OpenInfra Community Days Korea

쿠버네티스 레이블 자세히 살펴보기

임찬식, 라인플러스

Who Am I?

- 임찬식
- 소프트웨어 엔지니어 @ 라인플러스
- 쿠버네티스 플랫폼 운영 및 오퍼레이터 개발
- OpenInfra & Cloud Native Days Korea 2022
  쿠버네티스 오브젝트의 Conditions 속성 살펴보기

Index

쿠버네티스 레이블 기초

쿠버네티스 레이블

- 오브젝트의 특성을 식별하는데 사용하는 키와 값의 쌍
- 레이블은 생성 후 언제든지 수정 가능
- 오브젝트의 레이블 키는 고유한 값

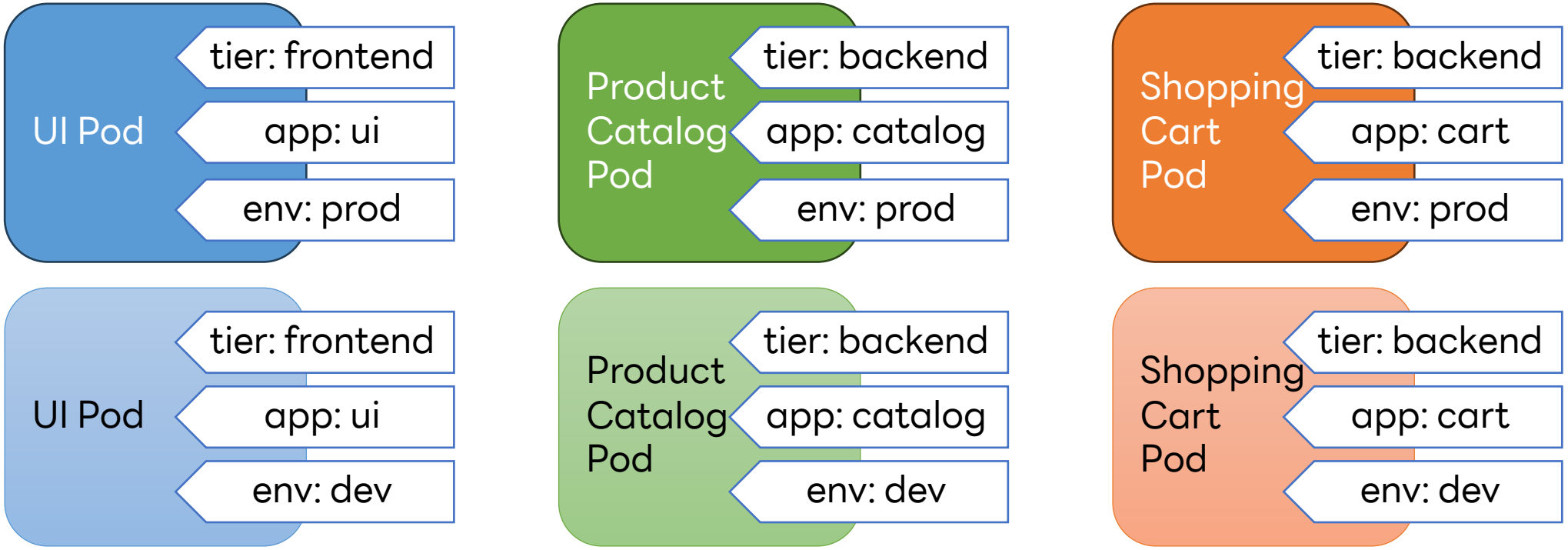
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: test-app
  labels:
    tier: frontend
    app: ui
    env: prod
```

레이블 주요 역할

- 리소스 오브젝트 선택
- 리소스 오브젝트 분류
- 워크로드 스케줄링
- 서비스 디스커버리
- 네트워크 및 보안 등의 정책 적용

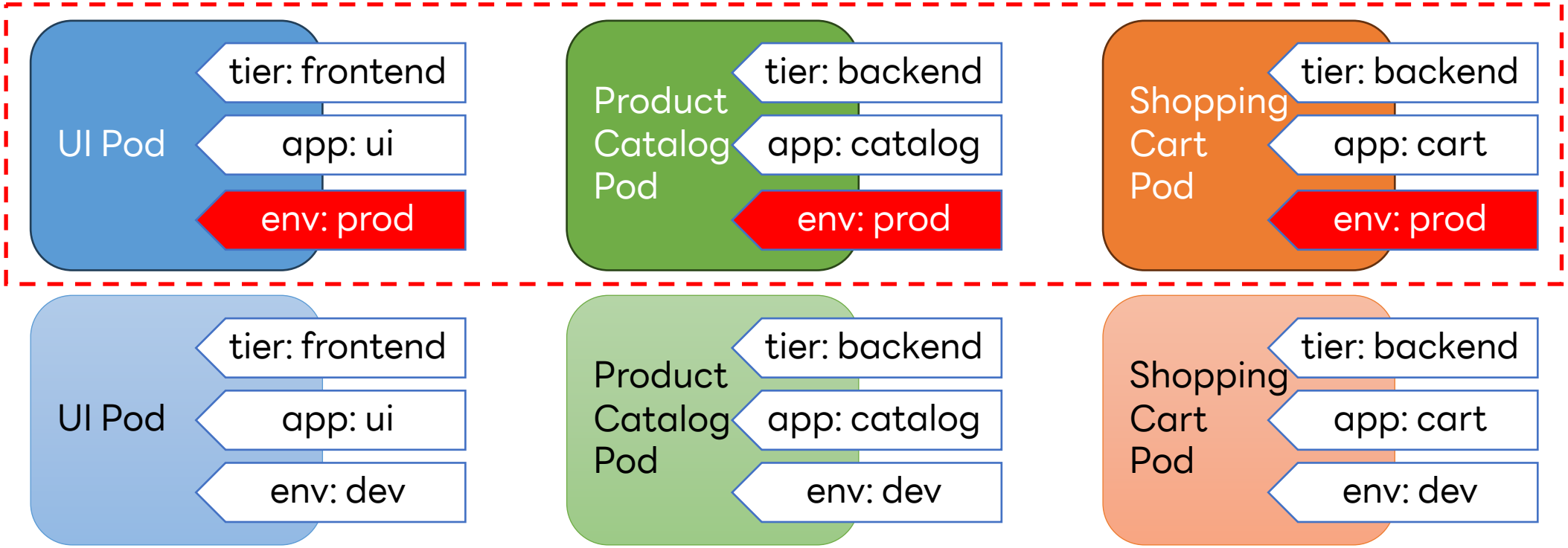
리소스 오브젝트 선택

- 레이블 키와 값을 이용해 원하는 조건의 리소스 선택
- 파드, 서비스, 디플로이먼트 등 다양한 리소스 선택 및 관리



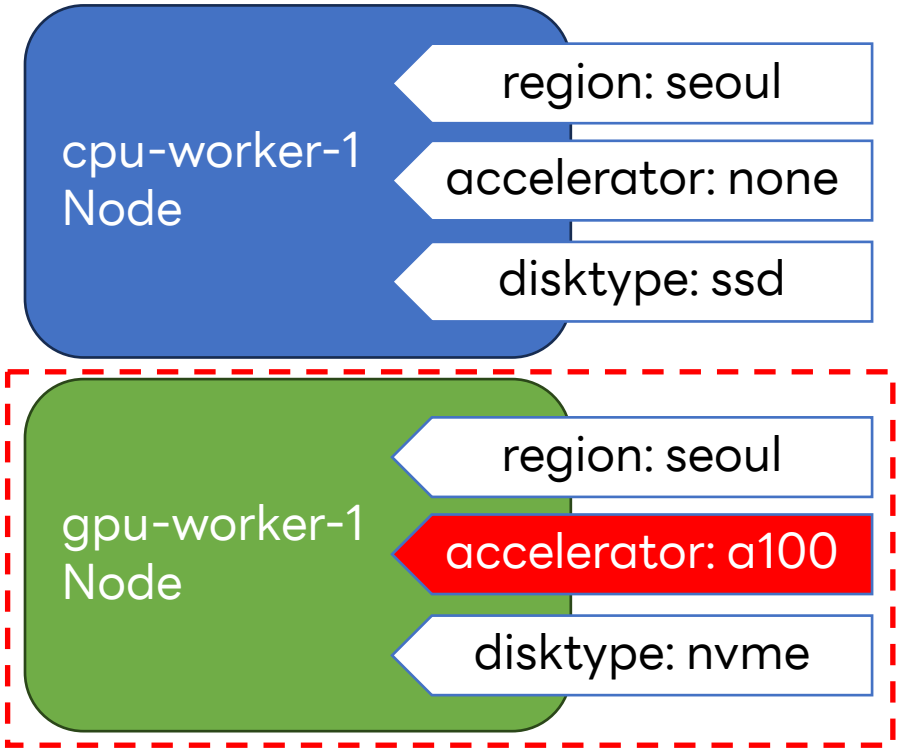
리소스 오브젝트 분류

- 레이블을 이용해 리소스를 논리적 그룹으로 분류
- env=prod 레이블을 이용해 애플리케이션 리소스 분류



워크로드 스케줄링

- 레이블을 이용해 특정 조건을 가진 노드에 워크로드 스케줄링
- GPU 자원이 필요한 워크로드를 배포하기 위해 레이블 사용

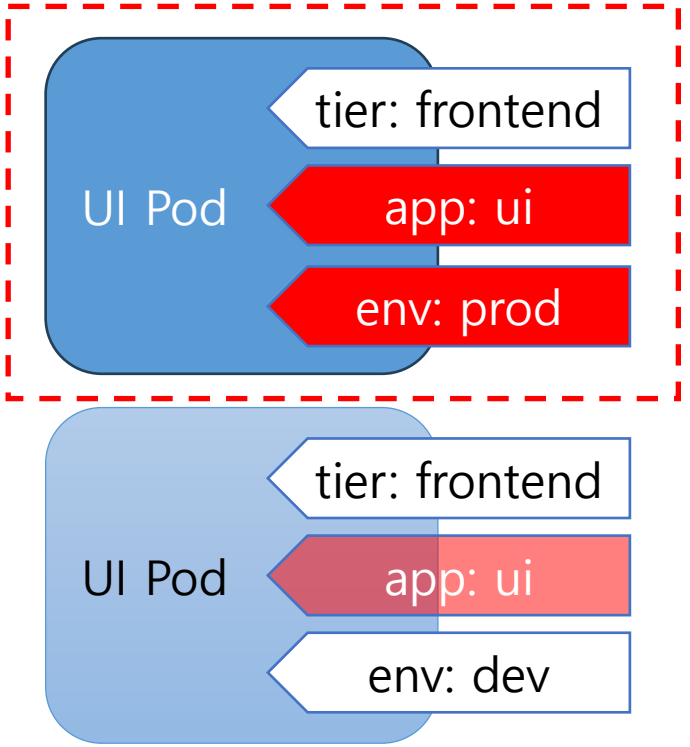



```yaml
spec:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          - key: accelerator
            operator: In
            values:
            - a100
```

서비스 디스커버리

- 서비스는 레이블 셀렉터를 이용해 특정 파드 집합을 선택
- 외부 네트워크를 통해 들어온 요청은 서비스를 통해 파드로 전달

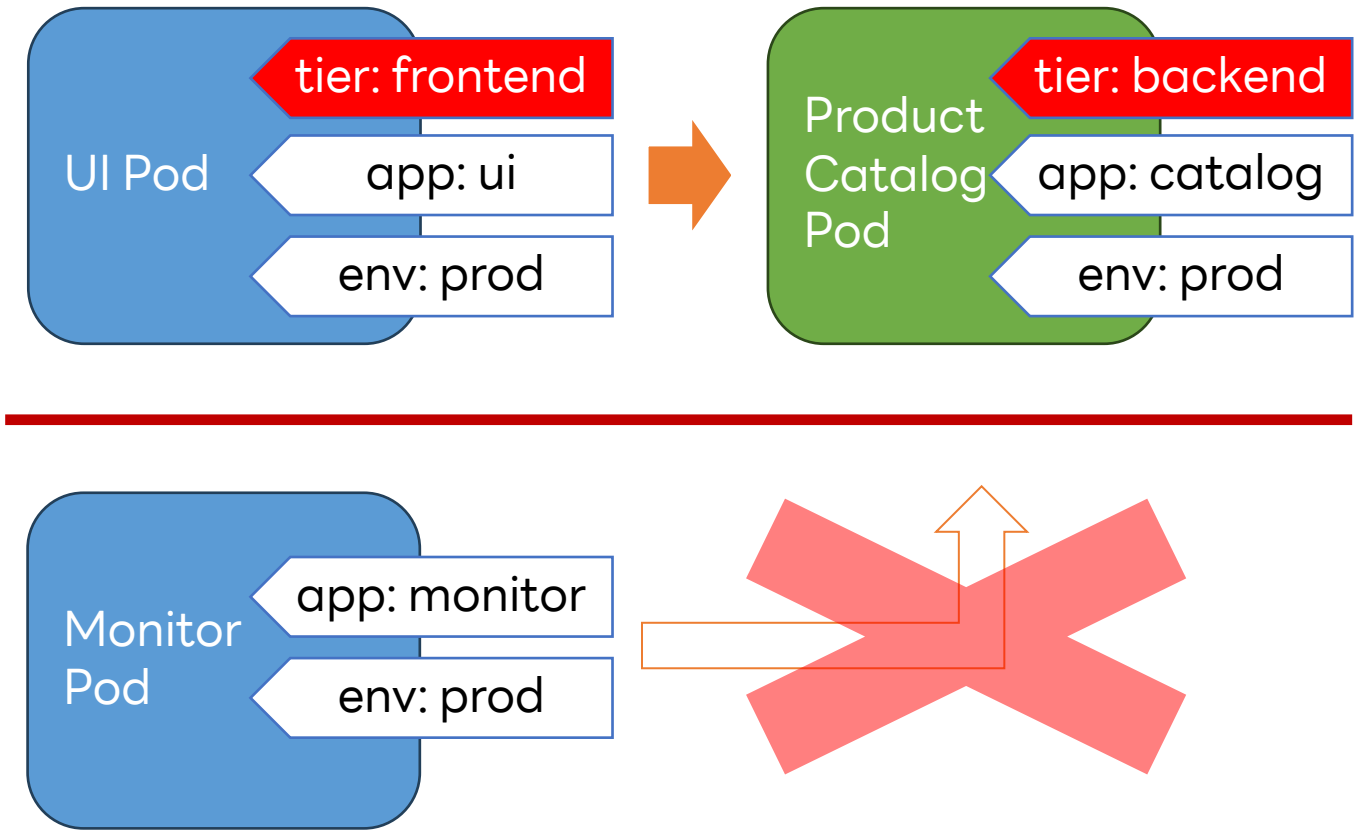
```yaml
apiVersion: v1
kind: Service
metadata:
  name: web-svc-ui-prod
spec:
  selector:
    app: ui
    env: prod
  ports:
    - protocol: TCP
      port: 80
      targetPort: 9376
```



레이블을 이용한 보안 정책 적용

- 레이블을 이용해 특정 파드에 대한 네트워크 정책 적용

```yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: backend-tier-network-policy
  namespace: web-app
spec:
  podSelector:
    matchLabels:
      tier: backend
  policyTypes:
    - Ingress
  ingress:
    - from:
        - podSelector:
            matchLabels:
              tier: frontend
      ports:
        - protocol: TCP
          port: 8080
```



레이블과 어노테이션 공통점

- 쿠버네티스 오브젝트에 메타데이터로 추가해 사용
- 키-값 구조를 사용해 정보 저장
- 레이블과 어노테이션 모두 언제든지 추가, 수정, 삭제 가능

레이블과 어노테이션 차이점

- 목적
  - 레이블은 오브젝트를 식별하고 선택할 때 사용
  - 어노테이션은 오브젝트에 추가 정보를 제공할 때 사용
- 질의
  - 레이블을 이용해 원하는 오브젝트 검색 가능
- 제약 조건
  - 레이블은 값에 넣을 수 있는 문자열을 최대 63자까지 허용
  - 어노테이션은 (현재) 256KB 길이까지 허용

권장 레이블

쿠버네티스 애플리케이션 권장 레이블

- 모든 도구들이 이해할 수 있는 공통의 방식으로 오브젝트 식별
- 다양한 도구들이 상호 운용적으로 작동할 수 있도록 보장
- 공통적으로 필요한 항목들을 표준화해 일관성 유지
- 여러 팀이나 프로젝트가 동일한 클러스터를 공유할 때 유용
- 공통 접두사로 app.kubernetes.io 사용

app.kubernetes.io 권장 레이블 (1)

레이블 키	설명	예시
app.kubernetes.io/name	애플리케이션 이름	mysql
app.kubernetes.io/instance	애플리케이션 인스턴스 식별용 고유 이름	mysql-abcxyz
app.kubernetes.io/version	애플리케이션의 현재 버전	5.7.21
app.kubernets.io/part-of	해당 애플리케이션이 포함된 전체 이름	wordpress
app.kubernetes.io/managed-by	애플리케이션 작동을 관리하는 도구	helm

app.kubernetes.io 권장 레이블 (2)

- 권장 레이블을 추가한 스테이트풀셋 오브젝트 예시

```yaml
apiVersion: apps/v1
kind: StatefulSet
metadata:
  labels:
    app.kubernetes.io/name: mysql
    app.kubernetes.io/instance: mysql-abcxzy
    app.kubernetes.io/version: "5.7.21"
    app.kubernetes.io/component: database
    app.kubernetes.io/part-of: wordpress
    app.kubernetes.io/managed-by: helm
```

레이블 제약 조건

레이블 키 제약 조건 (1)

- 레이블 키는 {접두사} + ‘/’ + {이름} 형태로 구성
- 키의 이름 부분
  - 63자 이하
  - 시작과 끝은 알파벳과 숫자만 가능
  - 알파벳, 숫자, 대시(-), 밑줄(_), 점(.) 문자 사용 가능
- 접두사는 선택 사항
  - 전체 253자 이하
  - 점(.)과 슬래시로 구분되는 DNS 레이블 (예: sub-domain.domain.com)

레이블 키 제약 조건 (2)

- 접두사는 RFC 1123 서브도메인 제약 조건 적용
  - 최대 253자를 넘을 수 없음
  - 소문자 알파벳, 숫자, 대시(-), 점(.) 문자만 허용
  - 시작 문자와 끝 문자는 알파벳 혹은 숫자만 가능
- 길이에 대한 내용은 RFC 1123 안에 정의
- 문자에 대한 상세한 제약 조건은 RFC 952 안에 정의

레이블 키 제약 조건 (3)

- 접두사를 생략하면 개인용 레이블로 간주
- 사용자 오브젝트에 레이블을 추가하는

  자동화된 시스템 구성 요소인 경우 접두사를 지정하는 것 필요

  - kube-scheduler, kube-controller-manager, kube-apiserver 등
  - 그 외 자동화 시스템

레이블 키 접두사 제약 조건 검사

staging/src/k8s.io/apimachinery/pkg/util/validation/validation.go

```
const dns1123LabelFmt string = "[a-z0-9]([-a-z0-9]*[a-z0-9])?"
const dns1123SubdomainFmt string = dns1123LabelFmt + "(\\." + dns1123LabelFmt + ")*"

// DNS1123SubdomainMaxLength is a subdomain's max length in DNS (RFC 1123)
const DNS1123SubdomainMaxLength int = 253

var dns1123SubdomainRegexp = regexp.MustCompile("^" + dns1123SubdomainFmt + "$")

// IsDNS1123Subdomain tests for a string that conforms to the definition of a
// subdomain in DNS (RFC 1123).
func IsDNS1123Subdomain(value string) []string {
    var errs []string
    if len(value) > DNS1123SubdomainMaxLength {
        errs = append(errs, MaxLenError(DNS1123SubdomainMaxLength))
    }
    if !dns1123SubdomainRegexp.MatchString(value) {
        errs = append(errs, RegexError(dns1123SubdomainErrorMsg, dns1123SubdomainFmt, "example.com"))
    }
    return errs
}
```

레이블 키 이름 부분 제약 조건 검사

staging/src/k8s.io/apimachinery/pkg/util/validation/validation.go

```
const qnameCharFmt string = "[A-Za-z0-9]"
const qnameExtCharFmt string = "[-A-Za-z0-9_.]"
const qualifiedNameFmt string = "(" + qnameCharFmt + qnameExtCharFmt + "*)?" + qnameCharFmt
const qualifiedNameMaxLength int = 63
var qualifiedNameRegexp = regexp.MustCompile("^" + qualifiedNameFmt + "$")

func IsQualifiedName(value string) []string {
    var errs []string
    parts := strings.Split(value, "/")
    var name string
    ...
    if len(name) == 0 {
        errs = append(errs, "name part "+EmptyError())
    } else if len(name) > qualifiedNameMaxLength {
        errs = append(errs, "name part "+MaxLenError(qualifiedNameMaxLength))
    }
    if !qualifiedNameRegexp.MatchString(name) {
        errs = append(errs, "name part "+RegexError(qualifiedNameErrMsg,
                                                    qualifiedNameFmt, "MyName", "my.name", "123-abc"))
    }
    return errs
}
```

레이블 값 제약 조건

- 최대 63자 이하 (공백 가능)
- 시작과 끝은 알파벳과 숫자만 가능
- 알파벳, 숫자, 대시(-), 밑줄(_), 점(.) 문자 사용 가능

레이블 값 제약 조건 검사

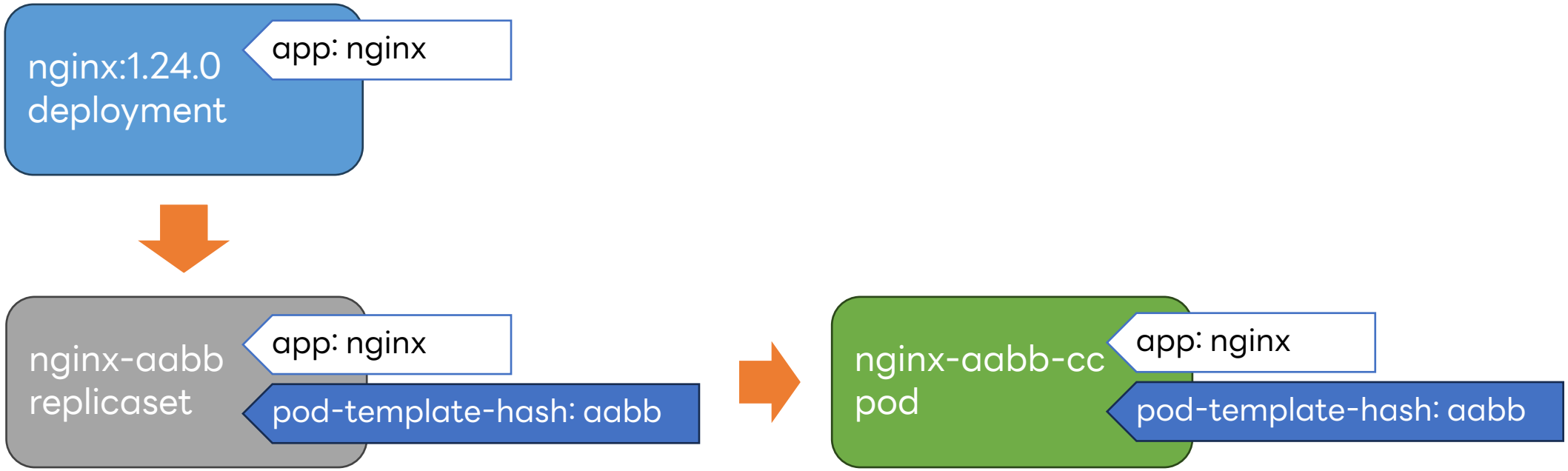
staging/src/k8s.io/apimachinery/pkg/util/validation/validation.go

```
const qnameCharFmt string = "[A-Za-z0-9]"
const qnameExtCharFmt string = "[-A-Za-z0-9_.]"
const qualifiedNameFmt string = "(" + qnameCharFmt + qnameExtCharFmt + "*)?" + qnameCharFmt
const labelValueFmt string = "(" + qualifiedNameFmt + ")?"
const LabelValueMaxLength int = 63

var labelValueRegexp = regexp.MustCompile("^" + labelValueFmt + "$")

func IsValidLabelValue(value string) []string {
    var errs []string
    if len(value) > LabelValueMaxLength {
        errs = append(errs, MaxLenError(LabelValueMaxLength))
    }
    if !labelValueRegexp.MatchString(value) {
        errs = append(errs, RegexError(labelValueErrMsg,
                                       labelValueFmt, "MyValue", "my_value", "12345"))
    }
    return errs
}
```

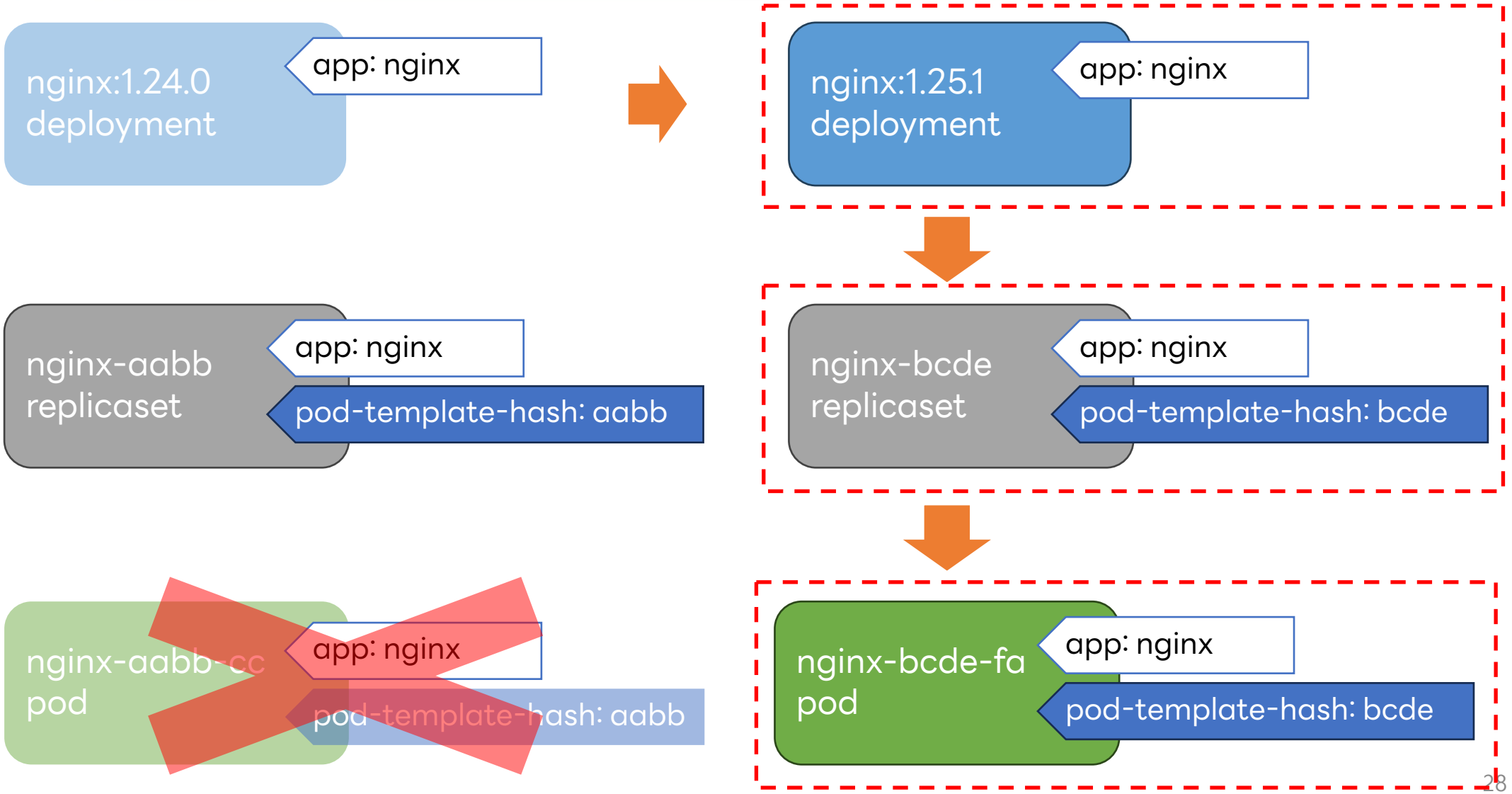
레이블과 디플로이먼트

레이블을 이용한 롤링 업데이트 (1)

- 디플로이먼트를 생성하면 템플릿 스펙을 이용해 해시 값 생성
- 해시 값을 레플리카셋 pod-template-hash 레이블에 주입



레이블을 이용한 롤링 업데이트 (2)



레이블을 이용한 롤링 업데이트 (3)

- 템플릿 스펙 변경은 새로운 레플리카셋을 해시 레이블과 생성
- 서비스에 영향이 없도록 새로운 레플리카셋에서 파드 확장
- 새로 배포되는 만큼 예전 레플리카셋에서 파드 축소
- 레플리카셋에 부착되어 있는 deployment.kubernetes.io/revision 어노테이션과 pod-template-hash 레이블을 이용해 리비전 관리

애플리케이션 배포 플랫폼 운영 사례

- 디플로이먼트 생성 규칙
  - {조직}-{애플리케이션} 형태로 디플로이먼트 생성
  - {조직}, {애플리케이션} 부분을 디플로이먼트의 레이블로 주입
- 디플로이먼트 이름 제약 조건
  - {조직}, {애플리케이션} 항목을 레이블 값으로 사용하기 때문에 길이 제한 필요
  - 각 부분은 최대 63자로 제한
  - 각 부분에 들어갈 수 있는 문자는 영어 소문자, 숫자, '-' 문자로 제한

레이블로 인한 디플로이먼트 생성 실패 (1)

- 일반적인 상황에서 아무런 문제 없이 애플리케이션 배포
- 사용자 요구사항을 구현하기 위해 istio 서비스 메시 설치
- 서비스 메시 구성 이후에 갑자기 디플로이먼트 생성 실패 에러
  - 레이블 제약 조건에 어긋난 값을 지정했다는 에러
  - {조직}, {애플리케이션} 부분에 들어간 값에는 문제가 없음

레이블로 인한 디플로이먼트 생성 실패 (2)

- 파드를 생성하며 istio-proxy 사이드카에서 주입한 레이블이 원인
  - service.istio.io/canonical-name 레이블에 워크로드 이름 주입 시도
  - {조직}, {애플리케이션} 각 부분은 63자 길이 제한
  - 두 부분을 합친 워크로드 이름은 63자를 초과할 수 있음
- 워크로드 이름을 63자로 제한하도록 구현
  - {조직}-{애플리케이션} 형태의 워크로드 마지막 문자가 대시(-) 인 경우 존재
- 63자를 넘을 경우 영문자로 끝나도록 변환 필요
  - 문자열의 마지막에 해시 문자열을 붙이는 형태

레이블로 인한 디플로이먼트 생성 실패 (3)

- 권장 레이블을 사용하는 형태로 문제 해결

istio/pkg/kube/inject/webhook.go

```
func extractCanonicalServiceLabel(podLabels map[string]string,
                                  workloadName string) string {
    if svc, ok := podLabels[model.IstioCanonicalServiceLabelName]; ok {
        return svc
    }

    if svc, ok := podLabels["app.kubernetes.io/name"]; ok {
        return svc
    }

    if svc, ok := podLabels["app"]; ok {
        return svc
    }

    return workloadName
}
```

레이블 필터링

레이블을 이용해 오브젝트 필터링

- kubectl get pod -l {key}={value} 형태로 리소스 검색 가능
- kube-apiserver 에서는 요청을 묶어 ListOptions 형태 변환
- 리소스 검색 조건과 ListOptions 데이터 결합
- etcd 저장소에서 조건에 만족하는 오브젝트 필터링
- 얻어온 오브젝트 목록을 클라이언트에 반환

레이블 및 다양한 조건을 이용해 검색 시작

staging/src/k8s.io/apiserver/pkg/registry/generic/registry/store.go

```
func (e *Store) List(ctx context.Context,
                     options *metainternalversion.ListOptions) (runtime.Object, error) {
    label := labels.Everything()
    if options != nil && options.LabelSelector != nil {
        label = options.LabelSelector
    }
    field := fields.Everything()
    if options != nil && options.FieldSelector != nil {
        field = options.FieldSelector
    }
    out, err := e.ListPredicate(ctx, e.PredicateFunc(label, field), options)
    if err != nil {
        return nil, err
    }
    if e.Decorator != nil {
        e.Decorator(out)
    }
    return out, nil
}
```

검색 조건을 이용해 저장소에 검색 요청

staging/src/k8s.io/apiserver/pkg/registry/generic/registry/store.go

```
func (e *Store) ListPredicate(ctx context.Context, p storage.SelectionPredicate,
                options *metainternalversion.ListOptions) (runtime.Object, error) {
    if options == nil {
        // By default we should serve the request from etcd.
        options = &metainternalversion.ListOptions{ResourceVersion: ""}
    }
    ...
    if name, ok := p.MatchesSingle(); ok {
        if key, err := e.KeyFunc(ctx, name); err == nil {
            storageOpts.Recursive = false
            err := e.Storage.GetList(ctx, key, storageOpts, list)
            return list, storeerr.InterpretListError(err, qualifiedResource)
        }
    }
  err := e.Storage.GetList(ctx, e.KeyRootFunc(ctx), storageOpts, list)
  return list, storeerr.InterpretListError(err, qualifiedResource)
}
```

검색 조건에 맞는 오브젝트 필터링 (1)

staging/src/k8s.io/apiserver/pkg/storage/etcd3/store.go

```
func appendListItem(v reflect.Value, data []byte, rev uint64,
        pred storage.SelectionPredicate, codec runtime.Codec, versioner
        storage.Versioner, newItemFunc func() runtime.Object) error {
    obj, _, err := codec.Decode(data, nil, newItemFunc())
    if err != nil {
        return err
    }

    if err := versioner.UpdateObject(obj, rev); err != nil {
        klog.Errorf("failed to update object version: %v", err)
    }
    if matched, err := pred.Matches(obj); err == nil && matched {
        v.Set(reflect.Append(v, reflect.ValueOf(obj).Elem()))
    }
    return nil
}
```

검색 조건에 맞는 오브젝트 필터링 (2)

staging/src/k8s.io/apiserver/pkg/storage/selection_predicate.go

```go
func (s *SelectionPredicate) Matches(obj runtime.Object) (bool, error) {
    if s.Empty() {
        return true, nil
    }
    labels, fields, err := s.GetAttrs(obj)
    if err != nil {
        return false, err
    }
    matched := s.Label.Matches(labels)
    if matched && s.Field != nil {
        matched = matched && s.Field.Matches(fields)
    }
    return matched, nil
}
```

검색 조건에 맞는 오브젝트 필터링 (3)

staging/src/k8s.io/apimachinery/pkg/labels/selector.go

```
func (r *Requirement) Matches(ls Labels) bool {
    switch r.operator {
    case selection.In, selection.Equals, selection.DoubleEquals:
        if !ls.Has(r.key) {
            return false
        }
        return r.hasValue(ls.Get(r.key))
    case selection.NotIn, selection.NotEquals:
        if !ls.Has(r.key) {
            return true
        }
        return !r.hasValue(ls.Get(r.key))
    case selection.Exists:
        return ls.Has(r.key)
    case selection.DoesNotExist:
        return !ls.Has(r.key)
        ...
```

결론

레이블 사용 방법 정리

- 레이블은 매우 유연하고 사용자가 원하는 데로 사용 가능
- 일관성을 유지하기 위해 명확한 의미를 가진 레이블 사용
- 쿠버네티스에서 권장하는 레이블은 사용하는 것을 추천
- 자동화 시스템에서 리소스에 레이블을 부착하는 경우 제약 조건에 어긋나지 않도록 명확한 규칙 정의 필요

레이블 제약 조건 (1)

- 레이블 키는 접두사와 이름 부분으로 나눠 제약 조건 적용
- 레이블 키 접두사
  - 최대 253자 길이
  - 알파벳, 숫자, 대시(-), 점(.) 문자 포함
  - 시작과 끝은 알파벳 혹은 숫자
- 레이블 키 이름 부분과 값
  - 최대 63자 길이
  - 알파벳, 숫자, 대시(-), 밑줄(_), 점(.) 문자 포함
  - 시작과 끝은 알파벳 혹은 숫자
  - 레이블 값은 공백 가능

레이블 제약 조건 (2)

- 레이블 키와 값 제약 조건을 통해 예측 가능한 레이블 생성
- 키와 값에 할당 가능한 문자를 제한해 쿠버네티스 안정성 확보
- 길이 제한을 통해 쿠버네티스에서 관리하는 데이터의 양 제어
- 길이와 문자 제한을 통해 쿠버네티스 성능과 효율성 향상

참고자료

- https://kubernetes.io/docs/concepts/overview/working-with-objects/names/
- https://kubernetes.io/docs/concepts/overview/working-with-objects/common-labels/
- https://github.com/kubernetes/kubernetes
- https://datatracker.ietf.org/doc/html/rfc1123
- https://datatracker.ietf.org/doc/html/rfc952
- https://istio.io/latest/docs/reference/config/labels/
- https://github.com/istio/istio

Q & A

감사합니다

임찬식
chanshik@gmail.com