

Kubernetes_Pod

- 실습가이드

최초 작성일 : 2024/02/28

최종 제출일 : 2024/02/29

김민경

내용

I. 실습	2
1. Master에서의 실습.....	2
2. 워커노드에서의 실습.....	5
3. Master에서의 실습.....	6

I. 실습

1. Master에서의 실습

1) Calico의 최신 calicoctl 바이너리를 다운로드하고 로컬 시스템에 저장하기

```
curl -L https://github.com/projectcalico/calico/releases/latest/download/calicoctl-linux-amd64 -o kubectl-calico
```

- curl : URL을 통해 데이터를 전송하는 명령줄 도구
- -L : 지정된 URL로 리다이렉트를 따라감
- -o kubectl-calico : 다운로드된 파일의 이름을 kubectl-calico로 지정

2) 칼리코(CNI 플러그인) 노드 상태 확인

kubectl-calico node status // Calico 네트워크에서 노드의 상태를 조회하기

```
(calico:N/A) root@k8s-m:~# kubectl-calico node status
Calico process is running.

IPv4 BGP status
+-----+-----+-----+-----+-----+
| PEER ADDRESS | PEER TYPE | STATE | SINCE | INFO |
+-----+-----+-----+-----+-----+
| 192.168.56.201 | node-to-node mesh | up | 05:56:32 | Established |
| 192.168.56.202 | node-to-node mesh | up | 05:56:32 | Established |
+-----+-----+-----+-----+-----+

IPv6 BGP status
No IPv6 peers found.
```

kubectl-calico ipam show // Calico 네트워크에서 IP 주소 관리 정보 조회

```
(calico:N/A) root@k8s-m:~# kubectl-calico ipam show
+-----+-----+-----+-----+-----+
| GROUPING | CIDR | IPS TOTAL | IPS IN USE | IPS FREE |
+-----+-----+-----+-----+-----+
| IP Pool | 172.30.0.0/16 | 65536 | 12 (0%) | 65524 (100%) |
+-----+-----+-----+-----+-----+
(calico:N/A) root@k8s-m:~#
```

- 기존 버전과 같으면 version-mismatch라는 명령어가 안보임
- ip pool 대역대를 볼 수 있음

3) 마스터 노드 컴포넌트 상세 정보 확인

```
kubectl describe pod -n kube-system kube-apiserver-k8s-m
```

```
(calico:N/A) root@k8s-m:~# kubectl describe pod -n kube-system kube-apiserver-k8s-m
Name: kube-apiserver-k8s-m
Namespace: kube-system
Priority: 2000001000
Priority Class Name: system-node-critical
Node: k8s-m/192.168.56.200
Start Time: Wed, 28 Feb 2024 11:33:44 +0900
Labels: component=kube-apiserver
tier=control-plane
Annotations: kubeadm.kubernetes.io/kube-apiserver.advertise-address.endpoint: 192.168.56.200:6443
kubernetes.io/config.hash: effeca44f7758314421e5c76544b4426
kubernetes.io/config.mirror: effeca44f7758314421e5c76544b4426
kubernetes.io/config.seen: 2024-02-28T11:33:43.017033502+09:00
kubernetes.io/config.source: file
Status: Running
SeccompProfile: RuntimeDefault
IP: 192.168.56.200
IPs:
  IP: 192.168.56.200
Controlled By: Node/k8s-m
Containers:
  kube-apiserver:
    Container ID: containerd://69e680071eb61f6fbf8c353541eb2169f0767c3fa64d1d4662e29aae4a7ee97d
```

- kube-system 네임스페이스에서 실행 중인 kube-apiserver-k8s-m라는 이름의 Pod에 대한 세부 정보를 조회
- kubectl describe pod : pod의 자세한 정보조회
- -n kube-system : pod가 속한 네임스페이스 지정
- 'kube-system' 네임스페이스에 있는 파드 조회
- kube-apiserver-k8s-m : 매니저 노드에서 실행중인 쿠버네티스 API

```
kubectl describe pod -n kube-system etcd-k8s-m
```

- etcd-k8s-m : 매니저 노드에서 실행중인 etcd 파드

```
kubectl describe node k8s-m //노드정보 확인하기
```

```
(calico:N/A) root@k8s-m:~# kubectl describe pod -n kube-system etcd-k8s-m
Name: etcd-k8s-m
Namespace: kube-system
Priority: 2000001000
Priority Class Name: system-node-critical
Node: k8s-m/192.168.56.200
Start Time: Wed, 28 Feb 2024 11:33:44 +0900
Labels: component=etcd
tier=control-plane
Annotations: kubeadm.kubernetes.io/etcd.advertise-client-urls: https://192.168.56.200:2379
kubernetes.io/config.hash: 209feb4215304f08b16c5d6245ef93a
kubernetes.io/config.mirror: 209feb4215304f08b16c5d6245ef93a
kubernetes.io/config.seen: 2024-02-28T11:33:43.017024354+09:00
kubernetes.io/config.source: file
Status: Running
SeccompProfile: RuntimeDefault
IP: 192.168.56.200
IPs:
  IP: 192.168.56.200
Controlled By: Node/k8s-m
Containers:
  etcd:
    Container ID: containerd://e02afb0c8a823d928c0356ccfbf90faa90fcbad3fee31e8413e15be2e3eae8a
    Image: registry.k8s.io/etcd:3.5.10-0
    Image ID: registry.k8s.io/etcd@sha256:22f892d7672adc0b9c86df67792afdb8b2dc08880f49f669eaaa59c47d7908c2
    Port: <none>
```

kubectl describe node k8s-w1

```

Kube Proxy Version: v1.27.11
PodCIDR: 172.30.1.0/24
PodCIDRs: 172.30.1.0/24
Non-terminated Pods: (5 in total)
  Namespace      Name      CPU Requests  CPU Limits  Memory Requests  Memory Limits
  ----
  calico-system   calico-node-5z4cj      0 (0%)      0 (0%)      0 (0%)      0 (0%)
  calico-system   csi-node-driver-89h6g   0 (0%)      0 (0%)      0 (0%)      0 (0%)
  kube-system     kube-proxy-lxxg5        0 (0%)      0 (0%)      0 (0%)      0 (0%)
  
```

pstree

```

(calico:N/A) root@k8s-m:~# pstree
systemd--ModemManager--2*[{ModemManager}]
          --VBoxService--8*[{VBoxService}]
          --accounts-daemon--2*[{accounts-daemon}]
          --2*[{agetty}]
          --atd
          --containerd--22*[{containerd}]
            --containerd-shim--kubernetes--8*[{kubernetes}]
              --pause
              --11*[{containerd-shim}]
            --containerd-shim--kubernetes--5*[{kubernetes}]
              --pause
              --11*[{containerd-shim}]
            --containerd-shim--etcd--9*[{etcd}]
              --pause
              --11*[{containerd-shim}]
            --containerd-shim--kubernetes--10*[{kubernetes}]
              --pause
              --12*[{containerd-shim}]
            --containerd-shim--kubernetes--5*[{kubernetes}]
              --pause
              --11*[{containerd-shim}]
            --containerd-shim--operator--8*[{operator}]
              --pause
              --11*[{containerd-shim}]
            --containerd-shim--pause
              --tini--calico-typha--10*[{calico-typha}]
              --10*[{containerd-shim}]
            --containerd-shim--pause
              --runsvdir--runsv--bird
                  --run--run--svlogd
                  --tee
              --3*[{runsv--calico-node--6*[{calico-node}]
                  --run--run--svlogd]
                  --tee]
              --runsv--bird6
                  --run--run--svlogd
  
```

- 누구와 연결고리 있는지 한번에 확인할 수 있음

ps axf |grep /usr/bin/containerd //이미지 확인

```

(calico:N/A) root@k8s-m:~# ps axf |grep /usr/bin/containerd
30145 pts/0 S+ 0:00 \_ grep /usr/bin/containerd
4609 ? Ssl 3:54 /usr/bin/containerd
5774 ? SL 0:34 /usr/bin/containerd-shim-runc-v2 -namespace k8s.io -id 043c167696a1e4419492659c1812657fb9264dc646bd04209dce572b101f7ef0 -address /run/containerd.sock
d/containerd.sock
5786 ? SL 0:34 /usr/bin/containerd-shim-runc-v2 -namespace k8s.io -id 31f29c350d16e2d7605a755300c3194611a01a860cde36b432af341830b58d64 -address /run/containerd.sock
5801 ? SL 0:22 /usr/bin/containerd-shim-runc-v2 -namespace k8s.io -id 2938055384b4a301a0558a3da7bf1a46f6807109371a44d0abce769b300700a4 -address /run/containerd.sock
5810 ? SL 0:15 /usr/bin/containerd-shim-runc-v2 -namespace k8s.io -id aecc811e41448fc7764db6ace2b3addce181426fef288e9d78aff3d23f44861e -address /run/containerd.sock
6202 ? SL 0:40 /usr/bin/containerd-shim-runc-v2 -namespace k8s.io -id b306ae6a2d0451880ea8b67f97b53d323a57d305effeba22b7b390f80c7ef5bd -address /run/containerd.sock
6235 ? SL 0:38 /usr/bin/containerd-shim-runc-v2 -namespace k8s.io -id 2f9afd9a2a70db28f9de1115b0478fdb4516b3089a8411fa20726bb68975bc6f -address /run/containerd.sock
7231 ? SL 0:27 /usr/bin/containerd-shim-runc-v2 -namespace k8s.io -id 2fbb1a36ccc068a34303dff61de3f00138b1ecfc6d61d18f4de9953ef924bc55 -address /run/containerd.sock
  
```

- 한 줄 한 줄이 이미지임

4) kube-controller-manager-k8s-m 파드 상세 및 로그 정보 확인

```
kubectl describe pod -n kube-system kube-controller-manager-k8s-m
```

// kube-system 네임스페이스에서 실행 중인 kube-controller-manager-k8s-m 라는 이름의 Pod 에 대한 세부 정보를 조회

참고)

- 스케줄러는 노드 배치만 함
- 생성, 삭제, cpu, 메모리 부족 등등의 내용은 controller 가 담당

```
kubectl logs -n kube-system kube-controller-manager-k8s-m
```

//가지고 있는 로그 에러 났을 때 사용하는 명령어

//kube-system 네임스페이스에서 실행 중인 kube-controller-manager-k8s-m라는 이름의 Pod의 로그를 조회

5) kube-scheduler-k8s-m 파드 상세 정보 및 로그 확인

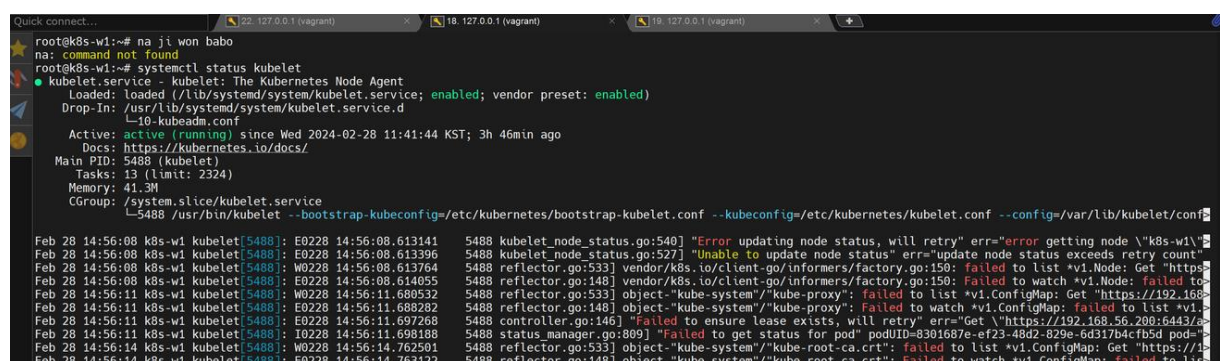
```
kubectl describe pod -n kube-system kube-scheduler-k8s-m
```

```
kubectl logs -n kube-system kube-scheduler-k8s-m
```

2. 워커노드에서의 실습

1) 워커노드에서 정보 확인하기

```
systemctl status kubelet
```



```
Quick connect...
root@k8s-w1:~# na ji won babo
na: command not found
root@k8s-w1:~# systemctl status kubelet
● kubelet.service - kubelet: The Kubernetes Node Agent
   Loaded: loaded (/lib/systemd/system/kubelet.service; enabled; vendor preset: enabled)
   Drop-In: /usr/lib/systemd/system/kubelet.service.d
            └─10-kubeadm.conf
   Active: active (running) since Wed 2024-02-28 11:41:44 KST; 3h 46min ago
     Docs: https://kubernetes.io/docs/
   Main PID: 5488 (kubelet)
    Tasks: 13 (limit: 2324)
   Memory: 41.3M
   CGroup: /system.slice/kubelet.service
           └─5488 /usr/bin/kubelet --bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf --kubeconfig=/etc/kubernetes/kubelet.conf --config=/var/lib/kubelet/confi

Feb 28 14:56:08 k8s-w1 kubelet[5488]: E0228 14:56:08.613141 5488 kubelet_node_status.go:540] "Error updating node status, will retry" err="error getting node \"k8s-w1\"
Feb 28 14:56:08 k8s-w1 kubelet[5488]: E0228 14:56:08.613396 5488 kubelet_node_status.go:527] "Unable to update node status" err="update node status exceeds retry count"
Feb 28 14:56:08 k8s-w1 kubelet[5488]: W0228 14:56:08.613764 5488 reflector.go:533] vendor/k8s.io/client-go/informers/factory.go:150: failed to list *v1.Node: Get "https://192.168.56.200:6443/api/v1/nodes" dial tcp 192.168.56.200:6443: connect: connection refused
Feb 28 14:56:11 k8s-w1 kubelet[5488]: E0228 14:56:11.680532 5488 reflector.go:148] object "kube-system"/"kube-proxy": failed to watch *v1.ConfigMap: Get "https://192.168.56.200:6443/api/v1/configmaps/kube-proxy?watch=true": dial tcp 192.168.56.200:6443: connect: connection refused
Feb 28 14:56:11 k8s-w1 kubelet[5488]: E0228 14:56:11.687268 5488 controller.go:146] "Failed to ensure lease exists, will retry" err="Get \"https://192.168.56.200:6443/api/v1/leases/kube-system/lease?watch=true\": dial tcp 192.168.56.200:6443: connect: connection refused"
Feb 28 14:56:11 k8s-w1 kubelet[5488]: T0228 14:56:11.698188 5488 status_manager.go:809] "Failed to get status for pod" podUID=8301687e-ef23-48d2-829e-6d317b4cfb5d pod="kube-system/kube-proxy"
Feb 28 14:56:14 k8s-w1 kubelet[5488]: W0228 14:56:14.762501 5488 reflector.go:533] object "kube-system"/"kube-root-ca.crt": failed to list *v1.ConfigMap: Get "https://192.168.56.200:6443/api/v1/configmaps/kube-root-ca.crt?watch=true": dial tcp 192.168.56.200:6443: connect: connection refused
Feb 28 14:56:14 k8s-w1 kubelet[5488]: E0228 14:56:14.763122 5488 reflector.go:148] object "kube-system"/"kube-root-ca.crt": failed to watch *v1.ConfigMap: failed to list *v1.ConfigMap: Get "https://192.168.56.200:6443/api/v1/configmaps/kube-root-ca.crt?watch=true": dial tcp 192.168.56.200:6443: connect: connection refused
```



```
root@k8s-w2:~# systemctl status kubelet
● kubelet.service - kubelet: The Kubernetes Node Agent
   Loaded: loaded (/lib/systemd/system/kubelet.service; enabled; vendor preset: enabled)
   Drop-In: /usr/lib/systemd/system/kubelet.service.d
           -10-kubeadm.conf
   Active: active (running) since Wed 2024-02-28 11:46:00 KST; 3h 42min ago
     Docs: https://kubernetes.io/docs/
   Main PID: 5514 (kubelet)
     Tasks: 13 (limit: 2324)
    Memory: 41.5M
   CGroup: /system.slice/kubelet.service
           └─5514 /usr/bin/kubelet --bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf --kubeconfig=/etc/kubernetes/kubelet.conf --config=/var/lib/kubelet/conf

Feb 28 14:56:16 k8s-w2 kubelet[5514]: I0228 14:56:16.295105 5514 trace.go:219] Trace[78198392]: "Reflector ListAndWatch" name:object-"calico-system"/"node-certs" (28-Feb-2024 14:56:16.295105)
Feb 28 14:56:16 k8s-w2 kubelet[5514]: Trace[78198392]: --"Objects listed" error:Get "https://192.168.56.200:6443/api/v1/namespaces/calico-system/secrets?fieldSelector=metadata.name=node-certs" [10.059421132s] END
Feb 28 14:56:16 k8s-w2 kubelet[5514]: E0228 14:56:16.295119 5514 reflector.go:148] object-"calico-system"/"node-certs": Failed to watch *v1.Secret: failed to list *v1.Secret: Get "https://192.168.56.200:6443/api/v1/namespaces/calico-system/secrets?fieldSelector=metadata.name=node-certs": http status 404: Not Found
Feb 28 14:56:16 k8s-w2 kubelet[5514]: I0228 14:56:16.295210 5514 status_manager.go:809] "Failed to get status for pod" podUID=fe3f3f888-e6ef-4cbc-95a5-cc220fcc52eb pod="calico-node"
Feb 28 14:56:16 k8s-w2 kubelet[5514]: E0228 14:56:16.296013 5514 controller.go:146] "Failed to ensure lease exists, will retry" err="Get \"https://192.168.56.200:6443/api/v1/leases/calico-system/calico-node\": http status 404: Not Found"
Feb 28 14:56:24 k8s-w2 kubelet[5514]: I0228 14:56:24.031170 5514 trace.go:219] Trace[956112111]: "Reflector ListAndWatch" name:object-"kube-system"/"kube-proxy" (28-Feb-2024 14:56:24.031170)
Feb 28 14:56:24 k8s-w2 kubelet[5514]: Trace[956112111]: --"Objects listed" error:nil> 30923ms (14:56:24.031170)
Feb 28 14:56:24 k8s-w2 kubelet[5514]: Trace[956112111]: [30.923365991s] [30.923365991s] END
Feb 28 15:20:25 k8s-w2 kubelet[5514]: E0228 15:20:25.178995 5514 kubelet.go:2431] "Housekeeping took longer than expected" err="housekeeping took too long" expected="15s" actual="15.178995s"
```

3. Master에서의 실습

1) alias 설정하기

- 편리한 실습 진행을 위해 'kublet'을 'k'로 alias 설정하기

2) 사용가능한 모든 namespace 조회

k get namespaces //Kubernetes 클러스터에서 현재 사용 가능한 모든 네임스페이스를 조회

```
(calico:N/A) root@k8s-m:~# k get namespaces
NAME                STATUS    AGE
calico-apiserver     Active    4h23m
calico-system        Active    4h26m
default              Active    4h26m
kube-node-lease      Active    4h26m
kube-public          Active    4h26m
kube-system          Active    4h26m
local-path-storage   Active    4h25m
tigera-operator      Active    4h26m
(calico:N/A) root@k8s-m:~#
```

k get ns

```
(calico:N/A) root@k8s-m:~# k get ns
NAME                STATUS    AGE
calico-apiserver     Active    4h25m
calico-system        Active    4h28m
default              Active    4h28m
kube-node-lease      Active    4h28m
kube-public          Active    4h28m
kube-system          Active    4h29m
local-path-storage   Active    4h27m
tigera-operator      Active    4h28m
(calico:N/A) root@k8s-m:~#
```

- default가 현재 위치한 아이임

3) 클러스터 정보확인

k cluster-info

```
(calico:N/A) root@k8s-m:~# k cluster-info
Kubernetes control plane is running at https://192.168.56.200:6443
CoreDNS is running at https://192.168.56.200:6443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
(calico:N/A) root@k8s-m:~#
```

k get pod -v9

// Kubernetes 클러스터에서 실행 중인 모든 파드에 대한 상세한 정보를 조회하는 데 사용

```
(calico:N/A) root@k8s-m:~# k get pod -v9
0228 16:04:09.315096 37343 loader.go:373] Config loaded from file: /root/.kube/config
0228 16:04:09.338384 37343 round_trippers.go:466] curl -v -XGET -H "Accept: application/json;as=Table;v=v1;g=meta.k8s.io,application/json;as=Table;v=v1beta1;g=meta.k8s.io,application/json" -H "User-Agent: kubectl/v1.27.11 (linux/amd64) kubernetes/b9e2ad6" 'https://192.168.56.200:6443/api/v1/namespaces/default/pods?limit=500'
0228 16:04:09.345864 37343 round_trippers.go:510] HTTP Trace: Dial to tcp:192.168.56.200:6443 succeed
0228 16:04:09.371662 37343 round_trippers.go:553] GET https://192.168.56.200:6443/api/v1/namespaces/default/pods?limit=500 200 OK in 33 milliseconds
0228 16:04:09.372612 37343 round_trippers.go:570] HTTP Statistics: DNSLookup 0 ms Dial 7 ms TLSHandshake 14 ms ServerProcessing 7 ms Duration 33 ms
0228 16:04:09.372792 37343 round_trippers.go:577] Response Headers:
0228 16:04:09.372876 37343 round_trippers.go:580] Audit-Id: ca3b128d-d599-45fa-bea0-7ec53d900fbc
0228 16:04:09.373196 37343 round_trippers.go:580] Cache-Control: no-cache, private
0228 16:04:09.373262 37343 round_trippers.go:580] Content-Type: application/json
0228 16:04:09.373321 37343 round_trippers.go:580] X-Kubernetes-Pf-Flowschema-Uid: d2cb573f-f4ac-416c-b7fc-ea6a0d4112ca
0228 16:04:09.373382 37343 round_trippers.go:580] X-Kubernetes-Pf-Prioritylevel-Uid: 2225cbc4-bc06-4f05-956b-226a807564dc
0228 16:04:09.373440 37343 round_trippers.go:580] Content-Length: 2957
0228 16:04:09.373597 37343 round_trippers.go:580] Date: Wed, 28 Feb 2024 07:04:09 GMT
0228 16:04:09.373783 37343 request.go:1188] Response Body: {"kind":"Table","apiVersion":"meta.k8s.io/v1","metadata":{"resourceVersion":"13129"},"columnDefinitions":[{"name":"Name","type":"string","format":"name","description":"Name must be unique within a namespace. Is required when creating resources, although some resources may allow a client to request the generation of an appropriate name automatically. Name is primarily intended for creation idempotence and configuration definition. Cannot be updated. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names#names","priority":0},{name":"Ready","type":"string","format":"","description":"The aggregate readiness state of this pod for accepting traffic","priority":0},{name":"Status","type":"string","format":"","description":"The aggregate status of the containers in this pod","priority":0},{name":"Restarts","type":"string","format":"","description":"The number of times the containers in this pod have been restarted and when the last restart occurred","priority":0}],"rows":[{"name":"default/tigera-operator","ready":"True","status":"Running","restarts":"0"}]}
```

- /root/.kube/config 파일 위치에서 권한을 체크받아 서비스에 적용됨 (정보 확인하고 해당 내용 출력해 줌)

4) 툴을 깔았던 정보 확인하기

kubens

```
(calico:N/A) root@k8s-m:~# kubens
calico-apiserver
calico-system
default
kube-node-lease
kube-public
kube-system
local-path-storage
tigera-operator
(calico:N/A) root@k8s-m:~#
```

5) 이름 바꾸고 싶을 때

kubens {바꾸고 싶은 이름}

```
(calico:N/A) root@k8s-m:~# kubens kube-system
Context "calico" modified.
Active namespace is "kube-system".
(calico:kube-system) root@k8s-m:~#
```

kubens -

//이름 다시 default로 바꿀 때

```
(calico:kube-system) root@k8s-m:~# kubens -  
Context "calico" modified.  
Active namespace is "default".  
(calico:default) root@k8s-m:~#
```

참고) 앞에 'calico:default'가 신경쓰여서 끄고 싶을 때 & 키고 싶을 때

kubeoff (↔ kubeon)

```
(calico:default) root@k8s-m:~# kubeoff  
root@k8s-m:~#
```

6) 파드 만들고 확인하기

kubectl run {파드 이름} --image=nginx //파드 만들기

```
(calico:default) root@k8s-m:~# kubectl run mzcwv --image=nginx  
pod/mzcwv created
```

kubectl get pod //만든 파드 확인하기

```
(calico:default) root@k8s-m:~# k delete pod mzcwv  
pod "mzcwv" deleted
```

참고) 명령형 프로그래밍, 선언형 프로그래밍

명령형 프로그래밍

- 프로그래밍의 데이터와 상태를 변경시키는 구문의 관점에서 연산을 설명하는 프로그래밍 패러다임의 일종
- 데이터와 상태를 어떻게 바꿀지 정의하는 프로그래밍 방법
- 알고리즘을 명시하고 목표는 명시하지x
- ex) "화장실로 가서 칫솔을 들고 치약을 칫솔에 묻힌 다음 칫솔에 물을 묻히고 이를 닦을 때는 윗몸부터 아래로 이를 쓸듯이 닦으렴"

선언형 프로그래밍

- 프로그램이 어떤 방법으로 해야 하는지를 나타내기보다 무엇과 같은지를 설명할 때
- 목표를 명시하고 알고리즘을 명시하지 x
- "준승아 양치 좀 해!!"

7) 만든 pod 제거하기

```
delete pod {파드이름}
```

```
(calico:default) root@k8s-m:~# k delete pod mzcweb  
pod "mzcweb" deleted
```

8) 파드 구성 미리 확인하기

- 파드를 실제로 생성하지 않고, 대신 생성된 파드의 구성을 YAML 형식으로 출력하여 미리 파드의 구성 확인하기

```
(calico:default) root@k8s-m:~# k run mzcweb --image nginx --dry-run -o yaml  
W0228 16:22:45.391267 42435 helpers.go:692] --dry-run is deprecated and can be replaced with --dry-run=client.  
apiVersion: v1  
kind: Pod  
metadata:  
  creationTimestamp: null  
  labels:  
    run: mzcweb  
  name: mzcweb  
spec:  
  containers:  
  - image: nginx  
    name: mzcweb  
    resources: {}  
  dnsPolicy: ClusterFirst  
  restartPolicy: Always  
status: {}  
(calico:default) root@k8s-m:~#
```

//kubectl run: 새로운 파드를 생성하는 Kubernetes 명령어

//mzcweb: 생성할 파드의 이름

//--image nginx: 파드에 사용할 컨테이너 이미지

//--dry-run: 파드를 실제로 생성하지 않고 구성 파일만 생성

//-o yaml: 생성된 파드 구성을 YAML 형식으로 출력

- apiVersion은 우리가 맞춰줘야함
- spec : 컨테이너 정보

9) yaml 파일 만들기

```
k run mzcweb --image nginx --dry-run -o yaml > mzcweb.yaml
```

```
(calico:default) root@k8s-m:~# k run mzcweb --image nginx --dry-run -o yaml > mzcweb.yaml  
W0228 16:24:30.695936 42893 helpers.go:692] --dry-run is deprecated and can be replaced with --dry-run=client.
```

// > mzcweb.yaml: 생성된 YAML 형식의 파드 구성을 mzcweb.yaml 파일에 저장

cat mzcweb.yaml //yaml 파일 출력하기

```
c(calico:default) root@k8s-m:~# cat mzcweb.yaml
File: mzcweb.yaml
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    creationTimestamp: null
5    labels:
6      run: mzcweb
7    name: mzcweb
8  spec:
9    containers:
10   - image: nginx
11     name: mzcweb
12     resources: {}
13     dnsPolicy: ClusterFirst
14     restartPolicy: Always
15   status: {}
```

- apiVersion : yaml에서 정의한 api 버전 정보
- kind : 리소스 종류