(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2020년02월07일
(11) 등록번호 10-2074909
(24) 등록일자 2020년02월03일

(51) 국제특허분류(Int. Cl.)
G06F 11/36 (2006.01) *G06F 21/56* (2013.01)
G06F 21/57 (2013.01) *G06N 3/08* (2006.01)
(52) CPC특허분류
G06F 11/3604 (2013.01)
G06F 21/563 (2013.01)
(21) 출원번호 10-2019-0110679
(22) 출원일자 2019년09월06일
심사청구일자 2019년09월06일
(56) 선행기술조사문헌
JP2019148917 A*
KR101935261 B1*
논문, 최민준 외, 기계학습 알고리즘을 이용한 소프트웨어 취약 여부 예측 시스템, 한국정보보호학회 28(3), 2018.6*
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
세종대학교산학협력단
서울특별시 광진구 능동로 209 (군자동, 세종대학교)
(72) 발명자
윤주범
서울특별시 송파구 충민로4길 19, 704동 401호(장지동, 송파파인타운7단지)
유지현
경기도 부천시 조마루로371번길 52, 402호(춘의동, 삼성리치빌)
(뒷면에 계속)
(74) 대리인
두호특허법인

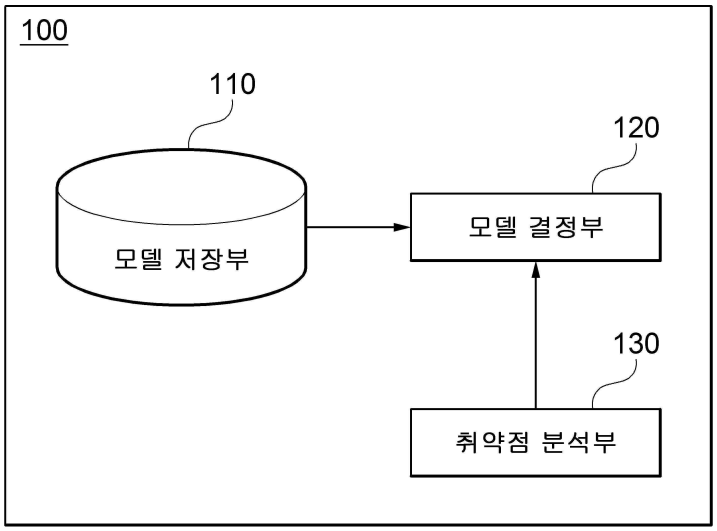
전체 청구항 수 : 총 9 항 심사관 : 이철수

(54) 발명의 명칭 **소프트웨어 취약점 분류 장치 및 방법**

(57) 요 약

소프트웨어 취약점 분류 장치 및 방법이 개시된다. 본 발명의 일 실시예에 따른 소프트웨어 취약점 분류 장치는, 각각 복수의 취약점 데이터 셋 중 하나와 복수의 학습 알고리즘 중 하나를 이용한 학습을 통해 생성된 복수의 취약점 분류 모델을 저장하는 모델 저장부; 상기 복수의 취약점 분류 모델 중 취약점 분류 대상인 소프트웨어 바이너리 파일에 대한 취약점 분류를 위해 이용할 취약점 분류 모델을 결정하는 모델 결정부; 및 상기 결정된 취약점 분류 모델을 이용하여 상기 소프트웨어 바이너리 파일에 포함된 취약점을 분류하는 취약점 분석부를 포함한다.

대 표 도 - 도1



(52) CPC특허분류
G06F 21/577 (2013.01)
G06N 3/08 (2013.01)
G06F 2221/033 (2013.01)
(72) 발명자
송준환
경기도 성남시 분당구 금곡로 171, 105동 402호(구미동, 현대노블리스)
박기웅
서울특별시 광진구 능동로17길 21(화양동), 304호

이 발명을 지원한 국가연구개발사업
과제고유번호 1711075702
부처명 과학기술정보통신부
연구관리전문기관 정보통신기술진흥센터
연구사업명 정보통신기술인력양성
연구과제명 지능형 비행로봇 융합기술 연구
기 여 율 1/1
주관기관 세종대학교 산학협력단
연구기간 2018.06.01 ~ 2021.12.31
공지예외적용 : 있음

명 세 서

청구범위

청구항 1

각각 복수의 취약점 데이터 셋 중 하나와 복수의 학습 알고리즘 중 하나를 이용한 학습을 통해 생성된 복수의 취약점 분류 모델을 저장하는 모델 저장부;

상기 복수의 취약점 분류 모델 중 취약점 분류 대상인 소프트웨어 바이너리 파일에 대한 취약점 분류를 위해 이용할 취약점 분류 모델을 결정하는 모델 결정부; 및

상기 결정된 취약점 분류 모델을 이용하여 상기 소프트웨어 바이너리 파일에 포함된 취약점을 분류하는 취약점 분석부를 포함하고,

상기 모델 결정부는, 상기 소프트웨어 바이너리 파일과 상기 복수의 취약점 데이터 셋 각각 사이의 유사도 및 상기 복수의 취약점 분류 모델 각각의 분류 정확도에 기초하여 상기 취약점 분류를 위해 이용할 취약점 분류 모델을 결정하는 소프트웨어 취약점 분류 장치.

청구항 2

삭제

청구항 3

청구항 1에 있어서,

상기 모델 결정부는, 상기 복수의 취약점 분류 모델 중 상기 유사도가 가장 높은 취약점 데이터 셋을 이용하여 학습된 하나 이상의 취약점 분류 모델을 결정하고, 상기 하나 이상의 취약점 분류 모델 중 상기 분류 정확도가 가장 높은 취약점 분류 모델을 상기 취약점 분류를 위해 이용할 취약점 분류 모델로 결정하는 소프트웨어 취약점 분류 장치.

청구항 4

청구항 1에 있어서,

상기 복수의 학습 알고리즘은, 랜덤 포레스트(random forest) 알고리즘, 소프트맥스 회귀(softmax regression) 및 CNN(Convolutional Neural Network) 알고리즘 중 적어도 하나를 포함하는 소프트웨어 취약점 분류 장치.

청구항 5

청구항 1에 있어서,

상기 복수의 취약점 데이터 셋 중 사용자에 의해 선택된 데이터 셋 및 상기 복수의 학습 알고리즘 중 상기 사용자에 의해 선택된 학습 알고리즘을 이용한 학습을 통해 상기 복수의 취약점 분류 모델을 생성하는 모델 생성부를 더 포함하는 소프트웨어 취약점 분류 장치.

청구항 6

취약점 분류 대상인 소프트웨어 바이너리 파일을 입력받는 단계;

각각 복수의 취약점 데이터 셋 중 하나와 복수의 학습 알고리즘 중 하나를 이용한 학습을 통해 생성된 복수의 취약점 분류 모델 중 상기 소프트웨어 바이너리 파일에 대한 취약점 분류를 위해 이용할 취약점 분류 모델을 결

정하는 단계; 및

상기 결정된 취약점 분류 모델을 이용하여 상기 소프트웨어 바이너리 파일에 포함된 취약점을 분류하는 단계를 포함하고,

상기 결정하는 단계는, 상기 소프트웨어 바이너리 파일과 상기 복수의 취약점 데이터 셋 각각 사이의 유사도 및 상기 복수의 취약점 분류 모델 각각의 분류 정확도에 기초하여 상기 취약점 분류를 위해 이용할 취약점 분류 모델을 결정하는 소프트웨어 취약점 분류 방법.

청구항 7

삭제

청구항 8

청구항 6에 있어서,

상기 결정하는 단계는, 상기 복수의 취약점 분류 모델 중 상기 유사도가 가장 높은 취약점 데이터 셋을 이용하여 학습된 하나 이상의 취약점 분류 모델을 결정하고, 상기 하나 이상의 취약점 분류 모델 중 상기 분류 정확도가 가장 높은 취약점 분류 모델을 상기 취약점 분류를 위해 이용할 취약점 분류 모델로 결정하는 소프트웨어 취약점 분류 방법.

청구항 9

청구항 6에 있어서,

상기 복수의 학습 알고리즘은, 랜덤 포레스트(random forest) 알고리즘, 소프트맥스 회귀(softmax regression) 및 CNN(Convolutional Neural Network) 알고리즘 중 적어도 하나를 포함하는 소프트웨어 취약점 분류 방법.

청구항 10

청구항 6에 있어서,

상기 입력받는 단계 이전에, 상기 복수의 취약점 데이터 셋 중 사용자에 의해 선택된 데이터 셋 및 상기 복수의 학습 알고리즘 중 상기 사용자에 의해 선택된 학습 알고리즘을 이용한 학습을 통해 상기 복수의 취약점 분류 모델을 생성하는 단계를 더 포함하는 소프트웨어 취약점 분류 방법.

청구항 11

하나 이상의 프로세서;

메모리; 및

하나 이상의 프로그램을 포함하는 장치로서,

상기 하나 이상의 프로그램은 상기 메모리에 저장되고 상기 하나 이상의 프로세서에 의해 실행되도록 구성되며,

상기 하나 이상의 프로그램은,

취약점 분류 대상인 소프트웨어 바이너리 파일을 입력받는 단계;

각각 복수의 취약점 데이터 셋 중 하나와 복수의 학습 알고리즘 중 하나를 이용한 학습을 통해 생성된 복수의 취약점 분류 모델 중 상기 소프트웨어 바이너리 파일에 대한 취약점 분류를 위해 이용할 취약점 분류 모델을 결정하되, 상기 소프트웨어 바이너리 파일과 상기 복수의 취약점 데이터 셋 각각 사이의 유사도 및 상기 복수의 취약점 분류 모델 각각의 분류 정확도에 기초하여 상기 취약점 분류를 위해 이용할 취약점 분류 모델을 결정하는 단계; 및

상기 결정된 취약점 분류 모델을 이용하여 상기 소프트웨어 바이너리 파일에 포함된 취약점을 분류하는 단계를 실행하기 위한 명령어들을 포함하는 소프트웨어 취약점 분류 장치.

발명의 설명

기 술 분 야

[0001] 본 발명의 실시예들은 소프트웨어 취약점 분석 기술과 관련된다.

배 경 기 술

[0002] 소프트웨어 바이너리의 취약 여부를 예측하기 위해 퍼징(fuzzing)과 기호 실행(symbolic execution) 등과 같이 다양한 기법이 사용되고 있다. 하지만, 이러한 종래 기법들은 탐지 시간이 많이 소요되므로 상황에 따라 오버헤드가 발생할 수 있으며, 전문가 수준의 데이터에 대한 이해가 필요한 단점이 있다.

[0003] 이러한 단점을 개선하고자 머신 러닝(machine learning) 또는 딥 러닝(deep learning) 알고리즘을 이용해 소프트웨어 바이너리의 취약점을 분석 및 분류하는 연구가 현재 진행되고 있지만 기존 연구들 또한 여러 문제점이 있다.

[0004] 첫 번째로 학습 데이터의 양에 따라 머신 러닝과 딥 러닝 알고리즘의 성능 차이가 있다. 예를 들어, 딥 러닝은 인간의 뇌처럼 데이터 특징을 스스로 처리하고 학습한 데이터의 양이 많을수록 가장 일반적인 모델을 만들어 정확도를 높인다. 하지만 전통적인 머신 러닝의 경우 인간이 일일이 학습할 데이터 특징을 알려주므로 딥 러닝보다 비교적 높은 정확도를 가지는 구간이 존재한다. 따라서, 대체로 데이터의 양이 적다면 머신 러닝 성능이 더 우수하고, 데이터의 양이 많다면 딥 러닝 성능이 더 우수하다.

[0005] 두 번째로 지도학습 알고리즘의 경우 데이터의 종류를 명시하는 라벨(label) 데이터가 있어야 하는데, 데이터 셋마다 라벨 데이터가 있을 수도, 없을 수도 있기 때문에 없다면 데이터에 라벨을 일일이 붙이거나 비지도 학습(unsupervised learning) 알고리즘을 새로 구현해야 하는 어려움이 존재한다.

[0006] 그리고 마지막으로 단일 알고리즘과 단일 데이터 셋은 하나의 학습모델을 생성하는데, 이는 검증할 바이너리가 데이터 셋과 소프트웨어 종류가 다르다면 예측 정확도가 떨어지고 이를 해결하기 위해 여러 데이터 셋을 하나로 합쳐 학습 모델을 만들면 복잡하고 일반적이지 않은 모델이 생성되어 정확도가 하락하는 과적합(overfitting) 문제가 발생할 수 있다.

선행기술문헌

특허문헌

[0007] (특허문헌 0001) 대한민국 등록특허 제10-1963756호 (2019.03.29. 공고)

발명의 내용

해결하려는 과제

[0008] 본 발명의 실시예들은 소프트웨어 취약점 분류 장치 및 방법을 제공하기 위한 것이다.

과제의 해결 수단

[0009] 본 발명의 일 실시예에 따른 소프트웨어 취약점 분류 장치는, 각각 복수의 취약점 데이터 셋 중 하나와 복수의 학습 알고리즘 중 하나를 이용한 학습을 통해 생성된 복수의 취약점 분류 모델을 저장하는 모델 저장부; 상기 복수의 취약점 분류 모델 중 취약점 분류 대상인 소프트웨어 바이너리 파일에 대한 취약점 분류를 위해 이용할 취약점 분류 모델을 결정하는 모델 결정부; 및 상기 결정된 취약점 분류 모델을 이용하여 상기 소프트웨어 바이너리 파일에 포함된 취약점을 분류하는 취약점 분석부를 포함한다.

[0010] 상기 모델 결정부는, 상기 소프트웨어 바이너리 파일과 상기 복수의 취약점 데이터 셋 각각 사이의 유사도 및 상기 복수의 취약점 분류 모델 각각의 분류 정확도 중 적어도 하나에 기초하여 상기 취약점 분류를 위해 이용할

취약점 분류 모델을 결정할 수 있다.

[0011] 상기 모델 결정부는, 상기 복수의 취약점 분류 모델 중 상기 유사도가 가장 높은 취약점 데이터 셋을 이용하여 학습된 하나 이상의 취약점 분류 모델을 결정하고, 상기 하나 이상의 취약점 분류 모델 중 상기 분류 정확도가 가장 높은 취약점 분류 모델을 상기 취약점 분류를 위해 이용할 취약점 분류 모델로 결정할 수 있다.

[0012] 상기 복수의 학습 알고리즘은, 랜덤 포레스트(random forest) 알고리즘, 소프트맥스 회귀(softmax regression) 및 CNN(Convolutional Neural Network) 알고리즘 중 적어도 하나를 포함할 수 있다.

[0013] 상기 복수의 취약점 데이터 셋 중 사용자에 의해 선택된 데이터 셋 및 상기 복수의 학습 알고리즘 중 상기 사용자에 의해 선택된 학습 알고리즘을 이용한 학습을 통해 상기 복수의 취약점 분류 모델을 생성하는 모델 생성부를 더 포함할 수 있다.

[0014] 본 발명의 일 실시예에 따른 소프트웨어 취약점 분류 방법은, 취약점 분류 대상인 소프트웨어 바이너리 파일을 입력받는 단계; 각각 복수의 취약점 데이터 셋 중 하나와 복수의 학습 알고리즘 중 하나를 이용한 학습을 통해 생성된 복수의 취약점 분류 모델 중 상기 소프트웨어 바이너리 파일에 대한 취약점 분류를 위해 이용할 취약점 분류 모델을 결정하는 단계; 및 상기 결정된 취약점 분류 모델을 이용하여 상기 소프트웨어 바이너리 파일에 포함된 취약점을 분류하는 단계를 포함한다.

[0015] 상기 결정하는 단계는, 상기 소프트웨어 바이너리 파일과 상기 복수의 취약점 데이터 셋 각각 사이의 유사도 및 상기 복수의 취약점 분류 모델 각각의 분류 정확도 중 적어도 하나에 기초하여 상기 취약점 분류를 위해 이용할 취약점 분류 모델을 결정할 수 있다.

[0016] 상기 결정하는 단계는, 상기 복수의 취약점 분류 모델 중 상기 유사도가 가장 높은 취약점 데이터 셋을 이용하여 학습된 하나 이상의 취약점 분류 모델을 결정하고, 상기 하나 이상의 취약점 분류 모델 중 상기 분류 정확도가 가장 높은 취약점 분류 모델을 상기 취약점 분류를 위해 이용할 취약점 분류 모델로 결정할 수 있다.

[0017] 상기 복수의 학습 알고리즘은, 랜덤 포레스트(random forest) 알고리즘, 소프트맥스 회귀(softmax regression) 및 CNN(Convolutional Neural Network) 알고리즘 중 적어도 하나를 포함할 수 있다.

[0018] 상기 입력받는 단계 이전에, 상기 복수의 취약점 데이터 셋 중 사용자에 의해 선택된 데이터 셋 및 상기 복수의 학습 알고리즘 중 상기 사용자에 의해 선택된 학습 알고리즘을 이용한 학습을 통해 상기 복수의 취약점 분류 모델을 생성하는 단계를 더 포함할 수 있다.

[0019] 본 발명의 일 실시예에 따른 소프트웨어 취약점 분류 장치는, 하나 이상의 프로세서; 메모리; 및 하나 이상의 프로그램을 포함하는 장치로서, 상기 하나 이상의 프로그램은 상기 메모리에 저장되고 상기 하나 이상의 프로세서에 의해 실행되도록 구성되며, 상기 하나 이상의 프로그램은, 취약점 분류 대상인 소프트웨어 바이너리 파일을 입력받는 단계; 각각 복수의 취약점 데이터 셋 중 하나와 복수의 학습 알고리즘 중 하나를 이용한 학습을 통해 생성된 복수의 취약점 분류 모델 중 상기 소프트웨어 바이너리 파일에 대한 취약점 분류를 위해 이용할 취약점 분류 모델을 결정하는 단계; 및 상기 결정된 취약점 분류 모델을 이용하여 상기 소프트웨어 바이너리 파일에 포함된 취약점을 분류하는 단계를 실행하기 위한 명령어들을 포함한다.

발명의 효과

[0020] 본 발명의 실시예들에 따르면, 취약점 분류 대상인 소프트웨어 바이너리 파일에 따라 각각 상이한 학습 데이터 셋과 학습 알고리즘을 통해 생성된 복수의 취약점 분류 모델 중 하나를 선택하여 취약점 분류를 수행함으로써, 단일 학습 알고리즘과 단일 학습 데이터 셋 사용으로 인한 과적합 문제를 해결함과 동시에 취약점 분류의 정확성을 향상시킬 수 있다.

도면의 간단한 설명

[0021] 도 1은 본 발명의 일 실시예에 따른 소프트웨어 취약점 분류 장치의 구성도

도 2는 본 발명의 추가적인 실시예에 따른 소프트웨어 취약점 분류 장치의 구성도

도 3은 본 발명의 일 실시예에 따른 소프트웨어 취약점 분류 모델 생성 방법을 나타낸 순서도

도 4는 본 발명의 일 실시예에 따른 소프트웨어 취약점 분류 방법의 순서도

도 5는 예시적인 실시예들에서 사용되기에 적합한 컴퓨팅 장치를 포함하는 컴퓨팅 환경을 예시하여 설명하기 위

한 블록도

발명을 실시하기 위한 구체적인 내용

[0022] 이하, 도면을 참조하여 본 발명의 구체적인 실시형태를 설명하기로 한다. 이하의 상세한 설명은 본 명세서에서 기술된 방법, 장치 및/또는 시스템에 대한 포괄적인 이해를 돕기 위해 제공된다. 그러나 이는 예시에 불과하며 본 발명은 이에 제한되지 않는다.

[0023] 본 발명의 실시예들을 설명함에 있어서, 본 발명과 관련된 공지기술에 대한 구체적인 설명이 본 발명의 요지를 불필요하게 흐릴 수 있다고 판단되는 경우에는 그 상세한 설명을 생략하기로 한다. 그리고, 후술되는 용어들은 본 발명에서의 기능을 고려하여 정의된 용어들로서 이는 사용자, 운용자의 의도 또는 관례 등에 따라 달라질 수 있다. 그러므로 그 정의는 본 명세서 전반에 걸친 내용을 토대로 내려져야 할 것이다. 상세한 설명에서 사용되는 용어는 단지 본 발명의 실시예들을 기술하기 위한 것이며, 결코 제한적이어서는 안 된다. 명확하게 달리 사용되지 않는 한, 단수 형태의 표현은 복수 형태의 의미를 포함한다. 본 설명에서, "포함" 또는 "구비"와 같은 표현은 어떤 특성들, 숫자들, 단계들, 동작들, 요소들, 이들의 일부 또는 조합을 가리키기 위한 것이며, 기술된 것 이외에 하나 또는 그 이상의 다른 특성, 숫자, 단계, 동작, 요소, 이들의 일부 또는 조합의 존재 또는 가능성을 배제하도록 해석되어서는 안 된다.

[0024] 도 1은 본 발명의 일 실시예에 따른 소프트웨어 취약점 분류 장치의 구성도이다.

[0025] 도 1을 참조하면, 본 발명의 일 실시예에 따른 소프트웨어 취약점 분류 장치(100)는 모델 저장부(110), 모델 결정부(120) 및 취약점 분석부(130)를 포함한다.

[0026] 모델 저장부(110)는 복수의 취약점 분류 모델을 저장한다.

[0027] 이때, 취약점 분류 모델은 소프트웨어 바이너리 파일에 포함된 취약점을 사전에 알려진 복수의 취약점 중 하나로 분류한 결과를 출력하도록 학습된 분류 모델을 의미한다.

[0028] 구체적으로, 모델 저장부(110)에 저장된 각 취약점 분류 모델은 복수의 취약점 데이터 셋(data set) 중 하나와 복수의 학습 알고리즘 중 하나를 이용한 학습을 통해 생성될 수 있다.

[0029] 이때, 각 취약점 데이터 셋은 예를 들어, KISA(Korean Internet & Security Agency), NIST(National Institute of Standards and Technology), ITC benchmark 등과 같이 알려진 소프트웨어 취약점에 대한 정보를 제공하는 다양한 종류의 공개된 데이터베이스로부터 획득될 수 있다. 또한, 각 취약점 데이터 셋은 예를 들어, 복수의 바이너리 파일 및 각 바이너리 파일에 포함된 취약점에 대한 분류 결과를 나타내는 라벨 데이터를 포함할 수 있다.

[0030] 한편, 복수의 학습 알고리즘은 예를 들어, 랜덤 포레스트(random forest) 알고리즘, 소프트맥스 회귀(softmax regression) 알고리즘 및 CNN(Convolutional Neural Network) 알고리즘 중 적어도 하나를 포함할 수 있으나, 반드시 이에 한정되는 것은 아니며 상술한 예 외에도 다양한 종류의 기계 학습(machine learning) 모델을 포함할 수 있다.

[0031] 한편, 모델 저장부(110)에 저장된 복수의 취약점 분류 모델은 각각 상이한 취약점 데이터 셋 및 상이한 학습 알고리즘을 이용하여 생성될 수 있다. 예를 들어, 복수의 취약점 데이터 셋이 데이터 셋 a 및 데이터 셋 b를 포함하고, 복수의 학습 알고리즘이 알고리즘 A 및 알고리즘 B를 포함하는 것으로 가정하면, 모델 저장부(110)에 저장되는 학습 모델은 예를 들어, 데이터 셋 a와 알고리즘 A를 이용한 학습을 통해 생성된 취약점 분류 모델 1, 데이터 셋 a와 알고리즘 B를 이용한 학습을 통해 생성된 취약점 분류 모델 2, 데이터 셋 b와 알고리즘 A를 이용한 학습을 통해 생성된 취약점 분류 모델 3 및 데이터 셋 b와 알고리즘 B를 이용한 학습을 통해 생성된 취약점 분류 모델 4 중 적어도 하나를 포함할 수 있다.

[0032] 모델 결정부(120)는 모델 저장부(110)에 저장된 복수의 취약점 분류 모델 중 취약점 분류 대상인 소프트웨어 바이너리 파일에 대한 취약점 분류를 위해 이용할 취약점 분류 모델을 결정한다.

[0033] 구체적으로, 모델 결정부(120)는 취약점 분류 대상인 소프트웨어 바이너리 파일과 복수의 취약점 분류 모델 각각의 생성을 위해 이용된 복수의 취약점 데이터 셋 각각 사이의 유사도 및 복수의 취약점 분류 모델 각각의 분류 정확도 중 적어도 하나에 기초하여 취약점 분류 모델을 결정할 수 있다. 예를 들어, 모델 결정부(120)는 모델 저장부(110)에 저장된 복수의 취약점 분류 모델 중 취약점 분류 대상인 소프트웨어 바이너리 파일과의 유사도가 가장 높은 취약점 데이터 셋을 이용하여 학습된 하나 이상의 취약점 분류 모델을 선택할 수 있다. 이후,

모델 결정부(120)는 선택된 하나 이상의 취약점 분류 모델 중 분류 정확도가 가장 높은 취약점 분류 모델을 분류 대상인 소프트웨어 바이너리 파일에 대한 취약점 분류를 위해 이용할 취약점 분류 모델로 결정할 수 있다.

[0034] 구체적인 예로, 모델 결정부(120)는 취약점 분류 대상 소프트웨어 바이너리 파일에서 추출된 특징 벡터와 각 취약점 데이터 셋에 포함된 각 바이너리 파일로부터 추출된 특징 벡터 사이의 코사인 유사도(cosine similarity)를 산출할 수 있다. 이때, 특징 벡터는 예를 들어, 바이너리 파일을 기 설정된 크기의 이미지 데이터로 변환한 후 변환된 이미지 데이터로부터 추출된 특징 값들을 포함하는 벡터일 수 있다.

[0035] 한편, 모델 결정부(120)는 모델 저장부(110)에 저장된 복수의 취약점 분류 모델 중 예를 들어, 산출된 코사인 유사도가 가장 높은 바이너리 파일을 포함하는 취약점 데이터 셋을 이용하여 학습된 하나 이상의 취약점 분류 모델을 선택하고, 선택된 하나 이상의 취약점 분류 모델 중 분류 정확도가 가장 높은 취약점 분류 모델을 취약점 분류 대상인 소프트웨어 바이너리 파일에 대한 취약점 분류를 위해 이용할 취약점 분류 모델로 결정할 수 있다.

[0036] 한편, 모델 저장부(110)에 저장된 각 취약점 분류 모델의 분류 정확도는 각 취약점 분류 모델의 생성을 위한 학습 과정에서 사전 산출될 수 있다.

[0037] 취약점 분석부(130)는 모델 결정부(120)에 의해 선택된 취약점 분류 모델을 이용하여 소프트웨어 바이너리 파일에 포함된 취약점을 분류한다.

[0038] 이때, 취약점 분석부(130)는 취약점 분류 대상인 소프트웨어 바이너리 파일을 변환하여 생성된 기 설정된 크기의 이미지 데이터를 선택된 취약점 분류 모델의 입력으로 이용할 수 있으며, 취약점 분류 대상인 소프트웨어 바이너리 파일에 대한 다양한 전처리 과정을 수행할 수 있다.

[0039] 도 2는 본 발명의 추가적인 실시예에 따른 소프트웨어 취약점 분류 장치의 구성도이다.

[0040] 도 2를 참조하면, 본 발명의 일 실시예에 따른 소프트웨어 취약점 분류 장치(100)는 모델 저장부(210), 모델 결정부(220), 취약점 분석부(230) 및 모델 생성부(240)를 포함한다.

[0041] 도 2에 도시된 예에서, 모델 저장부(210), 모델 결정부(220) 및 취약점 분석부(230)는 각각 도 1에 도시된 모델 저장부(110), 모델 결정부(120) 및 취약점 분석부(130)와 동일한 구성이므로, 이에 대한 중복적인 설명은 생략한다.

[0042] 모델 생성부(240)는 복수의 취약점 분류 모델을 생성하고, 생성된 각 취약점 분류 모델을 모델 저장부(210)에 저장한다.

[0043] 구체적으로, 모델 생성부(240)는 사용자로부터 복수의 취약점 데이터 셋 및 복수의 학습 알고리즘 중 취약점 분류 모델 생성을 위해 이용할 취약점 데이터 셋과 학습 알고리즘을 선택 받을 수 있다. 이때, 복수의 취약점 데이터 셋 및 복수의 학습 알고리즘은 사용자에 의해 추가 또는 삭제될 수 있다.

[0044] 한편, 사용자에 의해 취약점 데이터 셋 및 학습 알고리즘이 선택된 경우, 모델 생성부(240)는 선택된 학습 알고리즘을 이용하여 선택된 취약점 데이터 셋에 포함된 학습 데이터들을 학습함으로써 취약점 분류 모델을 생성할 수 있다.

[0045] 이때, 학습 데이터는 바이너리 파일과 해당 바이너리 파일에 대한 라벨 데이터(예를 들어, 취약점 분류 결과)를 포함할 수 있다. 또한, 모델 생성부(240)는 예를 들어, 선택된 취약점 데이터 셋에 포함된 각 학습 데이터의 바이너리 파일을 취약점 분류 모델의 입력 값으로 이용하고, 해당 바이너리 파일에 대한 라벨 데이터를 취약점 분류 모델의 목표 값으로 이용한 지도 학습 기법을 통해 취약점 분류 모델을 학습 시킬 수 있다.

[0046] 한편, 모델 생성부(240)는 실시예에 따라, 학습 데이터에 포함된 바이너리 파일을 기 설정된 크기의 이미지 데이터로 변환한 후, 변환된 이미지 데이터를 취약점 분류 모델의 입력으로 이용할 수 있으며, 이 외에도 학습 데이터에 대한 다양한 전처리 과정을 수행할 수 있다.

[0047] 한편, 취약점 분류 모델이 생성된 경우, 모델 생성부(240)는 생성된 취약점 분류 모델에 대한 분류 정확도를 산출할 수 있으며, 산출된 분류 정확도를 취약점 분류 모델과 함께 모델 저장부(210)에 저장할 수 있다.

[0048] 이때, 정확도는 예를 들어, 아래의 수학식 1에 따라 산출될 수 있으나 정확도 산출 방식이 반드시 특정한 방식으로 한정되는 것은 아니다.

[0049] [수학식 1]

[0050] Accuracy=C/T×100

[0051] 이때, T는 취약점 데이터 셋에 포함된 학습 데이터의 총 개수, C는 취약점 데이터 셋에 포함된 학습 데이터 중 취약점 분류 모델의 취약점 분류 결과와 라벨 데이터가 일치한 학습 데이터의 총 개수를 나타낸다.

[0052] 도 3은 본 발명의 일 실시예에 따른 소프트웨어 취약점 분류 모델 생성 방법을 나타낸 순서도이다.

[0053] 도 3에 도시된 방법은 예를 들어, 도 2에 도시된 소프트웨어 취약점 분류 장치(200)에 의해 수행될 수 있다.

[0054] 도 3을 참조하면, 우선, 소프트웨어 취약점 분류 장치(200)는 복수의 취약점 데이터 셋 및 복수의 학습 알고리즘 중 취약점 분류 모델 생성을 위해 이용할 취약점 데이터 셋 및 학습 알고리즘을 선택한다(310).

[0055] 이때, 취약점 데이터 셋 및 학습 알고리즘의 선택은 예를 들어, 사용자의 입력에 따라 수행될 수 있다. 이를 위해 소프트웨어 취약점 분류 장치(200)는 선택 가능한 취약점 데이터 셋과 학습 알고리즘의 리스트를 제공하고, 제공된 리스트에서 취약점 데이터 셋과 학습 알고리즘을 선택할 수 있도록 하는 사용자 인터페이스를 사용자에게 제공할 수 있다.

[0056] 한편, 취약점 데이터 셋 및 학습 알고리즘이 선택된 경우, 소프트웨어 취약점 분류 장치(200)는 선택된 취약점 데이터 셋 및 학습 알고리즘을 이용한 학습을 통해 취약점 분류 모델을 생성한다(320).

[0057] 도 4는 본 발명의 일 실시예에 따른 소프트웨어 취약점 분류 방법의 순서도이다.

[0058] 도 4에 도시된 방법은 예를 들어, 도 1 또는 도 2에 도시된 소프트웨어 취약점 분류 장치(100, 200)에 의해 수행될 수 있다.

[0059] 도 4를 참조하면, 우선, 소프트웨어 취약점 분류 장치(100, 200)는 취약점 분류 대상인 소프트웨어 바이너리 파일을 입력받는다(410).

[0060] 이후, 소프트웨어 취약점 분류 장치(100, 200)는 각각 복수의 취약점 데이터 셋 중 하나와 복수의 학습 알고리즘 중 하나를 이용한 학습을 통해 생성된 복수의 취약점 분류 모델 중 소프트웨어 바이너리 파일에 대한 취약점 분류를 위해 이용할 취약점 분류 모델을 결정한다(420).

[0061] 이때, 일 실시예에 따르면, 소프트웨어 취약점 분류 장치(100, 200)는 입력된 소프트웨어 바이너리 파일과 복수의 취약점 데이터 셋 각각 사이의 유사도 및 복수의 취약점 분류 모델 각각의 분류 정확도 중 적어도 하나에 기초하여 취약점 분류를 위해 이용할 취약점 분류 모델을 결정할 수 있다.

[0062] 이후, 소프트웨어 취약점 분류 장치(100, 200)는 결정된 취약점 분류 모델을 이용하여, 입력된 소프트웨어 바이너리 파일에 포함된 취약점을 분류한다(430).

[0063] 한편, 도 3 및 도 4에 도시된 순서도에서는 상기 방법을 복수 개의 단계로 나누어 기재하였으나, 적어도 일부의 단계들은 순서를 바꾸어 수행되거나, 다른 단계와 결합되어 함께 수행되거나, 생략되거나, 세부 단계들로 나뉘어 수행되거나, 또는 도시되지 않은 하나 이상의 단계가 부가되어 수행될 수 있다.

[0064] 도 5는 예시적인 실시예들에서 사용되기에 적합한 컴퓨팅 장치를 포함하는 컴퓨팅 환경을 예시하여 설명하기 위한 블록도이다. 도시된 실시예에서, 각 컴포넌트들은 이하에 기술된 것 이외에 상이한 기능 및 능력을 가질 수 있고, 이하에 기술되지 않은 것 이외에도 추가적인 컴포넌트를 포함할 수 있다.

[0065] 도시된 컴퓨팅 환경(10)은 컴퓨팅 장치(12)를 포함한다. 일 실시예에서, 컴퓨팅 장치(12)는 도 1 또는 도 2에 도시된 소프트웨어 취약점 분류 장치(100, 200)에 포함되는 하나 이상의 컴포넌트일 수 있다.

[0066] 컴퓨팅 장치(12)는 적어도 하나의 프로세서(14), 컴퓨터 판독 가능 저장 매체(16) 및 통신 버스(18)를 포함한다. 프로세서(14)는 컴퓨팅 장치(12)로 하여금 앞서 언급된 예시적인 실시예에 따라 동작하도록 할 수 있다. 예컨대, 프로세서(14)는 컴퓨터 판독 가능 저장 매체(16)에 저장된 하나 이상의 프로그램들을 실행할 수 있다. 상기 하나 이상의 프로그램들은 하나 이상의 컴퓨터 실행 가능 명령어를 포함할 수 있으며, 상기 컴퓨터 실행 가능 명령어는 프로세서(14)에 의해 실행되는 경우 컴퓨팅 장치(12)로 하여금 예시적인 실시예에 따른 동작들을 수행하도록 구성될 수 있다.

[0067] 컴퓨터 판독 가능 저장 매체(16)는 컴퓨터 실행 가능 명령어 내지 프로그램 코드, 프로그램 데이터 및/또는 다른 적합한 형태의 정보를 저장하도록 구성된다. 컴퓨터 판독 가능 저장 매체(16)에 저장된 프로그램(20)은 프로세서(14)에 의해 실행 가능한 명령어의 집합을 포함한다. 일 실시예에서, 컴퓨터 판독 가능 저장 매체(16)는 메

모리(랜덤 액세스 메모리와 같은 휘발성 메모리, 비휘발성 메모리, 또는 이들의 적절한 조합), 하나 이상의 자기 디스크 저장 디바이스들, 광학 디스크 저장 디바이스들, 플래시 메모리 디바이스들, 그 밖에 컴퓨팅 장치(12)에 의해 액세스되고 원하는 정보를 저장할 수 있는 다른 형태의 저장 매체, 또는 이들의 적합한 조합일 수 있다.

[0068] 통신 버스(18)는 프로세서(14), 컴퓨터 판독 가능 저장 매체(16)를 포함하여 컴퓨팅 장치(12)의 다른 다양한 컴포넌트들을 상호 연결한다.

[0069] 컴퓨팅 장치(12)는 또한 하나 이상의 입출력 장치(24)를 위한 인터페이스를 제공하는 하나 이상의 입출력 인터페이스(22) 및 하나 이상의 네트워크 통신 인터페이스(26)를 포함할 수 있다. 입출력 인터페이스(22) 및 네트워크 통신 인터페이스(26)는 통신 버스(18)에 연결된다. 입출력 장치(24)는 입출력 인터페이스(22)를 통해 컴퓨팅 장치(12)의 다른 컴포넌트들에 연결될 수 있다. 예시적인 입출력 장치(24)는 포인팅 장치(마우스 또는 트랙패드 등), 키보드, 터치 입력 장치(터치패드 또는 터치스크린 등), 음성 또는 소리 입력 장치, 다양한 종류의 센서 장치 및/또는 촬영 장치와 같은 입력 장치, 및/또는 디스플레이 장치, 프린터, 스피커 및/또는 네트워크 카드와 같은 출력 장치를 포함할 수 있다. 예시적인 입출력 장치(24)는 컴퓨팅 장치(12)를 구성하는 일 컴포넌트로서 컴퓨팅 장치(12)의 내부에 포함될 수도 있고, 컴퓨팅 장치(12)와는 구별되는 별개의 장치로 컴퓨팅 장치(12)와 연결될 수도 있다.

[0070] 이상에서 대표적인 실시예를 통하여 본 발명에 대하여 상세하게 설명하였으나, 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자는 전술한 실시예에 대하여 본 발명의 범주에서 벗어나지 않는 한도 내에서 다양한 변형이 가능함을 이해할 것이다. 그러므로 본 발명의 권리범위는 설명된 실시예에 국한되어 정해져서는 안 되며, 후술하는 특허청구범위뿐만 아니라 이 특허청구범위와 균등한 것들에 의해 정해져야 한다.

부호의 설명

[0071] 10: 컴퓨팅 환경

12: 컴퓨팅 장치

14: 프로세서

16: 컴퓨터 판독 가능 저장 매체

18: 통신 버스

20: 프로그램

22: 입출력 인터페이스

24: 입출력 장치

26: 네트워크 통신 인터페이스

100, 200: 소프트웨어 취약점 분류 장치

110: 210: 모델 저장부

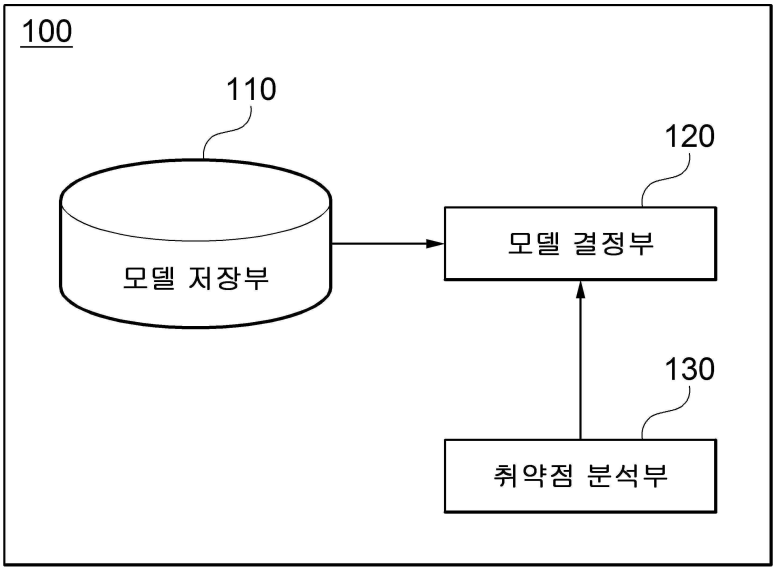
120, 220: 모델 결정부

130, 230: 취약점 분석부

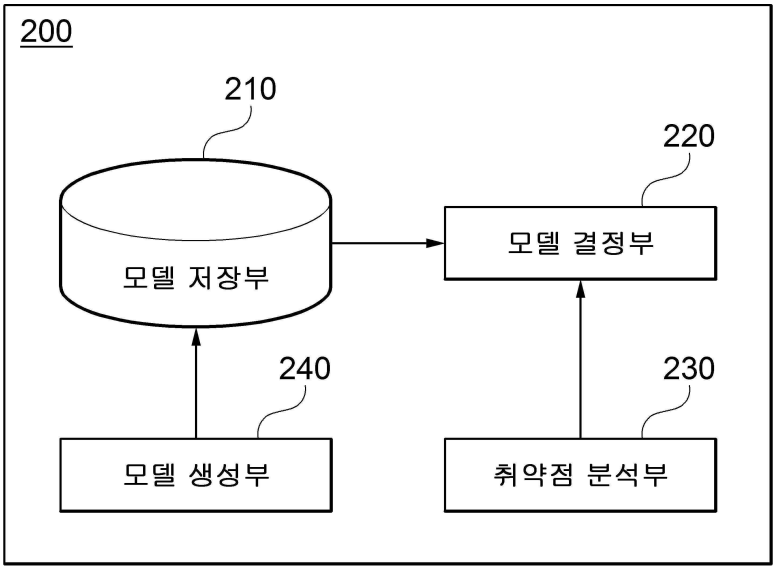
240: 모델 생성부

도면

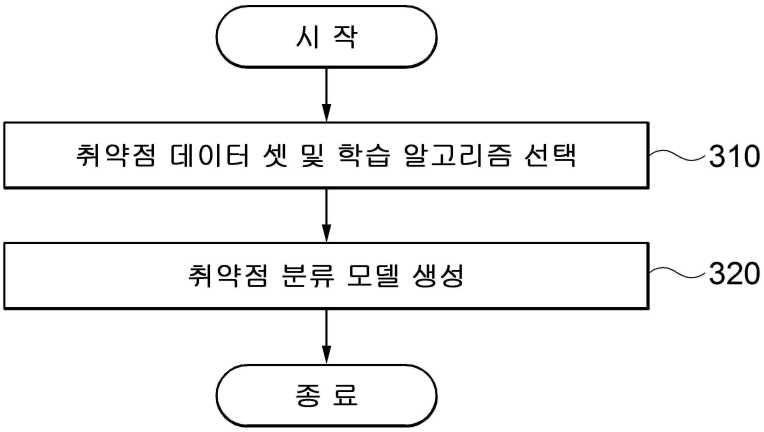
도면1


100
110
120
모델 저장부
모델 결정부
130
취약점 분석부

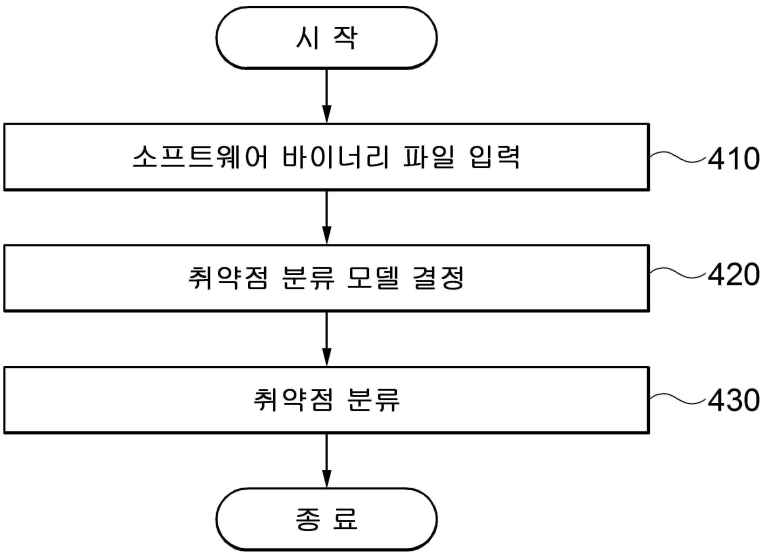
도면2


200
210
220
모델 저장부
모델 결정부
240
230
모델 생성부
취약점 분석부

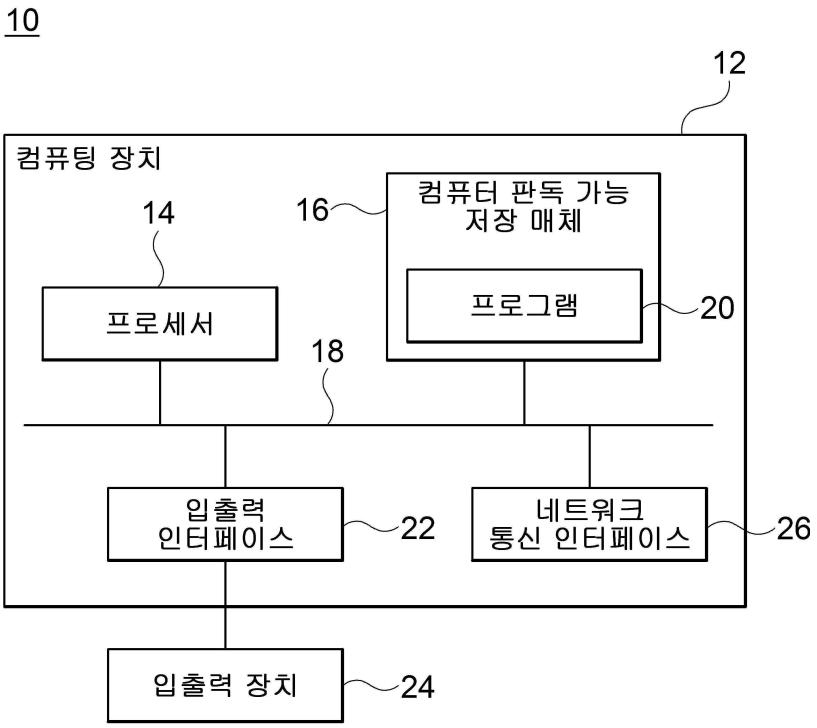
도면3


시 작
취약점 데이터 셋 및 학습 알고리즘 선택
310
취약점 분류 모델 생성
320
종 료

도면4


시 작
소프트웨어 바이너리 파일 입력
410
취약점 분류 모델 결정
420
취약점 분류
430
종 료

도면5


10
12
컴퓨팅 장치
14
프로세서
16
컴퓨터 판독 가능
저장 매체
프로그램
20
18
입출력
인터페이스
22
네트워크
통신 인터페이스
26
입출력 장치
24