Kubernetes 보안 취약 사례 분석 및 대응 연구

요 약

웹 서비스를 구축하여 kubernetes를 사용해 컨테이너 환경에 관리하게 되면 발생되는 보안 취약점이 있다. 그 보안 취약 사례를 가지고 분석하여 실제로 구현하고 어떻게 대응할 것인지 연구한다.

1. 서론

1.1. 연구배경

현대 컨테이너 기술은 가상화 기술에 비해 더욱 경량화되어 있고, 확장성이 높아 애플리케이션 배포 및 관리가 용이하다는 이점이 있다. 이러한 이점으로 인해 많은 기업들이 컨테이너 기술을 도입하고 있으며, Kubernetes는 이러한 컨테이너 환경에서 애플리케이션을 보다 효율적으로 관리하기 위해 사용되고 있다.

Kubernetes를 사용하는 이유 중 하나는, 여러 대의 서버에서 동작하는 컨테이너 애플리케이션을 효율적으로 관리하기 위해 필요한 다양한 기능들을 제공하기 때문이다. 예를 들어, Kubernetes는 컨테이너를 스케줄링하고 관리하기 위한 API, 라우팅, 스케일링, 로드 밸런싱, 애플리케이션 구성 등의 다양한 기능을 제공한다. 이러한 기능들을 통해 개발자는 애플리케이션을 보다 쉽게 배포하고 관리할 수 있으며, 이는 개발 생산성을 높이는 데에 큰 도움이 된다.

하지만 Kubernetes는 많은 기능들을 제공하기 때문에, 보안에 대한 취약점이 발생할 가능성이 있다. 예를 들어, 권한 관리, 인증, 인가 등의 보안 문제가 발생할 수 있다. 본 프로젝트는 실제로 Kubernetes 클러스터를 구축하고 보안 취약점을 조사해 실제로 구현하여 분석 및 대응법을 연구한다.

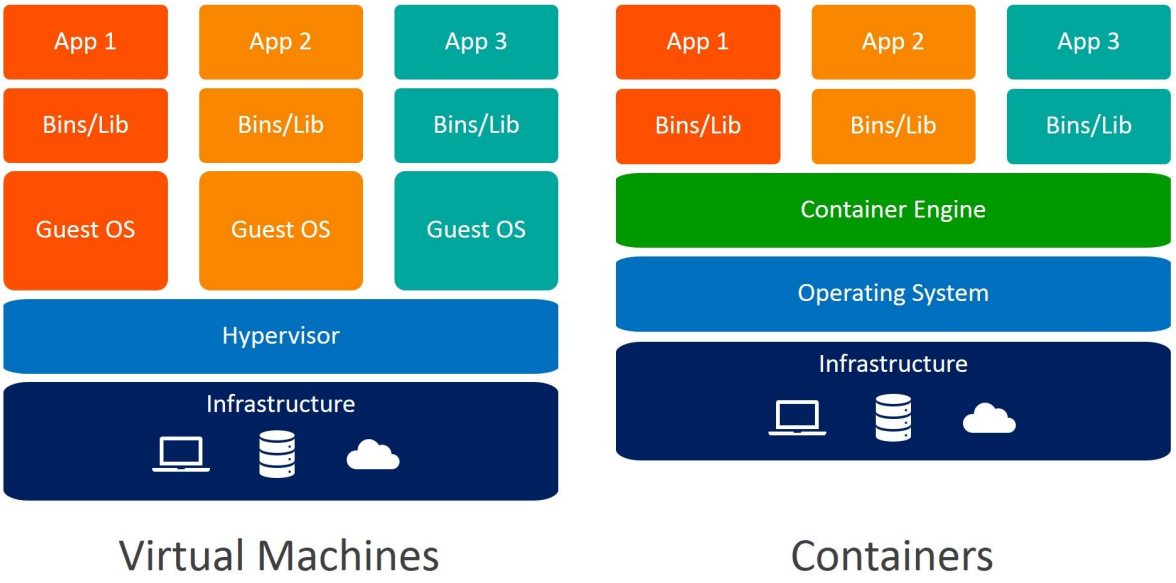
1.2. 연구목표

본 프로젝트의 연구 목표는 Kubernetes를 이용하여 간단한 웹 서비스를 구축하고, 보안 취약점을 조사하고 실제로 구현해 보안 대응 방법을 연구하는 것이다. 구체적으로, 구축한 웹 서비스 및 Kubernetes 클러스터에 내부에서 발생하는 보안 취약점을 식별하고 분석하며, 예방하고 대응하는 것을 목표로 한다.

이를 위해, 최대한 간소한 웹 페이지를 구성하고, 필요한 기술과 지식을 습득하여 웹 서비스를 구축하고 보안 취약점을 해결할 수 있도록 노력할 것이다. 또한, 이 연구는 Kubernetes를 사용하는 기업들에게 많은 도움을 줄 수 있으며, 보안 취약점에 대한 인식을 높이고 대응 방안을 제시하는 데에 큰 의의가 있다.

2. 관련 연구

2.1. 컨테이너 이미지 관리



가상화란 컴퓨터의 자원을 효율적으로 사용하기 위해 Host OS Kernel위에 Hypervisor을 사용하는 것이다. Hypervisor가 각각의 가상 머신을 관리하며 가상 머신 위에 가상의 Guest OS를 사용하여 각각 Application을 이용하는 방식이다. 각 가상 머신들끼리 독립적이며 Hypervisor가 이를 관리한다.

가상화를 통해 하나의 물리적 서버에서 여러 개의 가상 서버를 구축하여 서버 자원의 효율적인 사용이 가능해진다.

컨테이너는 가상화 기술 중 하나로, 운영 체제 수준에서 가상화를 구현한다. 각각의 컨테이너는 가상화된 운영 체제 위에 격리된 공간을 만들어 프로세스와 파일 시스템을 독립적으로 실행한다. 이러한 방식은 가상화에 비해 오버헤드가 적고 가볍다. 컨테이너를 실행하기 위한 파일 시스템과 애플리케이션을 패키징한 파일을 이미지라고 한다. 컨테이너 기술은 이미지 생성이 간편하며, 빠르고 확장성이 좋아서 인기를 얻고 있다.

본 프로젝트에서는 컨테이너 위에 이미지를 사용하여 웹 서비스를 구축한다. 사용할 프로그램은 다음과 같다.

2.1.1. Docker

Docker는 컨테이너 가상화 기술을 이용하여 애플리케이션을 개발, 배포 및 실행할 수 있는 플랫폼이다. Docker를 이용하면 애플리케이션과 그에 필요한 라이브러리, 환경 등을 패키징하여 컨테이너 이미지로 만들고, 이를 다른 환경에서도 동일하게 실행할 수 있다.

Docker는 다양한 운영 체제에서 사용할 수 있으며, 매우 가볍고 빠른 속도로 애플리케이션을 실행할 수 있다. 또한, 도커 이미지는 레이어(layer) 형태로 구성되어 있어서 여러 이미지를 공유하여 사용할 수 있고, 이를 이용하여 애플리케이션을 보다 쉽게 배포할 수 있다.

Docker는 다양한 기능을 제공합니다. CLI를 통해 컨테이너를 생성, 실행, 중지 및 삭제가 가능하며 컨테이너를 이미지로 빌드할 수 있다.

Docker는 다양한 플랫폼에서 사용되며, 애플리케이션을 빠르게 개발하고 배포하는 데 매우 유용하다. 본 프로젝트에서는 웹 서비스 이미지를 빌드하여 Docker Hub를 통해 이미지를 배포한다.

2.1.2. Kubernetes

Kubernetes는 컨테이너 오케스트레이션 플랫폼으로, 컨테이너화된 애플리케이션의 배포, 확장, 관리 등을 자동화하는 도구이다. Kubernetes는 컨테이너 애플리케이션을 자동으로 배포, 스케일링, 관리할 수 있도록 지원하며, 이를 위해 다양한 기능을 제공한다.

Kubernetes는 Control Plane과 Worker Node로 구성된다. Control Plane은 클러스터의 상태를 관리하고, Worker Node는 애플리케이션 컨테이너를 실행한다. Kubernetes는 API 서버, 스케줄러, 컨트롤러 매니저, etcd 등의 컴포넌트로 구성되어 있다.

Kubernetes는 컨테이너를 자동으로 배포하고 관리하는 기능, 애플리케이션을 무중단으로 업데이트하고 롤백하는 기능, 스케일링, 로드밸런싱, 자동 복구 등의 기능을 제공한다. 또한, Kubernetes는 컨테이너 간의 네트워크, 보안, 스토리지 등의 기능을 제공하여 컨테이너화된 애플리케이션을 보다 쉽게 관리할 수 있도록 지원한다.

Kubernetes는 다양한 리소스 오브젝트(Resource Object)를 사용하여 애플리케이션을 관리한다. 이러한 리소스 오브젝트는 노드, 파드(Pod), 서비스(Service), 볼륨(Volume), 디플로이먼트(Deployment), 스테이트풀셋(StatefulSet) 등이 있다. 이러한 리소스 오브젝트는 YAML 파일로 정의되며, Kubernetes는 이를 이용하여 애플리케이션을 배포하고 관리한다. 본 프로젝트에서는 Deployment를 이용해 웹 서비스 Pod를 관리하는 것을 구현한다.

2.2. 기존 보안 취약점 분석

CVE-2019-5736: 공격자가 호스트 운영 체제를 컨테이너 안의 프로세스와 교체하거나 수정할 수 있는 취약점
CVE-2022-0185: Kubernetes API 서버를 통해 실행되는 파드(pod) 내부의 프로세스를 임의로 조작할 수 있는 취약점

CVE-2021-2399: 미스매치된 인증서 검증으로 인해 특정 통신에 대해 중개자 공격이 가능한 취약점
CVE-2021-3118: Kubernetes API 서버에서 원격 코드 실행이 가능한 취약점
CVE-2021-25735: 컨테이너의 호스트 시스템에서 파일을 수정할 수 있는 취약점

CVE-2021-25741: 권한 없는 사용자가 기본 권한을 사용하여 서비스 계정 토큰을 획득할 수 있는 취약점

CVE-2021-25742: 권한 없는 사용자가 기본 권한을 사용하여 서비스 계정을 미리 예약할 수 있는 취약점

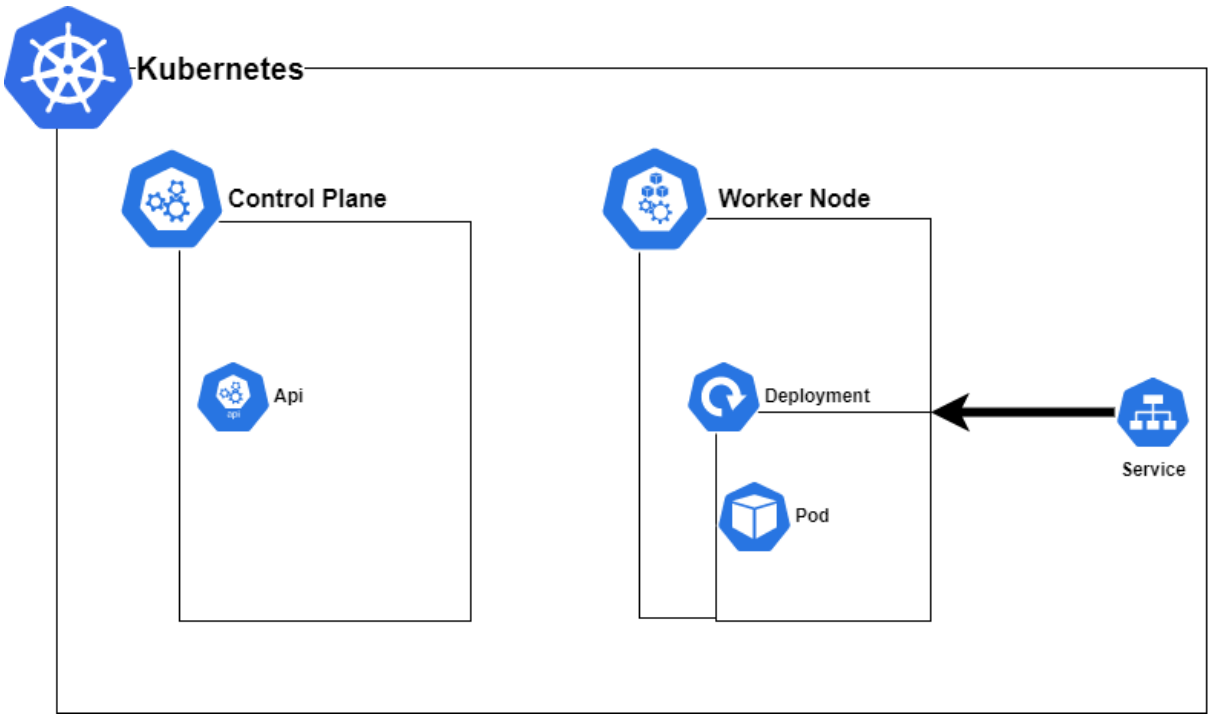
CVE-2021-25738: 노드 라벨링 기능에서 발생하는 취약점으로, 권한 없는 사용자가 노드 라벨링을 조작할 수 있음

CVE-2020-8555: kube-apiserver에 대한 DNS 구성이 잘못된 경우 DNS 구성에 의해 제어되는 도메인 이름을 가진 kubelet으로의 접근을 허용할 수 있는 취약점

3. 프로젝트 내용

3.1. 시나리오

3.1.1. Kubernetes 클러스터 구축



본 프로젝트에서는 Linux환경에서 Kubernetes를 설치한다. Kubernetes 설치는 공식 문서를 참고한다. Kubernetes 클러스터 구성은 Control Plane 1대와 Worker Node 1대를 사용한다. 웹 서비스는 Docker를 사용하여 이미지를 빌드한다. 빌드한 이미지를 배포할 수 있게 Docker Hub에 push한다. Kubernetes의 Deployment로 앞서 push한 이미지를 가지고 와 컨테이너를 구성한다.

3.1.2. 보안 취약점 구현

2.2. 에서 알아보았던 보안 취약점이 실제로 적용이 되는지 구현한다. 특히 보안 취약점 중에 위험하거나 파급력이 큰 취약점을 위주로 구현한다. 보안 취약점이 적용이 되는 구 버전의 Kubernetes를 사용하며 권한이 없는 공격자가 API서버 정보를 탈취하거나 파드 내부의 프로세스를 조작할 수 있는지 등의 취약점을 구현한다.

3.1.3. 대응 방안 계획

보안 취약점을 구현한 후 버전 업데이트 이외에도 취약점을 대응할 수 있는지 연구하여 방법을 모색한다. 취약점을 대응할 수 있는 결과가 나오는지 여러 취약점들을 분석하고 정리한다.

4. 향후 일정 및 역할 분담

매주 화요일 6시에 지정교수님과의 미팅
4월 : 웹서비스 제공, kubernetes올려 취약점 확인, 중간보고서 작성
5월 : 취약점을 확인 후 대응 방안 모색, 모색한 것을 토대로 해결방안 제시
6월 : 발표 준비, 최종보고서 작성

5. 결론 및 기대효과

Kubernetes 보안 취약점에 대한 연구를 통해, Kubernetes 클러스터 보안 강화에 필요한 대응 방안 및 추천 사항을 도출할 수 있다. 이를 통해 기업과 조직들은 Kubernetes 보안 강화를 위한 체계적인 대응 방안을 마련하고, 보안 취약점으로부터 발생할 수 있는 위협을 최소화할 수 있을 것이다.

또한, 본 프로젝트 연구를 통해 Kubernetes 보안에 대한 이해도를 높일 수 있으며, 컨테이너 보안 분야에 대한 관심을 높일 수 있다. 이는 Kubernetes를 사용하는 기업 및 조직들뿐만 아니라, 컨테이너 환경을 구축하고 운영하는 모든 사용자들에게 유익한 정보가 될 것이다.

6. 참고문헌

[1] https://www.cvedetails.com/vulnerability-list/vendor_id-15867/product_id-34016/Kubernetes-Kubernetes.html
[2] https://kubernetes.io/docs/home/
[3] https://hub.docker.com/