김일성종합대학학보 주체107(2018)년
정보과학 제64권 제3호

smali코드에서 식별자이름변경의 한가지 실현방법

오영근, 김경석

위대한 령도자 **김정일**동지께서는 다음과 같이 교시하시였다.

《다른 나라의 선진과학기술을 우리 혁명의 요구와 우리 나라의 실정에 맞게 연구도입하여 우리의것으로 만들면 그것도 주체적립장에서 과학기술을 발전시키는것으로 됩니다.》(《**김정일**선집》 증보판 제22권 24페지)

현재 국내에서 리용되고있는 Android응용프로그람들중 많은 부분이 국내외에서 이미 개발되였던 프로그람에 대한 역공학을 통해 재생산된것들이며 역공학을 통한 새 기술획득도 나라의 정보산업발전에 많은 기여를 하고있다. Apktool, VTS[3]와 같은 많은 도구들이 Android응용프로그람의 역공학에 리용되고있으며 이와 함께 보안기술도 발전하고있다.

론문에서는 smali코드를 리용하여 Android실행화일해석에 도움을 줄수 있는 식별자이름회복을 지원하는 한가지 방법에 대하여 제안하였다.

1. 해석방지기술

Android응용프로그람에 대한 해석은 크게 두 단계로 나누어 진행되는데 첫번째 단계는 원천코드화단계이고 두번째 단계는 코드해석 및 수정단계이다.

Apkprotect와 같이 허위코드와 무한순환, 오유코드를 삽입하여 원천코드추출을 막으려는 프로그람들이 이미 개발되여있지만 이에 대응하는 해석도구들도 많이 개발되여 이것은 피할수 없다.

두번째 단계인 코드해석 및 수정을 방지하는 기술을 코드혼란기술 또는 난해처리(obfuscation)기술[2]이라고 한다.

난해처리는 프로그람의 기능은 변화시키지 않으면서도 프로그람을 공격자가 리해하기 어렵도록 변경하는 의미보존프로그람변환[1]이다.

현재 가장 우수한 난해처리프로그람들로는 KlassMaster, J Shrink, ProGuard, Dash-O를 들수 있다.

난해처리기술에는 조종흐름변경(Changing of Control Flow), 문자렬암호화(Encoding Java Strings), 식별자이름변경(Name Mangling)들이 있다.

조종흐름변경난해처리는 허위코드삽입, 명령분해와 같은 기술을 리용하여 조종흐름을 복잡하게 만드는것이다.

문자렬암호화난해처리는 코드안에 있는 통보문과 같은 문자렬을 복호화함수를 거쳐야 정확한 문자렬이 나오도록 암호화하여 코드해석을 위한 정보를 얻지 못하도록 하는 기술이다.

식별자이름변경난해처리는 패키쥐이름, 클라스이름, 함수이름, 변수이름과 같은 의미있는 이름들을 의미없는 문자렬들로 교체하여 프로그람의 실행에는 영향을 주지 않으면

서도 역공학에 의해 얻어지는 코드를 리해하기 어렵게 만드는 기술이다.

현대적인 역공학도구들에서 조종흐름변경과 문자렬암호화난해처리에 대한 해석을 지원하지만 식별자이름변경난해처리에 대한 회복기능은 지원하지 않는다. 이로부터 론문에서는 Android응용프로그람의 역공학에서 식별자이름회복을 지원하는 기능을 실현하기 위한 한가지 방법을 제안하였다.

2. 식별자이름회복지원기능의 실현

우리는 baksmali/smali를 리용하여 얻어지는 smali코드를 해석하고 식별자들사이의 관계그라프를 구축하여 해석자가 식별자들의 이름을 회복할수 있도록 지원하는 기능을 실현하였다.

1) smali코드해석부의 설계

론문에서는 예약어방식으로 어휘해석을 진행하였다. 이때 클라스정의, 클라스계승, 대면부구현, 함수정의, 함수참조, 성원변수정의, 성원변수참조와 관련되는 예약어들은 표 1과 같다.

표 1. smali코드에서 일부 예약어들

예약어	해 설
.class	클라스정의
.super	클라스계승
.implements	대면부구현
.method	함수시작
.end method	함수끝
.field	성원변수정의
.local	국부변수정의
invoke-*	함수호출
const-class	클라스형상수창조
new-instance	클라스형변수창조
new-array	클라스형배렬창조
instance-of	자료형검사
sget-*	정적성원변수를 등록기에 적재
sput-*	등록기값을 정적성원변수에 넣기
iget-*	성원변수를 등록기에 적재
iput-*	등록기값을 성원변수에 넣기

표 1과 같은 예약어를 만나면 그 행을 문장해석단계에 넘긴다. 문장해석단계에서는 식별자만을 추출하여 해석자로부터 이름변경요청이 들어올 때마다 쉽게 변경을 진행할수 있는 호출계승관계그라프형 자료구조로 보관한다.

2) 호출, 계승관계그라프구축과 이름회복

호출, 계승관계그라프를 다음과 같이 정의한다.

$$G = \{P, C, F, V, PP, CP, CC, FC, FF, VC, FcF, FcV\}$$

여기서 *P*는 패키쥐모임, *C*는 클라스모임, *F*는 함수모임, *V*는 성원변수모임이며 이것들은 그라프에서 마디점들로 표시된다. 또한 *PP*는 패키쥐들의 소속관계, *CP*는 클라스와 패키쥐사이의 소속관계, *CC*는 클라스들사이의 계승관계, *FC*는 함수와 클라스사이의 소속관계, *FF*는 함수들사이의 재정의관계, *VC*는 성원변수와 클라스사이의 소속관계, *FcF*는 함수들사이의 호출관계, *FcV*는 함수와 변수사이의 참조관계를 나타내며 그라프에서 방향마디로 표시된다.

호출, 계승관계그라프구축흐름도는 다음과 같다.(그림)

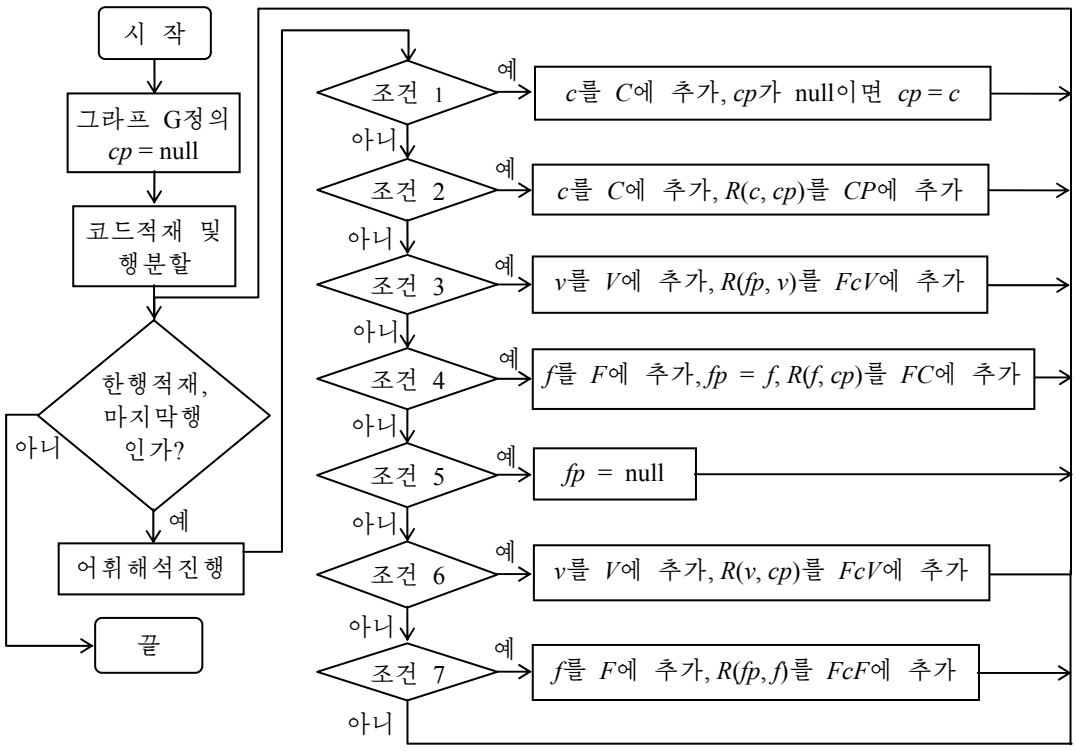

그림. 호출, 계승관계그라프구축흐름도

그림에서 조건 1－7의 내용은 표 2와 같다.

표 2. 호출, 계승관계그라프구축흐름도에서 조건들에 대한 해설

조건번호	조건내용
조건 1	행식별자가 .class, .local, const-class, new-instance, new-array, instance-of, check-cast인가?
조건 2	.super, .implements인가?
조건 3	sget-*, sput-*, iget-*, iput-*인가?
조건 4	.method인가?
조건 5	.end method인가?
조건 6	.field인가?
조건 7	invoke-*인가?

호출, 계승관계그라프를 리용한 이름회복과정은 다음과 같다.

① 이름변경요청을 접수한다.

② 이름변경요청의 목적대상을 확인하고 이름이 중복되지 않는가를 검사한다.

③ 목적대상이 패키쥐 *p*이면 그라프에서 관계 *PP*, *CP*를 조사하여 *p*에 속하는 패키쥐들과 클라스들에 대하여 *p*의 이전이름과 새 이름을 가지고 각각 smali문법의 패키쥐선언문을 만들어 문자렬치환처리를 진행한다.

④ 요청대상이 클라스 *c*이면 그라프에서 관계 *CC*, *FC*, *VC*를 조사하여 *c*와 관련되는 클라스, 함수, 성원변수들을 조사하여 *c*의 이전이름과 새 이름을 가지고 각각 smali문법의 클라스선언문을 만들어 문자렬치환처리를 진행한다.

⑤ 요청대상이 함수 *f*이면 그라프에서 관계 *FF*를 조사하여 함수 *f*를 재정의한 함수들과 *f*가 재정의한 함수들에 대하여 *f*의 이전이름과 새 이름을 가지고 각각 smali문법의 함수선언문을 만들어 문자렬치환처리를 진행한다.

⑥ 요청대상이 성원변수 *v*이면 관계 *VC*에서 *v*가 정의된 클라스 *c*를 찾고 관계 *CC*에서 *c*로부터 파생된 클라스들을 찾아서 *v*의 이전이름과 새 이름을 가지고 각각 smali문법의 변수선언문을 만들어 문자렬치환처리를 진행한다.

맺 는 말

이름변경난해처리된 Android실행화일해석에서 smali코드를 리용하여 식별자이름회복을 지원하는 방법을 제안하여 해석자가 이름회복뿐아니라 그것들사이의 구조적관계도 함께 참고할수 있게 하였다.

참 고 문 헌

[1] 김철혁; 정보은닉기술, 과학기술출판사, 138～160, 주체100(2011).

[2] **김일성**종합대학학보(자연과학), 61, 10, 28, 주체104(2015).

[3] Chandan Kumar Behera; Procedia Computer Science, 70, 757, 2015.

주체107(2018)년 5월 5일 원고접수

Study on a Way of Recovering Identifier Names in smali Code

O Yong Gun, *Kim Kyong Sok*

In this paper, we have presented a method that recovers the identifier name and supports the smali code analysis by defining the analysis module of smali code and building the call-relationship graph and inheritance-relationship graph.

Key words: reverse engineering, obfuscation