(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2009-0117705
(43) 공개일자 2009년11월12일

(51) Int. Cl.
G06F 15/00 (2006.01) *G06F 21/00* (2006.01)
G06F 11/00 (2006.01) *G06F 9/00* (2006.01)
(21) 출원번호 10-2009-7015518
(22) 출원일자 2007년12월31일
심사청구일자 없음
(85) 번역문제출일자 2009년07월23일
(86) 국제출원번호 PCT/US2007/089221
(87) 국제공개번호 WO 2008/083382
국제공개일자 2008년07월10일
(30) 우선권주장
11/618,470 2006년12월29일 미국(US)

(71) 출원인
마이크로소프트 코포레이션
미국 워싱턴주 (우편번호 : 98052) 레드몬드 원 마이크로소프트 웨이
(72) 발명자
로저스, 저스틴
미국 98052-6399 워싱턴주 레드몬드 원 마이크로소프트 웨이 마이크로소프트 코포레이션 국제 특허부 내
로렌스, 에릭 엠.
미국 98052-6399 워싱턴주 레드몬드 원 마이크로소프트 웨이 마이크로소프트 코포레이션 국제 특허부 내
브리지, 헨리 에프.
미국 98052-6399 워싱턴주 레드몬드 원 마이크로소프트 웨이 마이크로소프트 코포레이션 국제 특허부 내
(74) 대리인
양영준, 백만기

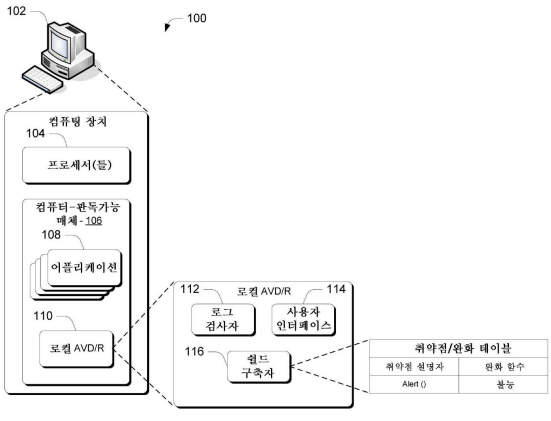
전체 청구항 수 : 총 20 항

(54) 자동적 취약점 검출 및 응답

(57) 요 약

다양한 실시예는 보안 취약점을 검출하고, 그에 따라, 익스플로이트가 실행되는 경우에도 프로그램의 무결성이 유지될 수 있도록 영향받은 프로그램을 수정할 수 있다. 적어도 몇몇 실시예에서, 로컬의 자동적 취약점 검출 및 응답(AVD/R) 컴포넌트는 사용자의 로컬 머신 상에서, 쉴드를 이용함으로써 잠재적인 취약점을 검출 및 완화하기 위해 실행되고, 원격의 자동적 취약점 검출 및 응답(AVD/R) 컴포넌트는 인지된 취약점을 완화하기 위해 하나 이상의 쉴드가 로컬로 전달 및 적용될 수 있도록, 인지된 취약점을 보고하기 위해 실행된다.

대 표 도



특허청구의 범위

청구항 1

컴퓨터-구현된 방법으로서,

로컬 프로그램 충돌을 검출하는 단계; 및

상기 검출에 응답하여, 상기 프로그램 충돌의 대상이었던 함수(function) 또는 인터페이스를 불능화(disabling)하는 단계

를 포함하는 방법.

청구항 2

제1항에 있어서,

상기 검출하는 단계 이후, 및 상기 불능화하는 단계 이전에, 충돌 로그(crash log)를 검사하여 상기 프로그램 충돌에 관련된 상기 함수 또는 인터페이스를 확인하는 단계를 더 포함하는 방법.

청구항 3

제1항에 있어서,

함수 또는 인터페이스가 불능화되었음을 사용자에게 알리는 단계를 더 포함하는 방법.

청구항 4

제1항에 있어서, 상기 불능화하는 단계는 쉴드(shield)를 구축 및 적용함으로써 수행되는 방법.

청구항 5

제4항에 있어서,

상기 쉴드는 자동적인 런타임 솔루션을 제공하는 방법.

청구항 6

제4항에 있어서,

상기 구축 및 적용하는 것은 취약점(vulnerability)과 완화 함수(mitigation function)를 관련시키는 취약점/완화 테이블을 이용함으로써 수행되는 방법.

청구항 7

제1항에 있어서,

상기 검출하는 단계 및 상기 불능화하는 단계는 웹 브라우저에 의해 수행되는 방법.

청구항 8

실행 시, 제1항의 방법을 구현하는 컴퓨터 판독가능 명령어들을 포함하는 하나 이상의 컴퓨터-판독가능 매체.

청구항 9

제8항의 하나 이상의 컴퓨터-판독가능 매체를 구현하는 컴퓨팅 시스템.

청구항 10

컴퓨터-구현된 방법으로서,

로컬 프로그램 충돌을 검출하는 단계;

상기 검출에 응답하여, 상기 프로그램 충돌이 보고될 수 있는지의 여부를 확인하기 위해 사용자에게 질의하는

단계;

상기 로컬 프로그램 충돌을 보고하는 단계; 및

상기 보고에 응답하여, 상기 프로그램 충돌의 대상이었던 함수 또는 인터페이스를 불능화하는 데에 효과적인 쉴드를 수신 및 적용하는 단계

를 포함하는 방법.

청구항 11

제10항에 있어서,

함수 또는 인터페이스가 불능화되었음을 사용자에게 알리는 단계를 더 포함하는 방법.

청구항 12

제10항에 있어서,

사용자 인터페이스를 통해, 불능화되었던 상기 함수 또는 인터페이스를 재활성화(re-enable)시키는 옵션을 상기 사용자에게 제공하는 단계를 더 포함하는 방법.

청구항 13

제10항에 있어서,

상기 적용하는 단계는 취약점과 완화 함수를 관련시키는 취약점/완화 테이블을 이용함으로써 수행되는 방법.

청구항 14

제10항에 있어서,

상기 검출하는 단계에 응답하여, 충돌 로그를 검사하여 상기 프로그램 충돌에 관련된 상기 함수 또는 인터페이스를 확인하는 단계를 더 포함하는 방법.

청구항 15

제10항에 있어서,

상기 검출하는 단계, 상기 질의하는 단계, 상기 보고하는 단계, 및 상기 수신 및 적용하는 단계는 웹 브라우저에 의해 수행되는 방법.

청구항 16

실행 시, 제10항의 방법을 구현하는 컴퓨터 판독가능 명령어들을 포함하는 하나 이상의 컴퓨터-판독가능 매체.

청구항 17

제16항의 하나 이상의 컴퓨터-판독가능 매체를 구현하는 컴퓨팅 시스템.

청구항 18

컴퓨터 구현된 방법으로서,

로컬 프로그램 충돌을 검출하는 단계;

상기 충돌을 원격 서버에 보고하기 위한 승인을 사용자에게 요청하는 단계;

사용자 승인이 수여되면,

상기 충돌을 상기 원격 서버에 보고하고,

상기 보고에 응답하여, 상기 충돌에 관련된 함수 또는 인터페이스를 불능화하도록 구성된 하나 이상의 쉴드를 다운로드하고,

상기 함수 또는 인터페이스를 불능화하기 위해 상기 하나 이상의 쉴드를 적용하는 단계; 및

사용자 승인이 수여되지 않으면, 상기 프로그램 충돌의 대상이었던 함수 또는 인터페이스를 불능화하는 단계

를 포함하는 방법.

청구항 19

제18항에 있어서,

사용자 인터페이스를 통해, 불능화되었던 상기 함수 또는 인터페이스를 재활성화시키는 옵션을 상기 사용자에게 제공하는 단계를 더 포함하는 방법.

청구항 20

제18항에 있어서,

상기 검출하는 단계 및 상기 요청하는 단계는 웹 브라우저에 의해 수행되는 방법.

명 세 서

배 경 기 술

<1> 소프트웨어 보안 취약점(software security vulnerability)을 완화하는 많은 방법은 반응이 빠르고(reactive) 시간에 매우 민감하다. 즉, 취약점이 발견되면, 소프트웨어 회사는 일반적으로, 얼마의 시간이 지난 후에, 공격자들이 그 취약점을 익스플로이트(exploit)하는 것을 방지하도록 지시된 패치(patch)를 발행한다. 과거에 이 전략은 사용자를 보호하는 데에 매우 효과적이었지만, 그 효율성은 부분적으로 (1) 취약점 발견자가 해커가 찾기 전에 취약점을 찾는 것, (2) 취약점 발견자가 문제점을 공개적으로 발표하기 전에 소프트웨어 회사에 문제점을 보고하는 것, 및 (3) 익스플로이트가 전개되는 경우 채용하는 사용자가 그 익스플로이트로부터 보호되도록 하는 높은 패치 채용률(patch adopting rates)을 필요로 한다.

<2> 불행히도, 최근의 추세는 이러한 필요조건에 있어서 좋은 징조가 아니다. 특히, 당일 익스플로이트의 비율(rate of 0-day exploit)(즉, 공개되지 않고 수정되지 않은 보안 취약점에 대해 발행된 익스플로이트)이 증가하였고, 패치 채용률은 계속해서 낮아지고 있다. 보안 전망이 현저히 악화되는 것을 방지하기 위해, 소프트웨어 제작자는 취약점을 더 빨리 발견하고 또한, 완화하는 방법을 찾아야 한다.

발명의 상세한 설명

<3> <요약>

<4> 다양한 실시예는 보안 취약점(security vulnerability)을 검출하고, 그에 따라, 익스플로이트(exploit)가 실행되는 경우에도 프로그램의 무결성이 유지될 수 있도록, 영향을 받은 프로그램을 수정할 수 있다.

<5> 적어도 몇몇 실시예에서, 로컬의 자동적 취약점 검출 및 응답(AVD/R) 컴포넌트는 사용자의 로컬 머신 상에서, 쉴드(shield)를 이용함으로써 잠재적인 취약점을 검출하고 완화하기 위해 실행되고, 원격의 자동적 취약점 검출 및 응답(AVD/R) 컴포넌트는 인지된 취약점을 완화하기 위해 하나 이상의 쉴드가 로컬로 전달 및 적용될 수 있도록, 인지된 취약점을 보고하기 위해 실행될 수 있다.

실 시 예

<12> <개관>

<13> 다양한 실시예는 보안 취약점(security vulnerability)을 검출하고, 그에 따라, 익스플로이트(exploit)가 실행되는 경우에도 프로그램의 무결성이 유지될 수 있도록 영향받은 프로그램을 수정할 수 있다.

<14> 적어도 몇몇 실시예에서, 로컬의 자동적 취약점 검출 및 응답(AVD/R) 컴포넌트는 사용자의 로컬 머신 상에서, 쉴드를 이용함으로써 잠재적인 취약점을 검출하고 완화하기 위해 실행되고, 원격의 자동적 취약점 검출 및 응답(AVD/R) 컴포넌트는 인지된 취약점을 완화하기 위해 하나 이상의 쉴드(shield)가 로컬로 전달 및 적용될 수 있도록, 인지된 취약점을 보고하기 위해 실행된다.

<15> 다음의 설명에서, "보안 취약점에 대한 개론" 섹션이 제공되고, 보안 취약점의 개념 및 보안 취약점이 어떻게 발생할 수 있는지를 매우 일반적으로 설명한다. 이에 후속하여, "로컬 AVD/R" 섹션이 제공되고, 취약점 검출

및 응답에 대한 로컬 솔루션을 논의한다. 이에 후속하여, "원격 AVD/R" 섹션이 제공되고, 취약점 검출에 대한 다양한 원격 솔루션을 논의한다. 마지막으로, "로컬 AVD/R과 원격 AVD/R 둘 다를 이용하기"가 제공되고, 보호의 연속을 제공하기 위해 로컬 접근과 원격 접근 둘 다가 어떻게 적용될 수 있는지를 설명한다.

<16> **보안 취약점에 대한 개론**

<17> 많은 보안 취약점은 프로그래밍 에러로부터 나온다. 취약점에 이를 수 있는 다수의 서로 다른 유형의 프로그래밍 에러들이 존재한다. 예를 들어, 흔한 프로그래밍 에러는 버퍼 오버플로우를 허용하는 에러이다. 이와 같은 상황에서, 프로그래머는 데이터를 유지하기 위해 특정 공간량을 할당할 수 있다. 익스플로이터(exploiter)는, 당신이 기대한 것보다 더 많은 데이터를 프로그램에 제공하는 경우, 및 버퍼 오버플로우의 가능성을 완화 또는 제거하기 위한 올바른 점검을 제자리에 놓지 않은 경우, 이 초과 데이터는 오버플로우를 야기할 수 있다는 것을 이해할 수 있다. 오버플로우 조건을 이용하여, 익스플로이터는 버퍼 내에 수신된 데이터의 끝에 데이터 또는 코드를 첨부하고 오버플로우를 야기할 수 있다. 첨부된 데이터 또는 코드가 실행되면, 그것은 프로그램을 변경하거나 임의의 방법으로 그것의 기능을 수정할 수 있다. 따라서, 프로그래밍 에러에 의해, 보안 취약점은 익스플로이트될 수 있다.

<18> 그러나, 종종, 보안 취약점의 익스플로이트화(exploitation)는 프로그램 충돌에 이를 수 있다. 상술한 예에서, 익스플로이트화는 프로그램이 메모리의 임의의 랜덤 부분을 감시하게 할 수 있고, 무효한 동작을 야기하는 코드를 실행하기 시작하여 프로그램이 충돌하게 할 수 있다.

<19> 따라서, 프로그램 충돌로부터, 우리는 프로그램에 문제가 있음을 유추할 수 있다. 이 문제는 보안 취약점과 관련될 수 있다. 즉, (1) 프로그램 충돌을 야기하는 동일한 프로그래밍 에러들 중 다수가 익스플로이트가능하고, (2) 익스플로이트 개발은 양호한 정도의 시도 및 에러를 수반하고 - 따라서, 익스플로이트 개발의 초기 단계동안 실패한 시도로 인해 프로그램이 충돌할 것임 - , (3) 익스플로이트는 종종, 특정 버전의 프로그램에 대해서만 동작하고 때때로 다른 버전들과 충돌할 것이기 때문에, 프로그램 충돌은 종종, 취약점의 표시이다.

<20> **로컬 AVD/R**

<21> 도 1은 일 실시예에 따른 시스템(총제적으로 100)을 도시한다. 시스템(100)은 하나 이상의 프로세서(104), 하나 이상의 컴퓨터-판독가능 매체(106), 및 컴퓨터-판독가능 매체 상에 상주하고 프로세서(들)에 의해 실행가능한 하나 이상의 어플리케이션(108)을 갖는 컴퓨팅 장치(102)를 포함한다. 또한, 컴퓨팅 장치(102)는 이 예에서, 소프트웨어로 구현되는 로컬 AVD/R 컴포넌트(110)를 포함한다.

<22> 컴퓨팅 장치(102)는 데스크탑 컴퓨터의 형태로 도시되어 있지만, 다른 컴퓨팅 장치들이 특허 청구범위의 대상의 취지 및 범주로부터 벗어나지 않고서 이용될 수 있음을 인식하고 이해해야 한다. 예를 들어, 다른 컴퓨팅 장치는 예를 들어, 휴대형 컴퓨터, PDA와 같은 핸드헬드 컴퓨터, 셀 폰 등을 포함할 수 있지만, 이것으로 제한되지 않는다.

<23> 이 예에서, 로컬 AVD/R 컴포넌트(110)는 로그 검사자(112), 사용자 인터페이스 컴포넌트(114) 및 쉴드 구축자(116)를 포함한다. 동작 시, 로컬 컴퓨팅 장치 상의 프로그램이 충돌할 때, 본 분야에 숙련된 기술을 가진 자에 의해 인지되는 바와 같이, 충돌에 관련된 정보가 충돌 로그(crash log)에 입력된다. 통상적으로, 충돌 로그는 특정 충돌에 관련된 파라미터를 설명하는 정보를 포함한다. 예를 들어, 충돌 로그는 충돌한 프로그램을 설명하는 정보, 프로그램 내의 어느 함수 또는 인터페이스가 충돌하였는지, 및/또는 충돌을 일으킨 함수 또는 인터페이스에 관련된 임의의 파라미터를 포함할 수 있다. 로컬 AVD/R 컴포넌트(110)의 로그 검사자(112)는 충돌에 대해 모니터링할 수 있고, 충돌이 일어나면, 자동적으로 충돌에 관련된 정보를 위해 충돌 로그를 검사할 수 있다. 이것은 어느 함수 또는 인터페이스가 충돌에 관련되는지를 확인하는 것을 포함할 수 있다. 로그 검사자가 충돌의 원인을 확인하였으면, 그것은 쉴드 구축자(116)를 이용하여, 함수 또는 인터페이스를 불능화시키는 자동 런타임 솔루션을 효과적으로 제공하는 쉴드를 구축할 수 있다. 그 후, 이 사실은 사용자 인터페이스(114)를 통해 사용자에게 보고될 수 있다.

<24> 예로서, 다음을 고려해 보자. 사용자가 자신의 브라우저 어플리케이션을 이용하고 있고 함수 Alert()가 충돌한다고 가정해 보자. 이 충돌에 응답하여, 충돌 로그는 충돌한 함수의 이름, 및 그것이 충돌한 장소와 같은 충돌에 속하는 정보로 업데이트된다. 이 정보를 이용하여, 로그 검사자(112)는 쉴드 구축자(116)를 이용하여, Alert() 함수를 자동으로 불능화시키는 쉴드를 구축할 수 있다. 하나 이상의 실시예에서, 쉴드 구축자는 도면에 도시된 것과 같은 취약점/완화 테이블을 유지할 수 있다. 여기서, 취약점/완화 테이블은 취약점 설명자들을 나열하는 컬럼, 및 완화 함수들을 나열하는 컬럼을 포함한다. 취약점 설명자는 완화 함수의 대상인 특정 함수

또는 인터페이스를 설명한다. 완화 함수는 이용되는 특정 완화 함수를 설명한다. 상술한 예에서, 충돌이 일어날 때, 쉴드 구축자는 취약점/완화 테이블 내에 엔트리를 형성(make an entry)하고, 취약점 설명자 컬럼 내에 "Alert()"를 추가한다. 또한, 쉴드 구축자는 완화 함수 컬럼 내의 대응하는 로우에 "불능(Disable)"을 추가한다. 이것은 어플리케이션(이 경우에는, 사용자의 브라우저)에 Alert() 함수가 불능화되었다는 것을 알린다.

<25> 또한, 적어도 몇몇 실시예에서, 이 사실은 사용자 인터페이스 컴포넌트(114)를 통해 사용자에게 보고된다. 대응하는 사용자 인터페이스를 이용하여, 사용자는 효과적으로, 이 함수를 다시 작동시키는 것을 선택할 수 있다. 따라서, 이 실시예에서, 취약점의 잠재적 존재가 검출되고, 대응하는 함수 또는 인터페이스가 선택적으로 불능화되어, 미래의 익스플로이트를 방지한다.

<26> 도 2는 일 실시예에 따른 방법의 단계들을 설명하는 흐름도이다. 이 방법은 임의의 적합한 하드웨어, 소프트웨어, 펌웨어 또는 이들의 조합에 관련하여 구현될 수 있다. 하나 이상의 실시예에서, 이 방법은 도 1에서 도시되고 설명된 것과 같은 시스템에 관련하여 구현될 수 있다. 다른 시스템들은 특허청구범위의 대상의 취지 및 범주로부터 벗어나지 않고서 이용될 수 있다.

<27> 단계(200)는 로컬 프로그램 충돌을 검출한다. 이것이 어떻게 행해질 수 있는지에 대한 예는 상술되었다. 단계(202)는 충돌 로그를 검사하여 충돌 주변의 환경을 확인한다. 단계(204)는 충돌의 대상이었던 함수 또는 인터페이스를 불능화한다. 이것이 어떻게 행해질 수 있는지에 대한 예는 위에서 제공되었다. 단계(206)는 사용자에게 불능화된 함수 또는 인터페이스를 알린다.

<28> **원격 AVD/R**

<29> 하나 이상의 실시예에서, 프로그램 충돌에 관련된 정보는 원격으로 이용될 수 있다. 특히, 충돌이 일어날 때, 이 정보는 추가의 분석을 위해 원격으로 보고될 수 있다. 이러한 분석은 예를 들어, 충돌의 소스 및 다양한 관련 파라미터들을 분석하는 것뿐만 아니라, 복수의 사용자들에 걸쳐 이러한 충돌을 평가하여 이 충돌에 관련된 패턴이 존재하는지의 여부를 확인하는 것을 포함할 수 있지만, 이것으로 제한되지 않는다. 취약점이 검출되면, 대응하는 쉴드가 구축되고, 그 취약점을 익스플로이트하려고 하는 익스플로이트화로부터 보호하기 위해 사용자에게 제공될 수 있다.

<30> 일례로서, 도 3을 고려해 보자. 여기서, 일 실시예에 따른 시스템(총체적으로, 300)이 도시된다. 시스템(300)은 하나 이상의 프로세서(304), 하나 이상의 컴퓨터-판독가능 매체(306), 및 컴퓨터-판독가능 매체(306) 상에 상주하고 프로세서(들)에 의해 실행되는 하나 이상의 어플리케이션(308)을 갖는 컴퓨팅 장치(320)를 포함한다. 또한, 컴퓨팅 장치(302)는 이 예에서 소프트웨어로 구현되는 원격 AVD/R 컴포넌트(310)를 포함한다.

<31> 컴퓨팅 장치(302)는 데스크탑 컴퓨터의 형태로 도시되어 있지만, 다른 컴퓨팅 장치들이 특허청구범위의 대상의 취지 및 범주로부터 벗어나지 않고서 이용될 수 있음을 인식하고 이해해야 한다. 예를 들어, 다른 컴퓨팅 장치는 예를 들어, 휴대형 컴퓨터, PDA와 같은 핸드헬드 컴퓨터, 셀 폰 등을 포함할 수 있지만, 이것으로 제한되지 않는다.

<32> 이 예에서, 원격 AVD/R 컴포넌트(310)는 로그 검사자(312), 사용자 인터페이스 컴포넌트(314) 및 충돌 보고자 컴포넌트(316)를 포함한다. 동작 시, 로컬 컴퓨팅 장치 상의 프로그램이 충돌할 때, 이 충돌에 관련된 정보는 상술된 바와 같이 충돌 로그 내에 입력된다. 원격 AVD/R 컴포넌트(310)의 로그 검사자(312)는 충돌에 대해 모니터링할 수 있고, 충돌이 일어날 때, 자동적으로 이 충돌에 관련된 정보를 위해 충돌 로그를 검사할 수 있다. 이것은 어느 함수 또는 인터페이스가 충돌에 관련되는지를 확인하는 것을 포함할 수 있다. 로그 검사자가 충돌의 원인을 확인하였으면, 원격 AVD/R 컴포넌트는 사용자 인터페이스(314)를 통해, 사용자에게 사용자가 추가의 분석을 위해 이 충돌을 원격 서버에 보고하기를 원하는지의 여부를 물어볼 수 있다. 사용자가 충돌 정보를 보고하는 것을 선택하면, 정보는 서버에 의해 결집되고 분석된다. 적어도 몇몇 실시예에서, 충돌 정보의 분석은 인간 전문가를 채용하여, 임의의 익스플로이트화가 채용되었는지의 여부를 분석하고 확인하는 것을 포함할 수 있다.

<33> 충돌 로그(들)의 분석이 취약점이 익스플로이트되었음을 나타내는 경우에, 상술한 것과 같은 하나 이상의 쉴드가 로컬로 다운로드 및 적용됨으로써 개발되고 이용될 수 있다. 하나 이상의 실시예에서, 사용자 인터페이스(314)는 불능화되었거나 불능화되어 있는 함수 또는 인터페이스를 다시 활성화시키는 옵션을 사용자에게 제공할 수 있다.

<34> 도 4는 일 실시예에 따른 방법의 단계들을 설명하는 흐름도이다. 이 방법은 임의의 적합한 하드웨어, 소프트웨어, 펌웨어 또는 이들의 임의의 조합에 관련하여 구현될 수 있다. 하나 이상의 실시예에서, 이 방법은 도 3에

서 도시되고 설명된 것과 같은 시스템에 관련하여 구현될 수 있다. 다른 시스템이 특허청구범위의 대상의 취지 및 범주로부터 벗어나지 않고서 이용될 수 있다.

<35> 단계(400)는 로컬 프로그램 충돌을 검출한다. 이것이 어떻게 행해질 수 있는지에 대한 예는 상술되었다. 단계(402)는 충돌 로그를 검사하여, 충돌 주변의 환경을 확인한다. 단계(404)는 충돌이 보고될 수 있는지의 여부를 확인하기 위해 사용자에게 질의한다. 단계(406)는 사용자가 인증을 부여한 경우에 충돌을 원격 서버에 보고한다. 단계(408)는 충돌의 대상이었던 함수 또는 인터페이스를 효과적으로 불능화하는 쉴드를 수신하고 구현한다. 쉴드는 인터넷과 같은 네트워크를 통해 그것을 다운로드함으로써 수신될 수 있다. 이 단계의 부분으로서, 사용자 인터페이스는 함수 또는 인터페이스를 불능화하는 옵션을 사용자에게 제공하거나 함수 또는 인터페이스를 재활성화시키기 위해 이용될 수 있다. 이것이 어떻게 행해질 수 있는지에 대한 예는 상술되었다.

<36> **로컬 AVD/R과 원격 AVD/R 둘 다를 이용하기**

<37> 하나 이상의 실시예에서, 프로그램 충돌에 관련된 정보는 로컬과 원격 둘 다로 이용될 수 있다. 특히, 충돌이 일어날 때, 이 정보는 쉴드를 적용함으로써 영향받은 함수 또는 인터페이스를 불능화시키기 위해 로컬로 이용될 수 있다. 또한, 이 정보는 상술된 바와 같은 분석을 수행하기 위해 원격으로 이용될 수 있다. 이러한 분석은 예를 들어, 충돌의 소스(source) 및 다양한 관련된 파라미터를 분석하는 것뿐만 아니라, 복수의 사용자들에 걸쳐 이러한 충돌을 평가하여 충돌에 관련된 패턴이 존재하는지의 여부를 확인하는 것을 포함하지만, 이것으로 제한되지 않는다. 취약점이 검출되면, 대응하는 쉴드가 구축되고 취약점을 익스플로이트하려는 임의의 익스플로이트화로부터 보호하기 위해 사용자에게 제공될 수 있다.

<38> 일례로서, 도 5를 고려해 보자. 여기에서, 일 실시예에 따른 시스템(총체적으로, 500)이 도시된다. 시스템(500)은 하나 이상의 프로세서(504), 하나 이상의 컴퓨터-판독가능 매체(506), 및 컴퓨터-판독가능 매체 상에 상주하고 프로세서(들)에 의해 실행될 수 있는 하나 이상의 어플리케이션(508)을 갖는 컴퓨팅 장치(502)를 포함한다. 또한, 컴퓨팅 장치(502)는 이 예에서, 소프트웨어로 구현되는 로컬/원격 AVD/R 컴포넌트(510)를 포함한다.

<39> 컴퓨팅 장치(502)는 데스크탑 컴퓨터의 형태로 도시되어 있지만, 다른 컴퓨팅 장치가 특허청구범위의 대상의 취지 및 범주로부터 벗어나지 않고서 이용될 수 있음을 인지하고 이해해야 한다. 예를 들어, 다른 컴퓨팅 장치는 예를 들어, 휴대형 컴퓨터, PDA와 같은 핸드헬드 컴퓨터, 셀 폰 등을 포함할 수 있지만, 이것으로 제한되지 않는다.

<40> 이 예에서, 로컬/원격 AVD/R 컴포넌트(510)는 로그 검사자(512), 사용자 인터페이스 컴포넌트(514), 보고자 컴포넌트(516) 및 쉴드 구축자(518)를 포함한다. 동작 시, 로컬 컴퓨팅 장치 상의 프로그램이 충돌할 때, 상술한 바와 같이, 충돌에 관련된 정보가 충돌 로그에 입력된다. 로컬/원격 AVD/R 컴포넌트(510)의 로그 검사자(512)는 충돌에 대해 모니터링하고, 충돌이 일어날 때, 자동적으로 충돌에 관련된 정보를 위해 충돌 로그를 검사할 수 있다. 이것은 어느 함수 또는 인터페이스가 충돌에 관련되는지를 확인하는 것을 포함할 수 있다. 로그 검사자가 충돌의 원인을 확인하였으면, 로컬/원격 AVD/R 컴포넌트는 상술한 바와 같이 로컬로 쉴드를 적용하여, 충돌에 관련된 함수 또는 인터페이스를 불능화할 수 있다. 이것은 또한, 사용자가 불능화되었던 함수 또는 인터페이스를 재활성화시키는 것도 허용할 수 있는 사용자 인터페이스 컴포넌트(514)를 통해 사용자에게 보고될 수 있다.

<41> 또한, 하나 이상의 실시예에서, 로컬/원격 AVD/R 컴포넌트는 사용자 인터페이스(514)를 통해, 사용자가 추가의 분석을 위해 충돌을 원격 서버에 보고하는 것을 원하는지의 여부를 사용자에게 물어볼 수 있다. 사용자가 충돌 정보를 보고하는 것을 선택하면, 정보는 서버에 의해 결집되고 분석된다. 적어도 몇몇 실시예에서, 충돌 정보의 분석은 인간 전문가를 채용하여, 어떤 익스플로이트화라도 채용되었는지의 여부를 분석하고 확인하는 것을 포함할 수 있다.

<42> 충돌 로그(들)의 분석이 취약점이 익스플로이트되었음을 나타내는 경우에, 상술한 것과 같은 하나 이상의 쉴드는 로컬 머신 상에 다운로드되고 적용됨으로써 개발되고 채용될 수 있다. 하나 이상의 실시예에서, 사용자 인터페이스(514)는 불능화되었거나 불능화되어 있는 함수 또는 인터페이스를 재활성화시키는 옵션을 사용자에게 제공할 수 있다.

<43> 도 6은 일 실시예에 따른 방법의 단계들을 설명하는 흐름도이다. 이 방법은 임의의 적합한 하드웨어, 소프트웨어, 펌웨어 또는 이들의 임의의 조합에 관련하여 구현될 수 있다. 하나 이상의 실시예에서, 이 방법은 도 6에서 도시되고 설명된 것과 같은 시스템에 관련하여 구현될 수 있다. 다른 시스템은 특허청구범위의 대상의 취지

및 범주로부터 벗어나지 않고서 이용될 수 있다.

<44> 단계(600)는 로컬 프로그램 충돌을 검출하고, 단계(602)는 충돌을 원격 서버에 보고하기 위한 승인을 사용자에게 요청한다. 단계(604)에서 사용자 승인이 수여되면, 단계(606)는 충돌의 원인을 확인하고, 충돌을 원격 서버에 보고하고, 충돌에 이용가능할 수 있는 임의의 솔루션에 대해 점검한다. 원격 서버에 충돌을 보고함으로써, 충돌, 및 복수의 사용자들에 걸친 임의의 관련된 패턴에 대한 분석이 수행될 수 있다. 분석은 자동적인 기계 분석과 인간의 분석 둘 다를 포함할 수 있다. 그 후, 단계(608)는 임의의 관련 쉴드를 로컬로 다운로드 및 적용하고, 단계(610)는 사용자에게 알린다.

<45> 한편, 단계(604)에서 승인이 수여되지 않으면, 단계(612)는 충돌하는 함수 또는 인터페이스를 식별하고, 단계(614)는 쉴드 자격(shield eligibility)을 판정하다. 단계(616)가 문제를 다루는 적격의 로컬 쉴드가 존재한다고 판정하면, 단계(618)는 상술한 바와 같이 쉴드를 적용하고, 단계(620)는 사용자에게 알린다. 한편, 적격의 쉴드가 존재하지 않으면, 단계(620)는 사용자에게 알린다.

<46> 상술된 실시예는 임의의 적합한 어플리케이션에 관련하여 구현될 수 있고, 어플리케이션의 부분, 또는 어플리케이션에 의해 이용되는 별도의 컴포넌트를 포함할 수 있다. 예를 들어, 하나 이상의 실시예에서, 상술된 기능은 웹 브라우저, 인스턴트 메시징 어플리케이션 또는 임의의 다른 적합한 어플리케이션 또는 소프트웨어 컴포넌트 또는 시스템의 부분으로서 구현될 수 있다. 예를 들어, 이 기능은 운영 체제의 부분으로서 구현될 수 있다.

<47> 하나 이상의 실시예에서, 하나 이상의 소위 평판 서비스(reputation service)가 보안을 더 강화하는 데에 이용될 수 있다. 특히, 평판 서비스 또는 제3자 서비스는 보안 익스플로이트화에 대해 광범위하게 모니터링하고, 임의의 인지된 취약점 또는 실제의 취약점을 적절한 당국에 보고할 수 있다. 예를 들어, 평판 서비스는 특정 웹 페이지에 관련된 특정 함수에 관련된 보안 취약점이 있다는 것을 검출할 수 있다. 일단 검출되면, 평판 서비스는 마이크로소프트와 같은 적절한 회사에 취약점을 보고하고/하거나, 쉴드가 선택적으로 다운로드 되게 하거나 다양한 사용자들이 이 인지된 취약점을 다룰 수 있게 할 수 있다.

<48> **결론**

<49> 다양한 실시예들이 보안 취약점을 검출하고, 그에 따라, 익스플로이트가 실행되는 경우에도 프로그램의 무결성이 유지될 수 있도록 영향받은 프로그램을 수정할 수 있다. 적어도 몇몇 실시예에서, 로컬의 자동적 취약점 검출 및 응답(AVD/R) 컴포넌트는 사용자의 로컬 머신 상에서, 쉴드를 이용함으로써 잠재적인 취약점을 검출 및 완화하기 위해 실행되고, 원격의 자동적 취약점 검출 및 응답(AVD/R) 컴포넌트는 인지된 취약점을 완화하기 위해 하나 이상의 쉴드가 로컬로 전달되고 적용될 수 있도록, 인지된 취약점을 보고하기 위해 실행된다.

<50> 본 발명은 구조적 특징 및/또는 방법론적 단계에 특정적인 언어로 설명되었지만, 첨부의 특허청구범위에서 정의된 본 발명은 설명된 특정 특징 또는 단계로 반드시 제한되는 것이 아님을 이해해야 한다. 오히려, 특정 특징 및 단계는 특허청구범위의 발명을 구현하는 바람직한 형태로서 개시된다.

도면의 간단한 설명

<6> 도 1은 일 실시예에 따른 시스템을 도시.

<7> 도 2는 일 실시예에 따른 방법의 단계들을 설명하는 흐름도.

<8> 도 3은 일 실시예에 따른 시스템을 도시.

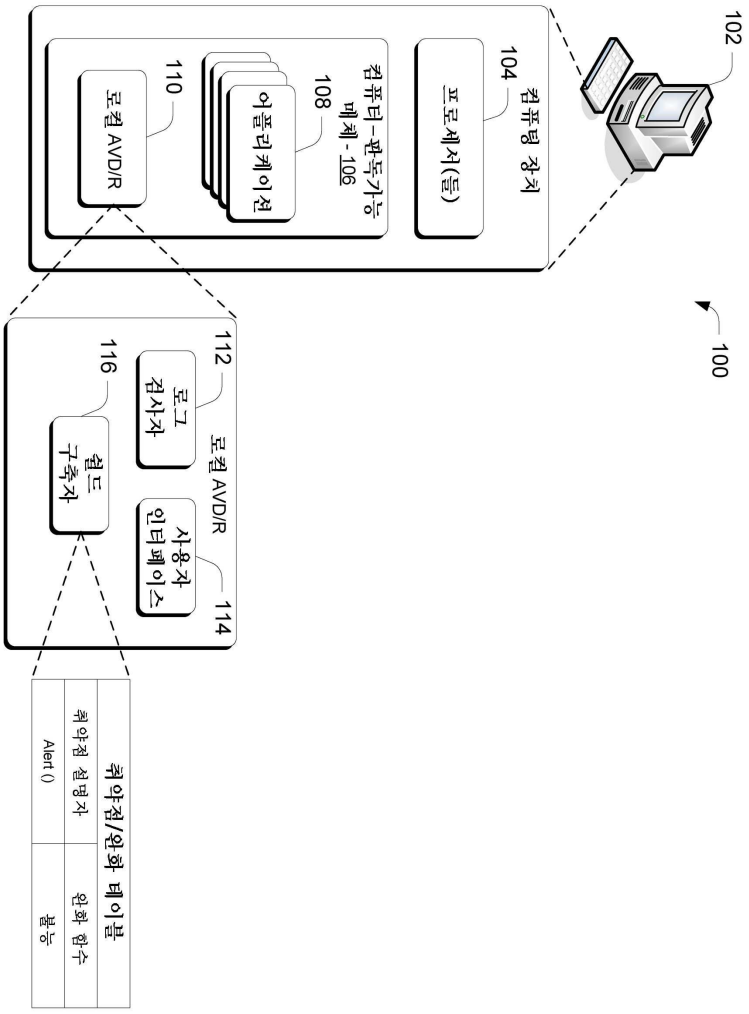
<9> 도 4는 일 실시예에 따른 방법의 단계들을 설명하는 흐름도.

<10> 도 5는 일 실시예에 따른 시스템을 도시.

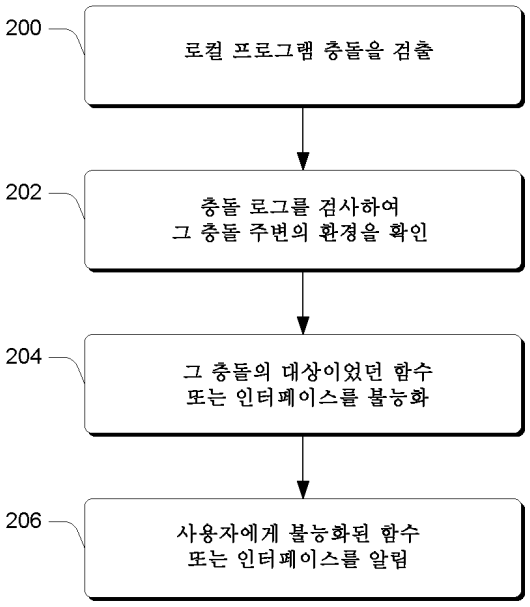
<11> 도 6은 일 실시예에 따른 방법의 단계들을 설명하는 흐름도.

도면

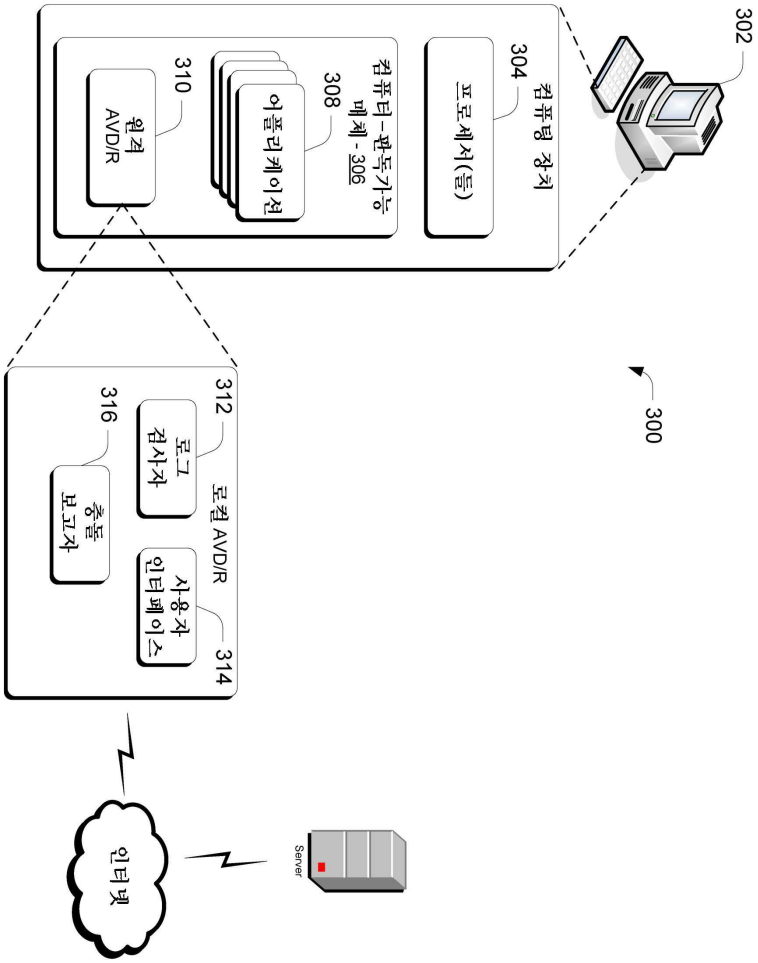
도면1


102
컴퓨팅 장치
104
프로세서(들)
컴퓨터-판독가능 매체 - 106
108
어플리케이션
110
로컬 AVD/R
100
112
로컬 AVD/R
로그 검사자
116
쉘드 구축자
사용자 인터페이스
114
취약점/완화 데이블
취약점 설명자
Alert ()
완화 함수
불능

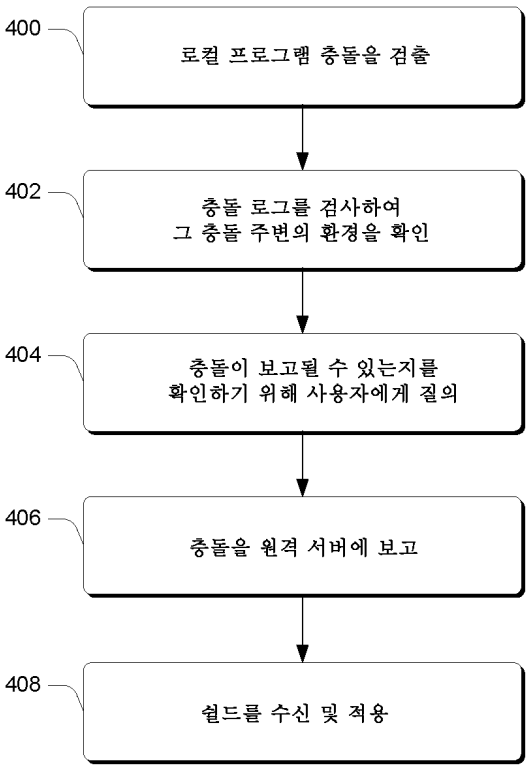
도면2


200
로컬 프로그램 충돌을 검출
202
충돌 로그를 검사하여
그 충돌 주변의 환경을 확인
204
그 충돌의 대상이었던 함수
또는 인터페이스를 불능화
206
사용자에게 불능화된 함수
또는 인터페이스를 알림

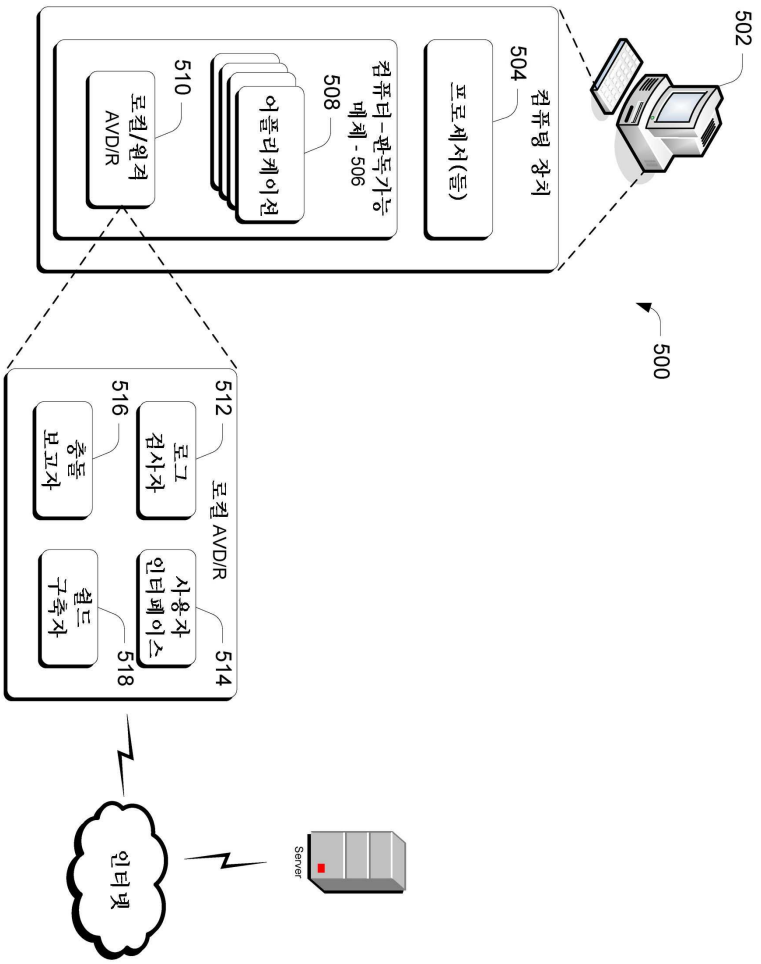
도면3


302
컴퓨팅 장치
304
프로세서(들)
컴퓨터-판독가능 매체 - 306
308
어플리케이션
310
원격 AVD/R
300
312
로컬 AVD/R
로그 검사자
316
충돌 보고자
사용자 인터페이스
314
인터넷
Server

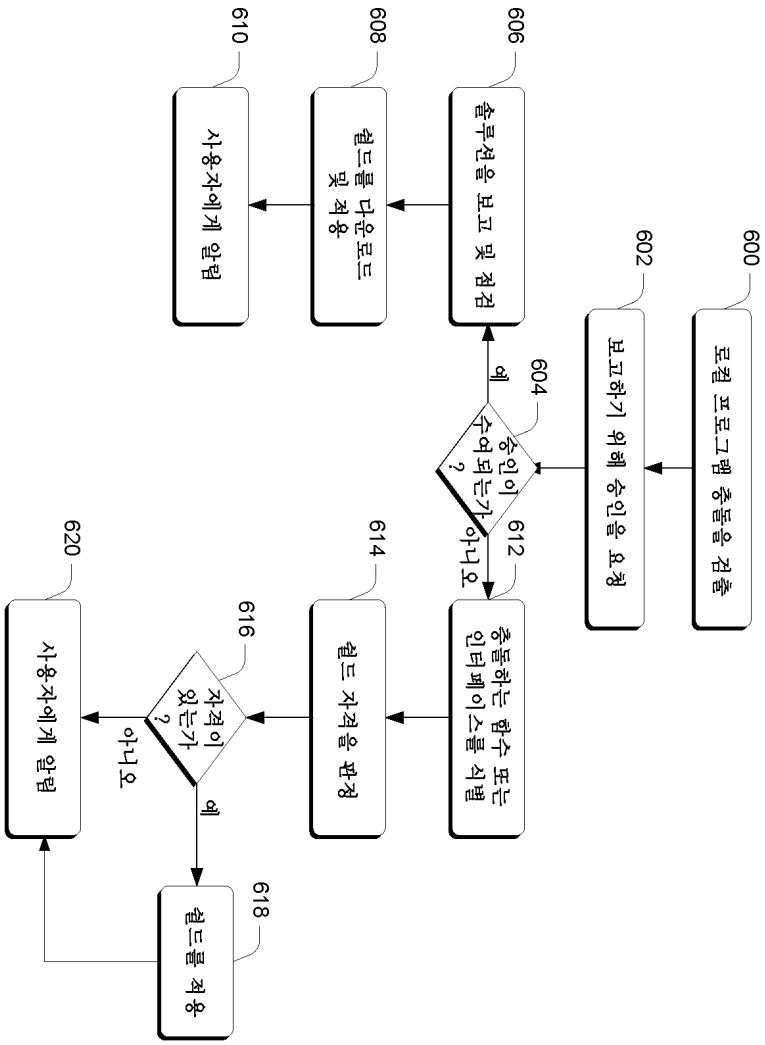
도면4


400
로컬 프로그램 충돌을 검출
402
충돌 로그를 검사하여
그 충돌 주변의 환경을 확인
404
충돌이 보고될 수 있는지를
확인하기 위해 사용자에게 질의
406
충돌을 원격 서버에 보고
408
쉴드를 수신 및 적용

도면5


502
컴퓨팅 장치
504
프로세서(들)
컴퓨터-판독가능 매체 - 506
508
어플리케이션
510
로컬/원격 AVD/R
500
512
로컬 AVD/R
로그 검사자
516
충돌 보고자
사용자 인터페이스
514
쉘드 구축자
518
인터넷
Server

도면6


600
로컬 프로그램 충돌을 검출
602
보고하기 위해 승인을 요청
604
승인이 수여되는가?
예
606
솔루션을 보고 및 점검
608
쉘드를 다운로드 및 적용
610
사용자에게 알림
아니오
612
충돌하는 함수 또는 인터페이스를 식별
614
쉘드 자격을 판정
616
자격이 있는가?
예
618
쉘드를 적용
아니오
620
사용자에게 알림