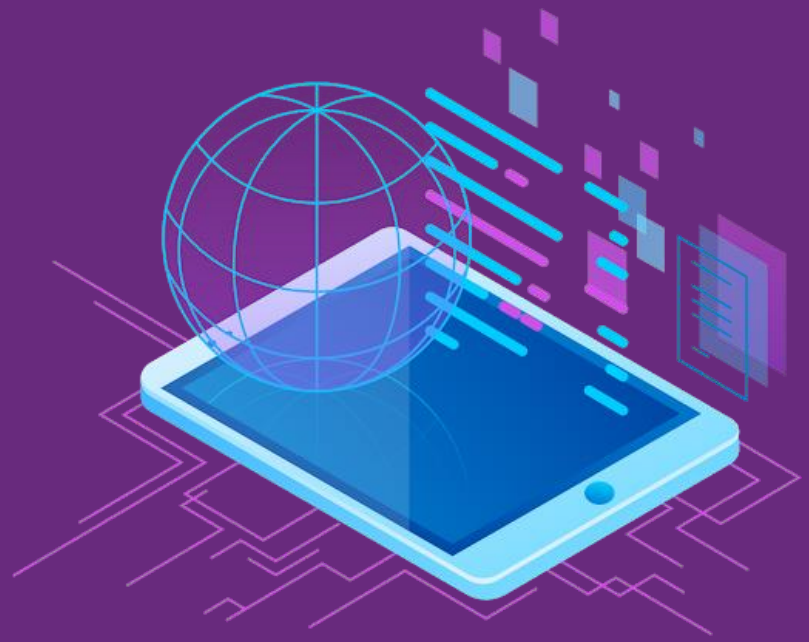
어플리케이션 컨테이너화,
디지털 트랜스포메이션으로 가는 첫걸음

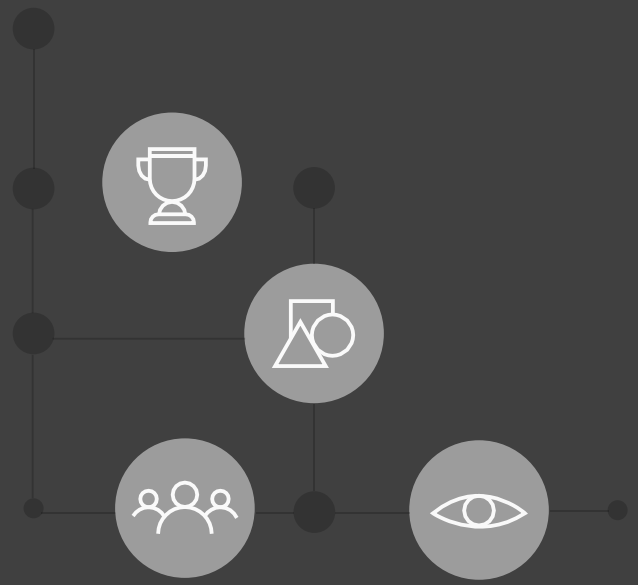


Cloocus

Gold Microsoft Partner Microsoft

Azure Expert MSP

Contents



01 Container

- Why Container?
- 기존의 어플리케이션 서비스 환경
- VM vs Container

02 Docker

- Docker란 무엇인가?
- Docker Image 만들기
- Dockerfile 이란?
- Dockerfile을 활용 Image 생성

03 Cloud Native로 가는 길

- Cloud Native Application
- Azure DevOps
- Azure Container Registry
- Azure AppService For Container
- Demo 시연

04 추가적인 Container 실행환경

- Azure Container Instance
- Azure Kubernetes Service

01 Container

Why Container?

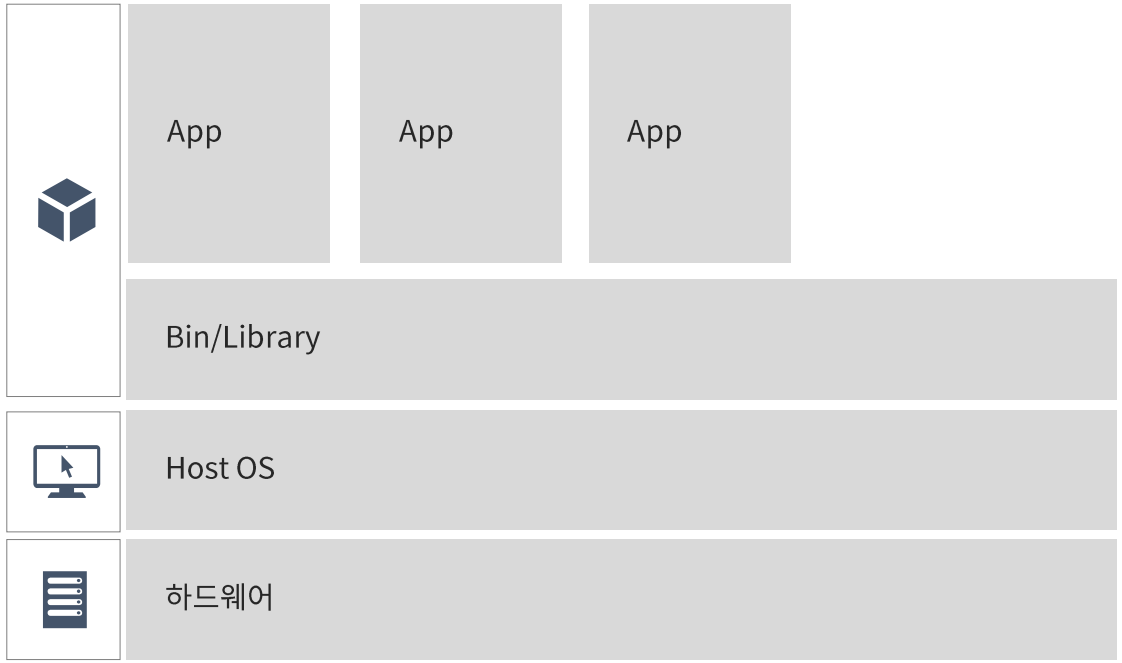
소비자	피드백이 빠른 서비스 최신의 기능이 포함된 서비스 다운타임이 없는 서비스
기업	가속화된 혁신 경쟁력 있는 빠른 서비스 출시 안정성/성능 확보

Why Container?

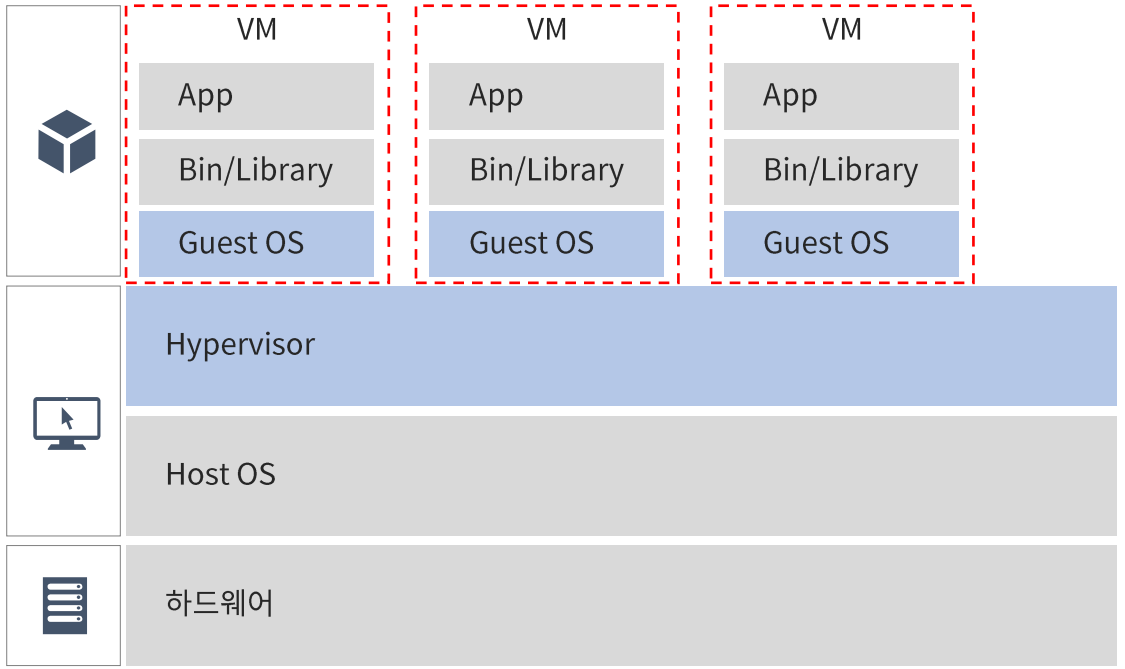
민첩성	표준화된 플랫폼에서 실행 배포 프로세스 간소화 향상된 앱을 더욱 빠르게 고객에게 제공
이식성	모든 패키지를 컨테이너에 보관 물리서버, 하이브리드, 클라우드 환경에 적합 일관된 앱 서비스 환경을 제공
신속한 확장성	컨테이너의 경량화 리소스 사용률이 효율적 신속한 확장 및 축소를 보장

기존의 어플리케이션 서비스 환경

Host OS 직접 설치

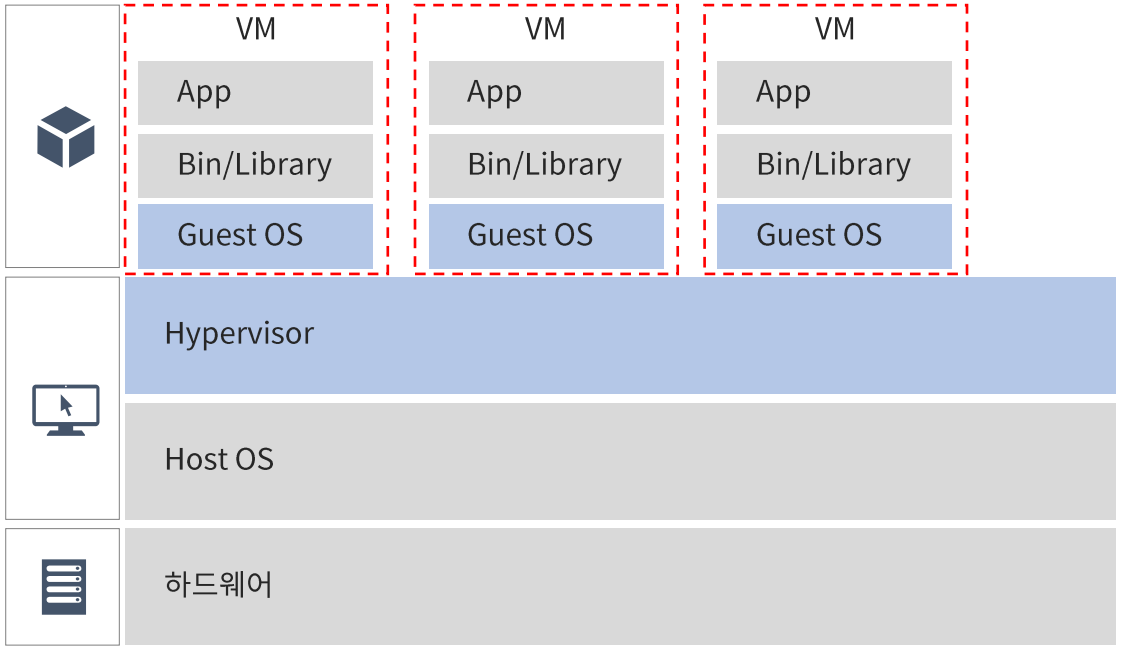


가상머신 – 하드웨어 가상화

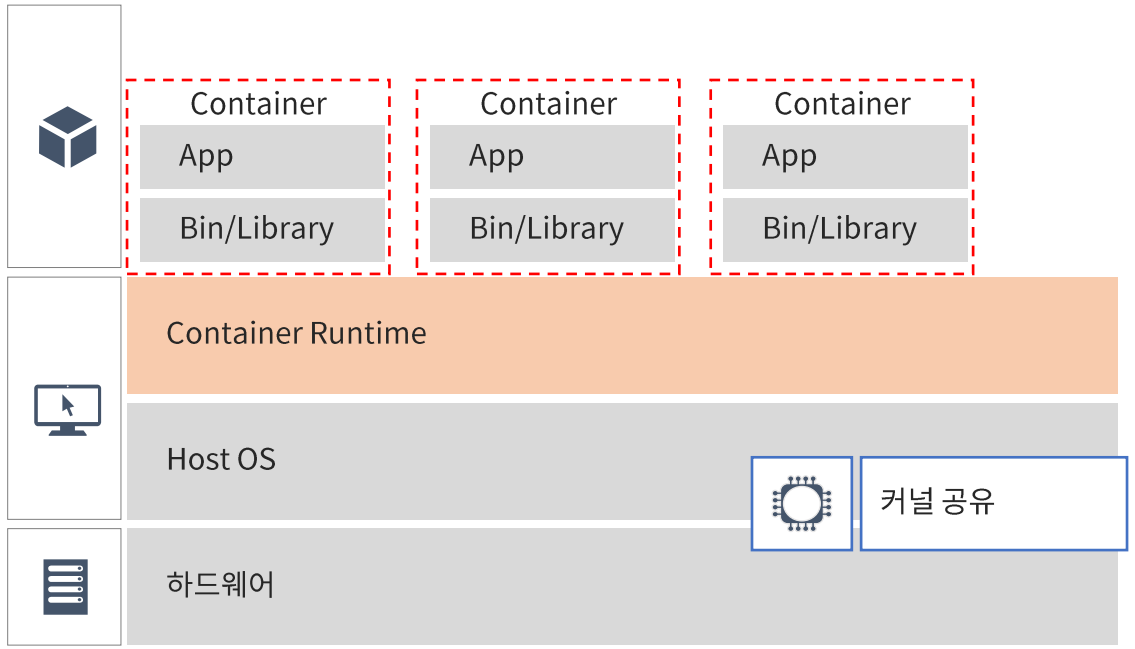


VM vs Container

가상머신 – 하드웨어 가상화



컨테이너 – 운영체제 가상화



02 Docker

Docker란 무엇인가?

1. Linux Container 기술 기반
2. Docker Engine / Hub / Compose 등을 포함
3. OpenVZ, LXC, cri-o보다 표준적
4. Linux / Windows / Mac등 다양한 OS에서 사용 가능
5. Container 환경의 표준화를 위하여 계속적으로 변화

Docker의 Image

1. Layer Image 사용한 컨테이너 실행

Docker Container 구조

```
root@webina-demo-vm1:~# docker inspect -f='{{json .RootFS.Layers}}' alpine | json_pp
[
   "sha256:1119ff37d4a9531330e3b8487863ee8ae0308337351be9d5f8bb38f80790acd9"
]
root@webina-demo-vm1:~# docker inspect -f='{{json .RootFS.Layers}}' imagelayertest:first | json_pp
[
   "sha256:1119ff37d4a9531330e3b8487863ee8ae0308337351be9d5f8bb38f80790acd9",
   "sha256:5caf298a48411a8612b858341ed2a3a0ddce08397534c3f0fa8e900ad7bcafe6"
]
root@webina-demo-vm1:~# docker inspect -f='{{json .RootFS.Layers}}' imagelayertest:second | json_pp
[
   "sha256:1119ff37d4a9531330e3b8487863ee8ae0308337351be9d5f8bb38f80790acd9",
   "sha256:5caf298a48411a8612b858341ed2a3a0ddce08397534c3f0fa8e900ad7bcafe6",
   "sha256:96f8e7b7c543aaf8310c1cf00e341d9b60ff132ed9d5c097c00c70aa62f1261c"
]
```

Docker Container 생성 및 실행

```
root@webina-demo-vm1:~# docker run -it alpine
Unable to find image 'alpine:latest' locally
latest: Pulling from library/alpine
4c0d98bf9879: Pull complete
Digest: sha256:08d6ca16c60fe7490c03d10dc339d9fd8ea67c6466dea8d558526b1330a85930
Status: Downloaded newer image for alpine:latest
/ #
```

1. docker run –it alpine 명령어 실행
2. docker hub에서 이미지 Pulling
3. Container Create → Container Start → -it option으로 Container 내부 쉘 환경 실행
4. Ctrl + p + q로 호스트 창으로 이동

Docker Container 생성 및 실행

```
root@webina-demo-vm1:~# docker image ls
REPOSITORY   TAG             IMAGE ID       CREATED         SIZE
alpine       latest          e50c909a8df2   2 weeks ago     5.61MB
nginx        1.19.6-alpine   629df02b47c8   8 weeks ago     22.3MB
centos       7               8652b9f0cb4c   3 months ago    204MB
ubuntu       14.04           df043b4f0cf1   4 months ago    197MB
root@webina-demo-vm1:~# docker ps
CONTAINER ID   IMAGE     COMMAND     CREATED          STATUS          PORTS     NAMES
3b732d328aec   alpine    "/bin/sh"   34 seconds ago   Up 33 seconds             gracious_heyrovsky
root@webina-demo-vm1:~# docker attach gracious_heyrovsky
/ #
```

1. docker image ls → 다운받은 image 확인
2. docker ps 명령어로 → 컨테이너 프로세스 확인
3. docker 실행시 container 이름을 명시하지 않았으므로 임의의 Name이 지정
4. docker attach <container_name> → 컨테이너 내부로 다시 진입

Docker Container App 설치

```
/ # apk add nodejs npm
(1/8) Installing ca-certificates (20191127-r5)
(2/8) Installing nghttp2-libs (1.42.0-r1)
(3/8) Installing brotli-libs (1.0.9-r3)
(4/8) Installing c-ares (1.17.1-r1)
(5/8) Installing libgcc (10.2.1_pre1-r3)
(6/8) Installing libstdc++ (10.2.1_pre1-r3)
(7/8) Installing nodejs (14.15.4-r0)
(8/8) Installing npm (14.15.4-r0)
Executing busybox-1.32.1-r2.trigger
Executing ca-certificates-20191127-r5.trigger
OK: 71 MiB in 22 packages
/ # apk add git
(1/4) Installing libcurl (7.74.0-r0)
(2/4) Installing expat (2.2.10-r1)
(3/4) Installing pcre2 (10.36-r0)
(4/4) Installing git (2.30.1-r0)
Executing busybox-1.32.1-r2.trigger
OK: 82 MiB in 26 packages
/ # git clone https://bluecwon@dev.azure.com/bluecwon/dockerWebinaDemo/_git/webinaDemo
Cloning into 'webinaDemo'...
remote: Azure Repos
remote: Found 31 objects to send. (55 ms)
Unpacking objects: 100% (31/31), 200.06 KiB | 2.63 MiB/s, done.
/ # cd webinaDemo/front-end/
/webinaDemo/front-end # npm install
```

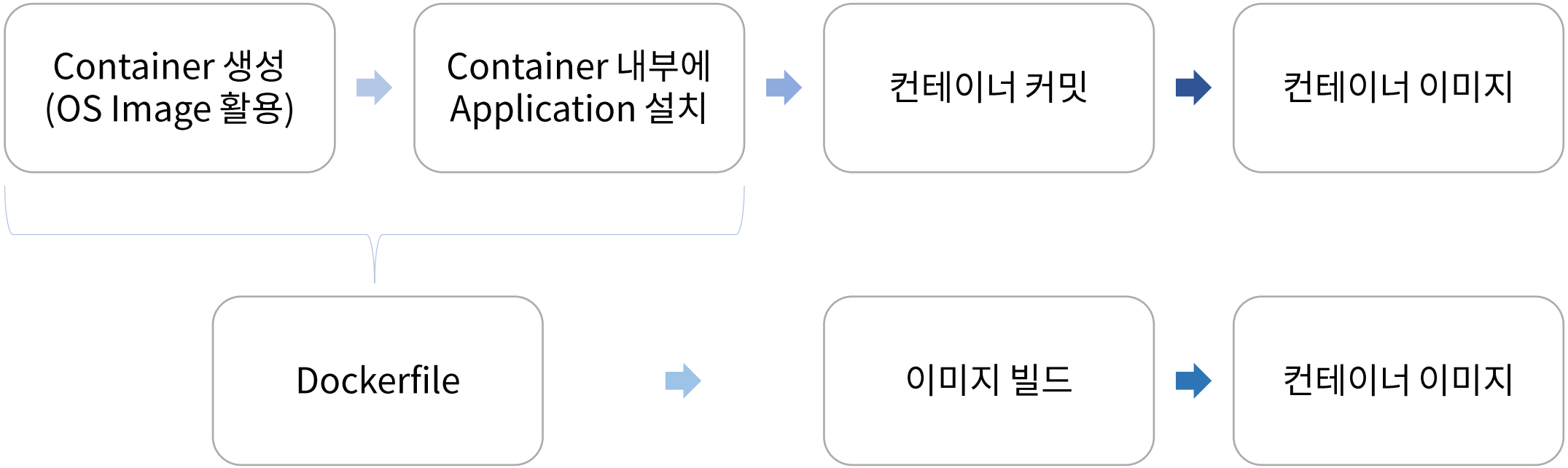
1. 앱 실행에 필요한 패키지 설치
2. 소스코드를 받기위해 git 설치
3. 소스코드 다운로드
4. 라이브러리 설치

Docker Container Image 만들기

```
root@webina-demo-vm1:~# docker ps -a
CONTAINER ID   IMAGE     COMMAND     CREATED          STATUS                      PORTS     NAMES
3b732d328aec   alpine    "/bin/sh"   5 minutes ago    Exited (0) 47 seconds ago             gracious_heyrovsky
root@webina-demo-vm1:~# docker commit -a bluecwon -m "api server" gracious_heyrovsky api_server:1
sha256:64ef5317f61e5f798ab1c95578dc3883c52534229fb0931bb38edcfc44c789c5
root@webina-demo-vm1:~# docker image ls
REPOSITORY   TAG              IMAGE ID       CREATED          SIZE
api_server   1                64ef5317f61e   20 seconds ago   78.5MB
alpine       latest           e50c909a8df2   2 weeks ago      5.61MB
nginx        1.19.6-alpine    629df02b47c8   8 weeks ago      22.3MB
centos       7                8652b9f0cb4c   3 months ago     204MB
ubuntu       14.04            df043b4f0cf1   4 months ago     197MB
```

1. 호스트 머신으로 나와 docker commit 명령어 실행
   - -a : 작성자
   - -m: commit message
2. Image가 제대로 만들어 졌는지 docker image ls 명령어로 확인

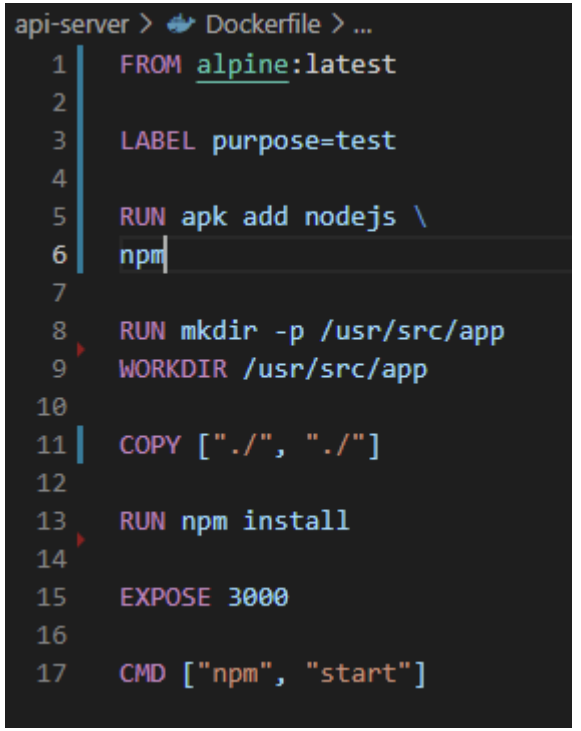
Dockerfile 이란?



1. Application을 컨테이너화 할 때 일련의 과정을 관리하는 파일
2. 직접 컨테이너를 생성하고 이미지를 커밋 해야하는 과정 생략, 빌드 및 배포 자동화를 지원

Dockerfile의 주요 명령어

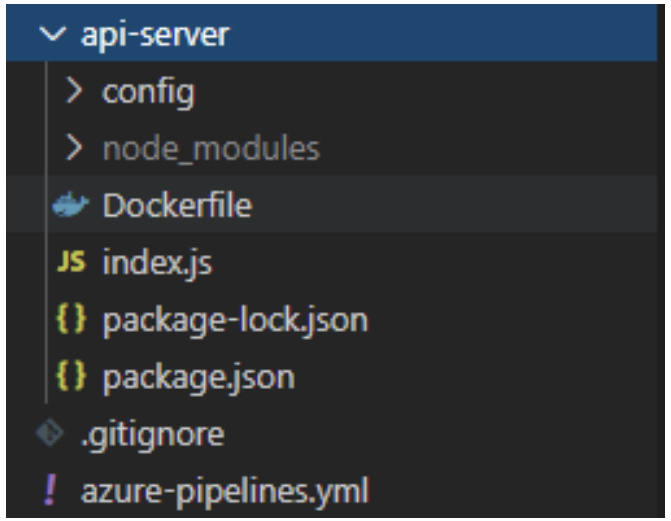
1. FROM : 빌드 프로세스를 시작할 때 사용할 기본 이미지를 정의
2. LABEL : 이미지의 메타데이터를 key=value 형태로 지정
3. COPY : Host 서버의 파일이나 디렉토리를 특정 이미지 경로에 복사
4. ADD : (COPY) 명령어 보다 기능이 추가됨
   4-1. 경로에 URL 사용 가능
   4-2. 압축파일의 경우, 자동으로 압축 해제 후에 복사함
5. RUN : 이미지가 빌드 되는 과정에서 실행되는 내용을 작성함
6. EXPOSE: 컨테이너 내부의 포트를 Open 추후 호스트 포트와 연결
7. CMD : 컨테이너를 생성한 이후 실행되는 명령어 지정
   6-1. shell 없이 실행할 경우 매개 변수를 설정한다.
   6-2. docker run 명령어가 우선 순위를 갖는다.
7. ENTRYPOINT : 생성된 컨테이너를 초기화 하면서 실행되는 스크립트 지정
   7-1. 컨테이너 생성시 반드시 지정할 명령을 수행
   7-2. CMD와 같이 사용할 수 있다.



(참고) Dockerfile 예문

Dockerfile로 이미지 빌드

```
root@webina-demo-vm1:~# docker build -t api_server:file ./webinaDemo/api-server/
Sending build context to Docker daemon  27.65kB
Step 1/9 : FROM alpine:latest
 ---> e50c909a8df2
Step 2/9 : LABEL purpose=test
 ---> Running in b1933bfefc35
Removing intermediate container b1933bfefc35
 ---> 25596e64b71d
Step 3/9 : RUN apk add nodejs npm
 ---> Running in ed0bc2b27938
fetch https://dl-cdn.alpinelinux.org/alpine/v3.13/main/x86_64/APKINDEX.tar.gz
fetch https://dl-cdn.alpinelinux.org/alpine/v3.13/community/x86_64/APKINDEX.tar.gz
(1/8) Installing ca-certificates (20191127-r5)
(2/8) Installing nghttp2-libs (1.42.0-r1)
(3/8) Installing brotli-libs (1.0.9-r3)
(4/8) Installing c-ares (1.17.1-r1)
(5/8) Installing libgcc (10.2.1_pre1-r3)
(6/8) Installing libstdc++ (10.2.1_pre1-r3)
(7/8) Installing nodejs (14.15.4-r0)
(8/8) Installing npm (14.15.4-r0)
Executing busybox-1.32.1-r2.trigger
Executing ca-certificates-20191127-r5.trigger
OK: 71 MiB in 22 packages
Removing intermediate container ed0bc2b27938
 ---> 15cedbd587b9
Step 4/9 : RUN mkdir -p /usr/src/app
```



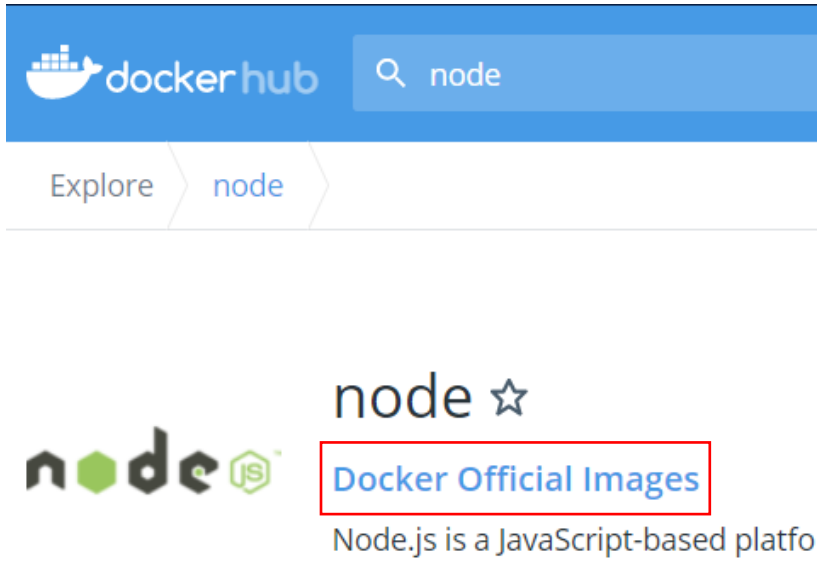
(참고) Dockerfile 위치

1. docker build –t <이미지네임:태그> <dockerfile 위치>

Dockerfile로 이미지 빌드

```
root@webina-demo-vm1:~# docker inspect -f='{{json .RootFS.Layers}}' api_server:file | json_pp
[
   "sha256:1119ff37d4a9531330e3b8487863ee8ae0308337351be9d5f8bb38f80790acd9",
   "sha256:eaf722e5c1d9c17b8535e886f8a6b7e6085c4d7ec84ea1ec59f6f138e077f4eb",
   "sha256:097f7658d427c705e36a42f94a2f9e42bd99e72f37a6d7bb922ef9f9da0043e7",
   "sha256:f6bd55247bf6b0fc75b73d43ba4a39d289beb566429ae5ef56f303bef2410ca8",
   "sha256:466dc7958b4242a3f4e8d1c99d2f6d4e30373291e06f3a20bb3680735f45a250"
]
root@webina-demo-vm1:~# docker inspect -f='{{json .RootFS.Layers}}' alpine:latest | json_pp
[
   "sha256:1119ff37d4a9531330e3b8487863ee8ae0308337351be9d5f8bb38f80790acd9"
]
root@webina-demo-vm1:~# docker image ls
REPOSITORY   TAG              IMAGE ID       CREATED         SIZE
api_server   file             f4f8d731dc19   2 minutes ago   66.5MB
api_server   1                64ef5317f61e   21 hours ago    78.5MB
mysql        5.7              5f47254ca581   5 days ago      449MB
alpine       latest           e50c909a8df2   2 weeks ago     5.61MB
nginx        1.19.6-alpine    629df02b47c8   8 weeks ago     22.3MB
centos       7                8652b9f0cb4c   3 months ago    204MB
ubuntu       14.04            df043b4f0cf1   5 months ago    197MB
node         12.2.0-alpine    f391dabf9dce   21 months ago   77.7MB
```

Dockerfile 사용시 주의사항



1. FROM을 통해 Base Image는 Official Image를 사용
2. Build context를 지정할 시 dockerfile의 위치 확인
3. RUN, COPY등의 각 명령어 마다 Layer가 생성됨
4. ADD 명령어의 경우 압축파일이나 URL보다는 Host에서 직접 Container에 파일을 추가를 권장
5. dockerfile로 빌드를 진행할 시 cache를 사용해 빌드 속도 향상

   주의) dockerfile의 내부에서 git clone등을 사용할 시 cache의 이유로 기존의 코드를 그대로 사용할 가능성이 존재
6. build 실행시 –-no-cache 옵션을 활용

   소스코드의 경우 Host에서 COPY 하는 방법을 권장

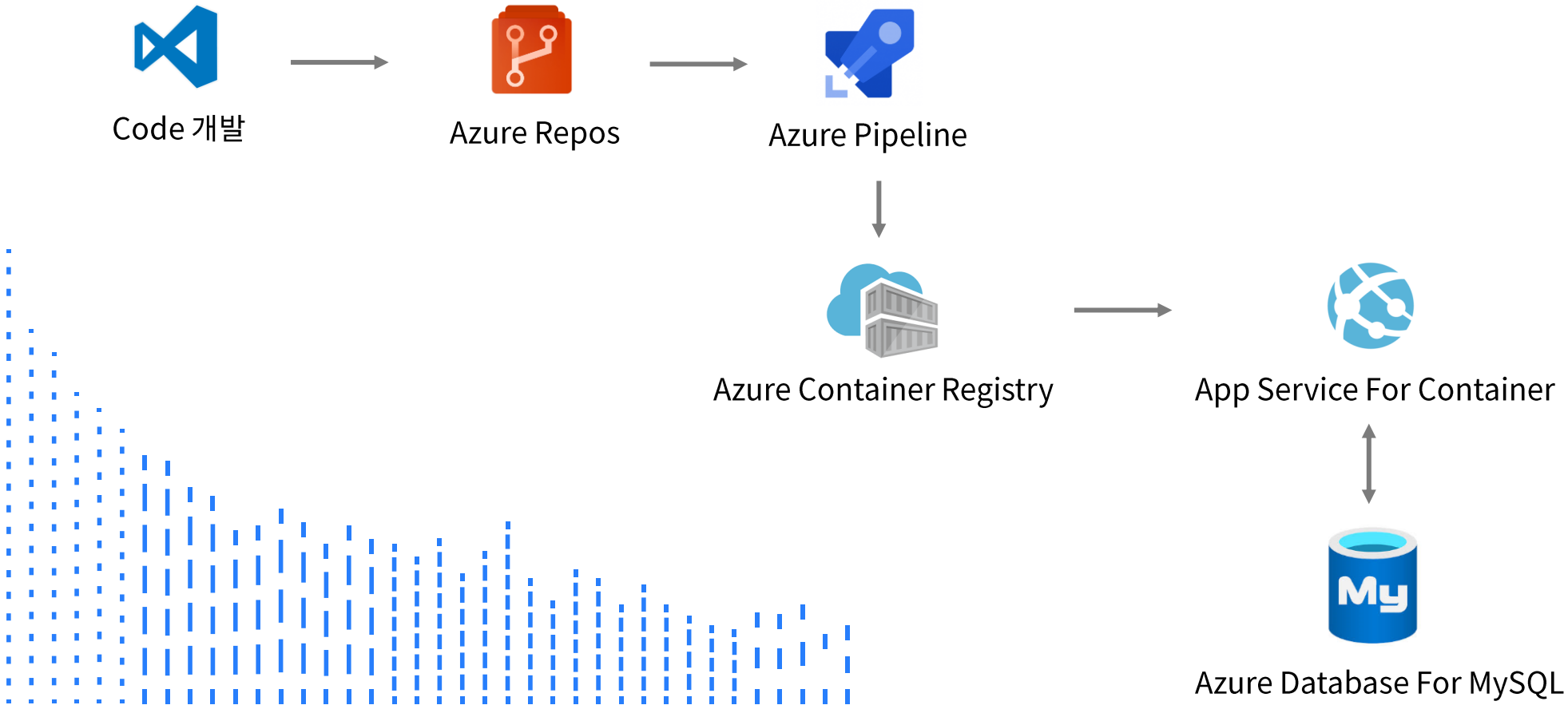
Docker 참고사항

1. Docker Docs: https://docs.docker.com/

2. Docker Hub: https://hub.docker.com/

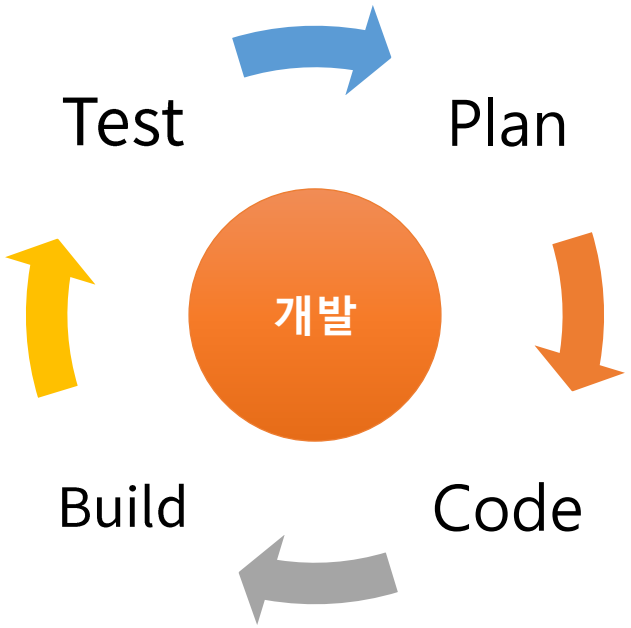
03 Cloud Native

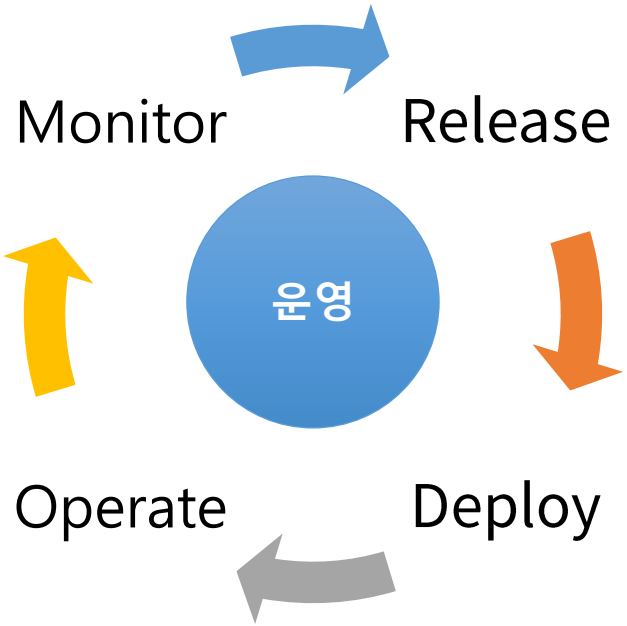
Container를 활용한 Cloud Native Application



Azure DevOps

DevOps는 소프트웨어 개발(Dev)과 정보 기술 운영(Ops)을 결합해 시스템 개발 수명 주기를 단축하는 동시에 기능, 수정, 업데이트를 비즈니스 목표에 근접하게 빠른 주기로 제공하는 소프트웨어 개발 수행 방법이다.





Azure DevOps

Azure DevOps

https://azure.com/devops

Azure Boards

신속하게 작업 계획 및 논의

Azure Repos

진보된 파일 관리 서비스 제공

Azure Pipelines

CI/CD 지원 지속적인 배포 실행

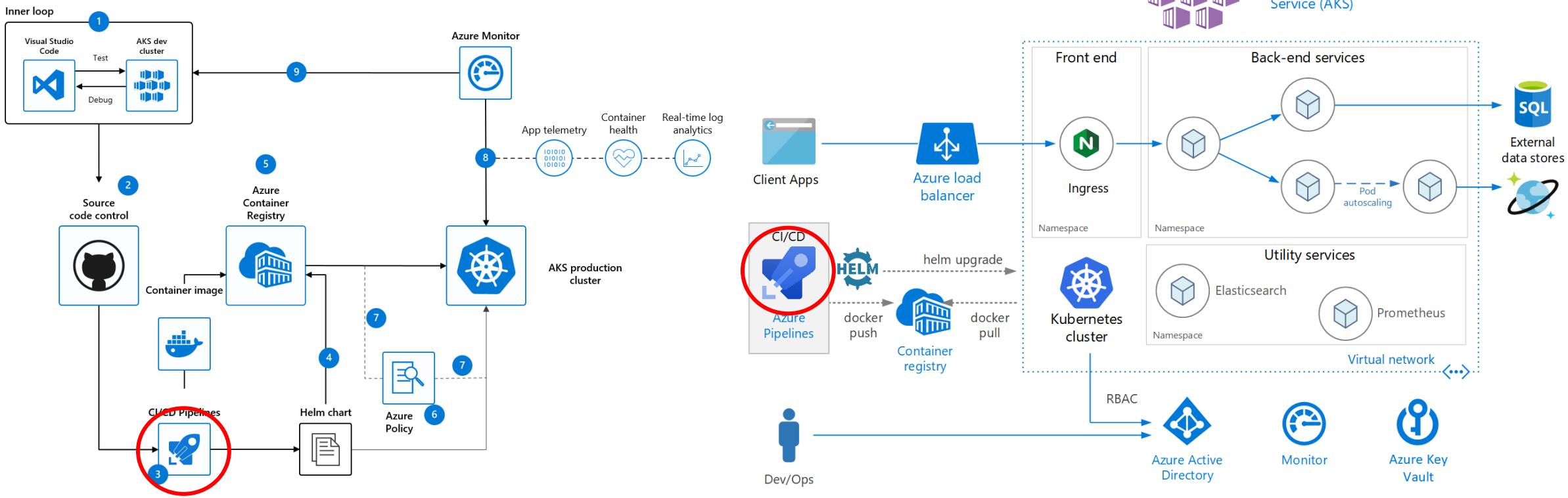
Azure Test Plans

테스트 관리 도구로 제품 신뢰성 향상

Azure Artifacts

패키지 생성, 호스트, 공유 수월하게 아티팩트 추가

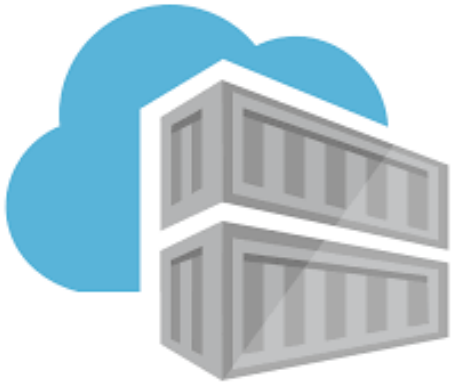
Azure DevOps



Azure Container Registry

[주요 기능]
- Docker Image Upload
- Private Docker Registry
- Azure Active Directory 인증
- Image Version 관리

[특징]
- Docker Registry 2.0 오픈 소스 기반
- TLS 지원
- Azure DevOps와 통합



Azure App Service For Container

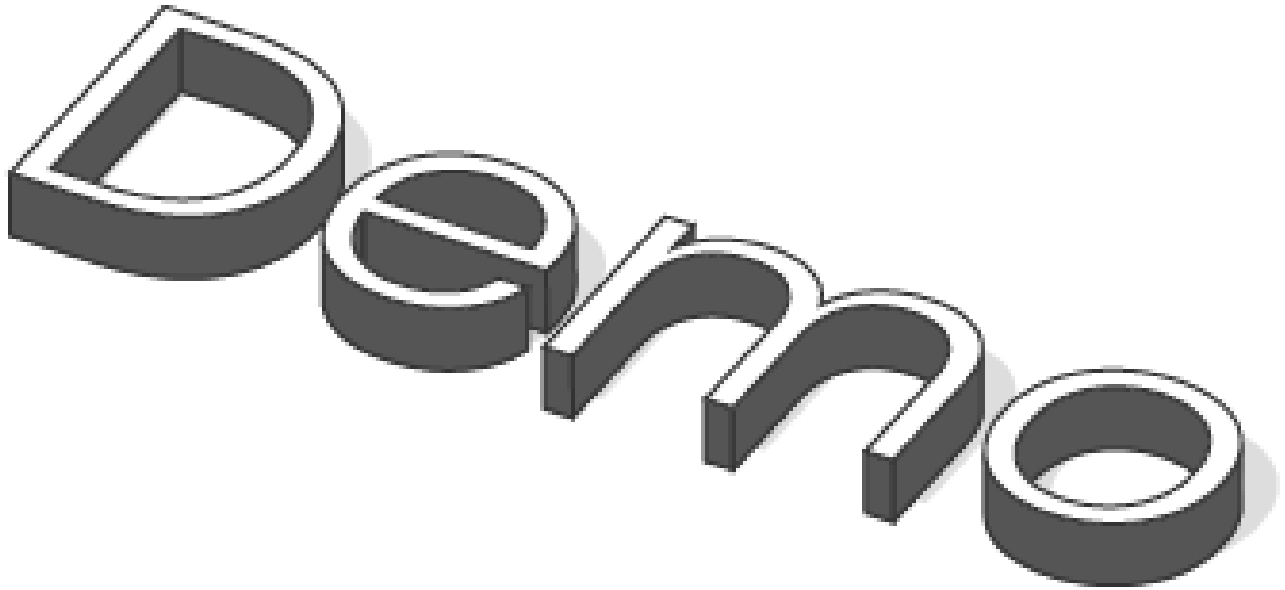
[주요 기능]

- App Services 에서 코드 배포
- Container로 배포 가능
- 데이터 입출력
- Azure Service에 대한 바인딩

[특징]

- Serverless
- Scale In / Out
- Github, Azure DevOps, BitBucker, Docker hub, Azure Container Registry 및 기타 개발 도구
- 사용량 단위 과금

Demo 시연



04

Container 실습하기

Azure Container Instance

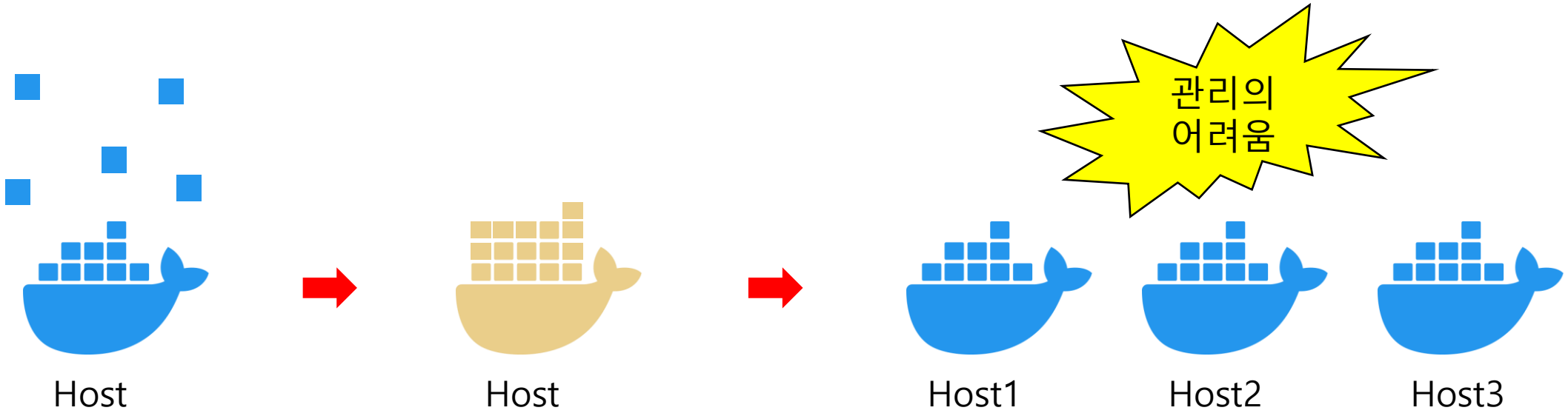
[주요 기능]

- Serverless의 Container 실행 환경
- 수 초 안에 컨테이너 실행
- OS 이미지를 캐시하여 배포 속도를 향상
- 내부 통신이 가능한 컨테이너들을 Group으로 배포

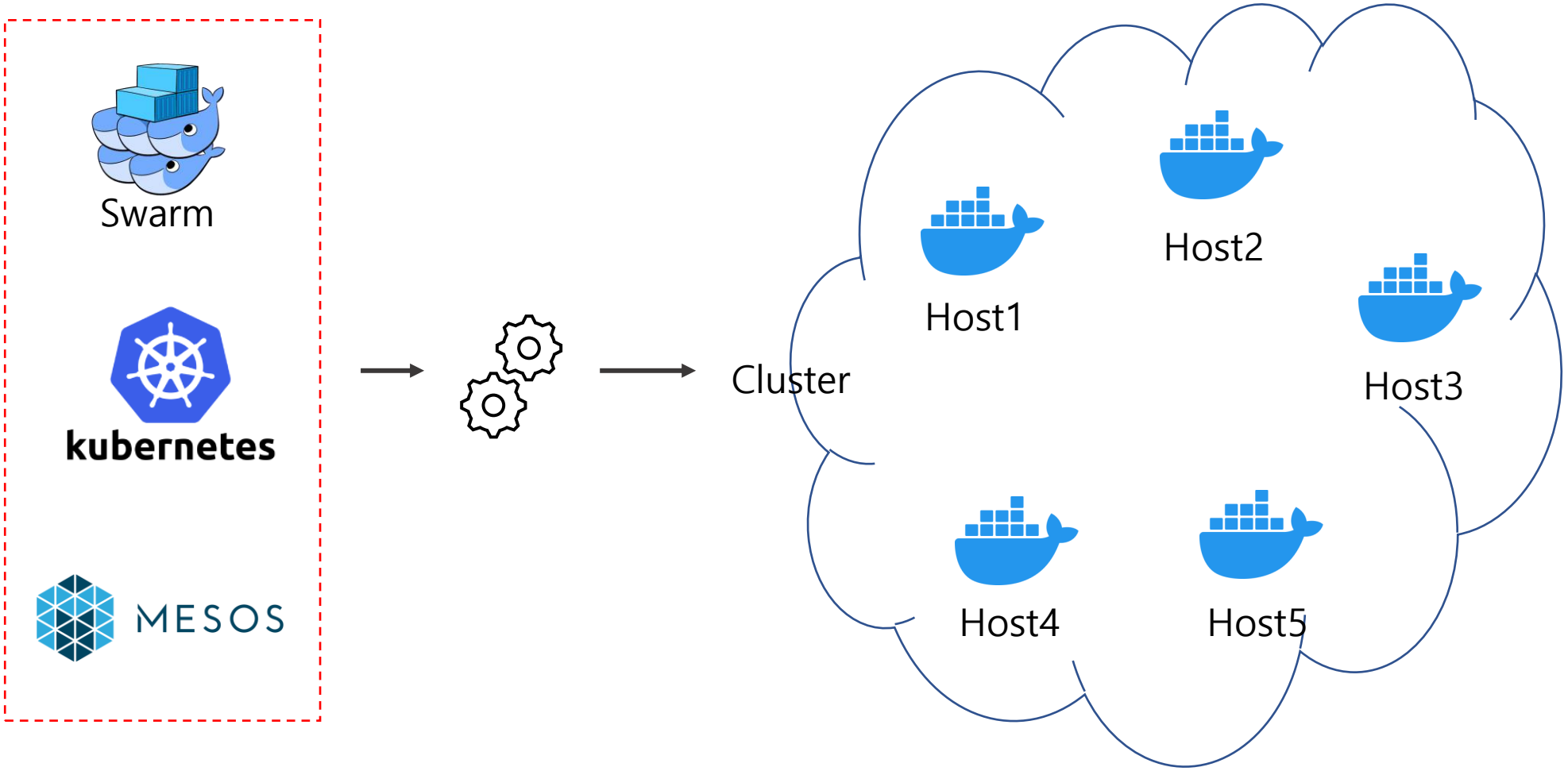
[특징]

- 애플리케이션, 작업 자동화 및 빌드 작업
- 99.9%의 SLA, 영구 스토리지(Azure Files 탑재), Vnet 배포
- 하이퍼바이저 격리를 사용

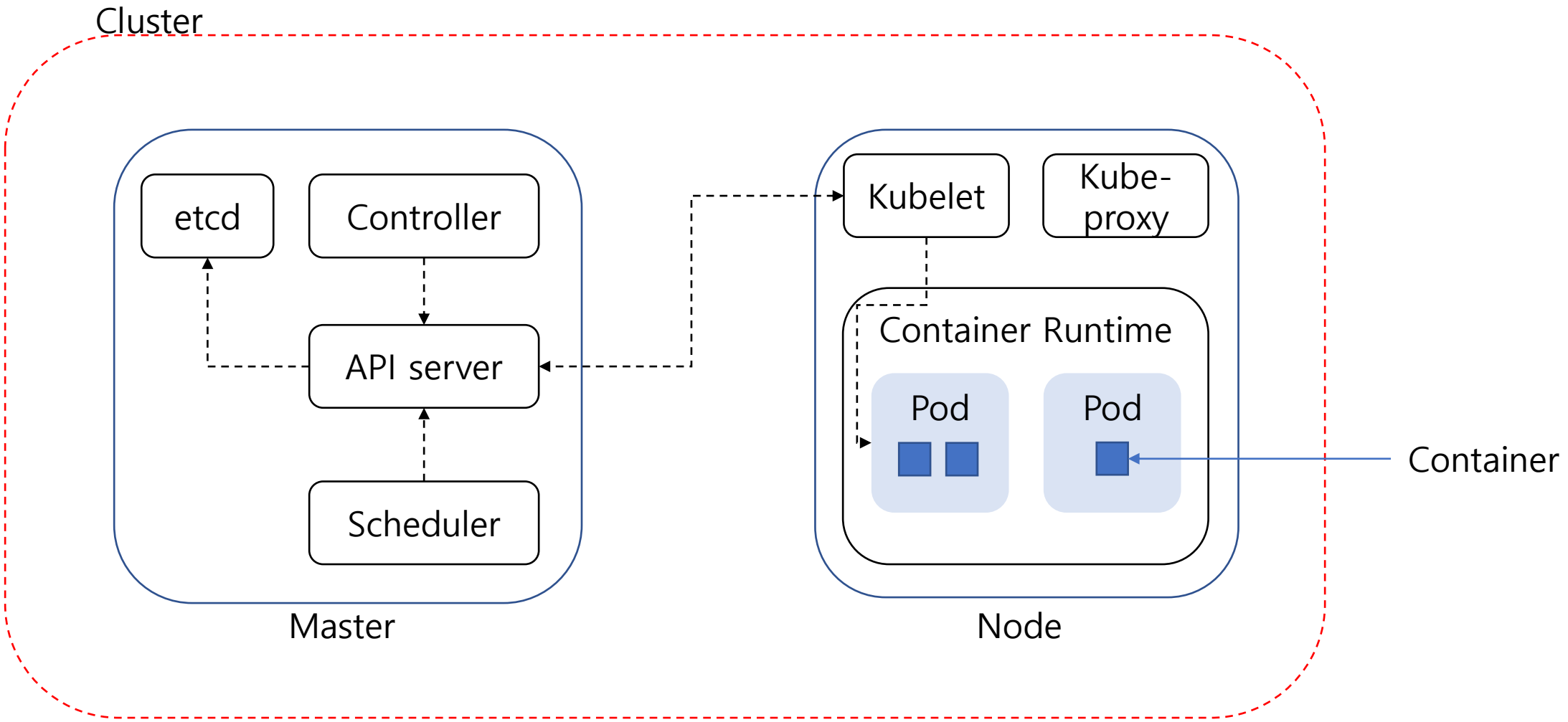
Kubernetes란?



Kubernetes란?



Kubernetes란?



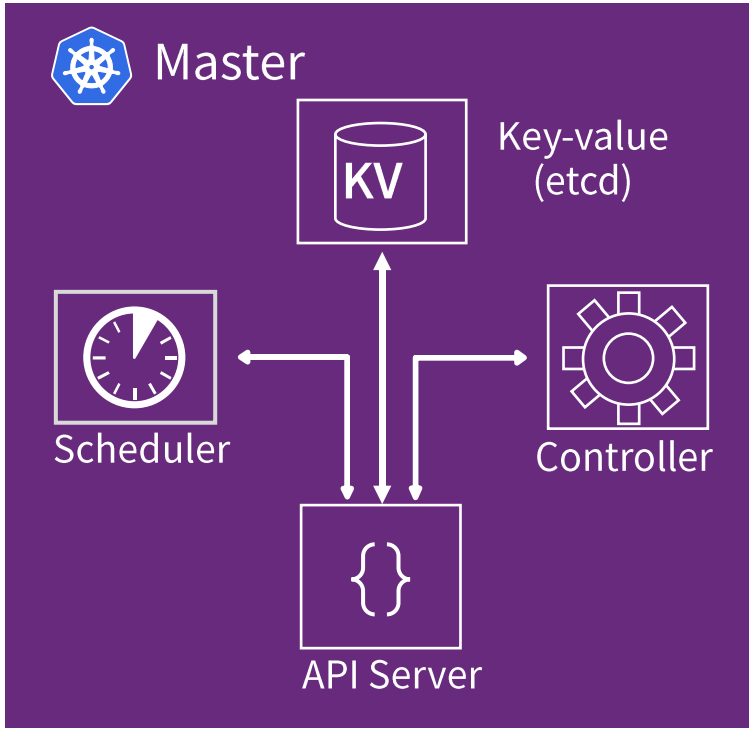
Master

클러스터 저장소(Cluster Store)

1. 영구적인 저장소
2. Cluster 상태와 구성이 지속적으로 저장된다.
3. Etcd 를 사용한다.

   Go 언어와 Raft 프레임 워크로 작성된 오픈소스 key-value 저장소
4. 저장소에 대한 백업 계획이 필요하다.

   AKS 에서는 Azure에서 저장소에 대한 백업을 관리



컨트롤러(Kube-controller-manager)

1. 변경 사항을 모니터링 한다.
2. 클러스터의 현재 상태를(desired state) 지속적으로 확인한다.
3. 컨트롤러는 노드, 엔드포인트, 네임스페이드 컨트롤러 등이 있다.

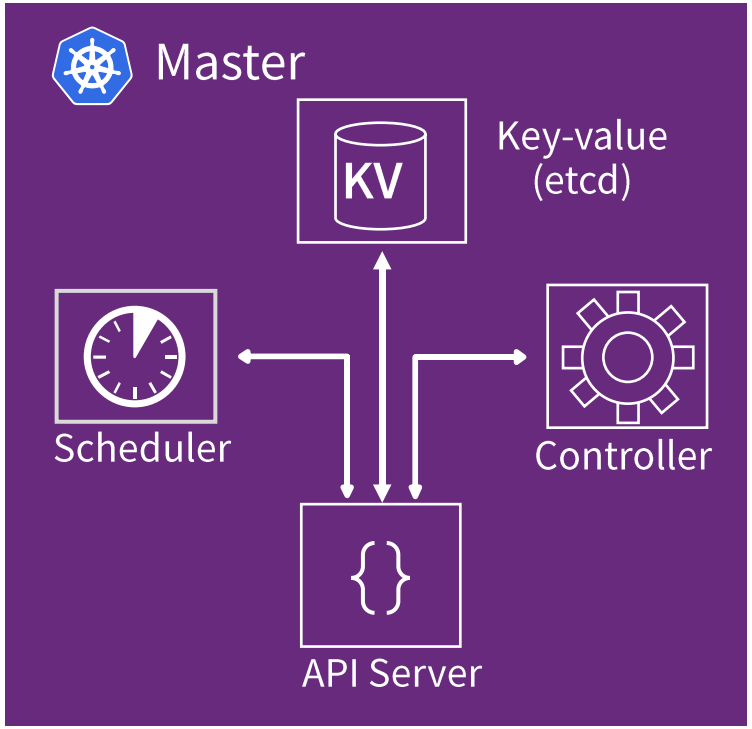
Master

스케줄러 (kube-scheduler)

1. 새로운 pod를 생성하기 위해서 apiserver를 감시한다.
2. 작업을 node에 할당한다.

API server

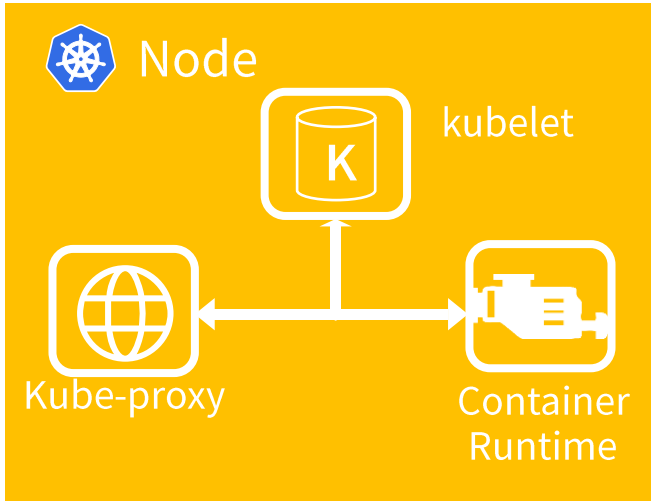
1. Master 서버는 API server 를 통해서만 외부와 통신한다.
2. AKS 에서 API server 를 제외한 Master의 다른 구성요소는 관리자에게 노출되지 않는다.
3. API server 는 기본적으로 443 포트를 사용한다.



Node

Kubelet

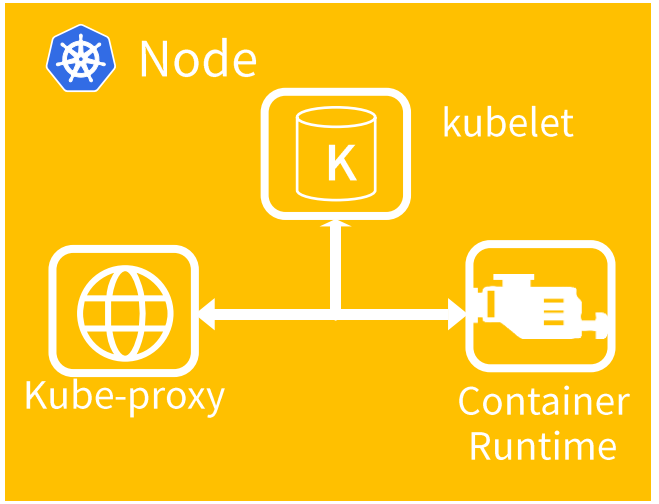
1. Kubernetes Agent 역할을 수행한다.
2. Kubernetes Cluster에 host 가상머신을 Node로 등록시킨다.
3. 작업을 할당 받기 위해 Master의 API sever을 지속적으로 체크한다.
4. Kubelet이 작업을 수행할 수 없을 때, Master에게 보고한다.
5. Kubelet이 작업을 수행할 수 없을 때, Master가 대신 결정한다.
6. Node에서 pod가 실패하면, Master에 상태를 보고한다.



Node

컨테이너 런타임(컨테이너 엔진)

1. 이미지 pulling
2. 컨테이너 시작 또는 중지 작업
3. 일반적으로 네이티브 Docker API 사용
4. AKS의 경우 v1.19 이전에는 Moby를 사용하였고 현재 Containerd를 Preview로 제공하고 있습니다.



Kube-proxy

1. 각 노드에서 실행되는 proxy 이다.
2. pod간 로드 밸런싱을 합니다.

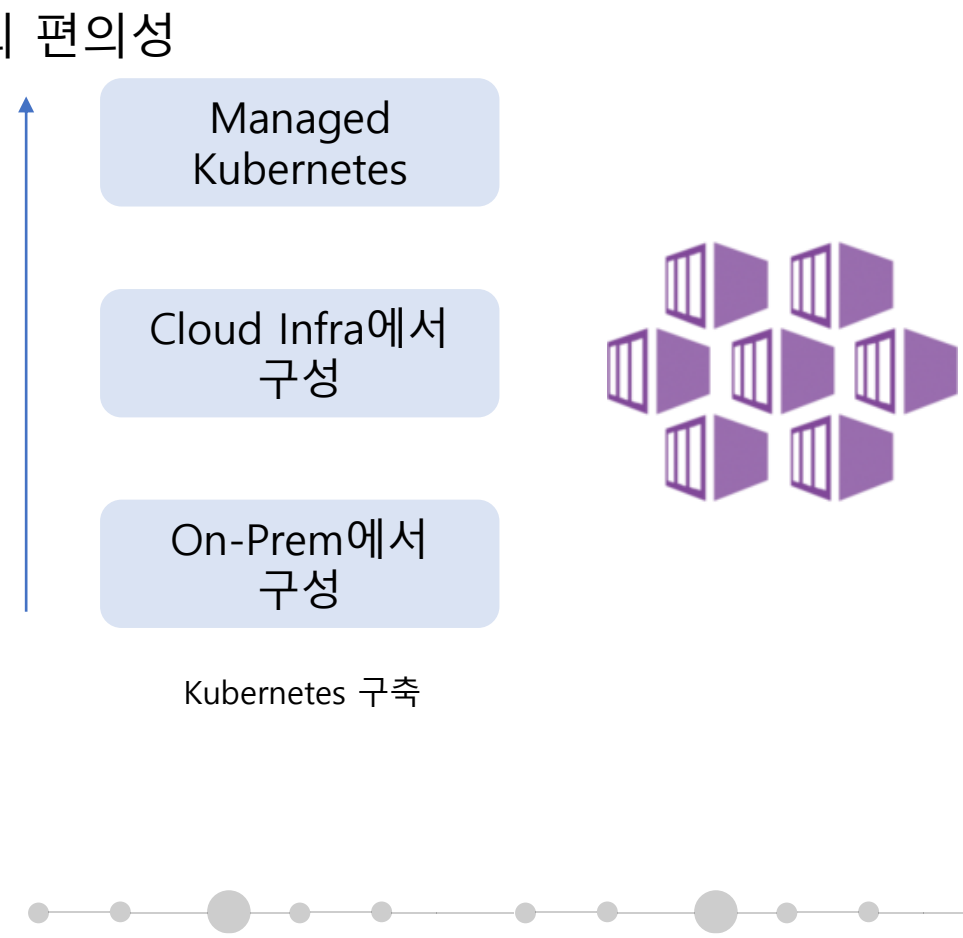
Azure Kubernetes Service

[주요 기능]

- Kubernetes (Kubectl) 지원
- RBAC 기반 엑세스 제어
- Azure Virtual Network와 통합
- Log Management(Log Analytics)
- Storage Volume 지원
- HTTP Application Routing(Ingress Controller)

[특징]

- Master 서버를 Azure에서 관리(무료)
- 클러스터 내의 노드 운영 비용만 지불
- Azure DevOps와 통합
- 가상노드를 지원



Summary

Application Container화	Container를 처음 사용하신다면 OS 컨테이너를 실행 후 어플리케이션 패키지를 설치해 테스트를 권장 Container화 진행시 Micro Service Architecture를 고려하시기 바랍니다.
Container 운영 환경	DevOps Pipeline을 활용한 CI/CD 구성으로 비즈니스 속도를 올려 보기 HTTP 기반의 소규모 서비스를 운영하신다면 Azure App Service 권장 2~3개의 컨테이너로 빠른 서비스 환경 구축을 원하신다면 ACI 권장 엔터프라이즈급의 대규모 컨테이너 운영을 원하신다면 AKS 권장

Thank you

Cloocus

Gold Microsoft Partner

Azure Expert MSP