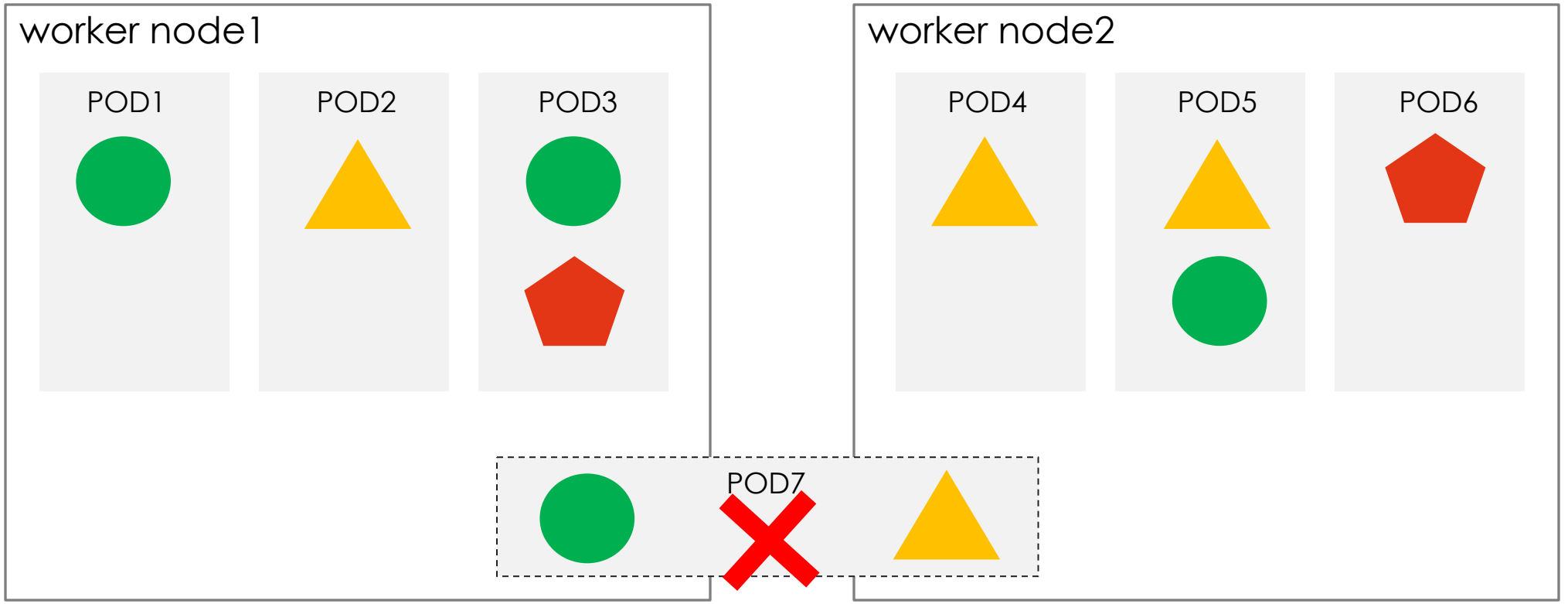
POD

POD

- POD(파드) 는 쿠버네티스 애플리케이션의 **가장 작은 기본 실행 단위** (만들고 배포할 수 있는 가장 작은 단위)
- 쿠버네티스 POD에서 사용되는 가장 대표적인 컨테이너 런타임은 Containerd (OCI 따름 : https://opencontainers.org/)
- 파드당 컨테이너 비율은 대체로 **1:1 이 관례**, 파드당 여러개의 컨테이너가 포함되는 경우도 있음
- POD내의 컨테이너는 **오로지 하나의 노드 내에만 존재** 합니다. 노드를 걸쳐서 파드 가 존재 하지 않음
- **동일 파드** 내의 **컨테이너**는 **네트워크 및 볼륨을 공유** 할 수 있습니다.



Kubernetes Network Basic

- K8S에 있는 POD들은 단순하고, 공유 가능한 네트워크 Address 주소 값을 가진다 (flat Natwork)
- 각각의 POD은 각각의 IP주소 값을 가지고 있으며, 이 IP를 이용해서 통신 허용
- NAT(Network Address Translation) gateway 같은 장비 없이- 마치 Local Area Network(LAN)처럼 통신이 가능.
- POD 내의 여러개 Container 는 서로 다른 포트를 통해 서비스 해야함

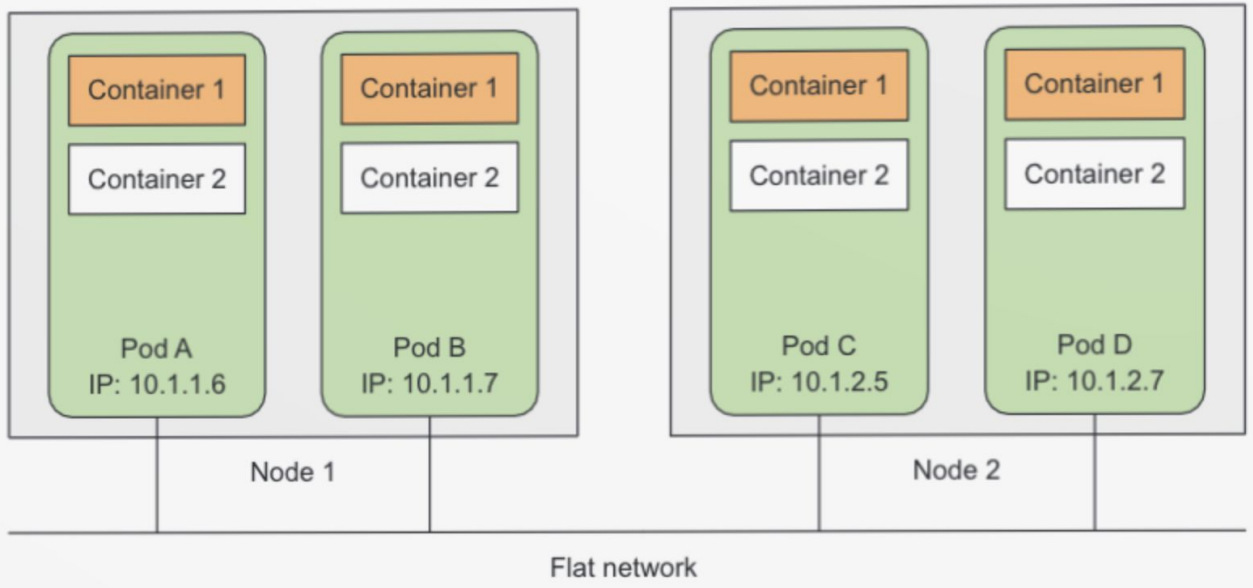


Figure 3.2 Each pod gets a routable IP address and all other pods see the pod under that IP address.

k8s 설치시에. pod network 지정 가능
--pod-network-cidr=192.168.0.0/16

POD 기본 명령

- pod 보기 명령어

```
# 기본 pod 조회 명령어
] kubectl get pods
# 축약어 사용
] kubectl get po
# 상세 정보까지 출력
] kubectl get po -o wide
# 레이블 까지 출력
] kubectl get po --show-labels
```

- K8s CLI를 이용한 POD 생성

```
# POD와 함께 Replication Controller 까지 생성 (Deprecate 될 예정)
] kubectl run <POD-NAME> --image=<IMAGE-NAME> --port=<SERVICE-PORT>
] kubectl run <POD-NAME> --image=<IMAGE-NAME> --port=<SERVICE-PORT>
```

POD 기본 명령

- pod 보기 명령어

```
# 기본 pod 조회 명령어
] kubectl get pod goapp-project-bcv5q -o yaml
# 축약어 사용
] kubectl create -f goapp.yaml
# 상세 정보까지 출력
] kubectl logs goapp-pod
```

- K8s CLI를 이용한 POD 생성

```
# POD와 함께 Replication Controller 까지 생성 (Deprecate 될 예정)
] kubectl run <POD-NAME> --image=<IMAGE-NAME> --port=<SERVICE-PORT> --generator= run/v1
# POD만 생성
] kubectl run <POD-NAME> --image=<IMAGE-NAME> --port=<SERVICE-PORT> --generator= run-pod/v1
```

POD 생성 Template 사용하기

```
apiVersion: v1
kind: Pod
metadata:
  name: myapp-pod
  labels:
    app: myapp
spec:
  containers:
  - name: myapp-container
    image: busybox
    command: ['sh', '-c', 'echo 안녕하세요 쿠버네티스! &&
sleep 3600']
```

Key	설명
apiVersion	k8s api version
kind	k8s 리소스 타입
metadata	이름,레이블 등의 부가정보
spec	컨테이너 정보

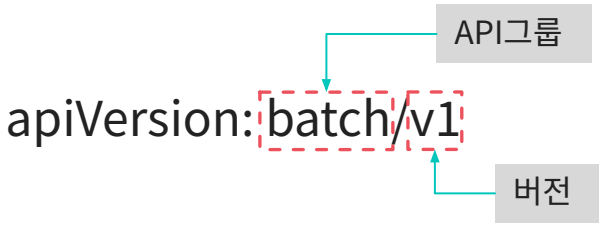
일반적으로 kubectl run 을 사용하는 것보다 실제 운영
환경에서는 yaml 파일을 이용해서 생성 합니다.
무엇보다 pod 에 대한 이력을 관리 하는 것이 중요

```
# yaml 파일을 이용해서 생성 하기
] kubectl create -f goapp.yaml
```

API 그룹

API 그룹은 쿠버네티스 API를 더 쉽게 확장하게 해준다.

Group	Version
admissionregistration.k8s.io	v1, v1beta1
apiextensions.k8s.io	v1, v1beta1
apiregistration.k8s.io	v1, v1beta1
apps	v1
authentication.k8s.io	v1, v1beta1
authorization.k8s.io	v1, v1beta1
autoscaling	v1, v2beta2, v2beta1
batch	v1, v1beta1
certificates.k8s.io	v1, v1beta1
coordination.k8s.io	v1, v1beta1
core	v1
discovery.k8s.io	v1, v1beta1
events.k8s.io	v1, v1beta1
extensions	v1beta1
flowcontrol.apiserver.k8s.io	v1beta1



단, core API 의 경우는 그룹을 명시 하지 않아도 됨

Group	Version
internal.apiserver.k8s.io	v1alpha1
networking.k8s.io	v1, v1beta1
node.k8s.io	v1, v1beta1, v1alpha1
policy	v1, v1beta1
rbac.authorization.k8s.io	v1, v1beta1, v1alpha1
scheduling.k8s.io	v1, v1beta1, v1alpha1
storage.k8s.io	v1, v1beta1, v1alpha1

알파(Alpha)

- 기본적으로 비활성화
- 활성화 하면 버그에 노출 될 수 있음
- 다음 릴리즈 에서 언제든지 삭제 되거나 변경 될 수 있음

(예: v1alpha1)

베타(Beta)

- 기본적으로 활성화 되어 있음. 코드테스트를 많이 했으며, 안전함
- 전반적인 기능에 대한 기술 지원이 중단되지 않음 (지속 가능함)
- 문법이나 사용 방식이 다음 릴리즈에서 바뀔 수 있음

(예: v2beta3)

안정화(Stable)

- 버전 이름이 vX 와 같이 표기 한다. 여기서 X 는 정수 이다.

(예: v1)

Pod 삭제

```
# 특정 Pod 삭제
$ kubectl delete pod <Pod-Name>

# 현재 네임스페이스의 모든 Pod 삭제
$ kubectl delete pod --all

# 네임스페이스내의 거의 모든 리소스 삭제
$ kubectl delete all --all
```

all –all 옵션을 사용해서 삭제해도 지워지지 않는 리소스가 있음 (예: secret 등)

생성된 POD에서 yaml 파일 정보 추출하기

```
#] kubectl get pod <POD-NAME> -o yaml
```