

[Operation] Logging & Troubleshooting, Backup, Upgrade

Logging & Troubleshooting

- 구글의 사이트 신뢰성 엔지니어링(<https://sre.google>)

Google Site Reliability Engineering Home Spotlight Resources Books Mobaa Classroom Careers

What is Site Reliability Engineering (SRE)?

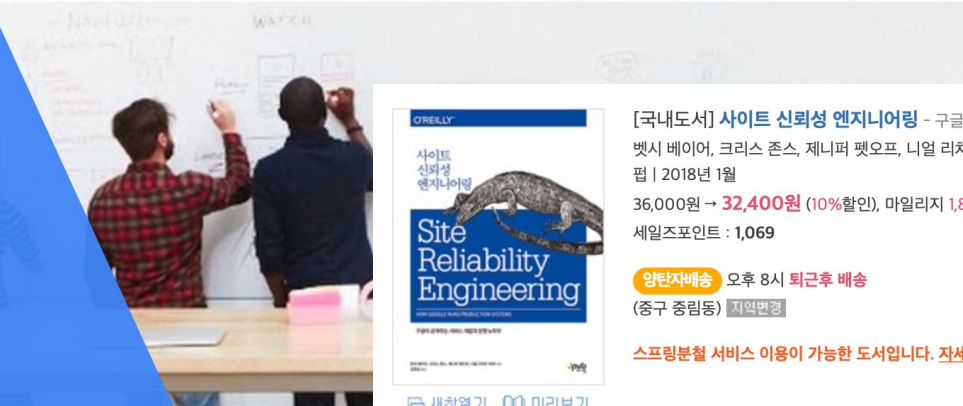
SRE is what you get when you treat operations as if it's a software problem. Our mission is to protect, provide for, and progress the software and systems behind all of Google's public services — Google Search, Ads, Gmail, Android, YouTube, and App Engine, to name just a few — with an ever-watchful eye on their availability, latency, performance, and capacity.

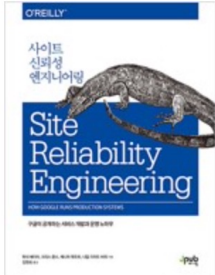
What is SRE?

Since 2004, SRE has evolved to become the industry-leading practice for service reliability.

Hear from key figures about the history of SRE and what's next for the SRE community.

[Watch video](#)





[국내도서] **사이트 신뢰성 엔지니어링** - 구글이 공개하는 서비스 개발과 운영 노하우
벋시 베이어, 크리스 존스, 제니퍼 펫오프, 니얼 리처드 머피 (지은이), 장현희 (옮긴이) | 제이
펍 | 2018년 1월
36,000원 → **32,400원** (10%할인), 마일리지 1,800원 (5% 적립)
세일즈포인트 : 1,069

양탄자배송 오후 8시 퇴근후 배송
(중구 중림동) **지역변경**

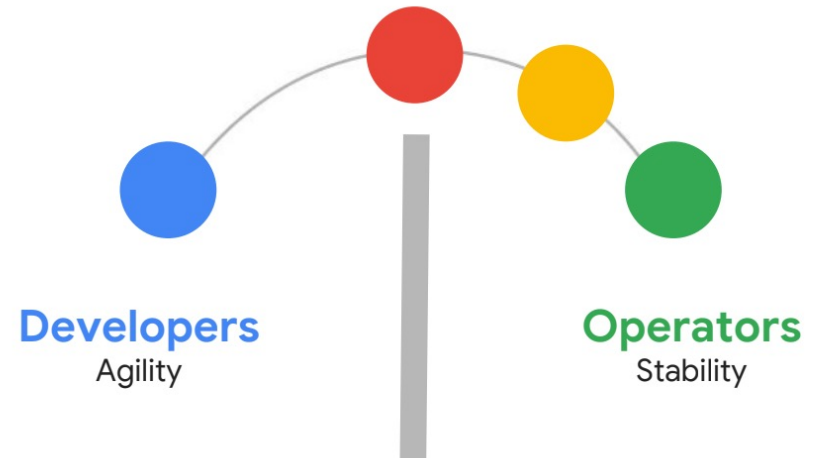
[스프링분철 서비스 이용이 가능한 도서입니다. 자세히보기](#)

[새창열기](#) [미리보기](#)

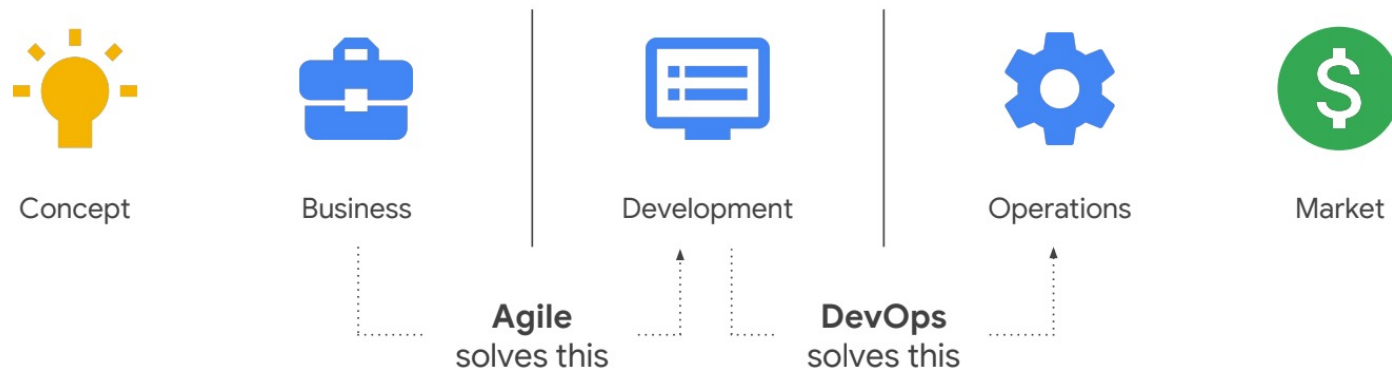
그림 출처 : 구글 SRE, 알라딘 131

Logging & Troubleshooting

- 개발자와 운용자는 추구 목표가 다름



- 아이디어가 마켓까지 오는 시간이 점점 빨라지고 있음



Logging & Troubleshooting

- SRE들은 DevOps를 구현
 - DevOps는 IT의 Silo, Ops, Network, Security 등을 허무는 방식, 가이드라인, 문화의 집합
 - SRE는 그동안 찾은 작업 방식, 이러한 사례로 구체화하는 신념, 그리고 역할에 대한 집합

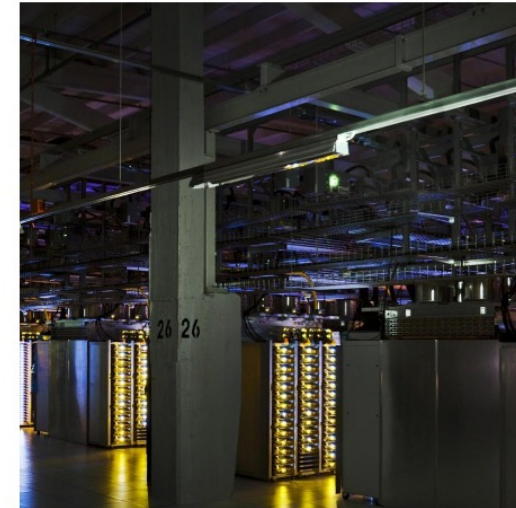
class SRE implements DevOps

DevOps

is a set of **practices**,
guidelines and culture
designed to break down
silos in IT, ops, networks,
security, etc.

Site Reliability Engineering

is a set of **practices** we've
found to work, some **beliefs**
that animate those practices,
and **a job role**.



Logging & Troubleshooting

- 메트릭&모니터링 / 용량 관리 / 변화 관리 / 긴급 대응 / 문화

- 메트릭&모니터링 : 모니터링 지표를 정의하고, 정의된 지표를 모니터링 시스템으로 구성.

인사이트를 통해 시스템이 안정적인 상황과 또는 장애가 나는 지표는 무엇인지, 왜인지? 어떻게 해결할지 고민.

기본적으로 SRE에서 가장 중요한 부분은 모든 것을 데이터 화하고, 의사결정을 데이터 기반으로 하는 것

Summary of SRE practices



Metrics & Monitoring

- SLOs
- Dashboards
- Analytics



Capacity Planning

- Forecasting
- Demand-driven
- Performance



Change Management

- Release process
- Consulting design
- Automation



Emergency Response

- Oncall
- Analysis
- Postmortems



Culture

- Toil management
- Engineering alignment
- Blamelessness

Logging & Troubleshooting

- 모니터링 : 쿼리 카운트와 종류, 에러 카운트와 종류, 프로세싱 타임, 서버의 라이프타임과 같은 수치를 실시간으로 수집/처리/집계/보여주는
 - 메트릭 : 특정 시스템에서 리소스, 응용 프로그램 작업 또는 비즈니스 특성이 특정 시점에서의 수치로 표현되는 것. 보통 키-밸류(key-value) 형태로 수집된 숫자가 일반적
 - 로깅(Logging) : 로깅은 메트릭보다 훨씬 많은 데이터를 포함하는 시스템이나 애플리케이션의 이벤트로 나타나며, 이러한 이벤트에 의해 생성되는 모든 정보를 포함
 - 트레이스(Tracing) : 특정 고유한 식별자가 모든 시스템에 걸쳐 전체 수명 주기 동안 추적될 수 있도록 제공하는 로깅의 특별한 경우

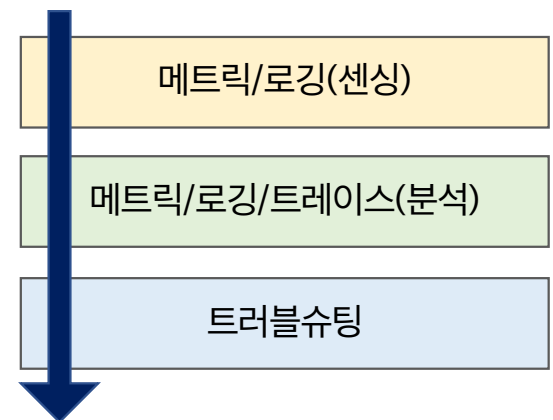
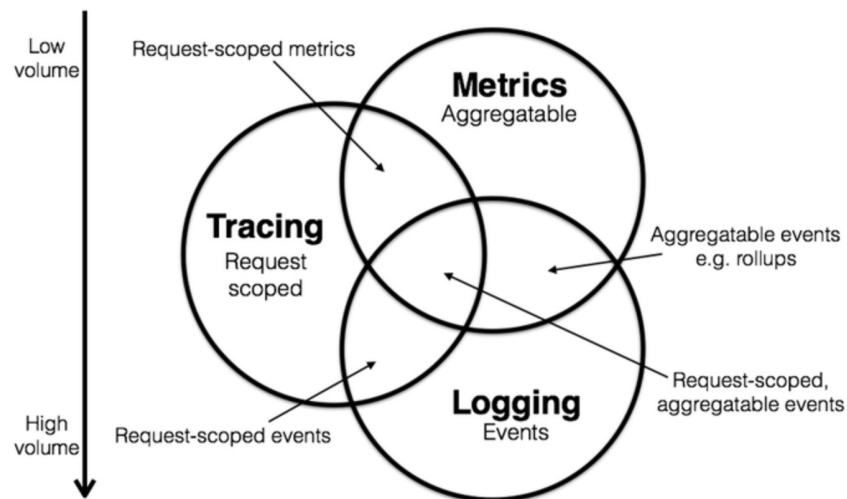


그림 출처 : <https://peter.bourgon.org/blog/2017/02/21/metrics-tracing-and-logging.html>
https://cdn.emetro.co.kr/pdf/2015/08/05/kr_metro_1_0805_14.pdf

Logging & Troubleshooting

- 쿠버네티스에서의 기본 로깅 (<https://kubernetes.io/ko/docs/concepts/cluster-administration/logging/#쿠버네티스의-기본-로깅>)

```
apiVersion: v1
kind: Pod
metadata:
  name: counter
spec:
  containers:
  - name: count
    image: busybox
    args: [/bin/sh, -c,
           'i=0; while true; do echo "$i: $(date)"; i=$((i+1)); sleep 1; done']
```

```
[centos@osk-master-01 ~]$ kubectl apply -f https://k8s.io/examples/debug/counter-pod.yaml
```

```
pod/counter created
```

```
[centos@osk-master-01 ~]$ kubectl logs counter
```

```
0: Sat Aug 21 03:18:01 UTC 2021
```

```
1: Sat Aug 21 03:18:02 UTC 2021
```

```
2: Sat Aug 21 03:18:03 UTC 2021
```

```
3: Sat Aug 21 03:18:04 UTC 2021
```

```
4: Sat Aug 21 03:18:05 UTC 2021
```

Logging & Troubleshooting

- 쿠버네티스에서의 기본 로깅 (<https://kubernetes.io/ko/docs/concepts/cluster-administration/logging/#쿠버네티스의-기본-로깅>)

```
[centos@osk-master-01 ~]$ kubectl logs counter
```

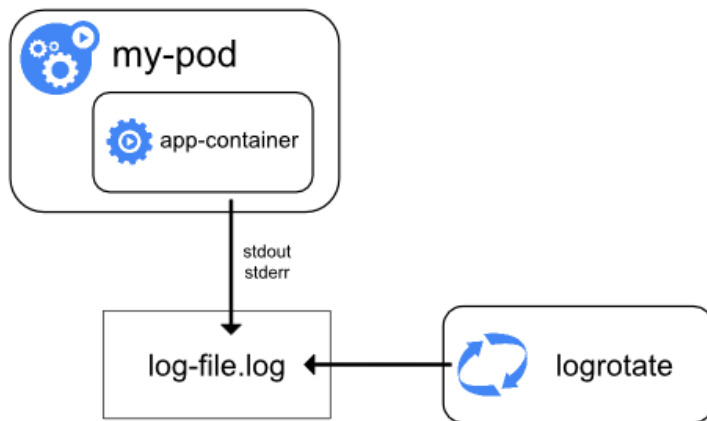
```
0: Sat Aug 21 03:18:01 UTC 2021
```

- 이 로그의 실제 위치는?

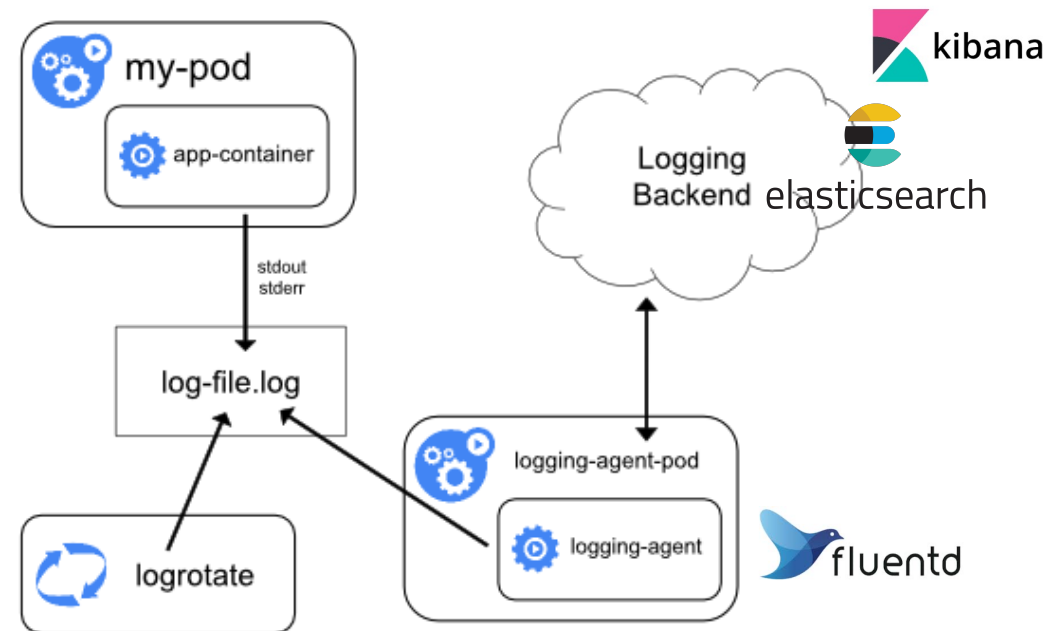
```
[root@osk-worker-02 containers]# ls -lrt /var/log/containers/counter_default_count-b7efee0b84250fd890a4e5275e6078c0e553e509ac56f100368bfa3650673190.log
lrwxrwxrwx. 1 root root 78 Aug 21 03:18 /var/log/containers/counter_default_count-b7efee0b84250fd890a4e5275e6078c0e553e509ac56f100368bfa3650673190.log -> /var/log/pods/default_counter_657461bd-1677-4384-bf7c-15dc29271837/count/0.log
[root@osk-worker-02 containers]# ls -lrt /var/log/pods/default_counter_657461bd-1677-4384-bf7c-15dc29271837/count/0.log
lrwxrwxrwx. 1 root root 165 Aug 21 03:18 /var/log/pods/default_counter_657461bd-1677-4384-bf7c-15dc29271837/count/0.log -> /var/lib/docker/containers/b7efee0b84250fd890a4e5275e6078c0e553e509ac56f100368bfa3650673190/b7efee0b84250fd890a4e5275e6078c0e553e509ac56f100368bfa3650673190-json.log
[root@osk-worker-02 containers]# ls -lrt /var/lib/docker/containers/b7efee0b84250fd890a4e5275e6078c0e553e509ac56f100368bfa3650673190/b7efee0b84250fd890a4e5275e6078c0e553e509ac56f100368bfa3650673190-json.log
-rw-r-----. 1 root root 215303 Aug 21 03:52 /var/lib/docker/containers/b7efee0b84250fd890a4e5275e6078c0e553e509ac56f100368bfa3650673190/b7efee0b84250fd890a4e5275e6078c0e553e509ac56f100368bfa3650673190-json.log
[root@osk-worker-02 containers]# file /var/log/pods/default_counter_657461bd-1677-4384-bf7c-15dc29271837/count/0.log
/var/log/pods/default_counter_657461bd-1677-4384-bf7c-15dc29271837/count/0.log: symbolic link to `/var/lib/docker/containers/b7efee0b84250fd890a4e5275e6078c0e553e509ac56f100368bfa3650673190/b7efee0b84250fd890a4e5275e6078c0e553e509ac56f100368bfa3650673190-json.log'
[root@osk-worker-02 containers]# file /var/lib/docker/containers/b7efee0b84250fd890a4e5275e6078c0e553e509ac56f100368bfa3650673190/b7efee0b84250fd890a4e5275e6078c0e553e509ac56f100368bfa3650673190-json.log
/var/lib/docker/containers/b7efee0b84250fd890a4e5275e6078c0e553e509ac56f100368bfa3650673190/b7efee0b84250fd890a4e5275e6078c0e553e509ac56f100368bfa3650673190-json.log: ASCII text
[root@osk-worker-02 containers]# head -3 /var/lib/docker/containers/b7efee0b84250fd890a4e5275e6078c0e553e509ac56f100368bfa3650673190/b7efee0b84250fd890a4e5275e6078c0e553e509ac56f100368bfa3650673190-json.log
{"log":"0: Sat Aug 21 03:18:01 UTC 2021\n","stream":"stdout","time":"2021-08-21T03:18:01.679793924Z"}
{"log":"1: Sat Aug 21 03:18:02 UTC 2021\n","stream":"stdout","time":"2021-08-21T03:18:02.681347872Z"}
```


Logging & Troubleshooting

- 쿠버네티스에서의 다른 로깅 아키텍처
- 노드 레벨에서의 로깅

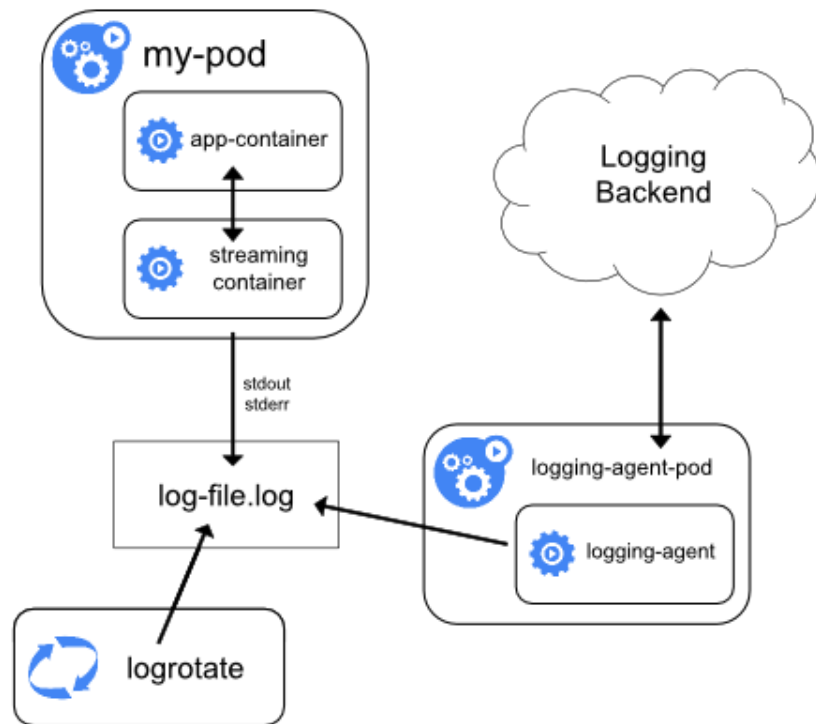


- 클러스터 레벨에서의 로깅(노드 로깅 에이전트사용)

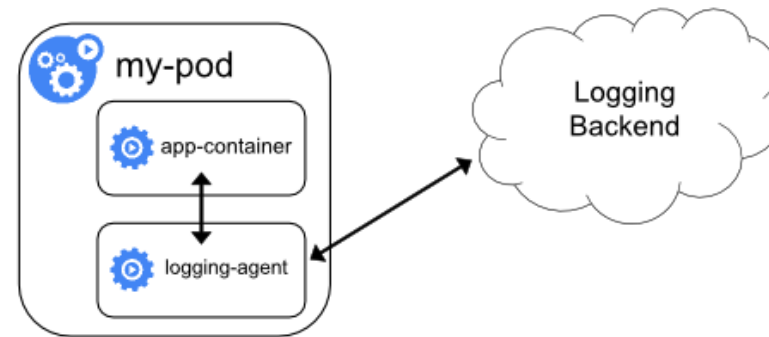


Logging & Troubleshooting

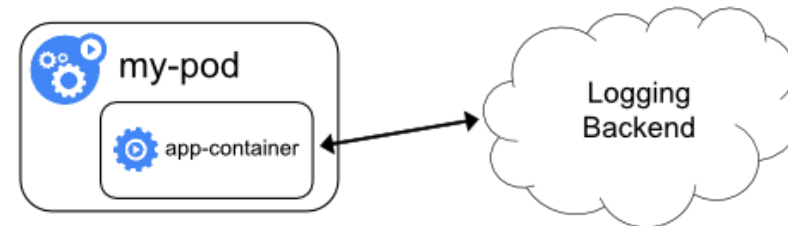
- 쿠버네티스에서의 다른 로깅 아키텍처
- 로깅 에이전트와 함께 사이드카 컨테이너 사용



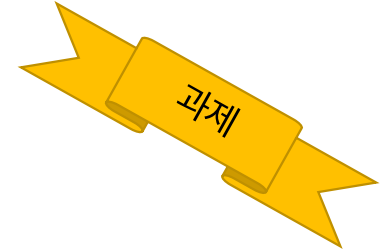
- 로깅 에이전트가 있는 사이드카 컨테이너



- 애플리케이션에서 직접 로그 노출



Troubleshooting 절차/Debug 순서/참고 자료



- Trouble Shooting 파일 전체 노드에 다운로드 및 실행
 - Pod 생성이 가능토록 조치하고, Pod가 생성된 화면 캡처 및 원인 / 해결방법 강사 메일로 제출 (ohsk@kakao.com)
- Trouble Shooting 절차/Debug 순서 : 로그를 확인, 쿠버네티스의 구조/특징을 생각
- 참고 자료 : 본 자료 앞 부분 쿠버네티스 구조

[master노드 1~3, worker 노드 1~2 모두]

```
# wget http://172.30.5.154/troubleshooting
```

```
# ./troubleshooting
```

```
[centos@osk-master-01 ~]$ ./troubleshooting
```

Create a pod. If you can't create pods, trouble shooting.

After the trouble shooting, capture the screen that created the pod and send an e-mail to ohsk@kakao.com along with the cause and solution.

```
[centos@osk-worker-01 ~]$ ./troubleshooting
```

Create a pod. If you can't create pods, trouble shooting.

After the trouble shooting, capture the screen that created the pod and send an e-mail to ohsk@kakao.com along with the cause and solution.

모니터링 클러스터 컴포넌트

- #kubectl top node
- #kubectl top pod
- (참고) 리눅스 TOP 명령어 (table of process) : CPU/메모리 정보 표시

```
[root@osk-master-01 ~]# kubectl top --help
Display Resource (CPU/Memory) usage.
```

The top command allows you to see the resource consumption for nodes or pods.
This command requires Metrics Server to be correctly configured and working on the server.

Available Commands:

node	Display Resource (CPU/Memory) usage of nodes
pod	Display Resource (CPU/Memory) usage of pods

Usage:

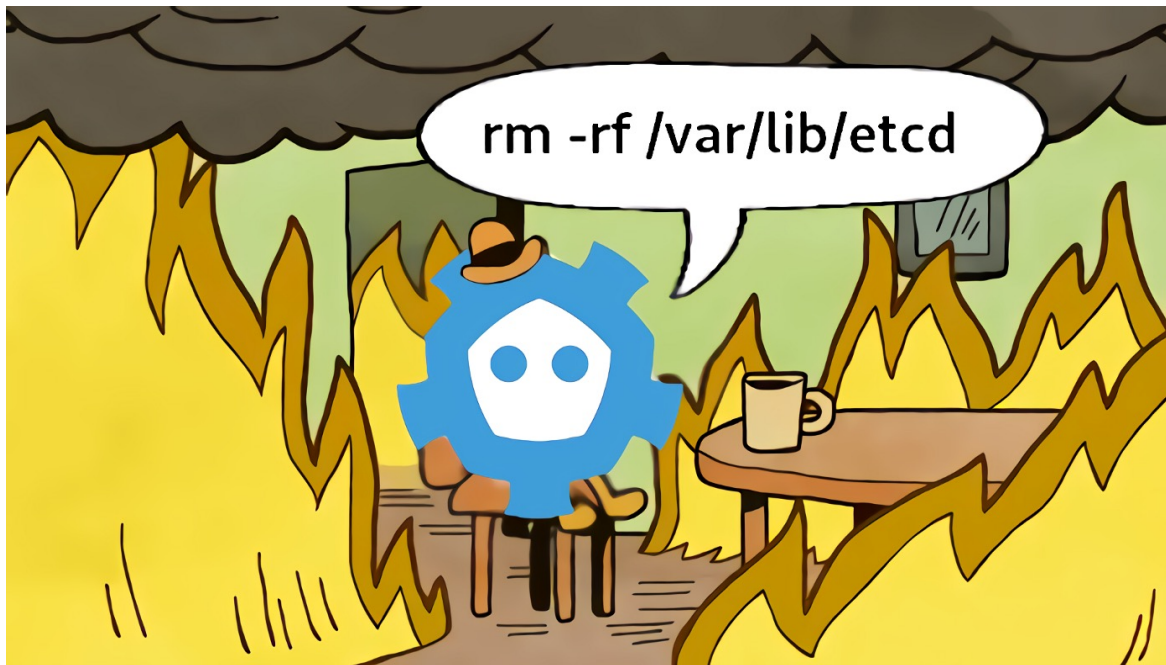
```
kubectl top [flags] [options]
```

Use "kubectl <command> --help" for more information about a given command.

Use "kubectl options" for a list of global command-line options (applies to all commands).

백업 및 복구

- 쿠버네티스 클러스터의 정보들을 갖고 있는 etcd.
- Etcd 문제 발생 시, 쿠버네티스 클러스터 전체가 영향을 받게 됨



`sudo rm -rf /`

진정 새로운 무언가를 창조하려 한다면 처음부터 다시 시작해야 합니다.

백업 및 복구

- Pod로 구동된 etcd 버전 확인 후 해당 etcdctl 설치 (<https://github.com/etcd-io/etcd/releases>)
[centos@osk-master-01 ~]\$ kubectl describe -n kube-system pod etcd-osk-master-01.kr-central-1.c.internal
Name: etcd-osk-master-01.kr-central-1.c.internal
Namespace: kube-system
생략
Controlled By: Node/osk-master-01.kr-central-1.c.internal
Containers:
 etcd:
 Container ID: docker://f8a0505c83155284b02973a4e2bcb2f10fa9c6d5017871ed5ec821a3c594b33a
 Image: k8s.gcr.io/etcd:3.5.0-0
[centos@osk-master-01 ~]# ETCD_VER=v3.5.0
[centos@osk-master-01 ~]# wget https://storage.googleapis.com/etcd/\${ETCD_VER}/etcd-\${ETCD_VER}-linux-amd64.tar.gz
[centos@osk-master-01 ~]# tar xzvf etcd-\${ETCD_VER}-linux-amd64.tar.gz
[centos@osk-master-01 ~]# sudo mv etcd-\${ETCD_VER}-linux-amd64/etcdctl /usr/local/bin/etcdctl

[centos@osk-master-01 ~]# etcdctl version
etcdctl version: 3.5.0
API version: 3.5

백업 및 복구



- etcd backup/restore 명령어 복마크!
- <https://discuss.kubernetes.io/t/etcd-backup-and-restore-management/11019/11>
- 백업 `ETCDCTL_API=3 etcdctl --endpoints=https://[127.0.0.1]:2379 --cacert=/etc/kubernetes/pki/etcd/ca.crt \ --cert=/etc/kubernetes/pki/etcd/server.crt --key=/etc/kubernetes/pki/etcd/server.key \ snapshot save /tmp/snapshot-pre-boot.db`
- 복구 `ETCDCTL_API=3 etcdctl --endpoints=https://[127.0.0.1]:2379 --cacert=/etc/kubernetes/pki/etcd/ca.crt | --name=master \ --cert=/etc/kubernetes/pki/etcd/server.crt --key=/etc/kubernetes/pki/etcd/server.key \ --data-dir /var/lib/etcd-from-backup \ --initial-cluster=master=https://127.0.0.1:2380 | --initial-cluster-token etcd-cluster-1 \ --initial-advertise-peer-urls=https://127.0.0.1:2380 | snapshot restore /tmp/snapshot-pre-boot.db`

백업 및 복구



- etcd backup/restore 명령어 복마크!

- <https://discuss.kubernetes.io/t/etcd-backup-and-restore-management/11019/11>

각 master node ip

- 백업 `ETCDCTL_API=3 etcdctl --endpoints=https://[127.0.0.1]:2379 --cacert=/etc/kubernetes/pki/etcd/ca.crt --cert=/etc/kubernetes/pki/etcd/server.crt --key=/etc/kubernetes/pki/etcd/server.key snapshot save /tmp/snapshot-pre-boot.db`

백업 명령어

백업을 저장할 경로/이름

현재 구동된 etcd 에서 확인
(인증서/키 경로)

- 복구 `ETCDCTL_API=3 etcdctl --endpoints=https://[127.0.0.1]:2379 --cacert=/etc/kubernetes/pki/etcd/ca.crt --name=master --cert=/etc/kubernetes/pki/etcd/server.crt --key=/etc/kubernetes/pki/etcd/server.key --data-dir /var/lib/etcd-from-backup --initial-cluster=master=https://127.0.0.1:2380 --initial-cluster-token etcd-cluster-1 --initial-advertise-peer-urls=https://127.0.0.1:2380 snapshot restore /tmp/snapshot-pre-boot.db`

복구 명령어

복구할 파일이 저장된 경로/이름

기존 백업시에 설정되었던 값
(or 문제에서 주어진 값)¹⁴⁵

백업 및 복구



- etcd backup/restore 명령어 복마크!

- <https://discuss.kubernetes.io/t/etcd-backup-and-restore-management/11019/11>

- 백업 `ETCDCTL_API=3 etcdctl --endpoints=https://[127.0.0.1]:2379 --cacert=/etc/kubernetes/pki/etcd/ca.crt --cert=/etc/kubernetes/pki/etcd/server.crt --key=/etc/kubernetes/pki/etcd/server.key snapshot save /tmp/snapshot-pre-boot.db`

etcd 데이터를 저장할 경로
(기존 데이터 덮어쓰지 않게 다른 공간 추천)

- 복구 `ETCDCTL_API=3 etcdctl --endpoints=https://[127.0.0.1]:2379 --cacert=/etc/kubernetes/pki/etcd/ca.crt --name=master --cert=/etc/kubernetes/pki/etcd/server.crt --key=/etc/kubernetes/pki/etcd/server.key --data-dir /var/lib/etcd-from-backup --initial-cluster=master=https://127.0.0.1:2380 --initial-cluster-token etcd-cluster-1 --initial-advertise-peer-urls=https://127.0.0.1:2380 snapshot restore /tmp/snapshot-pre-boot.db`

백업 및 복구



- etcd backup/restore 명령어 복마크!

- <https://discuss.kubernetes.io/t/etcd-backup-and-restore-management/11019/11>

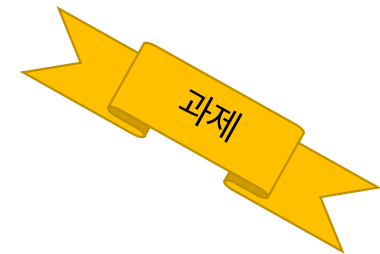
- 백업 `ETCDCTL_API=3 etcdctl --endpoints=https://[127.0.0.1]:2379 --cacert=/etc/kubernetes/pki/etcd/ca.crt --cert=/etc/kubernetes/pki/etcd/server.crt --key=/etc/kubernetes/pki/etcd/server.key snapshot save /tmp/snapshot-pre-boot.db`

etcd 데이터를 저장할 경로
(기존 데이터 덮어쓰지 않게 다른 공간 추천)

- 복구 `ETCDCTL_API=3 etcdctl --endpoints=https://[127.0.0.1]:2379 --cacert=/etc/kubernetes/pki/etcd/ca.crt --name=master --cert=/etc/kubernetes/pki/etcd/server.crt --key=/etc/kubernetes/pki/etcd/server.key --data-dir /var/lib/etcd-from-backup --initial-cluster=master=https://127.0.0.1:2380 --initial-cluster-token etcd-cluster-1 --initial-advertise-peer-urls=https://127.0.0.1:2380 snapshot restore /tmp/snapshot-pre-boot.db`

Etcd는 static Pod 로 구성되어 있음.
Static Pod는 어떻게 구성되었는지 떠올리기!

백업 및 복구



- 마스터 노드에 snapshot 파일 다운로드 및 복구
 - 복구 된 후 Pod 상태를 조회하고, 화면 캡처 하여 강사 메일로 제출 (ohsk@kakao.com)
 - 3개 master 노드를 모두 복구 하지 않고 1개만 복구 완료하여도 Pass로 인정
- Hint1 : Static Pod의 Volumes 수정
- Hint2 : 현재 각자 환경에 설치되어 실행중인 etcd config 준용

[master노드]

```
# wget http://172.30.5.154/snapshot-pre-boot.db
```

```
[root@dyjs-master-01 manifests]# kubectl get pod
```

NAME	READY	STATUS	RESTARTS	AGE
capture-the-screen-and-send-an-email-to-ohsk	1/1	Running	0	12m
	1/1	Running	0	12m
	1/1	Running	0	12m

복구 완료 되면 어떤 Pod가 생기는지 확인



그동안 수고 많으셨습니다



(참고) Kakao Kubernetes Engine 사용하기

(참고: https://console.kakaoi.io/docs/posts/k8se/k8se_htg/2021-05-31-k8se_htg_settingKubectl/k8se_htg_settingKubectl/)

The screenshot shows a web browser window with the URL `console.kakaoi.io`. The page is titled "Kubernetes Engine - 카카오톡의 노하우가 집약된 클라우드" and "사용자 가이드 - kubectl 제어 설정하기". The left sidebar shows the navigation menu with "Kubernetes Engine" expanded, listing "Overview", "Concepts", "How-to guides", and "클러스터 만들기", "클러스터 관리하기", "kubectl 제어 설정하기", "인그레스 컨트롤러 설정하기", and "로드밸런서 생성하기". The main content area is titled "kubectl 제어 설정하기" and contains the following text:

클러스터를 제어하기 위한 Kubernetes 커맨드 라인 도구인 **kubectl** 제어 설정 방법을 안내합니다. 여기에는 **kubectl** 클라이언트 설치 방법과 클러스터 접근 설정을 위한 **kubeconfig** 파일 다운로드 방법이 포함됩니다.

kubectl 설치 및 설정하기

kubectl 클라이언트 설치

Kubernetes 공식 가이드 문서를 따라 kubectl 클라이언트를 설치합니다. 사용자의 OS 환경에 따라 설치 방법을 확인 후 설치를 진행해주세요.

kubectl 제어 설정

kubectl 클라이언트를 설치한 후, 다음을 따라 클러스터에 대한 kubectl 제어 설정을 진행합니다.

(참고) Kakao Kubernetes Engine 사용하기 - PC환경설정

- 쿠버네티스 클러스터에 명령어를 내릴 수 있는 CLI 도구
- 애플리케이션 배포/ 클러스터 리소스 검사, 관리, 로그 확인 등 가능
 - ✓ 리눅스에 kubectl 설치하기 : <https://kubernetes.io/ko/docs/tasks/tools/install-kubectl-linux/>
 - ✓ macOS에 kubectl 설치하기 : <https://kubernetes.io/ko/docs/tasks/tools/install-kubectl-macos/>
 - ✓ 윈도우에 kubectl 설치하기 : <https://kubernetes.io/ko/docs/tasks/tools/install-kubectl-windows/>

kakaoenterprise



(참고) Kakao Kubernetes Engine 사용하기 - 클러스터 배포

- Kubernetes Engine > 새 클러스터 만들기
 - ✓ 클러스터 이름 : 수강생 간 구분될 수 있게 가급적 이름 약자를 포함해서 작성
 - ✓ 노드 풀 : lkln
 - ✓ 인스턴스타입 : a1.4c16m
 - ✓ 노드수 : 3

kakaoenterprise

나만의 쿠버네티스 클러스터



새 클러스터 만들기

클러스터 정보

클러스터 이름

osk-test

클러스터 설명 (선택)

60자 이내로 작성

네트워크 구성

관리 페이지

네트워크

likelion-private-01

서브넷

likelion-private-01

노드 풀

노드 풀 이름

lkln

노드 풀 설명 (선택)

60자 이내로 작성

인스턴스 타입

a1.4c16m (4vCPU 16GB 메모리)

볼륨 타입 / 크기

SSD 50

노드 수

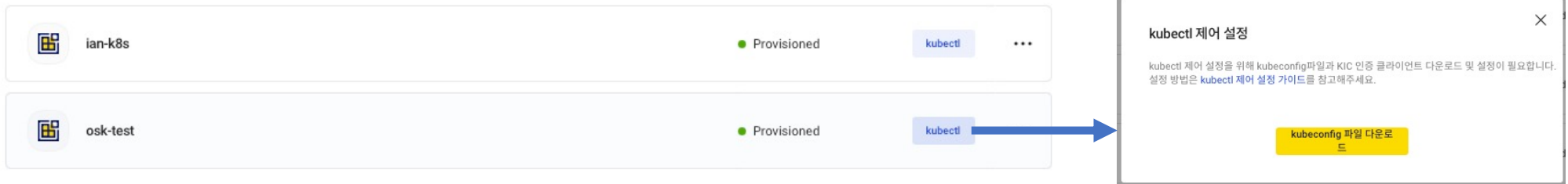
3

키페어

kakao-osk

(참고) Kakao Kubernetes Engine 사용하기 - 클러스터 배포

- Kubectl을 클릭하여 클러스터 접근 정보가 담긴 Kubeconfig 파일 다운로드 (파일명 : kubeconfig-{clusterName})



```
osk-MacBook-Pro:~$ cat kubeconfig-osk-test.yaml
apiVersion: v1
clusters:
- cluster:
  certificate-authority-data: LS0tLS1CRUdJTiBDRVJUSUJZQ0FURSBtLS0tCk1
  XcwQkFRc0ZBREFTVjN0b0VRWURWUFRXdwcmRXSmWkY201bGRHVnpNQjRFRjR3eE1EY3p
  ZURVRNQNkVHGFVRQpBeE1LYTNWapVpSnVWVjY3pDQ0FTSx0EUVlKS29aSWh2Y05BUUVC
  Wdm05MUt1QXh1TGZwS31jM01Baml5S0N2W0thY0ZlTX1PePNERUg4Z00ekdzazc5cj1K
  RE1oSzNRQ0J2J3kx3cWWhk256bDR0R11mVFA1Z21XVFRlSmNaZQorVnNkXDF2bXyRtPMV
  0hEbWRsREVDQ1p0N09QaW1mCl0eUx0dWFM05VRTVqczdub212Um9CemVnbWc1Tkhdz1
  FmWVR2Um3K0JIS0VnNy9yeTZ0Z01PZFPZpGTTZpWY2QVJFRU1HZ0U2dj1SMXRRLS1V
  IdGx0B3RUFBU1tTUNRd0RnWURWUjBQ0VFIL0JBUURBZ0trTUJJR0ExVWRFd0VCC193U
  U1NQV1QV RWJEWk4xbVRyNzRlS1B1VEM0TDRIaFQKTG82VXFZGg4eG0vUG1XUjN3Kzd
  VUUF3uOVNXU0hWdEUQpMaUk0Zn1xYjF2eJ3b0EtMZXJZKRI3ncy0HNZSDBWU5akV
  S1R2dPQUp1V09nd0l1230Mm1uTTZrc-jkx029nb0pjdKpR2x2Wf1NZWE3S01mTGE
  KRXdt7rb1d4Zzd1T2U4em16M2V4Q2FDeWNRajZyV2VYVWpVQjZPbnh6UFFocUZZT
  WpuWVNsSjd0cn RVJFRStsaTA9Ci0tLS0tRU5EIENFU1RJRklDQVRFLS0tLS0K
  server: https://10.183.67.34:6443
  name: osk-test
contexts:
- context:
  cluster: osk-test
  user: osk-test-admin
  name: osk-test-admin@osk-test
current-context: osk-test-admin@osk-test
kind: Config
preferences: {}
users:
- name: osk-test-admin
  user:
    exec:
      apiVersion: client.authentication.k8s.io/v1beta1
      args: null
      command: kic-keystone-auth
      env: null
osk-MacBook-Pro:~$
```

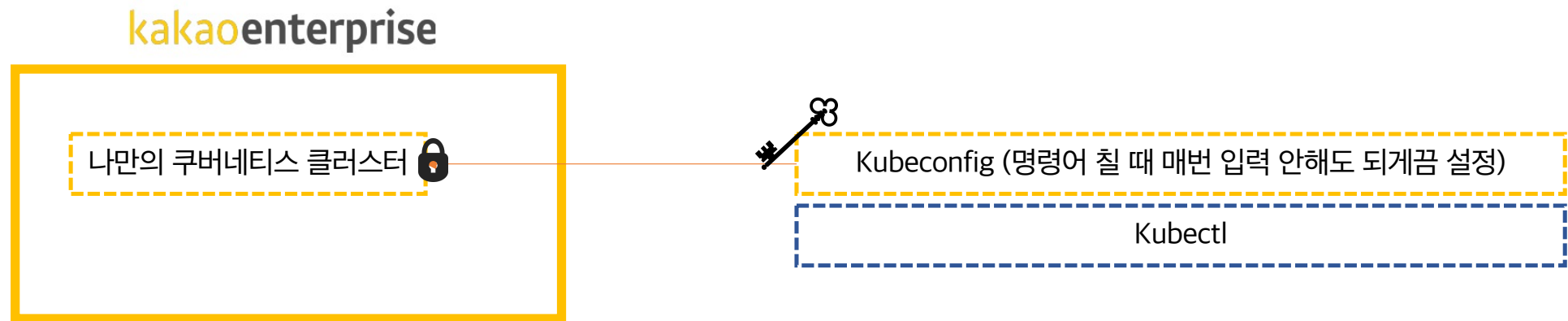

(참고) Kakao Kubernetes Engine 사용하기 - PC 환경설정

- PC에서 kubectl 명령어를 쉽게 입력할 수 있도록, 다운받은 Kubeconfig 을 활용한 환경설정 진행

✓ 맥북(리눅스) : `export KUBE_CONFIG="{DownloadPath}/{kubeconfigFile}"`

✓ 윈도우 : `set KUBE_CONFIG={DownloadPath}\{kubeconfigFile}`

`osk@osk-MacBook-Pro lkln-kakao % export KUBE_CONFIG=/Users/osk/Desktop/lkln-kakao/kubeconfig-osk-test.yaml`



(참고) Kakao Kubernetes Engine 사용하기 - PC 환경설정

- (추가 편의) 재부팅 마다 설정 하지 않도록 추가 세팅

✓ 맥북(리눅스) : export \$KUBE_CONFIG >> ~/.zshrc

```
osk@osk-MacBook-Pro lln-kakao % echo $KUBE_CONFIG
```

```
/Users/osk/Desktop/lln-kakao/kubeconfig-osk-test.yaml
```

```
osk@osk-MacBook-Pro lln-kakao % echo $KUBE_CONFIG >> ~/.zshrc
```

[맥북용 카카오 인증 파일은 zsh에서만 구동됨]

```
osk-MacBook-Pro:~ osk$
```

```
osk-MacBook-Pro:~ osk$ echo $SHELL
```

```
/bin/bash
```

```
osk-MacBook-Pro:~ osk$ chsh -s /bin/zsh
```

```
Changing shell for osk.
```

```
Password for osk:
```

```
osk-MacBook-Pro:~ osk$ zsh
```

```
osk@osk-MacBook-Pro ~ %
```

kakaoenterprise

나만의 쿠버네티스 클러스터

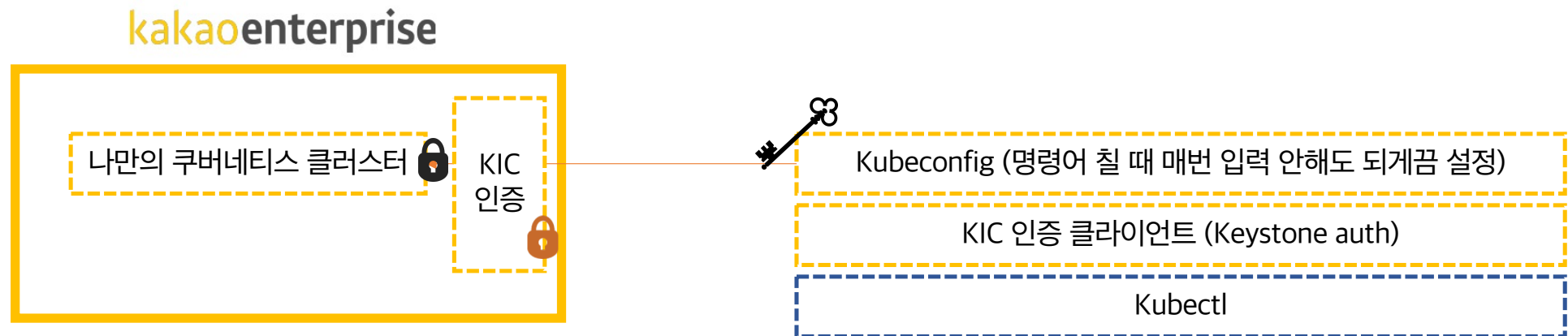


Kubeconfig
(재부팅 이후에도 명령어 칠 때 매번 입력 안해도 되게끔 설정)

Kubectl

(참고) Kakao Kubernetes Engine 사용하기 - PC 환경설정

- KIC(Kakao i Cloud) 인증 클라이언트 다운로드 및 설정
- 사용자 인증을 위해 KIC 인증 클라이언트를 다운로드
 - 맥북 : https://objectstorage.kr-central-1.kakaoi.io/v1/c901636667fe4668ae745ea7de035ae9/kic-keystone-auth-client/darwin_amd64/kic-keystone-auth
 - 윈도우 : https://objectstorage.kr-central-1.kakaoi.io/v1/c901636667fe4668ae745ea7de035ae9/kic-keystone-auth-client/windows_amd64/kic-keystone-auth.exe
 - 리눅스 : https://objectstorage.kr-central-1.kakaoi.io/v1/c901636667fe4668ae745ea7de035ae9/kic-keystone-auth-client/linux_amd64/kic-keystone-auth



(참고) Kakao Kubernetes Engine 사용하기 - API 키/인증 설정

- API 키 만들기


<https://console.kakaoi.io/settings/key>

사용자 설정

사용자 API 키

사용자 API 키 정보

사용자 API 키

 **ike-cluster-member-osk-test-likelion-ic**
osk-test 클러스터가 사용하는 API 키입니다. 삭제 시

사용자 API 키 이름

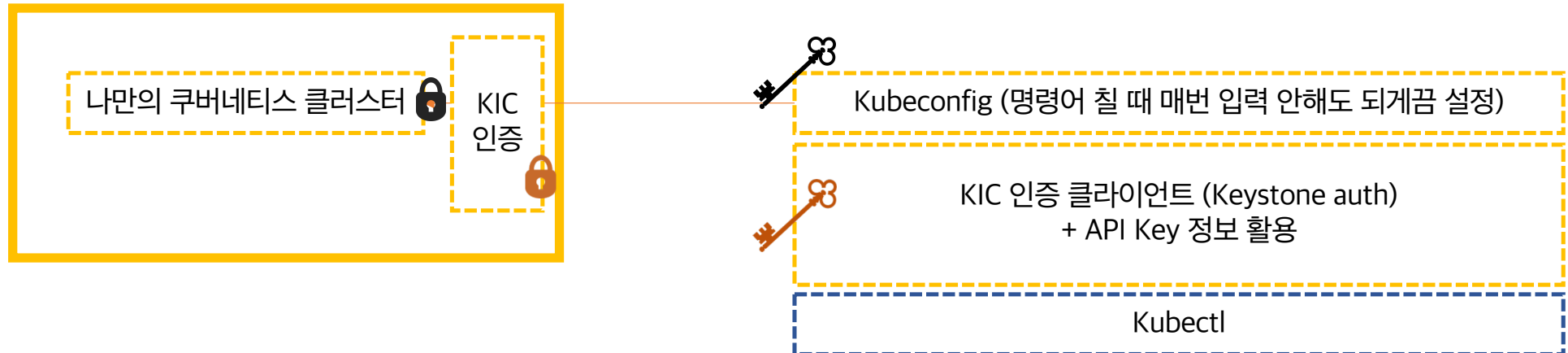
ike-cluster-member-osk-test-likelion-icloud

사용자 API 키 ID

사용자 API 키 정보 (선택)

osk-test 클러스터가 사용하는 API 키입니다. 삭제 시 클러스터의 노드 제어
가 제한됩니다.

kakaoenterprise

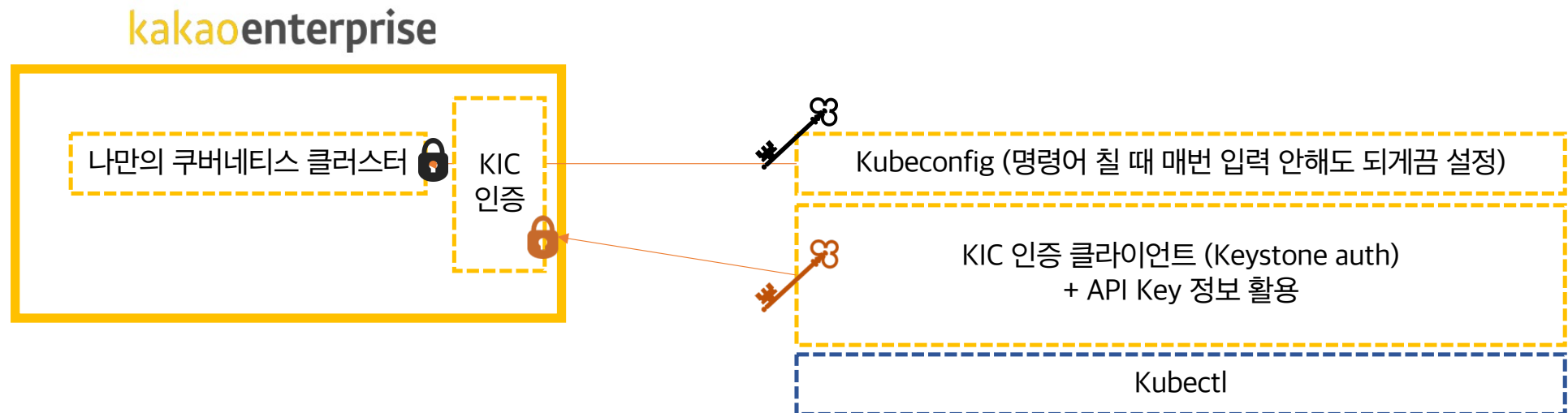


(참고) Kakao Kubernetes Engine 사용하기 - API 키/인증 설정

- 만들어진 API Key 설정하기

```
osk@osk-MacBook-Pro lklIn-kakao % export OS_AUTH_URL=https://keystone.kr-central-1.kakaoi.io/v3
osk@osk-MacBook-Pro lklIn-kakao % export OS_AUTH_TYPE=v3applicationcredential
osk@osk-MacBook-Pro lklIn-kakao % export OS_APPLICATION_CREDENTIAL_ID=각자 웹에서 확인
osk@osk-MacBook-Pro lklIn-kakao % export OS_APPLICATION_CREDENTIAL_SECRET=각자 웹에서 확인 (한번밖에 안보이므로 잘 기입)
osk@osk-MacBook-Pro lklIn-kakao % export OS_REGION_NAME=kr-central-1
osk@osk-MacBook-Pro lklIn-kakao %
osk@osk-MacBook-Pro lklIn-kakao % vi ~/.zshrc
```

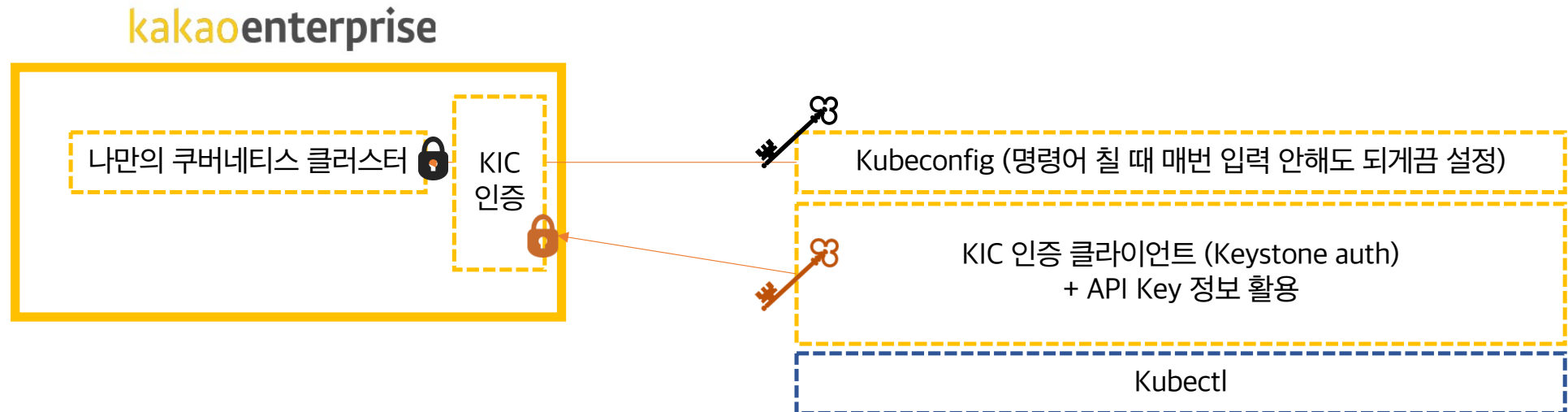
(재부팅시 편의를 위해 설정하는 단계임. .zshrc 파일을 열고 위에 export 부터 값까지 5줄 모두 복붙)



(참고) Kakao Kubernetes Engine 사용하기 - API 키/인증 설정

- 다운로드 인증 클라이언트에 실행 권한 부여 → (어떤 위치에서든 쉽게 실행가능토록) PATH 경로로 복사

```
osk@osk-MacBook-Pro lkl-kakao % pwd
/Users/osk/Desktop/lkl-kakao
osk@osk-MacBook-Pro lkl-kakao % chmod +x kic-keystone-auth
osk@osk-MacBook-Pro lkl-kakao %
osk@osk-MacBook-Pro lkl-kakao % cp kic-keystone-auth /usr/local/bin
osk@osk-MacBook-Pro lkl-kakao %
```



(참고) Kakao Kubernetes Engine 사용하기 - API 키/인증 설정

※ 맥에서 실행이 어려우면 Finder에서 Control 누르고 클릭

- <https://support.apple.com/ko-kr/guide/mac-help/mh40616/mac>



kakaoenterprise

