WEEK11-코드 난독화 도구 제작

# Test Plan / Test Cases Design

종합설계1

컴퓨터융합학부

**노형우 | 손지웅 | 조인우**

지도 교수 : 조은선

# 목차


01 연구 질문/가설

02 Test plan

1) 배경과 목적

2) 테스트 상세

3) 테스트 관리

03 Test cases

04 AI 도구 활용 정보

05 Q&A

# 01 연구 질문/가설

RQ1. 기존 난독화 도구에 비해 AI 기반 복원 도구(LLM)에 대한
저항성을 얼마나 향상시키는가?

H1. 기존 난독화 방식(Tigress) 보다
LLM(GPT-4o)기반 역난독화 시도에 대해 높은 복원 저항성을 보일 것

RQ2. 소스코드 수준 난독화와 IR 수준 난독화 중
어느 방식이 실행 성능과 보안성 균형 측면에서 더 효과적인가?

H2. 각각의 적용 대상과 상황에 따라
보안성과 성능 측면에서 상이한 효과를 보일 것입니다.

# 02 Test plan

## 1) 배경과 목적

### Test의 배경

대형 언어 모델 (LLM)의 발전으로
기존 코드 난독화 기법의
한계가 드러남

코드의 구조 자체를 변형하는
난독화 기법의 필요성이 대두

### Test의 목적

구조적 변형을 통한 구문 수준 코드 난독화 기법이
LLM기반 역난독화 저항성, 코드 복잡도, 실행 성능(시간) 등에 미치는 영향을
실험적으로 검증하는 것

# 02 Test plan

## 2) 테스트 상세

### 독립변수

1.원본 코드 (난독화가 적용되지 않은 원래의 코드)

2.ANLTR 기반 난독화 기법

3.기존 난독화 도구 ( Tigress의 다양한 옵션)

4.ANTLR + Tigress 복합 적용

### 종속변수

1. LLM 기반 역난독화 성공 여부

2.코드의 구조적 복잡도

3.프로그램의 실행 시간

# 02 Test plan

## 2) 테스트 상세

### 실험대상

테스트용 miniC 언어 기반, C언어 기반 프로그램 10종

조건문, 반복문, 함수 호출, assignment 등 다양한 구조를 포함

### 실험환경

하드웨어 : Apple Silcon M4, 16GB RAM, Mas OS 환경

소프트웨어 : 1. ANTLR 기반 구조적 난독화 도구 ( java 기반 )

2. LLM ( GPT-4o)

3. 실행 시간 측정 ( 터미널 time 명령어 )

네트워크 : LLM API 호출을 위한 인터넷 환경

# 02 Test plan

## 3) 테스트 관리

### 실험절차

1. 테스트 코드 준비
2. 난독화 적용
3. 지표 측정
4. 데이터 정리 및 분석

### 측정지표 및 도구

정량 평가 지표 : 사이클릭 복잡도, 실행 시간 증가율

정성 평가 지표 : LLM 기반 역난독화 성공 여부

사용 도구 : ANTLR, Java, Tigress, LLM API, 복잡도 분석 도구

# 03 Test Cases

| ID | 대상(모델/조건) | 실험 조건 | 테스트 데이터 | 평가지표 | 예상 결과 |
|---|---|---|---|---|---|
| TC-1 | 원본 코드 | 난독화하지 않은 원본 코드 | C코드 10종 | 사이클링 복잡도<br>실행 시간<br>LLM 복원 여부 | 가장 낮은 복잡도<br>최단 시간<br>LLM 복원 가능 |
| TC-2 | ANTLR 기반 난독화 코드 | 제안 도구 적용 | 동일 | 동일 | 복잡도 상승<br>시간 소폭 증가<br>복원률 감소 |
| TC-3 | Tigress -Flatten- | Flatten 적용 | 동일 | 동일 | 복잡도 상승<br>실행 시간 증가<br>복원률 일부 감소 |

# 03 Test Cases

| ID | 대상(모델/조건) | 실험 조건 | 테스트 데이터 | 평가지표 | 예상 결과 |
|---|---|---|---|---|---|
| TC-4 | Tigress -AddOPaque- | AddOPaque 적용 | C코드 10종 | 사이클링 복잡도<br>실행 시간<br>LLM 복원 여부 | 복잡도 상승<br>실행 시간 증가<br>복원률 일부 감소 |
| TC-5 | Tigress -Rename Identifiers- | RenameIdentifier 적용 | 동일 | 동일 | 복잠도 변화 미미<br>시간 변화 미미<br>복원률 거의 동일 |
| TC-6 | ANTLR<br>+<br>Tigress | 두 기법을<br>순차적으로 적용 | 동일 | 동일 | 복잡도 최고<br>실행 시간 증가<br>복원률 최저 예상 |

# 03 Test Cases

## 검증 기준(Metric)

| | |
|---|---|
| 사이클릭 복잡도 | 난독화 전후의 복잡도 차이로 평가<br>높을수록 구조적 난이도 증가로 간주 |
| 실행 시간 | 난독화 전후 평균 실행 시간의<br>상대적 증가율로 평가 |
| LLM 복원 성공률 | LLM에 “코드를 원래대로 복원하라” 입력 후<br>난독화 구조 제거 여부로 평가 |

# 04 AI 도구 활용 정보

| | |
|---|---|
| 사용 도구 | GPT-4o |
| 사용 목적 | 코드 역난독화 및 복원 저항성 실험, 소스코드 생성 |
| 프롬프트 | • 해당 코드를 원래대로 복원해줘<br>• 간단한 C언어 코드 작성해줘 |
| 반영 위치 | 1. 난독화 도구 평가<br>2. Testcase 작성 |
| 수작업 수정 | 있음 → AI가 제공한 Test 코드의 실행 검증, 논리 보강 등 |

# 05 질문과 답변 Q & A

궁금한 점이 있다면 자유롭게 질문 바랍니다 !

# THANK YOU

감사합니다