eBPF를 활용한 실시간 컨테이너 모니터링 시스템

김지수[1], 남재현[2]
[1]단국대학교 컴퓨터공학과 학부생
[2]단국대학교 컴퓨터공학과 교수
[1]imjs0807@dankook.ac.kr, [2]namjh@dankook.ac.kr

Real-Time Container Monitoring System using eBPF

Ji-Su Kim[1], Jaehyun Nam[2]
[1]Dept. of Computer Engineering, Dankook University (Undergraduate Student)
[2]Dept. of Computer Engineering, Dankook University (Professor)

요 약

컨테이너는 현대 클라우드 환경에서 핵심적인 역할을 수행하며, 이에 따라 많은 기업이 접근성과 확장성을 위해 이를 채택하고 있다. 그러나 컨테이너는 호스트 리눅스 커널과 컴퓨팅 자원을 공유하는 특징을 가지고 있어서, 한 컨테이너가 오작동하면 호스트 환경 전체에 악영향을 끼칠 수 있다. 따라서 실시간으로 컨테이너의 상태를 감시하고 이를 효과적으로 관리하는 것이 필요하다. 본 논문에서는 이러한 문제에 대응하기 위해 호스트에서 동작하는 모든 컨테이너의 활동을 실시간으로 통합 감시하고자 한다. 이를 위해, 본 논문에서는 Linux Namespace를 활용하는 컨테이너의 특징을 이용하여 호스트에서 실행되는 여러 프로세스 중 컨테이너 프로세스를 식별하고, 이후 eBPF (Extended Berkeley Packet Filter) 기술을 활용하여 컨테이너로부터 호출되는 시스템 콜을 kprobe와 kretprobe를 통해 모니터링하여 컨테이너의 활동을 실시간으로 감시할 수 있는 시스템을 제안하고자 한다.

1. 서론

많은 기업이 접근성 및 확장성을 이유로 클라우드 기술을 작업 환경에 도입하였다. 이와 같은 환경을 효과적으로 사용하기 위해 클라우드 가상화 및 컨테이너 기술을 적용하고 있다. 클라우드 환경에서 컨테이너를 효과적으로 운영 및 관리하기 위해, 다양한 컨테이너 오케스트레이션과 컨테이너 런타임 인터페이스가 개발되었다[1]. 컨테이너는 일반적인 가상 머신과 달리, 호스트 리눅스 커널과 컴퓨팅 자원을 공유한다. 이 같은 이유로 한 컨테이너의 오작동은 컨테이너가 운영되는 호스트 환경 전체에 악영향을 미칠 수 있다[2].

이러한 문제를 대비하기 위해 본 논문에서는 리눅스 커널에서 이뤄지는 컨테이너 내부 프로세스 동작 감시 시스템을 제시한다. 이를 위해 3가지 단계를 수행한다. 먼저 Linux Namespace를 활용해 컨테이너 메타데이터를 수집한다. 이후 eBPF를 통해 호스트 리눅스 커널에서 호출되는 시스템 콜을 식별한다. 마지막으로 컨테이너 메타데이터와 시스템에서 이뤄지는 활동을 병합하여 시스템 콜을 호출한 프로세스의 컨테이너를 추적한다. 이를 통해 해당 컨테이너의 활동을 감시할 수 있다.

2. 컨테이너 시스템과 eBPF

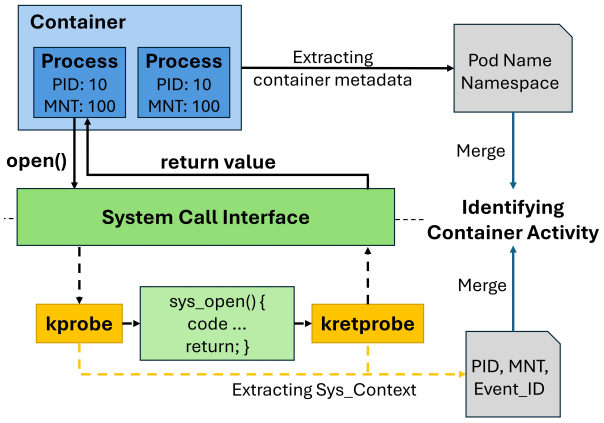
2.1 컨테이너 시스템

하나의 호스트에는 컨테이너를 포함한 다양한 프로세스가 실행되고 있으며, 컨테이너 내부에도 다양한 프로세스가 동작한다. 또한 가상 머신과 달리 컨테이너는 게스트 운영체제 없이 호스트 운영체제의 자원을 공유한다. 컨테이너는 Linux Namespace 기능을 이용하여 고유한 PID, MNT 등의 Namespace를 가지며, 이를 통해 독립적인 실행 환경을 가진다.

2.2 eBPF

eBPF (Extended Berkley Packet Filter)는 리눅스 시스템에 관측가능성을 부여하는 도구로 커널 내부에서 동적으로 사용자 프로그램을 로드하고 실행할 수 있는 기능을 제공한다[3]. 특히, Kprobe와 Kretprobe와 함께 이용하면 커널 내부의 동작(시스템 콜 호출 등)을 실시간으로 모니터링하고, 특정 이벤트가 발생하였을 때 즉각적으로 반응할 수 있다. 여기서 kprobe와 kretprobe는 시스템 콜들의 호출 직전 또는 종료 직후를 확인할 수 있는 Breakpoint이다. 또한 eBPF를 이용하면 커널 공간에서 만들어진 데이터를 사용자 공간으로 전송하여 추가적인 처리를 하거나 해당 데이터를 분석에 활용할 수 있다.

3. 컨테이너 모니터링 시스템



(그림 1) 실시간 컨테이너 모니터링 시스템

3.1 컨테이너 메타데이터 수집

단일 호스트 내에는 다수의 컨테이너가 동작하고 있는 만큼, 개별 컨테이너에 대한 활동을 모니터링하기 위해서는 실시간으로 변화하는 모든 컨테이너의 정보를 수집할 필요가 있다. 따라서 본 연구에서는 컨테이너 플랫폼으로 실시간으로 컨테이너 메타데이터 (Namespace, Pod, Container 이름 등)을 수집하고 내부적으로 최신 정보를 유지/관리한다. 또한, 컨테이너 메타데이터의 경우 사용자 정의 메타데이터로 컨테이너 내부 활동에 대한 시스템 메타데이터(PID, 시스템 콜 정보 등)과의 연계를 위해 개별 컨테이너의 Init 프로세스 PID 정보도 함께 수집한다.

3.2 컨테이너 내부 활동 모니터링

컨테이너의 프로세스는 호스트 커널을 공유하기 때문에 호스트 커널에서 시스템 콜을 모니터링하면 컨테이너의 프로세스가 호출하는 시스템 콜 또한 모니터링할 수 있다. 따라서 본 연구에서는 시스템 콜 모니터링을 위한 eBPF 프로그램을 kprobe와 kretprobe를 이용하여 호스트 커널에 삽입하여, 호스트 내에서 실행중인 프로세스가 호출하는 시스템 콜을 모니터링, 프로세스의 동작 흐름을 추적한다. 그리고 어떠한 시스템 콜이 호출되었는지에 따라 프로세스 실행, 파일 접근, 네트워크 통신 등 컨테이너의 내부 활동을 식별할 수 있다.

하지만, 시스템 콜 정보만으로는 시스템 콜을 호출한 프로세스가 일반 호스트 프로세스인지, 컨테이너 프로세스인지를 식별하기 어렵다. 따라서 본 연구에서는 PID와 MNT Namespace 정보를 추가로 활용한다. 여기서 PID와 MNT Namespace는 커널 내에서 프로세스 및 파일 시스템을 식별할 수 있는 고유한 식별자이다. 그리고 위 정보를 확인하면 시스템 콜을 호출한 주체가 컨테이너 프로세스인지 어떠한 컨테이너 속해 있는지 알 수 있을 뿐만 아니라 호스트 프로세스에 대한 불필요한 모니터링도 피할 수 있다.

2.3 컨테이너와 시스템 메타데이터 연계

마지막으로, 앞서 수집한 컨테이너 메타데이터와 시스템 콜에서 수집한 시스템 메타데이터를 연계하여 어떠한 컨테이너 프로세스가 해당 시스템 콜을 호출하였는지, 어떠한 활동을 하고 있는지 특정한다. 이를 위해 컨테이너 메타데이터 수집 시 획득한 Init 프로세스의 PID 정보를 이용하여 개별 컨테이너에 해당하는 PID, MNT Namespace 정보를 확인하고, 모니터링 과정에서 수집된 시스템 메타데이터 중 PID, MNT Namespace 정보와 비교한다. 최종적으로 컨테이너 메타데이터와 시스템 메타데이터를 종합하여 개별 컨테이너의 내부 활동을 모두 감시한다.

3. 결론

본 연구에서는 eBPF를 활용하여 호스트 커널 영역에서 발생하는 컨테이너 내부 프로세스의 활동을 감시하는 시스템을 제안하였다. 이를 위해, 컨테이너 플랫폼으로부터 컨테이너 메타데이터를 수집하였으며, Linux Namespace 기능을 활용하여 컨테이너 프로세스를 식별, eBPF와 k(ret)probe를 활용하여 특정 시스템 콜을 호출한 프로세스를 특정하였다. 마지막으로, 두 정보를 연계하여 컨테이너의 내부 활동을 감시할 수 있었다. 결과적으로 실시간 컨테이너 모니터링을 통해 컨테이너 기반 시스템의 안정성과 성능 향상에 기여할 수 있으며, 효과적인 컨테이너 운영 및 관리에 도움을 줄 수 있을 것으로 기대한다.

Acknowledgement

이 성과는 2024년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임. (RS-2023-00212738)

참고문헌

[1] Isam Mashhour Al Jawarneh, Paolo Bellavista, et al. "Container Orchestration Engines: A Thorough Functional and Performance Comparison", IEEE, 2019.

[2] Sari Sultan, Imtiaz Ahmad, et al. "Container Security: Issues, Challenges, and the Road Ahead", IEEE, 2019.

[3] Chang Liu, Zhengong Cai, et al. "A protocol-independent container network observanility analysis system based on eBPF", IEEE, 2020.