Kadapter: Kata 컨테이너를 위한 런타임 보안 집행 프레임워크

조치현[1], 허진[1], 이승수[2]

*인천대학교 (학부생[1], 교수[2])

Kadapter: A Runtime Security Enforce Framework for Kata Containers

Chi Hyeon Jo[1], Jin Her[1], Seungsoo Lee[2]

* Incheon National University

(Undergraduate student[1], Professor[2])

요 약

호스트 커널을 공유하는 컨테이너 환경에서 보다 고수준의 격리를 지원하기 위해 Kata 컨테이너라는 기술이 등장하였다. 하지만 기존 컨테이너 환경을 위한 런타임 보안 집행 시스템이 이를 지원하지 않기 때문에 컨테이너에 대한 제로데이 공격을 예방하기에는 한계가 존재한다. 본 논문에서는 이를 해결하기 위해 기존의 런타임 보안 집행 시스템을 Kata 컨테이너와 같은 격리가 강화된 컨테이너에서도 적용할 수 있도록 확장한 Kadapter 프레임워크를 제안한다. 실험을 통해 Kadapter의 실현 가능성, 커버리지를 평가하였으며 결론적으로 기존 런타임 보안 집행 시스템을 수정하지 않고 Kata 컨테이너에 런타임 보안을 집행할 수 있음을 입증하였다.

I. Introduction

컨테이너는 민첩한 배포 및 효율적인 자원 활용으로 인해 클라우드 네이티브 환경에서 적극적으로 채택되어 왔다. 기존 컨테이너는 namespace, cgroup 기술을 활용하여 격리된 실행 환경을 제공하지만, 호스트 커널을 공유한다. 공격자들은 이러한 저수준의 격리를 악용하여 취약한 컨테이너에서 호스트 시스템을 감시하거나, 호스트 시스템으로 탈출하여 악의적인 영향을 미칠 수 있다[1].

이러한 보안 한계를 극복하기 위해 컨테이너를 가상머신 계층으로 이중 격리하는 Kata 컨테이너[2]가 개발되었다. 공격자들은 각 Kata 컨테이너가 호스트 커널로부터 격리되는 특징으로 인해 기존 공격 방식으로 호스트에 악의적인 영향을 미칠 수 없다.

Kata 컨테이너를 사용함으로써 고수준의 격리를 제공하지만, 여전히 런타임 동안 Kata 컨테이너의 동작을 제한하는 것은 공격표면을 줄이기 위해 중요하다. 클라우드 네이티브 환경에서 사용자 정의 정책을 기반으로 런타임 동안 컨테이너의 동작을 제한하는 런타임 보안 집행 시스템으로 KubeArmor[3], Falco 등이 존재한다. 하지만 Falco는 Kata 컨테이너와 유사한 보안 컨테이너인 gVisor를 대상으로 감시 기능만 제공하며, Kata 컨테이너를 지원하는 런타임 보안 집행 프레임워크는 존재하지 않는다.

이러한 문제를 해결하고자 본 논문에서는 클라우드 네이티브 런타임 보안 집행 시스템을 확장하여 Kata 컨테이너의 런타임 보안 집행을 지원하는 Kadapter 프레임워크를 제안한다.

본 논문이 기여하는 바는 다음과 같다.

1. Kata 컨테이너의 런타임 보안 집행 지원: Kadapter를 통해 런타임 보안 집행 시스템인 KubeArmor를 확장하여 Kata 컨테이너의 런타임 보안 집행을 지원한다.

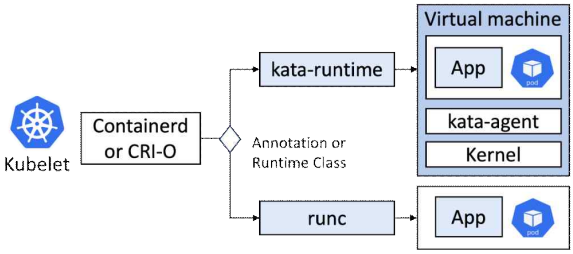

Fig 1. The Deployment Process of the Pod with runc and Kata Containers

2. Kata 컨테이너 정책의 효율적인 관리: Kadapter를 통해 KubeArmor 정책을 수정 없이 기존 컨테이너와 Kata 컨테이너가 공존하는 클러스터 내에 원활하게 적용할 수 있다. 이를 통해 정책 관리자는 정책 작성 시 컨테이너 런타임을 고려하지 않아도 된다.

II. Background and Motivation

2.1 Kata containers

Kata 컨테이너는 호스트 커널 및 자체 가상머신에 캡슐화된 다른 컨테이너로부터 격리된 커널을 가지는 가상머신에서 컨테이너 프로세스를 실행함으로써 고수준의 격리를 제공한다. 또한 kata-runtime은 OCI(Open Container Initiative)를 준수하여 쿠버네티스와 같은 컨테이너 관리 플랫폼과 통합하여 사용할 수 있다.

Fig 1은 쿠버네티스에서 Kata 컨테이너로 파드를 배포하는 과정을 나타낸 것이다. Kubelet으로부터 컨테이너 실행 요청을 받은 고수준 런타임은 어노테이션 또는 컨테이너 런타임 정보를 확인하여 저수준 런타임으로 kata-runtime 또는 runc 중 하나로 결정한다. kata-runtime은 경량 가상머신 실행을 위해 하이퍼바이저를 호출한다. 마지막으로, 경량 가상머신 내부에서 Kata 컨테이너와 컨테이너의 프로세스를 관리하기 위한 환경을 설정하는 kata-agent가 컨테이너를 실행한다.

2.2 KubeArmor

KubeArmor는 eBPF와 LSM(Linux Security Module)을 사용하여 정책 기반의 시스템을 제공함으로써 런타임에 클라우드 네이티브 워크로드의 원치 않는 악의적인 동작을 제한한다. KubeArmor는 실제 정책을 집행하기 위해 LSM 구현 중 AppArmor, SELinux, BPF-LSM을 사용하며, 현재 유연성이 뛰어난 BPF-LSM을 주로 사용한다. BPF-LSM을 정책 Enforcer로 사용하는 KubeArmor는 호스트 커널에 BPF-LSM 프로그램과 eBPF 맵을 설치하며, 사용자 정의 정책이 들어오면 eBPF 맵에 정책을 추가한다. 이후 정책 적용 대상 컨테이너의 동작이 감지되면 BPF-LSM 프로그램은 eBPF 맵을 조회하여 정책에 위반되는 동작인지 검사하여 허용 여부를 결정한다.

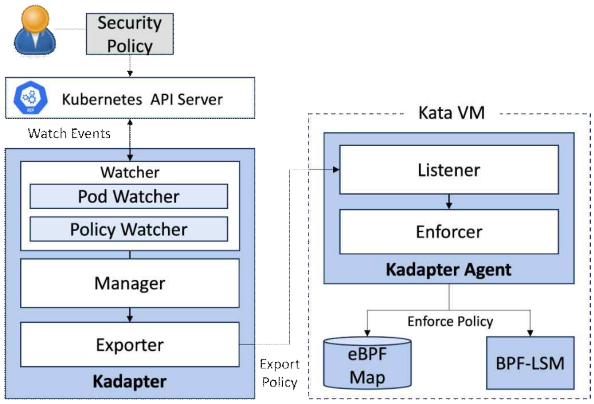

Fig 2. The Architecture of Kadapter

하지만 KubeArmor가 설치한 호스트 커널의 BPF-LSM 프로그램은 게스트 커널로 격리된 Kata 컨테이너의 동작을 감지할 수 없다. 따라서, 격리된 커널에서 실행되는 Kata 컨테이너는 KubeArmor를 지원할 수 없다는 한계가 있다.

III. System Design

3.1 Kadapter Design

Kadapter는 KubeArmor를 확장하여 Kata 컨테이너의 런타임 보안 집행을 지원하는 프레임워크로, 두 가지 요구사항을 충족하도록 설계되었다. 첫째, 사용자 정의 정책으로 런타임에 Kata 컨테이너의 악의적인 동작을 제한한다. 둘째, Kata 컨테이너에 대해 기존 KubeArmor 정책의 수정 없는 적용을 지원한다.

Kadapter의 동작 흐름은 Fig 2와 같다. Kata 가상머신에 배포된 Kadapter Agent의 Enforcer는 정책을 시행하기 위한 환경을 설정한다. 클

Table 1. Coverage in Runtime Security & Isolation Enhancements

Supports	Isolation Enhancements	Runtime Policy Enforcements
runc	X	X
kata	O	X
runc + KubeArmor	X	O
kata + KubeArmor	O	X
runc/Kata + KubeArmor + Kadapter	**O**	**O**

러스터 노드에 배포된 Kadapter의 Watcher는 각각 정책 이벤트와 파드 이벤트를 감지하여 Manager에게 전달한다. Manager는 정책 적용 대상인 파드가 Kata 컨테이너로 생성된 경우 Exporter에게 정책을 전달하고, Exporter는 정책을 Kata 가상머신에서 실행 중인 Agent의 Listener에게 전송한다. Listener는 정책 정보를 Enforcer에게 전달하고, Enforcer는 컨테이너 정보와 정책 정보를 조합하여 정책을 적용한다.

3.2 Policy Export

Kadapter는 Kata 컨테이너에 KubeArmor 정책의 수정 없는 적용을 지원한다. 이를 위해 Kadapter의 Watcher는 인포머를 사용하여 클러스터 내의 KubeArmor 정책 리소스와 파드 리소스 변화를 감지하여 Manager에게 전달한다. Manager는 리소스 변화를 전달받으면 쿠버네티스 API를 사용하여 정책 정보 또는 파드 정보를 조회하고 정책의 셀렉터, 파드의 레이블, 파드의 런타임 클래스를 조합하여 정책 전송 여부를 결정한다. Exporter는 Agent의 Listener에게 gRPC를 사용하여 정책 집행을 요청한다.

3.3 Policy Enforcement

Kadapter Agent는 게스트 커널의 BPF-LSM 모듈을 사용하여 실제 정책을 집행한다. KubeArmor의 정책을 지원하기 위해 KubeArmor의 BPF-LSM Enforcer 로직을 일부 변경하여 Enforcer로 사용하였다. Kata 컨테이너에서 제공하는 커널은 eBPF 프로그램 실행을 위한 BTF를 지원하지 않으므로, 해당 기능을 추가하여 게스트 커널을 빌드하여 사용하였다. Kadapter Agent의 Enforcer는 게스트 커널에 BPF-LSM 프로그램과 eBPF Map을 설치한다. Enforcer는 경량 가상머신 내의 다른 프로세스에 영향을 주지 않고, Kata 컨테이너에 대한 정책만 집행할 수 있도록 Kata 컨테이너를 특정해야 한다. 이를 위해 kata-agent가 실행한 초기 컨테이너 프로세스의 PID를 file에 쓰도록 수정하였다. 이후 Kadapter Agent는 file에 저장된 PID를 읽어, 초기 컨테이너 프로세스의 mount 네임스페이스 아이디와 pid 네임스페이스 아이디를 조합하여 Kata 컨테이너와 Kata 컨테이너가 실행하는 프로세스를 특정하는 키를 생성한다. Listener는 정책을 Kadapter로부터 전달받아 Enforcer로 전달하고, Enforcer는 Kata 컨테이너를 특정하는 키와 조합하여 eBPF 맵에 작성한다. 이후 Kata 컨테이너의 동작이 감지되면 게스트 커널에 설치된 BPF-LSM 프로그램이 eBPF 맵을 조회하여 동작의 허용 여부를 결정한다.

IV. Evaluation

4.1 Test Environments

실험 환경은 Ubuntu 22.04 기반 가상머신 1대와 베어메탈 머신 1대로, 가상 머신에는 마스터 노드를 두고, 베어메탈 머신은 워커 노드를 할당하여 쿠버네티스 클러스터를 구축했다.

위 실험은 실제 환경에서의 Kadatper의 적용 가능성과 Kadapter의 런타임 보안 기능을 평가하기 위해 진행되었다. 프로토타입을 실험하기 위해 마스터 노드에 Kadapter를 설치하고, 워커 노드에서 실행 중인 Kata 가상 머신에는 Kadapter Agent를 배치하였다.

4.2 Coverage

Table 1에서 볼 수 있듯이, 기존의 runc 환경에서는 저수준의 격리만을 제공하지만 KubeArmor와 같은 런타임 보안 시스템을 적용함으로써 런타임 보안 집행을 지원할 수 있다. Kata 컨테이너는 고수준의 격리를 제공하지만, 런타임 보안 집행이 지원되지 않는다. 반면, Kadapter는 Kata 컨테이너에 KubeArmor를 원활하게 연결하도록 지원하기 때문에 고수준의 격리를 유지하면서 런타임 보안 집행을 모두

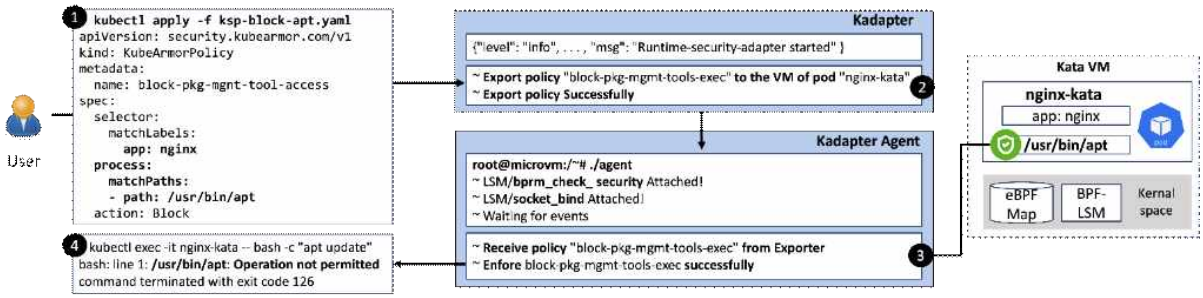

Fig 3. Use Case: Block the execution of the '/usr/bin/apt' for Kata Containers (app: nginx)

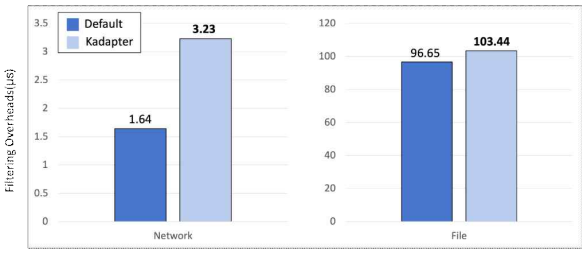

Fig 4. Filtering Overheads

지원한다. 이를 통해 격리가 강화된 Kata 컨테이너 환경에 런타임 보안을 적용함으로써 호스트 커널과 격리된 컨테이너에 최소한의 권한만을 부여할 수 있음을 확인하였다.

4.3 Use Case

Fig 3과 같이, 프로세스 실행을 제한하는 시나리오로 Kadapter의 실현 가능성을 평가하였다. Kadapter의 Agent는 Kata 가상머신에 배포되어 정책 집행에 필요한 eBPF Map과 BPF 프로그램을 실행한다. 사용자는 'app=nginx' 레이블을 가진 파드가 '/usr/bin/apt'를 실행하는 것을 차단하는 Kubearmor 정책을 배포(①)한다. Kadapter 시스템은 배포된 정책을 감지하여 대상 파드가 runc 혹은 kata-runtime으로 실행되었는지 확인하고, kata-runtime으로 실행된 경우 정책을 Kadapter Agent로 보낸다(②). 이를 전달받은 Kadapter Agent는 수신한 정책을 집행(③)한다. 정책이 정상적으로 적용된 후(④), Kata 컨테이너에서 'apt' 프로세스를 실행하면 Permission denied'가 반환되면서 실행이 차단된다. 이는 기존 KubeArmor 정책을 수정하지 않고 Kata 컨테이너에 성공적으로 적용되었음을 시사한다. 이외에도 file, network에 대한 제한 정책이 정상적으로 집행되는 것을 추가적인 실험을 통해 확인하였다. 이를 통해 기존 KubeArmor 시스템을 변경하지 않고 Kadapter를 추가 배포하여 Kata 컨테이너에 런타임 보안을 집행할 수 있음을 입증하였다.

4.4 Filtering Overheads

Fig 4는 컨테이너 동작 필터링으로 인한 오버헤드를 측정하기 위해 Kadapter를 실행하지 않은 Kata 컨테이너와 실행한 Kata 컨테이너에서 socket, open 시스템 호출을 각각 1,000,000번 호출한 후 실행까지 걸리는 시간의 평균을 계산하여 비교한 것이다.

V. Conclusion and Future Work

Kadapter는 클라우드 네이티브 런타임 보안 집행 시스템인 KubeArmor를 확장하여 Kata 컨테이너의 런타임 보안 집행을 지원하며, 정책 작성 시에 파드의 컨테이너 런타임을 고려하지 않아도 되도록 하여 정책 집행의 투명성을 제공한다. 실험을 통해 Kadapter가 기존 KubeArmorPolicy를 수정하지 않고 Kata 컨테이너에 적용할 수 있음을 입증하였다.

향후에는 현재 프레임워크에서 정책 집행을 위해 발생하는 오버헤드를 개선할 계획이다.

[참고문헌]

[1] CVE-2024-21626, https://nvd.nist.gov/vuln/detail/CVE-2024-21626, 2024

[2] The speed of containers, the security of VMs [Online]. Available: https://katacontainers.io/

[3] KubeArmor, May 2024, [online] Available: https://github.com/kubearmor/KubeArmor