삼족오

AI기반 취약점 자동탐지 트랙 Geumo 팀

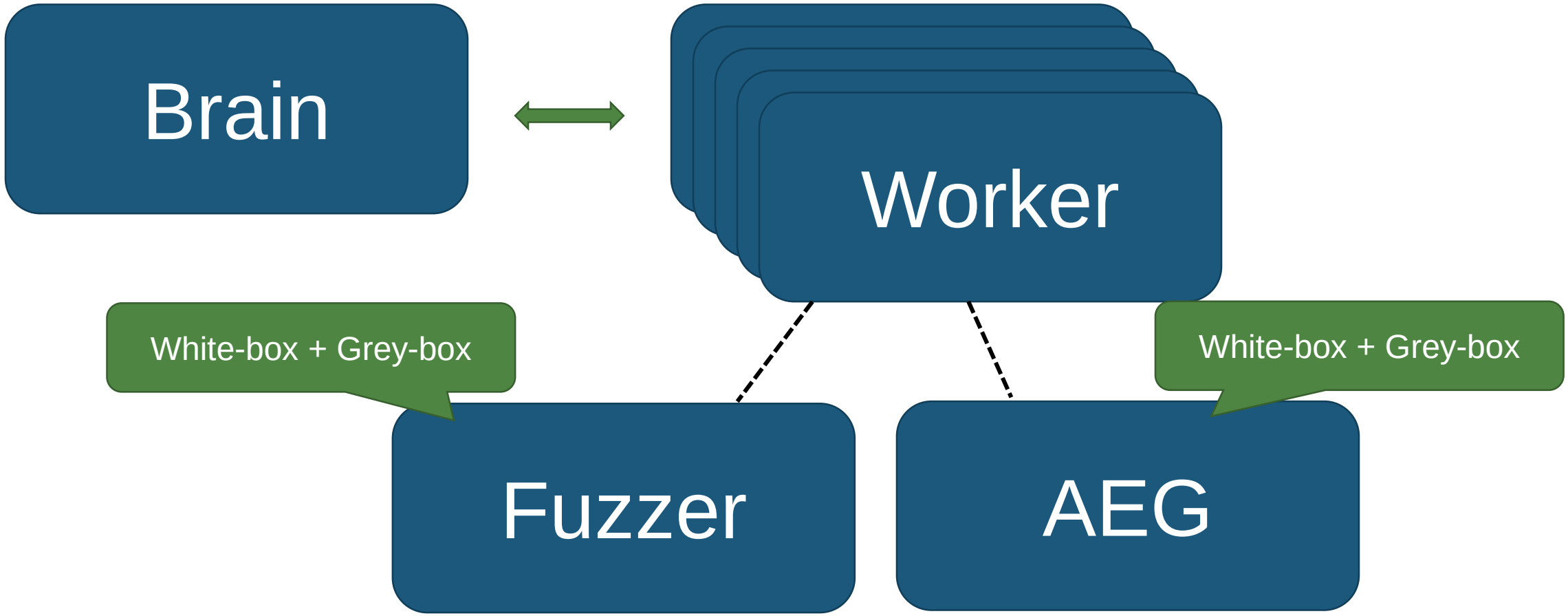
KAIST SoftSec Lab

목차

- 삼족오 시스템 소개
- 개선점, 대회 결과 및 분석
- 추후 개선 방향

삼족오 시스템 소개

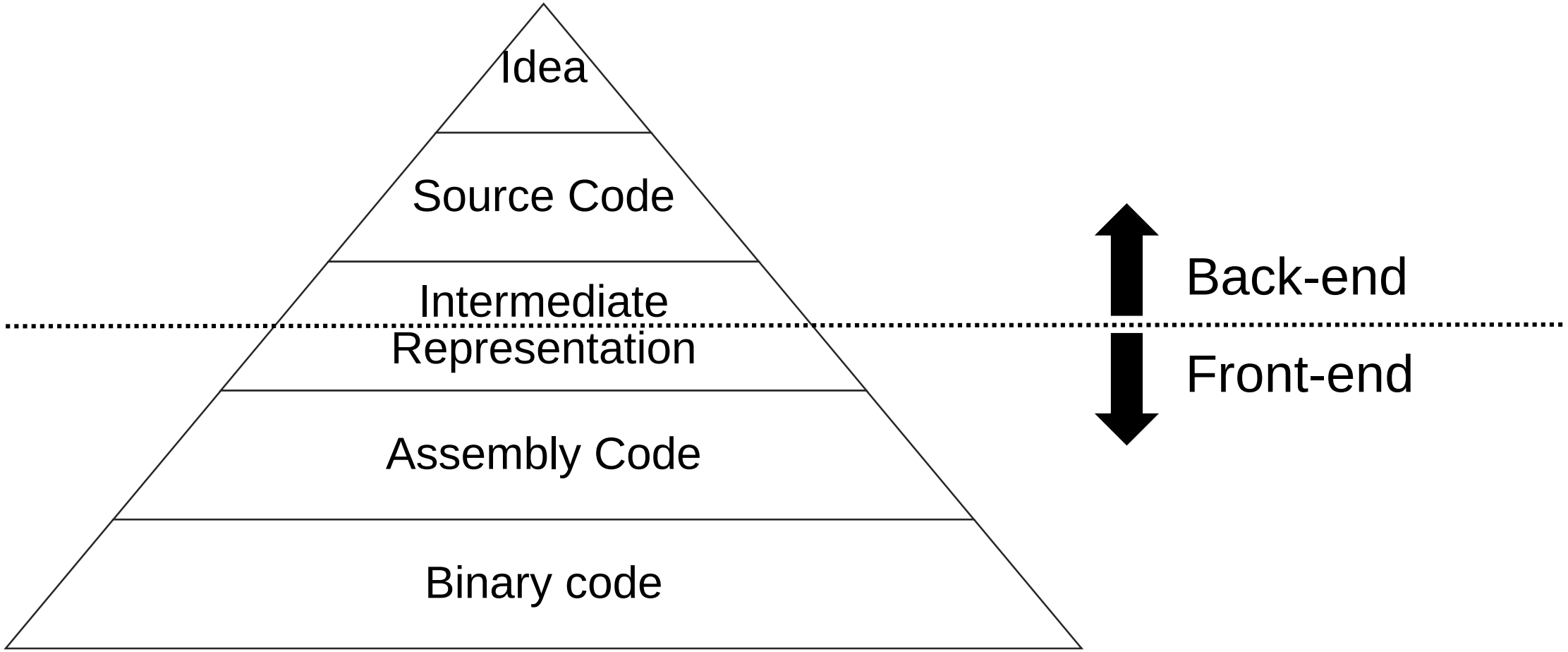
삼족오 (SamjoGo) 시스템



구현 상세

- 바이너리 분석 프론트 엔드
- 퍼징 엔진
- 공격 코드 작성 방법
- 확장성

바이너리 분석 프론트 엔드



바이너리 분석 프론트 엔드

- Disassembly & IR Lifting에 해당
- 단순 변환에 가까운 형태
  - 최근 연구는 거의 없는 편
- 이 부분이 빠르다 → 전체적인 분석 성능이 향상된다

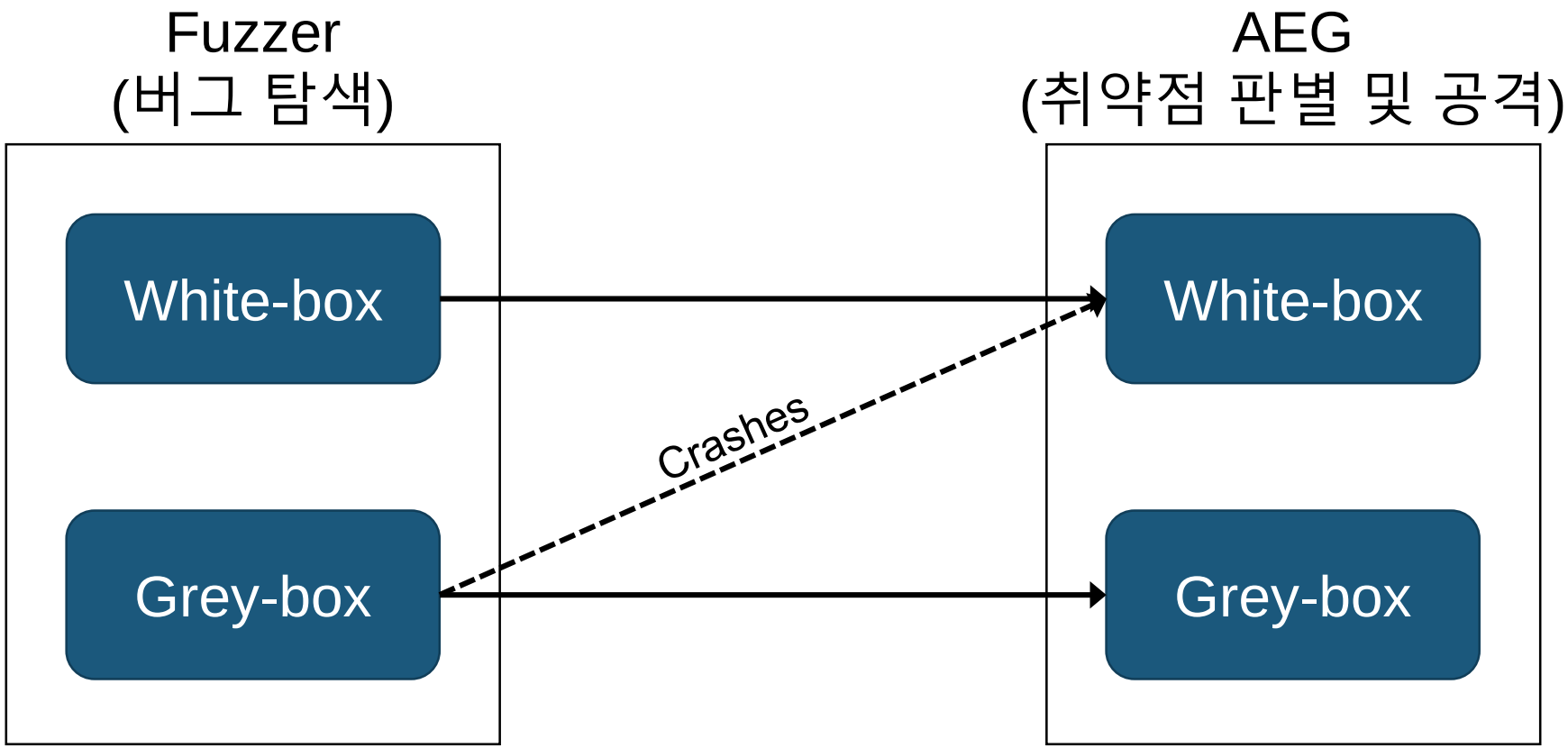
바이너리 분석 프론트 엔드

- B2R2
  - NDSS BAR 2019, “B2R2: Building an Efficient Front-End for Binary Analysis”

- 오픈 소스
  - https://github.com/B2R2-org/B2R2

Jung, M., Kim, S., Han, H., Choi, J., & Cha, S. K. (2019, February). B2R2: Building an Efficient Front-End for Binary Analysis. In The NDSS Workshop on Binary Analysis Research. Internet Society.

퍼징 엔진


Fuzzer
(버그 탐색)
AEG
(취약점 판별 및 공격)
White-box
Grey-box
Crashes
White-box
Grey-box

White-box + Grey-box?

White-box(기호실행)의 장점과 Grey-box의 장점을 결합

체계적 경로 탐색

빠른 탐색 속도

퍼징 엔진

- Eclipser
  - ICSE 2019, “Grey-box Concolic Testing on Binary Code”

- 오픈 소스
  - https://github.com/SoftSec-KAIST/Eclipser

Choi, J., Jang, J., Han, C., & Cha, S. K. (2019, May). Grey-box concolic testing on binary code. In Proceedings of the 41st International Conference on Software Engineering (pp. 736-747). IEEE Press.

공격 코드 작성 방법

- 취약점 유발 입력을 표현하는 독자적인 언어
  - Domain-Specific Language

- PoE (Proof of Exploit)
  - 취약점 유발 코드에 최적화된 동작으로 구성
  - Read, Write, Array slicing, Delay, Arithmetic 등등

예시

```
#include <stdio.h>
void func(){
  char *command;
  read(0, command, 0x10);
  system(command);
}
int main(){
  func();
  return 0;
}
```

Symbolic execve

예시에 대한 PoE

```
act exploit ():
  write 2F62696E2F7368007E7E7E7E7E7E7E7Ehs
  write 0Ahs
  delay 1000
  write 636174202E2F666C61670Ahs
  flag := (substr -128 -1 (read (-1)))
  return flag

submit:
  return (exploit ())
```

/bin/sh

cat ./flag

확장성

- Docker를 활용한 모듈화
  - 높은 재사용성
  - 빠른 시스템 배포

- Worker끼리 간섭을 최소화하여 병렬 작업

개선점, 대회 결과 및 분석

시범 대회 전 개선점

- 분석 엔진 최적화
  - 전반적인 성능 개선 및 버그 해결
  - 코드 및 의존성 정리
  - 문제 해결 전략 수정

- 분석 엔진 기능 확장
  - `libc` 라이브러리 함수 모델링
  - Format string bug 활용성 강화

시범 대회 중 개선점

- AI의 데이터 셋 적합도
  - One-shot 함수
  - `system` 함수 호출 탐지

시범 대회 결과

- 총 20개 중 11문제 해결

	White-box	Grey-box	Both	Failure	Total
Test	10	9	8	9	20

시범 대회 결과 분석

- 운에 의해서 경로 탐색의 성공 여부가 결정 – 2
- 공격 성립 조건이 아슬아슬하게 미치지 않음 – 5
- 기타 – 각 1
  - 휴리스틱 부족
  - IR 리프팅 실패
  - 예상 외의 용도의 FSB
  - 복잡한 공격 페이로드 요구

본선 대회 전 개선점

- 공격 성립 조건 확장
  - `system` 함수 호출 탐지 조건 완화
  - Command Injection 확장 및 버그 수정

- 시범 대회 문제 기준으로 16개 까지 해결

	White-box	Grey-box	Both	Failure	Total
Test	11	13	8	4	20

본선 대회 결과

- 총 40개 중 27개 해결

	White-box*	Grey-box*	Both*	Failure	Total
1st	8	3	0	9	20
2nd	8	10	2	4	20

* 대회 중 시스템은 문제를 푸는 즉시 해당 문제에 관련된 작업을 중지시키므로 실제보다 적을 수 있음

본선 대회 결과 분석

- 필터링 등의 우회에 실패 – 5
- 지원하지 않는 FSB 활용 형태 – 5
- 모델링 되지 않은 `libc` 함수 사용 – 1
- IR 리프팅 실패 – 1

추후 개선 방향

추후 개선 방향

- 추가 휴리스틱 탑재
  - Loop 탐색 개선
  - 필터링 우회

- 추가 공격 메커니즘 탑재
  - FSB 공격 확장
  - 여러 mitigation 조건에 따라 대응하는 공격 페이로드

추후 개선 방향

- 분석 엔진 개선
  - IR 리프팅 버그 해결
  - 더 많은 `libc` 함수 모델링

Question?