2 LTS	5.4.0-73-generic	docker://19.3.15

> kubectl get pods -o wide

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE	NOMINATED NODE	READINESS GATES
node-web-0	1/1	Running	0	17m	10.233.103.144	worker2	<none>	<none>
node-web-1	1/1	Running	0	18m	10.233.103.143	worker2	<none>	<none>
node-web-2	1/1	Running	0	18m	10.233.103.142	worker2	<none>	<none>

> kubectl get nodes -o wide *worker2 Node 전원을 꺼버렸다.*

NAME	STATUS	ROLES	AGE	VERSION	INTERNAL-IP	EXTERNAL-IP	OS-IMAGE	KERNEL-VERSION	CONTAINER-RUNTIME
master-stg	Ready	control-plane,master	26d	v1.20.6	192.168.100.111	<none>	Ubuntu 20.04.2 LTS	5.4.0-73-generic	docker://19.3.15
worker1	Ready	<none>	26d	v1.20.6	192.168.100.112	<none>	Ubuntu 20.04.2 LTS	5.4.0-73-generic	docker://19.3.15
worker2	NotReady	<none>	26d	v1.20.6	192.168.100.113	<none>	Ubuntu 20.04.2 LTS	5.4.0-73-generic	docker://19.3.15

> kubectl get pods -o wide

책에서는 Unknown이 되었다가 Terminating 된다고 했는데, Ready 상태로 계속 나오다가 한 참 후에 Terminating으로 바뀌었다..

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE	NOMINATED NODE	READINESS GATES
node-web-0	1/1	Terminating	0	27m	10.233.103.144	worker2	<none>	<none>
node-web-1	1/1	Terminating	0	28m	10.233.103.143	worker2	<none>	<none>
node-web-2	1/1	Terminating	0	28m	10.233.103.142	worker2	<none>	<none>

실습 : StatefulSet 장애 – Worker Node 오류 2/2

```
> kubectl delete pods node-web-0
```

```
pod "node-web-0" deleted
```

```
(종료 안됨)
```

```
^C
```

책에서는 삭제가 된 것처럼 된다고 하는데, 실제 해보면 종료가 안되고 계속 대기중인 상태로 되어있다.

```
> kubectl delete pods node-web-0 --force --grace-period 0
```

```
warning: Immediate deletion does not wait for confirmation that the running resource has been terminated. The resource may continue to run on the cluster indefinitely.
```

```
pod "node-web-0" force deleted
```

강제로 삭제를 해야 한다.

```
> kubectl get pods -o wide
```

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE	NOMINATED NODE	READINESS GATES
node-web-0	1/1	Running	0	5s	10.233.110.82	worker1	<none>	<none>
node-web-1	1/1	Terminating	0	46m	10.233.103.143	worker2	<none>	<none>
node-web-2	1/1	Terminating	0	47m	10.233.103.142	worker2	<none>	<none>

<https://kahoot.it/>