보안 원격 접근 및 CI/CD를 위한 도커 설정

최 혁

도커 API의 엔드포인트 형태

- 도커 엔진은 로컬 컴퓨터와 연결된 채널을 주시하도록 초기 설정되어 있다.
- 리눅스 소켓이나 윈도의 명명 파이프가 로컬 채널로 쓰이는데 이 두 가지는 모두 트래픽의 범위가 로컬 컴퓨터 내부로 제한된다.
- 도커 엔진을 원격으로 접근하려면 설정을 해야 하는데 그 중 HTTP를 통한 접근이 가장 간단하지만, 보안에 취약하다.
- (참고로 crul로도 도커 API를 호출할 수 있다)

보안 원격 접근을 위한 도커 엔진 설정

- HTTPS 프로토콜의 디지털 인증서와 같은 방식으로 암호화하여 인증서로 자신을 증명하고 전송되는 내용을 암호화한다.
- 상호 TLS는 널리 사용되는 방법이지만 인증서를 생성하고 교체하는 관리 업무에서 오버헤드가 발생한다.
- SSH 프로토콜로 사용자명과 비밀번호 혹은 비밀키를 통해 암호화한다.
- SSH 프로토콜의 장점은 도커 명령행 도구가 표준 SSH 클라이언트를 사용하기에 도커 엔진 쪽에서 설정을 변경할 필요도 없고, 인증서를 생성할 필요도 없다.
- 두 방법 전부 보안 접속에 대한 권한 조정 기능이나 감시 기능은 없기에 특정 사용자에게 특정 명령의 사용 권한을 지정하거나 어떤 사용자가 어떤 작업을 했는지 추적할 수 없다.

도커 컨텍스트를 사용해 원격 엔진에서 작업하기

- 도커 컨텍스트를 사용하면 도커 엔진에 쉽게 접근할 수 있다.
- 도커 컨텍스트는 도커 명령행 도구에서 원격 접근에 필요한 모든 상세 정보를 지정하면 생성할 수 있다.
- 컨텍스트에는 로컬 엔진이나 원격 엔진 간에 대상을 전환하기 위해 필요한 모든 정보가 들어간다.

```
# 대상 서버 내 도커 엔진의 도메인과 인증서로 컨텍스트 생성
docker context create pwd-tls --doocker "인증서.pem", key="key.pem"

# 컨텍스트 목록 확인
# 컨텍스트 목록을 보면 내부 채널로 접근하는 로컬 컴퓨터의 도커 엔진을 가리키는 기본 컨텍스트를 볼 수 있다.
docker context ls
```

정리

- 도커 엔진의 보안은 명령행 도구와 API 사이의 통신을 암호화하는 것과 허가받은 사용자만 API에 접근할 수 있도록 하는 것이다.
- 하지만 권한을 조정하는 기능은 없다.
- 쿠버네티스에는 도커 엔터프라이즈와 마찬가지로 역할 기반 접근 제어 모델이 있어 어떤 사용자가 접근할 수 있는 리소스가 무엇이고 이들 리소스에 어떤 작업을 수행할 수 있는지 지정할 수 있다.
- 풀링 기반 모델을 사용하여 클러스터에서 새 빌드가 승인됐는지 여부를 파악해 업데이트를 스스로 배포하는 GitOps적인 접근법도 있다.