(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2016년01월11일
(11) 등록번호 10-1583932
(24) 등록일자 2016년01월04일

(51) 국제특허분류(Int. Cl.)
G06F 21/52 (2013.01) *G06F 21/56* (2013.01)
(21) 출원번호 10-2014-0060322
(22) 출원일자 2014년05월20일
심사청구일자 2014년05월20일
(65) 공개번호 10-2015-0133498
(43) 공개일자 2015년11월30일
(56) 선행기술조사문헌
KR1020120078018 A*
KR1020120073018 A
JP2004236288 A
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
한양대학교 산학협력단
서울특별시 성동구 왕십리로 222(행당동, 한양대학교내)
(72) 발명자
김태근
경기도 수원시 영통구 동탄원천로 1089 금성아파트 104동 505호
임을규
서울특별시 서초구 신반포로 9 주공아파트 2동 105호
(74) 대리인
특허법인 무한

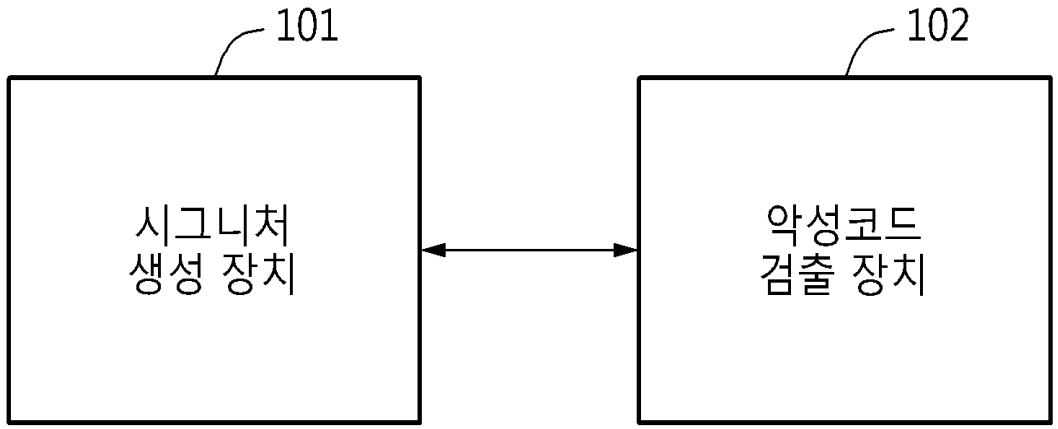
전체 청구항 수 : 총 13 항 심사관 : 구본재

(54) 발명의 명칭 **프로그램의 시그니처를 생성하는 시그니처 생성 장치 및 방법, 시그니처의 악성 코드를 검출하는 악성 코드 검출 장치 및 방법**

(57) 요 약

프로그램에 포함된 사용자 지정 함수의 호출 관계를 나타내는 호출 그래프를 이용하여 기호 실행 분석을 통해 사용자 지정 함수마다 생성된 기호 공식을 병합하여, 상기 프로그램의 특징을 나타내는 시그니처를 생성하는 시그니처 생성 장치 및 방법과 프로그램의 특징을 나타내는 시그니처와 악성 코드의 특징을 나타내는 시그니처의 유사성 분석을 통해 악성 코드를 검출하는 악성 코드 검출 장치 및 방법에 관한 것이다.

대 표 도 - 도1



명 세 서

청구범위

청구항 1

프로그램에 포함된 사용자 지정 함수의 호출 관계를 나타내는 호출 그래프를 생성하는 단계;

상기 호출 그래프를 구성하는 사용자 지정 함수마다 기호 실행 분석을 수행하여 기호 공식을 생성하는 단계; 및

상기 호출 그래프에 포함된 사용자 지정 함수마다 생성된 기호 공식을 병합하여, 상기 프로그램의 특징을 나타내는 시그니처를 생성하는 단계

를 포함하는 시그니처 생성 방법.

청구항 2

제1항에 있어서,

상기 호출 그래프를 생성하는 단계는,

상기 프로그램에 포함된 사용자 지정 함수 및 상기 사용자 지정 함수로부터 파생된 자식 함수를 포함하는 호출 그래프를 생성하는 시그니처 생성 방법.

청구항 3

제1항에 있어서,

상기 호출 그래프를 생성하는 단계는,

상기 프로그램에 포함된 코드 섹션을 역어셈블하여 함수 코드를 추출하고, 상기 추출한 함수 코드로부터 컴파일러 생산 함수, 라이브러리 함수 및 API 함수를 제외한 나머지 사용자 지정 함수를 추출하는 시그니처 생성 방법.

청구항 4

제1항에 있어서,

상기 기호 공식을 생성하는 단계는,

상기 사용자 지정 함수에 포함된 호출 변수를 이용하여 프로그램의 실행 흐름을 분석하는 기호 실행 분석을 수행하는 시그니처 생성 방법.

청구항 5

제1항에 있어서,

상기 시그니처를 생성하는 단계는,

상기 호출 관계에 따라 사용자 지정 함수 및 사용자 지정 함수로부터 파생되는 자식 함수를 호출하는 호출 변수를 사용자 지정 함수의 기호 공식 또는 자식 함수의 호출 구문으로 변환하여 상기 사용자 지정 함수마다 생성된 기호 공식을 병합하는 시그니처 생성 방법.

청구항 6

제1항에 있어서,

상기 시그니처를 생성하는 단계는,

상기 병합된 기호 공식에 입력되는 입력값에 따라 서로 다른 실행 경로를 통해 출력되는 출력 정보를 이용하여 사용자 지정 함수의 특징을 나타내는 시그니처를 생성하는 시그니처 생성 방법.

청구항 7

프로그램에 포함된 사용자 지정 함수의 호출 관계에 이용하여 상기 사용자 지정 함수마다 기호 실행 분석을 수행하는 단계;

상기 기호 실행 분석을 통해 생성된 기호 공식을 병합하여 상기 프로그램의 특징을 나타내는 시그니처를 추출하는 단계;

상기 추출한 시그니처와 비교 대상 프로그램의 시그니처를 비교하여 프로그램의 동작에 관한 유사성을 분석하는 단계;

상기 분석한 유사성의 결과에 따라 상기 추출한 시그니처에 포함된 악성 코드를 검출하는 단계

를 포함하는 악성 코드 검출 방법.

청구항 8

삭제

청구항 9

프로그램에 포함된 사용자 지정 함수의 호출 관계를 나타내는 호출 그래프를 생성하는 호출 그래프 생성부;

상기 호출 그래프를 구성하는 사용자 지정 함수마다 기호 실행 분석을 수행하여 기호 공식을 생성하는 기호 공식 생성부; 및

상기 호출 그래프에 포함된 사용자 지정 함수마다 생성된 기호 공식을 병합하여, 상기 프로그램의 특징을 나타내는 시그니처를 생성하는 시그니처 생성부

를 포함하는 시그니처 생성 장치.

청구항 10

제9항에 있어서,

상기 호출 그래프 생성부는,

상기 프로그램에 포함된 사용자 지정 함수 및 상기 사용자 지정 함수로부터 파생된 자식 함수를 포함하는 호출 그래프를 생성하는 시그니처 생성 장치.

청구항 11

제9항에 있어서,

상기 호출 그래프 생성부는,

상기 프로그램에 포함된 코드 섹션을 역어셈블하여 함수 코드를 추출하고, 상기 추출한 함수 코드로부터 컴파일러 생산 함수, 라이브러리 함수 및 API 함수를 제외한 나머지 사용자 지정 함수를 추출하는 시그니처 생성 장치.

청구항 12

제9항에 있어서,

상기 기호 공식 생성부는,

상기 사용자 지정 함수에 포함된 호출 변수를 이용하여 프로그램의 실행 흐름을 분석하는 기호 실행 분석을 수행하는 시그니처 생성 장치.

청구항 13

제9항에 있어서,

상기 시그니처 생성부는,

상기 호출 관계에 따라 사용자 지정 함수 및 사용자 지정 함수로부터 파생되는 자식 함수를 호출하는 호출 변수를 사용자 지정 함수의 기호 공식 또는 자식 함수의 호출 구문으로 변환하여 상기 사용자 지정 함수마다 생성된 기호 공식을 병합하는 시그니처 생성 장치.

청구항 14

제9항에 있어서,

상기 시그니처 생성부는,

상기 병합된 기호 공식에 입력되는 입력값에 따라 서로 다른 실행 경로를 통해 출력되는 출력 정보를 이용하여 사용자 지정 함수의 특징을 나타내는 시그니처를 생성하는 시그니처 생성 장치.

발명의 설명

기 술 분 야

[0001] 아래의 설명은 장치 및 방법에 관한 것으로, 사용자 지정 함수를 포함하는 프로그램의 시그니처를 생성하고, 생성된 시그니처 내에 악성 코드를 검출하는 장치 및 방법들에 관한 것이다.

배 경 기 술

[0002] 종래의 프로그램 내에 존재하는 악성 코드를 식별하기 위한 악성 코드 식별 방법은 분석이 요구되는 프로그램에 대하여 정적인 분석 방법 또는 동적인 분석 방법을 수행하여 악성 코드를 식별한다.

[0003] 여기서, 정적인 분석 방법은 프로그램을 실행하지 않고, 프로그램의 특징으로 시그니처를 이용하여 프로그램을 분석한다. 정적인 분석 방법은 프로그램을 구성하는 문법적 특징으로부터 시그니처를 생성한다. 그리고, 정적인 분석 방법은 프로그램의 실행여부와 관계없이 프로그램의 정적인 정보와 악성 코드의 정적인 정보를 비교하여 각 정보 간의 유사성을 기반으로 악성 코드에 대한 존재 여부를 판단한다.

[0004] 이 때, 정적인 분석 방법은 미리 알려진 악성 코드의 정적인 정보를 이용하여 유사성 검사를 수행하기 때문에 악성 코드를 개발하는 개발자들에 의해 쉽게 우회되는 문제가 존재한다. 또한, 정적인 분석 방법은 프로그램의 실행 행위 정보를 이용하여 악성 코드를 식별하는 경우, 프로그램의 실행 행위가 쉽게 바뀌지 않기 때문에 우회가 힘들다.

[0005] 반면, 동적인 분석 방법은 정적인 분석 방법의 문제점을 보완하는 방법으로 프로그램이 행위 분석을 수행하여 악성 코드의 행위 정보를 기반으로 식별이 가능하다. 그러나, 동적인 분석 방법은 프로그램 내에 존재하는 악성 코드를 실제로 실행해야 함에 따라 추가적인 비용이 든다. 구체적으로, 동적인 분석 방법은 프로그램을 실행하기 위해, PC의 바이러스 감염을 막기 위한 분석 환경, PC의 자원 및 실행에 따른 오버헤드 등의 추가적인 비용이 든다.

[0006] 특히나, 동적인 분석 방법은 정적인 분석 방법과 다르게 상황에 따라 바뀔 수 있는 프로그램의 실행 흐름을 고려하지 못한다.

[0007] 따라서, 정적인 분석 방법 및 동적인 분석 방법을 통한 악성 코드 검출 방법의 한계를 극복하며, 프로그램의 특징을 이용하여 악성 코드를 보다 효율적으로 검출하기 위한 방법이 필요하다.

발명의 내용

해결하려는 과제

[0008] 일실시예에 따라 프로그램에 포함된 사용자 지정 함수 및 사용자 지정 함수로부터 파생되는 자식함수를 이용하여 기호 실행 분석을 수행함으로써, 프로그램을 직접적으로 실행시키지 않고, 프로그램의 실행 흐름 및 프로그램의 행위를 분석하는 시그니처 생성 장치 및 방법을 제공할 수 있다.

[0009] 일실시예에 따라 사용자 지정 함수마다 수행한 기호 실행 분석을 통해 프로그램의 특징을 나타내는 시그니처를 생성함으로써, 시그니처를 통해 프로그램의 제한된 환경에서 실행되는 호출문, 어셈블리 코드 분석 등에 대하여 보다 정확한 분석 결과를 갖는 시그니처 생성 장치 및 방법을 제공할 수 있다.

[0010] 일실시예에 따라 프로그램의 특징을 나타내는 시그니처와 악성 코드의 특징을 나타내는 시그니처를 비교하여 유

사성 여부에 따라 프로그램의 특징을 나타내는 시그니처에 포함된 악성 코드를 추출함으로써, 정적 분석에 의한 악성 코드 회피 또는 동적 분석에 의한 바이러스 감염, 이벤트 트리거 등의 악성 코드를 검출하기 위해 발생하는 문제점을 최소화하는 악성 코드 검출 장치 및 방법을 제공할 수 있다.

과제의 해결 수단

[0011] 일실시예에 따른 시그니처 생성 방법은 프로그램에 포함된 사용자 지정 함수의 호출 관계를 나타내는 호출 그래프를 생성하는 단계; 상기 호출 그래프를 구성하는 사용자 지정 함수마다 기호 실행 분석을 수행하여 기호 공식을 생성하는 단계; 및 상기 호출 그래프에 포함된 사용자 지정 함수마다 생성된 기호 공식을 병합하여, 상기 프로그램의 특징을 나타내는 시그니처를 생성하는 단계를 포함할 수 있다.

[0012] 일실시예에 따른 호출 그래프를 생성하는 단계는 상기 프로그램에 포함된 사용자 지정 함수 및 상기 사용자 지정 함수로부터 파생된 자식 함수를 포함하는 호출 그래프를 생성할 수 있다.

[0013] 일실시예에 따른 호출 그래프를 생성하는 단계는 상기 프로그램에 포함된 코드 섹션을 역어셈블하여 함수 코드를 추출하고, 상기 추출한 함수 코드로부터 컴파일러 생산 함수, 라이브러리 함수 및 API 함수를 제외한 나머지 사용자 지정 함수를 추출할 수 있다.

[0014] 일실시예에 따른 기호 공식을 생성하는 단계는 상기 사용자 지정 함수에 포함된 호출 변수를 이용하여 프로그램의 실행 흐름을 분석하는 기호 실행 분석을 수행할 수 있다.

[0015] 일실시예에 따른 시그니처를 생성하는 단계는 상기 호출 관계에 따라 사용자 지정 함수 및 사용자 지정 함수로부터 파생되는 자식 함수를 호출하는 호출 변수를 사용자 지정 함수의 기호 공식 또는 자식 함수의 호출 구문으로 변환하여 상기 사용자 지정 함수마다 생성된 기호 공식을 병합할 수 있다.

[0016] 일실시예에 따른 시그니처를 생성하는 단계는 상기 병합된 기호 공식에 입력되는 입력값에 따라 서로 다른 실행 경로를 통해 출력되는 출력 정보를 이용하여 사용자 지정 함수의 특징을 나타내는 시그니처를 생성할 수 있다.

[0017] 일실시예에 따른 악성 코드 검출 방법은 프로그램의 특징을 나타내는 시그니처를 추출하는 단계; 상기 추출한 시그니처와 악성 코드의 특징을 나타내는 시그니처를 비교하여 프로그램의 동작에 관한 유사성을 분석하는 단계; 상기 분석한 유사성의 결과에 따라 상기 추출한 시그니처에 포함된 악성 코드를 검출하는 단계를 포함할 수 있다.

[0018] 일실시예에 따른 시그니처를 추출하는 단계는 상기 프로그램에 포함된 사용자 지정 함수의 호출 관계에 따라 사용자 지정 함수마다 기호 실행 분석을 수행하고, 상기 기호 실행 분석을 통해 생성된 기호 공식을 병합한 시그니처를 추출할 수 있다.

[0019] 일실시예에 따른 시그니처 생성 장치는 프로그램에 포함된 사용자 지정 함수의 호출 관계를 나타내는 호출 그래프를 생성하는 호출 그래프 생성부; 상기 호출 그래프를 구성하는 사용자 지정 함수마다 기호 실행 분석을 수행하여 기호 공식을 생성하는 기호 공식 생성부; 및 상기 호출 그래프에 포함된 사용자 지정 함수마다 생성된 기호 공식을 병합하여, 상기 프로그램의 특징을 나타내는 시그니처를 생성하는 시그니처 생성부를 포함할 수 있다.

[0020] 일실시예에 따른 호출 그래프 생성부는 상기 프로그램에 포함된 사용자 지정 함수 및 상기 사용자 지정 함수로부터 파생된 자식 함수를 포함하는 호출 그래프를 생성할 수 있다.

[0021] 일실시예에 따른 호출 그래프 생성부는 상기 프로그램에 포함된 코드 섹션을 역어셈블하여 함수 코드를 추출하고, 상기 추출한 함수 코드로부터 컴파일러 생산 함수, 라이브러리 함수 및 API 함수를 제외한 나머지 사용자 지정 함수를 추출할 수 있다.

[0022] 일실시예에 따른 기호 공식 생성부는 상기 사용자 지정 함수에 포함된 호출 변수를 이용하여 프로그램의 실행 흐름을 분석하는 기호 실행 분석을 수행할 수 있다.

[0023] 일실시예에 따른 시그니처 생성부는 상기 호출 관계에 따라 사용자 지정 함수 및 사용자 지정 함수로부터 파생되는 자식 함수를 호출하는 호출 변수를 사용자 지정 함수의 기호 공식 또는 자식 함수의 호출 구문으로 변환하여 상기 사용자 지정 함수마다 생성된 기호 공식을 병합할 수 있다.

[0024] 일실시예에 따른 시그니처 생성부는 상기 병합된 기호 공식에 입력되는 입력값에 따라 서로 다른 실행 경로를 통해 출력되는 출력 정보를 이용하여 사용자 지정 함수의 특징을 나타내는 시그니처를 생성할 수 있다.

[0025] 일실시예에 따른 악성 코드 검출 장치는 프로그램의 특징을 나타내는 시그니처를 추출하는 시그니처 추출부; 상기 추출한 시그니처와 악성 코드의 특징을 나타내는 시그니처를 비교하여 프로그램의 동작에 관한 유사성을 분석하는 유사성 분석부; 상기 분석한 유사성의 결과에 따라 상기 추출한 시그니처에 포함된 악성 코드를 검출하는 악성 코드 검출부를 포함할 수 있다.

[0026] 일실시예에 따른 시그니처 추출부는 상기 프로그램에 포함된 사용자 지정 함수의 호출 관계에 따라 사용자 지정 함수마다 기호 실행 분석을 수행하고, 상기 기호 실행 분석을 통해 생성된 기호 공식을 병합한 시그니처를 추출할 수 있다.

발명의 효과

[0027] 시그니처 생성 장치 및 방법은 프로그램에 포함된 사용자 지정 함수 및 사용자 지정 함수로부터 파생되는 자식 함수를 이용하여 기호 실행 분석을 수행함으로써, 프로그램을 직접적으로 실행시키지 않고, 프로그램의 실행 흐름 및 프로그램의 행위를 분석할 수 있다.

[0028] 시그니처 생성 장치 및 방법은 사용자 지정 함수마다 수행한 기호 실행 분석을 통해 프로그램의 특징을 나타내는 시그니처를 생성함으로써, 시그니처를 통해 프로그램의 제한된 환경에서 실행되는 호출문, 어셈블리 코드 분석 등에 대하여 보다 정확한 분석 결과를 가질 수 있다.

[0029] 악성 코드 검출 장치 및 방법은 프로그램의 특징을 나타내는 시그니처와 악성 코드의 특징을 나타내는 시그니처를 비교하여 유사성 여부에 따라 프로그램의 특징을 나타내는 시그니처에 포함된 악성 코드를 추출함으로써, 정적 분석에 의한 악성 코드 회피 또는 동적 분석에 의한 바이러스 감염, 이벤트 트리거 등의 악성 코드를 검출하기 위해 발생하는 문제점을 최소화할 수 있다.

도면의 간단한 설명

[0030] 도 1은 일실시예에 따른 시그니처 생성 장치 및 악성 코드 검출 장치를 도시한 도면이다.

도 2는 일실시예에 따른 시그니처 생성 장치 및 악성 코드 검출 장치의 세부 구성을 도시한 도면이다.

도 3은 일실시예에 따른 사용자 지정 함수의 호출 관계를 나타내는 호출 그래프를 도시한 도면이다.

도 4는 일실시예에 따른 복수의 사용자 지정 함수를 하나의 사용자 지정 함수로 병합하는 도면이다.

도 5는 일실시예에 따른 기호 공식을 이용하여 서로 다른 실행 경로에 따라 출력되는 출력 정보에 대하여 시그니처화하는 도면이다.

도 6은 일실시예에 따른 시그니처 생성 방법을 도시한 도면이다.

도 7은 일실시예에 따른 악성 코드 검출 방법을 도시한 도면이다.

발명을 실시하기 위한 구체적인 내용

[0031] 이하, 본 발명의 실시예를 첨부된 도면을 참조하여 상세하게 설명한다.

[0032] 도 1은 일실시예에 따른 시그니처 생성 장치 및 악성 코드 검출 장치를 도시한 도면이다.

[0033] 도 1을 참고하면, 시그니처 생성 장치(101)는 기호 실행 분석을 통해 프로그램의 특징을 나타내는 시그니처를 생성할 수 있다. 구체적으로, 시그니처 생성 장치(101)는 프로그램에 포함된 사용자 지정 함수마다 기호 실행 분석을 수행하고, 기호 실행 분석을 수행한 결과로 생성되는 기호 공식을 이용하여 프로그램의 시그니처를 생성할 수 있다. 여기서, 기호 실행 분석은 사용자 지정 함수에 포함된 호출 변수를 이용하여 프로그램의 실행 흐름을 분석하는 방법일 수 있다.

[0034] 또한, 시그니처 생성 장치(101)는 추출한 사용자 지정 함수를 기준으로 사용자 지정 함수의 호출 관계를 나타내는 호출 그래프를 생성하여 프로그램의 시그니처를 생성할 수 있다. 다시 말해, 시그니처 생성 장치(101)는 프로그램에 포함된 사용자 지정 함수의 호출 그래프를 생성하기 위해 다음과 같은 과정을 수행할 수 있다. 시그니처 생성 장치(101)는 프로그램을 실행할 수 있는 실행 파일을 입력 받아, 역어셈블 모듈에 의해 실행 파일의 바이트 시퀀스로 표현된 코드 섹션을 명령어 시퀀스로 해석할 수 있다. 그리고, 시그니처 생성 장치(101)는 명령어 시퀀스 중 컴파일러 생산 함수 및 라이브러리 함수들을 추출할 수 있다.

[0035] 그리고, 시그니처 생성 장치(101)는 추출한 컴파일러 생산 함수 및 라이브러리 함수를 제외한 나머지 사용자 지

정 함수를 추출하여 사용자 지정 함수를 루트 노드로 갖는 호출 그래프를 생성할 수 있다. 여기서, 호출 그래프는 사용자 지정 함수의 호출 관계를 나타내는 것으로 사용자 지정 함수 및 사용자 지정 함수로부터 파생되는 자식함수의 관계를 나타낼 수 있다.

[0036] 시그니처 생성 장치(101)는 호출 그래프를 구성하는 사용자 지정 함수마다 실행 기호 분석을 수행하여 생성된 기호 공식을 호출 그래프의 호출 관계에 따라 하나로 병합할 수 있다. 시그니처 생성 장치(101)는 하나로 병합된 기호 공식을 기반으로 프로그램의 서로 다른 실행 경로에 따른 출력 정보를 추출하여 프로그램의 특징을 나타내는 시그니처를 생성할 수 있다.

[0037] 그리고, 시그니처 생성 장치(101)를 통해 생성된 시그니처는 악성 코드 검출 장치(102)를 통해 프로그램에 포함된 악성 코드가 추출될 수 있다. 악성 코드 검출 장치(102)는 유사성을 판단하고자 하는 프로그램의 시그니처를 입력 받을 수 있다. 일례로, 악성 코드 검출 장치(102)는 시그니처 생성 장치(101)로부터 생성된 시그니처를 추출하거나, 미리 생성된 시그니처를 입력받을 수 있다.

[0038] 그리고, 악성 코드 검출 장치(102)는 입력 받은 프로그램에 포함된 사용자 지정 함수에서 사용되는 변수의 개수 등을 고려하여 유사한 프로토콜 타입을 갖는 비교 대상 함수를 추출할 수 있다.

[0039] 악성 코드 검출 장치(102)는 사용자 지정 함수와 비교 대상 함수 간에 유사성을 판단하기 위해 동일한 값을 갖는 변수 입력값을 생성할 수 있다. 이 때, 악성 코드 검출 장치(102)는 사용자 지정 함수와 비교 대상 함수에서 사용되는 변수의 개수와 실행 경로를 고려하여 변수 입력값을 생성할 수 있다.

[0040] 악성 코드 검출 장치(102)는 생성한 변수 입력값을 사용자 지정 함수의 기호 공식 및 비교 대상 함수의 기호 공식에 대입할 수 있다. 그리고, 악성 코드 검출 장치(102)는 사용자 지정 함수의 기호 공식 및 비교 대상 함수의 기호 공식에 대입한 변수 입력값에 대한 시그니처를 비교할 수 있다. 즉, 악성 코드 검출 장치(102)는 변수 입력값에 대응하여 생성된 시그니처를 비교함으로써, 두 함수 간의 실행과 관련된 유사성을 판단할 수 있다.

[0041] 그리고, 악성 코드 검출 장치(102)는 유사성 결과에 따라 사용자 지정 함수에 포함된 악성 코드를 추출할 수 있다.

[0042] 도 2는 일실시예에 따른 시그니처 생성 장치 및 악성 코드 검출 장치의 세부 구성을 도시한 도면이다.

[0043] 도 2를 참고하면, 시그니처 생성 장치(201)는 호출 그래프 생성부(202), 기호 공식 생성부(203), 시그니처 생성부(204)를 포함할 수 있다.

[0044] 호출 그래프 생성부(202)는 프로그램에 포함된 사용자 지정 함수의 호출 관계를 나타내는 호출 그래프를 생성할 수 있다. 프로그램은 라이브러리 함수, API 함수, 컴파일러가 생산하는 추가적인 함수, 개발자가 직접 구현한 사용자 지정 함수 등을 포함할 수 있다.

[0045] 호출 그래프 생성부(202)는 프로그램의 실행 파일을 역어셈블하여 실행 파일에 바이트 시퀀스로 표현된 모든 코드 섹션을 추출할 수 있다. 이 때, 코드 섹션은 함수 단위로 구성될 수 있다. 그리고, 호출 그래프 생성부(202)는 추출한 코드 섹션을 명령서 시퀀스로 해석하고, 해석한 명령어 시퀀스를 함수 단위로 분할할 수 있다.

[0046] 호출 그래프 생성부(202)는 함수 단위로 분할된 함수들 중 사용자 지정 함수를 추출할 수 있다. 여기서, 사용자 지정 함수는 프로그램을 개발하는 과정에서 특정 행위를 수행하도록 개발자에 의해 생성된 함수로써, 일반적으로 라이브러리 함수, API 함수들과 달리 컴파일되는 과정에서 오류가 발생할 수 있다. 다시 말해, 사용자 지정 함수는 개발자에 의해 생성되기 때문에 개발자의 코딩 실수로 인하여 프로그램 전반에 악의적인 행위를 수행하는 코드가 포함될 수 있다.

[0047] 따라서, 호출 그래프 생성부(202)는 프로그램에 악의적인 행위를 담당하는 사용자 지정 함수에 대한 기호 실행 분석을 수행하기 위해 프로그램에 포함된 사용자 지정 함수를 추출할 수 있다.

[0048] 호출 그래프 생성부(202)는 추출한 사용자 지정 함수의 호출 관계를 고려하여 호출 그래프를 생성할 수 있다. 여기서, 호출 그래프 생성부(202)는 사용자 지정 함수 및 상기 사용자 지정 함수로부터 파생된 자식 함수를 포함하는 호출 그래프를 생성할 수 있다.

[0049] 기호 공식 생성부(203)는 호출 그래프를 구성하는 사용자 지정 함수마다 기호 실행 분석을 수행하여 기호 실행 분석을 수행한 결과로써 기호 공식을 생성할 수 있다. 기호 공식 생성부(203)는 사용자 지정 함수에 포함된 호출 변수를 이용하여 사용자 지정 함수마다 실행 흐름에 따른 기호 공식을 생성할 수 있다. 일례로, 기호 공식 생성부(203)는 사용자 지정 함수의 호출 관계를 나타내는 매개 변수를 이용하여 사용자 지정 함수의 처리 동작

에 대한 흐름을 파악하고, 사용자 지정 함수에 대응하여 기호값을 갖는 기호 공식을 생성할 수 있다.

[0050] 시그니처 생성부(204)는 사용자 지정 함수마다 생성된 기호 공식을 병합하여 프로그램의 특징을 나타내는 시그니처를 생성할 수 있다. 구체적으로, 시그니처 생성부(204)는 사용자 지정 함수의 호출 관계에 따라 사용자 지정 함수의 호출 변수를 기호 공식으로 대체할 수 있다. 다시 말해, 시그니처 생성부(204)는 서로 다른 사용자 지정 함수 또는 사용자 지정 함수로부터 파생되는 자식 함수를 호출하는 호출 변수에 대응하여 호출된 사용자 지정 함수의 기호 공식 또는 자식 함수의 호출 구문으로 대체할 수 있다.

[0051] 일례로, 시그니처 생성부(204)는 부모 노드가 자식 노드를 호출하는 공식의 구문에서 자식 노드를 호출하는 구문을 자식 노드의 기호 공식으로 대체할 수 있다. 즉, 시그니처 생성부(204)는 사용자 지정 함수의 호출 관계에 따라 호출하는 사용자 지정 함수 또는 자식 노드를 나타내는 공식으로 대체할 수 있다. 여기서, 시그니처 생성부(204)는 사용자 지정 함수인 경우, 기호 실행 분석이 수행된 결과에 대한 기호 공식으로 대체할 수 있다. 그리고, 시그니처 생성부(204)는 자식 함수인 경우, 자식 함수를 호출했음을 의미하는 호출 구문으로 대체할 수 있다. 여기서, 자식 함수는 사용자 지정 함수가 호출하는 사용자 지정 함수 이외에 나머지 함수로 라이브러리 함수, API 함수, 시스템 함수 등을 포함할 수 있다.

[0052] 결국, 시그니처 생성부(204)는 호출 관계에 따라 사용자 지정 함수의 호출 변수를 사용자 지정 함수의 기호 공식 또는 자식 함수의 호출 구문으로 대체함으로써, 사용자 지정 함수마다 생성된 기호 공식을 하나의 공식으로 병합할 수 있다.

[0053] 여기서, 기호 공식은 변수로 입력되는 입력값에 따라 실행 가능한 모든 코드의 경로를 파악할 수 있다. 따라서, 시그니처 생성부(204)는 입력값을 고려하여 병합된 사용자 지정 함수의 기호 공식의 모든 실행 경로를 파악할 수 있다. 그리고, 시그니처 생성부(204)는 각각의 실행 경로에 따라 발생되는 출력 정보를 수집할 수 있다. 시그니처 생성부(204)는 수집한 출력 정보를 통합하여 프로그램의 시그니처를 생성할 수 있다. 여기서, 출력 정보는 입력값에 고려하여 실행 경로에 따라 실행 가능한 메모리와 관련된 연산, 사칙 연산, 함수 호출 연산 등의 시퀀스 일 수 있다.

[0054] 시그니처 생성 장치는 프로그램의 실행 없이 정적으로 프로그램의 특징을 나타내는 시그니처를 생성하는 장치로써, 기호 실행 분석을 이용하여 프로그램의 실행 흐름 및 행위에 기초한 프로그램의 시그니처를 생성할 수 있다.

[0055] 악성 코드 검출 장치(205)는 프로그램의 시그니처를 비교하여 프로그램에 포함된 악성 코드를 검출할 수 있다. 악성 코드 검출 장치(205)는 시그니처 추출부(206), 유사성 분석부(207), 악성 코드 검출부(208)을 포함할 수 있다.

[0056] 시그니처 추출부(206)는 악성 코드의 존재 여부를 파악하기 위한 프로그램의 특징을 나타내는 시그니처를 추출할 수 있다. 일례로, 시그니처 추출부(206)는 시그니처 생성 장치(201)를 이용하여 생성된 프로그램의 시그니처 또는 외부 메모리에 미리 저장된 프로그램의 시그니처를 추출할 수 있다.

[0057] 유사성 분석부(207)는 추출한 시그니처와 비교 대상 프로그램의 시그니처를 비교하여 프로그램의 행위에 관한 유사성을 분석할 수 있다. 여기서, 비교 대상 프로그램은 악성 코드의 존재 여부를 파악하기 위한 프로그램에 포함된 사용자 정의 함수의 구성과 유사한 구성을 갖는 프로그램일 수 있다. 일례로, 비교 대상 프로그램과 악성 코드의 존재 여부를 파악하기 위한 프로그램은 각각 4개의 매개 변수를 가지며, 사칙연산과 관련된 함수를 포함하는 프로그램일 수 있다.

[0058] 그리고, 악성 코드 검출부(208)는 두 시그니처 간의 유사성을 기반으로 악성 코드 존재 유무를 판단할 수 있다. 즉, 악성 코드 검출부(208)는 각각의 시그니처 가 나타내는 기호 공식에 입력되는 입력값에 따라 서로 다른 실행 경로를 통해 출력되는 출력 정보를 비교할 수 있다. 그리고, 악성 코드 검출부(208)는 비교 결과에 따른 유사성 여부에 따라 시그니처에 포함된 악성 코드를 검출 할 수 있다.

[0059] 도 3은 일실시예에 따른 사용자 지정 함수의 호출 관계를 나타내는 호출 그래프를 도시한 도면이다.

[0060] 도 3을 참고하면, 호출 그래프는 사용자 지정 함수(301), (302), (303), (304) 및 사용자 지정 함수(302), (303), (304)로부터 파생되는 자식 함수(305), (306)를 포함할 수 있다. 호출 그래프는 각각의 사용자 지정 함수(301), (302), (303), (304) 및 자식 함수(305), (306)를 간선으로 연결하여 나타낼 수 있다. 호출 그래프는 함수를 연결하는 간선을 통해 방향성을 가지며, 함수 내 서로 다른 함수로의 호출 관계를 나타낼 수 있다. 여기서, 간선의 시작점은 함수를 호출하는 함수 노드이며, 간선의 도착점은 함수가 호출되는 피 호출 함수 노드

일 수 있다.

[0061] 호출 그래프는 사용자 지정 함수(301), (302), (303), (304)의 호출 관계에 따라 생성될 수 있다. 이에 따라, 사용자 지정 함수(301)은 루트 노드일 수 있다. 그리고, 호출 그래프는 사용자 지정 함수(301)를 기준으로 파생되는 서로 다른 사용자 지정 함수(302), (303), (304) 간의 호출 관계를 나타낼 수 있다.

[0062] 여기서, 프로그램에 포함된 함수에는 사용자 지정 함수(301), (302), (303), (304) 이외에 라이브러리 함수, API 함수, 시스템 함수 등 다양한 함수를 포함할 수 있다. 그리고, 사용자 지정 함수(301), (302), (303), (304)는 라이브러리 함수, API 함수, 시스템 함수 등 다양한 함수를 호출할 수 있다.

[0063] 이 때, 호출 그래프는 사용자 지정 함수(302), (303), (304)로부터 파생되는 라이브러리 함수, API 함수, 시스템 함수 등에 대한 자식 함수(305), (306)까지만 나타낼 수 있다. 다시 말해, 호출 그래프는 사용자 지정 함수(302), (303), (304)로부터 파생된 자식 함수(305), (306)에 존재하는 다른 함수에 대한 호출 관계는 나타내지 않는다.

[0064] 일례로, 호출 그래프는 라이브러리 함수, API 함수의 경우, 자식 함수(305), (306) 안에 다른 함수로의 호출이 존재하더라도 더 이상 탐색하지 않고 바로 자식 함수(305), (306)를 리프노드, 즉 최하위노드로 나타낼 수 있다.

[0065] 또한, 호출 그래프는 사용자 지정 함수(301), (302), (303), (304)의 경우, 함수 호출이 포함되지 않음에 따라 해당 사용자 지정 함수(301), (302), (303), (304)를 리프 노드로 나타낼 수 있다.

[0066] 결과적으로, 호출 그래프는 사용자 지정 함수(301), (302), (303), (304)마다 포함된 명령어 시퀀스에 기초하여 호출되는 함수 및 호출되는 함수가 호출하는 다른 함수에 대한 호출 관계를 나타낼 수 있다. (단, 사용자 지정 함수로부터 파생되는 자식 함수만을 나타낼 수 있다.)

[0067] 시그니처 생성 장치는 보다 정확한 호출 그래프를 생성하기 위해 재귀적으로 사용자 지정 함수(301), (302), (303), (304) 및 자식 함수(305), (306)에 대한 호출 관계를 찾기 위한 탐색을 수행할 수 있다. 시그니처 생성 장치는 라이브러리 함수나 API 함수, 혹은 사용자 지정 함수이지만 내부코드에 다른 함수 호출이 없는 함수가 탐색되면 재귀 탐색을 멈추고 해당 함수를 리프 노드로 설정할 수 있다.

[0068] 도 4는 일실시예에 따른 복수의 사용자 지정 함수를 하나의 사용자 지정 함수로 병합하는 도면이다.

[0069] 도 4를 참고하면, 시그니처 생성 장치는 사용자 지정 함수(401), (402) 및 사용자 지정 함수(401), (402)로부터 파생되는 자식 함수(403) 간의 호출 관계를 나타내는 호출 그래프를 포함할 수 있다.

[0070] 이 때, 시그니처 생성 장치는 사용자 지정 함수(401)를 기준으로 사용자 지정 함수(401), (402) 및 사용자 지정 함수(401), (402)로부터 파생되는 자식 함수(403)를 나타내는 공식의 구문을 병합할 수 있다. 다시 말해, 시그니처 생성 장치는 사용자 지정 함수(401), (402) 및 사용자 지정 함수(401), (402)로부터 파생되는 자식 함수(403)를 나타내는 호출 변수를 해당 기호 공식으로 대체하여 하나의 사용자 지정 함수(404)로 병합할 수 있다.

[0071] 여기서, 시그니처 생성 장치는 사용자 지정 함수(401), (402)인 경우, 해당 기호 공식을 이용하여 호출 변수를 대체하고 자식 함수(403)인 경우, 자식 함수를 호출했다는 의미를 가지는 호출 구문(Statement)을 이용하여 호출 변수를 대체할 수 있다.

[0072] 이 때, 시그니처 생성 장치는 병합된 사용자 지정 함수(404)의 기호 공식에 입력되는 입력값에 따라 서로 다른 실행 경로를 파악할 수 있다. 시그니처 생성 장치는 실행 경로에 따라 영향을 주는 변수와 변수의 입력값을 계산할 수 있다. 일례로, 시그니처 생성 장치는 Constraint Solver를 이용하여 실행 경로에 따른 입력값을 계산할 수 있다.

[0073] 그리고, 시그니처 생성 장치는 입력값을 고려하여 서로 다른 실행 경로를 통해 출력되는 출력 정보를 수집할 수 있다. 시그니처 생성 장치는 수집한 출력 정보를 병합하여 사용자 지정 함수(404)의 기능을 정의할 수 있다.

[0074] 도 5는 일실시예에 따른 기호 공식을 이용하여 서로 다른 실행 경로에 따라 출력되는 출력 정보에 대하여 시그니처화하는 도면이다.

[0075] 도 5를 참고하면, 시그니처 생성 장치는 입력값에 따라 서로 다른 실행 경로(501)를 가지는 기호 공식을 이용하여 각 실행 경로에 따라 출력되는 출력 정보(502)를 수집할 수 있다.

[0076] 다시 말해, 시그니처 생성 장치는 사용자 지정 함수에 포함된 공식의 구문에 기초하여 입력 가능한 입력값을 생

성할 수 있다. 여기서 공식의 구문은 프로그램의 실제 동작이 수행되는 구문으로 실행문, 조건문, 반복문을 포함할 수 있다. 그리고, 시그니처 생성 장치는 공식의 구문을 분석하여 수행 조건에 따른 입력값을 생성할 수 있다.

[0077] 시그니처 생성 장치는 공식의 구문에 대응하여 입력 가능한 입력값을 대입하고, 대입된 결과로 출력되는 출력 정보를 수집할 수 있다. 여기서, 출력 정보는 입력값을 이용하는 연산과 관련된 시퀀스일 수 있다. 시그니처 생성 장치는 실행 경로에 따른 시퀀스에 대한 출력 정보를 수집할 수 있다.

[0078] 또한, 시그니처 생성 장치는 라이브러리 함수, API 함수를 호출하는 실행 경로를 포함하는 경우, 라이브러리 함수, API 함수를 호출하는 의미를 갖는 호출 구문으로 출력 정보를 수집할 수 있다.

[0079] 일례로, 사용자 지정 함수는 입력값 a에 따라 4가지 실행 경로(path1, path2, path3, path4)를 포함할 수 있다. 그리고, 4가지 실행 경로에 따른 출력 정보는 서로 다른 정보를 포함할 수 있다. 즉, 출력 정보는 실행 경로에 따라 연산과 관련된 시퀀스 또는 라이브러리 함수, API 함수를 호출하는 의미를 갖는 호출 구문을 포함할 수 있다.

[0080] 시그니처 생성 장치는 기호 공식을 이용하여 입력값에 따라 서로 다른 실행 경로를 통해 출력되는 출력 정보를 수집하고, 실행 경로에 따른 출력 정보를 시그니처에 포함시킴으로써, 보다 정확하게 사용자 지정 함수의 행위적 특성을 시그니처에 반영할 수 있다.

[0081] 도 6은 일실시예에 따른 시그니처 생성 방법을 도시한 도면이다.

[0082] 단계(601)에서 시그니처 생성 장치는 프로그램의 실행 없이 정적으로 프로그램의 특징을 나타내는 시그니처를 생성하기 위한 프로그램의 실행 파일을 입력할 수 있다.

[0083] 단계(602)에서 시그니처 생성 장치는 프로그램으로부터 사용자 지정 함수를 추출하고, 사용자 지정 함수 별로 호출 그래프를 생성할 수 있다. 시그니처 생성 장치는 단계(603)에서 단계(605)과정을 통해 호출 그래프를 생성할 수 있다.

[0084] 단계(603)에서 시그니처 생성 장치는 프로그램에 포함된 코드섹션을 역어셈블하여 함수 코드를 추출할 수 있다. 이 때, 함수 코드는 바이트 시퀀스로 표현되어 함수 단위로 분할될 수 있다.

[0085] 단계(604)에서 시그니처 생성 장치는 함수 코드 내에 사용자 지정 함수에 대한 목록을 추출할 수 있다. 여기서, 시그니처 생성 장치는 단계(606)에서 단계(608)과정을 통해 사용자 지정 함수의 목록을 추출할 수 있다.

[0086] 단계(606) 및 단계(607)에서 시그니처 생성 장치는 프로그램에 포함된 모든 함수 중에서 사용자 지정 함수를 추출하기 위한 전처리 과정을 수행할 수 있다. 다시 말해, 시그니처 생성 장치는 프로그램에 포함된 사용자 지정 함수 이외에 존재하는 라이브러리 함수 또는 컴파일러에 의해 생성되는 컴파일 함수 등에 대하 명령어 코드를 수집하는 전처리 과정을 수행할 수 있다. 이 때, 시그니처 생성 장치는 동일한 컴파일러에 의해 컴파일된 바이너리를 수집하고, 수집한 바이너리로부터 동일하게 나타나는 모든 함수를 추출할 수 있다. 그리고, 시그니처 생성 장치는 추출한 함수들 중 휴리스틱하게 컴파일 생성 함수 또는 라이브러리 함수를 정의할 수 있다.

[0087] 단계(608)에서 시그니처 생성 장치는 전처리 과정을 통해 정의된 컴파일 생성 함수 또는 라이브러리 함수를 제외하고 모든 함수에 포함된 나머지 함수에 대하여 사용자 지정 함수로 정의할 수 있다.

[0088] 단계(609)에서 시그니처 생성 장치는 사용자 지정 함수의 호출 관계를 파악하기 위한 재귀 탐색을 통해 호출 그래프를 생성할 수 있다. 다시 말해, 시그니처 생성 장치는 사용자 지정 함수마다 포함하고 있는 명령어 시퀀스를 검사하여 사용자 지정 함수가 호출하는 함수를 추출하기 위해 재귀적으로 사용자 지정 함수의 호출 관계를 파악할 수 있다. 그리고, 시그니처 생성 장치는 재귀 탐색을 통해 사용자 지정 함수의 호출 관계를 나타내는 호출 그래프를 생성할 수 있다.

[0089] 단계(610)에서 시그니처 생성 장치는 호출 그래프를 구성하는 사용자 지정 함수마다 기호 실행 분석을 수행하여 기호 공식을 생성할 수 있다. 호출 그래프를 구성하는 각 사용자 지정 함수 별로 실행 흐름을 파악하는 기호 실행 분석을 수행하고, 수행한 결과로 기호 공식을 생성할 수 있다.

[0090] 단계(611)에서 시그니처 생성 장치는 사용자 지정 함수마다 생성된 기호 공식을 병합할 수 있다. 시그니처 생성 장치는 단계(612)에서 단계(613)과정을 통해 기호 공식을 병합할 수 있다.

[0091] 단계(612)에서 시그니처 생성 장치는 호출 그래프에 기초하여 사용자 지정 함수에 포함되어 서로 다른 사용자 지정 함수 또는 자식 함수를 호출하는 호출 변수를 탐색할 수 있다. 여기서, 호출 그래프는 방향성을 갖는 간선으로 두 노드가 연결된 형태일 수 있다.

[0092] 일례로, 시그니처 생성 장치는 호출 그래프의 호출 관계에 기초하여 부모 노드가 자식 노드를 호출하는 구문을 탐색할 수 있다.

[0093] 단계(613)에서 시그니처 생성 장치는 탐색한 호출 변수에 대응하여 해당 사용자 지정 함수의 기호 공식 또는 자식 노드의 호출 구문으로 대체하여 기호 공식을 병합할 수 있다.

[0094] 일례로, 시그니처 생성 장치는 간선의 시작점으로 사용자 지정 함수를 부모 노드로 하고, 도착점을 사용자 지정 함수 또는 자식 노드라고 가정할 수 있다. 그리고, 시그니처 생성 장치는 부모 노드의 기호 공식 중 사용자 지정 함수 또는 자식 노드를 호출하는 호출 변수에 대응하여 사용자 지정 함수의 기호 공식 또는 자식 노드의 호출 구문 대체할 수 있다. 이때 시그니처 생성 장치는 병합되는 과정에서 공식이 의미하는 바가 달라지지 않도록 처리해야 한다.

[0095] 단계(614)에서 시그니처 생성 장치는 입력값에 따라 실행경로 별로 출력되는 출력 정보를 수집하여 프로그램의 특징을 나타내는 시그니처를 생성할 수 있다.

[0096] 시그니처 생성 장치는 프로그램을 실행하는데 발생하는 악성 행위를 해결하기 위해 프로그램에 포함된 사용자 지정 함수에 대하여 기호 실행 분석을 수행함으로써, 프로그램을 실행하지 않으면서도 실행 경로에 따른 출력 결과를 보다 정확하게 파악할 수 있다.

[0097] 도 7은 일실시예에 따른 악성 코드 검출 방법을 도시한 도면이다.

[0098] 단계(701)에서 악성 코드 검출 장치는 악성 코드 여부를 검출하고자 하는 시그니처를 입력 받을 수 있다. 그리고, 악성 코드 검출 장치는 시그니처에 대응하여 사용자 지정 함수를 추출할 수 있다. 악성 코드 검출 장치는 사용자 지정 함수의 출력 변수의 개수를 비교하여 동일한 프로토콜 타입의 비교 대상 프로그램의 비교 대상 함수를 추출할 수 있다.

[0099] 단계(702)에서 악성 코드 검출 장치는 사용자 지정 함수 및 비교 대상 함수의 변수로 입력될 입력값을 생성할 수 있다. 이 때, 입력값은 사용자 지정 함수와 비교 대상 함수에 적용 가능한 상수일 수 있다.

[0100] 단계(703)에서 악성 코드 검출 장치는 사용자 지정 함수의 기호 공식 및 비교 대상 함수의 기호 공식에 생성된 입력값을 각각 대입할 수 있다.

[0101] 단계(704)에서 악성 코드 검출 장치는 대입한 입력값에 대응하여 각 함수 별로 출력되는 함수 호출 실행 결과를 비교하여 두 함수 간의 유사성을 판단할 수 있다.

[0102] 단계(705)에서 악성 코드 검출 장치는 유사성 결과에 따라 프로그램에 포함된 악성 코드를 검출할 수 있다.

[0103] 본 발명의 실시 예에 따른 방법들은 다양한 컴퓨터 수단을 통하여 수행될 수 있는 프로그램 명령 형태로 구현되어 컴퓨터 판독 가능 매체에 기록될 수 있다. 상기 컴퓨터 판독 가능 매체는 프로그램 명령, 데이터 파일, 데이터 구조 등을 단독으로 또는 조합하여 포함할 수 있다. 상기 매체에 기록되는 프로그램 명령은 본 발명을 위하여 특별히 설계되고 구성된 것들이거나 컴퓨터 소프트웨어 당업자에게 공지되어 사용 가능한 것일 수도 있다.

[0104] 이상과 같이 본 발명은 비록 한정된 실시예와 도면에 의해 설명되었으나, 본 발명은 상기의 실시예에 한정되는 것은 아니며, 본 발명이 속하는 분야에서 통상의 지식을 가진 자라면 이러한 기재로부터 다양한 수정 및 변형이 가능하다.

[0105] 그러므로, 본 발명의 범위는 설명된 실시예에 국한되어 정해져서는 아니 되며, 후술하는 특허청구범위뿐 아니라 이 특허청구범위와 균등한 것들에 의해 정해져야 한다.

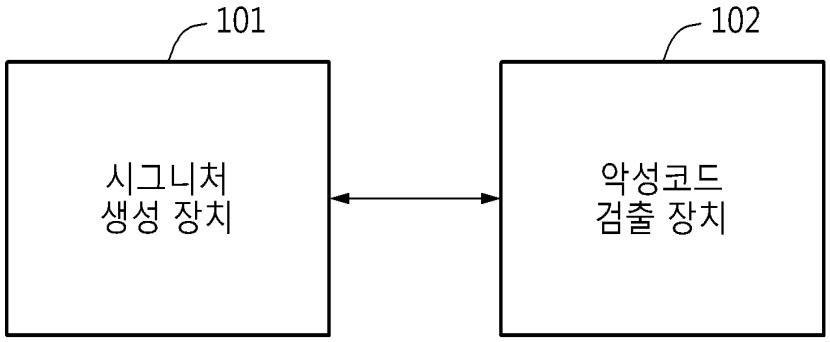
부호의 설명

[0106] 101: 시그니처 생성 장치

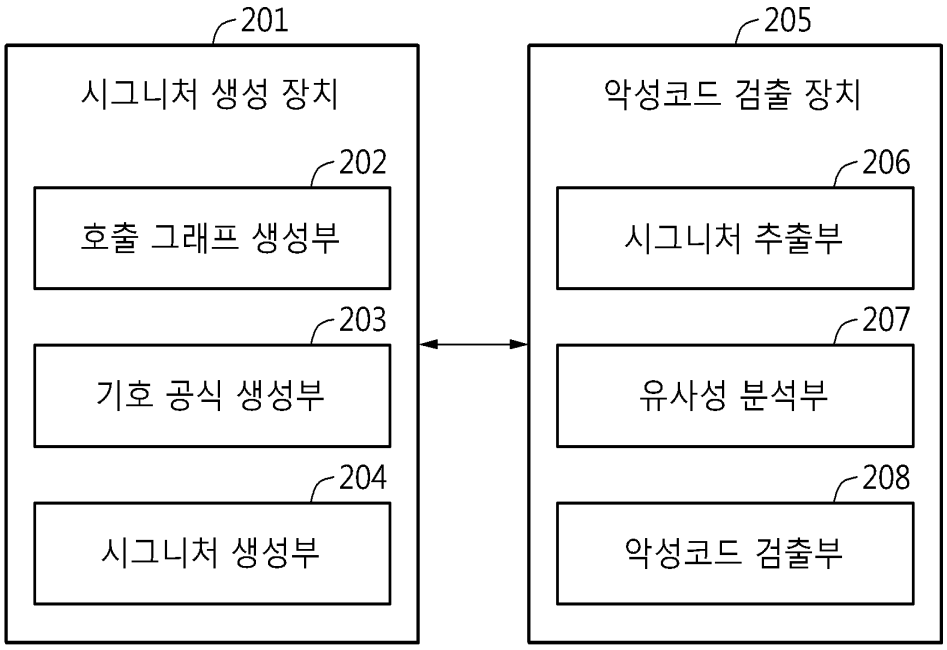
102: 악성 코드 검출 장치

도면

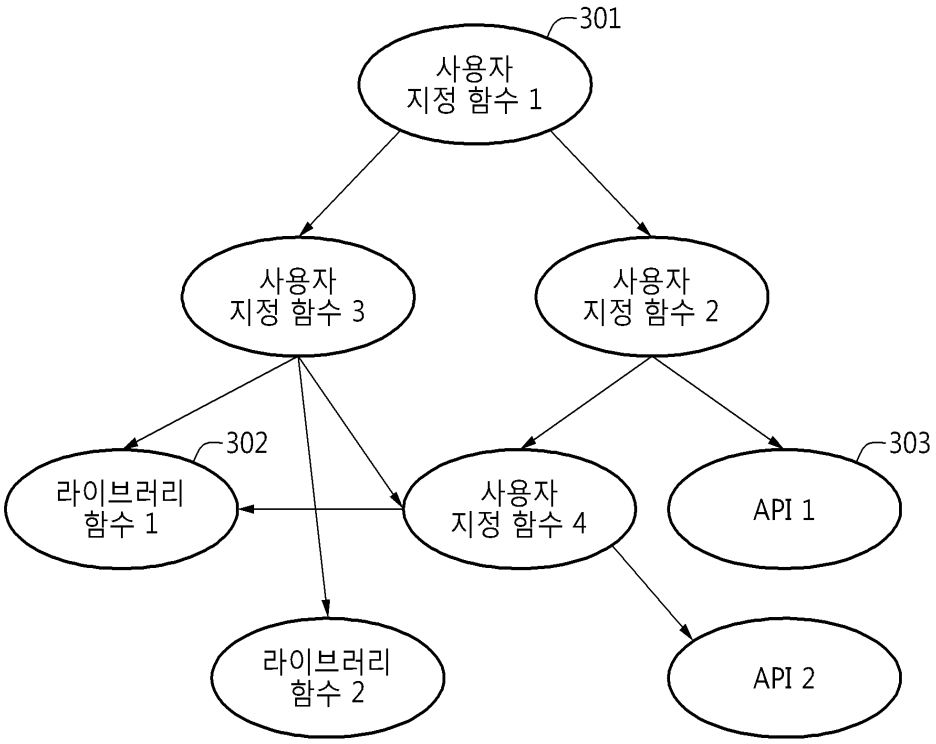
도면1


101
시그니처
생성 장치
102
악성코드
검출 장치

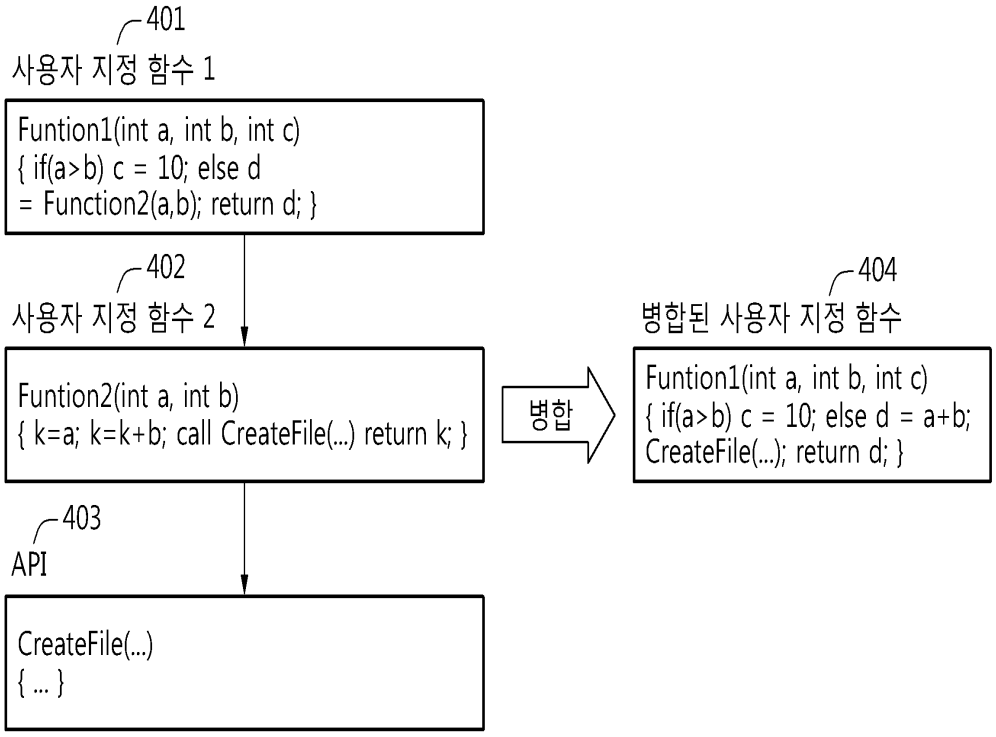
도면2


201
시그니처 생성 장치
202
호출 그래프 생성부
203
기호 공식 생성부
204
시그니처 생성부
205
악성코드 검출 장치
206
시그니처 추출부
207
유사성 분석부
208
악성코드 검출부

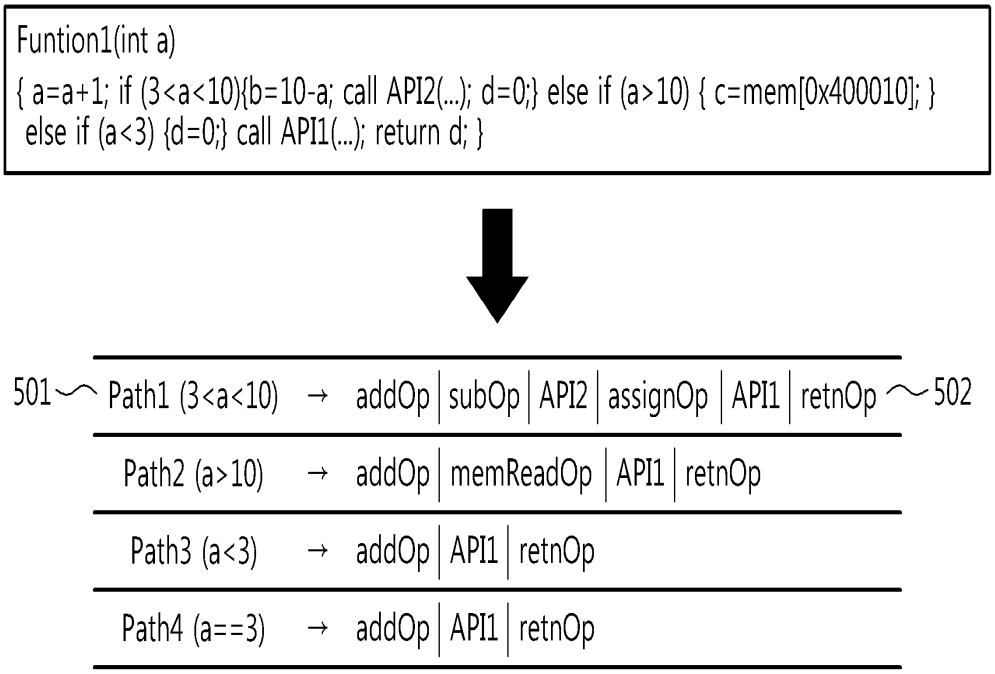
도면3


301
사용자
지정 함수 1
사용자
지정 함수 3
사용자
지정 함수 2
302
라이브러리
함수 1
사용자
지정 함수 4
303
API 1
라이브러리
함수 2
API 2

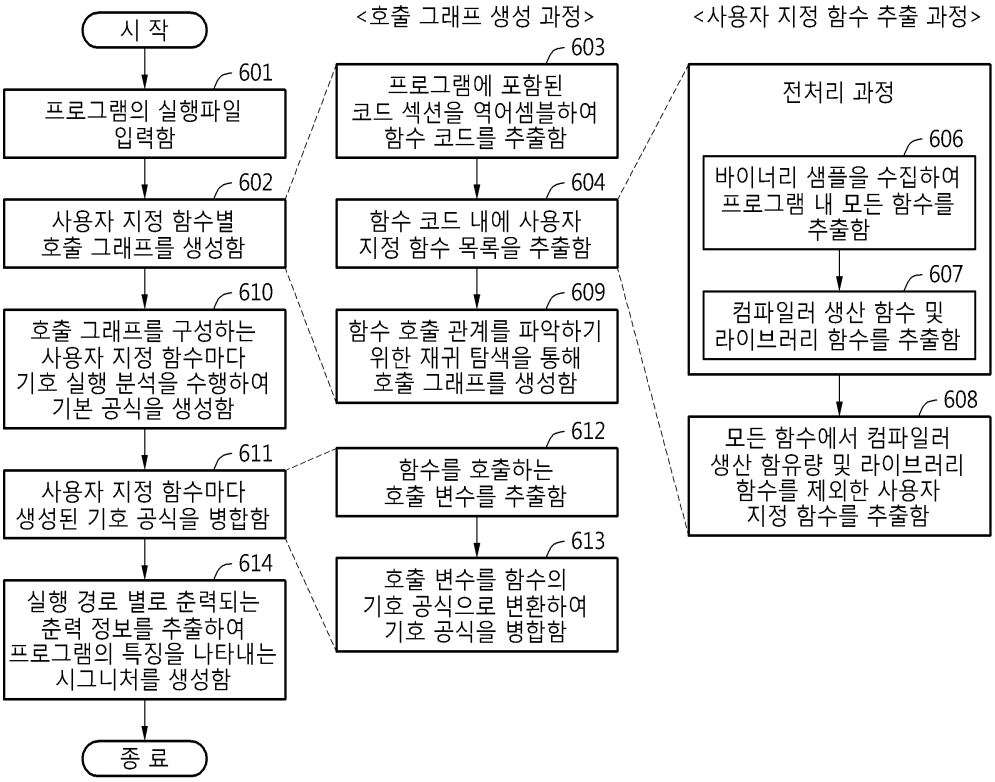
도면4


401
사용자 지정 함수 1
Funtion1(int a, int b, int c)
{ if(a>b) c = 10; else d
= Function2(a,b); return d; }
402
사용자 지정 함수 2
Funtion2(int a, int b)
{ k=a; k=k+b; call CreateFile(...) return k; }
403
API
CreateFile(...)
{ ... }
병합
404
병합된 사용자 지정 함수
Funtion1(int a, int b, int c)
{ if(a>b) c = 10; else d = a+b;
CreateFile(...); return d; }

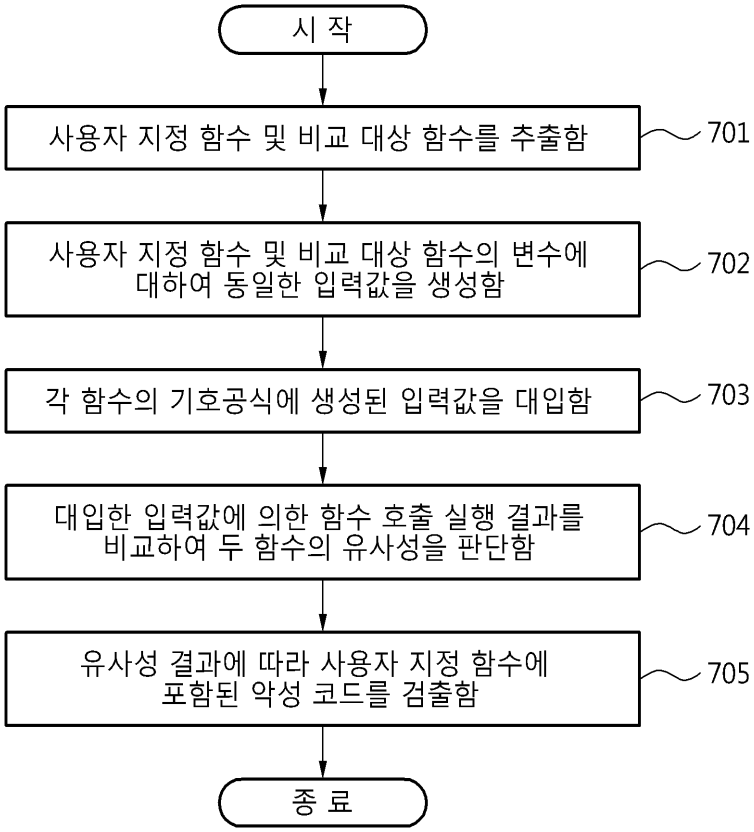
도면5


Funtion1(int a)
{ a=a+1; if (3<a<10){b=10-a; call API2(...); d=0;} else if (a>10) { c=mem[0x400010]; }
else if (a<3) {d=0;} call API1(...); return d; }
501
Path1 (3<a<10) → addOp | subOp | API2 | assignOp | API1 | retnOp
502
Path2 (a>10) → addOp | memReadOp | API1 | retnOp
Path3 (a<3) → addOp | API1 | retnOp
Path4 (a==3) → addOp | API1 | retnOp

도면6


시 작
601
프로그램의 실행파일 입력함
602
사용자 지정 함수별 호출 그래프를 생성함
610
호출 그래프를 구성하는 사용자 지정 함수마다 기호 실행 분석을 수행하여 기본 공식을 생성함
611
사용자 지정 함수마다 생성된 기호 공식을 병합함
614
실행 경로 별로 춘력되는 춘력 정보를 추출하여 프로그램의 특징을 나타내는 시그니처를 생성함
종 료
<호출 그래프 생성 과정>
603
프로그램에 포함된 코드 섹션을 역어셈블하여 함수 코드를 추출함
604
함수 코드 내에 사용자 지정 함수 목록을 추출함
609
함수 호출 관계를 파악하기 위한 재귀 탐색을 통해 호출 그래프를 생성함
612
함수를 호출하는 호출 변수를 추출함
613
호출 변수를 함수의 기호 공식으로 변환하여 기호 공식을 병합함
<사용자 지정 함수 추출 과정>
전처리 과정
606
바이너리 샘플을 수집하여 프로그램 내 모든 함수를 추출함
607
컴파일러 생산 함수 및 라이브러리 함수를 추출함
608
모든 함수에서 컴파일러 생산 함유량 및 라이브러리 함수를 제외한 사용자 지정 함수를 추출함

도면7


시 작
사용자 지정 함수 및 비교 대상 함수를 추출함
701
사용자 지정 함수 및 비교 대상 함수의 변수에 대하여 동일한 입력값을 생성함
702
각 함수의 기호공식에 생성된 입력값을 대입함
703
대입한 입력값에 의한 함수 호출 실행 결과를 비교하여 두 함수의 유사성을 판단함
704
유사성 결과에 따라 사용자 지정 함수에 포함된 악성 코드를 검출함
705
종 료