FaaSMon

프로비넌스 GNN 기반 서버리스 침입 탐지 시스템

팀원: 정지환, 이혜진, 양정용 지도교수: 김진우

서버리스와 컨테이너 재사용 공격

서버리스

- API를 사용해 함수 단위로 서비스를 제공하는 개발 및 실행모델이다.
- 각 함수는 컨테이너로 구성된다.

웜 컨테이너 (warm container)

- 컨테이너 재실행(Cold Start)으로 발생하는 오버헤드를 줄이기 위해 컨테이너를 종료하지 않고 재사용한다.

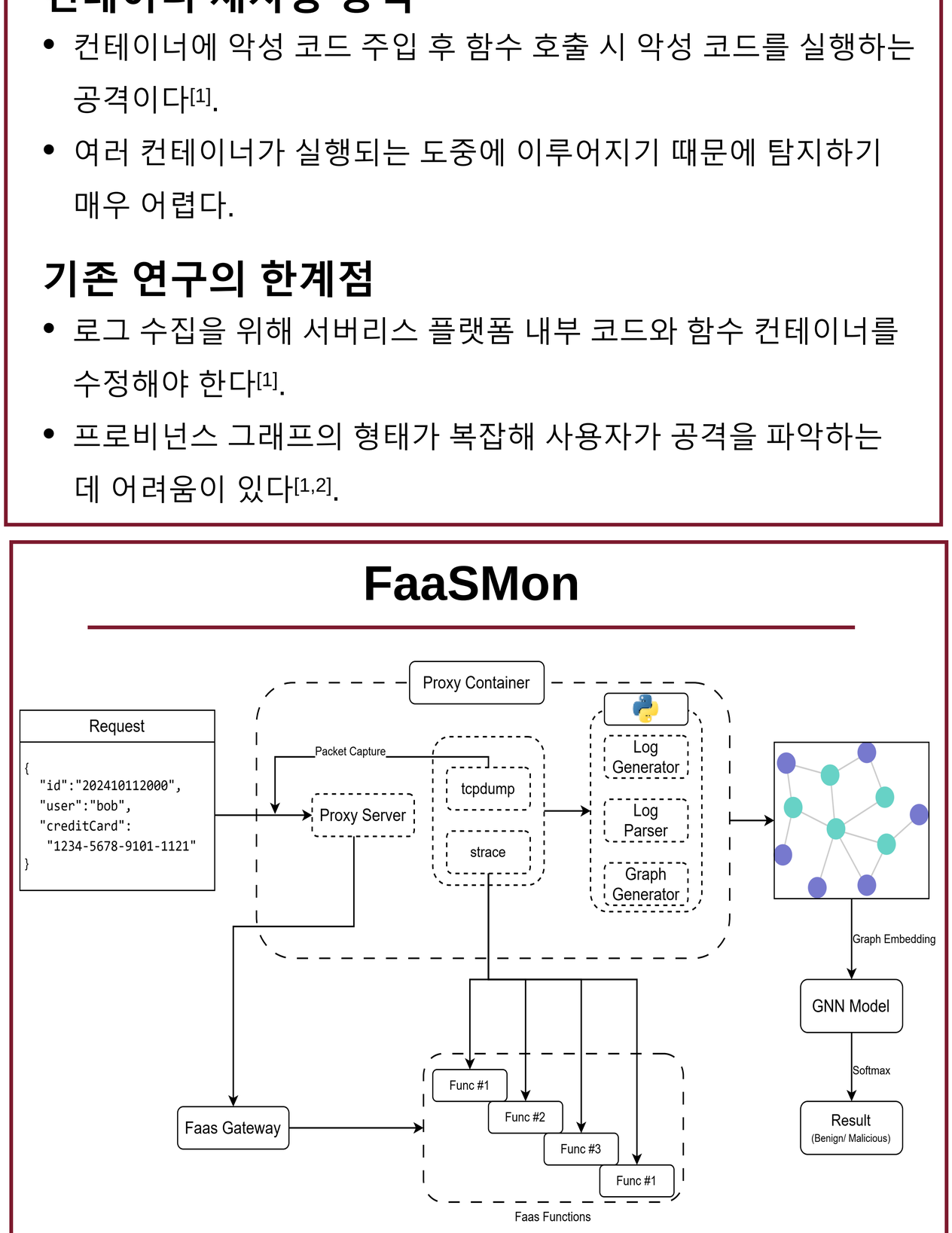
컨테이너 재사용 공격

- 컨테이너에 악성 코드 주입 후 함수 호출 시 악성 코드를 실행하는 공격이다[1].
- 여러 컨테이너가 실행되는 도중에 이루어지기 때문에 탐지하기 매우 어렵다.

기존 연구의 한계점

- 로그 수집을 위해 서버리스 플랫폼 내부 코드와 함수 컨테이너를 수정해야 한다[1].
- 프로비넌스 그래프의 형태가 복잡해 사용자가 공격을 파악하는 데 어려움이 있다[1,2].

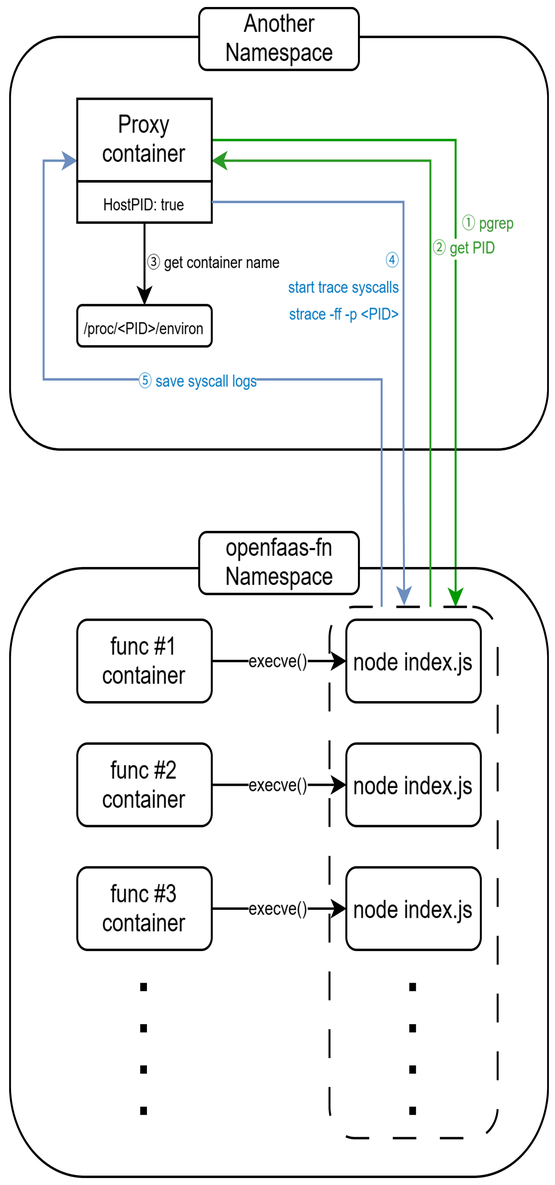
FaaSMon



<FaaSMon 아키텍처>

시스템 구조

- 사이드카 패턴을 통해 OpenFaaS watchdog 코드 및 기존에 생성한 함수 컨테이너 설정 변경을 최소화하고, 패킷 캡쳐 및 함수 컨테이너 내부 시스템 콜 수집을 수행한다.
- 수집한 로그를 파싱해 프로비넌스 그래프(Provenance Graph)를 생성하고 GNN을 사용한 분류기를 통해 서버리스 침입 여부를 파악한다.



패킷 캡쳐 및 시스템 콜 수집

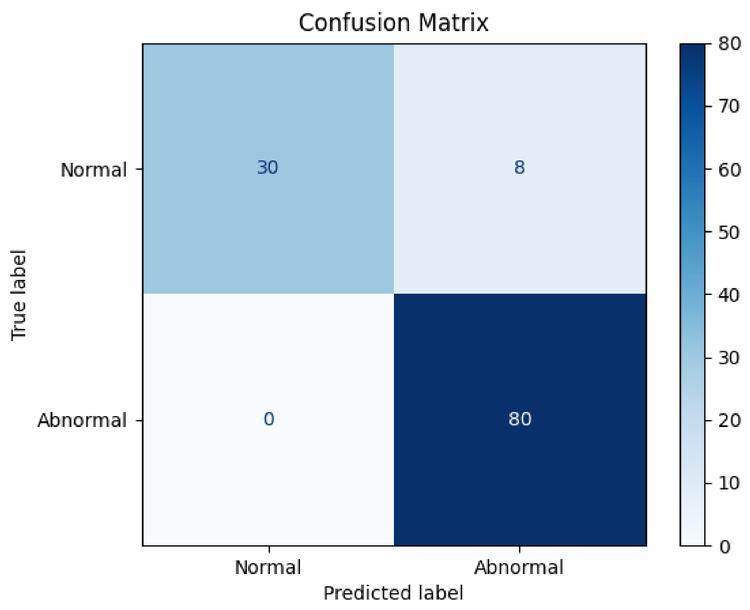
①pgrep을 사용해 함수 핸들러 프로세스를 검색한다.

②해당 프로세스의 PID를 가져온다.

③PID를 바탕으로 컨테이너 이름을 검색해 로그와 그래프에 사용한다.

④strace를 통해 함수 컨테이너 내부에서 시스템 콜을 수집한다.

⑤Python을 사용해 수집한 로그를 파싱하고 그래프를 생성한다.



GNN 기반 그래프 분류

- 패킷 캡쳐와 시스템 콜 로그를 통해 생성한 그래프가 정상적인 함수 흐름인지 검사하는 분류기를 제작했다.
- 학습에는 정상 그래프 187개, 공격 그래프 400개를 사용했다. (Train set : test set = 80 : 20)
- F1 Score: 0.9816, 테스트 정확도 96.97%

결론

기대 효과

- 사용자가 기존에 이용하던 플랫폼의 수정을 최소화하면서 서버리스 공격 탐지 솔루션을 활용할 수 있다.
- 그래프 분류를 통해 복잡한 함수 흐름 상에서 서버리스 공격 여부를 간편하게 파악할 수 있다.

[1] Datta, Pubali, et al. "ALASTOR: Reconstructing the provenance of serverless intrusions." 31st USENIX Security Symposium (USENIX Security 22). 2022.
[2] Chen, Xutong, et al. "CLARION: Sound and clear provenance tracking for microservice deployments." 30th USENIX Security Symposium (USENIX Security 21). 2021.