

# C/C++ 프로그램 대상

## Greybox Fuzzer의

### 사용성 향상을 위한 인터페이스 개발

참여기업: V+ Lab

지도교수님: 고윤민, 홍신 교수님

하정원

김연희



#### I. 문제 배경

##### 과제의 필요성

- 현재의 Fuzzing은 효과적이지만, 복잡한 상황에서 완전 자동화된 접근법은 Analyst의 의도대로 Fuzzing이 진행되지 않는 경우가 존재
- Analyst의 의도를 반영하기 위해 User와 Fuzzer의 Interaction을 지원하는 Fuzzing은 거의 제안되지 않음
- 위에 제시된 문제점을 극복하기 위해 User와 Fuzzer의 Interaction을 지원하는 방법으로 IJON, FuzzFactory가 제안되었지만, Public Archive된 AFL Fuzzing tool에 구현되어 한계성이 명확함

##### Problem Statement

- Fuzzing은 효과적이나, Analyst의 의도를 완벽히 반영하지 못함
- User와 Fuzzer의 상호작용하는 방법 부족

##### Constraints

- AFL++가 본 project를 수용할 수 있도록 기존 Code 보존을 최대한 보장하며 수정

##### Objectives

- AFL++에 User-Guided Fuzzing을 위한 Interface 개발
- 본 AFL++ Project Contribution

##### Function

- 특정 함수에 가중치를 빼서 Fuzzing을 억제하는 Suppress Mode
- 특정 함수에 가중치를 더해서 Fuzzing을 강화하는 Enhance Mode

#### 2. 기존 연구 비교 분석

##### 기존 연구

- Aschermann, Cornelius, et al. "Ijon: Exploring deep state spaces via fuzzing." 2020 IEEE Symposium on Security and Privacy (SP). IEEE, 2020.
- 요약: User와 Fuzzer의 Interaction으로 기존 Fuzzing의 문제 해결

##### Advantages

- Black Box 수준의 Fully Automated Fuzzing의 한계점인 Incomplete coverage, Limited Scope 해결
- 기존 Fully Automated Fuzzing으로 해결하기 어려운 Complex State Machine 탐색 가능

##### Disadvantages

- Public Archive된 AFL Fuzzing tool로 구현
- AFL의 Fuzzing Performane 한계
- AFL의 여러 Source Code들 대량 수정/AFL 병합 불가
- Target Source Code에 직접 Annotation 추가

#### 3. 핵심 내용 요약

##### Key Technologies

- Fuzzing: 무작위로 생성된 데이터를 프로그램에 입력하여 취약점을 발견하는 기법
- AFL/AFL++: 소프트웨어 테스팅에 사용되는 Fuzzing 도구
- Branch Coverage: 현재 실행이 코드의 모든 branch 중 커버하고 있는 정도를 나타내는 지표
- Instrumentation: 프로그램의 성능이나 다른 특정 측면을 분석하기 위해 컴파일 레벨에서 Target Code에 추가적인 코드를 삽입하는 과정
- Suppress/Enhance: Fuzzer에게 특정 함수에 대한 Fuzzing을 억제하거나 강화하도록 명령하는 모드

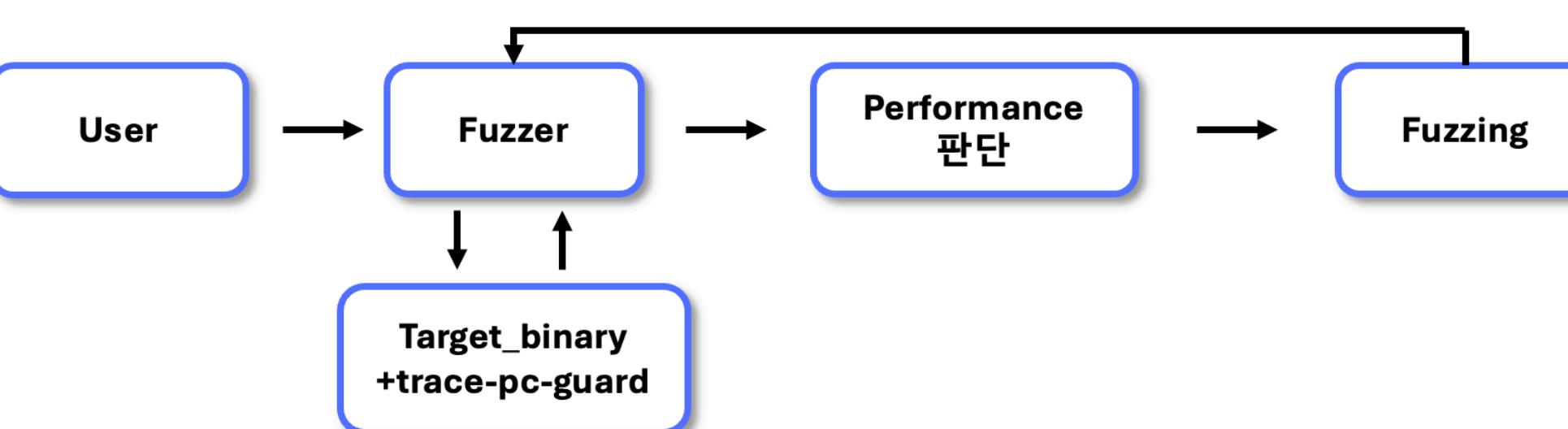
##### Specification

- Testcase 생성 및 Coverage 측정
  - Virgin AFL Fuzzing Testcase 생성 및 Coverage 측정
  - User-Guided Fuzzing Testcase 생성 및 Coverage 측정
- AFL++ 기반 User-Guided Fuzzing Interface 구현
  - 기존 AFL++ Source Code를 최대한 유지한채로 기능 구현
  - Suppress Mode와 Enhance Mode 구현

#### 4. 실험 결과 및 평가

##### Implementations

- AFL++에 enhance, suppress mode 활성화를 위한 option 추가  
입력값은 target function에 관한 정보가 담긴 JSON file path.
- testcase마다 target function coverage 파악 후 저장
- testcase의 target function coverage 정보로  
performance score에 weight 부여
- Performance score에 weight를 부여하여 Fuzzer가  
해당 testcase의 중요도를 판단하는데 있어서 User의 의도 삽입



##### Evaluation & Analysis

- Enhance Mode
  - 관련 branch count 평균 2.19배 상승, 그외 평균 5.8배 감소
  - 이전에 cover하지 못했던 branch 커버
- Suppress Mode
  - 관련 branch count 평균 7배 감소, 그외 평균 2배 감소
  - 이전에 cover하지 못했던 branch 커버

##### Future work

- weight값을 Fuzzing 시간에 따라 조절하여 유연한 Fuzzing 구현
- 다른 Fuzzing Interface mode 구현