Fuzz Testing을 통한 위성 SW 분석

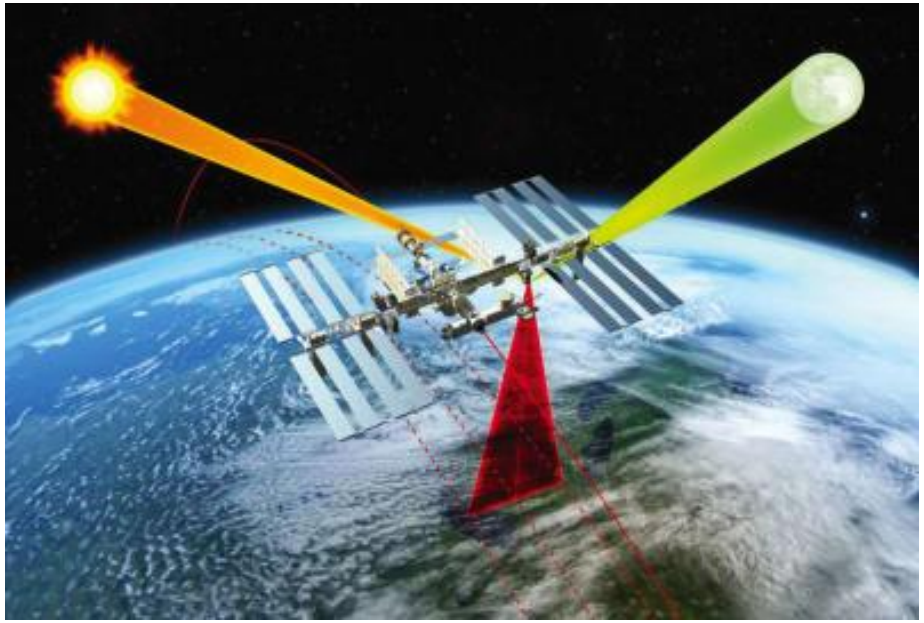
종합설계1 Week5

202002473 김승혁
201902733 이정윤
202002699 조민기

연구 배경

왜 필요한가 ?

- 최근 저비용, 소형 위성 개발 기술이 급속이 발전하면서 NASA의 오픈소스 위성 비행 소프트웨어인 fprime이 널리 사용됨.
- 하지만, 여전히 테스트 사례와 전용 도구가 부족함.
- Fuzzing은 결함을 찾아내는데 매우 효과적인 방법임.
- 그럼에도 fprime의 구조적 특성을 고려한 fuzzing 연구는 아직 초기 단계임.



nasa/**fprime**

F´ - A flight software and embedded systems framework

179 Contributors | 19 Used by | 569 Discussions | 10k Stars | 1k Forks

연구 목적

- Fprime 환경에서 다양한 fuzzing 도구를 비교 분석

- 컴포넌트 기반 구조에 최적화된 fuzzing 전략을 제시하는 것

-〉 발사 전 지상 테스트 단계에서 결함을 미리 잡아내고, 실제 위성 운영 중 발생할 수 있는 문제를 사전에 차단하는 것이 목표

연구 질문 및 가설

연구 질문

RQ1

- Fprime 환경에서 fuzzing을 사용하는 것이 기존 테스트 방식과 비교했을 때, 결함 탐지율에 어떤 영향을 미치는가?

RQ2

- Coverage-Guided 퍼저, Generation-Based 퍼저 등 다양한 퍼저 유형이 fprime의 컴포넌트 기반 아키텍처와 FPP 통신 방식에 따라, 결함 탐지 성능에 어떤 차이를 보이는가?

연구 가설

H1

- 퍼징을 활용한 결함 탐지 활동은 기존 방식보다 결함 탐지율을 유의미하게 향상시킬 것이다.

H2

- Generation-Based 퍼저는 컴포넌트 간 복잡한 통신을 다루는 데 더 효과적이어서, 더 많은 시스템 상태를 탐색하고 더 높은 결함 탐지 성능을 보일 것이다.

유스케이스 다이어그램

소프트웨어 사용 시나리오

주요 사용자는 위성 소프트웨어 개발자와 테스트 엔지니어

다음과 같은 흐름으로 fuzzing 수행

1. 위성 소프트웨어의 소스 코드와 테스트 데이터를 fuzzing 도구에 입력
2. 퍼저가 다양한 입력값을 자동 생성하여 소프트웨어를 반복적으로 테스트
3. 테스트 결과로 결함 탐지 리포트와 코드 커버리지 데이터 생성

이 과정은 클라우드 서비스를 연계해 자동화.

Docker 컨테이너를 활용해 일관된 테스트 환경 보장

Usecase Diagram

1.4. 소프트웨어의 사용 사례 Diagram

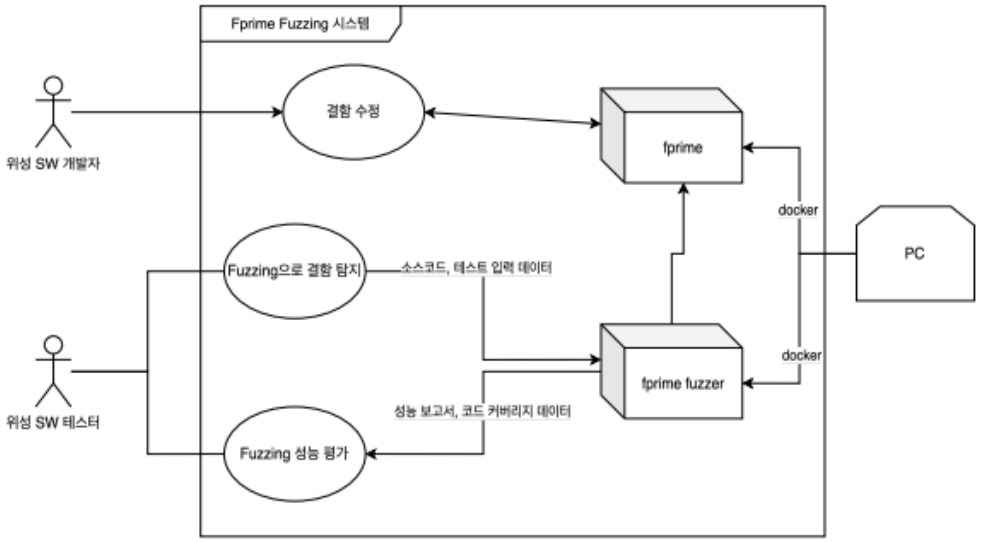


그림 1 Fuzzer 적용 시스템

문제 해결에 대한 사용

직접적인 요인

1. 구조적 복잡성
2. FPP 통신의 특수성
3. 상태 기반 시스템

간접적인 요인

1. 기존 퍼저는 fprime의 특수 구조를 고려하지 못함
2. 사용자 입력값이 예측 불가능해 특정 결함 재현이 어려움
3. 테스트 환경 구축과 통합이 복잡함
4. 예산이 한정되어 고가의 상용 도구 사용하기 힘듬

1.5. 문제 해결에 대한 사용 사례 Diagram

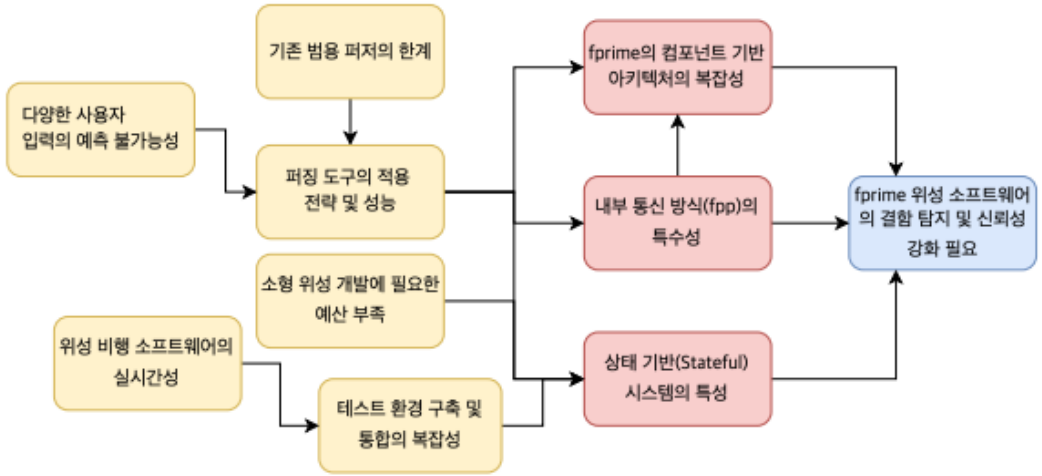


그림 2

Thank You