(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2017년12월22일
(11) 등록번호 10-1811325
(24) 등록일자 2017년12월15일

(51) 국제특허분류(Int. Cl.)
G06F 21/56 (2013.01)
(52) CPC특허분류
G06F 21/564 (2013.01)
G06F 21/563 (2013.01)
(21) 출원번호 10-2015-7017244
(22) 출원일자(국제) 2014년01월16일
심사청구일자 2015년06월26일
(85) 번역문제출일자 2015년06월26일
(65) 공개번호 10-2015-0091492
(43) 공개일자 2015년08월11일
(86) 국제출원번호 PCT/US2014/011907
(87) 국제공개번호 WO 2014/113597
국제공개일자 2014년07월24일
(30) 우선권주장
50/KOL/2013 2013년01월16일 인도(IN)
(56) 선행기술조사문헌
US20090070663 A1*
US20120266244 A1*
US20110197272 A1
Charlie Cusrtsinger Univ. of Mass et al. "ZOZZLE: Fast and Precise In-Browser Javascript Malware Detection", Usenix, 9 June 2011 pp. 1-16
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
맥아피 인코퍼레이티드
미국 95054 캘리포니아 산타클라라 미션컬리지 블러바드 2821
(72) 발명자
슈, 총
미국 94086 캘리포니아주 서니베일 웨스트 아이오와 애비뉴 1141
선, 빙
미국 95051 캘리포니아주 산타 클라라 릴리크 드라이브 3700 넘버303
(뒷면에 계속)
(74) 대리인
양영준, 김연송, 백만기

전체 청구항 수 : 총 23 항 심사관 : 윤혜숙

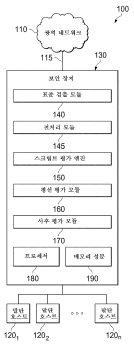
(54) 발명의 명칭 **네트워크 환경에서의 악성 스크립트 언어 코드의 검출**

(57) 요 약

방법은 한 예시적 실시예에 제공되고, 또한 컴파일된 스크립트의 실행을 개시하는 단계, 컴파일된 스크립트에서 호출되는 평션을 평가하는 단계, 적어도 제1 기준에 기초하여 실행 이벤트를 검출하는 단계, 및 실행 이벤트 큐에 실행 이벤트와 연관되는 정보를 저장하는 단계를 포함한다. 방법은 실행 이벤트 큐에서의 적어도 하나의 실

(뒷면에 계속)

대 표 도 - 도1



행 이벤트와 연관되는 정보에 기초하여 상관 시그니처를 검증하는 단계를 더 포함한다. 특정 실시예들에서, 방법은 컴파일러에 의한 스크립트의 컴파일 동안 스크립트의 대입문을 평가하는 단계, 적어도 제2 기준에 기초하여 컴파일 이벤트를 검출하는 단계, 및 컴파일 이벤트와 연관되는 정보를 컴파일 이벤트 큐에 저장하는 단계를 포함한다. 추가적 실시예들에서, 상관 시그니처의 검증은 컴파일 이벤트 큐에서의 하나 이상의 컴파일 이벤트들과 연관되는 정보에 부분적으로 기초한다.

(52) CPC특허분류
G06F 21/566 (2013.01)
G06F 21/567 (2013.01)

(72) 발명자
싱흐, 나브테지
인도 560017 방갈로레 카르나타카 올드 에어포트 로드 러스텀바그 레이아웃 3번 크로스 옥스포드 스튜디오 넘버302
린, 이총
미국 94538 캘리포니아주 프리몬트 베르네 스트리트 40461
부, 젱
미국 94539 캘리포니아주 프리몬트 세인트 필립 코트 216

명 세 서

청구범위

청구항 1

스크립트에서 악성 코드를 검출하는 방법으로서:

상기 방법은, 적어도 하나의 프로세서에 의해 실행되고,

실행 엔진에 의해, 컴파일된 스크립트의 실행을 개시하는 단계;

코드에 기초하여 상기 컴파일된 스크립트에 의해 호출되는 펑션(function)의 제어를 전달하는 단계 - 상기 코드는 상기 제어가 상기 실행 엔진으로부터 펑션 평가 모듈로 전달되게 하는 상기 펑션과 연관되고, 상기 펑션 평가 모듈은 상기 펑션을 실행하고 상기 펑션의 실행의 평가를 수행함 -;

상기 펑션의 실행의 평가에 기초하여 실행 이벤트를 검출하는 단계 - 상기 실행 이벤트는 적어도 제1 기준에 기초함 -;

상기 실행 이벤트와 연관되는 제1 정보를 실행 이벤트 큐에 저장하는 단계; 및

상기 실행 이벤트 큐에서의 상기 실행 이벤트와 연관되는 상기 제1 정보에 적어도 부분적으로 기초하여 상관 시그니처를 검증하는 단계 - 상기 상관 시그니처는 적어도 2개의 이벤트와 연관된 이벤트 정보의 조합을 정의하고, 상기 조합은 악성 코드의 존재를 표시함 -

를 포함하는 악성 코드 검출 방법.

청구항 2

제1항에 있어서,

상기 컴파일된 스크립트의 실행을 개시하는 단계 전에,

컴파일러에 의한 상기 스크립트의 컴파일 동안 상기 스크립트의 대입문(assignment statement)을 평가하는 단계 - 상기 컴파일된 스크립트는 상기 스크립트로부터 발생됨 -;

적어도 제2 기준에 기초하여 컴파일 이벤트를 검출하는 단계; 및

상기 컴파일 이벤트와 연관되는 제2 정보를 컴파일 이벤트 큐에 저장하는 단계를 더 포함하는 악성 코드 검출 방법.

청구항 3

제2항에 있어서,

상기 컴파일 이벤트와 연관되는 상기 제2 정보를 저장하는 단계 후에, 상기 컴파일 이벤트 큐에서의 상기 컴파일 이벤트와 연관되는 상기 제2 정보에 부분적으로 기초하여 상기 상관 시그니처를 검증하는 단계를 더 포함하는 악성 코드 검출 방법.

청구항 4

제2항에 있어서,

상기 컴파일 이벤트 큐 및 상기 실행 이벤트 큐는 통합되는 악성 코드 검출 방법.

청구항 5

제2항에 있어서,

상기 컴파일 이벤트를 검출하는 단계는, 상기 제2 기준이 만족되었는지를 결정하기 위해 상기 대입문의 우변 값을 평가하는 단계를 더 포함하는 악성 코드 검출 방법.

청구항 6

제2항에 있어서,

상기 컴파일 이벤트를 검출하는 단계는, 상기 제2 기준이 만족되었는지를 결정하기 위해 상기 대입문의 좌변 변수 명을 평가하는 단계를 더 포함하는 악성 코드 검출 방법.

청구항 7

제1항에 있어서,

상기 상관 시그니처를 검증하는 단계는 상기 컴파일된 스크립트의 실행이 끝나기 전에 수행되는 악성 코드 검출 방법.

청구항 8

제1항에 있어서,

상기 상관 시그니처를 검증하는 단계는 상기 컴파일된 스크립트의 실행이 끝난 후에 수행되는 악성 코드 검출 방법.

청구항 9

제1항에 있어서,

상기 상관 시그니처는 사용자에 의해 구성가능한 악성 코드 검출 방법.

청구항 10

제1항에 있어서,

상기 실행 이벤트를 검출하는 단계는 파라미터를 상기 펑션에게 전달하는 단계를 더 포함하고, 상기 제1 기준은 상기 파라미터의 미리 정해진 임계 길이에 기초하는 악성 코드 검출 방법.

청구항 11

제1항에 있어서,

상기 펑션은 데이터를 디코딩하고, 상기 제1 기준은 상기 펑션으로부터 귀결(result)되는 스트링의 미리 정해진 임계 길이에 기초하는 악성 코드 검출 방법.

청구항 12

제1항에 있어서,

상기 펑션은 데이터를 연접(concatenate)시키고, 상기 제1 기준은 연접된 데이터로부터 귀결되는 스트링의 미리 정해진 임계 길이에 기초하는 악성 코드 검출 방법.

청구항 13

제1항 내지 제12항 중 어느 한 항에 있어서,

상기 펑션이 실행을 끝낼 때 상기 펑션 평가 모듈로부터 상기 실행 엔진으로 제어를 전달하는 단계가 수행되는 악성 코드 검출 방법.

청구항 14

제1항 내지 제12항 중 어느 한 항에 있어서,

상기 실행 이벤트와 연관되는 상기 제1 정보는 상기 실행 이벤트의 식별, 상기 실행 이벤트가 검출된 횟수, 미리 정해진 이벤트들의 시퀀스의 식별, 및 상기 실행 이벤트와 하나 이상의 다른 실행 이벤트들 사이의 거리 중 하나 이상을 포함하는 악성 코드 검출 방법.

청구항 15

제1항 내지 제12항 중 어느 한 항에 있어서,

상기 상관 시그니처는 상기 평션에 대입되는(assigned) 가중(weight)에 부분적으로 기초하여 검증되며, 상기 가중은 상기 평션의 상대적 중요성을 나타내는 악성 코드 검출 방법.

청구항 16

제1항 내지 제12항 중 어느 한 항에 있어서,

상기 평션은 상기 컴파일된 스크립트에 의해 호출되는 하나 이상의 관련 평션들 중 하나이고, 각각의 관련 평션은 특정한 관련 평션의 제어가 상기 실행 엔진으로부터 상기 평션 평가 모듈로 전달되게 하는 각각의 코드와 연관되는 악성 코드 검출 방법.

청구항 17

스크립트에서 악성 코드를 검출하기 위해 그 상에 저장된 명령어들을 갖는 비-일시적 컴퓨터-판독가능한 저장 매체로서:

상기 명령어들은 적어도 하나의 프로세서에 의해 실행될 때, 상기 적어도 하나의 프로세서로 하여금:

실행 엔진에 의해, 컴파일된 스크립트의 실행을 개시하고;

코드에 기초하여 상기 컴파일된 스크립트에 의해 호출되는 펑션(function)의 제어를 전달하고 - 상기 코드는 상기 제어가 상기 실행 엔진으로부터 평션 평가 모듈로 전달되게 하는 상기 평션과 연관되고, 상기 평션 평가 모듈은 상기 평션을 실행하고 상기 평션의 실행의 평가를 수행함 -;

상기 평션의 실행의 평가에 기초하여 실행 이벤트를 검출하고 - 상기 실행 이벤트는 적어도 제1 기준에 기초함 -;

상기 실행 이벤트와 연관되는 제1 정보를 실행 이벤트 큐에 저장하고; 및

상기 실행 이벤트 큐에서의 상기 실행 이벤트와 연관되는 상기 제1 정보에 적어도 부분적으로 기초하여 상관 시그니처를 검증하게 하는 - 상기 상관 시그니처는 적어도 2개의 이벤트와 연관된 이벤트 정보의 조합을 정의하고, 상기 조합은 악성 코드의 존재를 표시함 -

비-일시적 컴퓨터-판독가능한 저장 매체.

청구항 18

제17항에 있어서,

상기 명령어들은 상기 적어도 하나의 프로세서에 의해 실행될 때, 상기 적어도 하나의 프로세서로 하여금:

컴파일러에 의한 상기 스크립트의 컴파일 동안 상기 스크립트의 대입문을 평가하고 - 상기 컴파일된 스크립트는 상기 스크립트로부터 발생됨 -;

적어도 제2 기준에 기초하여 컴파일 이벤트를 검출하고; 및

상기 컴파일 이벤트와 연관되는 제2 정보를 컴파일 이벤트 큐에 저장하게 하는 비-일시적 컴퓨터-판독가능한 저장 매체.

청구항 19

제18항에 있어서,

상기 명령어들은, 상기 적어도 하나의 프로세서에 의해 실행될 때, 상기 적어도 하나의 프로세서로 하여금:

상기 컴파일 이벤트 큐에서의 상기 컴파일 이벤트와 연관되는 상기 제2 정보에 부분적으로 기초하여 상기 상관 시그니처를 검증하게 하는 비-일시적 컴퓨터-판독가능한 저장 매체.

청구항 20

제18항에 있어서,

상기 명령어들은, 상기 적어도 하나의 프로세서에 의해 실행될 때, 상기 적어도 하나의 프로세서로 하여금:

상기 제2 기준이 만족되었는지를 결정하기 위해 상기 대입문의 우변 값을 평가하게 하는 비-일시적 컴퓨터-판독가능한 저장 매체.

청구항 21

제18항 내지 제20항 중 어느 한 항에 있어서,

상기 명령어들은, 상기 적어도 하나의 프로세서에 의해 실행될 때, 상기 적어도 하나의 프로세서로 하여금:

상기 제2 기준이 만족되었는지를 결정하기 위해 상기 대입문의 좌변 변수 명을 평가하게 하는 비-일시적 컴퓨터-판독가능한 저장 매체.

청구항 22

제17항 내지 제20항 중 어느 한 항에 있어서,

상기 명령어들은, 상기 적어도 하나의 프로세서에 의해 실행될 때, 상기 적어도 하나의 프로세서로 하여금:

상기 컴파일된 스크립트의 실행이 끝나기 전에 상기 상관 시그니처를 검증하게 하는 비-일시적 컴퓨터-판독가능한 저장 매체.

청구항 23

제17항 내지 제20항 중 어느 한 항에 있어서, 상기 상관 시그니처는 상기 평션에 대입되는 가중에 부분적으로 기초하여 검증되며, 상기 가중은 상기 평션의 상대적 중요성을 나타내는 비-일시적 컴퓨터-판독가능한 저장 매체.

청구항 24

삭제

청구항 25

삭제

발명의 설명

기 술 분 야

[0001] 본 발명은 전체적으로 네트워크 보안 분야와 관련되는데, 보다 상세하게는 네트워크 환경에서의 악성 스크립트 언어 코드(malicious scripting language code)의 검출과 관련된다.

배 경 기 술

[0002] 컴퓨터 네트워크 보안 분야는 오늘날의 세상에서 점점 더 중요해지고 복잡해지고 있다. 컴퓨터 네트워크 환경들이 전형적으로 다중의 상호 연결된 컴퓨터(예를 들어, 최종 사용자 컴퓨터들, 랩톱들, 서버들, 모바일 장치들, 기타 등등)를 가진 사실상 모든 기업 또는 조직들을 위해 구성된다. 전형적으로, 이들 네트워크 환경들은 인터넷과 같은 WAN(wide area network)들을 통해 다른 네트워크 환경들과 통신하도록 구성된다. 그러므로, 다양한 파일들이 동일하거나 상이한 네트워크들에서 어느 한 말단 호스트에서 또 다른 말단 호스트로 전송될 수 있다. 이들 파일들은, 예를 들어 이메일('email') 메시지들, HTTP(HyperText Transfer Protocol), FTP(file transfer protocol), 및 피어 투 피어 파일 공유를 포함하는 임의의 적합한 전자적 통신 메커니즘을 통해 전송될 수 있다.

[0003] 실행 가능 소프트웨어 파일들 및 그 외의 오브젝트들을 스캐닝하는 것은 악성 코드("멀웨어(malware)") 또는 컴퓨터에 대한 다른 위협들을 검출하기 위해 종종 이용되는 기술이다. 멀웨어에 대해 오브젝트들을 스캐닝하는 것은 알려진 악성 코드의 정적 시그니처(static signature)를 이용하여 일반적으로 실행된다. 컴퓨터 공격자들은 악성 코드를 그렇지 않은 경우에는 선의인 파일들(benign files)에 삽입하기 위해, 자바스크립트와 같은 스

크립트 언어를 종종 이용한다. 스크립트 언어는 속성상 동적일 수 있고, 이 밖에도 악성 코드의 검출을 추가로 방해하기 위해 난독화 기술(obfuscation techniques)들이 일반적으로 이용된다. 그러므로, 컴퓨터 네트워크 환경들에 걸쳐서 전송되는 오브젝트들에서 스크립트 언어 형태로 된 악성 코드를 검출하기 위해 독창적 도구들이 필요하다.

도면의 간단한 설명

[0004] 본 개시와 이것의 특징 및 장점에 대한 철저한 이해를 도모하기 위해서, 첨부된 도면들과 연계하여 취해지는 하기 설명에 대해 참조가 이루어지는데, 이 도면들에서 유사 참조 번호들은 유사 부분들을 나타낸다:

도 1은 예시적 실시예에 따라 네트워크 환경에서 악성 스크립트 언어 코드의 검출을 위한 통신 시스템을 예시하는 단순화된 블록도이다;

도 2는 예시적 실시예에 따른 시스템의 추가적인 상세 사항들들을 예시하는 단순화된 블록도이다;

도 3은 본 개시의 실시예들과 연관될 수 있는 예시적 동작들을 설명하는 단순화된 흐름도이다;

도 4는 본 개시의 실시예들과 연관될 수 있는 추가적 예시적 동작들을 설명하는 단순화된 흐름도이다;

도 5는 본 개시의 실시예들과 연관될 수 있는 추가적 예시적 동작들을 설명하는 단순화된 흐름도이다;

도 6은 본 개시의 실시예들과 연관될 수 있는 또 다른 예시적 동작들을 설명하는 단순화된 흐름도이다;

도 7은 실시예에 따른 예시적 프로세서의 단순화된 블록도이다; 및

도 8은 실시예에 따른 예시적 컴퓨팅 시스템의 단순화된 블록도이다.

발명을 실시하기 위한 구체적인 내용

[0005] 도 1은 악성 스크립트 언어 코드의 검출을 위한 통신 시스템(100)의 예시적 구현을 도해하는 단순화된 블록도이다. 말단 호스트들(120_1 내지 120_n)이 네트워크(115)에 제공될 수 있고, 이것은 근거리 네트워크(LAN), 인트라넷, 또는 인터넷과 같은 광역 네트워크(WAN(110))에의 및/또는 다른 네트워크들에의 액세스를 제공하는 다른 네트워크일 수 있다. 네트워크(115)는 WAN(110)과 같은 다른 네트워크들로부터 네트워크(115)에 들어오는 네트워크 트래픽을 수신하는 보안 장치(130)를 또한 포함할 수 있다. 보안 장치는 또한 WAN(110)과 같은 다른 네트워크들에게 나가는 네트워크 트래픽을 또한 수신할 수 있다. 보안 장치(130)는 표준 검출 모듈(140), 전처리 모듈(145), 스크립트 평가 엔진(150), 평션 평가 모듈(160), 및 사후 평가 모듈(170)을 포함할 수 있다. 보안 장치는 또한 프로세서(180) 및 메모리 성분(190)을 포함할 수 있다. 본 명세서에서 적절할 때, 말단 호스트들(120_{1-n})은 집합적으로 '말단 호스트들(120')로서 지칭되고, 참조 편의성을 위해 '말단 호스트(120)'로서 단수로 지칭된다.

[0006] 도 1의 요소들은 네트워크 통신들을 위한 실행 가능 경로들을 제공하는 임의의 적절한 연결들(유선 또는 무선)을 채택하는 하나 이상의 인터페이스들을 통해 서로 결합될 수 있다. 덧붙여, 도 1의 이들 요소들 중 임의의 하나 이상은 특정 구성 필요에 기초하여 아키텍처에 조합될 수 있거나 그로부터 제거될 수 있다. 통신 시스템(100)은 네트워크에서의 패킷들의 송신 또는 수신을 위한 TCP/IP(transmission control protocol/Internet protocol) 통신을 할 수 있는 구성을 포함할 수 있다. 통신 시스템(100)은 적절한 경우 및 특정 필요에 기초하여 UDP/IP(user datagram protocol/IP) 또는 임의의 다른 적절한 프로토콜과 연계하여 또한 동작할 수 있다. 추가적으로, 셀 방식 네트워크상에서의 무선 신호 통신들이 또한 통신 시스템(100)에 제공될 수 있다.

[0007] 통신 시스템(100)의 기술을 예시하기 위한 목적으로, 도 1에 도시된 네트워크(115)와 같은 주어진 네트워크에 존재할 수 있는 활동들 및 보안 관심사들을 이해하는 것이 중요하다. 하기 기본 정보는 본 개시가 적절히 설명될 수 있는 기초로서 볼 수 있다. 그러한 정보는 단지 설명의 목적을 위해 제공된 것이고, 따라서 어떤 식으로든 본 개시 및 그 잠재적 응용들의 넓은 범위를 한정하는 것으로 해석하지 말아야 한다.

[0008] 네트워크 트래픽은 컴퓨터 네트워크에서 장치들에 대한 많은 보안 위협들을 나타낼 수 있다. 보안 위협들은 일반적으로 하기 범주들 중 하나 이상과 맞아 떨어진다: (1) 멀웨어; (2) 스파이웨어; (3) 사적 기밀 위협들; 및 (4) 취약한 애플리케이션들. 멀웨어는 전자 장치에 대한 악의적, 적대적, 침입적, 원치 않는, 및/또는 비인가 거동에 관여하도록 설계되는 소프트웨어 또는 코드를 포함한다. 멀웨어의 예들은 컴퓨터 바이러스들, 웜들, 봇들(bots), 및 트로이 목마들을 포함하지만 이것들에만 한정되지는 않는다. 멀웨어는 일반적으로 비인가 접근,

파괴, 폭로(disclosure), 데이터의 수정, 및/또는 서비스 거부를 통하여 컴퓨터 또는 네트워크의 정상 작동을 방해하도록 설계되는 임의의 소프트웨어를 포함한다. 멀웨어는, 말단 호스트와 같은 장치에의 변화들을 이루거나, 감염된 장치로부터 다른 장치들에게 요구하지 않은(unsolicited) 메시지들을 보내거나, 공격자에게 장치에 대한 원격 제어권을 부여하는 것과 같은 행위들을 사용자에게 알리지 않고 실행하도록 종종 설계된다. 멀웨어는 또한 ID 절도, 금융 사기, 또는 그 외의 사적 기밀(예를 들어, 개인 의료 정보) 침해들을 낳을 수 있는 개인 정보를 장치로부터 훔치는데 사용될 수 있다.

[0009] 스파이웨어는 사용자의 인지 또는 승인 없이 데이터를 수집하거나 이용하도록 설계되는 소프트웨어이다. 사적 기밀 위협들은, 반드시 장치에게 해를 입히는 것은 아닐 수 있지만, 자신들의 주요 평션들을 실행하는 데에 불필요한 정보를 인가 없이 모으거나 이용할 수 있는 애플리케이션들에 의해 야기될 수 있다. 그와 같은 정보의 예들은 사용자의 로케이션, 접촉 목록들, 개인적으로 식별 가능한 정보, 금융 정보, 의료 정보, 기밀이거나 민감한 기업 데이터, 기타 등등을 포함할 수 있지만, 이것들에만 한정되지는 않는다.

[0010] 취약한 애플리케이션들은 악의적 목적들을 위해 악용될 수 있는 소프트웨어 취약점들을 포함할 수 있다. 예를 들어, 취약점들은 종종 공격자로 하여금 민감한 정보에 접근하고, 바람직하지 않은 행위들을 실행하고, 서비스가 올바르게 기능하는 것을 중단시키고, 자동적으로 악성 소프트웨어를 다운로딩하거나, 또는 다른 식으로 바람직하지 않은 거동에 관여하기 위해 악성 코드를 이용하도록 허용할 수 있다. 몇몇 예들에서, 악성 코드는 취약한 애플리케이션의 제어권을 획득하고, 취약한 애플리케이션을 붕괴시키고, 및/또는 취약한 애플리케이션이 그 상에서 실행 중인 장치를 감염시킬 수 있다.

[0011] 네트워크 트래픽은 컴퓨터 네트워크 환경들에서 말단 호스트들 및 네트워크 요소들에게 악성 코드를 포함하는 소프트웨어 애플리케이션들 및 기타 오브젝트들을 전송할 수 있다. 예를 들어, 다운로딩 가능한 오브젝트들, 이메일 메시지들에 첨부된 오브젝트들, 또는 피어 투 피어 통신을 통해 전송되는 오브젝트들은 네트워크 환경들에 걸쳐서 수신하는 말단 호스트 또는 네트워크 요소에의 악성 코드의 전송을 낳을 수 있다. 특히, 자바스크립트와 같은 ECMA 스크립트 또는 이것의 변형들과 같은 스크립트 언어는 HTML(HyperText Markup Language) 파일들 및 PDF(Portable Document Format) 파일들과 같은 파일들에 코드를 임베딩하는데 사용될 수 있다. ECMA 스크립트는 ECMA 인터내셔널에 의해 ECMA-262 사양 및 ISO/IEC 16262으로 표준화된 스크립트 언어이다. 자바스크립트 코드가, 예를 들어 HTML 파일들에 대한 <script> 태그를 이용하여 또는 PDF 파일들에 대한 /JS 또는 /JavaScript를 이용하여 스크립트로서 파일들에 임베딩될 수 있다. 자바스크립트는 직접적으로 파일 내로 임베딩될 수 있거나 또는 자바스크립트를 포함하는 외부 파일이 특정될 수 있다.

[0012] 여러가지 유형의 보안 솔루션들이 멀웨어 공격들을 방지하기 위해, 컴퓨터들에 대한 멀웨어 및 다른 위협들을 검출하기 위해, 및/또는 필요한 경우 컴퓨터들을 치료하기 위해 시도하는 데에 사용될 수 있다. 예를 들어, 시그니처 기반 위협 검출은 오브젝트 내의 알려진 데이터의 패턴들을 검색하는 것을 수반하는 흔한 바이러스 차단 기술이다. 침입 방지 시스템(intrusion prevention system: IPS)이 한 예이고, 네트워크상의 모든 트래픽을 듣고 또한 네트워크 트래픽 내에서의 악성 패턴들 또는 공격들을 찾아 내기 위해 트래픽을 파싱하려고 시도하는 네트워크 기반 애플리케이션으로서 구성될 수 있다. IPS는 네트워크 트래픽에서 수신되는 오브젝트들의 시그니처들을 발생하고 또한 이 시그니처들을 현재 알려진 악성 시그니처들에 대하여 비교할 수 있다. 오브젝트의 시그니처는 암호화 해시 함수로부터 발생될 수 있는데, 이것은 오브젝트로부터 데이터의 블록을 취하고 고정 사이즈 비트 스트링을 반환하는 알고리즘이다. 고정 사이즈 비트 스트링은 해시값 또는 시그니처이다. 해시 함수들의 예들은 하기를 포함할 수 있지만 이것들에만 국한되지는 않는다: 1) Network Working Group, Requests for Comments (RFC) 1321, R. Rivest, et al., April 1992에 의해 정의된 메시지 다이제스트 알고리즘(예로, MD5), 및 2) NIST(United States National Institute of Standards and Technology)에 의해 미연방 정보 처리 표준으로서 공표된 보안 해시 알고리즘(SHA-1, SHA-2, SHA3).

[0013] 특정 예에서, 네트워크 트래픽은 웹 서버로부터 말단 호스트상의 브라우저에게 다운로딩되는 PDF 파일일 수 있다. 수신자 말단 호스트가 파일 포맷을 이해하고 해석하기 위해서 무엇이 전송되고 있는지를 알기 때문에, IPS는 해당 네트워크 트래픽을 들을 수 있다. IPS에 의한 표준 검출은 PDF 파일의 하나 이상의 MD5 해시들을 계산하는 것과 하나 이상의 해시들을 알려진 악성 코드의 정적 시그니처들과 비교하는 것을 포함할 수 있다. 일치가 있다면, PDF 파일은 악성인 것으로서 식별될 수 있다.

[0014] 시그니처 검출이 변하지 않는 악성 정적 오브젝트들을 식별하는 데에 유용하기는 하지만, 스크립트 언어들이 검출을 모면하기 위해 소정 오브젝트들에 사용될 수 있다. 유선상에서 보이는 대로의 (즉, 네트워크 트래픽에서의) 자바스크립트는 무형식 텍스트(free form text)이다. 자바스크립트의 동적 속성은 공격자들로 하여금 IPS

시그니처 검출을 우회할 수 있는 다중 회피 기술을 도입하도록 허용한다. 각각의 자바스크립트에 대해 새로운 회피가 생성될 수 있다. 그러므로, 정적 시그니처들을 이용하는 표준 시그니처 검출 기법은 악성 코드를 검출하는 데 실패할 수 있다.

[0015] 도 1의 악성 스크립트 언어 코드의 검출을 위한 통신 시스템은 전술한 문제들(및 더 많은 것)을 해결한다. 스크립트 언어 코드를 가진 오브젝트가 네트워크 트래픽으로 보안 장치에 의해 수신될 때, 자바스크립트와 같은 스크립트 언어 코드가 추출된다. 통신 시스템(100)이 다른 형태의 스크립트 언어를 수용할 수 있지만, 참조 편의성을 위해 '자바스크립트'가 본 실시예들의 다양한 양태들을 기술하기 위해 본 명세서에서 참조될 것이다. 공격자에 의해 이용되는 잠재적 회피 기술들을 검출하기 위해 휴리스틱(heuristics)이 자바스크립트 코드의 컴파일(compilation) 동안 적용된다. 회피 기술들은 스크립트에서 데이터 또는 코드를 은닉시키는데 사용되는 인코딩, 디코딩 및/또는 계산들을 포함할 수 있다. 이들 회피 기술들 중 임의의 것이 검출되면, 컴파일 이벤트가 제기되고 또한 이 이벤트와 연관되는 정보가 저장되고 사후 평가 분석(post evaluation analysis) 동안 다른 이벤트들과 상관(correlate)될 수 있다.

[0016] 자바스크립트가 컴파일된 후, ('관련 평션들'로서 본 명세서에서 지칭되는) 소정 평션들에 대한 호출들을 모니터링하는 한편 컴파일된 스크립트가 실행되는데, 이것은 종종 스크립트의 회피 설계 및 난독화에 있어서 공격자를 돕는다. 휴리스틱 및 트리거링 기반 메커니즘은 관심 포인트들에서 이 흐름을 캡처할 수 있다. 실행에서의 관심 포인트들은 실행 스크립트가 잠재적으로 악성인 것을 표시하는 이벤트들을 포함할 수 있다. 그와 같은 이벤트들은 코드가 힙 스프레이(heap spray)를 실행하는 것과 같은 코드의 난독화를 표시하는 평션들을 포함할 수 있다. 또 다른 관심 포인트는 셸코드(shellcode)의 식별을 포함하는데, 이것은 감염된 말단 호스트상에서 실행하기 위해 공격자에 의해 이용되는 악성 코드이다. 이벤트들과 연관되는 정보는 사후 실행 분석에 대해 저장될 수 있다. 컴파일 이벤트들 및/또는 실행 이벤트들과 연관되는 이벤트 정보는 악의적 거동을 검증하기 위해 상관될 수 있다. 자바스크립트와 거동을 캡처하고 이벤트 정보를 상관시키는 것은 자바스크립트의 흐름의 핵심을 드러낼 수 있고, 이것은 자바스크립트가 임의의 악의적 행위들을 취하려고 의도하는지를 결정하는 것을 도울 수 있다.

[0017] 도 2를 참조하면, 도 2는 통신 시스템(100)과 연관되는 상세 사항들의 가능한 세트를 도해하는 단순화된 블록도이다. 스크립트 평가 엔진(150)은 컴파일러(152), 컴파일 시간 휴리스틱 검출 모듈(154), 및 실행 엔진(156)을 포함할 수 있다. 컴파일러(152)는 자바스크립트(200)와 같은 입력 파일을 수신할 수 있고, 입력 파일을 컴파일한 후에 컴파일된 스크립트(300)를 실행 엔진(156)에게 공급할 수 있다. 컴파일된 스크립트가 실행 엔진(156)에게 제공되기 전에, 컴파일 시간 휴리스틱 검출 모듈(154)은 컴파일 동안 회피 및/또는 난독화 기술들의 휴리스틱 검출을 실행할 수 있고 또한 컴파일 이벤트 큐(157)에 검출된 컴파일 이벤트들을 더할 수 있다. 실행 엔진(156)은 평션 평가 모듈(160) 및 사후 평가 모듈(170)과 상호 작용할 수 있다. 평션 평가 모듈(160)은 이벤트 분석 모듈(162) 및 관계 데이터 분석 모듈(166)을 포함할 수 있는데, 이것들은 제각기 힙 스프레이 검출(164) 및 셸코드 검출(168)을 실행할 수 있다. 평션 평가 모듈(160)은 검출된 실행 이벤트들과 연관되는 정보를 실행 이벤트 큐(159)에 추가할 수 있다. 사후 평가 모듈(170)은 컴파일 이벤트 큐(157) 및 실행 이벤트 큐(159)로부터 상관 시그니처들(175) 및 정보에 기초하여 악성 이벤트 상관들(172)을 실행할 수 있다. 도 1-2의 아키텍처들과 연관되는 잠재적 흐름들을 논의하기 전에, 통신 시스템(100)에 포함될 수 있는 가능한 인프라스트럭처의 일부에 대한 간략한 논의가 제공된다.

[0018] 일반적으로, 통신 시스템(100)은 임의 유형의 또는 토폴로지의 네트워크들로 구현될 수 있다. 네트워크(115) 및 광역 네트워크(WAN)(110) 각각은 통신 시스템(100)을 통하여 전파되는 정보의 패킷들을 수신하고 전송하기 위한 상호 연결된 통신 경로들의 포인트들 또는 노드들의 시리즈를 나타낸다. 이들 네트워크들은 노드들 사이의 통신 인터페이스를 제공하고, 또한 임의의 LAN(local area network), 가상 LAN(VLAN), WAN(wide area network), WLAN(wireless LAN), MAN(metropolitan area network), 인트라넷(Intranet), 엑스트라넷(Extranet), VPN(virtual private network), 및 네트워크 환경에서의 통신을 용이하게 하는 임의의 다른 적절한 아키텍처 또는 시스템, 또는 유선 및/또는 무선 통신을 포함하는 이것들의 임의의 적절한 조합으로서 구성될 수 있다.

[0019] 통신 시스템(100)에서, 패킷들, 프레임들, 신호들, 데이터, 기타 등등을 포함하는 네트워크 트래픽이 임의의 적절한 통신 메시징 프로토콜들에 따라 송신되고 수신될 수 있다. 적절한 통신 메시징 프로토콜들은 OSI(Open Systems Interconnection) 모델, 또는 이것들의 임의의 파생물들 또는 변형들 (예를 들어, TCP/IP(Transmission Control Protocol/Internet Protocol), UDP/IP(user datagram protocol/IP))과 같은 다층 스킴을 포함할 수 있다. 덧붙여, 셀 방식 네트워크상에서의 무선 신호 통신들도 통신 시스템(100)에 제공될

수 있다. 적절한 인터페이스들 및 인프라스트럭처가 셀 방식 네트워크와의 통신을 가능하게 하기 위해 제공될 수 있다.

[0020] 본 명세서에 사용되는 용어 '데이터'는, 임의 유형의 이진, 수치, 음성, 비디오, 미디어, 텍스트, 또는 스크립트 데이터, 또는 임의 유형의 소스 또는 오브젝트 코드, 및/또는 전자 장치들 및/또는 네트워크들에서의 어느 한 지점에서 또 다른 지점으로 통신될 수 있는 임의의 적절한 포맷으로 된 임의의 기타 정보를 지칭한다. 오브젝트는 컴퓨터에 의해 이해되고 처리될 수 있는 적어도 몇몇 명령어들을 포함하는 임의의 소프트웨어 파일 또는 기타 데이터를 포함하도록 의도된다. 오브젝트들은, 예를 들어, 실행 가능 파일들, 라이브러리 모듈들, 오브젝트 코드, 소스 코드, 기타 실행 가능 모듈들, 스크립트 파일들, 인터프리터 파일들, 자바스크립트와 같은 임베딩된 스크립트를 가진 PDF 파일들, 자바스크립트와 같은 임베딩된 스크립트를 가진 HTML 파일들, 기타 등등을 포함할 수 있다.

[0021] 가능한 실시예에 있어서, 통신 시스템(100)은 WAN(110) 또는 임의의 다른 네트워크를 통해 말단 호스트들(120)과 다른 말단 호스트들 간에서 통신되고 있는 네트워크 트래픽에서 오브젝트들을 수신하기 위한 보안 장치(130)를 포함한다. 보안 장치(130)는, 라우터들, 스위치들, 게이트웨이들, 브리지들, 로드 밸런서들, 방화벽들, 인라인 서비스 노드들, 프록시들, 서버들, 어플라이언스들, 프로세서들, 모듈들, 또는 네트워크 환경에서 정보를 교환하도록 동작 가능한 임의의 다른 적절한 장치, 컴포넌트, 요소, 또는 독점적 장치를 포괄하는 것으로 의도되는 네트워크 요소일 수 있다. 이 네트워크 컴포넌트는 임의의 적절한 하드웨어, 소프트웨어, 컴포넌트들, 모듈들, 엔진들, 또는 이것들의 동작들을 용이하게 하는 인터페이스들을 포함할 수 있다. 이것은 데이터의 효과적 교환을 허용하는 적절한 알고리즘 및 통신 프로토콜을 포함할 수 있다.

[0022] 말단 호스트들(120)은 몇몇 네트워크를 통해 통신 시스템(100) 내에서 음성, 오디오, 비디오, 미디어, 및/또는 데이터 교환들을 실행할 수 있는 임의의 전자 장치, 컴포넌트, 또는 요소를 포함하도록 의도된다. 말단 호스트들은 개인용 컴퓨터들, 랩톱들, 모바일 장치들, 스마트 어플라이언스들, 및 다른 인터넷 연결된 장치들(예를 들어, 텔레비전, DVR(digital video recorder), 셋톱박스, IRD(Internet Radio Device), 기타 등등)을 포함한다. 모바일 장치들은 모바일 폰들, 스마트 모바일 폰들(스마트폰들), 전자책 리더들, 태블릿들, 아이패드들, PDA들(personal digital assistants), 랩톱들 또는 전자 노트북들, 휴대용 내비게이션 시스템들, 멀티미디어 기기들(예를 들어, 카메라들, 비디오 및/또는 오디오 플레이어들, 기타 등등), 게임 시스템들, 기타 핸드헬드 전자 장치들, 기타 등등을 포함하도록 의도된다. 말단 호스트들은 또한 WAN(110) 또는 몇몇 다른 네트워크를 통해 오브젝트들을 수신할 수 있는 서버들과 같은 네트워크 요소들을 나타낼 수 있다. 말단 호스트는 네트워크 트래픽의 발신지 노드로서 및/또는 수신지 노드로서 기능할 수 있다.

[0023] 다양한 실시예들에서, 보안 장치(130)는, 본 명세서에서 개괄된 것처럼, 악성 스크립트 언어 코드의 검출을 달성하기 위해 조정하고, 관리하고, 또는 다른 식으로 협력할 수 있는 로직 (및/또는 교환 로직(reciprocating logic))을 포함한다. 각각의 이들 요소들은 검출 동작들의 일부를 용이하게 하기 위해, 본 명세서에서 추가로 기술되는 것처럼, 내부 구조(예를 들어, 프로세서, 메모리 성분, 기타 등등)를 가질 수 있다는 것을 유의한다. 검출 활동들의 로직은 소프트웨어, 하드웨어, 펌웨어, 또는 이것들의 임의의 적절한 조합으로 구현될 수 있다. 또한, 이들 요소들은 이것들의 동작을 용이하게 하는 임의의 적절한 알고리즘들, 하드웨어, 펌웨어, 소프트웨어, 컴포넌트들, 모듈들, 엔진들, 인터페이스들, 또는 오브젝트들을 포함할 수 있다.

[0024] 몇몇 실시예들에서, 일부 또는 모든 검출 활동들은 이들 의도된 기능성들을 달성하기 위해 기타 장치들에 포함되거나 또는 임의의 적절한 방식으로 통합되어 이들 요소들에 외부적으로(예를 들어, 보안 장치(130)에 외부적으로) 제공될 수 있다. 한 예시적 구현에서, 스크립트 평가 엔진(150), 평션 평가 모듈(160), 및/또는 사후 평가 모듈(170)에 의해 제공되는 기능성은 WAN(110)(예를 들어, 인터넷) 또는 임의의 다른 네트워크를 통해 접근 가능 클라우드 네트워크에 제공될 수 있다. 또 다른 예시적인 구현에서, 스크립트 평가 엔진(150), 평션 평가 모듈(160), 및/또는 사후 평가 모듈(170)에 의해 제공되는 기능성은 하나 이상의 말단 호스트들(120) 또는 네트워크 환경(115)의 임의의 또 다른 네트워크 요소에 제공될 수 있다.

[0025] 컴파일 이벤트 큐(157), 실행 이벤트 큐(159), 및 상관 시그니처들(175)은 하나 이상의 큐들, 캐시들, 테이블들, 리스트들, 데이터베이스들, 또는 임의의 다른 적절한 스토리지 구조, 또는 이것들의 임의의 적절한 조합으로서 구성될 수 있다. 컴파일 이벤트 큐(157), 실행 이벤트 큐(159), 및 상관 시그니처들(175)은, 예를 들어 NAS(network attached storage) 또는 SAN(storage area network)과 같은 몇몇 네트워크 스토리지 기술을 이용하여 보안 장치(130) 내부에 또는 외부에 (전체적으로 또는 부분적으로) 있을 수 있다. 대안적으로, 이들 메모리 성분들은 스크립트 평가 엔진(150), 평션 평가 모듈(160), 및/또는 사후 평가 모듈(170)을 가진 다른 네

트워크 요소들 또는 말단 호스트들에 구성될 수 있다.

[0026] 보안 장치(130)는 잠재적으로 악의 있는 것으로 알려진 소정 패턴들을 평가하기 위한 표준 검출 모듈(140)을 포함할 수 있다. 한 예에서, 표준 검출 모듈(140)은 수신된 오브젝트의 시그니처(또는 다중 시그니처)를 계산하고, 계산된 시그니처와 알려진 악성 시그니처들의 비교에 기초하여 오브젝트가 악성 코드를 포함하는지를 결정한다.

[0027] 오브젝트가 표준 검출 메커니즘들을 통과하면, 전처리 모듈(145)은 오브젝트로부터 자바스크립트(200)와 같은 스크립트 부분들을 획득하기 위해 오브젝트를 전처리할 수 있다. 전처리는 오브젝트의 사전 필터링, 추출, 압축 해제, 및 위생 처리(sanitization)를 포함하여 자바스크립트(200)가 예를 들어 말단 호스트(120)에게 전송되도록 본질적으로 구성되게 할 수 있다. 초기에, 오브젝트는 유선으로부터 추출되고 또한 말단 호스트(120)가 수신하는 최초 포맷으로 될 수 있다. 이 오브젝트는 (예를 들어, 사인(sign)들보다 더 작은 것 및 더 큰 것을 발견함으로써) 자바스크립트 코드를 발견하도록 파싱될 수 있다. 식별된 스크립트 및 다른 관련 정보(예를 들어, 이벤트 기반 트리거 펑션들, 폼들, PDF 파일들을 위한 메타데이터, 기타 등등)가 추출될 수 있다. 덧붙여, 오브젝트의 임의의 부분들이 압축되어 있다면, 보안 장치(130)는 자바스크립트의 정확한 코드를 얻기 위해 그 부분들을 압축 해제할 수 있다.

[0028] 스크립트 평가 엔진(150)은 자바스크립트가 악성 코드를 포함하는지를 결정하기 위해 고립된 환경에서 자바스크립트(200)의 실행을 에뮬레이팅한다. 스크립트 평가 엔진은 자바스크립트(200)로 하여금 마치 자신이 실제 애플리케이션에서 실행하는 것처럼 작업하도록 허용하는 최소 실행 환경을 생성하기 위한 컴파일러(152) 및 실행 엔진(156)을 포함한다.

[0029] 컴파일 동안, 컴파일러(152)는 자바스크립트(200)를 나타내기 위해 AST(Abstract Syntax Tree)와 같은 트리 구조를 발생할 수 있다. 컴파일러(152)는 종종 '오브젝트 코드'라고 지칭되고 또한 도 2에서 컴파일된 스크립트(300)로서 참조되는 또 다른 컴퓨터 언어로 자바스크립트(200)를 변환한다. 컴파일된 스크립트(300)는 실행 엔진(156)에 의해 실행될 수 있다.

[0030] 본 명세서의 예시적 실시예들에서, 컴파일 시간 휴리스틱 검출 모듈(154)은 종종 악성 코드에 사용되는 소정 회피 및/또는 난독화 기술들을 하나 이상의 기준들에 기초하여 검출하기 위해 컴파일 동안 컴파일러(152)와 협력한다. 회피 기술들은 악성 코드를 난독화하고, 따라서 검출을 더 어렵게 하는 데에 사용될 수 있다. 한가지 회피 기술은 악성 코드의 프레젠테이션을 바꾸기 위한 인코딩 또는 디코딩을 포함한다. 인코딩이 이용될 때, 자바스크립트(200)의 판독 가능 포맷은 몇몇 다른 포맷으로 인코딩될 수 있다. 예시적인 시나리오에서, 인코딩은 공격자에 의해 자바스크립트를 생성하기 위해 이용될 수 있다. 자바스크립트가 실행될 때, 이것은 악성 셸코드를 획득하기 위해 데이터를 디코딩하기 위한 펑션들을 포함할 수 있다. 컴파일 동안 평가될 수 있는 또 다른 회피 기술은 악성 활동에 대해 필요한 값을 획득하기 위해 계산들을 이용하는 것을 수반한다. 값은 자신을 상이한 폼으로 표현함으로써 은닉될 수 있다. 악성 코드에 대한 바라는 값을 획득하기 위해, 계산들이 하나 이상의 상이한 폼들에 대해 실행될 수 있다.

[0031] 회피 기술들은 자신들의 목적들을 성취하기 위해 소정 펑션 호출들을 전형적으로 이용한다. 일반적으로, 한 펑션은 해당 펑션이 호출될 때 실행되는 로직 또는 코드의 블록이다. 회피 기술에 사용될 수 있고 또한 하나 이상의 다른 기준들을 충족하는 소정 펑션들은 컴파일 이벤트들일 수 있고 컴파일 시간 휴리스틱 검출 모듈(154)에 의해 검출 가능할 수 있다. 예를 들어, 컴파일 시간 휴리스틱 검출 모듈(154)은 자바스크립트(200)를 검사할 수 있고 또한 회피 및/또는 난독화를 표시할 수도 있는 소정 펑션 재대입들 및 지나치게 긴 변수 명들을 식별할 수 있다. 그와 같은 펑션들은 인코딩 또는 디코딩 활동들(예를 들어, UNESCAPE 자바스크립트 펑션) 또는 상이한 값들에 대한 계산들(예를 들어, EVAL 자바스크립트 펑션)을 수반할 수 있다. 이 휴리스틱 검출은 코드를, 컴파일 동안 AST를 생성하는 것을 책임지는 코드 내부에 위치시킴으로써 성취될 수 있다.

[0032] 컴파일 시간 휴리스틱 검출 모듈(154)은 이것이 컴파일 이벤트를 검출하면 이 컴파일 이벤트를 제기(raise)할 수 있다. 본 명세서에 사용되는 바로는, '이벤트를 제기한다'라는 것은 특정 이벤트와 연관되는 정보를 적절한 이벤트 큐에 추가하는 것을 포함하도록 의도된다. 그러므로, 컴파일 이벤트를 제기하는 것은 그 외의 컴파일 및/또는 실행 이벤트들과 연관되는 정보와의 상관을 위해 컴파일 이벤트와 연관되는 정보를 컴파일 이벤트 큐(157)에 추가하는 것을 포함할 수 있다. 예시적 실시예에서, 컴파일 이벤트와 연관되는 정보는 이벤트를 트리거링한 펑션의 식별, 이벤트가 스크립트에서 검출되는 총 횟수, 컴파일 미리 정해진 이벤트들의 시퀀스의 식별, 및 스크립트에서의 소정의 그 외의 컴파일 이벤트들에 상대적인 특정 컴파일 이벤트의 시간과 거리를 포함할 수 있다. 컴파일된 스크립트(300)가 실행 엔진(156)에 의해 실행된 후에 상관이 실행될 수 있다. 그러나, 몇몇

경우들에서, 상관들은 실행 엔진(156)에 의한 컴파일된 스크립트(300)의 실행 동안 실행될 수 있다.

[0033] 실행 엔진(156)은, 스크립트 평가 엔진(150)에서, 컴파일된 스크립트(300)에서의 평션들이 실행되도록 허용하는 자바스크립트 엔진일 수 있다. 실행 엔진(156)은 컴파일된 스크립트(300)가 마치 자신이 실제 애플리케이션에서 실행되고 있었던 것처럼 실행되도록 허용하기 위해 구성된다. 예시적 구현에서, 코드는 자바스크립트가 호출하는 기존 평션들 내로 후크(hook)된다. 그러므로, 실행 엔진(156)은 실행시간 동안 악성 코드를 검출하는 것과 관련되는 평션(본 명세서에서 '관련 평션'으로서 지칭됨)이 컴파일된 스크립트(300)에서 호출될 때 이 후크된 코드를 실행한다. 일부 평션들은 악성 코드를 검출하는 것과 관련되지 않을 수 있고, 그러므로 후크들을 포함하지 않을 수 있다. 그러나, 관련 평션이 컴파일된 스크립트(300)에서 호출될 때, 후크된 코드는 실행 엔진(156)으로 하여금 제어를 평션 평가 모듈(160)에게 넘기도록 허용한다.

[0034] 관련 평션이 컴파일된 스크립트(300)의 실행 동안 호출될 때, 실행 엔진(156)은 관련 데이터를 평션 평가 모듈(160)에게 전송한다. 관련 데이터는 현재 실행의 평션과 문맥(context)을 결정하기 위한 식별 번호를 포함할 수 있다. 평션 평가 모듈(160)은 관련 평션의 최초 코드를 실행하고 또한 악의적 활동의 표시들을 위해 이것의 거동을 분석할 수 있다. 평션과 그 파라미터들의 문맥은 그것의 거동을 분석하는데 사용될 수 있다.

[0035] 평션 평가 모듈(160)은 하나 이상의 기준들에 기초하여 상이한 실행 이벤트들을 검출하기 위해 다양한 분석들을 실행하기 위한 하나 이상의 모듈들을 포함할 수 있다. 실행 이벤트들은 이벤트 분석 모듈(162)과 관계 데이터 분석 모듈(166)에 의해 이들이 관련 평션들을 평가할 때 검출될 수 있다. 예시적 실시예에서, 실행 이벤트는 다음을 포함할 수 있다: 1) 미리 정의된 임계값을 충족하는 평션의 파라미터; 2) 회피 및/또는 난독화 기술들에 사용될 수 있는 평션; 3) 악의적 거동에 사용되는 것으로 높은 신뢰도로 알려진 평션, 및/또는 4) 자신이 특정 오브젝트 타입들에 의해 이용되면 악의적 거동을 위해 비정상적으로 이용될 수 있는 평션. 평션 평가 모듈(160)은 자신이 실행 이벤트를 검출하면 실행 이벤트를 제기할 수 있다. 실행 이벤트를 제기하는 것은 실행 이벤트 큐(159)에 실행 이벤트와 연관되는 정보를 추가하는 것을 포함할 수 있다. 예시적 실시예에서, 실행 이벤트와 연관되는 정보는 이벤트를 트리거링한 평션의 식별, 이벤트가 실행 스크립트에서 발생한 총 횟수, 컴파일 및/또는 실행 미리 정해진 이벤트들의 시퀀스의 식별, 및 실행 스크립트에서의 소정의 그 외의 실행 이벤트들에 상대적인 특정 실행 이벤트의 시간과 거리를 포함할 수 있는데, 이것들에만 제한되지는 않는다.

[0036] 미리 정해진 이벤트들의 시퀀스는, 스크립트의 컴파일 및/또는 실행 동안 제기될 때, 잠재적으로 악의적 거동을 표시하는 둘 이상의 특정 이벤트들을 정의할 수 있다. 그러므로, 검출된 미리 정해진 이벤트들의 시퀀스의 식별은 컴파일 또는 실행 이벤트가 제기될 때 저장될 수 있다. 미리 정해진 이벤트들의 시퀀스는 그 정의에서 특정 이벤트들이 무작위적으로 발생하거나 실행되고, 또는 특정 순서로 발생하거나 실행되는 것을 요구할 수 있다. 예시적 시나리오에서, 미리 정해진 이벤트들의 시퀀스는 그 정의에서, 이벤트들이 컴파일 동안 발생하거나 또는 이벤트들이 실행되는지에 상관없이, 동일 변수를 수반하는 둘 이상의 이벤트들을 포함할 수 있다. 이 경우에, 이전 이벤트가 그에 대해 제기된 변수를 수반하는 제2 이벤트(컴파일 또는 실행)가 제기될 때, 미리 정해진 이벤트들의 시퀀스가 검출될 수 있고 정보(예를 들어, 시퀀스의 식별)는 적절한 이벤트 큐(157 또는 159)에 추가될 수 있다.

[0037] 실시예에서, 이벤트 분석 모듈(162)은 파라미터들과 관계되는 실행 이벤트들을 검출하고 제기할 수 있다. 실행 이벤트는 하나 이상의 파라미터들이 한 평션에게 전달되고 또한 파라미터들 중 임의의 것이 미리 정의된 임계값을 만족할 때 제기될 수 있다. 일 예에서, 파라미터에서의 캐릭터들의 수가 계수될 수 있다. 캐릭터들의 수가 파라미터 캐릭터들에 대한 미리 정의된 임계값보다 더 크다면 실행 이벤트는 제기될 수 있다. 제기된 실행 이벤트와 연관되는 정보는 실행 이벤트 큐(159)에 추가될 수 있다.

[0038] 실시예에서, 이벤트 분석 모듈(162)은 회피 및/또는 난독화 기술들과 관계되는 실행 이벤트들을 검출하고 제기할 수 있다. 컴파일 시간에 더하여 실행 시간 동안 분석될 수 있는 한가지 회피 기술은 악성 코드의 프레젠테이션을 바꾸기 위한 인코딩 및/또는 디코딩을 수반한다. 인코딩 또는 디코딩 평션이 큰 스트링에 대해 사용되고 또한 결과적 스트링의 길이가 미리 정해진 임계 길이보다 더 크다면, 실행 이벤트가 제기될 수 있다. 예를 들어, UNESCAPE 자바스크립트 평션이 호출되고 결과적 스트링 길이가 700 이진/원시 바이트들보다 더 크다면, 실행 이벤트가 제기될 수 있다. 그에 따라, 제기된 실행 이벤트와 연관되는 정보는 실행 이벤트 큐(159)에 추가될 수 있다.

[0039] 컴파일 시간에 더하여, 실행시간 동안 분석될 수 있는 또 다른 회피 기술은 표현들의 계산들을 수반한다. 계산이 표현에 대해 실행되면, 결과는 악성 코드의 일부이거나 악성 코드에 의해 이용되는 또 다른 스트링을 포인팅하는데 사용될 수 있다. 그러므로, 계산 평션이 실행되고 또한 데이터 크기가 미리 정해진 임계 사이즈보다 더

크다면, 실행 이벤트가 제기될 수 있다. 예를 들어, EVAL 자바스크립트 평션이 호출되고 EVAL 데이터가 1024 이진/원시 바이트들보다 더 크다면, 실행 이벤트가 제기될 수 있다. 그에 따라서, 제기된 실행 이벤트와 연관되는 정보는 실행 이벤트 큐(159)에 추가될 수 있다.

[0040] 실행시간 동안 분석될 수 있는 또 다른 회피 기술은 데이터의 스트링에 대한 또는 변수에 대한 포인터들을 수반하는데, 여기서 스트링 또는 변수의 일부는 셸코드를 포함한다. 포인터는 스트링 내의 바이트 로케이션(예를 들어, 2000 바이트 스트링 중의 20 바이트에 대한 바이트 로케이션 1000) 또는 변수 내의 바이트 로케이션을 표시할 수 있다. 지정된 바이트들은 셸코드를 포함할 수 있고, 이것은 이 정보를 이용하여 추출되고 실행될 수 있다. 대안적으로, 이 기술은 스트링 또는 변수 내의 로케이션으로부터 셸코드를 추출하고 이후 이것을 다른 로케이션들에 저장된 셸코드의 다른 청크(chunk)들과 연접(concatenate)시키는데 사용될 수 있다. 포인터들을 회피 기술로서 사용할 수 있는 평션들의 특정 예들은 STRINGFROMCHARCODE, CHARCODEAT, 또는 CHARAT 자바스크립트 평션들을 포함한다. 회피 및/또는 난독화 목적들을 위한 이들 평션들의 사용은 매우 보편적이기 때문에, 이들 평션들의 실행만으로도 실행 이벤트로서 구성될 수 있다. 따라서, 몇몇 실시예들에서, 이들 평션들이 실행될 때, 실행 이벤트는 임계 파라미터를 만족하지 않고서 제기될 수 있다. 제기된 실행 이벤트와 연관되는 정보는 실행 이벤트 큐(159)에 추가될 수 있고 실행 이벤트들은 이후의 다른 제기된 이벤트들과 상관될 수 있다.

[0041] 또 하나의 회피 기술은 셸코드(즉, 악성 코드)를 상이한 코드의 청크들로 나누고 이 청크들을 상이한 로케이션들에 저장하는 것을 수반한다. 청크들은 이후에 하나의 코드 피스가 되도록 연접될 수 있다. 다양한 로케이션들에서의 코드의 다중 청크들은 검출 기회들을 감소시킨다. 스트링들에 대한 연접 평션이 호출될 때, 이것은 연접된 스트링의 최종 길이가 미리 정해진 임계 길이를 만족하면 이벤트로서 제기될 수 있다. 예를 들어, STRCAT 자바스크립트 평션이 호출되고 연접된 스트링의 결과적 길이가 2*1024 바이트들보다 더 크다면 실행 이벤트가 제기될 수 있다. 그에 따라서, 제기된 실행 이벤트와 연관되는 정보는 실행 이벤트 큐(159)에 추가될 수 있다.

[0042] 스트링 연접 기술과 유사한 회피 기술은 어레이들을 연접하는 것을 포함한다. 어레이들은 바라는 데이터를 가진 매우 다량의 메모리를 더 효율적으로 플러싱(flush)하기 위해 연접될 수 있다. 특정 시나리오에서, 힙 스프레이는 NOP(No Operation) 명령어들이 메모리의 큰 청크에 저장되고 악성 코드가 해당 청크 내의 어딘가에 저장되는 악의적 기술이다. 힙 스프레이에 대한 성공 확률은 메모리에 저장되는 NOP 명령어들의 수가 증가함에 따라 증가한다. 어레이들을 연접하는 것은 NOP들을 메모리의 매우 큰 청크에 저장하기 위한 효과적 방식이다. 그러므로, 어레이 연접 평션이 호출되면, 이것은 연접된 어레이들의 최종 길이가 미리 정해진 임계 길이를 만족하면 실행 이벤트로서 제기될 수 있다. 예를 들어, ARRAY.CONCAT 자바스크립트 평션이 호출되고 연접된 어레이의 결과적 길이가 2*1024 바이트보다 더 크다면, 실행 이벤트가 제기될 수 있다. 그에 따라서, 제기된 실행 이벤트와 연관되는 정보는 실행 이벤트 큐(159)에 추가될 수 있다.

[0043] NOP들을 인코딩하는 것은 또한 검출을 피하기 위한 회피 기술로서 이용될 수 있다. 그러므로, 디코딩 평션이 검출되고 NOP 명령어들 또는 다른 무위(do-nothing) 명령어들이라고 알려진 캐릭터들을 낳는다면, 실행 이벤트가 제기될 수 있다. 특정 예에서, UNESCAPE 자바스크립트 평션이 이용되고 결과적 캐릭터들이 0x9090, 0x4141, 또는 0x0c0c이면, 실행 이벤트가 제기될 수 있다. 그에 따라서, 제기된 실행 이벤트와 연관되는 정보가 실행 이벤트 큐(159)에 추가될 수 있다.

[0044] 회피를 달성하는 또 다른 방식은 스트링을 구축하고 스트링 내의 한 로케이션에서 일부 데이터를 교체하는 것을 수반하는데, 여기서 데이터는 악성 코드이거나 궁극적으로 악성 코드로 변환되는 것이다. 이 시나리오에서, 임의의 주어진 시간에, 스트링의 패턴 일치는 알려진 악성 코드와의 일치를 산출하지 않을 것이다. 그러므로, 교체 평션이 호출되고 또한 교체된 데이터의 길이가 미리 정해진 임계 길이를 만족하면 실행 이벤트가 제기될 수 있다. 예를 들어, REPLACE 자바스크립트 평션이 호출되고 또한 교체된 데이터의 길이가 512 바이트보다 더 크면, 실행 이벤트가 제기될 수 있다. 그에 따라서, 제기된 실행 이벤트와 연관되는 정보가 실행 이벤트 큐(159)에 추가될 수 있다.

[0045] 실시예에서, 평션 평가 모듈(160)의 관계 데이터 분석 모듈(166)은, 평션이 호출되고 또한 평션이 악의적 활동들에 사용되는 것으로 높은 신뢰도로 알려져 있을 때 실행 이벤트들을 검출하고 제기할 수 있다. 높은 신뢰도의 평션의 한 예는 동적 평션 생성을 실행하는 것이다. 일반적으로, 자바스크립트가 웹 페이지 또는 PDF 오브젝트를 통해 다운로딩될 때, 자바스크립트가 오브젝트에 임베딩되면, 이것은 전형적으로 매우 단순하다. 예를 들어, 그와 같은 자바스크립트는 HTML 페이지를 렌더링하는 것을 돕기 위해 구성될 수 있다. 동적 평션들은 임의의 콘텐츠를 렌더링하는 데에 일반적으로 필요하지 않고, 그러므로 동적 평션들은 악의적 거동의 큰

표시이다. 그러므로, 동적 평션 생성이 검출될 때, 실행 이벤트가 제기될 수 있고 동적 평션 생성과 연관되는 정보는 실행 이벤트 큐(159)에 저장될 수 있다. 실시예에서, 평션 보디 자체가 또한 제기되고 저장될 수 있다.

[0046] 높은 신뢰도의 평션의 또 다른 예는 말단 호스트상의 취약점들의 셸코드 부당 활용(exploitation)이다. 관계 데이터 분석 모듈(166)은 실행 평션이 말단 호스트상의 취약점들을 부당 활용하도록 구성되는지를 결정하기 위해 셸코드 검출(168)을 실행할 수 있다. 그와 같은 부당 활용의 예는 평가된 자바스크립트를 포함하는 오브젝트와 연관되는 소프트웨어 애플리케이션에 대한 제어를 획득하는 것을 수반한다. 예를 들어, 자바스크립트(200)가 PDF 오브젝트에 임베딩되면, Adobe PDF 리더가 PDF 오브젝트를 판독하기 위해 수신지 말단 호스트상에서 이용될 수 있다. Adobe PDF 리더가 임의의 취약점들을 포함하면, 이후 악성 자바스크립트 평션이 리더에 대한 제어를 획득하고, 잠재적으로 리더를 붕괴시키고, 및/또는 Adobe PDF 리더를 실행하는 말단 호스트의 시스템을 감염시킬 수 있다.

[0047] 기술된 셸코드 부당 활용을 달성하기 위해, 자바스크립트(200)는 악성 코드로 하여금 특정 프로세서를 갖는 말단 호스트 내의 네이티브 명령어들로서 실행되는 것을 허용하는 셸코드를 포함할 수 있다. 예를 들어, 말단 호스트의 취약점들을 부당 활용하는 악성 코드는 말단 호스트의 특정 프로세서에 의해 이해할 수 있는 명령어들을 포함하는 포맷으로 기입될 수 있다. 그러므로, 말단 호스트는 악성 코드를 실행할 수 있고, 그러므로 악성 코드에 의해 제어되게 된다.

[0048] 셸코드 검출(168)의 실시예에서, 컴파일된 스크립트 자체가 조사될 수 있다. 컴파일된 스크립트(300)가 수신지 말단 호스트의 특정 프로세서가 이해할 수 있는 임의의 명령어들을 포함하면, 이것은 악성 소프트웨어의 높은 신뢰도를 보여주는 평션이고, 그러므로 실행 이벤트이다. 실행 이벤트는 제기될 수 있고 실행 이벤트와 연관되는 정보는 실행 이벤트 큐(159)에 저장될 수 있다. 임의의 적절한 알려진 메커니즘들(예를 들어, 플러그가능한 제3자 라이브러리)이 셸코드를 검출하기 위해, 예를 들어 관계 데이터 분석 모듈(166)에 의해 이용될 수 있다.

[0049] 실시예에서, 이벤트 분석 모듈(162)은 특정 오브젝트 타입들에 의해 사용되는 평션들과 관계되는 실행 이벤트들을 추가로 검출하고 제기할 수 있다. 실시예에서, HTML(Hypertext Markup Language) 오브젝트들에 의해 사용되는 소정 자바스크립트 평션들이 실행 이벤트들로서 제기될 수 있다. 소정 자바스크립트 평션들은 데이터 및/또는 스크립트들을 발생하기 위해 및 '온 더 플라이(on-the-fly)'로 다운로딩되고 있는 현재 파일 내에 이들을 라이트 백하기 위해 HTML 파일 내에 사용될 수 있다. 데이터 및/또는 스크립트들은 길이와 같은 미리 정해진 임계값에 기초하여 평가될 수 있고, 대응 평션은 실행 이벤트로서 제기될 수 있다. 한 예시적인 구현에서, 이 타입의 자바스크립트 평션이 호출되고 기입된 데이터의 길이가 5*1024보다 더 크면, 실행 이벤트가 제기될 수 있다. 그에 따라서, 실행 이벤트와 연관되는 정보는 실행 이벤트 큐(159)에 추가될 수 있다. 게다가, 실행 엔진(156)은 평션을 평가하기 위해, 기입된 데이터가 스크립트를 포함하면 추가적 스크립트를 처리할 수 있다.

[0050] 소정 구현들에서 제기될 수 있는 또 다른 실행 이벤트는 스크립트 태그 없는 평션을 포함한다. 스크립트 태그들은 추출되고 또한 실행될 자바스크립트 섹션들을 정의한다. 스크립트 태그들 내에서, 소정 변수들이, 브라우저 또는 PDF 리더가 코드를 이해하여 이것이 실행될 수 있도록 하기 위해 스크립트가 따라야 하는 포맷을 표시하기 위해 정의된다. 스크립트 태그가 손실되면, 브라우저는 포맷을 이해하지 못할 수 있고 실행은 실패하거나 제대로 동작하지 않을 것이다. 이 이벤트는 제기될 수 있거나, 또는 특정 오브젝트가 이미 비정상 또는 다른 이벤트들을 가질 때만 제기될 수 있다.

[0051] 예시적 실시예에서, 평션 평가 모듈(160)의 이벤트 분석 모듈(162)은 또한, 기타 제기된 실행 이벤트들에 기초하여 힙 스프레이 검출(164)을 실행할 수 있다. 이 분석은, 높은 신뢰도 상관 시그니처가 제기된 이벤트들에 기초하여 검증되자마자 경보를 일으키는 것이 더 효율적일 수 있으므로, 컴파일된 스크립트(300)의 실행 동안 실행될 수 있다. 그러나, 기타 실시예들에서, 힙 스프레이 검출은, 컴파일된 스크립트(300)가 실행을 종료한 후에 실행될 수도 있다. 이들 실시예들에서, 이것은 사후 평가 모듈(170)에서 기타 상관 시그니처 검사들로 실행될 수 있다.

[0052] 힙 스프레이(heap spray)는 말단 호스트의 프로세서상에서 셸코드를 전달하는 네이티브 코드의 실행의 성공을 증가시키는데 널리 사용되는 기술이다. 셸코드들은 버퍼 오버플로를 이용하는 말단 호스트상의 취약점을 부당 활용할 수 있다. 버퍼 오버플로는, 버퍼가 정의되고, 평션이 호출되고, 이후 반환 주소가 버퍼 내로 푸시될 때 생길 수 있다. 전형적으로, 오버플로 기술들은 반환 주소를 바꾸고 또한 새로운 반환 주소에 의해 표시되는 로케이션에 악성 셸코드를 저장하는데 사용되었다. 그러므로, 평션이 반환될 때, 이것은 악성 셸코드의 로케이션에 반환된다. 그러나, 현행의 보안 메커니즘들은 이 기술을 공격자가 성취하기에 어렵게 만든다.

[0053] 힙 스프레이 기술은 공격자가 버퍼 오버플로 기술을 이용하는데 있어서의 무능을 우회하도록 돕는다. 힙 스프레이들은 큰 청크의 NOP(즉, 'No Operation'의 약자) 명령어들을, 셸코드와 함께 메모리에 복사한다. NOP 명령어는 실효적으로 아무것도 하지 않는 어셈블리어 명령어이다. 힙 스프레이에서, 메모리는 NOP 명령어들에 의해 플러싱되고, 공격자의 셸코드는 해당 메모리의 일부에 임베딩된다. 고정 반환 주소가 최초 반환 주소를 오버플로하기 위해 공격자에 의해 이용되지 않을 수 있더라도, NOP 명령어를 가진 메모리의 소정 피스로의 반환의 기회들은 상당할 수 있다. 이는 NOP 명령어들이 큰 범위의 메모리를 소모한다면 특히 그러하다.

[0054] 실시예에서, 이벤트 분석 모듈(162)은 컴파일된 스크립트(300)의 실행 동안 힙 스프레이 행동들을 검출하기 위해 구성된다. 힙 스프레이는, 실행 이벤트 큐(159)에서의 정보가 스트링 연접 평션이 여러 번 호출되었고 또한 결과적 연접된 스트링의 길이가 미리 정해진 임계 길이를 만족하는 것을 표시할 때 검출된다. 예를 들어, STRCAT 자바스크립트 평션이 여러 번 호출되었다면, 및 결과적 스트링 길이가 3*1024보다 더 크다면, 힙 스프레이 실행 이벤트가 제기된다. 변수들은 이 결정을 제어할 수 있다. 예시적 실시예에서, 변수들은 실행 이벤트 큐(159)에 저장되거나 이것에게 링크될 수 있다. 힙 스프레이 실행 이벤트가 검출되면, 적절한 행동들이 취해질 수 있고 컴파일된 스크립트(300)의 실행은 몇몇 실시예들에서 종결될 수 있다.

[0055] 평션 평가 모듈(160)이 평션의 평가를 완료한 후, 제어는 실행 엔진(156)에게 반환될 수 있다. 제어는 컴파일된 스크립트(300)의 관련 평션이 호출되는 때마다 평션 평가 모듈(160)에게 되돌려 넘겨질 수 있다. 일단 컴파일된 스크립트(300)의 모든 평션들이 평가와 실행을 완료했다면, 사후 평가 모듈(170)은 사후 평가 분석을 제공할 수 있다.

[0056] 사후 평가 모듈(170)은 컴파일 이벤트 큐(157)와 실행 이벤트 큐(159)에서의 이벤트 정보에 및 상관 시그니처들(175)에 기초하여 악성 이벤트 상관들(172)을 실행할 수 있다. 실시예에서, 상관 시그니처들은 이벤트 정보의 다양한 조합들을 정의한다. 각각의 조합은 컴파일 및/또는 실행 이벤트들의 해당 특정 조합을 가진 스크립트가 악성이라는 것을 표시하도록 결정되었다. 이벤트 정보는, 예를 들어 이벤트의 발생들의 횟수, 그 외의 이벤트들에 상대적인 이벤트의 발생들의 시간과 로케이션, 및/또는 이벤트를 트리거링한 각각의 평션에 할당되는 가중을 포함할 수 있다. 악성 이벤트 상관들(172)은, 자바스크립트 및 따라서 그것의 대응 오브젝트가 악성인지를 결정하기 위해, 상관 시그니처들(175)에 기초하여 이벤트 큐들(157 및 159)에서의 이벤트 정보를 평가하는 것을 포함한다.

[0057] 실시예에서, 일부 상관 시그니처들은 상관 시그니처가 고정 상관 시그니처일 때 영구적일 수 있다. 예에서, 고정 상관 시그니처는 미리 정해진 임계 횟수만큼 호출되는 포인터 평션들을 포함할 수 있다. 예를 들어, 고정 상관 시그니처는 CHARAT, CHARCODEAT, 및/또는 STRINGFROMCHARCODE 자바스크립트 평션들이 5회보다 더 많이 호출되면 악성인 것을 표시할 수 있다. 또 다른 예에서, 고정 상관 시그니처는 인코딩, 디코딩, 및/또는 계산 평션들과 관계되는 이벤트들을 포함할 수 있다. 예를 들어, 고정 상관 시그니처는, 제1 이벤트가 다량의 데이터를 낳은 인코딩/디코딩 평션(예를 들어, UNESCAPE 자바스크립트 평션)에 대해 제기되었고 또한 또 다른 이벤트가 인코딩/디코딩 이벤트의 5개의 이벤트 내에 발생한 계산 평션(예를 들어, EVAL 자바스크립트 평션)에 대해 제기되었다면 자바스크립트가 악성이라는 것을 표시할 수 있다. 추가적 예에서, 고정 상관 시그니처는, 동적 평션 생성을 위한 실행 이벤트가 교체 평션에 대응하는 실행 이벤트의 소정 거리(예를 들어, 이벤트들의 수) 내에 발생했다면 악성 자바스크립트를 표시할 수 있다. 예를 들어, 고정 상관 시그니처는, 실행 이벤트가 REPLACE 자바스크립트 평션에 대해 제기된 이벤트의 2개의 이벤트 내에 동적 평션 생성에 대해 제기되었다면 자바스크립트가 악성인 것을 제공할 수 있는데, 여기서 스트링에서의 교체된 데이터는 미리 정해진 임계 길이(예를 들어, >512 바이트)를 만족하였다.

[0058] 다양한 관련 평션들, 미리 정해진 임계값들, 및 상관 시그니처들에 대해 본 명세서에서 제공된 특정 예들은 설명의 목적을 위해 의도된 것이고, 제한하기 위한 의도는 아니다. 본 개시의 개념들이 많은 다양한 평션들과 이 평션들의 임계값들에 적용될 수 있다는 것이 분명할 것이다. 유사하게, 본 명세서에서 제공된 개념들은 실행 및/또는 컴파일 이벤트들의 다양한 조합들을 이용함으로써 많은 다양한 상관 시그니처들을 정의하도록 적용될 수 있다. 더욱이, 개념들은 본 명세서에서 특정하게 열거되지 않은 다른 자바스크립트 평션들(또는 다른 스크립트 언어 평션들)에 추가로 적용될 수 있다. 끝으로, 실행 이벤트 및/또는 컴파일 이벤트를 트리거링하는 관련 평션들 및 대응하는 임계값들은 제조자, 최종 사용자, 또는 양쪽에 의해 구성될 수 있다. 유사하게, 상관 시그니처들을 정의하는 이벤트들의 조합들은 제조자, 최종 사용자, 또는 양쪽에 의해 구성될 수 있다.

[0059] 도 3을 참조하면, 흐름도는 스크립트 평가 엔진(150)의 컴파일 시간 휴리스틱 검출 모듈(154)에 의해 적어도 부분적으로 실행될 수 있는 활동들의 흐름 300을 도해한다. 흐름 300은 컴파일러(152)에 의해 자바스크립트(20

0)의 컴파일 동안 실행될 수 있다. 한 예시적 실시예에서, 흐름 300은 자바스크립트(200)에서의 각각의 문(statement)에 대하여 실행될 수 있다.

[0060] 302에서, 컴파일 시간 휴리스틱 검출 모듈(154)은, 대입문(assignment statement)의 우변 값 또는 좌변 변수일 수 있는 적어도 하나의 기준에 기초하여 컴파일 이벤트를 검출하기 위해 자바스크립트(200)와 같은 스크립트에서의 문을 평가한다. 예를 들어 304에서 문이 대입이 아니라고 결정되었다면, 흐름은 종료할 수 있다. 그러나, 문이 변수들을 재대입하려는 시도라면, 306에서, 대입문의 우변 값은 우변 값이 회피 및/또는 난독화 평션으로부터 발생되었는지를 결정하기 위해 평가될 수 있다. 회피 및/또는 난독화 평션은 어느 한 포맷으로부터 또 다른 포맷으로 데이터를 변환하는 인코딩/디코딩 평션(예를 들어, UNESCAPE 자바스크립트 평션)일 수 있다. 또 다른 회피 및/또는 난독화 평션은 표현들 또는 값들에 대한 계산들을 실행하는 계산 평션(예를 들어, EVAL 자바스크립트 평션)일 수 있다. 회피 및/또는 난독화 평션은 308에서 결정된 것처럼, 대입문의 우변에서 발견되면, 컴파일 이벤트는 310에서 컴파일 이벤트와 연관되는 정보를 컴파일 이벤트 큐(157)에 추가함으로써 제기될 수 있다. 컴파일 이벤트와 연관되는 정보는 값이 대입문에 대입되고 있는 변수의 범위에 기초한 것일 수 있다. 이 범위는 변수가 국소적인지(즉, 하나의 평션만에 대한 것) 또는 전역적인지(즉, 스크립트 전체에 걸친 것)를 표시할 수 있다. 덧붙여, 다중 회피 및/또는 난독화 평션이 대입문들에서 발견되면, 다중 컴파일 이벤트가 제기될 수 있다.

[0061] 312에서, 대입문의 좌변이 변수 명이 특이한지를 결정하기 위해 또한 평가될 수 있다. 일반적으로, 변수 명은 그것의 길이에 기초하여 (예를 들어, 미리 정해진 임계값보다 더 큰 것) 또는 그것의 무작위성에 기초하여 특이한 것으로서 식별될 수 있다. 길이와 무작위성 모두는 합법적 프로그래머가 아니라 악성 알고리즘에 의해 생성된 변수 명을 표시할 수 있다. 이들 예들은 제한하도록 의도되지 않았고, 변수 명의 다른 특징들이 또한 악의적 거동을 표시할 수 있다는 것이 분명할 것이다. 변수 명이 314에서 특이한 것으로 결정되면, 컴파일 이벤트는, 316에서 컴파일 이벤트와 연관되는 정보를 컴파일 이벤트 큐(157)에 추가함으로써 제기될 수 있다. 정보는 대입문에서의 변수의 범위에 기초할 수 있다. 범위는 변수가 스크립트에서 국소적인지 전역적인지를 표시할 수 있다. 덧붙여, 다중의 특이한 변수 명이 대입문들에서 발견되면, 다중 컴파일 이벤트가 제기될 수 있다.

[0062] 몇몇 실시예들에서, 또 다른 검사가 실행될 수 있다. 대입문의 우변 값은 우변 값은 이 우변 값이 비정상적으로 긴지를 결정하기 위해 잠재적으로 평가될 수 있다. 우변 값이 비정상적으로 긴 것으로(예를 들어, 미리 정해진 임계 길이보다 더 크다고) 결정되면, 컴파일 이벤트는 이벤트와 연관되는 정보를 컴파일 이벤트 큐에 추가함으로써 제기될 수 있다. 이 정보는 값이 대입문에 대입되고 있는 변수의 범위에 기초할 수 있다. 그러나, 다른 실시예들에서, 이 검사는 컴파일 동안 실행되지 않을 수 있다. 그 대신에, 도 5를 참조하여 본 명세서에서 추가로 기술될 것처럼, 비슷한 평가들이 컴파일된 스크립트의 실행 동안 실행될 수 있다.

[0063] 도 4를 참조하면, 흐름도는 스크립트 평가 엔진(150)의 실행 엔진(156)에 의해 적어도 부분적으로 실행될 수 있는 활동들의 흐름 400을 예시한다. 흐름 400은 이것이 컴파일러(152)로부터 컴파일된 스크립트(300)를 수신할 때 실행될 수 있다.

[0064] 402에서, 실행 엔진(156)은 컴파일러(152)로부터 컴파일된 스크립트(300)(예를 들어, 실행 가능 자바스크립트)를 수신한다. 404에서, 실행 엔진(156)은 컴파일된 스크립트의 실행을 개시한다. 406에서, 호출된 평션이 악성 코드에 대한 분석과 관련되는지에 대한 결정이 이루어진다. 실시예에서, 이 결정은 코드가 평션 내로 후크되는지에 기초하여 이루어질 수 있고, 제어가 평션의 시작에서의 실행 엔진으로부터 없어지게 되는 결과를 낳는다.

[0065] 평션이 관련성 있다면, 408에서, 후크된 코드는, 도 4의 B에 의해 표시된 대로, 평션 평가 모듈(160)에게 제어를 넘길 수 있다. 일단 평션 평가 모듈이 그 평가들을 끝냈다면, 410에서, 실행 엔진(156)은 도 4의 A에 의해 표시된 대로 평션 평가 모듈(160)로부터 제어를 되돌려 수신한다. 일단 실행 엔진(156)이 제어를 되돌려 수신하면, 또는 평션이 406에서 결정된 것처럼 관련성이 없다면(예를 들어, 평션이 코드 후크를 포함하지 않으면), 412에서, 실행 엔진(156)은 컴파일된 스크립트를 실행하기를 계속할 수 있다.

[0066] 414에서, 또 다른 평션이 컴파일된 스크립트에서 호출되는지에 대한 결정이 이루어진다. 또 다른 평션이 호출되면, 흐름은 406에게 되돌아 갈 수 있다. 제어는, 평션이 악성 코드를 검출하는 것과 관련되고 또한 따라서 평션 평가 모듈(160)에 대한 후크를 포함하면, 평션 평가 모듈(160)에게 넘겨질 수 있다. 이 처리는 컴파일된 스크립트(300)에서 호출되는 각각의 관련 평션에 대해 계속될 수 있다. 더 이상의 평션들이 414에서 호출되지 않을 때, 이후 컴파일된 스크립트의 모든 평션들은 실행되었고 그리고 컴파일 및 실행 이벤트들의 사후트 평가 분석은 416에서 실행될 수 있다.

[0067] 도 5에서, 흐름도는 펑션 평가 모듈(160)에 의해 적어도 부분적으로 실행될 수 있는 활동들의 흐름 500을 예시한다. 실시예에서, 흐름 500은, 관련 펑션에서의 후크된 코드가 도 5의 B에 의해 표시된 것처럼 실행 엔지(156)으로부터 펑션 평가 모듈(160)로 제어를 넘길 때 실행될 수 있다.

[0068] 502에서, 펑션 평가 모듈(160)은 컴파일된 스크립트가 실행 엔진(156)에서 실행되고 있음에 따라 컴파일된 스크립트(300)에서 호출되는 관련 펑션(예를 들어, 후크되었던 자바스크립트 펑션)의 제어를 수신한다. 504에서, 피호출 펑션의 실행은 적어도 하나의 기준(예를 들어, 펑션의 타입, 미리 정해진 임계값)에 기초하여 실행 이벤트를 검출하기 위하여 평가될 수 있다. 506에서, 펑션 내로 넘겨진 임의의 파라미터들이 미리 정해진 임계값보다 더 큰 길이를 갖는지가 결정된다. 그와 같은 파라미터들이 발견되면, 실행 이벤트는 파라미터 실행 이벤트와 연관되는 정보를 실행 이벤트 큐(159)에 추가함으로써 516에서 제기될 수 있다.

[0069] 어떤 파라미터들도 임계값을 만족하지 않으면, 510에서, 피호출 펑션이 임의의 회피 및/또는 난독화 기술들을 사용하는지가 결정된다. 예를 들어, 본 명세서에서 전술된 것처럼 회피 및/또는 난독화 기술들에 사용될 수 있는 자바스크립트 펑션들은 하기를 포함하지만 이것들에만 제한되지는 않는다: UNESCAPE, EVAL, STRCAT, STRINGFROMCHARCODE, CHARCODEAT, CHARAT, ARRAY.CONCAT, UNESCAPE, 및 REPLACE. 피호출 펑션이 회피 또는 난독화 펑션이면 그리고 이 펑션이 해당 특정 펑션에 대한 기타 요구된 기준(예를 들어, 길이들, 사이즈들, 기타 등등과 같은 미리 정해진 임계값들)을 만족시키면, 이후 이벤트가, 실행 이벤트 큐(159)에 회피 및/또는 난독화 실행 이벤트와 관계되는 정보를 추가함으로써 516에서 제기될 수 있다.

[0070] 피호출 펑션이 회피 또는 난독화 펑션이 아니라면, 또는 피호출 펑션이 회피 또는 난독화 펑션이지만 그 제각기의 다른 기준들(예를 들어, 임계값을 만족하는 것)을 만족하지 않으면, 512에서, 펑션이 높은 신뢰도의 악의적 거동 펑션인지가 결정된다. 예를 들어, 높은 신뢰도의 악의적 거동을 가진 자바스크립트 펑션은, 본 명세서에서 전술된 것처럼, 동적 펑션 생성 및 셸코드 검출을 낳는 펑션들을 포함하지만 이것에만 제한되지는 않는다. 피호출 펑션이 높은 신뢰도의 악의적 거동을 가진 펑션이라면, 실행 이벤트는 높은 신뢰도의 실행 이벤트와 관계되는 정보를 실행 이벤트 큐(159)에 추가함으로써 516에서 제기될 수 있다.

[0071] 피호출 펑션이 높은 신뢰도의 펑션이 아니라면, 514에서, 펑션이 HTML과 같은 특정 오브젝트 타입에 대해 비정상적으로 이용될 수 있는 것인지가 결정된다. 예를 들어, HTML을 가진 악성 기술로서 이용될 수 있는 소정 자바스크립트 펑션들은 다량의 데이터 및/또는 스크립트들을 발생시키기 위해 및 다량의 데이터 및 스크립트 데이터를 '온 더 플라이'로 다운로딩되고 있는 현재 파일 내로 되돌려 기입하기 위해 HTML 파일 내에서 이용되는 펑션들을 포함한다. 피호출 펑션이 HTML과 같은 특정 오브젝트 타입을 가진 악성 기술로서 비정상적으로 이용될 수 있는 펑션이라면 그리고 이 펑션이 다른 임의의 요구된 기준을 만족하면, 실행 이벤트는 특정 오브젝트 타입 실행 이벤트와 관계되는 정보를 실행 이벤트 큐(159)에 추가함으로써 516에서 제기될 수 있다.

[0072] 일단 평가가 특정 피호출 펑션에 대해 완료되면, 518에서, 실행 엔진(156)에게 제어를 되돌려 넘겨주기 전에, 높은 신뢰도의 상관 시그니처의 존재를 결정하기 위한 검증이 실행될 수 있다. 예를 들어, 이 검증은, 이벤트 큐들(157 및 159)에서의 이벤트 정보가 힙 스프레이에 대한 상관 시그니처와 같은 높은 신뢰도의 상관 시그니처와 일치하는지를 결정할 수 있다. 높은 신뢰도의 상관 시그니처가 520에서 존재한다고 결정되면, 522에서 임의의 적절한 행동이 취해질 수 있다. 예를 들어, 스크립트에 대응하는 오브젝트가 차단되거나 격리된다. 또한 경보가 보내질 수 있고, 악성 코드 검출의 로그가 기록될 수 있고, 기타 등등과 같이 된다. 일단 흐름 500이 실행 종료되면, 도 5의 A에 의해 표시된 것처럼, 제어는 실행 엔진(156)에게 되돌려 넘겨질 수 있다.

[0073] 도 6에서, 흐름도는 사후 평가 모듈(170)에 의해 적어도 부분적으로 실행될 수 있는 활동들의 흐름 600을 예시한다. 실시예에서, 흐름 600은 컴파일된 스크립트(300)에서의 모든 펑션들이 실행 엔진(156)에 의해 실행되었을 때 실행될 수 있다.

[0074] 602에서, 이벤트 큐들(157 및 159)로부터의 이벤트 정보는 스크립트가 악성 코드를 포함하는지를 결정하기 위해 상관될 수 있다. 이는 컴파일 이벤트 큐(157) 및/또는 실행 이벤트 큐(159)로부터의 이벤트 정보에 기초하여(예를 들어, 상관 시그니처들(175)로부터의) 고정 상관 시그니처를 검증함으로써 달성될 수 있다. 예를 들어, 하나의 고정 상관 시그니처는 미리 정해진 임계 횟수만큼 호출된 포인터 펑션들을 포함할 수 있다. 또 다른 고정 상관 시그니처는 미리 정해진 계산 펑션들의 로케이션 내의 인코딩 및/또는 디코딩 펑션들을 포함할 수 있다. 또 다른 고정 상관 시그니처는 교체 펑션의 시간과 거리 내에서 발생하는 동적 펑션 생성을 포함할 수 있다. 고정 상관 시그니처가 604에서 검증되면, 610에서 임의의 적절한 행동이 취해질 수 있다(예를 들어, 오브젝트를 차단하고, 오브젝트를 격리하고, 경보를 보내고, 악성 코드 검출을 로그하고, 기타 등등).

[0075] 그러나, 고정 상관 시그니처가 604에서 검증되지 않으면, 606에서 (예를 들어, 상관 시그니처들(175)로부터의) 수정 가능 상관 시그니처가 검증될 수 있다. 수정 가능 상관 시그니처들은 사용자에 의해 구성 가능할 수 있고 및/또는 디폴트 상관 시그니처들을 포함할 수 있다. 수정 가능 상관 시그니처가 608에서 검증되면, 610에서 임의의 적절한 행동이 취해질 수 있다(예를 들어, 오브젝트를 차단하고, 오브젝트를 격리하고, 경보를 보내고, 악성 코드 검출을 로그하고, 기타 등등). 그러나, 수정 가능 상관 시그니처가 608에서 검증되지 않으면, 흐름 600은 끝날 수 있고 또한 컴파일된 스크립트는 악성이 아닌 것으로 간주될 수 있다.

[0076] 상관 시그니처들(고정된 상관 시그니처들과 수정 가능 상관 시그니처들 모두)은 이벤트들의 특정 조합들, 이벤트들의 발생들의 횟수, 그 외의 이벤트들에 상대적인 발생들의 시간 및/또는 로케이션, 및/또는 스크립트가 악성인지를 결정하기 위한 평션들의 가중을 포함할 수 있다.

[0077] 도 7-8은 본 명세서에 개시된 실시예들에 따라서 이용될 수 있는 예시적 컴퓨터 아키텍처들의 블록도들이다. 프로세서들 및 컴퓨팅 시스템들에 대해 본 분야에 알려진 기타 컴퓨터 아키텍처 설계들도 또한 이용될 수 있다. 일반적으로, 본 명세서에 개시된 실시예들에 대한 적절한 컴퓨터 아키텍처들은 도 7-8에 예시된 구성들을 포함할 수 있지만 여기에만 한정되지는 않는다.

[0078] 도 7은 실시예에 따른 프로세서의 예시적 도해이다. 프로세서(700)는 보안 장치(130)의 프로세서(180)의 한 예시적 실시예이다.

[0079] 도 7은 실시예에 따라 프로세서(700)에 결합되는 메모리(702)를 또한 도해한다. 메모리(702)는 보안 장치(130)의 메모리 성분(190)의 한 예시적 실시예이다. 메모리(702)는 공지되거나 그렇지 않으면 통상의 기술자가 이용 가능한 (메모리 위계 구조(memory hierarchy)의 다양한 계층들을 포함하는) 매우 다양한 메모리들 중 임의의 것일 수 있다. 그와 같은 메모리 성분들은 RAM(random access memory), ROM(read only memory), FPGA(field programmable gate array)의 로직 블록들, EPROM(erasable programmable read only memory), 및 EEPROM(electrically erasable programmable read only memory)을 포함할 수 있지만, 이것들에만 한정되지는 않는다.

[0080] 프로세서(700)는 본 명세서에 상술된 악성 스크립트 언어 코드를 검출하기 위해 동작들과 연관되는 임의 타입의 명령어들을 실행할 수 있다. 일반적으로, 프로세서(700)는 하나의 상태 또는 실체(thing)로부터 다른 상태 또는 실체로 성분 또는 아티클(예로, 데이터)을 변환할 수 있다.

[0081] 프로세서(700)에 의해 실행될 하나 이상의 명령어일 수 있는 코드(704)는 메모리(702)에 저장될 수 있다. 코드(704)는, 소프트웨어, 하드웨어, 펌웨어, 또는 이것들의 임의의 적절한 조합으로 저장되거나, 또는 적절한 경우에 그리고 특정 필요에 기초하여 임의의 다른 내부 또는 외부 컴포넌트, 장치, 요소, 또는 오브젝트에 저장될 수 있는 다양한 모듈들(예를 들어, 표준 검출 모듈(140), 전처리 모듈(145), 스크립트 평가 엔진(150), 펑션 평가 모듈(160), 사후 평가 모듈(170), 기타 등등)의 명령어들을 포함할 수 있다. 한 예에서, 프로세서(700)는 코드(704)에 의해 표시되는 명령어들의 프로그램 시퀀스를 따를 수 있다. 각각의 명령어는 프론트 엔드 로직(706)에 진입하고 또한 하나 이상의 디코더들(708)에 의해 처리된다. 디코더는 미리 정의된 포맷으로 고정 폭 마이크로 연산과 같은 마이크로 연산을 그 출력으로서 발생할 수 있거나, 또는 기타 명령어들, 마이크로명령어들(microinstructions), 또는 최초 코드 명령어를 반영하는 제어 신호들을 생성할 수 있다. 프론트 엔드 로직(706)은 또한, 일반적으로 리소스들을 할당하고 또한 실행을 위한 명령어에 대응하는 동작을 큐잉하는 레지스터 리네이밍 로직(710) 및 스케줄링 로직(712)을 포함할 수 있다.

[0082] 프로세서(700)는 또한 한 세트의 실행 유닛들(716_1 내지 716_m)을 갖는 실행 로직(714)을 포함할 수 있다. 몇몇 실시예들은 특정 펑션들 또는 펑션들의 세트들에 전용인 다수의 실행 유닛을 포함할 수 있다. 기타 실시예들은 특정 펑션을 실행할 수 있는 단 하나의 실행 유닛 또는 하나의 실행 유닛을 포함할 수 있다. 실행 로직(714)은 코드 명령어들에 의해 지정되는 동작들을 실행한다.

[0083] 코드 명령어들에 의해 지정된 동작들의 실행을 완료한 후에, 백 엔드 로직(718)은 코드(704)의 명령어들을 리타이어할 수 있다. 일 실시예에서, 프로세서(700)는 비순차적 실행을 허용하지만 명령어들의 순차적 리타이어먼트를 요구한다. 리타이어먼트 로직(720)은 다양한 알려진 폼들(예를 들어, 리오더 버퍼들 또는 그와 유사한 것)을 취할 수 있다. 이러한 방식으로, 프로세서(700)는, 적어도 디코더에 의해 발생된 출력, 레지스터 리네이밍 로직(710)에 의해 활용되는 하드웨어 레지스터들과 테이블들, 및 실행 로직(714)에 의해 수정되는 임의의 레지스터들(도시 안됨)의 관점에서, 코드(704)의 실행 동안 변환된다.

[0084] 도 7에 도시되지는 않았지만, 처리 성분은 프로세서(700)와 함께 칩상의 다른 성분들을 포함할 수 있다. 예를

들어, 처리 성분은 프로세서(700)와 함께 메모리 제어 로직을 포함할 수 있다. 처리 성분은 I/O 제어 로직을 포함할 수 있고 및/또는 메모리 제어 로직과 통합되는 I/O 제어 로직을 포함할 수 있다. 처리 성분은 하나 이상의 캐시들을 또한 포함할 수 있다. 몇몇 실시예들에서, 비휘발성 메모리(플래시 메모리 또는 퓨즈들과 같은 것)는 또한 프로세서(700)와 함께 칩상에 포함될 수 있다.

[0085] 도 8을 이제 참조하면, 도 8은 실시예에 따라 포인트 투 포인트(PtP) 구성으로 배열되는 컴퓨팅 시스템(800)을 도해한다. 특히, 도 8은 프로세서들, 메모리, 및 입력/출력 장치들이 다수의 포인트 투 포인트 인터페이스에 의해 상호 접속되는 시스템을 도해한다. 일반적으로, 통신 시스템(100)의 컴퓨팅 시스템들 중 하나 이상은 컴퓨팅 시스템(800)과 동일한 또는 비슷한 방식으로 구성될 수 있다. 예를 들어, 본 명세서에서 보여지고 기술되는 보안 장치(130) 및 말단 호스트들(120)은 예시적 컴퓨팅 시스템(800)과 동일하거나 비슷한 방식으로 구성될 수 있다.

[0086] 프로세서들(870 및 880)은 각각이 또한 메모리 성분들(832 및 834)과 통신하기 위해 통합된 메모리 컨트롤러 로직(MC)(872 및 882)을 포함할 수 있다. 대안 실시예들에서, 메모리 컨트롤러 로직(872 및 882)은 프로세서들(870 및 880)과 별개인 이산 로직일 수 있다. 본 명세서에서 요약된 것처럼, 메모리 성분들(832 및/또는 834)은 악성 스크립트 언어 코드를 검출하는 것과 연관되는 동작들을 달성하는데 있어서 프로세서들(870 및 880)에 의해 이용될 다양한 데이터를 저장할 수 있다.

[0087] 프로세서들(870 및 880)은 도 7의 프로세서(700) 및 도 1의 프로세서(180)를 참조하여 논의된 것들과 같은 임의 타입의 프로세서일 수 있다. 프로세서들(870, 880)은 제각기 PtP 인터페이스 회로들(878 및 888)을 이용하는 PtP 인터페이스(850)를 통해 데이터를 교환할 수 있다. 프로세서들(870 및 880)은 각각 PtP 인터페이스 회로들(876, 886, 894, 및 898)을 이용하는 개개의 PtP 인터페이스들(852 및 854)을 통해 칩셋(890)과 데이터를 교환할 수 있다. 칩 셋(890)은 또한, PtP 인터페이스 회로일 수 있는 인터페이스 회로(892)를 이용하여, 고성능 그래픽 인터페이스(839)를 통해 고성능 그래픽 회로(838)와 데이터를 교환할 수 있다. 대안 실시예들에서, 도 8에 예시된 임의의 또는 모든 PtP 링크들은 PtP 링크가 아니라 멀티 드롭 버스로서 구현될 수 있다.

[0088] 칩셋(890)은 인터페이스 회로(896)를 통해 버스(820)와 통신 상태에 있을 수 있다. 버스(820)는 버스 브리지(818) 및 I/O 장치들(816)과 같이 자신상에서 통신하는 하나 이상의 장치들을 가질 수 있다. 버스(810)를 통해, 버스 브리지(818)는 키보드/마우스(812) (또는 터치스크린, 트랙볼, 기타 등등과 같은 다른 입력 장치들), 통신 장치들(826)(모뎀들, 네트워크 인터페이스 장치들, 또는 컴퓨터 네트워크(860)를 통해 통신할 수 있는 다른 유형의 통신 장치들과 같은 것), 오디오 I/O 장치들(814), 및/또는 데이터 스토리지 장치(828)와 같은 기타 장치들과 통신 상태에 있을 수 있다. 데이터 스토리지 장치(828)는 프로세서들(870 및/또는 880)에 의해 실행될 수 있는 코드(830)를 저장할 수 있다. 대안 실시예들에서, 버스 아키텍처들 중 임의의 부분들은 하나 이상의 PtP 링크들로 구현될 수 있다.

[0089] 도 8에 도시된 컴퓨터 시스템은 본 명세서에서 논의되는 다양한 실시예들을 구현하기 위해 이용될 수 있는 컴퓨팅 시스템의 실시예의 구성도이다. 본 명세서에 제공된 것처럼, 도 8에 묘사된 시스템의 다양한 컴포넌트들은 SoC 아키텍처에서 또는 악성 스크립트 언어 코드를 검출할 수 있는 임의의 다른 적절한 구성에서 조합될 수 있다는 것을 이해할 것이다.

[0090] 본 명세서에서 요약되는 통신 시스템(100)의 검출 기능들은 하나 이상의 유형 매체에 인코딩되는 로직(예를 들어, ASIC에 제공되는 임베디드 로직, DSP 명령어들, 프로세서(예로, 프로세서(180)) 또는 다른 유사한 머신, 기타 등등에 의해 실행될 소프트웨어(오브젝트 코드 및 소스 코드를 잠재적으로 포함))에 의해 구현될 수 있다. 유형의 매체는 적어도 몇몇 실시예들에서 비일시적일 수 있다. 이러한 경우들의 일부에서, 메모리(예를 들어, 메모리 성분(190))는 본 명세서에서 기술되는 동작들에 사용되는 데이터를 저장할 수 있다. 이는 본 명세서에 기술되는 활동들을 수행하기 위하여 실행되는 소프트웨어, 로직, 코드, 또는 프로세서 명령어들을 저장할 수 있는 메모리를 포함한다. 실시예에서, 유형의 매체는 보안 장치(130)에 제공될 수 있다.

[0091] 덧붙여, 통신 시스템(100)(예를 들어, 실행 이벤트 정보, 컴파일 이벤트 정보, 상관 시그니처들, 기타 등등)에서 추적되고, 보내지고, 수신되고, 또는 저장되는 정보는 특정 필요들과 구현들에 기초하여, 그 모두가 임의의 적절한 타임프레임에 참조될 수 있는, 임의의 데이터베이스, 레지스터, 테이블, 캐시, 큐, 제어 리스트, 또는 스토리지 구조에 제공될 수 있다. 본 명세서에 논의된 메모리 아이템들 중 임의의 것은 "메모리 성분"이라는 광범위 용어 내에 포괄되는 것으로 해석되어야 한다. 유사하게, 본 명세서에 기술된 잠재적 처리 성분들, 모듈들, 및 머신들 중 임의의 것은 '프로세서'라는 광범위 용어 내에 포괄되는 것으로 해석되어야 한다.

[0092] 본 명세서에 제공되는 수많은 예들에 대해 상호 작용들이 다중 네트워크 성분, 컴퓨팅 시스템들, 모듈들, 및/또는 기타 컴포넌트들의 관점에서 기술될 수 있다는 것을 주목하라. 그러나, 이는 단지 명확성 및 예시적 목적으로만 행해진 것이다. 시스템이 임의의 적절한 방식으로 통합될 수 있다는 것을 알아야 한다. 비슷한 설계 대안들을 따라, 도 1의 예시된 모듈들, 노드들, 성분들, 및 기타 컴포넌트들 중 임의의 것은 다양한 가능한 구성들로 조합되고 분할될 수 있는데, 이 모든 것들은 본 명세서의 광범위한 범위 내에 명확하게 속하는 것이다. 도 1의 시스템들(및 그 교시들)이 용이하게 스케일링 가능하고 또한 대량의 컴포넌트뿐만 아니라 보다 복잡하고/정교한 배열들 및 구성들을 수용할 수 있다는 것을 이해해야 한다. 따라서, 제공되는 예시들은 범위를 제한하거나 또는 무수히 많은 다른 아키텍처들에 잠재적으로 적용되는 바와 같은 시스템(100)의 넓은 교시 범위를 금지하는 것이 아니다.

[0093] 앞선 도면들을 참조하여 설명된 동작들이 시스템에 의해 또는 시스템 내에서 실행될 수 있는 가능한 시나리오들 중 단지 일부를 예시하고 있다는 것을 주목하는 것이 중요하다. 이러한 동작들의 일부는 적절한 경우 삭제되거나 제거될 수 있고, 또는 이들 동작들은 논의된 개념의 범위에서 벗어나지 않고 상당하게 수정되거나 변화될 수 있다. 추가로, 이들 동작들의 타이밍은 상당히 변경될 수 있으면서도 본 개시에서 교시된 결과들을 달성할 수 있다. 선행 동작 흐름들은 예시와 논의의 목적을 위해 제공되었다. 논의된 개념의 교시에서 벗어나지 않고 임의의 적합한 배열들, 연대순들, 구성들, 및 타이밍 메커니즘들이 제공될 수 있는 시스템에 의해 상당한 융통성이 제공된다.

[0094] 다른 예들에서, 본 명세서에서 기술되는 평션들은 (예를 들어, 바이러스 차단 해결책의 일부로서의) 독점적 성분을 수반할 수 있는데, 이것은 이들 식별된 성분들에서 제공되거나(또는 이것들에 인접할 수 있거나), 또는 임의의 다른 네트워크 성분 또는 다른 장치에 제공되거나, 또는 (예를 들어, 방화벽과 연계하여) 보완 솔루션으로서 제공되거나, 또는 네트워크에서의 어딘가에 프로비저닝될 수 있다. 게다가, 본 명세서에서 기술되는 평션들은 임의의 적절한 방식으로 통합될 수 있다.

[0095] 하기 예들은 본 명세서에 따른 실시예들과 관련된다. 하나 이상의 실시예들은 네트워크 환경에서의 스크립트에서 악성 코드를 검출하는 방법을 제공할 수 있다. 이 방법은 하기를 포함할 수 있다: 컴파일된 스크립트의 실행을 개시하는 단계; 컴파일된 스크립트에서 호출되는 평션을 평가하는 단계; 적어도 제1 기준에 기초하여 실행 이벤트를 검출하는 단계; 실행 이벤트 큐에 실행 이벤트와 연관되는 정보를 저장하는 단계; 및 실행 이벤트 큐에서의 적어도 하나의 실행 이벤트와 연관되는 정보에 기초하여 상관 시그니처를 검증하는 단계.

[0096] 실시예의 예는 하기를 추가로 포함할 수 있다: 컴파일러에 의한 스크립트의 컴파일 동안 스크립트의 대입문을 평가하는 단계 - 컴파일된 스크립트는 스크립트로부터 발생됨-; 적어도 제2 기준에 기초하여 컴파일 이벤트를 검출하는 단계; 및 컴파일 이벤트 큐에 컴파일 이벤트와 연관되는 정보를 저장하는 단계.

[0097] 실시예의 예에서, 상관 시그니처를 검증하는 단계는 컴파일 이벤트 큐에서의 컴파일 이벤트와 연관되는 정보에 적어도 부분적으로 기초한다.

[0098] 실시예의 예에서, 컴파일 이벤트 큐 및 실행 이벤트 큐는 통합된다.

[0099] 실시예의 예에서, 검증하는 단계는 컴파일된 스크립트가 실행되고 있을 때 실행된다.

[0100] 실시예의 예에서, 검증하는 단계는 컴파일된 스크립트가 실행을 완료했을 때 실행된다.

[0101] 실시예의 예에서, 상관 시그니처는 사용자에 의해 구성가능하다.

[0102] 실시예의 예에서, 하나 이상의 파라미터들이 실행동안 평션에게 넘겨지고, 제1 기준은 파라미터들 중 임의의 하나의 미리 정해진 임계값 길이에 기초한다.

[0103] 실시예의 예에서, 평션은 데이터를 인코딩하거나 디코딩하고, 제1 기준은 평션으로부터 귀결되는 스트링의 미리 정해진 임계값 길이에 기초한다.

[0104] 실시예의 예에서, 평션은 데이터를 연접시키고, 제1 기준은 연접된 데이터로부터 귀결되는 스트링의 미리 정해진 임계값 길이에 기초한다.

[0105] 실시예의 예는 하기를 추가로 포함할 수 있다: 평션이 미리 정해진 관련 평션이라고 결정한 것에 응답하여, 평션이 호출될 때 실행 엔진으로부터 평션 평가 모듈로 제어를 넘기는 단계; 및 평션이 실행을 완료했을 때 평션 평가 모듈로부터 실행 엔진으로 제어를 넘기는 단계.

[0106] 실시예의 예에서, 정보는 실행 이벤트의 식별, 실행 이벤트가 검출된 횟수, 미리 정해진 이벤트들의 시퀀스의

식별, 및 실행 이벤트와 하나 이상의 다른 실행 이벤트들 사이의 거리 중 하나 이상을 포함한다.

[0107] 실시예의 예에서, 상관 시그니처를 검증하는 단계는 부분적으로 평션에 할당된 가중에 기초하는데, 여기서 가중은 평션의 상대적 중요성을 나타낸다.

[0108] 실시예의 예에서, 제2 기준은 대입문의 우변 값을 평가하는 것을 포함한다.

[0109] 실시예의 예에서, 제2 기준은 대입문의 좌변 변수 명을 평가하는 것을 포함한다.

[0110] 실시예의 예에서, 평션은 하나 이상의 미리 정해진 관련 평션들 중 하나이다.

[0111] 하나 이상의 실시예들은 네트워크 환경의 스크립트에서 악성 코드를 검출하기 위한 장치를 제공할 수 있다. 장치는 하기를 포함할 수 있다: 하나 이상의 프로세서들; 컴파일된 스크립트의 실행을 개시하기 위해 하나 이상의 프로세서들 중 적어도 하나상에서 실행하도록 구성되는 실행 엔진; 및 컴파일된 스크립트에서 호출되는 평션을 평가하고; 적어도 제1 기준에 기초하여 실행 이벤트를 검출하고; 및 실행 이벤트 큐에 실행 이벤트와 연관되는 정보를 저장하기 위해 프로세서들 중 적어도 하나상에서 실행되도록 구성되는 평션 평가 모듈 - 상관 시그니처가 실행 이벤트 큐에서의 적어도 하나의 실행 이벤트와 연관되는 정보에 기초하여 검증됨-.

[0112] 실시예의 예에서, 평션 평가 모듈은 컴파일된 스크립트의 실행이 끝나기 전에 상관 시그니처를 검증한다.

[0113] 실시예의 예는 하기를 추가로 포함할 수 있다: 컴파일된 스크립트의 실행이 끝난 후에 상관 시그니처를 검증하기 위해 프로세서들 중 적어도 하나상에서 실행되도록 구성되는 사후 평가 모듈.

[0114] 실시예의 예는 하기를 추가로 포함할 수 있다: 컴파일러에 의한 스크립트의 컴파일 동안 스크립트의 대입문을 평가하고 - 컴파일된 스크립트는 스크립트로부터 발생됨-; 적어도 제2 기준에 기초하여 컴파일 이벤트를 검출하고; 및 컴파일 이벤트와 연관되는 정보를 컴파일 이벤트 큐에 저장하기 위해 프로세서들 중 적어도 하나상에서 실행되도록 구성되는 컴파일 시간 휴리스틱 검출 모듈.

[0115] 실시예의 예에서, 상관 시그니처는 컴파일 이벤트 큐에서의 컴파일 이벤트와 연관되는 정보에 부분적으로 기초하여 검증된다.

[0116] 실시예의 예에서, 컴파일 이벤트 큐 및 실행 이벤트 큐는 통합된다.

[0117] 실시예의 예에서, 상관 시그니처는 사용자에 의해 구성 가능하다.

[0118] 실시예의 예에서, 실행 엔진은 파라미터를 평션 평가 모듈에게 넘겨 주기 위해 추가로 구성될 수 있고, 제1 기준은 파라미터의 미리 정해진 임계 길이에 기초한다.

[0119] 실시예의 예에서, 평션은 데이터를 인코딩하거나 디코딩하고, 제1 기준은 평션으로부터 귀결되는 스트링의 미리 정해진 임계 길이에 기초한다.

[0120] 실시예의 예에서, 평션은 데이터를 연접시키고, 제1 기준은 연접된 데이터로부터 귀결되는 스트링의 미리 정해진 임계 길이에 기초한다.

[0121] 실시예의 예에서, 실행 엔진은, 평션이 미리 정해진 관련 평션이라고 결정한 것에 응답하여, 평션이 호출될 때 평션 평가 모듈에게 제어를 넘기도록 추가로 구성될 수 있고; 및 평션 평가 모듈은 평션이 실행을 끝낼 때 제어를 실행 엔진에게 넘기도록 추가로 구성될 수 있다.

[0122] 실시예의 예에서, 정보는 실행 이벤트의 식별, 실행 이벤트가 검출된 횟수, 미리 정해진 이벤트들의 시퀀스의 식별, 및 실행 이벤트와 하나 이상의 다른 실행 이벤트들 사이의 거리 중 하나 이상을 포함한다.

[0123] 실시예의 예에서, 상관 시그니처는 평션에 대입된 가중에 부분적으로 기초하여 검증되며, 여기서 가중은 평션의 상대적 중요성을 나타낸다.

[0124] 실시예의 예에서, 제2 기준은 대입문의 우변 값을 평가하는 것을 포함한다.

[0125] 실시예의 예에서, 제2 기준은 대입문의 좌변 변수 명을 평가하는 것을 포함한다.

[0126] 실시예의 예에서, 평션은 하나 이상의 미리 정해진 관련 평션들 중 하나이다.

[0127] 하나 이상의 실시예들은 네트워크 환경에서 스크립트에서 악성 코드를 검출하기 위해 그 상에 저장된 명령어들을 갖는 적어도 하나의 기계 접근 가능 저장 매체를 제공할 수 있다. 명령어들은 프로세서에 의해 실행될 때 프로세서로 하여금: 컴파일된 스크립트의 실행을 개시하고; 컴파일된 스크립트에서 호출되는 평션을 평가하고;

적어도 제1 기준에 기초하여 실행 이벤트를 검출하고; 실행 이벤트 큐에 실행 이벤트와 연관되는 정보를 저장하고; 및 실행 이벤트 큐에서의 적어도 하나의 실행 이벤트와 연관되는 정보에 기초하여 상관 시그니처를 검증하도록 야기한다.

[0128] 실시예의 예는 프로세서에 의해 실행될 때 프로세서로 하여금: 컴파일러에 의한 스크립트의 컴파일 동안 스크립트의 대입문을 평가하고 - 컴파일된 스크립트는 스크립트로부터 발생됨-; 적어도 제2 기준에 기초하여 컴파일 이벤트를 검출하고; 및 컴파일 이벤트 큐에 컴파일 이벤트와 연관되는 정보를 저장하도록 야기하는 명령어들을 추가로 포함할 수 있다.

[0129] 실시예의 예는 프로세서에 의해 실행될 때 프로세서로 하여금: 컴파일 이벤트 큐에서의 컴파일 이벤트와 연관되는 정보에 부분적으로 기초하여 상관 시그니처를 검증하도록 야기하는 명령어들을 추가로 포함할 수 있다.

[0130] 실시예의 예에서, 컴파일 이벤트 큐와 실행 이벤트 큐는 통합된다.

[0131] 실시예의 예는 프로세서에 의해 실행될 때 프로세서로 하여금: 컴파일된 스크립트의 실행이 끝나기 전에 상관 시그니처를 검증하도록 야기하는 명령어들을 추가로 포함할 수 있다.

[0132] 실시예의 예는 프로세서에 의해 실행될 때 프로세서로 하여금: 컴파일된 스크립트의 실행이 끝난 후에 상관 시그니처를 검증하도록 야기하는 명령어들을 추가로 포함할 수 있다.

[0133] 실시예의 예는 프로세서에 의해 실행될 때 프로세서로 하여금: 상관 시그니처가 사용자에 의해 구성되는 것을 허용하도록 야기하는 명령어들을 추가로 포함할 수 있다.

[0134] 실시예의 예는 프로세서에 의해 실행될 때 프로세서로 하여금: 실행 동안 파라미터를 펑션에게 넘기도록 야기하는 명령어들을 추가로 포함할 수 있고, 제1 기준은 파라미터의 미리 정해진 임계 길이에 기초한다.

[0135] 실시예의 예에서, 펑션은 데이터를 인코딩하거나 디코딩하고, 제1 기준은 펑션으로부터 귀결되는 스트링의 미리 정해진 임계 길이에 기초한다.

[0136] 실시예의 예에서, 펑션은 데이터를 연접시키고, 제1 기준은 연접된 데이터로부터 귀결되는 스트링의 미리 정해진 임계 길이에 기초한다.

[0137] 실시예의 예는 프로세서에 의해 실행될 때 프로세서로 하여금: 펑션이 미리 정해진 관련 펑션인 것을 결정한 것에 응답하여, 펑션이 호출될 때 실행 엔진으로부터 펑선 평가 모듈로 제어를 넘기고; 및 펑션이 실행을 끝낼 때 펑션 평가 모듈로부터 실행 엔진으로 제어를 넘기도록 야기하는 명령어들을 추가로 포함할 수 있다.

[0138] 실시예의 예에서, 정보는 실행 이벤트의 식별, 실행 이벤트가 검출된 횟수, 미리 정해진 이벤트들의 시퀀스의 식별, 및 실행 이벤트와 하나 이상의 다른 실행 이벤트들 사이의 거리 중 하나 이상을 포함한다.

[0139] 실시예의 예에서, 상관 시그니처는 펑션에 대입되는 가중에 부분적으로 기초하여 검증되며, 여기서 가중은 펑션의 상대적 중요성을 나타낸다.

[0140] 실시예의 예는 프로세서에 의해 실행될 때 프로세서로 하여금: 제2 기준이 만족되었는지를 결정하기 위해 대입문의 우변 값을 평가하도록 야기하는 명령어들을 추가로 포함할 수 있다.

[0141] 실시예의 예는 프로세서에 의해 실행될 때 프로세서로 하여금: 제2 기준이 만족되었는지를 결정하기 위해 대입문의 좌변 변수 명을 평가하도록 야기하는 명령어들을 추가로 포함할 수 있다.

[0142] 실시예의 예에서, 펑션은 하나 이상의 미리 정해진 관련 펑션들 중 하나이다.

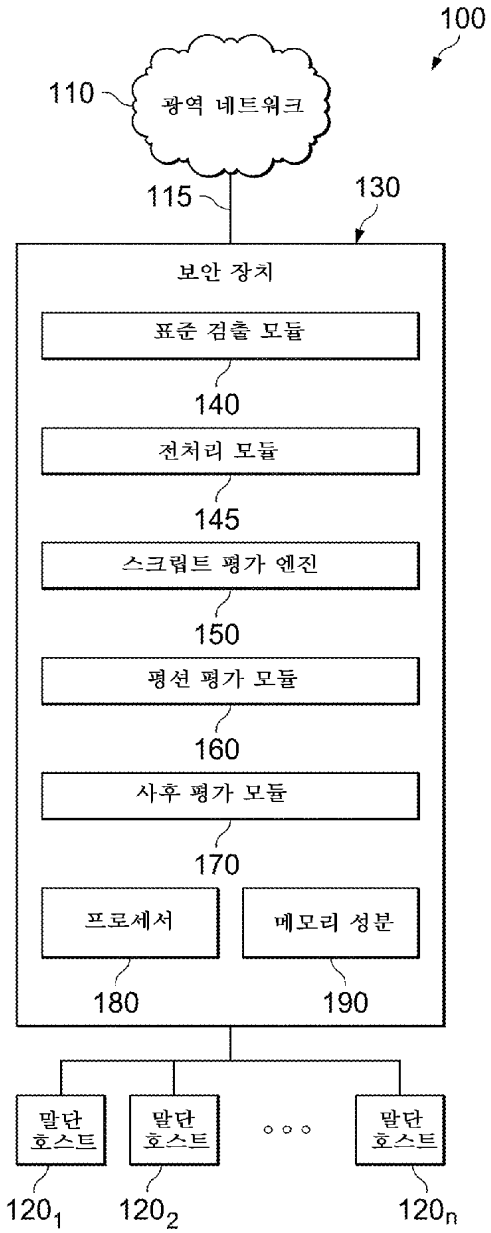
[0143] 실시예의 예에서, 펑션의 평가는 컴파일된 스크립트에서 호출되는 펑션이 미리 정해진 관련 펑션인지를 결정한 것에 응답한 것이다.

[0144] 네트워크 환경에서 스크립트에서 악성 코드를 검출하기 위한 예시적인 구현은 컴파일된 스크립트의 실행을 개시하기 위한 수단; 컴파일된 스크립트에서 호출되는 펑션을 평가하기 위한 수단; 적어도 제1 기준에 기초하여 실행 이벤트를 검출하기 위한 수단; 실행 이벤트와 연관되는 정보를 실행 이벤트 큐에 저장하기 위한 수단; 및 실행 이벤트 큐에서의 적어도 하나의 실행 이벤트와 연관되는 정보에 기초하여 상관 시그니처를 검증하기 위한 수단을 포함할 수 있다. 이 구현은 또한 컴파일러에 의한 스크립트의 컴파일 동안 스크립트의 대입문을 평가하기 위한 수단 - 컴파일된 스크립트는 스크립트에서 발생됨 -; 적어도 제2 기준에 기초하여 컴파일 이벤트를 검출하기 위한 수단; 및 컴파일 이벤트 큐에 컴파일 이벤트와 연관되는 정보를 저장하기 위해 수단을 더 포함할 수 있다. 상관 시그니처를 검증하기 위한 수단은 컴파일된 스크립트의 실행이 끝나기 전에 및/또는 끝난 후에 실행

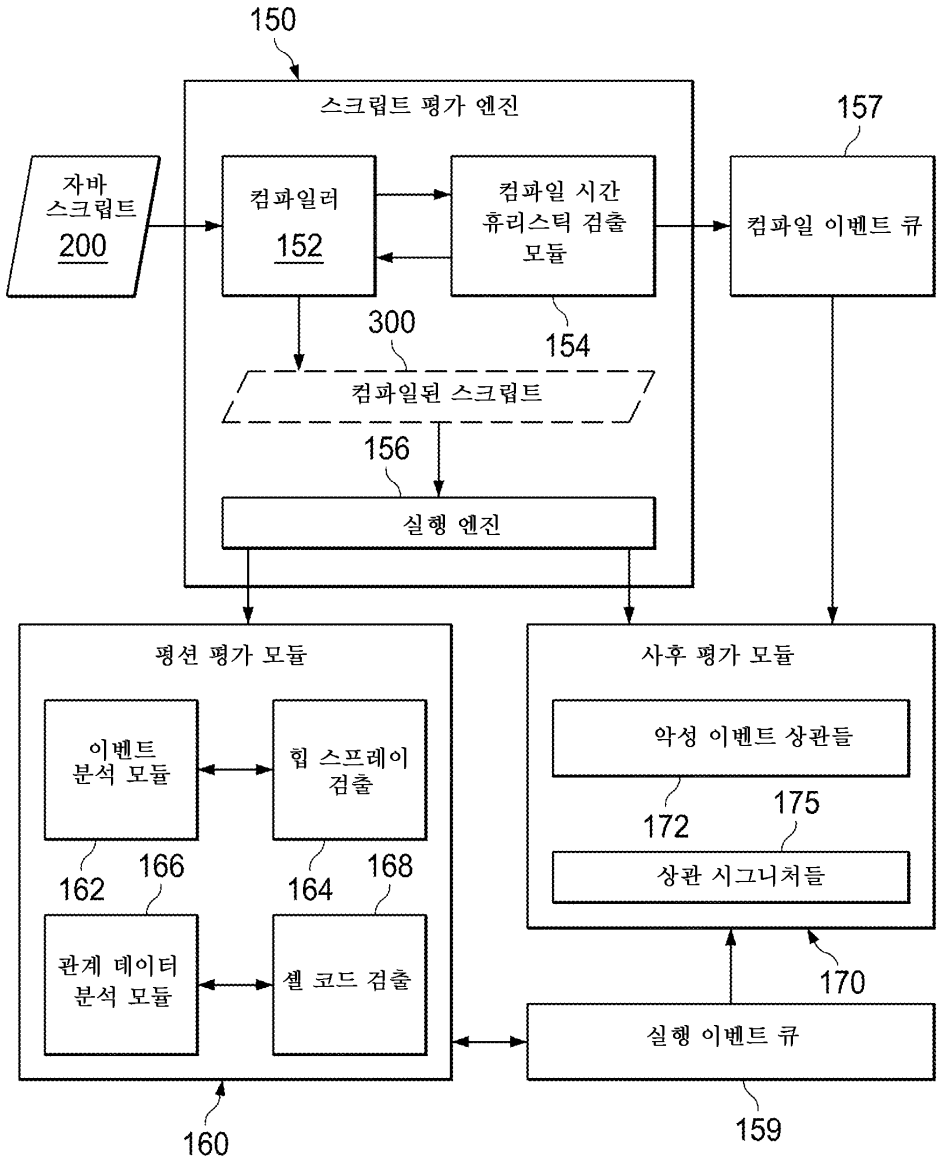
될 수 있다. 이 구현은, 평션이 미리 정해진 관련 평션이라는 것을 결정한 것에 응답하여, 평션이 호출될 때 실행 엔진으로부터 평션 평가 모듈로 제어를 넘기기 위한 수단; 및 평션이 실행을 끝낼 때 평션 평가 모듈으로부터 실행 엔진으로 제어를 넘기기 위한 수단을 포함할 수 있다. 이 구현은 제2 기준이 만족되는지를 결정하기 위해 대입문의 좌변 값 및/또는 우변 변수 명을 평가하기 위한 수단을 더 포함할 수 있다. 이 구현은 실행 동안 파라미터를 평션으로 넘기기 위한 수단, 및 제1 기준이 만족되었는지를 결정하기 위해 파라미터의 길이를 평가하기 위한 수단을 포함할 수 있다.

도면

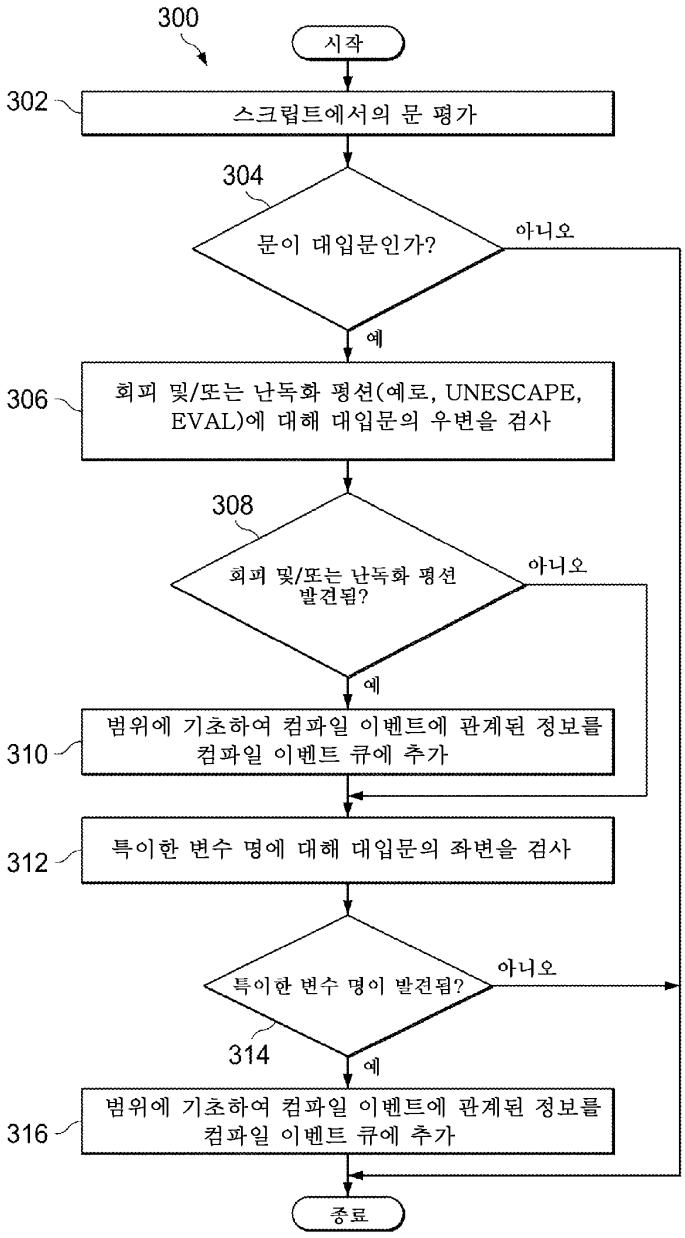
도면1



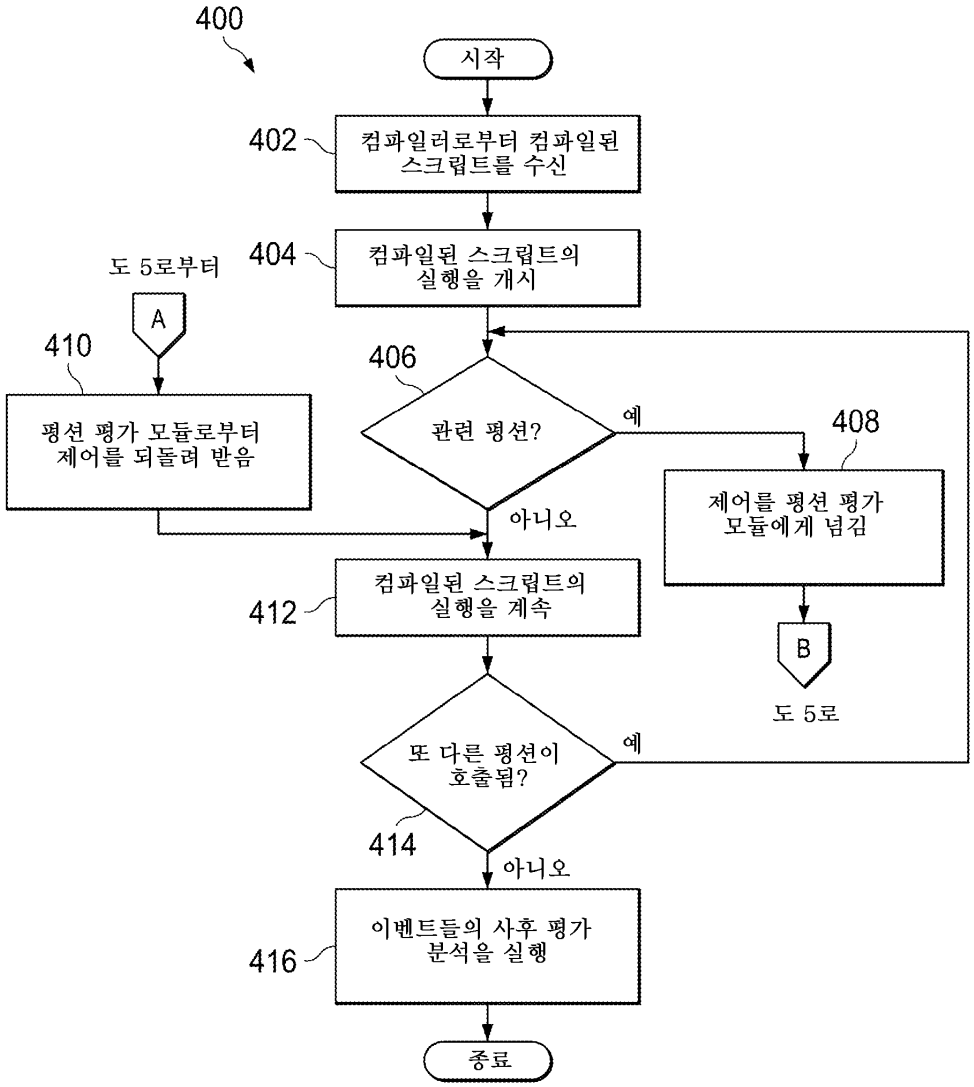
도면2


150
스크립트 평가 엔진
157
자바 스크립트
200
컴파일러
152
컴파일 시간 휴리스틱 검출 모듈
컴파일 이벤트 큐
300
154
컴파일된 스크립트
156
실행 엔진
평션 평가 모듈
사후 평가 모듈
이벤트 분석 모듈
힙 스프레이 검출
악성 이벤트 상관들
172
175
상관 시그니처들
162
166
164
168
관계 데이터 분석 모듈
셸 코드 검출
170
실행 이벤트 큐
160
159

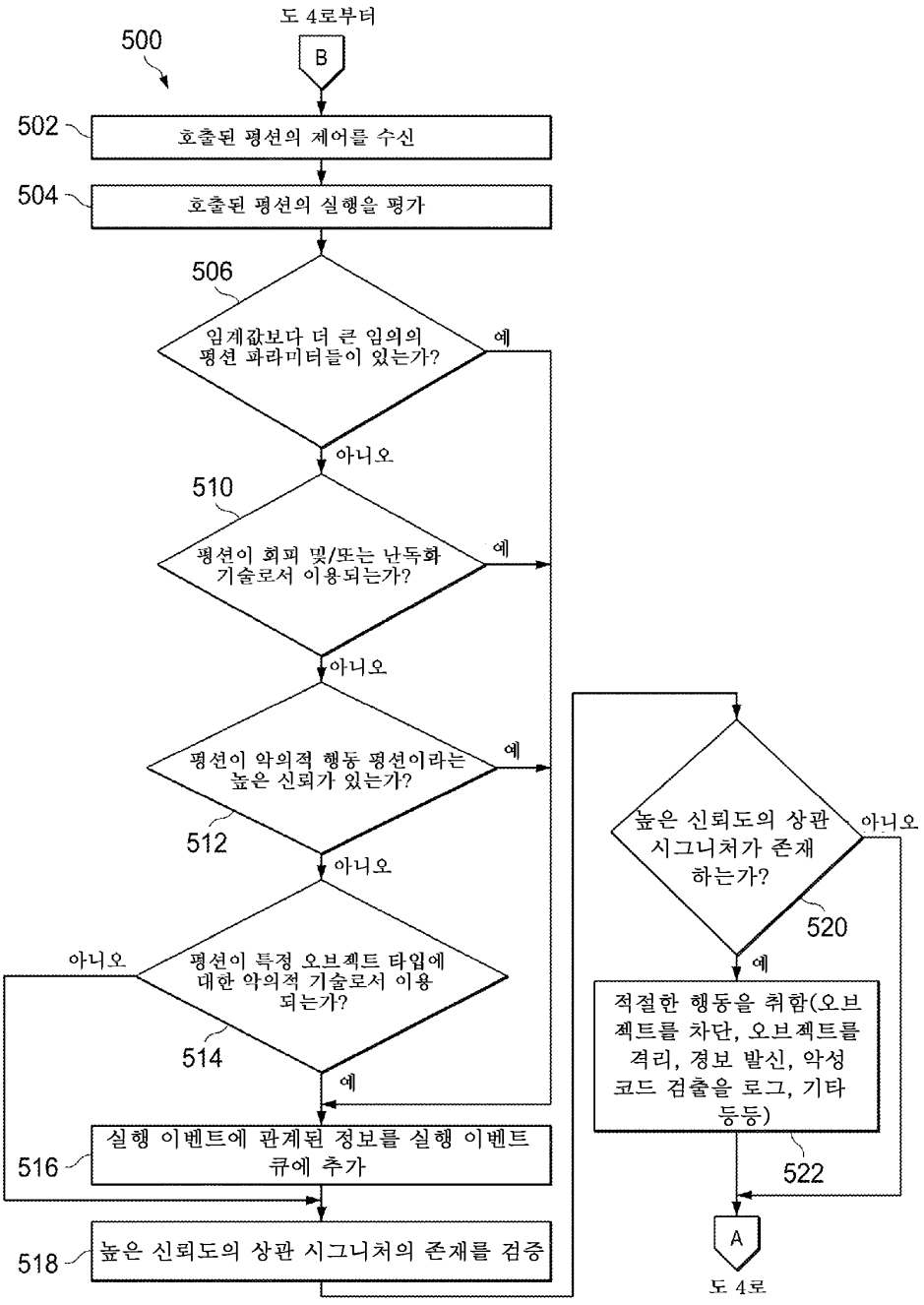
도면3


300
시작
302
스크립트에서의 문 평가
304
문이 대입문인가?
아니오
예
306
회피 및/또는 난독화 평션(예로, UNESCAPE, EVAL)에 대해 대입문의 우변을 검사
308
회피 및/또는 난독화 평션 발견됨?
아니오
예
310
범위에 기초하여 컴파일 이벤트에 관계된 정보를 컴파일 이벤트 큐에 추가
312
특이한 변수 명에 대해 대입문의 좌변을 검사
특이한 변수 명이 발견됨?
아니오
314
예
316
범위에 기초하여 컴파일 이벤트에 관계된 정보를 컴파일 이벤트 큐에 추가
종료

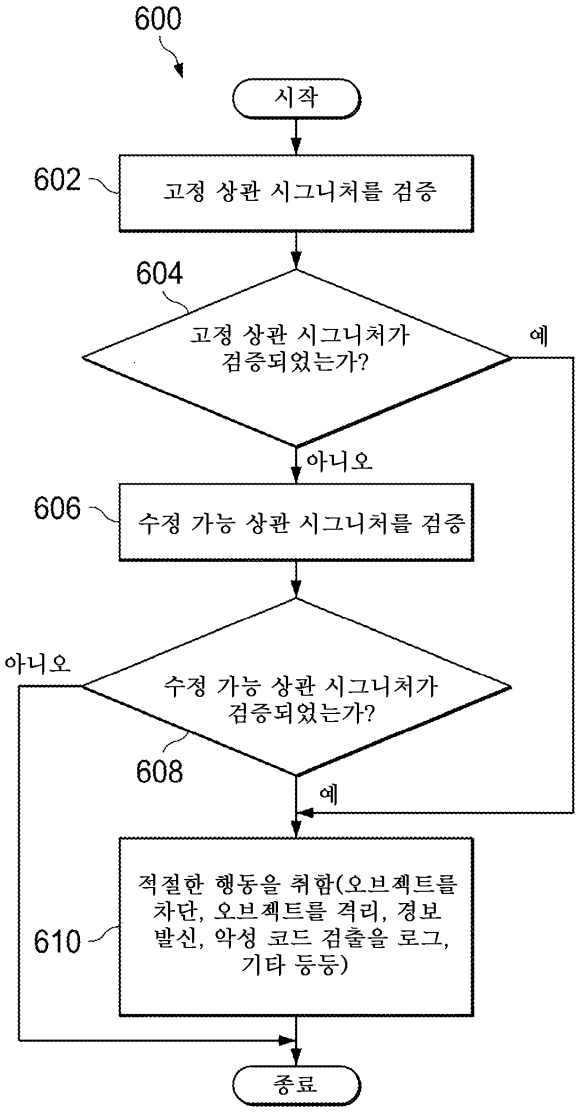
도면4


400
시작
402
컴파일러로부터 컴파일된 스크립트를 수신
404
컴파일된 스크립트의 실행을 개시
도 5로부터
A
410
평션 평가 모듈로부터 제어를 되돌려 받음
406
관련 평션?
예
408
제어를 평션 평가 모듈에게 넘김
아니오
B
도 5로
412
컴파일된 스크립트의 실행을 계속
또 다른 평션이 호출됨?
예
414
아니오
416
이벤트들의 사후 평가 분석을 실행
종료

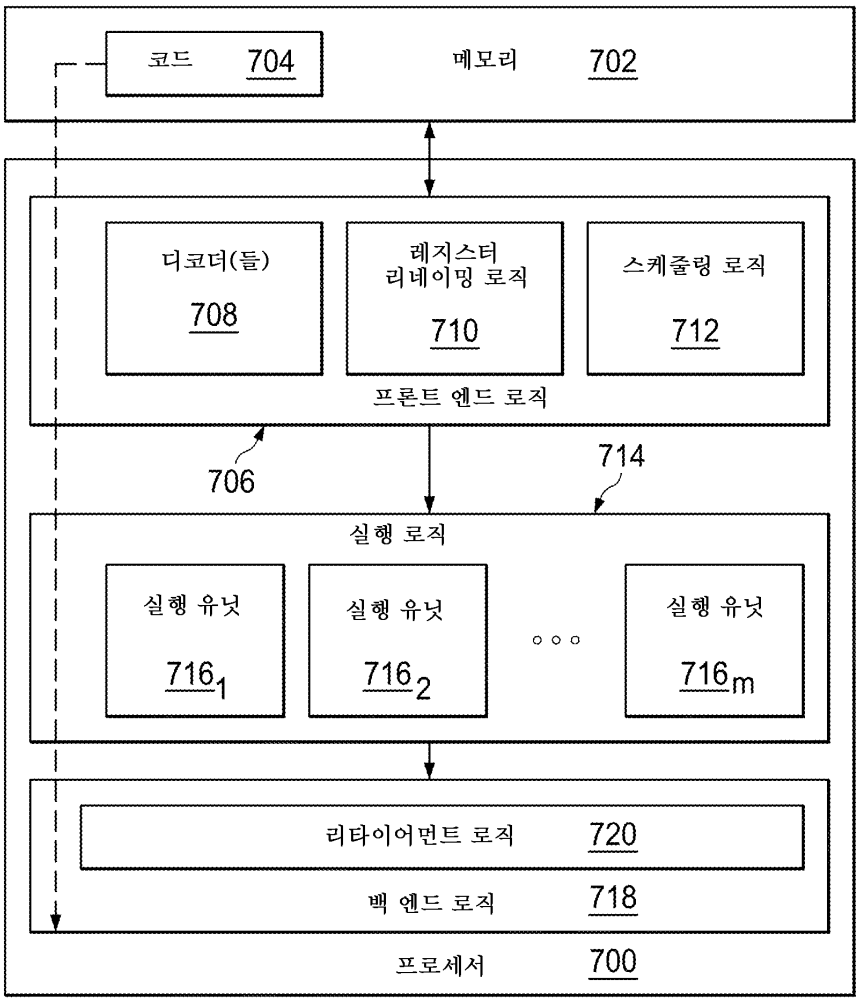
도면5


도 4로부터
B
500
502
호출된 평션의 제어를 수신
504
호출된 평션의 실행을 평가
506
임계값보다 더 큰 임의의 평션 파라미터들이 있는가?
예
아니오
510
평션이 회피 및/또는 난독화 기술로서 이용되는가?
예
아니오
평션이 악의적 행동 평션이라는 높은 신뢰가 있는가?
예
512
아니오
평션이 특정 오브젝트 타입에 대한 악의적 기술로서 이용되는가?
아니오
514
예
실행 이벤트에 관계된 정보를 실행 이벤트 큐에 추가
516
높은 신뢰도의 상관 시그니처의 존재를 검증
518
높은 신뢰도의 상관 시그니처가 존재하는가?
아니오
520
예
적절한 행동을 취함(오브젝트를 차단, 오브젝트를 격리, 경보 발신, 악성 코드 검출을 로그, 기타 등등)
522
A
도 4로

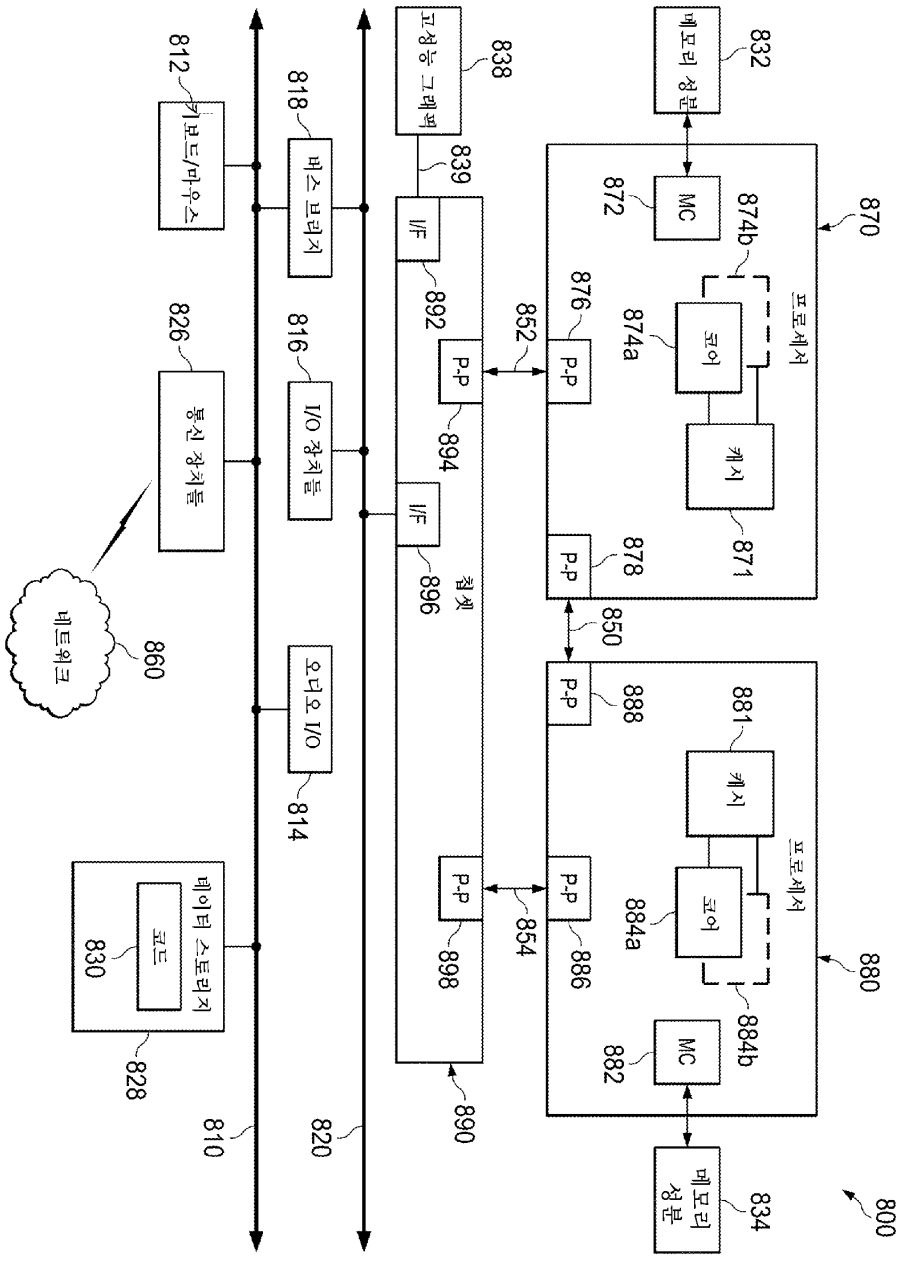
도면6


600
시작
602
고정 상관 시그니처를 검증
604
고정 상관 시그니처가 검증되었는가?
예
아니오
606
수정 가능 상관 시그니처를 검증
아니오
수정 가능 상관 시그니처가 검증되었는가?
608
예
610
적절한 행동을 취함(오브젝트를 차단, 오브젝트를 격리, 경보 발신, 악성 코드 검출을 로그, 기타 등등)
종료

도면7


코드 704
메모리 702
디코더(들) 708
레지스터 리네이밍 로직 710
스케줄링 로직 712
프론트 엔드 로직
706
714
실행 로직
실행 유닛 716_1
실행 유닛 716_2
실행 유닛 716_m
리타이어먼트 로직 720
백 엔드 로직 718
프로세서 700

도면8


800
810
812
키보드/마우스
818
버스 브리지
820
826
통신 장치들
816
I/O 장치들
860
네트워크
814
오디오 I/O
828
데이터 스토리지
830
코드
838
고성능 그래픽
839
I/F
892
P-P
894
I/F
896
칩셋
898
890
852
854
832
메모리 성분
872
MC
870
874b
876
874a
코어
캐시
871
878
850
888
881
캐시
884a
코어
884b
886
880
882
MC
834
메모리 성분