kubectl 200% 활용하기

feat. krew

이병우

이병우

- 👩‍💻MUSINSA Site Reliability Engineer
- 🚀NLP Developer ➡️ DevOps ➡️ SRE
- 🐳K8S ☁️AWS
- 🏃Running

Disclaimer

이 발표가 도움이 되는 분

- Kubernetes에서 개발/운영을 시작하는 분
- Kubernetes 오픈소스 생태계에 흥미를 가지고 싶으신 분

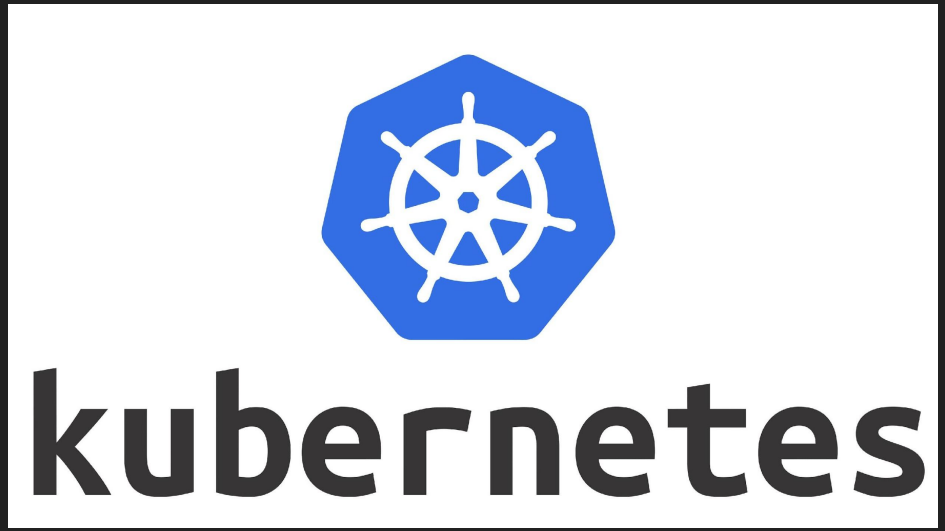
이 발표가 도움이 되지 않는 분

- Kubernetes 환경에 능숙하신 분
- CK* 자격증을 준비하거나 이미 보유하신 분

목차

- kubectl 소개
- kubectl 사용자를 위한 팁
- kubectl plugin 및 krew 소개
- 기본 추천 플러그인
- 개발자를 위한 플러그인
- 인프라 엔지니어를 위한 플러그인

Kubernetes 잘 쓰고 계신가요?



누구에게나 어려운 Kubernetes

Kubernetes를 잘 쓰려면 어떻게 해야할까요?

- 개발자와 인프라 엔지니어 모두에 해당
- 잘 쓰는 것 = 익숙해지는 것
- 좋은 도구를 찾아서 사용해보기!

kubectl 소개

kubectl

Kubernetes API를 사용하여 컨트롤 플레인과 통신하기 위한 커맨드라인 도구

```
# Creating objects
kubectl apply -f ./my-manifest.yaml # create resource(s)
kubectl create deployment nginx --image=nginx # create deployment

# Viewing and finding resources
kubectl get pods -o wide # List all pods with more details
kubectl get deployment my-dep # List a particular deployment

# Updating resources
kubectl set image deployment/frontend www=image:v2 # Rolling update

# Scaling resources
kubectl scale --replicas=3 rs/foo # Scale a replicaset

# Deleting resources
kubectl delete -f ./pod.json # Delete resource(s)
```

kubectl 왜 사용하나요?

- 튜토리얼에 나와서?...
- 일단 공식 기본 클라이언트 도구
- 필요성을 알기 위해서는 없는 경우를 생각해보아야 함
- kubectl을 사용할 수 없는 경우에는 kube-apiserver에 직접 요청을 보내야 함!

curl vs. kubectl

```
# list pods
curl --cacert ${CACERT} \
  -X GET \
  -H "Authorization: Bearer ${TOKEN}" \
  ${APISERVER}/namespaces/${namespace}/pods

# get pod
curl --cacert ${CACERT} \
  -X GET \
  -H "Authorization: Bearer ${TOKEN}" \
  ${APISERVER}/namespaces/${namespace}/pods/${pod}

# get pod logs
curl --cacert ${CACERT} \
  -X GET \
  -H "Authorization: Bearer $TOKEN" \
  ${APISERVER}/namespaces/${namespace}/pods/${pod}/log

# delete pod
curl --cacert ${CACERT} \
  -X DELETE \
  -H "Authorization: Bearer ${TOKEN}" \
  ${APISERVER}/namespaces/${namespace}/pods/${pod}
```

```
# list pods
kubectl get pods

# get pod
kubectl get pods ${pod}

# get pod logs
kubectl logs ${pod}

# delete pod
kubectl delete pod ${pod}
```

kubectl 꼭 사용해야 하나요?

- 웹/데스크톱 관리도구로 충분하지 않나요?
- 관리도구도 kubectl과 마찬가지로 kube-apiserver에 요청을 보내는 방식
- kubectl을 사용하면 Kubernetes 환경에서
  세밀한 작업을 수행할 수 있고 작동 방식을 이해하는 데 도움!

kubectl 사용자를 위한 팁

kubectl autocomplete & alias

- kubectl 사용시 피로도를 낮춰줄 기본 설정

```
# autocomplete
source <(kubectl completion bash)
echo "source <(kubectl completion bash)" >> ~/.bashrc

# alias
alias k=kubectl
complete -o default -F __start_kubectl k

# k as kubectl
k get pods
k apply -f ./pod.yaml
# ...
```

kubecolor (1)

- https://github.com/hidetatz/kubecolor
- 출력결과의 가독성을 높여줌

```
~/repos/dty1er/kubecolor $ kubecolor get pods
NAME                     READY   STATUS    RESTARTS   AGE
nginx-6799fc88d8-dnmv5   1/1     Running   0          2d4h
nginx-6799fc88d8-m8pbc   1/1     Running   0          2d4h
nginx-6799fc88d8-qdf9b   1/1     Running   0          2d4h
nginx-8spn9              1/1     Running   0          2d4h
nginx-dplns              1/1     Running   0          2d4h
nginx-lpv5x              1/1     Running   0          2d4h
```

kubecolor (2)

```
~/repos/dty1er/kubecolor $ kubecolor describe pods nginx-6799fc88d8-dnmv5
Name:           nginx-6799fc88d8-dnmv5
Namespace:      default
Priority:       0
Node:           minikube/172.17.0.3
Start Time:     Sat, 10 Oct 2020 14:07:14 +0900
Labels:         app=nginx
                pod-template-hash=6799fc88d8
Annotations:    <none>
Status:         Running
IP:             172.18.0.5
Controlled By:  ReplicaSet/nginx-6799fc88d8
Containers:
  nginx:
    Container ID:   docker://98483161b34785bc9feb594f44593ed3695f9e032e95f6f1faf16535aeba9046
    Image:          nginx
    Image ID:       docker-pullable://nginx@sha256:fc66cdef5ca33809823182c9c5d72ea86fd2cef7713cf3363e1a0b12a5d77500
    Port:           <none>
    Host Port:      <none>
    State:          Running
      Started:      Sat, 10 Oct 2020 14:07:35 +0900
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from default-token-4vw2q (ro)
Conditions:
  Type              Status
  Initialized       True
  Ready             True
  ContainersReady   True
  PodScheduled      True
Volumes:
  default-token-4vw2q:
    Type:        Secret (a volume populated by a Secret)
    SecretName:  default-token-4vw2q
    Optional:    false
QoS Class:       BestEffort
Node-Selectors:  <none>
Tolerations:     node.kubernetes.io/not-ready:NoExecute for 300s
                 node.kubernetes.io/unreachable:NoExecute for 300s
Events:          <none>
```

kube-ps1

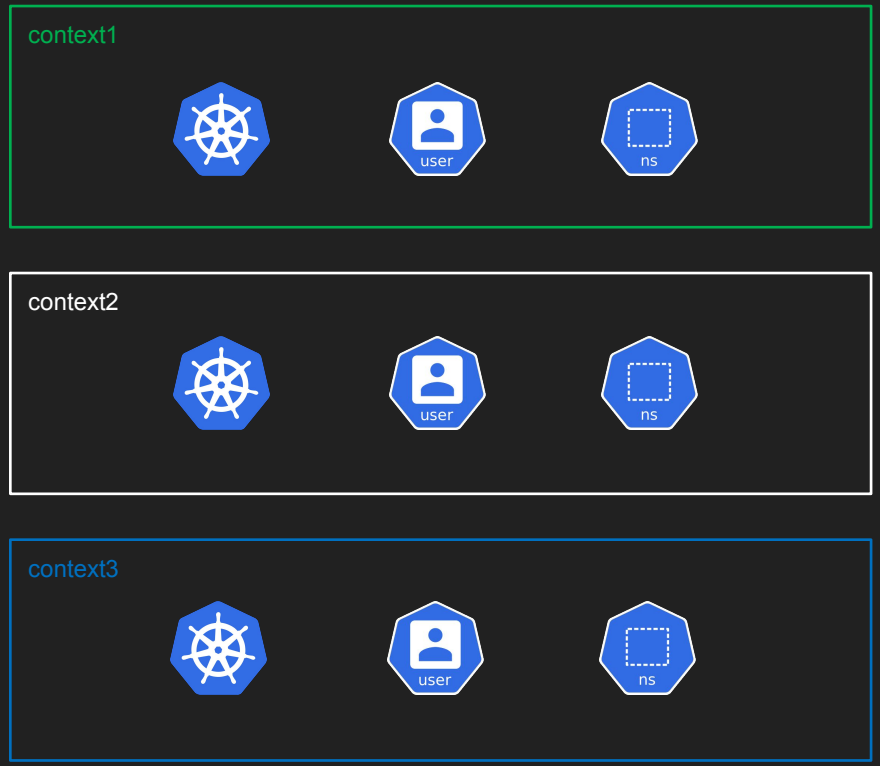
- https://github.com/jonmosco/kube-ps1
- 현재 context와 namespace를 prompt shell에 표시

```
# install
brew install kube-ps1

# .zshrc
PROMPT='$(kube_ps1)'$PROMPT
```

Kubernetes Context?

- user + cluster + namespace
- (user) 나는 누구이고
- (cluster) 무슨 클러스터를 바라보고 있고
- (namespace) 무슨 네임스페이스를 사용하는가
- kubeconfig 파일에 설정하여 사용



kubeconfig

- kubectl이 참조하는
  클러스터, 컨텍스트, 사용자 설정 파일

```
# 기본 ~/.kube/config
kubectl get pods

# 환경변수
export KUBECONFIG=/path/to/kube/config
kubectl get pods

# 명령어 인자
kubectl --kubeconfig=./kube/config get pods
```

```
# ~/.kube/config
apiVersion: v1
kind: Config
current-context: ${current_context}
clusters:
- cluster:
    certificate-authority-data: ${cluster_1_ca}
    server: ${cluster_1_server}
  # ...
users:
- name: ${user_1}
  user:
  # ...
contexts:
- name: ${context_1}
  context:
    cluster: ${cluster_1}
    user: ${user_1}
    namespace: ${namespace_1}
# ...
```

context & namespace switch

- 컨텍스트와 네임스페이스 변경

```
# 현재 컨텍스트 확인
kubectl config current-context
# 조회
kubectl config get-contexts
# 변경
kubectl use-context ${변경할 컨텍스트}

# 현재 네임스페이스 확인
kubectl config view --minify -o jsonpath='{..namespace}'
# 조회
kubectl get namespaces
# 변경
kubectl config set-context --current --namespace=${변경할 네임스페이스}
```

kubectl plugin & krew 소개

kubectl plugins

- https://kubernetes.io/docs/tasks/extend-kubectl/kubectl-plugins/
- 기본 커맨드 외에 추가하는 커스텀 커맨드
- 직접 작성 가능
- ex) kubectl ${플러그인 커맨드}

```
# foo 플러그인 작성
> cat ./kubectl-foo
if [[ "$1" == "version" ]]
then
    echo "1.0.0"
    exit 0
fi
if [[ "$1" == "config" ]]
then
    echo "$KUBECONFIG"
    exit 0
fi
echo "I am a plugin named kubectl-foo"

# 실행 설정
> sudo chmod +x ./kubectl-foo
> sudo mv ./kubectl-foo /usr/local/bin

# 사용
> kubectl foo
I am a plugin named kubectl-foo
> kubectl foo version
1.0.0
> KUBECONFIG=/etc/kube/config kubectl foo config
/etc/kube/config
```

krew

- https://krew.sigs.k8s.io/
- kubectl 플러그인 매니저

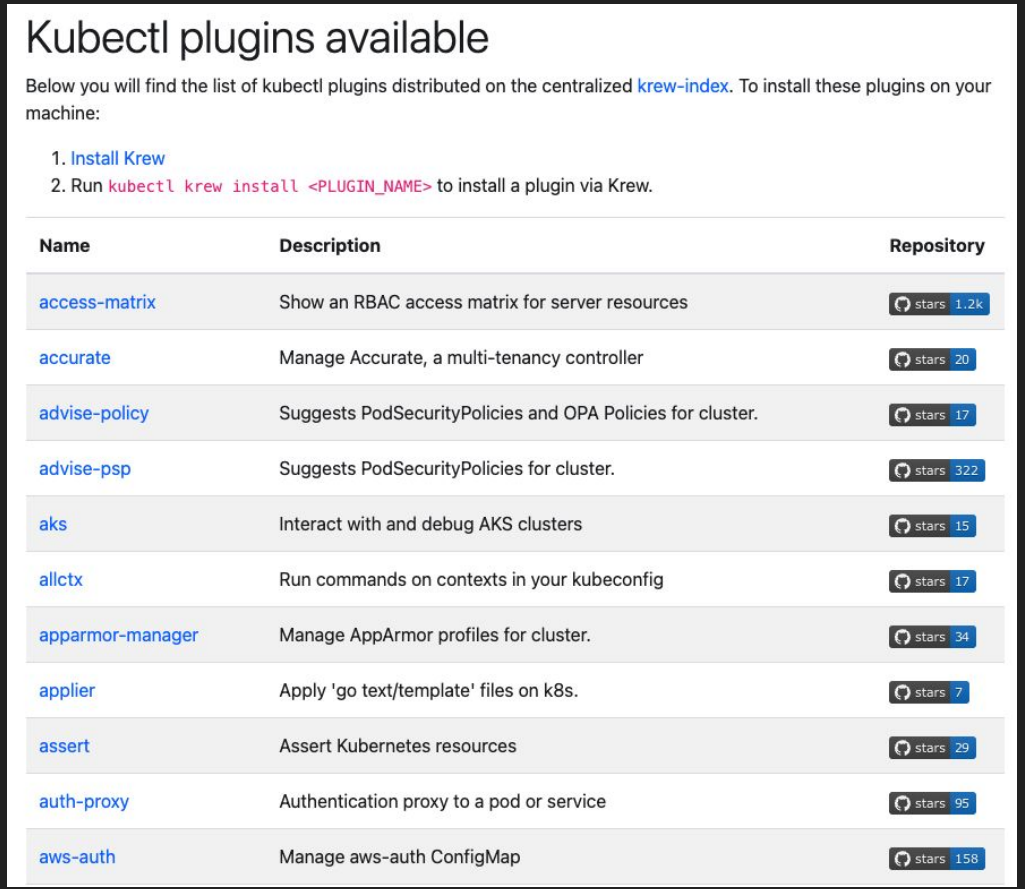

```
# 플러그인 검색
> kubectl krew search access
NAME                            DESCRIPTION                                       INSTALLED
access-matrix                   Show an RBAC access matrix for server resources   no
# ...

# 플러그인 설치
> kubectl krew install access-matrix

# 플러그인 확인
> kubectl krew list
access-matrix
# ...
```

krew kubectl plugins

- https://krew.sigs.k8s.io/plugins/
- 200개 이상 플러그인
- Github Star 개수 참고



Kubectl plugins available

Below you will find the list of kubectl plugins distributed on the centralized krew-index. To install these plugins on your machine:

1. Install Krew
2. Run `kubectl krew install <PLUGIN_NAME>` to install a plugin via Krew.

Name	Description	Repository
access-matrix	Show an RBAC access matrix for server resources	stars 1.2k
accurate	Manage Accurate, a multi-tenancy controller	stars 20
advise-policy	Suggests PodSecurityPolicies and OPA Policies for cluster.	stars 17
advise-psp	Suggests PodSecurityPolicies for cluster.	stars 322
aks	Interact with and debug AKS clusters	stars 15
allctx	Run commands on contexts in your kubeconfig	stars 17
apparmor-manager	Manage AppArmor profiles for cluster.	stars 34
applier	Apply 'go text/template' files on k8s.	stars 7
assert	Assert Kubernetes resources	stars 29
auth-proxy	Authentication proxy to a pod or service	stars 95
aws-auth	Manage aws-auth ConfigMap	stars 158

기본 추천 플러그인

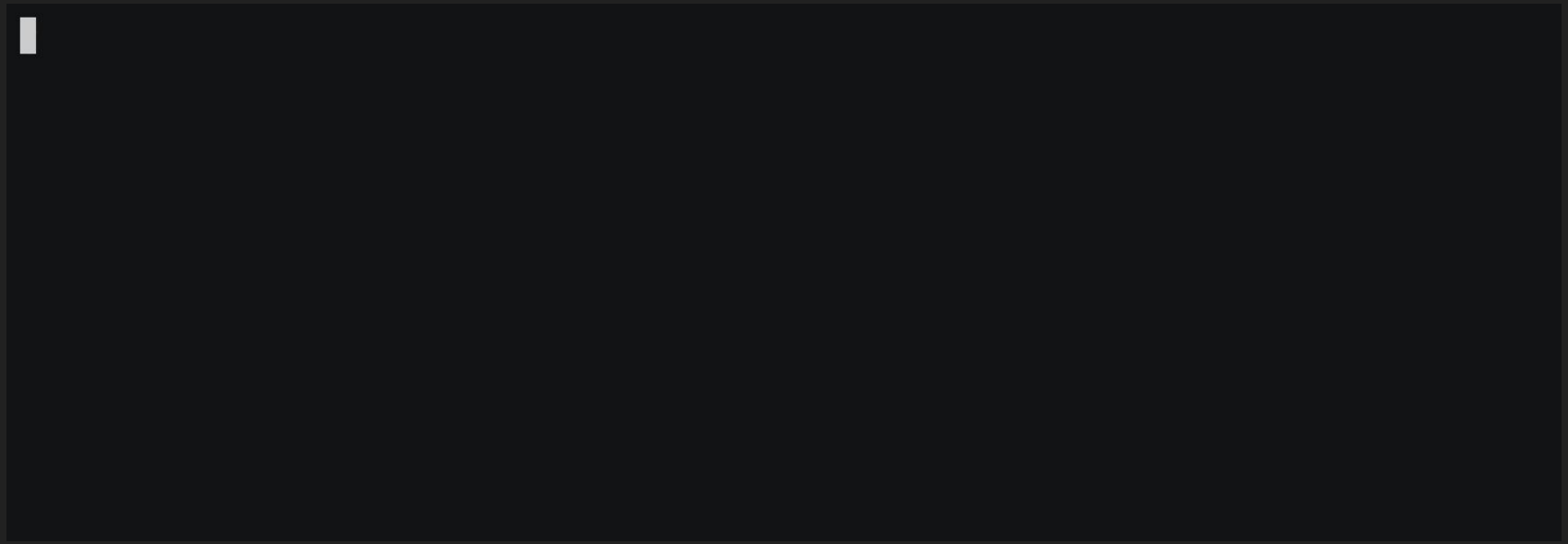
kubectx & kubens

- https://github.com/ahmetb/kubectx/
- 간단하게 컨텍스트 및 네임스페이스 변경

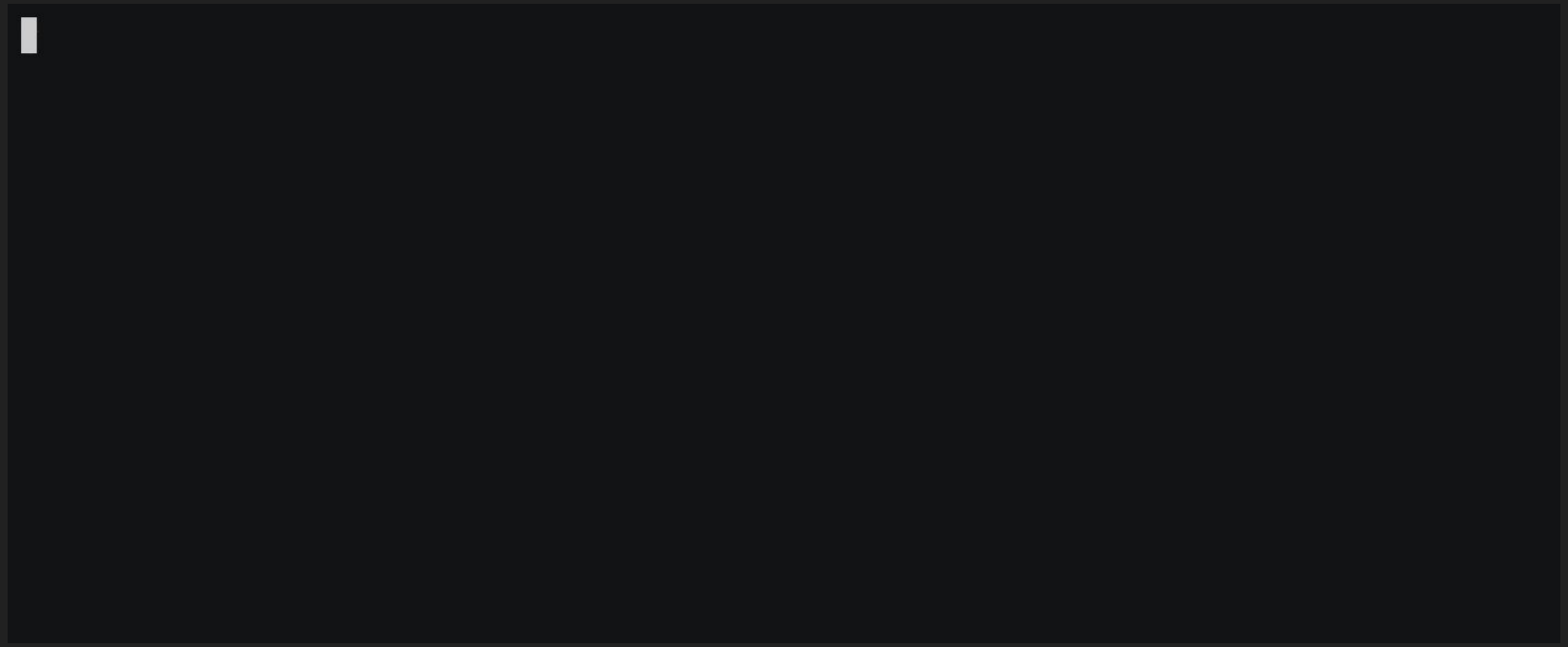
```
$ kubectl ctx
> context_1
context_2
# ...
Switched to context "context_1"

# 네임스페이스 조회 및 변경
$ kubectl ns
namespace_1
> namespace_2
# ...
Context "context_1" modified.
Active namespace is "namespace_1"
```

kubectx



kubens



neat

- Kubernetes가 자동으로 추가하는 필드 제거
- 리소스 import시 유용

```
kubectl get pod myapp -o yaml | kubectl neat > myapp.yaml
```

```
apiVersion: v1
kind: Pod
metadata:
- creationTimestamp: "2019-04-24T19:55:27Z"
  labels:
    name: myapp
  name: myapp
  namespace: default
- resourceVersion: "274103"
- selfLink: /api/v1/namespaces/default/pods/myapp
- uid: e8330f3c-66ca-11e9-b6fa-0800271788ca
spec:
  containers:
  - image: nginx
-   imagePullPolicy: Always
    name: myapp
    ports:
    - containerPort: 1234
-     protocol: TCP
-   resources: {}
-   terminationMessagePath: /dev/termination-log
-   terminationMessagePolicy: File
-   volumeMounts:
-   - mountPath: /var/run/secrets/kubernetes.io/serviceaccount
-     name: default-token-nmshj
-     readOnly: true
- dnsPolicy: ClusterFirst
- enableServiceLinks: true
- nodeName: minikube
  priority: 0
- restartPolicy: Always
- schedulerName: default-scheduler
- securityContext: {}
- serviceAccount: default
  serviceAccountName: default
- terminationGracePeriodSeconds: 30
```

개발자를 위한 플러그인

개발자의 관심사

- 내 서비스의 리소스는 어떻게 확인할 수 있는가?
- 로그를 어떻게 확인할 수 있는가?
- 배포해도 되는가?

리소스 – lineage

- https://github.com/tohjustin/kube-lineage
- 리소스의 상태와 의존관계를 보여줌
- Kubernetes 리소스 의존관계를 학습하기 좋음

```
> kubectl lineage deployment search-log-api
NAME                                                READY   STATUS          AGE
Deployment/search-log-api                           1/1                     2y48d
├── PodDisruptionBudget/search-log-api-default-pdb  -       SufficientPods  237d
├── ReplicaSet/search-log-api-549cf57f7f            0/0                     381d
├── ReplicaSet/search-log-api-6ccc6f8cfc            0/0                     148d
├── ReplicaSet/search-log-api-7566b7564             0/0                     157d
├── ReplicaSet/search-log-api-79986c456             1/1                     134d
│   └── Pod/search-log-api-79986c456-grtbw          3/3     Running         10d
│       ├── PodDisruptionBudget/search-log-api-default-pdb  -       SufficientPods  237d
│       └── Service/search-log-api                  -                       2y48d
│           └── EndpointSlice/search-log-api-ndt6m  -                       2y48d
├── ReplicaSet/search-log-api-7db788b746            0/0                     156d
└── ReplicaSet/search-log-api-85c69b9954            0/0                     382d
```

로그 – tail (1)

- https://github.com/boz/kail
- 필터 방식으로 로그 출력
- 리소스를 지정하는 logs보다 간편함

로그 – tail (2)

배포 – confirm

- https://github.com/brianpursley/kubectl-confirm
- 변경점을 확인한 후에 배포하고 싶을 때
- kubectl diff와 kubectl apply --dry-run을 미리 실행하고 결과를 보여줌

```
> kubectl confirm apply -f ~/changed.yaml
========== Config ===========
Context:    kind-kind
Cluster:    kind-kind
User:       kind-kind
Namespace:  default

========== Dry Run ==========
deployment.apps/foo configured (server dry run)

========== Diff =============
diff -u -N /tmp/LIVE-
2275701238/apps.v1.Deployment.default.foo /tmp/MERGED-
1055657464/apps.v1.Deployment.default.foo
--- /tmp/LIVE-2275701238/apps.v1.Deployment.default.foo
 2022-07-28 10:27:25.690604172 -0400
+++ /tmp/MERGED-
1055657464/apps.v1.Deployment.default.foo   2022-07-28
10:27:25.694604038 -0400
# ...
-  generation: 1
+  generation: 2
# ...
-  revisionHistoryLimit: 10
+  revisionHistoryLimit: 11
# ...

========== Confirm ==========
The following command will be executed:
kubectl apply -f /home/bpursley/changed.yaml
Enter 'yes' to continue:
```

인프라 엔지니어를 위한 플러그인

인프라 엔지니어의 관심사

- 문제가 있는 리소스는 어떻게 점검할 수 있을까?
- 어떻게 클러스터를 업그레이드할 수 있을까?
- 어떻게 워크로드의 안정성을 테스트할 수 있을까?

점검 (1) – sick-pods

- https://github.com/alecjacobs5401/kubectl-sick-pods
- Not Ready 상태의 pod를 보여줌

```
> kubectl sick-pods
'bad-pod-764ccf854d-kbsq2' is not ready! Reason Provided: None
    Failed Pod Conditions:
        CONDITION       REASON      MESSAGE
        Ready           ContainersNotReady  containers with unready status: [bad-container]
        ContainersReady     ContainersNotReady  containers with unready status: [bad-container]

    Pod Events:
        LAST SEEN           TYPE    REASON          MESSAGE
        2020-05-29 14:15:32 -0700 PDT   Warning DNSConfigForming    Search Line limits were
exceeded, some search paths have been omitted, the applied search line is: some-
namespace.svc.cluster.local svc.cluster.local cluster.local ec2.internal cluster-dns dns

    Container 'bad-container' is not ready!
        Container Logs:
            -e:1:in `<main>': ah (RuntimeError)
```

점검 (2) – oomd

- https://github.com/jdockerty/kubectl-oomd
- 최근에 OOM 발생한 pod를 보여줌

```
> kubectl oomd
POD                         CONTAINER       REQUEST     LIMIT     TERMINATION TIME
my-app-5bcbcdf97-722jp      infoapp         1G          8G        2022-11-07 13:03:49 +0000 GMT
my-app-5bcbcdf97-7j5rd      infoapp         1G          8G        2022-11-07 14:35:34 +0000 GMT
my-app-5bcbcdf97-k8g8g      infoapp         1G          8G        2022-11-07 14:35:02 +0000 GMT
my-app-5bcbcdf97-mf65j      infoapp         1G          8G        2022-11-07 14:34:57 +0000 GMT
```

업그레이드 (1) – kubent

- https://github.com/doitintl/kube-no-trouble
- 업그레이드 전 API deprecation check

```
> kubent -t 1.25
2:29PM INF >>> Kube No Trouble `kubent` <<<
2:29PM INF version 0.7.0 (git sha d1bb4e5fd6550b533b2013671aa8419d923ee042)
2:29PM INF Initializing collectors and retrieving data
2:29PM INF Target K8s version is 1.25.0
2:29PM INF Retrieved 1315 resources from collector name=Cluster
2:29PM INF Retrieved 120 resources from collector name="Helm v3"
2:29PM INF Loaded ruleset name=custom.rego.tmpl
2:29PM INF Loaded ruleset name=deprecated-1-16.rego
2:29PM INF Loaded ruleset name=deprecated-1-22.rego
2:29PM INF Loaded ruleset name=deprecated-1-25.rego
2:29PM INF Loaded ruleset name=deprecated-1-26.rego
2:29PM INF Loaded ruleset name=deprecated-future.rego

----------------------------------------------------------------------------------------
>>> Deprecated APIs removed in 1.25 <<<
----------------------------------------------------------------------------------------
KIND                  NAMESPACE     NAME                        API_VERSION      REPLACE_WITH (SINCE)
CronJob               k6            test-service-cronjob        batch/v1beta1    batch/v1 (1.21.0)
PodDisruptionBudget   kube-system   ebs-csi-controller          policy/v1beta1   policy/v1 (1.21.0)
# ...
```

Kubent

업그레이드 (2) – deprecations

- https://github.com/rikatz/kubepug
- 업그레이드 전 API deprecation check

```
> kubectl depcrecations --k8s-version=v1.18.6
RESULTS:
Deprecated APIs:

Ingress found in extensions/v1beta1
        ├─ Ingress is a collection of rules that allow inbound connections to reach the
endpoints defined by a backend. An Ingress can be configured to give services externally-
reachable urls, load balance traffic, terminate SSL, offer name based virtual hosting etc.
DEPRECATED - This group version of Ingress is deprecated by networking.k8s.io/v1beta1
Ingress. See the release notes for more information.
                -> OBJECT: nginxnok namespace: default
                -> OBJECT: nginxok namespace: default


Deleted APIs:

DaemonSet found in extensions/v1beta1
        ├─ API REMOVED FROM THE CURRENT VERSION AND SHOULD BE MIGRATED IMMEDIATELY!!
                -> OBJECT: kindnet namespace: kube-system
                -> OBJECT: kube-proxy namespace: kube-system

Deployment found in extensions/v1beta1
        ├─ API REMOVED FROM THE CURRENT VERSION AND SHOULD BE MIGRATED IMMEDIATELY!!
                -> OBJECT: coredns namespace: kube-system
                -> OBJECT: local-path-provisioner namespace: local-path-storage

ReplicaSet found in extensions/v1beta1
        ├─ API REMOVED FROM THE CURRENT VERSION AND SHOULD BE MIGRATED IMMEDIATELY!!
                -> OBJECT: coredns-6dcc67dcbc namespace: kube-system
                -> OBJECT: local-path-provisioner-56fcf95c58 namespace: local-path-storage
```

kubent vs. deprecations

- 무엇을 사용해야 하나요?
- 둘 다 사용하는 것을 추천
- 두 deprecation 결과물 모두에 대응하는 것이 좋음

업그레이드 (3) – convert (1)

- https://kubernetes.io/ko/docs/tasks/tools/included/kubectl-convert-overview/
- deprecated API 버전 매니페스트 파일을 stable API 형식으로 변경

업그레이드 (3) – convert (2)

```
kubectl convert -f old.ingress.yaml > new.ingress.yaml
```

```yaml
# old.ingress.yaml
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: beta-ingress
  annotations:
    kubernetes.io/ingress.class: alb
    alb.ingress.kubernetes.io/scheme: internal
    alb.ingress.kubernetes.io/target-type: ip
spec:
  rules:
    - http:
        paths:
          - backend:
              serviceName: example
              servicePort: 8080
            path: /*
```

```yaml
# new.ingress.yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  annotations:
    alb.ingress.kubernetes.io/scheme: internal
    alb.ingress.kubernetes.io/target-type: ip
    kubernetes.io/ingress.class: alb
  name: beta-ingress
spec:
  rules:
  - http:
      paths:
      - backend:
          service:
            name: example
            port:
              number: 8080
        path: /*
        pathType: ImplementationSpecific
```

- https://github.com/honk-ci/kubectl-snap
- 절반의 pod를 삭제함
- 실행시 주의! 운영환경 실행 금지! 🚨

```
# 모든 파드의 절반
kubectl snap -a

# 특정 네임스페이스에서 절반
kubectl snap -n kube-system
```

https://alex-moss.medium.com/down-with-the-krew-my-favourite-kubectl-plugins-279d8d2d5640

기타 유용한 플러그인

- access-matrix - RBAC을 테이블로 정리해서 보여 줌
- rbac-lookup - 사용자에 연결된 Role을 찾아줌
- resource-capacity – 클러스터 리소스 사용현황을 정리해서 보여줌
- view-allocations – CPU, Memory, GPU 사용현황을 정리해서 보여줌
- pod-dive – 노드 관점에서 pod 상세 정보 조회

Summary

- kubectl 플러그인은 Kubernetes에 익숙해질 수 있는 좋은 도구🧰
- 그러나 플러그인이 절대 만능은 아님⚠️
- 플러그인으로 가능한 것은 kubectl으로 가능한 작업 💡
- 궁금하다면 플러그인 Github에서 소스 코드를 확인해보자😄

References

- Kubernetes
- Krew

Q & A